Garvin Hicking

Serendipity

Individuelle Weblogs für Einsteiger und Profis

Alle in diesem Buch enthaltenen Programme ...

Bibliografische Information Der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Lektorat: Lektorat: Franz Mayer
Satz: Open Source Press (LaTeX)
Umschlaggestaltung: www.fritzdesign.de
Gesamtherstellung: Kösel, Krugzell

ISBN 978-3-937514-54-3 http://www.opensourcepress.de

Vorwort

Als Mitte 2003 mein Mitteilungsbedürfnis so weit überhand nahm, dass ich ein eigenes Blog startete, fiel meine Wahl auf Serendipity, das mir von anderen PHP-Entwicklern bekannt war. Kurz darauf begann ich, den Sourcecode meinen Vorstellungen anzupassen, ohne zu ahnen, dass ich über Jahre hinweg an der Weiterentwicklung dieses Blog-Systems beteiligt sein würde.

Mit diesem Buch komme ich dem Wunsch vieler Serendipity-Benutzer nach, die lange auf eine ausführliche Dokumentation gewartet haben. Viel Herzblut und viel Zeit sind in dieses Buch wie auch das Projekt selbst geflossen, und nach wie vor ist es für mich eine große Freude, jeden Tag mit Serendipity-Benutzern zu kommunizieren und ihnen zu helfen.

Vielen dieser Benutzer bin ich zu Dank verpflichtet: Jannis Hermanns für die Geburtshilfe des Projekts und die Pflege der Server, Robert Lender für seine Arbeit als „Serendipity-Evangelist“, Falk Döring für sein konstantes und hilfreiches Feedback, Kristian Köhntopp fürs Borgen, Judebert, Don Chambers und Matthias Mees für ihre tatkräftige Hilfe im Serendipity-Forum und bei der Gestaltung von Themes, Nadine Oberstein für ihre Begeisterung und moralische Unterstützung, Martin Sallge für die Zerstreuung. Und natürlich auch vielen Dank dem großartigen Redaktionsteam von Open Source Press, ohne dessen Hilfe das vorliegende Werk niemals Gestalt angenommen hätte.

Der größte Dank aber geht an meine Lebensgefährtin Emba, für ihr Verständnis für mein Hobby – aber auch für die nötige, wunderschöne Ablenkung.

Ich wünsche Ihnen viel Spaß bei der Entdeckung von Serendipity und freue mich sehr über Feedback: `garvin@s9y.org`

Garvin Hicking — Köln, April 2008

Inhaltsverzeichnis

Kapitel 1

Einführung

Chats und Foren dominierten noch in den 90er Jahren das interaktive Internet. Diskussionen wurden darüber hinaus allenfalls über Newsgroups und Mailbox-Systeme geführt. Mit der allgemeinen Verfügbarkeit von Internetzugängen wuchs auch das Bedürfnis nach einem System, um sich selbst in diesem neuen Medium zu präsentieren: Das sogenannte *Self-Publishing* war geboren.

Während der Boom der *Wiki-Communities* aus dem vornehmlich akademischen Wunsch nach Wissensaustausch resultierte, entstand auch für das Self-Publishing eine neue Art von Software – *Weblogs*, kurz: *Blogs*. Der Name ist Programm, denn die Grundfunktion ist ein persönliches Tagebuch (ein Logbuch) im Internet. Blogsysteme zeichnet aus, dass sie Inhalte sehr einfach darstellen und fassbar machen, denn schließlich sollen die persönlichen Bemerkungen viele – auch zufällige – Besucher schon auf den ersten Blick ansprechen.

Ganz bewusst setzen sich Blogsysteme von den *Content-Management-Systemen* (CMS) ab, die aus Blogger-Sicht zu teuer, vor allem aber zu komplex sind. Mit steigender Popularität entstanden zahlreiche Blogsysteme im Open-Source- wie im kommerziellen Umfeld: Serendipity, Movable Type, Blogger.com, b2 Evolution, WordPress und andere.

Mittlerweile wird Blog-Software nicht nur für persönliche oder firmenspezifische Tagebücher eingesetzt, sondern ebenso für kleine, stark angepasste und individualisierte Projekt-/Produktseiten, Firmenpräsentationen oder redaktionelle Inhalte jeder Art.

Serendipity ist eines dieser Systeme. Es entstand aus dem Wunsch nach einer freien Software mit größtmöglicher Flexibilität, worauf auch der – für die deutsche Zunge zugegebenermaßen etwas gewöhnungsbedürftige – Name hinweist, der etwa bedeutet: „eine zufällige Entdeckung, die Überraschendes und Großartiges mit sich bringt“. Und da der Name nicht nur schwierig auszusprechen, sondern auch umständlich zu schreiben ist, kürzt man ihn in Fachkreisen einfach ab: *s9y*. Geeks ersetzen längere Wörter gerne durch deren Anfangs- und Endbuchstaben und schreiben die Zahl der ausgelassenen Buchstaben einfach dazwischen …

Serendipity hat den Anspruch, von Anfängern einfach zu bedienen, aber von fortgeschrittenen

Anwendern weitestgehend modifizierbar zu sein. Um auf möglichst vielen verschiedenen Serversystemen zu laufen, sollte es mit mehreren Datenbanksystemen kompatibel sein, die verbreitete Skriptsprache PHP einsetzen und einfach auf dem eigenen Webspace installierbar sein. Die Wahl der Lizenz für das System fiel auf *BSD*, denn diese räumt im Gegensatz zur GNU GPL auch die Möglichkeit ein, das Blogsystem in kommerziellem Umfeld sinnvoll zu nutzen.

So hob Anfang 2002 ein Kernteam bekannter PHP-Entwickler Serendipity aus der Taufe: Jannis Hermanns, Sterling Hughes, George Schlossnagle, Wez Furlong, Joyce Park und Joseph Tate (Letzterer ist auch Autor der Cracklib[1] und von `mod-pubsub`[2]) sowie Sebastian Bergmann vom phpUnit-Projekt.[3] Mit den Jahren entwickelten wechselnde Programmierer das System stetig und konsequent weiter. Seit 2003 ist der Autor dieses Buchs leitender Entwickler.

Dieses Buch bezieht sich auf die aktuelle Version 1.3 und möchte einen umfassenden Überblick über das System und seine Anpassbarkeit vermitteln. Neben der Bedienung der Software werden Sie darum auch lernen, eigene Plugins zu schreiben, Templates anzupassen und Konfigurationsänderungen vorzunehmen.

1.1 Was unterscheidet ein Blog von einem CMS?

Ein Blogsystem zielt darauf ab, einzelne Artikel möglichst einfach zu erfassen und in einem festen Rahmendesign darzustellen. Ein CMS hingegen verwaltet ganz unterschiedliche Inhalte in individuellen Designs und baut komplexe Navigationsstrukturen auf.

Für Blogsysteme sind alle Artikel grundsätzlich gleichwertig und werden nur chronologisch sortiert dargestellt. Ein CMS sollte Seiten unterschiedlich gewichten und diese verschlagworten und kategorisieren können. Darum bietet es sich für umfangreiche Webseiten an, die nicht nur eine „Seitenansammlung“ darstellen, sondern eine komplexe Hierarchie abbilden, was den Umgang mit sog. „Teasern“, Übersichten, Unter- und Parallelseiten erfordert.

Dies ist natürlich eine grobe Vereinfachung, da viele Blogsysteme inzwischen auch solch komplexe Anwendungen meistern. Für Serendipity gibt es beispielsweise das Plugin *Statische Seiten*, mit dem man Inhalte losgelöst von den chronologisch sortierten Artikeln erstellen und auch Hierarchien mit Unterseiten abbilden kann (siehe Seite 410).

Ein weiterer fundamentaler Unterschied ist die Ausrichtung eines CMS auf Mehrbenutzerfähigkeit und Workflows, so dass ein Artikel vor seiner Veröffentlichung verschiedene Arbeitsphasen möglicherweise verschiedener Redakteure durchläuft. Gerade für umfangreiche Webseiten ist es wichtig, dass mehrere Artikelversionen nebeneinander existieren können: Während man bereits an neuen Fassungen einer Seite arbeitet, soll die Online-Version unverändert für Besucher zur Verfügung stehen. Solch ausgefeilte Mechanismen sind im Blog-Umfeld meist

[1] `http://www.cracklib.net/`
[2] `http://sourceforge.net/projects/mod-pubsub`
[3] Die PHP-Testsuite ist unter `http://www.phpunit.de/` erhältlich.

nur unnötige Last.

Somit lässt sich ein Blog grundsätzlich als funktionsreduziertes CMS beschreiben. Das mag negativ klingen, hat aber tatsächlich einen großen Vorteil: Ein Blog geht zielgerichtet auf Bedürfnisse des Web-Publishings ein. Die Bearbeitungsprozesse sind einfacher, und Blog-spezifische Techniken (RSS-Feeds, XML-RPC, Widgets, Web-Services, interaktive Funktionalitäten wie Kommentare und Trackbacks[4]) lassen sich meist mit wesentlich geringerem Aufwand einbinden als in umfangreichen Content-Management-Systemen.

1.2 Warum Serendipity?

Serendipity versteht sich als erweiterbares Blog-Framework, dessen Kernarchitektur so offen konzipiert ist, dass man individuelle Anpassungen vornehmen kann, ohne an den Sourcecode selbst Hand anzulegen. Änderungen am Aussehen des Blogs lassen sich durch Templates vornehmen, Funktionalitäten durch Seitenleisten- und Ereignis-Plugins nachrüsten.

Diese Modularität macht es möglich, eigenen Code vor Updates der Serendipity-Version zu schützen. Seit Erscheinen der ersten Version legen die Entwickler sehr viel Wert auf Abwärtskompatibilität. Im Gegensatz zu anderen Systemen ließen sich bestehende Plugins bisher selbst bei großen Versionssprüngen weiterverwenden. Dies spricht zum einen für eine gesunde Weiterentwicklung des Systems, zum anderen auch für die Plugin-Architektur, die sich als sehr stabil erwiesen hat.

Gerade die Auslagerung von Code macht Serendipity auch im kommerziellen Umfeld interessant. Aufgrund der BSD-Lizenz kann man dieses Blogsystem verwenden, ohne hinzugeschriebenen Code veröffentlichen zu müssen.

Der Einsatz dynamischer PHP-Skripte gewährleistet, dass Serendipity im Gegensatz zu mit Perl, Ruby oder Python entwickelten Blogsystemen auf beinahe jedem Webserver lauffähig ist. Zudem macht die Unterstützung mehrerer Datenbank-Management-Systeme die Software portabel, so dass sie mit unterschiedlichen Datenbank-Servern einsatzfähig ist.

Darüber hinaus legen die Entwickler viel Wert auf guten Programmierstil (mit phpDoc-Funktionskommentaren, einheitlichen Einrückungen und einer kleinteiligen Funktions-API) und sicheren Code. Im Laufe der Serendipity-Entwicklung gab es bislang nur wenige Sicherheitslücken, die bei Bekanntwerden innerhalb weniger Stunden offiziell behoben wurden. Auch gegen weniger verbreitete Angriffsmöglichkeiten auf eine Webanwendung wie XSRF[5] stellte Serendipity frühzeitig Schutzmechanismen bereit.

Serendipity-Templates werden unter Verwendung des Templating-Frameworks *Smarty*,[6] einem De-facto-Standard unter PHP, entwickelt. Zahlreiche Entwickler schwören auf dessen einfache Syntax, die es selbst Anfängern ermöglicht, die HTML-Ausgabe anzupassen, ohne

[4] Eine Erklärung dieser Begriffe finden Sie im Kapitel 1.4 auf Seite 36.

[5] XSRF steht für *Cross-Site Request Forgery* und beschreibt eine Angriffsmethode, bei der eine fremde Website mit Privilegien eines authentifizierten Benutzers Aktionen (z. B. das Löschen aller Artikel) durchführt.

[6] `http://smarty.php.net/`

PHP lernen zu müssen.

Das zentrale Plugin- und Template-Archiv namens *Spartacus*[7] bietet derzeit mehr als 160 Plugins und knapp 100 Templates an. Diese können Sie direkt aus der Verwaltungsoberfläche Ihrer eigenen Blog-Installation heraus installieren oder aktualisieren. Die dort aufgeführten Plugins werden großteils offiziell gewartet und entsprechen den vom Serendipity-Projekt vorgegebenen Anforderungen an Programmierstil und Sicherheit. Plugins mit identischer Funktionalität, die den Benutzer vor die Qual der Wahl stellen, findet man hier nicht, und auch auf Kompatibilität zu allen verfügbaren Serendipity-Versionen wird geachtet.

Da Serendipitys Kernentwickler aus Deutschland kommen, haben gerade deutsche Anwender den Vorteil einer stets aktuellen deutschen Übersetzung der Benutzeroberfläche. Zudem sorgen eine engagierte Community und ein großes Supportforum mit bislang über 65.000 Beiträgen dafür, dass Fragen zum System oder Rückfragen selten unbeantwortet bleiben. Serendipity wird sehr nah an den Wünschen der Community entwickelt – das Feature-Diktat von „oben herab“ ist verpönt.

Natürlich kann Serendipity zum jetzigen Zeitpunkt noch nicht alles. Der größte Nachteil besteht wohl darin, dass man auf eine einzelne Blog-Instanz festgelegt ist. Unter-Blogs (thematisch oder auch pro Benutzer) lassen sich zwar hintricksen, aber an ein echtes Multi-Blogsystem wie Movable Type kommt die Software in dieser Hinsicht nicht heran.

Aufgrund der dynamischen Kompilierung jedes Seitenaufrufs verursacht Serendipity auch mehr Prozessorlast als solche Systeme, die mit exportiertem HTML arbeiten. Serendipity kann auf gut eingerichteten Servern zwar mehrere Dutzend Seitenaufrufe parallel verarbeiten, für den Einsatz einer Community mit Tausenden parallelen Seitenaufrufen pro Sekunde ist das System jedoch nicht konzipiert.

Serendipity hebt sich von der Konkurrenz anderer frei oder kommerziell erhältlicher Systeme vor allem in folgenden Punkten ab:

- Die Browser-basierte Installationsoberfläche erlaubt sowohl eine weitgehend automatische Konfiguration, bei der nur essentielle Daten abgefragt werden, als auch die Einflussnahme auf nahezu alle Parameter bei der **Fortgeschrittenen Installation** (Seite 56). Jede installierte Version lässt sich über die integrierte Oberfläche auf die jeweils aktuellste updaten (Seite 459).

- Texte pflegt man auf Wunsch mit übersichtlichen, einfach verständlichen und funktionalen WYSIWYG-Editoren ein (Seite 104).

- Die integrierte Mediendatenbank für Bilder, PDFs, MP3s etc. gestattet Rechtemanagement, Stichwortsuche und automatische Vorschaugrafiken (Seite 124). Über einen gesonderten Dialog (Seite 107) bindet man die Dateien aus der Mediendatenbank mit wählbaren Layout-Optionen einfach in Blog-Artikel ein. Die Inhalte dieser Mediendatenbank lassen sich dynamisch in einer Verzeichnisansicht darstellen, wie Sie es z. B. vom Windows Explorer gewöhnt sind.

[7] `http://spartacus.s9y.org/`

- Serendipitys gruppenbasiertes Rollenkonzept erlaubt das Anlegen unbegrenzt vieler Autorengruppen und das Vergeben individueller Rechte (Seite 182 und 185).

- Kommentare und Kategorien lassen sich verschachteln, Einträge mehreren Kategorien zuordnen. Natürlich implementiert Serendipity alle gängigen Blog-Standards (und ist somit vollständig Buzzword-kompatibel): Trackback, Pingback, XML-RPC, XHTML 1.1, CSS, RSS, Atom . . . (siehe Seite 36).

- Eine flexible Plugin-API erlaubt es Ihnen, Seitenleisten- und Ereignis-Plugins einzusetzen oder zu entwickeln, die das System ohne Eingriffe in den Serendipity-Code beliebig erweitern (Seite 642).

- Das bereits erwähnte Online-Plugin-Archiv (Spartacus) erlaubt die Ein-Klick-Installation von mehr als 160 Plugins (Seite 265).

- Dank Plugin-Verwaltung per Drag&Drop können Sie das Aussehen und die Inhalte der Seitenleisten des Blogs nach Ihren Wünschen gestalten, ohne Templatedateien manuell ändern zu müssen (Seite 145).

- Das dynamische, Smarty-basierte Template-Konzept erlaubt die strukturelle Änderung sämtlicher Frontend-Elemente über aufeinander aufbauende Templatedateien. Smarty-Templates dürfen ohne Einschränkung Kontrollstrukturen wie Schleifen, Abfragen und Variablen enthalten und optional mit zusätzlichem PHP-Code versehen werden (Seite 480).

- Einfache Integration in bestehende Webseiten. Das Konzept der *Shared Installation* ermöglicht den Betrieb beliebig vieler Blog-Instanzen mit nur einer Code-Basis (Seite 673).

- Serendipity kämpft an Ihrer Seite mit umfangreichen, konfigurierbaren Anti-Spam-Maßnahmen gegen unerwünschte Kommentare oder Trackbacks. Als Schlagworte seien Captcha, automatische Moderation, Akismet und Blacklists genannt (Seite 239).

- Beim Datenbanksystem haben Sie die Wahl zwischen MySQL(i), PostgreSQL und SQLite.

- Serendipity ist nicht nur Open Source, sondern unterliegt sogar der BSD-Lizenz. Somit kann das System auch in kommerziellen Programmen seinen Einsatz finden.

- Ob WordPress, MoveableType, b2Evo, blogger oder andere – Serendipity importiert auf Wunsch den Inhalt zahlreicher anderer Blogsysteme (Seite 192).

1.3 Voraussetzungen

Serendipity als Software wird nicht auf Ihrem persönlichen Computer ausgeführt, sondern läuft im Internet auf einem Webserver. Einen solchen mietet man meistens bei Providern wie Manitu[8], 1&1[9], Strato[10], tiggerswelt[11] oder anderen[12]; die Dienstleistung, die diese erbringen, nennt man *Webhosting*.

Für einen gewissen Monatsbeitrag erhalten Sie Zugangsdaten zu einem Rechner, auf den Sie HTML-Dateien und Software wie Serendipity mittels eines *FTP-Programms* hochladen können. So gespeicherte Software können Sie auf dem Webserver als Anwendungen ausführen; darauf greifen Sie mittels eines Webbrowsers von Ihrem PC aus zu – ganz so, wie Sie es von Webseiten wie eBay oder Amazon kennen. Manche Provider (z. B. `http://www.domainfactory.`
bieten Serendipity für ihre Kunden sogar vorinstalliert an.

Sollten Sie über keinen eigenen Webspace verfügen, gibt es noch eine andere Möglichkeit, Serendipity zu nutzen. Einige kostenlose oder kostengünstige Provider haben die Software auf ihren Servern vorinstalliert und gewähren externen Nutzern Zugriff darauf. Hier können Sie meist nur aus einer Liste vorinstallierter Plugins wählen und auch keine Änderungen an Templates vornehmen oder eigene Plugins einspielen. Um das System erst einmal nur zu testen oder als einfaches Kommunikationsmittel einzusetzen, ist dies sicher eine gute Alternative. Eine Liste dieser Provider finden Sie auf der Serendipity-Webseite.[13]

Die Dateien, die zu Serendipity gehören, enthalten Quellcode, der in der *Skriptsprache* PHP[14] geschrieben ist. Ein PHP-Interpreter macht diesen bei jedem Aufruf auf dem Webserver maschinenlesbar. Das Ergebnis dieser Aktion liefert der Webserver an den Browser des Zielsystems. Laienhaft gesprochen, ließe sich PHP als Betriebssystem für Serendipity ansehen.

Dies ist wichtig, um die generelle Funktionsweise von Serendipity zu verstehen. Im Gegensatz zu Webbrowsern oder einer Office-Anwendung kann Serendipity selbst keine Echtzeit-Eingaben von Ihnen annehmen. Serendipity läuft komplett auf dem Webserver und wird über Ihren Browser sozusagen ferngesteuert. Dabei sendet Ihr Browser eine Anfrage an den Server, Serendipity verarbeitet diese und schickt ein Resultat. Danach geht der Vorgang wieder von vorne los.

Dies hat den Vorteil, dass auch die Besucher Ihrer Seite stets aktuelle Daten zu sehen bekommen. Da jeder Aufruf dynamisch die aktuellsten Daten darstellt, kann Serendipity beispielsweise vorbereitete Artikel mit Erreichen eines speziellen Datums automatisch einbinden, ohne dass Sie nochmals tätig werden müssen. Das ist ein fundamentaler Unterschied zu einem statischen Blogsystem wie MovableType. Dort speichert der Server (vereinfacht gesagt) nur HTML-Dateien, in die sich dynamische Komponenten wie die neuesten Kommentare oder

[8] `http://www.manitu.de/`
[9] `http://www.einsundeins.de/`
[10] `http://www.strato.de/`
[11] `http://www.tiggerswelt.net/`
[12] `http://www.webhostlist.de/`
[13] `http://www.s9y.org/61.html`
[14] PHP ist die Abkürzung von *PHP Hypertext Preprocessor*, siehe `http://www.php.net/`.

die aktuelle Server-Uhrzeit nicht ohne Weiteres einbinden lassen.

Serendipitys Vorgehensweise hat zahlreiche Vorteile: Ein Besucher kann, abhängig von seinen Zugriffsrechten, Artikel sehen, die andere nicht zu Gesicht bekommen. Ein Besucher kann das Design der Webseite selbständig durch Wahl einer Designvorlage verändern. Die Sprache der Benutzeroberfläche richtet sich auf Wunsch nach seinen Einstellungen usw.

Diese Dynamik hat zwar auch Nachteile (so verlangt sie nach mehr Speicher und macht das System langsamer), ermöglicht Ihnen als Blogbetreiber aber auch interessante Möglichkeiten, die wir im Folgenden detailliert beschreiben werden.

Um Serendipity einzusetzen, benötigen Sie FTP-Zugriff auf einen Webserver, auf dem eine aktuelle PHP-Version (mindestens PHP 4.3.1, PHP5 wird unterstützt) installiert ist *und* auf dem der Webhoster Zugriff auf eine Datenbank (MySQL, PostgreSQL, SQLite) gewährt. Als Webserver-Software eignen sich *Lighthttpd*, *Apache* und *Microsoft IIS*. Das Blogsystem lässt sich somit sowohl auf Windows- als auch auf Unix-Derivaten einsetzen.

Damit Serendipity auf dem eigenen Webserver laufen kann, muss man einige PHP-Einstellungen berücksichtigen. Üblicherweise sind sie passend konfiguriert, so dass Sie als unbedarfter Benutzer direkt zum Kapitel 2.2 ab Seite 52 springen können, wenn Sie Ihren Webserver nicht selbst aufsetzen und dem Techniker bei Ihrem Provider keine Hinweise geben müssen.

1.3.1 Systemseitige Einstellungen

Um herauszufinden, welche PHP-Module auf Ihrem Webserver installiert sind, laden Sie folgende kleine Datei namens `info.php` in das Webdaten-Stammverzeichnis (auch *Document Root* genannt)[15] auf den Server (im Buch `www.example.com` genannt) und rufen diese in einem Browser über `http://www.example.com/info.php` auf:

```
<?php phpinfo(); ?>
```

Der Browser sollte dann eine Übersicht darstellen, aus der Sie Ihre aktuellen Einstellungen herauslesen können. Folgende Module benötigt Serendipity:

- Das PHP-Modul `session` muss installiert und aktiviert sein sowie Session-Cookies zulassen. Dies erreicht man mit der `php.ini`-Einstellung

  ```
  session.use_cookies = On
  ```

  Serendipity verlangt darüber hinaus, dass die PHP-Sessiondateien schreibbar sind. Die `php.ini`-Einstellung `session.save_path` muss demnach auf ein Verzeichnis

[15] Das ist meistens das Stammverzeichnis, in dem Sie bei Benutzung Ihres FTP-Zugangs standardmäßig zu Beginn landen. Je nachdem, wie Ihr Provider den Webserver konfiguriert hat, kann es aber auch sein, dass Sie zuerst in ein Verzeichnis wie `htdocs` oder `httpdocs` wechseln müssen, um in das Stammverzeichnis des Webservers zu gelangen.

zeigen, in dem der Webserver bzw. PHP Schreibrechte hat. Gerade bei Windows-Servern enthält diese Variable häufig ein nicht existierendes oder nicht beschreibbares Verzeichnis.

- Damit Serendipity in anderen Zeichensätzen verfasste Texte nach UTF-8 konvertieren und seine XML-Funktionalitäten ausspielen kann, muss das (standardmäßig vorhandene) PHP-Modul `xml/libxml` installiert und aktiviert sein.

- Für andere Zeichensatzkonvertierungen, die besonders bei Import- und Exportvorgängen (z. B. von RSS-Feeds) nötig werden, kann Serendipity sowohl das PHP-Modul `iconv` als auch `mbstring` verwenden. Beide Module sind in aktuellen PHP5-Versionen standardmäßig aktiviert.

- Um Zeichenketten zu finden und zu bearbeiten, setzt Serendipity häufig *reguläre Ausdrücke* ein. Diese Funktionalität liefert das (standardmäßig installierte) PHP-Modul `pcre`.

- Um die Mediendatenbank vernünftig nutzen zu können, benötigen Sie entweder das PHP-Modul `gd` oder das Paket ImageMagick. Sollten Sie Letzteres benutzen, müssen die Dateien des Pakets für Ihren Web-Benutzer ausführbar sein.

- Damit Serendipity auf fremde Webseiten zugreifen kann, um z. B. Webservices zu integrieren und den Download von Plugins zu ermöglichen, benötigt es das (standardmäßig aktivierte) PHP-Modul `socket`, das Netzwerkfunktionen bereitstellt. Selbst wenn das Modul vorhanden ist, kann es sein, dass Firewall-Einstellungen am Server die Verbindung zu fremden Servern auf HTTP-Port 80 (oder anderen) verbieten. Je nach Einsatzzweck müssen Sie Verbindungen zu den Servern, die Serendipity von sich aus anspricht (z. B. `spartacus.s9y.org` und `netmirror.org` für den Download von Plugins), explizit erlauben. Sollte dies nicht möglich sein, können Sie Serendipity zwar nutzen, müssen aber auf die entsprechende Funktionalität verzichten.

- Damit die Serendipity-Komponenten intern auf `https`-URLs zugreifen können (und beispielsweise Trackbacks zu solchen geschützten URLs senden können), wird das PHP-Modul `openssl` benötigt.

- Soll Serendipity die HTML-Seiten per `gzip`-Kompression bündeln, wird das PHP-Modul `zlib` benötigt.

- Um E-Mails zu versenden (z. B. um auf neue Kommentare hinzuweisen), muss PHP mit der `sendmail`-Option kompiliert worden sein.

- Abgesehen von einem Datenbanksystem muss auf Ihrem Server auch die Client-Schnittstelle für die jeweilige Datenbank (MySQL(i), PostgreSQL, SQLite) als PHP-Modul (`mysql`, `mysqli`, `pgsql`, `sqlite`) eingebunden sein. Standardmäßig wird PHP mit MySQL-Bibliotheken eingerichtet; seit PHP 5 ist auch das Datenbanksystem SQLite üblicherweise aktiviert.

- Um die Geschwindigkeit zu erhöhen, empfiehlt sich der Einsatz von PHP-Bytecode-Caches wie APC (`apc`), ZendCache oder ionCube.

PHP bietet zahlreiche sicherheitsrelevante Parameter, deren aktueller Wert bei der Installation des Blogs angezeigt und auf optimale Einstellung überprüft wird. All diese Einstellungen werden in der `php.ini`-Konfigurationsdatei geändert und bedürfen je nach Serverumgebung eines Neustarts der Webserver-Software. Für Hilfe zur Konfiguration von PHP und des jeweiligen Webservers schlagen Sie bitte in der Anleitung der jeweiligen Software nach, da dies den Rahmen dieses Buchs sprengen würde.

Kommt Apache zum Einsatz, können die Einstellungen auch pro Domain konfiguriert werden. Im jeweiligen `<VirtualHost>`-Abschnitt der Konfigurationsdatei (je nach Apache-Version unterschiedlich, meist `httpd.conf` oder auch in Dateien unterhalb des Apache `conf.d`-Verzeichnisses) muss dazu `php_admin_value` *`option wert`* bzw. `php_value` *`option wert`*
eingetragen werden. Alle per `php_value` einstellbaren Optionen lassen sich auch über eine `.htaccess`-Datei im Stammverzeichnis des Webauftritts verändern. Einstellungen in dieser Datei haben Vorrang vor den globalen Optionen, die in der Webserverkonfiguration oder in `php.ini` definiert wurden. Via `php_admin_value` gesetzte Werte lassen sich von `.htaccess`-Dateien nicht mehr verändern.

Serendipity selbst legt bei der Installation eine solche Datei im Serendipity-Verzeichnis an. Sollte Ihr Webserver deren Auswertung unterbinden, können Sie die von Serendipity erstellte Datei gefahrlos löschen, verlieren dadurch aber die Möglichkeit, *sprechende URLs* zu benutzen sowie den Download einiger interner Dateien zu verhindern. Daher sollten Sie dies nur dann tun, wenn Ihr Provider auch nach Rücksprache die Auswertung der `.htaccess`-Dateien nicht genehmigt.

Haben Sie die Möglichkeit, die `.htaccess`-Nutzung über die Option `AllowOverride` in der Webserver-Konfigurationsdatei zuzulassen, geschieht dies im `<VirtualHost>`-Abschnitt mittels

```
<Directory /pfad/zum/serendipity-verzeichnis>
  AllowOverride All
</Directory>
```

Notfalls reicht auch `AllowOverride Options FileInfo`. Der Wert `All` erlaubt es Plugins grundsätzlich, weitere Webserver-Optionen zu setzen, und gibt Ihnen damit mehr Flexibilität.

Serendipity setzt in seiner `.htaccess`-Datei (siehe Seite 34) die Optionen `session.use_trans_sid` (Seite 31) und `register_globals` (Seite 32). Bei der Benutzung des Apache-Moduls `mod_rewrite` kommen auch die Optionen `RewriteEngine`, `RewriteBase` und `RewriteRule` zum Einsatz. Die `.htaccess`-Optionen `ErrorDocument` und `DirectoryIndex` ermöglichen freie Permalinks (siehe Seite 168). Damit der Webserver bestimmte Serendipity-Dateien nicht im Klartext darstellt, folgt ein `Deny from All` als Sicherheitsmaßnahme.

Sprechende URLs

`mod_rewrite` ist kein PHP-Modul, sondern eine Funktionalität des Apache-Webservers. Wenn das Modul in Apache eingebunden wurde, erlaubt es einer Web-Anwendung, URLs zu benutzen, die nichts mit der tatsächlichen Dateistruktur auf dem Server zu tun haben. Statt die Nutzer damit zu behelligen, dass sie gerade ein PHP-Skript namens `serendipity_admin.php` nutzen, lässt man sie besser auf eine URL wie `http://www.example.com/serendipity/admin/` zugreifen, die den Browser auf die tatsächlich existierende Datei `serendipity_admin.php` umleitet.

Das `mod_rewrite`-Modul ermöglicht es Serendipity, sogenannte *sprechende URLs* (auch *URL-Umformung* genannt) einzusetzen, die zum Beispiel Artikeltitel oder Kategorienamen enthalten. Für den Benutzer hat das den Vorteil, dass er eine URL schon anhand ihres Namens eindeutig zuweisen kann. Zudem können Suchroboter wie Google Ihre Artikel aufgrund der Schlagwörter in der URL besser indizieren.

Serendipity kann auch mit sprechenden URLs arbeiten, wenn `mod_rewrite` nicht verfügbar ist. Dann leitet der Apache-Webserver den Browser mittels eines Tricks an die echte URL weiter: Kann er eine Seite nicht finden, hängt es vom *Error-Handling* (genau: der `ErrorDocument`-Anweisung) ab, ob er die Fehlermeldung an den Browser durchreicht oder diesen auf eine andere Seite führt. Diese Methode ist jedoch einen Tick langsamer und unflexibler als `mod_rewrite` und klappt leider nicht auf allen Servern, da sie die oben erwähnte `AllowOverride`-Einstellung `All` voraussetzt. Andere Webserver wie Microsoft IIS oder lighttpd bieten diese Möglichkeiten leider nicht, daher ist dort keine URL-Umformung möglich[16].

Bei der Installation wird Serendipity versuchen, automatisch die passendste Methode für Sie herauszufinden. Abhängig davon wird die `.htaccess`-Datei unterschiedlich erstellt, damit sie die jeweils erforderlichen Anweisungen enthält. Die gewünschte Methode zur URL-Umformung kann später in der Konfiguration natürlich auch umgestellt werden.

Sobald für Serendipity das `mod_rewrite`-Modul aktiviert ist, wird Serendipity *sämtliche* Aufrufe in seinem Unterverzeichnis umleiten. Sollten Sie später also einmal andere Programme unterhalb des Serendipity-Verzeichnisses aufrufen wollen, wird dies womöglich dazu führen, dass Sie statt des installierten Programms Serendipity sehen. Dies liegt daran, dass durch die `.htaccess`-Datei von Serendipity die URL-Umformungsregeln auch für alle Unterverzeichnisse gelten. Abhilfe können Sie schaffen, indem Sie in jedem Unterverzeichnis einer fremden Anwendung eine eigene `.htaccess`-Datei erstellen, mit dem Inhalt:

```
RewriteEngine off
```

Diese Anweisung sorgt dafür, dass für das jeweilige Programm die URL-Umformung deaktiviert wird.

[16] Mit etwas manuellem Aufwand und Kenntnis der Webserver lassen sich die benötigten Umformungsregeln jedoch auch nachrüsten, mehr dazu unter `http://s9y.org/119.html`

1.3.2 Für Serendipity relevante PHP-Konfigurationsoptionen

Folgende Optionen sind für Serendipity maßgeblich:

`php_value magic_quotes_gpc`

`php_value magic_quotes_runtime` setzbar in `.htaccess`, `php.ini` oder in der Webserverkonfiguration;

Diese Variablen bestimmen, ob PHP Sonderzeichen in URL-Variablen (`GET`, `POST`, `COOKIE`) automatisch in ein sicheres Format umwandelt. Serendipity benötigt diesen Service nicht, da es die entsprechende Umformung selber vornimmt. Steht eine dieser Optionen auf `true`, macht Serendipity die entsprechende Konvertierung rückgängig. Daher empfiehlt es sich aus Performancegründen, die beiden Einstellungen auf `false` zu setzen und damit zu deaktivieren.

`php_value session.use_cookies`
`.htaccess`, `php.ini` oder Webserverkonfiguration;

Serendipity speichert temporäre Dateien in PHP-Sessions. Um eine Session eindeutig zu identifizieren, vergibt das System Session-IDs, die üblicherweise in einem HTTP-Cookie gespeichert werden. Damit Serendipity PHP-Session-Cookies nutzen kann, muss diese Option aktiviert sein.

`php_value session.use_trans_sid`
wird von Serendipity selbst in der `.htaccess`-Datei auf `0` gesetzt, kann aber auch in der `php.ini` oder der Webserverkonfiguration geändert werden;

Wenn der Browser keine Cookies akzeptiert, kann die Session-ID auch an die URL angehängt werden. PHP macht dies automatisch, wenn die Option `session.use_trans_sid` aktiviert ist. Diese Automatik ist seitens Serendipity jedoch aus Sicherheitsgründen unerwünscht. Sollte die Option dennoch aktiviert sein, können fremde Nutzer ungültige Session-IDs in die URL einschleusen und somit möglicherweise Ihre Session-Dateien kompromittieren. Ein Serendipity-Blog sollte daher niemals die Session-ID in einer URL angeben. Gibt Ihnen jemand eine Serendipity-URL wie `http://www.example.com/serendipity/index.php?PHPSESSID=23213123adasd`, kann es sein, dass diese Person Ihre Daten ausspähen will. Die serverseitige Deaktivierung der PHP-Option `session.use_trans_sid` beugt solchem Missbrauch vor.

`php_admin_value allow_url_fopen`
nur in der `php.ini` oder in der Webserverkonfiguration änderbar;

PHP liest und behandelt URLs wie normale Dateien, wenn diese Option aktiviert ist. Gerade in älteren PHP-Versionen öffnet die Aktivierung dieser Option die Tür für Angriffsszenarien, in denen Code von fremden Internetseiten bei Ihnen ausgeführt werden kann (und so das System kompromittiert).

Damit die Aktivierung dieser sicherheitsrelevanten Option nicht erforderlich ist, greifen alle wichtigen Serendipity-Funktionen stattdessen über *Sockets* auf Netzwerk-Ressourcen zu. Nur veraltete Plugins erfordern möglicherweise die Aktivierung dieser Option. Es empfiehlt sich, sie zu deaktivieren und nur bei Bedarf zuzuschalten.

`php_value register_globals`
: wird von Serendipity selbst in der `.htaccess`-Datei auf `off` gesetzt, kann aber auch in `php.ini` oder der Webserverkonfiguration festgelegt werden;

  Wenn diese Variable aktiviert ist, stellt PHP URL-Variablen im globalen Namensraum zur Verfügung. Ältere PHP-Programme benötigen diese Option, Serendipity greift jedoch nicht auf dieses Feature zurück. Aus Sicherheitsgründen ist es daher sehr zu empfehlen, die Option zu deaktivieren, sofern sie nicht für andere auf dem Webserver laufende Anwendungen benötigt wird.

`php_admin_value safe_mode`

`php_admin_value safe_mode_include_dir`

`php_admin_value safe_mode_gid` lediglich via `php.ini` oder Webserverkonfiguration änderbar;

: PHPs *Safe Mode* stellt sicher, dass lokale PHP-Anwendungen wie Serendipity nur auf die Daten im eigenen Stammverzeichnis zugreifen und keine fremden Dateien einbinden können. Obwohl Serendipity mit dieser Einschränkung generell funktioniert, hat die Aktivierung dieses Modus folgende Nachteile: Die Optionen `safe_mode_include_dir` und `safe_mode_gid` legen fest, auf welche Verzeichnisse ein PHP-Skript Zugriff hat. Solange die Benutzerrechte (*Permissions*) und die Eigentumsverhältnisse (*Ownership*) auf dem Webserver für das Serendipity-Installationsverzeichnis korrekt gesetzt sind, lassen sich z. B. Mediendatenbank-Dateien problemlos hochladen. Ein falsch eingerichteter Safe Mode hat zur Folge, dass man mittels FTP oder Administrationsoberfläche keine Templates und andere Dateien hochladen und keine Bilder im Nachhinein löschen oder bearbeiten kann.

`php_admin_value open_basedir`
: nur via `php.ini` oder in der Webserverkonfiguration änderbar;

  Ähnlich wie der Safe Mode lässt sich der Datei-Zugriffsschutz von PHP auf gewisse Verzeichnisse beschränken. Üblicherweise wird diese Option in Zusammenhang mit den Safe-Mode-Parametern konfiguriert. Während der Safe Mode lediglich den globalen Dateizugriff auf Gruppen- und Benutzerebene regelt, schränkt `open_basedir` den Zugriff auf vordefinierte Verzeichnisse ein.

  Verwendet Ihr Webserver diese Option, gehört das Serendipity-Installationsverzeichnis unbedingt in die damit konfigurierte Verzeichnisliste. Anderenfalls kann der Webserver die Installationsdateien nicht aufrufen. Als Trennzeichen zwischen zwei Verzeichnispfaden dient in dieser Option das Semikolon (`;`).

Bleibt diese Option leer, ist der Zugriff auf alle Verzeichnisse möglich. Befindet sich bereits der Verzeichniseintrag `.` in der Auflistung, steht dieser Punkt synonym für das Verzeichnis, in dem ein PHP-Programm ausgeführt wird.

Gerade auf Servern, die viel mit symbolischen Dateisystemlinks arbeiten, muss der Administrator sorgfältig auf die korrekte Konfiguration der involvierten Verzeichnispfade achten.

`php_value session.cookie_domain`
: `.htaccess`, `php.ini` oder Webserverkonfiguration;

Wenn Serendipity einen HTTP-Cookie setzt (also z. B. die Session-ID oder Login-Daten), enthalten diese den Namen der Domain, auf dem sich die Serendipity-Installation befindet. Falls in der PHP-Variable `session.cookie_domain` ein Domainname eingetragen ist, nutzt Serendipity stattdessen diese Domainangabe. In dem Fall stellen Sie sicher, dass sich dieser Wert mit dem Domainnamen des s9y-Servers deckt. Wenn ein Webserver für mehrere Domains konfiguriert ist und an dieser Stelle einen zentralen Domainnamen setzt, ist dies oft Ursache für spätere Probleme.

`php_value session.save_path`
: `.htaccess`, `php.ini` oder Webserverkonfiguration;

Alle von Serendipity gespeicherten Sessiondaten werden im mit dieser Option angegebenen Verzeichnis gespeichert. Der ausführende PHP-Prozess muss darauf Schreibrechte besitzen. Stellen Sie bei Windows-Servern sicher, dass hier ein gültiger Verzeichnisname (z. B. `C:/Temp`) eingetragen ist. Sollten PHP-Sessions nicht schreibbar sein, zeigt Serendipity Fehlermeldungen an.

`php_admin_value file_uploads`

`php_value post_max_size`

`php_value upload_max_filesize`

`php_value max_input_time` `.htaccess`, `php.ini` oder Webserverkonfiguration – `file_uploads` nur via `php.ini` oder Webserverkonfiguration änderbar;

Damit man überhaupt Dateien hochladen kann, muss `file_uploads` aktiv sein. Die maximale Dateigröße einer hochgeladenen Datei legen `post_max_size`, das ein Limit für HTTP-Requests mitsamt aller Daten definiert, und `upload_max_filesize` fest, das seinerseits eine spezielle Obergrenze nur für die mittels HTTP-Request übermittelten Dateien setzt. Die maximale Zeit für die Verarbeitung einer Datei beim Upload legt die Variable `max_input_time` fest.

`php_admin_value memory_limit`
: Einstellung via `php.ini` oder Webserverkonfiguration;

Serendipity benötigt einiges an Arbeitsspeicher auf dem Webserver. In einer Grundkonfiguration reichen meist zwischen drei und vier MB RAM. Der Speicherbedarf kann mit der Anzahl und der Komplexität von Plugins jedoch zunehmen.

Wie viel verfügbaren Speicher PHP-Anwendungen insgesamt nutzen dürfen, legt die Variable `memory_limit` fest. Sollte der Arbeitsspeicher Fehlermeldungen zufolge nicht mehr ausreichen, müssen Sie entweder dieses Limit hochsetzen oder Serendipity-Plugins löschen.

`php_value display_errors`

`php_value error_log .htaccess`, `php.ini` oder `httpd.conf`;

Einige Server geben PHP-Fehlermeldungen nicht an den Browser weiter, um weniger leicht angreifbar zu sein.[17] Wenn Sie allerdings einmal auf Probleme stoßen oder eigene Anpassungen eine leere Seite zum Ergebnis haben, empfiehlt es sich ggf., diese beiden Optionen temporär zu aktivieren. Die bessere Alternative heißt: häufiger die Fehler-Logfiles des Webservers prüfen.

1.3.3 Serendipitys .htaccess-Datei

Befehle, die Serendipity eigenständig in die `.htaccess`-Datei im Stammverzeichnis einträgt, fasst die Software in einen Block ein, der mit `#Begin s9y` beginnt und mit `#End s9y` endet. Wenn Sie eigene Parameter in diese Datei einfügen, tun Sie dies daher vor `#Begin s9y` oder nach `#End s9y`. Das verhindert, dass Serendipity diese Zeilen ändert. Wenn Sie eine von Serendipity gesetzte Option überschreiben müssen, fügen Sie die Option mit dem von Ihnen gewünschten Wert am besten nach dem `s9y`-Block nochmals ein.

Auf einem Server ohne `mod_rewrite` sieht der `s9y`-Block wie folgt aus:

```
# BEGIN s9y
DirectoryIndex /serendipity/index.php
php_value session.use_trans_sid 0
php_value register_globals off

<Files *.tpl.php>
    deny from all
</Files>

<Files *.tpl>
    deny from all
</Files>
```

[17] Fehlermeldungen können kritische Dateipfade eines Servers offenlegen (*Information Disclosure*). Solche Informationen erlauben es Angreifern, bestimmte Rückschlüsse auf die Einrichtung eines Server zu ziehen.

```
<Files *.sql>
    deny from all
</Files>

<Files *.inc.php>
    deny from all
</Files>

<Files *.db>
    deny from all
</Files>

# END s9y
```

Der erste Parameter, `DirectoryIndex`, legt die zentrale Serendipity-Datei `index.php` als Startseite des Blogs fest. Wenn in der Serendipity-Konfiguration bei der **URL-Umformung** (siehe Seite 168) die Option **Apache ErrorHandling** eingestellt wurde, folgt als nächste Zeile der Eintrag `ErrorDocument 404 /serendipity/index.php`. Die darin genannte Datei dient als Weiterleitungsziel für alle virtuellen URLs, auch bei der Verwendung von `mod_rewrite`.

Die nächsten beiden Zeilen deaktivieren die PHP-Einstellungen `session.use_trans_sid` und `register_globals`.

Zuletzt folgen mehrere Abschnitte, die den Zugriff auf interne Serendipity-Dateien (`*.tpl.php`, `*.tpl`, `*.sql`, `*.inc.php` und `*.db`) verbieten.

Wurde `mod_rewrite` aktiviert, sieht die `.htaccess` bis auf einen Block genauso aus wie die eben gezeigte:

```
# BEGIN s9y
ErrorDocument 404 /serendipity/index.php
DirectoryIndex /serendipity/index.php
php_value session.use_trans_sid 0
php_value register_globals off

RewriteEngine On
RewriteBase /serendipity/
RewriteRule ^((id/([0-9]+))/?) index.php?/$1 [NC,L,QSA]
RewriteRule ^(authors/([0-9]+)-[0-9a-z\.\_!;,\+\-\%]+)
  index.php?/$1 [NC,L,QSA]
[...]
RewriteRule ^htmlarea/(.*) htmlarea/$1 [L,QSA]
RewriteRule (.*.html?) index.php?url=/$1 [L,QSA]

<Files *.tpl.php>
    deny from all
```

```
</Files>
[...]

# END s9y
```

Hinzugekommen ist der Befehl `RewriteEngine On`, der `mod_rewrite` aktiviert. `RewriteBase` legt den Stammpfad für alle Operationen fest. Die einzelnen `RewriteRule`s legen fest, wohin der Nutzer weitergeleitet wird, wenn er spezielle URLs aufruft. Der erste Teil hinter einer `RewriteRule` gibt an, welches Muster in der URL gesucht wird, der zweite Teil legt fest, welche Datei aufgerufen wird, und zuletzt folgen spezielle `mod_rewrite`-Optionen in eckigen Klammern.

`L` gibt eine abschließende Regel an: Sobald eine URL auf das jeweilige Muster zutrifft, werden die übrigen Muster nicht mehr geprüft. `QSA` bedeutet, dass URL-Parameter (die sogenannten *GET-Variablen*) an die Ziel-URL angehängt werden. `NC` besagt, dass Groß- und Kleinschreibung bei einer URL nicht unterschiedlich behandelt wird.

Die jeweiligen Regeln werden mittels regulärer Ausdrücke formuliert, die bestimmte Muster in Zeichenketten detailliert beschreiben.

1.3.4 PEAR

Das *PHP Extension and Application Repository* (kurz PEAR) ist eine Sammlung von PHP-Skripten und -Modulen, von der zahlreiche Web-Anwendungen Gebrauch machen. Auch Serendipity setzt mehrere PEAR-Komponenten ein: `Cache::Lite`, `HTTP::Request`, `Onyx`, `Net::Socket`, `Text::Wiki` und `XML::RPC`. Damit diese nicht zentral auf dem Webserver installiert sein müssen, liefert Serendipity die Komponenten im Unterverzeichnis `bundled-libs` mit.

Wenn die entsprechenden PEAR-Komponenten bereits auf dem Webserver vorhanden sind, räumt Serendipity diesen standardmäßig den Vorrang ein, damit Systemadministratoren bei kritischen Updates nicht auch das `bundled-libs`-Verzeichnis anfassen müssen. Dabei kommt es manchmal zu (Versions-)Konflikten.

In diesem Fall kann man in der globalen Serendipity-Konfigurationsdatei `serendipity_config.inc.php` mit

```
$serendipity['use_PEAR'] = false;
```

darauf beharren, dass die von s9y mitgelieferten Komponenten zum Einsatz kommen.

1.4 Terminologie

In Zeiten des *Web 2.0* (siehe Seite 38) buhlen zahlreiche allgegenwärtige Marketing- und Technologiebegriffe um Aufmerksamkeit. Da diese auch im Zusammenhang mit Serendipity

wichtig sind, gehen wir darauf im Folgenden gezielt und knapp ein. Detaillierte Definitionen liefert z. B. die deutsche Wikipedia.[18]

1.4.1 XHTML, XML und CSS

HTML stellte ursprünglich eine einfache Syntax zur Formatierung von Dokumenten im Web bereit. Während es anfangs nur sehr primitive Auszeichnungselemente gab (Überschriften, Fettungen, Tabellen), sorgten viele Entwickler-Gremien schnell und leider auch unübersichtlich für Erweiterungen. Parallel zu HTML wurde XML (*Extensible Markup Language*) als universelles Dateiformat entwickelt, um beliebige maschinen- und menschenlesbare Inhalte darzustellen. Im Gegensatz zu Binärformaten lassen sich für XML plattformübergreifend Schnittstellen bereitstellen und standardisiert eigene Unterformate für alle erdenklichen Einsatzzwecke erfinden.

Auch HTML wurde mittels XHTML-Standard mit der XML-Syntax aufgefrischt und konsolidiert. Strenge Regeln sorgen nun dafür, dass sich HTML-Dokumente von vielerlei Programmen interpretieren und „validieren“ (also auf syntaktische Korrektheit prüfen) lassen. In der Theorie ermöglicht dies weitgehende Abstraktion von Layout und Inhalt. Der besseren Interpretation durch Software wegen konnte das CSS-Format (*Cascading Style Sheets*) seinen Siegeszug antreten. CSS-Dateien lassen sich auf Elemente in XHTML-Dokumenten anwenden und legen so die Formatierung der strukturellen Elemente eines Dokuments fest.

Auch wenn viele Webseiten heute noch nicht mit gültigem XHTML entwickelt werden und es Bemühungen gibt, die „dumme Einfachheit“ von HTML mit laxeren Prüfungen wieder aufleben zu lassen, ist ein Vorteil von XHTML unumstritten: Weiterverwendbarkeit.

1.4.2 Browser

Ein Browser ist eine Software auf Ihrem Computer, mit der Sie Internetadressen aufrufen und ansehen können. Bekannte Browser sind der *Microsoft Internet Explorer*, *Mozilla Firefox* oder auch *Apple Safari*. Der Begriff „Browser“ steht dabei für das „Stöbern“ (neudeutsch „Surfen“) im Internet, denn erst durch diese Software wird aus einem speziell formatierten HTML-Dokument etwas, das ein Besucher am Bildschirm wirklich ansehen und benutzen kann.

Die am Markt verfügbaren Browser unterscheiden sich in Merkmalen wie Sicherheit, Geschwindigkeit, Funktionsumfang und auch Darstellungsqualität. Obwohl HTML und CSS festgelegte Standards sind, weicht jeder Browser bei deren Interpretation doch geringfügig ab, daher stellen verschiedene Browser ein und dieselbe Webseite möglicherweise unterschiedlich dar.

[18] `http://de.wikipedia.org/`

1.4.3 Client, Server

Bei der Kommunikation im Internet sind Millionen von Computern miteinander vernetzt. Beim Abrufen von Daten aus dem Internet über Ihren Browser verbindet sich Ihr Computer mit einem fremden Computer, auf dem diese Daten liegen.

Ihr Computer stellt dabei einen *Client* dar, also den *Empfänger*. Der Rechner, der Daten ausliefert, dient dabei als *Server*, also als *Absender*. Bei Nutzung fast jedes Internet-Protokolls spielen für die Kommunikation zwei Datenkategorien eine Rolle: das, was ein Client sendet und empfängt, und das, was ein Server sendet, empfängt und verarbeitet.

1.4.4 RSS, Feeds, Syndication und Aggregation

RSS ist die Abkürzung für *Rich Site Summary* oder auch *Really Simple Syndication* und der Name eines Datenformats, das gemeinsam mit den Blogs die Web-Welt verändert hat.

Eine Datei im RSS-Format ist grundsätzlich erst einmal nur eine XML-Datei, die einen speziellen Satz von Elementen enthält. Diese legen die Eigenschaften von Blog-Artikeln, wie Titel, Inhalt oder Hyperlinks, fest.

Eine RSS-Datei stellt eine Art Newsticker dar, der meistens die chronologisch neuesten Änderungen an News-Artikeln oder Blog-Einträgen aufführt. Solche Dateien werden kontinuierlich (meist automatisch) aktualisiert und auch als *Feed* bezeichnet. Dafür ausgelegte Software kann eine solche maschinenlesbare RSS-Datei weiterverarbeiten und mit dem Inhalt anderer Webseiten zusammenführen (hier seien die Schlagworte *Aggregation* und *Syndication* genannt).

RSS-Reader-Programme ermöglichen es Internet-Nutzern, die RSS-Dateien von Webseiten ähnlich wie mit einem E-Mail-Programm regelmäßig zu überprüfen, und informieren so leicht und übersichtlich über Änderungen.

Manche RSS-Dateien liefern den Inhalt eines neuen Artikels mit, so dass der Nutzer ihn sich unabhängig von der Webseite ansehen kann. Diese Einbindungsweise fokussiert auf den eigentlichen Inhalt von Webseiten, da sie es erlaubt, Inhalte beinahe darstellungsneutral zu beziehen.

Unterschiedliche Gremien haben zahlreiche RSS-Versionen erarbeitet, teilweise inkompatibel zueinander und mit jeweils anderen Attributen. Das verbreitetste RSS-Format ist Version 2.0. Serendipity unterstützt auch RSS 0.9, 0.91 und 1.0.

1.4.5 Webservices, Web 2.0, Social Web

In dem Maße, wie Webserver in Skriptsprachen wie PHP und Perl geschriebene Anwendungen zu interpretieren lernten, wurde es immer einfacher und interessanter, verschiedene Komponenten miteinander zu verbinden. Dynamische Software eröffnete ganz neue Möglichkeiten, Besuchern einer Webseite Zusatzdienste, sogenannte Webservices, zu präsentieren. Als

Beispiel seien die Anzeige von neuen Terminen, die Darstellung von Wetterdaten, spezielle passwortgeschützte Bereiche oder auch die Darstellung von Bookmarks genannt.

Auf zahlreichen Webseiten kann man oft kostenlos Community-Dienste in Anspruch nehmen, die sich mittels Webservices an die eigene Webseite anbinden lassen – sei es die Liste aktueller Fotos von der eigenen Flickr-Fotoseite, die Darstellung der Lieblingslinks, des aktuellen Wetters zuhause oder auch die Einbindung fremder Werbung. Derartige Services bezeichnet man auch als *Social Web*, da ein Gedankenaustausch in einer gemeinsamen Community stattfindet.

Der Begriff *Web 2.0* steht gesamtheitlich für die Wandlung des ehemals statischen *Web 1.0* hin zu einer Web-Welt mit verflochtenen Diensten und großen Interaktionsmöglichkeiten unter Einbeziehung der Web-Nutzer.

1.4.6 XML-RPC, SOAP, REST

Damit ein Webservice angesprochen werden kann, benötigt man eine gemeinsame Schnittstelle zwischen Empfänger (Client) und Absender (Server). Eine solche lässt sich mittels verschiedener Techniken anbieten.

Die im PHP-Umfeld verbreitetste Schnittstelle nennt sich XML-RPC. RPC steht hierbei für *Remote Procedure Call*, also der Aufruf einer Funktion durch ein fremdes Programm. Zur Datenübergabe und -annahme wird hierbei XML verwendet. Der Server erhält eine XML-Datei mit speziellen Anforderungen. Die ausführende Server-Software interpretiert diese, wertet sie aus und schickt das Ergebnis wieder im XML-Format zurück an den Client.

Eine dazu konkurrierende Schnittstelle nennt sich SOAP (*Simple Object Access Protocol*). Auch SOAP arbeitet mit XML-Datensätzen, hat aber eine wesentlich komplexere Datenarchitektur und wird aufgrund dieses Overheads im Web eher selten eingesetzt.

Als dritte bekannte Schnittstelle bietet sich REST an: *Representational State Transfer*. Diese simpelste der Schnittstellen führt Abfragen anhand einfacher URL-Parameter durch und gibt üblicherweise auch einfach zu interpretierende XML-Daten zurück.

XML-RPC ist für die Weblog-Welt insofern interessant, als sie es erlaubt, Inhalte neutral von der verwendeten Blogsoftware zu erfassen. Die meisten Blogsysteme binden eine gemeinsame XML-RPC-Schnittstelle ein, die das Erstellen und Bearbeiten von Artikeln mittels „Fernsteuerung“ ermöglicht. Dadurch kann ein Benutzer seine Artikel mit ganz normaler Desktop-Software wie z. B. ecto, marsEdit, WindowsLive Writer oder sogar Microsoft Word schreiben, anstatt sie in der Web-Oberfläche seiner Blogsoftware zu verfassen. Gerade für Benutzer, die sich nicht mit neuer Schreib-Software auseinandersetzen wollen, ist diese Variante sehr interessant.

1.4.7 Atom

Um die verschiedenen Varianten von RSS wieder zueinanderzuführen, entwickelte das W3C-Gremium einen Standard namens Atom. In einem vollständig gültigen XML-Format deckt es alle Fähigkeiten von RSS ab, wird aber aufgrund der höheren Komplexität heute immer noch als Konkurrenz zu RSS 2.0 angesehen.

Der Begriff Atom steht nicht nur für ein Dateiformat, sondern auch für eine Implementation der XML-RPC- oder SOAP-Protokollschnittstelle, die es erlaubt, entsprechend ausgerüstete Weblog-Software fernzusteuern.

1.4.8 Trackback und Pingback

Da Blogs letztlich Sprachrohre von Individuen darstellen, ist es für einen Blog-Betreiber meist sehr wichtig, sein Blog mit anderen zu vernetzen. Ganz im Sinne des interaktiven Webs bestand ein wichtiger Gedanke bei der Erfindung des Blogs darin, Beiträge eines Autoren auf dessen Seite kommentieren zu können und so in Dialog zu treten.

Leser von Blogs sind häufig selber Blogbetreiber und möchten – gerade bei abweichender Meinung – eine Diskussion gern auf der eigenen Seite weiterführen, ohne dabei die Verbindung mit dem Initiator zu verlieren. Hierfür wurde die Technik der *Trackbacks* erfunden, die es einem Autor erlaubt, seinen Artikel mit einem anderen zu verketten. Die Besonderheit dabei: Die fremde Webseite wird davon automatisch in Kenntnis gesetzt. Somit befindet sich nicht nur auf der eigenen Seite ein Link auf den Quellartikel, sondern es gibt auch einen, der vom Quellartikel auf die eigene Seite zeigt. Wie das bei Serendipity funktioniert, behandelt Kapitel 8.1 ab Seite 433.

Ein *Pingback* ist eine abgeschwächte Form des Trackbacks. Der Initiator kann auf diese Weise den Ursprungsautor darauf hinweisen, dass er sich auf ihn bezogen hat. Dabei wird auf dessen Blog kein (ausführlicher) Verweis für andere Besucher erzeugt.

1.4.9 API und Plugin

Eine API (*Application Programming Interface*) stellt ein Software-Regelwerk (meist eine konkrete Liste an Funktionen oder Klassen) zur Verfügung, an das man sich als Programmierer halten kann, um auf gewisse Funktionen einer Software zuzugreifen. Bei Serendipity benötigt man ein solches Regelwerk vor allem für Erweiterungen mit eigenen Plugins oder den Zugriff von fremder Software.

Serendipity bietet eine funktionsorientierte Kern-API zur Verwaltung von Artikeln und Datenbankabfragen. Weiterhin gibt es eine spezielle, objektorientierte Plugin-API, die alle notwendigen Methoden abdeckt, die man als Entwickler eigener Plugins benötigt.

Ein *Plugin* ist Software, die sich nahtlos in ein Hauptprogramm einbindet (*to plug* bedeutet *einstöpseln*). Es hat Vorteile, gewisse Funktionalitäten in Plugins bzw. Programmmodule auszulagern. Dadurch hält man zum einen die Kernsoftware schlank, zum anderen kann man los-

gelöst von der Kernsoftware Aktualisierungen durchführen und Funktionalitäten erweitern. Ein Benutzer hält sein System schlank und performant, indem er nur die Plugins installiert, die er für seine Anwendungsfälle nutzt. Auch bei Webservices gibt es zahlreiche APIs, die den Verkehr zwischen Client und Server jeweils strukturieren.

1.4.10 Mashup

Der Begriff *Mashup* bezeichnet die Verkettung mehrerer Webservices. Dank der Offenheit vieler APIs lassen sich die abgefragten Daten beliebig miteinander kombinieren, z. B. die Wetterdaten eines mittels Google Maps dargestellten Orts anzeigen. Oder man bindet auf der Karte die Bilder ein, die Flickr-Benutzer zu diesen Geo-Koordinaten eingestellt haben. Die Verknüpfungsmöglichkeiten sind oft nur durch Ihre Phantasie begrenzt.

1.4.11 Usability, Barrierefreiheit, Accessibility

Usability ist ein Forschungsfeld, das sich damit beschäftigt, wie man z. B. eine Webseite am besten *bedienen* kann. Es beschäftigt sich u. a. mit der konzeptionellen Struktur einer Seite, dem Aufbau der Navigation, der Gliederung der Inhalte und auch der Einbindung von Formularen, Suchen und anderen interaktiven Elementen.

Da sich eine Webseite häufig stark von einer normalen Software auf dem Computer unterscheidet, lassen sich Software-Usability-Weisheiten nicht einfach auf Webseiten übertragen. Wie man dennoch zielgerichtet auf Benutzer eingeht und deren Erwartungen erfüllt, ist Aufgabe der Web-Usability.

Eine Teilmenge des Oberbegriffs Usability ist die Barrierefreiheit (englisch *Accessibility*). Diese beschreibt Techniken und Maßnahmen, die auch behinderten Personen die Benutzung einer Webseite ermöglichen. Darunter zählen Dinge wie eine geschickte Farbwahl, sinnvolle Kontraste, aber auch der gezielte Einsatz von Bildern nur an Stellen, wo sie nötig sind.

XHTML erlaubt zudem die Trennung von Layout und Inhalt, die man der Barrierefreiheit zuliebe stets beachten sollte. Gültiger, valider Quellcode ist Voraussetzung dafür, dass jeder Browser (und damit jeder Besucher) Ihre Webseite problemlos bedienen kann. Denken Sie auch an blinde Menschen, die die Inhalte Ihrer Seite vorgelesen bekommen müssen: Wenn Sie die Navigation nicht speziell hervorheben, kann es sein, dass dem Benutzer diese beim Vorlesen vorenthalten wird.

Serendipitys Standard-Templates versuchen so barrierefrei wie möglich zu sein und bei der Usability auf klare Strukturen zu achten. Erfahrungsgemäß ist dies bei einem Open-Source-Projekt recht schwer, da meist nur Programmierer und erfahrene Benutzer am Werk sind, die keinen Wert auf einfache Zugänglichkeit legen. Serendipity hat hier im Laufe der letzten Jahre sehr stark zugelegt und ist daher für eine barrierefreie Zukunft bestens gerüstet. Wie Sie diese in Ihrem Blog letztlich selber umsetzen, wird ab Seite 469 ausführlich besprochen.

1.4.12 BBCode, Textile, Markup und Textformatierung

Üblicherweise werden Blog-Artikel in HTML-Syntax formatiert. Da HTML jedoch gerade für Anfänger nicht ganz trivial ist, gibt es mehrere Standards, die das Ziel einfacher Textformatierung verfolgen. Solche Standards, darunter BBCode[19] oder Textile[20] bieten eigene *Tags* (also Formatierungsmarker) an, die ein System wie Serendipity später in HTML-Format konvertiert.

BBCode basiert auf in Foren üblichen Markierungen: `[b]`*`text`*`[/b]` formatiert z. B. das Wort *`text`* fett. Alle Tags werden in eckigen Klammern geschrieben; Vereinfachungen wie `[URL=http://www.google.com]Google[/url]` erlauben es, relativ einfach Links zu setzen.

Serendipity bietet für alle bekannteren Markups Plugins an, so dass auch Kommentatoren zu einem Blog-Eintrag auf diese Syntax zugreifen können. Die direkte Eingabe von HTML ist bei Kommentaren nicht möglich, daher stellen solche Plugins die einzige Möglichkeit der Formatierung dar.

1.4.13 WYSIWYG

Eine weitere Möglichkeit, die Redakteuren die Eingabe von Text erleichtert, ist der sogenannte WYSIWYG-Modus: *What You See Is What You Get*. JavaScript-Module erstellen im Webbrowser ein Texteingabefeld. Dieses enthält Formatierungsbuttons ähnlich wie bei OpenOffice bzw. Microsoft Word, die es erlauben, die Farbe oder den Schriftstil zu ändern, per Drag&Drop Texte und eingebundene Bilder zu verschieben oder Tabellen einzufügen.

Ein Nachteil solcher WYSIWYG-Komponenten ist der, dass sie komplett im Browser ablaufen und daher stark vom eingesetzten Browser abhängen. Gerade zwischen Internet Explorer und Mozilla Firefox gibt es hier öfters große Unterschiede, was Kompatibilitätsprobleme nach sich zieht. Da die WYSIWYG-Editoren die Umformung in HTML selbständig vornehmen müssen, führt dies oft zu Problemen mit unsauberem bzw. kaputtem HTML.

Trotz des Fortschritts im Web 2.0 sind solche Probleme immer noch nicht Browser-übergreifend gelöst, so dass Benutzer von WYSIWYG-Komponenten oft Einschränkungen hinnehmen müssen. Serendipity selbst ermöglicht die Einbindung freier WYSIWYG-Module wie tinyMCE[21], Xinha[22] und FCKEditor[23].

---

[19] `http://www.phpbb.com/community/faq.php?mode=bbcode`
[20] `http://www.textism.com/tools/textile/`
[21] `http://tinymce.moxiecode.com/`
[22] `http://xinha.webfactional.com/`
[23] `http://www.fckeditor.net/`

1.4.14 Widgets, Nuggets, Blogroll

Viele Webservices erlauben es, ihre Dienste mit einem einfachen Code-Schnipsel auf der eigenen Seite einzubinden. Beliebte Beispiele dafür sind z. B. Google AdSense, das aktuelle Wetter, die letzten Flickr-Bilder und vieles mehr. Solche eingebundenen Schnipsel bezeichnet man als *Widgets* oder *Nuggets/Klötze*; sie lassen sich auch bei Serendipity leicht ins eigene Blog einbauen.

Ein Beispiel für ein Widget ist die sogenannte *Blogroll*. Dies ist letztlich nichts anderes als eine Liste von Blogs, die man persönlich liest oder als empfehlenswert betrachtet. Sie erlaubt es z. B., den eigenen Freundeskreis virtuell zu präsentieren. Durch Blogrolls werden die Leser eines Blogs auch auf thematisch ähnliche oder sozial verbundene Blogs aufmerksam. Kein unwichtiges Feature, wenn man bedenkt, dass sich die Popularität häufig verlinkter Blogs in Suchmaschinen steigert.

1.5 Templates, Themes und Styles

Alles, was der Besucher eines Blogs sieht, entspringt einem zentralen Layout. Die Begriffe *Template*, *Theme*, *Style* und*Layout* sind im Kontext von Serendipity grundsätzlich synonym und bezeichnen lediglich eine Ansammlung von Dateien, die später in HTML-Code und Design übersetzt werden. Der HTML-Code eines Templates enthält Platzhalter (sogenannte Smarty-Variablen), an deren Stelle später der Artikeltitel, -text oder andere dynamische Inhalte erscheinen.

1.6 Spam, Bots, Captcha

Mit zunehmendem kommerziellen Gewicht des Internets ist es attraktiv, für Webseiten auf so vielen Partnerseiten wie möglich zu werben. Sobald der „Partner“ aber mit Werbung bombardiert wird oder diese unfreiwillig z. B. in Form von Gästebucheinträgen und Kommentaren zu redaktionellen Inhalten erhält, spricht man gemeinhin von *Spam*. Dabei helfen automatisierte Systeme (*Bots*) den Spammern, ihre Werbung möglichst vielfältig und schnell auf fremde Webseiten zu schleusen. Wie beim Medium E-Mail, in dem mehr als zwei Drittel aller weltweit versandten E-Mails unerwünschte Werbemails sind, geht man bei Foren und Weblogs davon aus, dass eine ähnlich hohe Quote von Beiträgen aus automatisiertem (und teilweise abgewehrtem) Spam besteht.

Software kann nur sehr schwer unterscheiden, ob der Textbeitrag eines Besuchers „echt“ ist oder von einem automatischen System stammt. Aufgrund der Funktionsweise des Internets ist es nicht möglich herauszufinden, ob ein Aufruf einer Webseite von einem Menschen oder von einer Maschine stammt. Daher beschäftigen sich die meisten Techniken zur Spamabwehr damit, diesen Unterschied herauszuarbeiten.

Schon in den Anfangszeiten der Informatik postulierte der Mathematiker Alan Turing, wie

sich Mensch und Maschine unterscheiden lassen: Bei Turing-Tests setzt man Abfragetechniken ein, die die Interaktion eines Menschen erfordern: Er soll logische Aufgaben lösen, Farben oder Bilder analysieren etc. – alles Dinge, die eine Maschine nach heutigem Forschungsstand nicht können sollte.

Eine gebräuchliche Form dieses Tests nennt sich CAPTCHA (*Completely Automated Public Turing Test to tell Computers and Humans Apart*). Captchas nutzen Grafiken mit einer Zahlen- oder Buchstabenkombination, die häufig noch dazu visuell verfremdet dargestellt wird. Der vor dem Rechner sitzende Mensch soll diese identifizieren und eingeben. Nur wenn die eingetippten Zeichen mit denen auf dem Bild übereinstimmen, gilt eine Nachricht nicht als Spam.

Ein großes Problem dieser Grafiken ist der höhere Performancebedarf sowie die Tatsache, dass sehbehinderte Menschen bei der Entzifferung oft Probleme haben. Auch sind Bot-Entwickler nicht untätig und haben mittels Schrifterkennung per OCR[24] große Fortschritte gemacht, so dass sie geläufige Captchas doch automatisiert entziffern können.

1.7 RDF, Semantic Web, Mikroformate

Schon heute lässt sich die Datenflut des Internets kaum bewältigen, noch ist sie annähernd überschaubar. Volltextsuchmaschinen sind normalerweise nicht in der Lage, die im Web publizierten Texte anhand der üblichen grammatikalischen Syntax miteinander in Beziehung zu bringen. Um dieses Problem zu beseitigen, wurde die Initiative des *Semantic Web* (Semantisches Internet) gegründet. Diese entwarf eine Syntax namens RDF (*Resource Description Framework*) auf Basis der XML-Regeln. RDF-Anweisungen lassen sich in XHTML-Daten einbetten und repräsentieren *Metadaten* (etwa Bildunterschriften, thematische Eingrenzungen, Querverbindungen . . .) zu einem gegebenen Dokument.

Anhand der Metadaten können Suchroboter und andere Webservices Seiten leichter analysieren und gewichten. Je mehr Metadaten verfügbar sind, desto engmaschiger lassen sich alle Informationen des Internets miteinander verknüpfen – und das nun auch nicht mehr nur einseitig.

Zum heutigen Zeitpunkt ist die Erfassung von Metadaten eine recht aufwändige Sache und obliegt dem Redakteur eines Textes. Daher sind brauchbare RDF-Metadaten noch recht wenig verbreitet und beschränken sich auf sogenannte *Mikroformate*. Darunter versteht man Abwandlungen des RDF-Standards oder einfacher: XML-Attribute, die ein Dokument um spezialisierte Metadaten ergänzen, z. B. um Visitenkarten, Kalenderdaten oder Produktbewertungen.

[24] *Optical Character Recognition*

1.8 Tagging

Häufig handelt es sich bei Blogs um lose Artikelsammlungen, die der Autor mal mehr und mal weniger miteinander in Bezug gesetzt hat. Klassischerweise wurden Artikel thematisch ausschließlich festen Kategorien zugeordnet. Damit stand der Redakteur vor dem Problem, sich bereits von vornherein Gedanken über eine hierarchische Struktur seiner zukünftigen Texte machen zu müssen.

Aus dieser Not heraus wurde das *Tagging* erfunden: Einem Artikel lässt sich so dynamisch eine beliebige Menge an Stichwörtern zuordnen. Mit deren Hilfe kann sich ein Benutzer zu Artikeln weiterführen lassen, die dasselbe Stichwort tragen. Das geht z. B. über *Tagwolken*, also Übersichten der meistgebrauchten Tags, die häufig durch unterschiedliche Schriftgrößen gewichtet werden.

Gerade im Kontext von Foto-Providern wie Flickr ist Tagging sehr sinnvoll. Ein Bild einer starren Hierarchie zuzuordnen bedeutete viel Arbeit, ohne zwingend für mehr Übersicht zu sorgen. Anhand der Stichwörter jedoch kann man zügig auch andere Bilder mit ähnlichem Kontext auffinden, z. B. weitere Fotos, die mit dem Begriff „Hochzeit“ verkettet sind.

1.9 Wiki

Der Begriff *Wiki* steht für eine offene Webseite, auf der sich Informationen von vielen Benutzern gleichzeitig zusammentragen lassen. Mittels Versionierung und Protokollierung werden unterschiedliche Bearbeitungen hervorgehoben und miteinander kombiniert.

Zum großen Erfolg der Wikis trug ihre einfache Syntax bei. Anstelle komplizierter HTML-Regeln beschränken sich Wiki-Anwendungen auf einfache Auszeichnungen (die je nach Software jedoch variieren können). Beispielsweise lässt sich ein Wort durch Einschließen in Sternchen (`*`) fetten.

Das zentrale Problem von Wikis ist der Vandalismus ihrer Benutzer – die Freiheit in einer kollaborativen Umgebung können Einzelne ausnutzen, indem sie destruktive Änderungen vornehmen, was zu hohem Moderationsaufwand führt. Gerade im Bereich des freien Wissensmanagements (Stichwort *Wikipedia*) hat sich eine solche Form der technisch lockeren Zusammenarbeit jedoch bewährt.

1.10 AJAX

Eine Technologiekomponente, die das Web 2.0 nachhaltig verändert hat, wird unter dem Schlagwort AJAX (*Asynchronous Javascript and XML*) geführt. Letztlich handelt es sich dabei um eine einfache Form von JavaScript, die XML als Basis für Variablen und Daten verwendet. Innerhalb des Browsers können in dieser Sprache geschriebene Skripte mit Webservices kommunizieren und so Webseiten weiter dynamisieren. Setzte früher jede Aktion auf

einer Webseite das Ausfüllen und Absenden von Formularen voraus, können nun auch Aktionen ausgeführt werden, ohne dass der Benutzer merkt, dass neue Komponenten nachgeladen werden.

Dies bringt Web-Anwendungen den gewohnten Desktop-Anwendungen um einiges näher, stellt jedoch auch höhere Anforderungen an Programmierer und Clients und führt zu neuen Arten von Sicherheitslücken.

1.11 Pod- und Vodcast

Dass Internet-Surfer über ständig höhere Bandbreiten verfügen, schafft den wachsenden Bedarf an Multimedia-Inhalten. Webseiten (und damit auch Blogs) lassen sich um Videos oder Audio-Dateien ergänzen, die im Browser automatisch abgespielt werden. Da Videos und Audio-Dateien auch in die RSS-Feeds der Blogs eingebunden werden können, die MP3-Player wie Apples iPod automatisch importieren, spricht man hier von *Podcast* (Audio) oder *Vodcast* (Video).

1.12 Moblogging

Das mobile Bloggen via Handy oder PDA bezeichnet man als *Moblogging*. Auf Reisen hat man meist keinen Zugriff auf einen normalen Webbrowser. Um dennoch Inhalte in einem Blog publizieren zu können, gibt es die Möglichkeit, eine E-Mail an das eigene Blog zu schicken, die automatisch als Beitrag veröffentlicht wird. Auch gibt es inzwischen mobile XML-RPC-Anwendungen, die über eine GPRS/UMTS-Verbindung direkt auf das Blog zugreifen.

1.13 (Hyper-)Links, URLs und Permalinks

Ein *Hyperlink* (Kurzform: *Link*), umgangssprachlich auch *Verknüpfung* oder *Verweis* genannt, gibt die Fundstelle eines Dokuments im Internet an und erlaubt es so, verschiedene Dateien miteinander zu vernetzen. Das Konzept ähnelt Querverweisen oder Fußnoten bei einem Buch: Zeigt Ihnen Ihr Browser einen speziell markierten Begriff in einer HTML-Seite, können Sie durch dessen Anwahl zur referenzierten Internetseite springen. Erst durch diese Verweise werden die Dateien unterschiedlicher Webserver miteinander in Verbindung gesetzt und stellen dadurch das ursprüngliche Kernkonzept der HTML-Auszeichnungssprache dar.

Den Begriff *Link* benutzt man oft als Synonym für den Fachbegriff URL (*Uniform Resource Locator*). Was sich so kompliziert anhört, ist das, was Sie üblicherweise unter dem Namen *Internetadresse* bereits kennen. Diese beschreibt, auf welche Art und Weise (das heißt: über welches Protokoll, siehe Seite 48) und von welchem Rechner sich das Dokument beziehen

lässt (`http://www.ebay.de`, `http://www.example.com`). Oft enthält sie zudem Pfadangaben/Dateinamen wie in `http://www.s9y.org/12.html`.

Außer den URLs gibt es noch die URIs (*Uniform Resource Identificators*). Deren Definition geht über die der URLs hinaus. Eine URI kann sogenannte *URL-Variablen* enthalten, die nicht Bestandteil einer URL sind. Sie werden üblicherweise von einem `?`-Zeichen eingeleitet und sind mittels `&` voneinander getrennt. In vielen Fällen ist der Unterschied zwischen URL und URI für Sie als Leser unerheblich, und oft wird von einer URL gesprochen, wenn man fachlich eigentlich URI sagen müsste. Als Merksatz gilt: Sobald an eine URL Variablen angehängt sind, muss man von einer URI sprechen.

Ein Permalink ist eine spezielle URL, die den Inhalt eigentlich dynamischer Webseiten permanent „eingefroren“ zugänglich macht. Bei deren Aufruf erhält ein Besucher immer dieselbe Seite, egal, wann er sie aufruft. Obwohl sich Übersichtsseiten aufgrund des chronologischen Bezugs bei jedem neuen Blogeintrag ändern, erlaubt es ein Permalink, Artikel eindeutig zu identifizieren und immer wieder aufrufen zu können.

Permalinks können in Serendipity nach Benutzervorgabe formatiert werden und unterschiedliche Prä-/Postfixe aufweisen (siehe Seite 77).

1.14 Suchmaschinenoptimierung

Der Bereich der *Suchmaschinenoptimierung* (kurz SEO für *Search Engine Optimization*) ist ein sehr weites Feld. Um Ihre Webseite möglichst weit oben in den Suchergebnissen von Google und Co. erscheinen zu lassen (ihnen also ein hohes *Ranking* zu verschaffen), gibt es einige Tricks und Kniffe und vor allem auch Konzeptionelles zu beachten.

Webseiten werden heutzutage fast automatisch von allen Suchrobotern gefunden. Jeder Link, der auf Ihre Seite führt, gibt dieser mehr Gewicht bei der Berechnung des *Page Rank* bei Google[25]. Die Begriffe, die Sie auf Ihren Unterseiten angeben, werden jeweils verschlagwortet und erscheinen in Abhängigkeit von der Gewichtung an oberer oder unterer Stelle in der Liste entsprechender Suchmaschinenergebnisse.

Serendipity kann Ihnen teilweise bei der Suchmaschinenoptimierung helfen. Durch sprechende URLs (siehe Seite 30) lassen sich Artikeltitel mit in die URL aufnehmen, was die Gewichtung erhöht. Die Verwendung korrekt verschachtelter Überschriften innerhalb der HTML-Seite und eine gültige HTLM-Syntax steigern die Effektivität eines Suchroboters. Weiterhin gibt es Plugins, mit denen Sie Kreuzverkettungen zu ähnlichen Artikeln in Ihrem Blog erstellen können (siehe Seite 399) oder auch eine spezielle *Google-Sitemap*-Datei (siehe Seite 306) anlegen.

Letzten Endes zählt aber auch die Strukturierung Ihrer Artikeltexte sehr viel. Die Verteilung der (Schlüssel-)Wörter ist ebenso wichtig wie die Benutzung von Wörtern, die Ihre Besucher womöglich zu einem Thema erwarten. Nicht unwichtig kann der Aufbau eines

[25] Andere Suchmaschinen wie Yahoo benutzen vergleichbare Algorithmen, die ähnlich wie die Rezeptur eines schwarzen Zuckergetränkes als Firmengeheimnis gelten und Grund vieler Spekulationen sind.

Partner-Netzwerks sein, da Verlinkungen auf Ihre Seite den Page Rank erhöhen. Hier wird jedoch auch viel unseriöse Geschäftemacherei betrieben. Hüten Sie sich also vor Tricks, die womöglich ins Gegenteil umschlagen können.

Letztlich erhalten Sie gute Suchmaschinenergebnisse nur dann, wenn Sie für Ihre Besucher interessante Texte oder Services anbieten. Alle anderen technischen Kniffe nutzen Ihnen nichts, wenn die Besucher herausfinden, dass Ihre Webseite nicht das hält, was sie durch eine Suchmaschine verspricht.

1.15 Protokolle: HTTP, FTP, SSH

Im Internet gibt es eine beinahe unendliche Zahl verschiedener Protokolle, die Regeln definieren, nach denen Rechner, Anwendungen und Server miteinander kommunizieren. Im Zusammenhang mit Serendipity sind folgende Protokolle von Belang:

- HTTP, das *Hypertext Transfer Protocol*, ist sicherlich das geläufigste, denn Sie verwenden es bei jedem Abruf einer Internetseite. Erkennbar ist dies daran, dass die meisten URLs, die Ihr Webbrowser anzeigt, ein vorgestelltes `http://` enthalten. Über dieses Protokoll überträgt der Webserver ein Dokument an den eigenen Computer, wobei der Webbrowser als Empfänger dient.

  Die Sonderform `https` tritt auf, wenn ein Webserver eine sichere, verschlüsselte Verbindung zur Datenübertragung nutzt. Dabei kommt das Protokoll `SSL` (`Secure Socket Layer`) zum Einsatz, so dass die übertragenen Daten durch den Einsatz von Zertifikaten ausschließlich vom Server und vom Client entschlüsselt werden können. Bei `http` ist es grundsätzlich möglich, dass auch andere Benutzer oder Proxies die Übertragung belauschen können.

- Obwohl auch HTTP als Protokoll zum Dateitransfer dient und mittlerweile oft dafür eingesetzt wird, gibt es ein noch spezielleres (und älteres) Protokoll für den Dateiaustausch: das *File Transfer Protocol* FTP. Dieses Protokoll ermöglicht nicht nur den Empfang (*Download*), sondern auch ein Senden von Dateien (*Upload*).

- Im Gegensatz zu grafischen Betriebssystemen wie Windows kann Unix komplett mit textbasierten Befehlen gesteuert werden, was sich gerade im Einsatz als Webserver als große Erleichterung herausgestellt hat. Das geht auch ferngesteuert übers Netz, am besten über einen abgesicherten SSH-Zugang.[26]

  Diese Zugriffsmöglichkeit auf einen Server bezeichnen viele Leute irrtümlicherweise auch als `root`-Zugang. Den hat man aber nur, wenn man über das Login (meist als User `root`) volle Zugriffsrechte für den Server bekommt. SSH-Zugang ist aber auch mit reduzierten Zugriffsrechten (als unprivilegierter Nutzer) möglich.

[26] SSH steht für *Secure Shell*; mit *Shell* bezeichnet man ein Programm, das die interaktive Steuerung über Textbefehle ermöglicht.

Web-Provider bieten selten SSH-Zugang an. Häufig steht ausschließlich FTP und HTTP zum Zugriff auf einen Server zur Verfügung. Die Bedienung einer SSH-Konsole setzt zudem einiges an Fachkenntnissen voraus, die wir in diesem Buch nicht vermitteln können. Dennoch wird an Stellen, wo SSH-Zugriff die Arbeit erleichtert, auf diesen eingegangen.

1.16 Frontend, Backend, Admin-Oberfläche

Bei vielen Web-Anwendungen unterscheidet man zwischen *Frontend* und *Backend*. Vereinfacht ausgedrückt ist ein Frontend all das, was der Besucher einer Webseite sieht. Als Backend oder *Admin-Oberfläche* bezeichnet man das, was ein Redakteur oder Administrator nach einem Login in den geschützten Bereich sehen kann.

1.17 Cookies und Sessions

Da HTTP ein *verbindungsloses* Protokoll ist, „vergisst“ der Webserver nach jedem Zugriff, was der Benutzer beim vorigen Aufruf getan hat. Das ist besonders bei dynamischen Web-Anwendungen ärgerlich, da man sich normalerweise bei jeder Aktion erneut einloggen müsste. Als rettenden Kniff gibt es das Konzept der *Sessions* oder Sitzungen.

Webbrowser besitzen einen kleinen Datenspeicher. Dort kann ein Webserver den Browser anweisen, eine kleine Datenmenge auf der eigenen Festplatte zu speichern. Diesen kleinen Datensatz nennt man *Cookie*. Er wird meistens dazu eingesetzt, einen Besucher eindeutig zu identifizieren, so dass er autorisierte Aktionen ausführen kann. Cookies haben einen schlechten Ruf erworben, da viele Werbetreibende sie dazu benutz(t)en, ihre Werbe-Zielgruppe eindeutig wiederzuerkennen und Statistiken über deren Nutzerverhalten zu sammeln. Auch besteht oft die Angst, dass ein Webserver so auf die eigene Festplatte zugreifen könnte.

Diese Furcht ist normalerweise unbegründet. Fast alle Browser ermöglichen es Ihnen, Cookies nur von vertrauenswürdigen Quellen und nur im Bedarfsfall anzunehmen oder abzuweisen. Serendipity benötigt solche Cookies ebenfalls für das Frontend und das Backend, daher muss der Webbrowser so eingestellt werden, dass er Cookies annimmt.

Um Nutzeraktionen als zusammengehörige Session zu erkennen, speichert Serendipity eine einfache Zufallskombination aus Buchstaben und Zahlen, die *Session-ID*, als Cookie im Browser des Besuchers. Bei jedem Folgeaufruf überträgt dieser die ID zurück an den Server, der damit den Besucher identifizieren und zuvor gesetzte Variablen (die *Sessiondaten*) wiederherstellen kann. Sessiondaten werden auf dem Server gespeichert und nicht in Cookies, so dass der Nutzer sie nicht beliebig verändern kann.

Eine Sitzung ist zeitlich begrenzt. Das heißt, dass nach einem gewissen *Timeout* die Sessiondaten auf dem Server (automatisch) gelöscht werden und keinen unnötigen Speicherplatz beanspruchen.

1.18 Parsen und Kompilieren

Der englische Begriff *to parse* beschreibt das maschinelle, automatisierte Auslesen einer Datei. Bei diesem Vorgang interpretiert ein *Parser* etwaige Platzhalter und Instruktionen (Schleifen, Bedingungen . . .).

Die Smarty-Templatedateien enthalten allesamt Anweisungen, die erst nach diesem Vorgang des Parsens gültige HTML-Ausgaben enthalten.

Alternativ bezeichnet man diesen Vorgang auch als *Kompilierung* (von englisch: *to compile*). PHP selbst stellt eine interpretierende Skriptsprache dar, die erst nach der Interpretation der PHP-Funktionen und -Anweisungen einen maschinenausführbaren Code zurückliefert.

Kapitel 2

Einrichtung

Die folgenden Installationsanweisungen sind bewusst sehr ausführlich gehalten, um auch Anfängern den Einstieg zu ermöglichen. Fortgeschrittene Anwender finden eine Zusammenfassung im Abschnitt 2.3 auf Seite 70.

2.1 Wahl der Waffen

Um Serendipity voll auszunutzen, benötigt man einiges an Zusatzsoftware, als Erstes ein Programm, um Dateien (und auch Serendipity selbst) auf den eigenen Server hochzuladen. Üblicherweise geschieht dies mittels eines FTP-Programmes, in manchen Fällen aber auch per SSH bzw. SFTP[1]. Sollten Sie FTP einsetzen, empfiehlt sich ein Programm, das mit *Threads*[2] umgehen kann, was den Upload vieler kleiner Dateien beschleunigt. Für Windows ist SmartFTP[3] für solche Zwecke empfehlenswert. Weiterhin sollte das FTP-Programm auch ermöglichen, Dateirechte mit dem Befehl `chmod` zu ändern.

Zum Entpacken von Serendipity benötigen Sie ein Programm wie 7-Zip oder Winzip oder entsprechende Kommandozeilen-Tools in einer Linux-Umgebung.

Um das Blogsystem zu bedienen, benötigen Sie selbstverständlich einen Webbrowser. Serendipity läuft mit allen gängigen Webbrowsern (Mozilla Firefox, Internet Explorer, Opera, Safari) auch älterer Generation, da JavaScript/Ajax nur im Bedarfsfall verwendet (aber nicht benötigt) wird. Um später leicht Änderungen an Ihrem Layout durchzuführen, empfiehlt der Autor den Einsatz von Mozilla Firefox und die Installation der Extensions „Firebug“[4] und

[1] siehe Abschnitt 1.15 auf Seite 48

[2] *Threading* wird von FTP-Programmen benutzt, um mehrere gleichzeitige Verbindungen zu einem Server aufzubauen.

[3] `http://www.smartftp.com/`

[4] `http://www.getfirebug.com/`

„Web-Developer“[5]. Mittels dieser beiden Erweiterungen können Sie Designänderungen direkt in der Browser-Ansicht testen. Dies erleichtert die Template-Erstellung ungemein.

Zur Verwaltung von Datenbanken und Tabellen gibt es zahlreiche SQL-Anwendungen. Bei der Benutzung von MySQL ist das Web-basierte phpMyAdmin[6] sehr zu empfehlen, für PostgreSQL gibt es analog phpPgAdmin[7] und für SQLite phpSQLiteAdmin.[8] Die Installationsanleitung für diese Web-Anwendungen schlagen Sie bitte auf den jeweiligen Projekt-Webseiten nach.

2.2 Installation

Im Folgenden gehen wir davon aus, dass Sie entweder selbständig einen Webserver mit PHP und Datenbanken eingerichtet haben oder von einem Dienstleister einen entsprechend aufgesetzten Webserver bereitgestellt bekommen.

Dort müssen Sie Serendipity unterhalb des Document Root (siehe Seite 27) ablegen. Im Folgenden nennen wir dieses Verzeichnis exemplarisch `/var/www/example.com`. Alle Angaben, die diesen Pfad enthalten, müssen Sie auf Ihre individuellen Gegebenheiten anpassen.

Sie werden Serendipity in einem Unterverzeichnis dieses Stammpfads installieren, so dass Sie das Blogsystem später über die URL `http://www.example.com/serendipity/` aufrufen können. Ob Sie dieses Unterverzeichnis ebenfalls `serendipity` nennen, bleibt Ihnen überlassen – es darf auch `blog`, `tagebuch` o. ä. heißen. Sie sollten jedoch Sonderzeichen wie den Unterstrich (_) sowie Zahlen am Anfang oder Ende des Verzeichnisnamens vermeiden, da dies zu Problemen mit Serendipitys dynamisch generierten URLs führen kann.

Sie können Serendipity natürlich auch direkt ins Stammverzeichnis selbst installieren. Beachten Sie jedoch dabei, dass dadurch die Installation anderer Software, etwa eines Forums oder eines Statistikprogramms, schwieriger werden könnte, wenn die URL-Umformung aktiviert wurde und Serendipity direkte Zugriffe auf Unterverzeichnisse möglicherweise als eigene Seitenausgabe auffasst.

2.2.1 Upload der Dateien

Das Serendipity-Paket bekommen Sie entweder von der Projekt-Webseite `http://www.s9y.org/` im Bereich **Downloads** oder direkt von der SourceForge-Projektseite.[9]

Auf beiden Seiten finden Sie sowohl die aktuellste Ausgabe als auch Beta- und ältere Versionen und die täglichen *Snapshots*. Obwohl es sich bei Letzteren um die tagesaktuellen Entwicklerversionen handelt, sind diese bei Serendipity gewöhnlich sehr stabil. Neben den

[5] `https://addons.mozilla.org/de/firefox/addon/60`
[6] `http://www.phpmyadmin.net/`
[7] `http://phppgadmin.sourceforge.net/`
[8] `http://www.phpguru.org/static/phpSQLiteAdmin.html`
[9] `http://sourceforge.net/project/showfiles.php?group_id=75065`

vollständigen Installationspaketen gibt es zu einigen Releases auch sogenannte *LITE Releases*. Diese enthalten nur das Standard-Template und sind mit etwa 3 MB etwas kleiner als die anderen Pakete, die ca. 5 MB umfassen.

Außerdem können Sie die Pakete in mehreren Archivformaten herunterladen: als `zip`-Datei sowie als `bzip2` (Dateinamensendung `tar.bz2`) oder `gzip` (Dateinamensendung `tar.gz`) gepackten *Tarball*. Alle drei Varianten enthalten die gepackten PHP-Dateien, die Sie auf Ihren Webserver hochladen müssen. Für Windows-Benutzer ist das `zip`-Archiv am einfachsten zu handhaben, unter Unix empfiehlt sich der Tarball.

Falls Sie via `ssh` direkten Shellzugriff auf Ihren Webserver haben, laden Sie das Paket mittels `wget` ohne Umwege direkt auf den Server und entpacken es mit einem Kommandozeilen-Programm:

```
user@linux:> cd /var/www/example.com/
user@linux:> wget "http://downloads.sourceforge.net/php-blog/serendipity
-1.3.tar.gz?use_mirror=mesh"
[...]
user@linux:> tar -xvzf serendipity-1.3.tar.gz
[...]
```

Anderenfalls laden Sie das Paket auf Ihren eigenen Rechner und entpacken es dort mit einem geeigneten Programm in ein temporäres Verzeichnis. Aus diesem Ordner heraus laden Sie alle Dateien und Unterverzeichnisse des entpackten `serendipity`-Unterverzeichnisses mit allen Dateien mittels FTP-Programm auf Ihren Webserver.

Achten Sie beim Aufspielen der Dateien unbedingt darauf, dass der Transfer nicht an einer Stelle abbricht oder fehlschlägt. Teilweise hochgeladene Dateien können Serendipity unbenutzbar machen. Da insgesamt mehrere hundert kleine Dateien hochgeladen werden, empfiehlt es sich, im FTP-Programm die Benutzung sogenannter Threads zu aktivieren. Damit öffnet es mehrere parallele FTP-Verbindungen und lädt die Dateien so schneller hoch. In jedem Fall sollten Sie sorgfältig auf etwaige Fehlermeldungen achten.

2.2.2 Einrichten der Verzeichnisse

Nach dem Upload finden Sie auf Ihrem Webserver im Verzeichnis `serendipity` einige Dateien mit `.php`-Endung (`index.php`, `comment.php`, `serendipity_admin.php` ...) sowie einige Unterverzeichnisse (`bundled-libs`, `docs`, `htmlarea`, `include`, `lang` ...).

Um Zugriffsrechte auf einem Server zu verwalten, bedienen sich sowohl Windows- als auch Linux-Systeme sogenannter Eigentümer- und Gruppenrechte. Jeder Benutzer des Systems kann Dateieigentümer und jeder Eigentümer kann Mitglied einer oder mehrerer Benutzergruppen sein.

Jede Datei (und jedes Verzeichnis) auf einem Server ist einem Eigentümer und einer Gruppe zugewiesen. Für beide wird der Zugriff auf eine Datei über ein Zugriffs- (oder Ausführbarkeits-

), Lese- und Schreibrecht geregelt. Meist hat ein Eigentümer die vollen Rechte an einer Datei: Er kann sie ansehen, ändern und löschen. Die anderen Mitglieder derselben Benutzergruppe können jedoch abweichende Rechte haben, so dass sie die Datei nur ansehen, aber nicht verändern dürfen. Zudem muss noch geregelt werden, welche Zugriffsrechte Benutzer haben, die weder Eigentümer noch Gruppenmitglieder sind. Auch für diese „anderen Benutzer“, also den „Rest der Welt“, werden Zugriffs-, Lese und Schreibrecht aufgeteilt.

Wenn Sie eine Datei mittels FTP auf einen Webserver laden, tun Sie dies (zwangsläufig) mit dem Benutzeraccount, der für den FTP-Zugriff eingerichtet wurde. Hochgeladene Dateien werden diesem Benutzeraccount und standardmäßig einer Gruppe zugewiesen, der Ihrem Benutzerkonto angehört. Dieses Vorgehen stellt sicher, dass andere Benutzer, die auf demselben Server tätig sind, nicht einfach Ihre Dateien lesen oder verändern können. Optimalerweise haben diese keine Möglichkeit, auf von Ihnen hochgeladene Dateien zuzugreifen.

Die Webserver-Software, die die von Ihnen hochgeladenen Dateien interpretiert und als Webseite ausliefert, läuft ebenfalls mit den Rechten eines bestimmten Benutzers. Dieser heißt häufig `www-data`, `wwwrun` oder auch `nobody`. Damit dieser Systembenutzer Software wie Serendipity ausführen kann, muss er bestimmte Rechte an den von Ihnen hochgeladenen Dateien haben. Das müssen Sie sicherstellen. In üblichen Konfigurationen ist Ihr Benutzeraccount Mitglied derselben Gruppe wie der Webserver-Nutzer, die Lesezugriff auf alle hochgeladenen Dateien hat.

Beim Hochladen von Dateien mittels FTP können Sie in diese Rechte eingreifen. Dazu bieten viele Programme bei Rechtsklick auf eine Datei oder ein Verzeichnis ein Menü namens **Permission** oder **CHMOD** an (Abbildung 2.1).

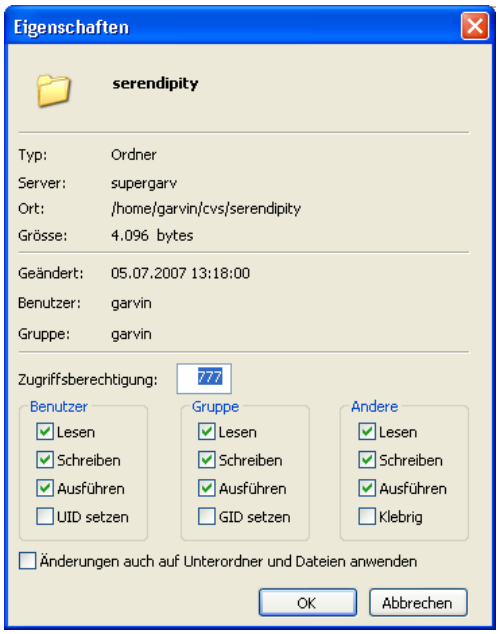

Abbildung 2.1: **CHMOD**-Maske des FTP-Programmes SmartFTP

Dort trägt man, Unix-Konventionen entsprechend, die Zugriffsrechte für Eigentümer/Benutzer (*Owner*), Gruppe (*Group*) und Rest (*Other*) ein: für jede dieser drei Nutzerkategorien eine Zahl. Eine `0` bedeutet „kein Zugriff“, eine `1` „Ausführen möglich“, eine `2` „Schreiben erlaubt“ und eine `4` „Lesen erlaubt“. Diese Zahlen lassen sich addieren, so dass ein Lese- und Schreibzugriff durch die Zahl `6` angegeben wird. Für Verzeichnisse gilt dabei, dass diese nur *betreten* werden dürfen, wenn das Recht mit der Zahl `1` in der Summe enthalten ist, etwa beim Vollzugriff mit `7`.

Mit der Zugriffsmaske `777` erhält jeder Nutzer volle Zugriffsrechte auf die betroffene Datei. Damit stellt man einfach sicher, dass sowohl der eigene Benutzeraccount als auch der des Webservers volle Zugriffsrechte haben – allerdings auch andere Benutzer auf demselben Server. Daher ist es besser, die Zugriffsrechte von Dateien und Verzeichnissen so strikt wie möglich zu setzen. Wie dies konkret bei Ihnen aussieht, klären Sie am besten mit Ihrem Provider.

Damit der Webserver Serendipity ausführen kann, benötigt er Leserechte für *alle* Dateien und Verzeichnisse. Um Serendipity zu installieren, braucht er zudem anfangs Schreibzugriff auf das Stammverzeichnis, damit er die Dateien `serendipity_config_local.inc.php` und `.htaccess` erstellen kann. Darüber hinaus wird das Installationsprogramm die Unterverzeichnisse `templates_c`, `archives` und `uploads` zu erstellen versuchen. Sollte es diese schon geben, müssen Schreibrechte für den Webserver-Benutzer dafür vergeben werden. Nach der Installation benötigt der Webserver nur noch Schreibrechte auf die genannten drei Unterverzeichnisse und zwei Dateien, den globalen Schreibzugriff auf das Stammverzeichnis können Sie also wieder entfernen.

Wenn Sie darüber hinaus das Spartacus-Plugin zum Download von Templates und Plugins nutzen wollen (siehe Kapitel 6.2.7 ab Seite 265), müssen auch die Verzeichnisse `plugins` und `templates` beschreibbar sein (und bleiben).

2.2.3 Einrichten der Datenbank

Nachdem nun also alle notwendigen Dateien hochgeladen wurden und Sie die Zugriffsrechte konfiguriert haben, müssen Sie eine Datenbank für Serendipity einrichten.

Je nachdem, welche Datenbanksoftware auf Ihrem Webserver zur Verfügung steht, kann dieser Vorgang unterschiedlich ausfallen.

Verwendet Ihr Webserver PHP5, ist die Datenbank SQLite automatisch verfügbar. Dieses Datenbanksystem speichert alle seine Datenbanken und Tabellen in einer einzigen Datei. Diese wird im Serendipity-Verzeichnis abgelegt und während der Installation ohne Ihr Zutun automatisch erstellt, es sind daher keine vorbereitenden Eingriffe notwendig.

Beim Einsatz von MySQL und PostgreSQL geht Serendipity davon aus, einen Datenbankbenutzer und eine leere Datenbank vorzufinden. Üblicherweise erhält man die Zugangsdaten von seinem Provider, da der Benutzeraccount bereits besteht. Manche Provider gestehen Ih-

nen nur eine einzelne Datenbank zu, so dass Sie keine neuen erstellen können. Das ist nicht weiter tragisch, da sich Serendipity problemlos mit anderen Anwendungen innerhalb derselben Datenbank betreiben lässt. Viele Provider bieten zudem eigene Oberflächen zur Erstellung einer Datenbank an.

Serendipity benötigt lediglich irgendeine Datenbank für seine Tabellen – man kann also problemlos eine bereits bestehende Datenbank für die Installation benutzen.

Sollte noch kein Datenbankbenutzeraccount bestehen, müssen Sie ihn anlegen. Dabei ist wichtig, dass der Benutzer über alle Rechte verfügt, die Serendipity später im Betrieb benötigt. Konkret heißt das, dass er Tabellen anlegen (`CREATE`), aktualisieren (`ALTER`) und indizieren (`INDEX`) sowie Datensätze anlegen (`INSERT`), aktualisieren (`UPDATE`), lesen (`SELECT`) und löschen (`DELETE`) darf.

Um für MySQL einen solchen Benutzer anzulegen, benutzt man (in einem Programm wie phpMyAdmin, welches die meisten Provider anbieten) folgende SQL-Syntax:

```
CREATE DATABASE serendipity;
GRANT SELECT, CREATE, INSERT, UPDATE, DELETE, ALTER, INDEX ON serendipity
TO 'serendipity'@'%' IDENTIFIED BY PASSWORD 'passwort';
```

Diese beiden Befehle erstellen die leere Datenbank namens `serendipity` und den gleichnamigen Benutzer, der die notwendigen Zugriffsrechte erhält und sich über das Passwort *`passwort`* ausweisen muss. Bei PostgreSQL benutzt man folgende Syntax:

```
CREATE USER serendipity WITH PASSWORD 'passwort';
CREATE DATABASE serendipity WITH OWNER = serendipity;
GRANT SELECT, CREATE, INSERT, UPDATE, DELETE, ALTER ON DATABASE
serendipity TO 'serendipity'
```

Um die Serendipity-Installation erfolgreich durchzuführen, benötigen Sie den Benutzernamen des Datenbanknutzers (hier `serendipity`), dessen Passwort, den Namen einer leeren Datenbank und den Namen (oder die IP-Adresse) des Datenbankservers.

2.2.4 Die grafische Installationsroutine

Nach diesen Vorarbeiten rufen Sie die URL Ihrer Serendipity-Installation im Browser auf, hier also `http://www.example.com/serendipity/`. Versuchen Sie nicht, eine Datei Ihrer lokalen Festplatte aufzurufen oder eine Datei mittels FTP-Zugriff zu *öffnen* (z. B. via Doppelklick). Dies würde Ihnen lediglich den Quellcode der PHP-Anwendung anzeigen. Stattdessen müssen Sie den Webbrowser einsetzen, damit Ihr Webserver die PHP-Anwendung ausführt.

Bei korrekter Einrichtung des Webservers sehen Sie nun eine Kurzübersicht wie in Abbildung 2.2, die aufführt, welche Voraussetzungen der Server erfüllen muss, ehe Sie fortfahren

können. Sollte sich diese entgegen Ihren Wünschen in englischer Sprache präsentieren, konfigurieren Sie Ihren Browser so um, dass Deutsch die bevorzugte Sprache darstellt.

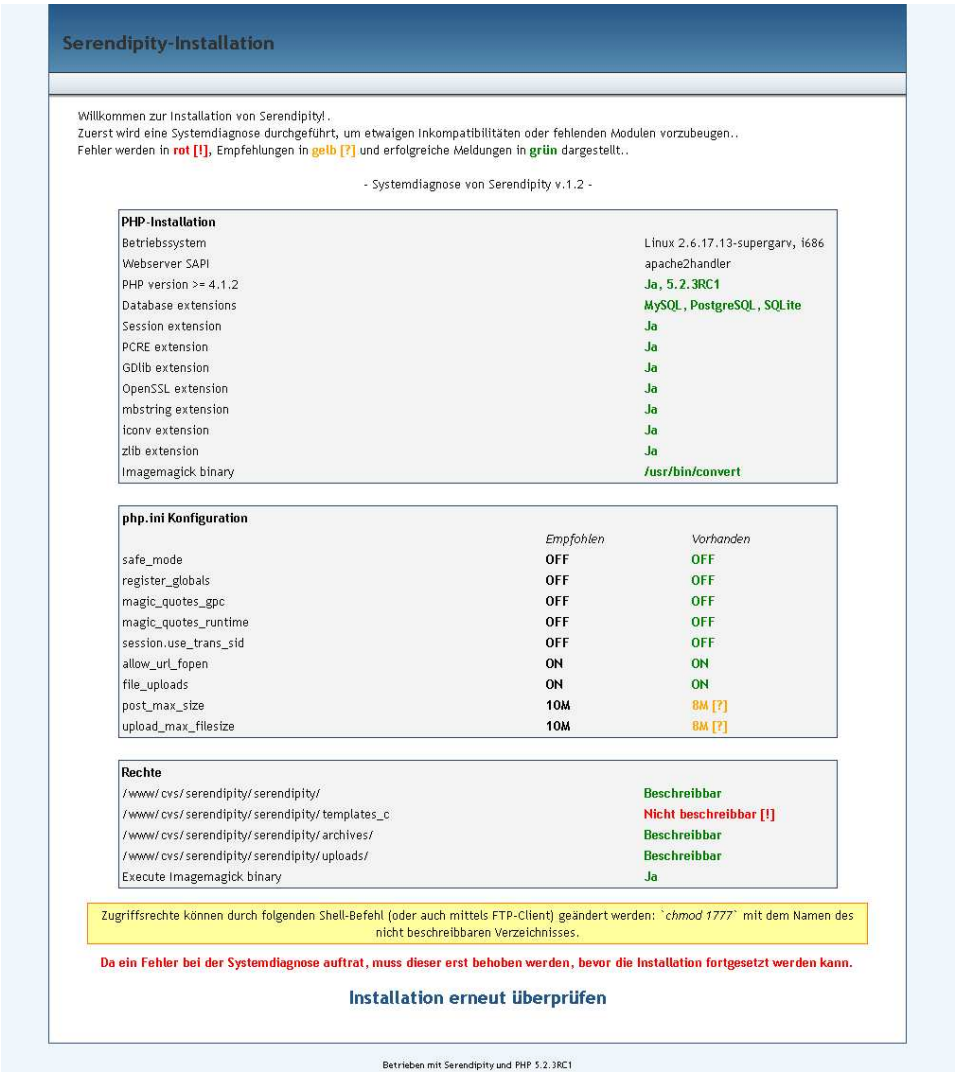

Abbildung 2.2: Installationsbildschirm

Weist Sie die Übersicht auf einen Fehler bei den Schreibrechten oder den Server-Einstellungen hin, beheben Sie ihn und rufen Sie die Installationsoberfläche erneut auf. Warnungen werden mit roter Schriftfarbe dargestellt, eingeschränkte Funktionen oder Hinweise in Gelb; korrekte Einstellungen sind grün hervorgehoben. Für Personen mit Farbschwächen folgt jedem gelben Hinweis ein `[?]`, jeder Warnung ein `[!]`.

Hinweise der Installationsoberfläche zeigen dabei einen Unterschied zwischen dem von Serendipity empfohlenen Wert und der Einstellung auf Ihrem Webserver an. Im Gegensatz zu

Fehlern können Sie Serendipity bei solchen Hinweisen aber trotzdem betreiben, jedoch entweder mit Einbußen bei der Performance oder Funktionalität.

Stimmen die systemseitigen Voraussetzungen, starten Sie die Installationsoberfläche über einen der beiden Links am Ende der Seite: **Einfache Installation** oder **Fortgeschrittene Installation**.

Die **Einfache Installation** verlangt von Ihnen nur die absolut notwendigen Angaben. Dagegen bietet die **Fortgeschrittene Installation** sämtliche Konfigurationsoptionen an, die sich auch später im laufenden Betrieb ändern lassen. Da wir in Kapitel 4.7 ab Seite 155 detailliert auf alle Optionen eingehen, klicken Sie an dieser Stelle am besten auf **Einfache Installation**.

Abbildung 2.3: Initiale Konfiguration

Die Folgeseite (Abbildung 2.3) stellt alle Installationsoptionen in Gruppen unterteilt dar.

Der Abschnitt Datenbankeinstellungen

In diesem Bereich stellen Sie die Zugangsdaten für die gewünschte Datenbank ein. Im Feld **Datenbanktyp** können Sie zwischen allen im PHP-Kern verfügbaren Datenbanktypen wählen. Beim Einsatz von PHP4 wird der Datenbanktyp **SQLite** also gar nicht erst angezeigt.

Den **Servernamen** (meist `localhost` oder ein Name wie `db1231231.puretec.de`) tragen Sie genauso wie den Namen und das Passwort des Datenbankbenutzers in die zugehörigen Felder ein. Beim Datenbanktyp **SQLite** spielen diese Einstellungen keine Rolle und dürfen leer bleiben.

Als Datenbankname tragen Sie den Namen der, entsprechend der Beschreibung auf Seite 55, erstellten Datenbank ein. Darin erstellt Serendipity Tabellen, deren Namen es das Präfix `serendipity_` voranstellt. Diese Vorsilbe ermöglicht die Installation in eine Datenbank, in der bereits andere Tabellen liegen. Wählen Sie in der **Fortgeschrittenen Installation** ein anderes Präfix, können Sie auch mehrere Serendipity-Instanzen auf einem Server installieren.

Der Abschnitt Generelle Einstellungen

Die Optionen im Abschnitt **Generelle Einstellungen** legen den Standard-Benutzer für das zu installierende Blog fest. Tragen Sie als **Admin-Benutzername** also den Namen ein, den Sie später beim Login verwenden wollen. Er sollte möglichst keine Sonderzeichen oder Umlaute enthalten, da dies zu Problemen bei unterschiedlichen Zeichensätzen führen kann. Das Gleiche gilt für das **Admin-Passwort**; lediglich im Feld **Voller Name** können Sie auch auf Sonderzeichen zurückgreifen.

Die Angabe der **Admin-E-Mail**-Adresse benötigt Serendipity beim Verschicken von Hinweisen über neu eingegangene Kommentare und Trackbacks.

Den **Blog-Titel** und die **Blog-Beschreibung** zeigt das Frontend im Kopfbereich an und verwendet beides auch für die Titelzeile im Browser.

Die **Sprache** des Blogs gibt die Standardsprache vor, in der Serendipity Nachrichten und Meldungen darstellt. Später lässt sich für Besucher der Webseite und für jeden Redakteur individuell eine Sprache einstellen (siehe Seite 97).

Der Abschnitt Design und Optionen

Der letzte Einstellungsblock legt bei der einfachen Installation lediglich fest, ob der Standard-Redakteur den WYSIWYG-Editor zur Erstellung von Einträgen benutzen möchte.

Abschluss der Installation

Sind alle Optionen eingetragen (keine Angst, Sie können sie später allesamt wieder verändern), schließen Sie den Installationsvorgang durch einen Klick auf **Vollständige Installation** ab.

Die letzte Seite der Installationsroutine (Abbildung 2.4) informiert Sie über die von Serendipity durchgeführten Aktionen. Dazu gehört das Anlegen der notwendigen Tabellen und des Standardbenutzers und die Einrichtung der Standard-Plugins. Sollten Sie an dieser Stelle noch nicht über eine passende Datenbank verfügen oder andere Fehler auftreten, informiert Sie Serendipity auch darüber. Dann heißt es zurück zu Abschnitt 2.2.3 ab Seite 55.

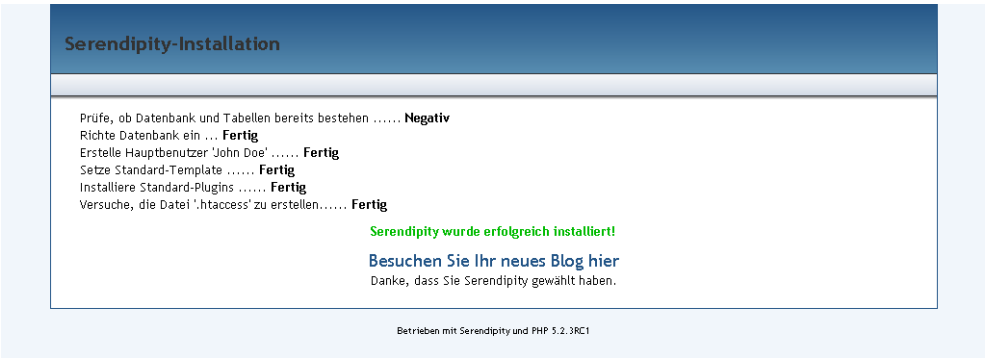

Abbildung 2.4: Einrichtung abgeschlossen

Bei fehlerfreier Installation können Sie über die URL `http://www.example.com/serendipity/` das Frontend aufrufen und unter `http://www.example.com/serendipity/serendipity_admin.php` mit den gewählten Login-Daten auf das Backend zugreifen.

2.2.5 Die Konfigurationsdatei serendipity_config_local.inc.php

Alle grundlegenden Konfigurationsvariablen wie die Daten für den Datenbankzugang und die aktuelle Versionsnummer speichert Serendipity in der Datei `serendipity_config_local.inc.php` im Stammverzeichnis. Diese Datei sieht ungefähr wie folgt aus:

```
<?php
        /*
          Serendipity configuration file
          Written on Wed, 06 Jun 2007 11:22:27 +0200
        */

        $serendipity['versionInstalled']  = '1.3';
        $serendipity['dbName']            = 'serendipity';
        $serendipity['dbPrefix']          = 'serendipity10_';
        $serendipity['dbHost']            = '127.0.0.1';
        $serendipity['dbUser']            = 'root';
        $serendipity['dbPass']            = 'root';
        $serendipity['dbType']            = 'mysql';
```

```
        $serendipity['dbPersistent']      = false;

        // End of Serendipity configuration file
        // You can place your own special variables after here:
?>
```

Ähnlich wie bei der `.htaccess`-Datei (siehe Seite 34) können Sie in dieser Datei später auch eigene Konfigurationsparameter nachtragen. Fügen Sie diese *nach* der Zeile `You can place your own special variables after here:` ein, so dass Serendipity sie bei einer Konfigurationsänderung nicht versehentlich überschreibt. Je nach Konfiguration des Webservers kann es sein, dass Sie keine Schreibrechte für diese Datei besitzen. Dies lässt sich mit dem `fixperm.php`-Skript (siehe Seite 64) ändern.

2.2.6 Fehler bei der Installation

Die zahlreichen Konfigurationsmöglichkeiten von Webservern geben Spielraum für hinterlistige Fehler.

Nur eine leere Seite erscheint

Wenn an einer Stelle der Installation lediglich eine weiße Seite erscheint, liegt dies meist an sogenannten *PHP Fatal Errors*. Normalerweise sieht man diese direkt auf der Webseite, aber einige Webserver sind so eingestellt, dass sie PHP-Fehlermeldungen unterdrücken. Wenn das Installationsskript endet, ohne vollständig ausgeführt worden zu sein, führt dies dann zu den weißen/leeren Seiten.

Um die Fehlermeldung dennoch zu sehen, versuchen Sie, in den PHP-Fehlerlogfiles Ihres Webservers nachzuschauen. Diese liegen meist in einem `logs`-Unterverzeichnis. Den genauen Speicherort solcher Dateien teilt Ihnen der Provider mit.

Man kann aber auch versuchen, im Serendipity-Stammverzeichnis eine Datei namens `.htaccess` anzulegen und mit folgender Zeile zu füllen:

```
php_value display_errors on
```

Beim Speichern der Datei achten Sie bitte unbedingt darauf, dass PHP dafür Schreibrechte erhalten muss, da es diese Datei während der Installation automatisch verändert. Achten Sie auch darauf, dass einige FTP-Programme Dateien, die mit einem Punkt anfangen, verstecken, so dass Sie diese erst nach Umkonfiguration des Programms sehen.

Die Fehlermeldungen geben meist Aufschluss über die Ursache. Oft sind dies fehlende oder defekte Dateien. Enthält eine Datei `Parse Errors`[10], ist dies meist auf fehlerhafte FTP-Uploads zurückzuführen. Auch fehlende Leserechte kommen als Ursache in Frage.

[10] PHP meldet einen Parse Error, wenn eine Datei fehlerhafte Zeichen oder eine ungültige Syntax enthält.

Am häufigsten kann der Ordner `templates_c` nicht beschrieben werden. Dieser ist für von Smarty kompilierte Dateien gedacht, die für die Darstellung des Frontends verwendet werden.

HTTP/500 Fehlermeldung

Serendipity setzt einige Standardoptionen über die Datei `.htaccess`. Von einigen Webservern wird dieser Mechanismus jedoch nicht unterstützt und kann gar zu einem Abbruch des Skripts führen. Sollte sich Serendipity nach der Installation nicht aufrufen lassen, löschen Sie die `.htaccess`-Datei einfach. Sie ist für den Betrieb nicht zwingend notwendig. Ohne diese Datei können Sie jedoch keine URL-Umformung benutzen, die für Serendipity „sprechende URLs“ ermöglicht (siehe Seite 168).

Seiten werden nicht gefunden

Um „sprechende URLs“ wie `http://www.example.com/serendipity/archives/1-Mein-erster-Artikel.html` zu unterstützen, kann Serendipity zwei Arten der URL-Umformung unterstützen. Da Pfade und Dateien wie oben genannt nur virtuell erstellt werden und nicht wirklich auf dem Server liegen, muss Serendipity über Umleitungen auf seine zentrale Datei `index.php` geleitet werden.

Diese Umleitung konfiguriert Serendipity anhand der Datei `.htaccess`. Dort gibt es zwei Varianten der URL-Umformung: *mod_rewrite* und *Apache Errorhandling*.

mod_rewrite ist die komfortabelste Variante der Umformung, benötigt aber ein zusätzliches Webserver-Modul, welches nicht immer verfügbar ist. *Apache Errorhandling* setzt einen Trick ein, der eigentlich nicht gefundene Seiten (*HTTP/404 Not Found*) virtuell auf eine zentrale Seite umleitet, die dann den eigentlichen Inhalt ausgibt.

Serendipity versucht bei der Installation die beste Art für Sie automatisch zu erkennen. Auf manchen Servern gelingt dies jedoch nicht, und es kann sein, dass Serendipity die URL-Umformung fälschlicherweise aktiviert. Wenn diese Umformung nicht klappt, gehen alle Links von Serendipity somit „ins Leere“. Dazu zählen auch Links zum Stylesheet des Frontends, die das Layout beeinflussen.

Löschen Sie die `.htaccess`, um dieses Problem zu lösen. Danach suchen Sie die Serendipity-Administrationsoberfläche via `http://www.example.com/serendipity/serendipity_admin.php` auf, loggen sich ein und gehen über den Menüpunkt **Konfiguration** zu der Unterebene **Design und Optionen**. Dort findet sich die Option **URL-Formung**. Diese Option stellen Sie auf *None* und speichern die Konfiguration. Daraufhin wird die `.htaccess`-Datei neu erstellt.

Nun sollten Sie Serendipity-Unterseiten wieder problemlos öffnen können.

Installation erneut ausführen

Wenn die Installation aus irgendwelchen Gründen fehlgeschlagen ist, möchten Sie Serendipity vielleicht von Grund auf neu installieren. Dazu muss Serendipity in den Ursprungszustand zurückversetzt werden. Löschen Sie dafür die Datei `serendipity_config_local.inc.php`. Diese Datei enthält die Basis-Konfigurationsparameter des Blogs, und sobald diese Datei nicht mehr vorhanden ist, erkennt Serendipity nicht mehr, dass es installiert ist.

Auch die Datei `.htaccess` sollte vor einer neuen Installation gelöscht werden. Bei beiden Dateien ist zu beachten, dass sie von PHP erstellt wurden und von Serendipity mit minimalen Rechten versehen werden, um diese Daten zu schützen. Daher kann es unter Umständen sein, dass Sie mit Ihrem FTP-Zugang keinen Zugriff mehr auf diese Datei haben!

Um dies zu beheben, bedienen Sie sich eines kleinen PHP-Skripts. Speichern Sie eine Datei namens `fixperm.php` im Serendipity-Stammverzeichnis mit folgendem Inhalt:

```
<?php
$ziel = '/var/www/example.com/serendipity/serendipity_config_local.inc.
php';
if (chmod($ziel, 0777)) {
  echo "Rechte geändert.";
} else {
  echo "Fehler: Rechte durften nicht verändert werden.";
  echo "Sie müssen den Provider kontaktieren.";
}
?>
```

Rufen Sie danach die Datei mittels `http://www.example.com/serendipity/fixperm.php` im Browser auf. Nach dem Aufruf sollten Sie die Ausgabe „Rechte geändert“ sehen, und die Dateirechte sollten nun so verändert worden sein, dass Sie wieder vollen Zugriff auf die Datei haben.

Übrigens können Sie dieses kleine Skript auch später benutzen, um andere Dateien im Serendipity-Verzeichnis wieder mit Zugriffsrechten für Sie auszustatten. Dabei müssen Sie lediglich die Code-Zeile verändern, die den Namen der zu ändernden Datei enthält.

Zuletzt müssen Sie noch die von Serendipity angelegten Datenbanktabellen wieder löschen. Wenn diese Tabellen nicht gelöscht werden, würde Serendipity diese bei einer Neuinstallation schützen wollen. Nur wenn alle Tabellen von Serendipity vorher mittels phpMyAdmin oder Ähnlichem gelöscht werden, wird eine vollständige Neuinstallation möglich sein.

Fehler beim Login

Wenn Sie Fehlermeldungen am Anfang der Seite erhalten, die aussehen wie diese:

```
Warning: session_write_close() [function.session-write-close]:
open(/var/lib/php/session/sess_h5a8jerb22q54pkqcjb4qtnqr1, O_RDWR)
failed: Permission denied
```

dann bedeutet dies, dass Ihr Webserver keinen gültigen Speicherpfad für die PHP-Sessiondateien eingetragen hat. Entweder können Sie den Fehler beheben, indem Sie korrekte Schreibrechte zu dem in der Fehlermeldung genannten Pfad einrichten. Oder Sie müssen einen gültigen Pfad in der Datei `php.ini` in der Variable `session.save_path` eintragen.

Ein weiteres Problem, das bei falsch eingerichteten PHP-Sessions auftreten kann, ist, wenn Sie sich nach jedem Klick im Serendipity-Backend erneut einloggen müssen. Dies bedeutet, dass der Webserver Ihre Logindaten nicht in einer Session speichern und Sie somit nicht eindeutig wiedererkennen kann.

Ein ähnliches Problem kann auch dann auftreten, wenn Ihr Browser keine Cookies annimmt, weil sie z. B. von einem Werbefilter oder Anti-Virus-Programm gefiltert werden. Auch kann es passieren, dass, wenn Sie die Serendipity-Oberfläche mit einer URL wie `http://localhost/serendipity/` aufrufen, die Cookies für eine solche URL nicht angenommen werden. In einem solchen Fall müssten Sie Serendipity über die IP-Adresse wie `http://127.0.0.1/serendipity/` aufrufen.

Weiterleitung auf verschiedene URLs

Wenn Sie Serendipity installieren, möchten Sie Ihr Blog möglicherweise unter mehreren URLs aufrufen können.

Ein klassischer Fall dieser unterschiedlichen URLs ist, wenn Ihr Blog sowohl unter `http://example.com/serendipity` als auch `http://www.example.com/serendipity/` verfügbar sein soll. Damit Serendipity die jeweils genutzte URL als Standard für Folgeseiten übernimmt, müssen Sie lediglich in der Konfiguration Serendipitys die Option **HTTP-Hostname** (siehe Kapitel 23 ab Seite 160) aktivieren.

Diese Funktion hilft Ihnen jedoch nur dann, wenn der benutzte Pfad (in diesem Fall `/serendipity/`) in allen Fällen gleich bleibt. Sollten Sie Ihr Blog aber sowohl unter der URL `http://example.com/serendipity/` als auch `http://serendipity.example.com/` darstellen wollen, dann würden Sie diese Vorbedingung nicht erfüllen können. Serendipity kann mit einer derartigen Einstellung nicht umgehen.

Der Grund dafür ist, dass bei einer abweichenden Pfadkomponente die relativen Links nicht mehr korrekt funktionieren. Bei der einen URL müsste Serendipity für den HTTP-Pfad immer `/serendipity/` nutzen, während bei der anderen URL immer `/` genutzt werden müsste. Die Links würden daher nicht übereinstimmen und zu zahlreichen Problemen bei der Darstellung von Grafiken und Verweisen führen.

Das bessere Vorgehen in diesem Fall wäre also die Einrichtung einer Weiterleitung. Dabei sollten Sie sich eine endgültige URL aussuchen und alle weiteren optionalen URLs zu dieser Zielseite weiterleiten.

Weiterleitungen können Sie auf unterschiedliche Weisen realisieren. In unserem Beispiel gehen wir davon aus, dass `http://serendipity.example.com/` die Zielseite und `http://example.com/serendipity/` die Quellseite darstellt.

Bei vielen Standardinstallationen für mehrere verfügbare URLs würden beide URLs bereits auf dasselbe Verzeichnis verweisen. `http://serendipity.example.com/index.php` als auch `http://example.com/serendipity/index.php` würden physikalisch auf dieselbe Datei zeigen, man spricht daher von einem `Alias`. Sollte dies bei Ihnen der Fall sein, haben Sie zwei Möglichkeiten der Weiterleitung:

- Weiterleitung via `.htaccess` bei identischen Serendipity-Verzeichnissen

  Wenn auf Ihrem Server mod_rewrite zur Verfügung steht, können Sie an den Anfang Ihrer `.htaccess`-Datei des Serendipity-Verzeichnisses vor dem Blog `#Begin s9y` folgende Zeilen einfügen:

  ```
  RewriteEngine On
  RewriteCond %{HTTP_HOST} ^example.com [NC]
  RewriteRule ^serendipity/(.*)$ http://serendipity.example.com/$1 [QSA]
  ```

  Diese Regel weist den Webserver an, dass sämtliche Zugriffe zu `http://example.com/` auf die Zielseite `http://serendipity.example.com/` weitergeleitet werden. Sie müssen also jeweils den Quellservernamen, das Quellverzeichnis auf dem Quellserver und den Zielserver in den obigen Zeilen an Ihre Gegebenheiten anpassen.

- Weiterleitung mittels PHP bei identischen Serendipity-Verzeichnissen

  Ohne mod_rewrite müssen Sie die Weiterleitung innerhalb der Datei `serendipity_config_local.in` einprogrammieren. Öffnen Sie diese Datei in einem Editor (bei fehlendem Schreibzugriff schauen Sie sich bitte `fixperm.php` auf Seite 64 an) und fügen Sie in die Zeile vor dem letzten `?>` Folgendes ein:

  ```
  // Ist der ausführende Server der Quellserver?
  if ($_SERVER['HTTP_HOST'] == 'example.com') {
      // Ersetze böse Sonderzeichen
      $ziel = str_replace(
        array("\n", "\r"),
        array('', ''),
        $_SERVER['REQUEST_URI']
      );

      // Entferne Quell-Pfad zur Weiterleitung
      $ziel = preg_replace('@^/serendipity/@i', '/', $ziel);

      // Leite zu Ziel-Server weiter
      header('Location: http://serendipity.example.com' .$ziel);
      exit;
  }
  ```

  Dieser Code sorgt dafür, dass, wenn Serendipity auf dem Quellserver aufgerufen wird, der Besucher automatisch auf die Zielseite weitergeleitet wird.

Sollten durch die Art der Einrichtung jedoch beide URLs (Weiterleitungsquelle und Weiterleitungsziel) auf unterschiedliche Verzeichnisse Ihrer Domain zugreifen, dann können Sie die einfachste Form der Weiterleitung nutzen:

- Weiterleitung mittels `index.html` bei unterschiedlichen Serendipity-Verzeichnissen

  Erstellen Sie die Datei `index.html` in dem Verzeichnis der Quell-URL (in unserem Beispiel `/serendipity/`):

  ```
  <html>
    <head>
      <meta http-equiv="refresh"
      content="0;url=http://serendipity.example.com/">
    </head>
    <body>
    </body>
  </html>
  ```

  Beim Aufruf der Quell-URL wird dann unkompliziert auf die Ziel-URL weitergeleitet.

- Weiterleitung mittels `index.php` bei unterschiedlichen Serendipity-Verzeichnissen

  Etwas schneller geht die Weiterleitung mittels PHP, da die Weiterleitung dann intern im Browser ausgeführt wird und Sie nicht vorher kurz eine weiße Seite im Browser aufflackern sehen. Speichern Sie im Quellverzeichnis folgende `index.php`-Datei:

  ```
  <?php
  header('Location: http://serendipity.example.com/');
  ?>
  ```

Einen Sonderfall dieser Weiterleitung stellt das Beispiel dar, wenn Sie Serendipity zwar in einem Unterverzeichnis Ihres Webservers installiert haben (`http://example.com/blog/`), aber gerne möchten, dass Serendipity beim Aufruf der Haupt-Domain (`http://example.com/`) direkt aufgerufen wird. Die einfachste Variante hier wäre natürlich, dass Sie Serendipity in das Stammverzeichnis Ihrer Webseite installieren, da dies problemlos möglich ist. Sollten Sie dies aus irgendwelchen Gründen nicht bevorzugen, können Sie ebenfalls die oben genannte Methode der Weiterleitung mittels `index.php` oder `index.html` nutzen.

Falsche Anzeige von Datumsangaben oder Sonderzeichen

Serendipity benutzt zur Darstellung von Datumsangaben für Artikel und im Kalender ein serverseitiges System namens *locales*. Diese locales sind, grob gesagt, eine Sammlung von nationalen Sprachbesonderheiten und geben an, wie in beinahe allen gesprochenen Sprachen die Währungs- und Datumsangaben formatiert und geschrieben werden. Damit Serendipity auf möglichst vielen Sprachumgebungen ohne viel Aufwand lauffähig ist, verlässt es sich auf diesen De-facto-Standard.

Ebenfalls zur weitreichenden Unterstützung verschiedener Sprachen unterstützt Serendipity mehrere Zeichensätze. Ein Zeichensatz regelt, mit welchem Binärsystem Buchstaben für den Computer abgespeichert werden – denn je nach Zeichensatz ist für einen Computer „ä“ nicht gleich „ä“. Für die deutsche Sprache ist der standardisierte *ISO-8859-1*-Zeichensatz üblich. Um jedoch auch mit anderen Sprachen benutzbar zu sein, hat sich der Zeichensatz *UTF-8* etabliert, da dieser beinahe alle Sonderzeichen jeder Sprache enthält. Ein arabischer Artikel könnte so also direkt neben einem deutschen Artikel stehen, ohne dass Sonderzeichen falsch dargestellt würden.

Sobald die Zeichensätze vermischt werden, kann es zu Fehldarstellungen im Browser kommen – Mozilla Firefox zeigt statt des korrekten Zeichens dann ein Karo-Symbol mit einem Fragezeichen an. Wenn Sie dies einmal in Ihren Artikeln beobachten, liegt ein solcher Zeichensatzkonflikt vor.

In der Serendipity-Konfiguration können Sie einstellen, ob Serendipity mit nationalem Zeichensatz oder UTF-8 betrieben wird. Standardmäßig wird dies bei der Installation für größtmögliche Kompatibilität direkt auf UTF-8 gesetzt. Sie sollten daher bei einer frischen Installation keinerlei Probleme mit Zeichensätzen erwarten – erst bei späteren Änderungen am System könnten möglicherweise einmal Probleme auftreten.

Die Behandlung von Sonderzeichen zur Speicherung in Datenbanken ist recht komplex und kann mehrere Fehlerursachen haben.

Die erste Fehlerursache kann Ihr Browser sein. Stellen Sie sicher, dass dieser den UTF-8-Zeichensatz darstellen kann. Wenn Sie beim Besuch der Seite `http://www.columbia.edu/kermit/utf8` falsche Sonderzeichen sehen, ist dies ein Indiz für eine fehlerhafte Browserkonfiguration, die Sie mithilfe der Dokumentation Ihres Webbrowser lösen müssen.

Die zweite Fehlerursache kann der Webserver sein, der Zeichensätze selbständig verändert. Apache bietet eine Option an, ein sogenanntes *DefaultCharset* zu setzen, das die Ausgabe von Sonderzeichen eventuell verändern könnte. Wenn der Webserver nicht die von Serendipity bestimmten *Content-Type*-HTTP-Kopfzeilen an den Browser sendet, kann es sein, dass der Webserver durch seine Standardeinstellung die Umbelegung durch Serendipity verhindert. Sprechen Sie daher mit ihrem Provider und bitten Sie ihn, diese HTTP-Kopfzeilen zu überprüfen.

Zu guter Letzt gibt es nun noch die Datenbank, die Artikel mit unterschiedlicher Kodierung speichern kann. Gerade zwischen MySQL 4.0 und 4.1 gab es weitreichende Änderungen in der Behandlung von Zeichensätzen, die sich bei einem MySQL-Update darin äußern konnten, dass Serendipity die Sonderzeichen falsch anzeigte. Um dies zu korrigieren, müssen Sie sicherstellen, dass alle MySQL-Tabellen und -Spalten als *Collation* den übereinstimmenden Zeichensatz benutzen (also `de_latin` beim ISO-Zeichensatz und `utf8` für UTF-8). Sobald dies korrekt übereinstimmt, kann in der Serendipity-Konfiguration die Option **Datenbank-Zeichensatzkonvertierung aktivieren** gewählt werden, was dann wieder zu einer korrekten Darstellung der Sonderzeichen führen sollte.

In einigen Fällen ist es möglicherweise notwendig, einen vollständigen SQL-Export der

Datenbank vorzunehmen und die Datei mittels eines Editors vom ISO-8859-1- ins UTF-8-Format zu überführen, neu zu importieren und danach Serendipity zentral auf UTF-8-Zeichensätze umzustellen. Alternativ können Sie das mit folgendem PHP-Skript erreichen, das einen Datenbankdump namens `dump-iso.sql` in `dump-utf8.sql` vom ISO-8859-1-Zeichensatz in den UTF-8-Zeichensatz verwandelt. Etwaige SQL-Befehle (`COLLATION`), die MySQL-Zeichensätze bestimmen, müssen Sie jedoch manuell in der Datei umwandeln, um sie später als vollständiges UTF-8 importieren zu können:

```
<?php
$input  = file_get_contents('dump-iso.sql');
$output = utf8_encode($input);
$fp     = fopen('dump-utf8.sql', 'wb');
if ($fp) {
  fwrite($fp, $output);
  fclose($fp);
} else {
  echo "Keine Schreibrechte, bitte korrigieren.";
}
?>
```

Zurück aber zu dem Problem mit falsch angezeigten Datumsangaben. Damit der Wochentag „Freitag“ nicht als „Friday“ oder Ähnliches ausgegeben wird, muss eine deutsche Sprachdatei für das locales-System auf dem Server hinterlegt werden. Locales sind auf allen Betriebssystemen verfügbar und in fast allen Fällen vorinstalliert. Je nach System können sie auch nachinstalliert werden (bei Linux z. B. mittels `locale-gen`). Für jede Sprache gibt es abhängig vom Zeichensatz eine Sprachdatei. Für Deutsch ist es üblicherweise `de_DE.UTF8` (UTF-8-Zeichensatz) oder `de_DE.ISO88591` (nationaler Zeichensatz). Manchmal ist auf Servern nur das ISO-Locale installiert, und deshalb wird der Monat „März“ mit einem falschen Sonderzeichen dargestellt. Abhilfe schafft hier die Installation der passenden UTF8-Locale-Datei.

Die Installation der Locales kann nur vom Serverbetreiber vorgenommen werden, im Problemfall müssen Sie sich also an diesen wenden. Die Locales, die Serendipity anwendet, finden Sie im Übrigen in der Datei `lang/serendipity_lang_de.inc.php` bzw. `lang/UTF-8/serendipi de.inc.php` in einer Zeile wie:

```
@define('DATE_LOCALES', 'de_DE.ISO-8859-1, de_DE.ISO8859-1, german,
de_DE, de_DE@euro, de');
```

2.3 Schnelle Installation

- Serendipity-Release-Paket von der Homepage herunterladen und via FTP auf den Webserver hochladen.
- Zugriffsrechte überprüfen: Stammverzeichnis und Unterverzeichnis `archives, templates_c, uploads` müssen Schreibrechte für PHP besitzen, alle anderen Dateien und Verzeichnisse Leserechte.
- Eine leere Datenbank erstellen, falls noch nicht vorhanden. Sicherstellen, dass der Datenbankbenutzer die Rechte `CREATE`, `INSERT`, `UPDATE`, `DELETE`, `ALTER`, `INDEX`, `SELECT` besitzt.
- Via HTTP die Installationsroutine aufrufen: `http://www.example.com/serendipity/`

- Der Installationsroutine folgen und Anfangskonfiguration vornehmen.

Kapitel 3

Frontend

Nach der Installation können Sie unter `http://www.example.com/serendipity/` das Frontend aufrufen. Hier spielt sich alles ab, was jeder Besucher sehen kann: Man kann Einträge lesen und kommentieren, in älteren Archiven stöbern, eine Suche durchführen und auch etliche Plugin-Funktionalitäten (Umfrage, Flickr-Fotos ...) ansehen.

Im Folgenden gehen wir von dem standardmäßig genutzten Theme (siehe Seite 43) *Serendipity 3.0* von Carl Galloway aus. Die meisten anderen Themes richten sich in der Darstellung der Seitenelemente nach demselben Schema, können aber in Details davon abweichen. Wie man die Darstellung anpassen kann, werden Sie in späteren Kapiteln ab Seite 469 lernen.

3.1 Übersicht

Die Startseite eines Serendipity-Blogs zeigt üblicherweise die chronologisch letzten 15 Artikel in allen Kategorien untereinander an, wobei der erste Artikel der aktuellste ist. Je nach Template ist dieser Inhaltsbereich von ein bis zwei sogenannten Seitenleisten umgeben. In den Seitenleisten können sich Seitenleisten-Plugins ansiedeln, die beliebige Inhalte darstellen können. Ebenfalls abhängig vom Template sehen Sie meist eine zentrale Seitenüberschrift (Kopf- oder auch Banner-Bereich genannt) und eine weitere Überschrift vor der Artikelübersicht.

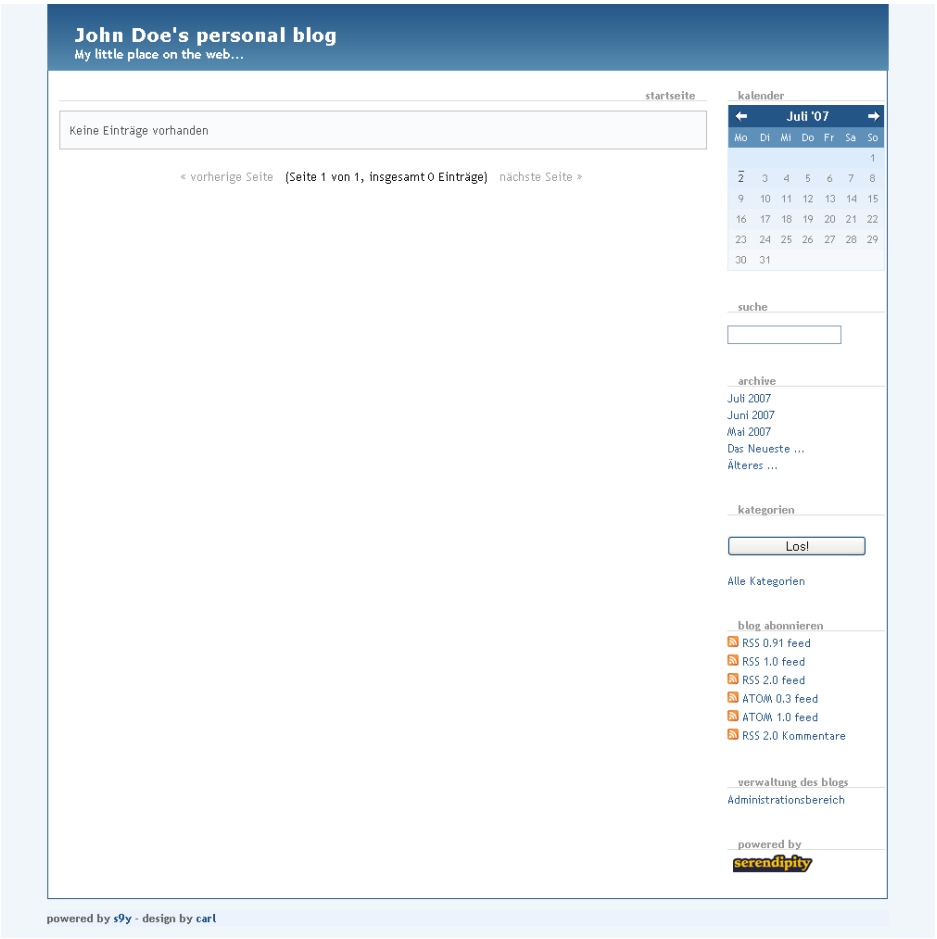

Abbildung 3.1: Frontend nach der Installation

Die Standard-Installation von Serendipity richtet eine rechte Seitenleiste ein, in der folgende Elemente dargestellt werden:

Kalender

Der Kalender stellt den aktuellen Monat in einer grafischen Übersicht dar. Der aktuelle Tag wird hervorgehoben, und jeder Tag, an dem ein Artikel verfasst wurde, ist ebenfalls hervorgehoben. Bei einem Klick auf einen Tag oder auf die Navigationspfeile in der Kopfzeile des Kalenders kann man sich die Artikel anzeigen lassen, die im gewählten Zeitraum verfasst wurden.

Suche

In die Eingabebox der Schnellsuche kann man einen Suchbegriff eingeben und den

Suchvorgang mittels *(Enter)*-Taste ausführen. Die Suche umfasst dabei sowohl Artikelüberschrift als auch den Artikelinhalt und zeigt in der Ergebnisseite die gefundenen Artikel genauso an wie in einer normalen Artikelübersicht eines Zeitraumes.

Details zur Suche sind am Ende des Kapitels (Seite 88) erwähnt.

Archive

Die Links dieses Plugins zeigen unterschiedliche chronologische Abschnitte der Blogeinträge an. Standardmäßig werden die letzten drei Monate mit Link angezeigt, und ein Klick darauf ruft die übliche Artikelansicht auf, bei der jedoch nur der gewünschte Zeitraum berücksichtigt wird.

Der Link **Das neueste** führt zur üblichen Startseite und der Link **Älteres** zu einer besonderen Übersichtsseite (siehe Abschnitt *Archive* auf Seite 77).

Kategorien

In dem Seitenleisten-Block *Kategorien* wird eine Liste aller eingerichteten Kategorien des Blogs dargestellt. Ein Klick auf eine Kategorie wird daraufhin ausschließlich Artikel anzeigen, die dieser Kategorie zugeordnet sind. Neben jedem Kategorienamen befindet sich zudem ein kleines Symbol, das auf den RSS-Feed dieser Kategorie zeigt.

Außerdem besteht für den Besucher die Möglichkeit, mehrere Kategorien zur Ansicht zu kombinieren. So könnte man sich alle Beiträge mehrerer Kategorien anzeigen lassen, indem man die Auswahlboxen neben mehreren Kategorien ankreuzt und danach auf den Button **Los!** darunter klickt.

Alle Kategorien führt zurück auf die Übersichtsseite des Blogs, in der wieder alle Beiträge aller Kategorien gezeigt werden.

Blog abonnieren

Alle Feed-Formate (siehe Terminologie, Seite 38), die das Blog anbietet, werden in dieser Box dargestellt. Die unterstützten Formate sind *RSS 0.91, RSS 2.0, Atom 0.3* und *Atom 1.0*. Zusätzlich gibt es noch einen speziellen RSS 2.0-Feed, der alle Kommentare zu den Beiträgen enthält. Mit diesem Feed können Besucher des Blogs auch auf dem Laufenden über neue Kommentare bleiben.

Verwaltung des Blogs

Damit ein Redakteur sich leicht in das Backend des Blogs einloggen kann, wird ein Link dorthin angeboten. Je nachdem, ob der Benutzer bereits eingeloggt ist, steht hier entweder **Zum Login** oder **Administrationsbereich**.

Powered by

Zuletzt befindet sich in diesem Block eine kleine Werbung für Serendipity. Besucher Ihres Blogs finden die Technik dahinter womöglich faszinierend, so dass sie dieser Hinweis über das eingesetzte System interessiert. Da ein Open-Source-System von seinen Benutzern und seiner Verbreitung lebt, freuen sich die Entwickler immer sehr, wenn diese Werbung nicht entfernt wird.

3.2 Beiträge

Da nach der Installation noch keine Einträge verfasst wurden, wird im Inhaltsbereich nur die Meldung „Keine Einträge vorhanden“ angezeigt.

Unterhalb des Inhaltsbereichs befindet sich der Fußbereich, in dem man je nach dargestellter Seite vor- und zurückblättern kann. Viele Archivansichten können so chronologisch von vorne nach hinten durchblättert werden.

Sobald sich in dem Blog ein Artikel befindet, wird dessen Titel und der einfache Eintragstext angezeigt. Zusätzlich werden Meta-Angaben über einen Artikel eingeblendet: die Erstellungszeit, die Kategorie, in der sich ein Artikel befindet, der Name des Autors und die Anzahl der Kommentare zu diesem Artikel.

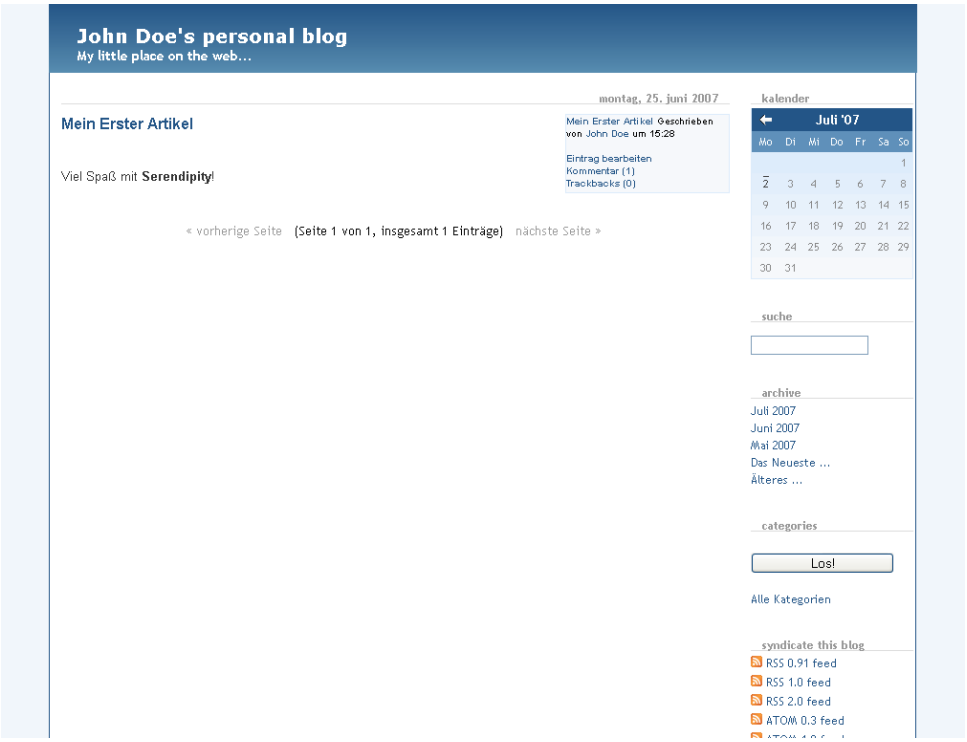

Abbildung 3.2: Frontend mit einem eingetragenen Artikel

Jeder Beitrag kann zudem einen sogenannten *Erweiterten Eintrag* aufweisen. Dieser Zusatztext wird in der Übersichtsseite nicht angezeigt, sondern erst bei der Detailansicht eines Artikels. Sobald ein Artikel solchen Zusatztext enthält, führt der Link **„Artikeltitel“ vollständig lesen** oder ein Klick auf den Titel des Beitrags zu dieser Detailseite.

Auf dieser Detailseite befinden sich außerdem alle Kommentare und Trackbacks zu einem

Artikel. Auch in der Seitenüberschrift verändert sich der Kopfbereich und stellt durch die Aufnahme des Artikeltitels dar, dass man sich auf einer einzelnen Artikelseite befindet.

Sollte man sich bereits in das Backend eingeloggt haben, erscheint bei den Meta-Angaben zu einem Artikel auch ein Link **Eintrag bearbeiten**, mittels dessen man den entsprechenden Artikel direkt bearbeiten kann.

3.3 Archive

Wie bereits erwähnt, verfügt Serendipity über einige besondere chronologische Artikelübersichten, abhängig von der gewählten Kategorie, dem Zeitraum (nach Jahr, Monat, Woche oder Tag), einem Autor und einer Seitenzahl.

Diese Seiten sind alle vom Inhalt her gleich aufgebaut, da alle Ansichten die Artikel gleichartig auflisten. Lediglich die dargestellten Inhalte sind nach dem gewünschten Kriterium gefiltert, und der Kopfbereich sowie der Fußbereich geben die Art der Übersichtsseite an.

Alle Seiten haben zudem einen Stamm-Permalink (siehe Kapitel 1.4 auf Seite 47) gemeinsam: `http://www.example.com/serendipity/archives/`*`XXX`*. Alle Archivseiten sind so einem virtuellen Verzeichnis zugeordnet, die Darstellung richtet sich nach dem dahinterstehenden *XXX*. Unterschiedliche Pfadkomponenten legen dabei fest, was genau angezeigt wird:

`/archives/`*`2007`*`.html`
: stellt alle Artikel des Jahres 2007 dar.

`/archives/`*`2007/12`*`.html`
: stellt alle Artikel aus Dezember 2007 dar.

`/archives/`*`2007/12/01`*`.html`
: stellt alle Artikel vom ersten Dezember 2007 dar.

`/archives/`*`1-Beispiel`*`.html`
: stellt eine einzelne Artikelseite dar.

`/archives/C`*`1`*`.html`
: stellt alle Artikel der Kategorie mit der Nummer 1 dar. Die Pfadkomponente `C` (für *Category*) wird dabei gefolgt von einer Zahl.

`/archives/P`*`2`*`.html`
: stellt die zweite Seite einer Übersicht dar. Die Pfadkomponente `P` (für *Page*) wird gefolgt von einer Seitennummer.

`/archives/A`*`1`*`.html`
: stellt alle Artikel des Autors mit der Nummer 1 dar. Die Pfadkomponente `A` (für *Author*) wird gefolgt von einer Zahl, die den Autor angibt.

`/archives/W52.html`
: stellt alle Artikel der Kalenderwoche 52 dar. Die Pfadkomponente `W` (für *Week*) wird gefolgt von der Wochennummer.

`/archives/summary.html`
: legt fest, dass die Artikelübersicht ausschließlich Datum und Artikeltitel (also ohne jeglichen Inhalt) darstellt.

Wichtig ist dabei, dass die Pfadkomponenten (bis auf die für eine einzelne Artikelseite) hierbei beliebig miteinander kombinierbar sind. Dabei ist nur zu beachten, dass die Angabe des Datums (Jahr, Monat, Tag) immer direkt am Anfang des Permalinks stehen muss. Hier einige Beispiele für die Kombinationsmöglichkeiten:

`/archives/2007/C1/A2.html`
: stellt alle Artikel des Jahres 2007 dar, die in Kategorie 1 vom Autor 2 geschrieben wurden.

`/archives/2007/12/P2/summary.html`
: stellt die zweite Seite der Artikelübersicht (nur Datum und Titel) aller Artikel im Dezember 2007 dar.

`/archives/W52/A1.html`
: stellt alle Artikel des ersten Autors der 52. Kalenderwoche des aktuellen Jahres dar.

Die URL muss immer auf `.html` enden, die letzte Pfadkomponente darf somit keinen abschließenden / enthalten.

Da für die Ansicht der Artikel nach Autoren oder nach Kategorie eigenständige Permalinks wünschenswert sind, bietet Serendipity dieselben Artikelübersichten auch bei Aufruf folgender URLs an:

`/authors/1-AutorenName.html`
: stellt die Artikelübersicht nach Autor dar, wobei in diesem Fall der Name des Autors zu Suchmaschinenzwecken in der URL stehen kann. Mehrere Autoren können hierbei mittels `;` nach der Nummer des Autors kombiniert werden: `authors/1;2;3-Autorenuebersicht.html`. Der Parameter *P* für die gewünschte Seite kann angehängt werden.

`/categories/1-KategorieName.html`
: stellt die Artikelübersicht nach Kategorie dar, wobei in diesem Fall der Name der Kategorie zu Suchmaschinenzwecken in der URL stehen kann. Mehrere Kategorien können hierbei mittels `;` nach der Nummer der Kategorie kombiniert werden: `categories/1;2;3-KategorieName.html`. Die Parameter *P* für die gewünschte Seite sowie *A* für die Einschränkung nach Autoren können zusätzlich angehängt werden.

Über `http://www.example.com/serendipity/archive/` gelangt man zu einer Übersichtsseite mit einer speziellen chronologischen Übersicht.

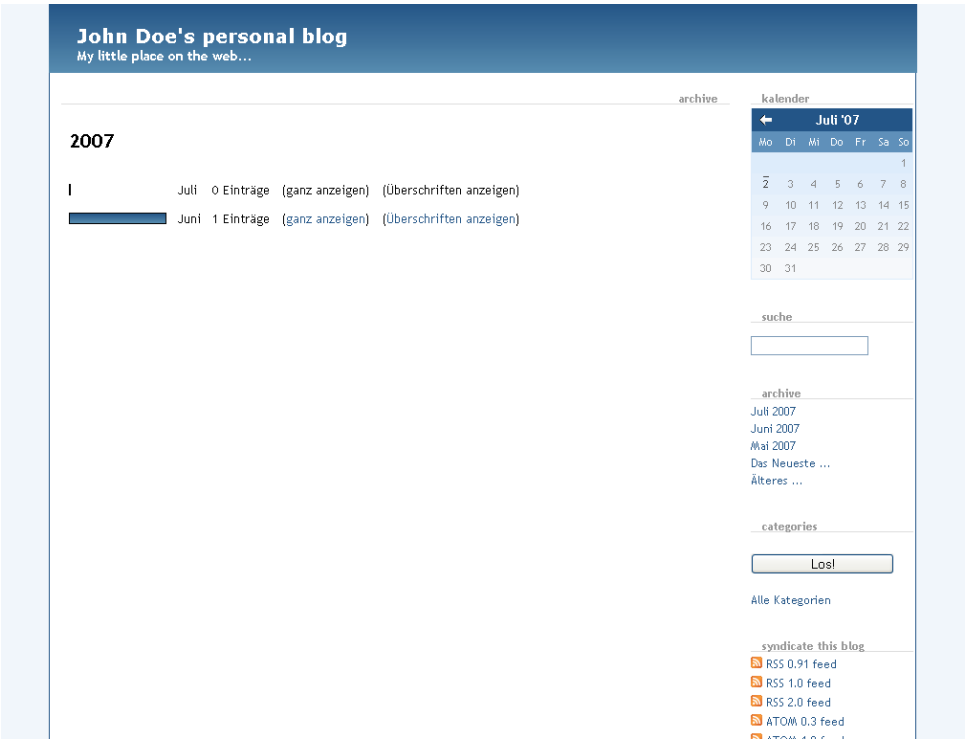

Abbildung 3.3: Archiv-Übersicht

Dort werden für alle Jahre, in denen Artikel verfasst wurden, die Monate dargestellt. Eine Balkengrafik visualisiert hier zusätzlich die Menge der Artikel, die in dem jeweiligen Zeitraum verfasst wurden. Die Links **ganz anzeigen** und **Überschriften anzeigen** führen hierbei wieder zu den eingangs erwähnten Archiv-Übersichtsseiten.

Auch die URL für die spezielle chronologische Übersicht kann mit speziellen Pfadkomponenten eingeschränkt werden – eine Einschränkung nach Zeitraum erfolgt dabei jedoch nicht, da die Übersicht immer den ganzen Zeitraum berücksichtigt.

`/archive/C1.html`
: stellt die Übersicht aller Artikel der Kategorie mit der Nummer 1 dar. Die Pfadkomponente `C` (für *Category*) wird dabei gefolgt von einer Zahl.

`/archive/A1.html`
: stellt die Übersicht aller Artikel des Autors mit der Nummer 1 dar. Die Pfadkomponente `A` (für *Author*) wird gefolgt von einer Zahl, die den Autor angibt.

3.4 Kommentare und Trackbacks

Artikel des Blogs können von Besuchern kommentiert oder referenziert werden. Diese Kommentare werden von Serendipity auf unterschiedliche Weise im Frontend eingebunden und können in weiteren Übersichtsseiten speziell gefiltert werden.

3.4.1 Kommentare zu einzelnen Artikeln

Die Ansicht der Kommentare (und Trackbacks) zu einem Beitrag erreicht man entweder durch einen Klick auf die Detailseite eines Beitrags oder indem man auf den Link **Kommentare (x)** klickt. Je nach Einstellung des Blogs (siehe Seite 167) öffnet sich dann ebenfalls die Detailseite des Beitrags oder ein Popup-Fenster mit den Kommentaren.

Trackbacks zu einem Artikel werden unterhalb eines Beitrags vor den Kommentaren chronologisch geordnet angezeigt. Ein Trackback ist ein Artikel eines fremden Blogs, der sich auf Ihren geschriebenen Beitrag bezieht. Dabei führt der Link des Trackbacks zu diesem fremden Beitrag, und zur Orientierung werden die ersten Absätze des fremden Beitrags mit angezeigt. Damit ein fremdes Blog sich auf Ihren Artikel beziehen kann, ist der Link **Trackback für spezifische URI dieses Eintrags** vorgesehen. Ein Klick darauf verrät Ihnen, dass dieser Link nicht für die Ansicht im Browser gedacht ist, sondern für fremde Blogsysteme übernommen werden muss. Besucher sollten diesen Link also verwenden, um ein Trackback zu Ihnen zu senden.

Nach den Trackbacks folgt die Auflistung der Kommentare. Standardmäßig werden diese *thread-basiert* aufgelistet. Dabei startet die Auflistung mit dem chronologisch ersten Kommentar, und mögliche andere Kommentare, die sich auf diesen beziehen, werden eingerückt darunter dargestellt. Daher nennt sich diese Ansicht *Verschachtelt*. Falls Sie eine chronologisch sortierte Ansicht bevorzugen, kann man diese über den Link **Linear** aufrufen.

Als Redakteur des Blogs sehen Sie zudem noch den Link **Kommentare für diesen Eintrag nicht mehr zulassen**, mit dem Sie die Kommentarfunktionalität eines Beitrags (auch übergangsweise) sperren können. An derselben Stelle erscheint danach auch wieder ein Link zum Entsperren.

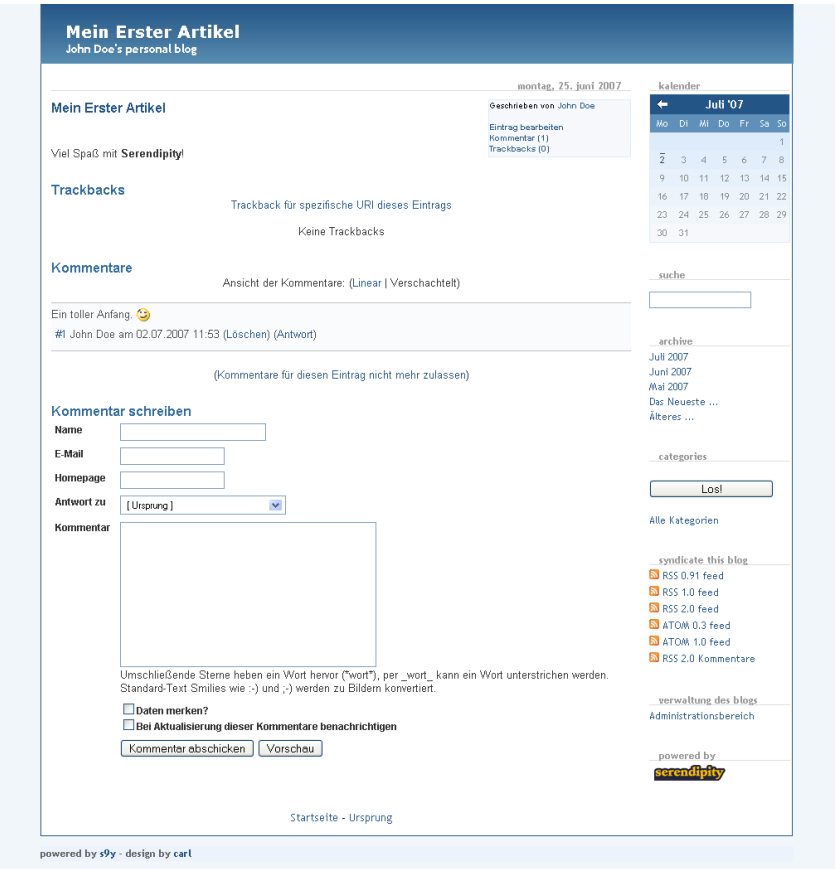

Abbildung 3.4: Artikel-Ansicht mit Kommentardarstellung

Zu jedem Kommentar wird der Name des Kommentatoren dargestellt, sowie (falls von ihm angegeben) seine E-Mail-Adresse und Homepage.

Nach der Auflistung der vorhandenen Kommentare kann ein eigener, neuer Kommentar eingefügt werden. In das Eingabeformular trägt man dazu seine Daten (Name, E-Mail, Homepage, Bezug) sowie den gewünschten Kommentartext ein. Unterhalb der Eingabebox befindet sich ein Hinweis, welche Textformatierungsmöglichkeiten es gibt, um gewisse Dinge hervorzuheben.

Die Auswahlbox **Daten merken?** kann genutzt werden, damit der eingetragene Name in Zukunft vom Browser vorausgefüllt wird. Die Box **Bei Aktualisierung dieser Kommentare benachrichtigen** bestimmt, ob man bei einem neuen Kommentar zu dem Beitrag per E-Mail informiert werden soll.

Um einen Kommentar vor Übermittlung zu überprüfen, kann man den Button **Vorschau** benutzen. Die Folgeseite zeigt den eigenen Kommentar dann so innerhalb der Webseite, wie

er danach für alle Besucher erscheinen sollte. Gerade wenn Sie Sonderzeichen oder HTML-Code in Ihrem Kommentartext benutzen, dient die Vorschau der Kontrolle, ob später auch wirklich alles angezeigt werden kann.

Nach Abschicken eines Kommentars kann es je nach Einstellung des Artikels zu einer Moderation kommen – in diesem Fall muss ein Redakteur einen Kommentar erst autorisieren, bevor er angezeigt wird (siehe Seite 114).

3.4.2 Kommentarübersichten

Neben der gezielten Ansicht von Kommentaren zu einem Artikel gibt es auch eine Übersichtsseite, auf der alle Kommentare unabhängig vom Artikel dargestellt werden.

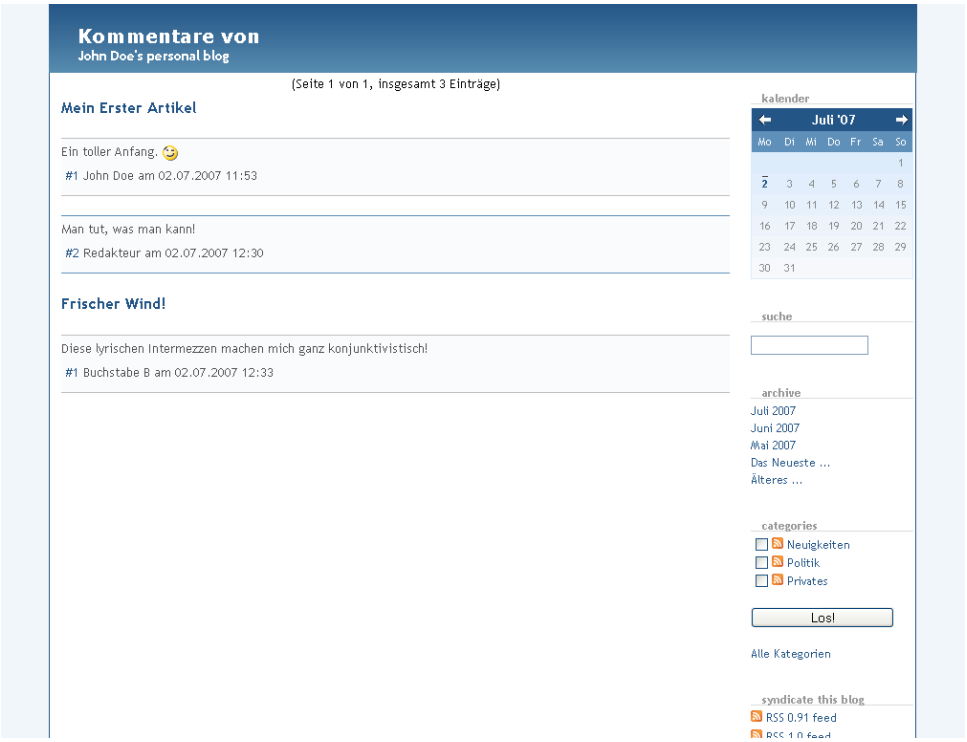

Abbildung 3.5: Kommentarübersicht

Dies eignet sich besonders, um zu sehen, zu welchen Einträgen bestimmte Personen kommentiert haben. Die Seite zeigt dabei wie üblich seitenweise blätterbar die Überschrift eines Artikels an, und darunter die getätigten Kommentare.

Diese Übersichtsseite ist bisher in wenigen Templates eingebaut und wird daher von Besuchern auch seltener genutzt.

Das Permalink-Schema sieht dabei wie folgt aus:

`/comments/`*`kommentator/`*
: stellt die Übersicht aller Kommentare dar, die vom Benutzer `kommentator` stammen. Die Kommentatoren werden dabei in der Datenbank nach exaktem Übereinstimmen mit dem Namen in der URL herausgesucht. Ohne Angabe eines solchen Wertes werden die Kommentare aller Benutzer angezeigt.

`/comments/last_`*`5/`*
: stellt die letzten fünf Kommentare zu allen Artikeln dar. Ohne Angabe eines solchen Wertes werden alle Kommentare angezeigt.

`/comments/from_`*`2007-12-01/`*
: stellt alle Kommentare seit dem 01.12.2007 dar. Das Datum muss hierbei der *GNU DATE*-Syntax entsprechen, siehe `http://www.gnu.org/software/tar/manual/html_node/tar_109.html`. Ohne Angabe dieses Wertes wird keine minimale Zeitbeschränkung gesetzt.

`/comments/to_`*`2007-12-31/`*
: stellt alle Kommentare dar, die bis zum 31.12.2007 gemacht wurden. Das Datum muss ebenfalls der *GNU DATE*-Syntax entsprechen. Ohne Angabe dieses Wertes wird keine maximale Zeitbeschränkung gesetzt.

`/comments/trackbacks/`
: stellt anstelle einer Übersicht zu Kommentaren die Übersicht der Trackbacks dar.

`/comments/comments_and_trackbacks/`
: stellt sowohl Kommentare als auch Trackbacks dar.

`/comments/comments/`
: stellt nur Kommentare dar. Da die Variable dem Standard entspricht, kann man sie üblicherweise weglassen.

`/comments/P`*`2`*`.html`
: stellt die zweite Seite einer Übersicht dar. Die Pfadkomponente `P` (für *Page*) wird gefolgt von einer Seitennummer.

Auch bei diesem Schema lassen sich die Pfadkomponenten miteinander kombinieren, um besondere Ansichten zu erhalten:

`/comments/garvin/last_5`
: stellt die Übersicht der letzten fünf Kommentare des Benutzers `garvin` dar.

`/comments/from_2007-01-01/to_2007-12-31/`
: stellt die Übersicht aller Kommentare vom 01.01. bis zu 31.12.2007 dar.

`/comments/comments_and_trackbacks/from_2007-01-01/P2/`
: stellt die zweite Übersichtsseite aller Kommentare und Trackbacks seit dem 01.01.2007 dar.

3.5 Seite nicht gefunden (404)

Immer wenn man eine URL des Serendipity-Blogs aufruft, die zu einer ungültigen Seite führt, wird statt einer Fehlerseite die Standard-Übersicht angezeigt. Selbstverständlich wird für Suchroboter und Ähnliches dennoch eine HTTP-404-Statusmeldung ausgegeben, um die Ungültigkeit der angeforderten Seite herauszustellen.

Die Standard-Übersicht wird dargestellt, damit Besucher des Blogs dazu motiviert werden, sich über die restlichen Inhalte des Blogs schnell einen Überblick zu verschaffen, anstatt nur eine leere, nichtssagende Fehlermeldung zu sehen.

3.6 RSS-Feeds

Streng genommen zählen RSS-Feeds nicht zum eigentlichen Frontend, da ein RSS-Feed außerhalb von Serendipity angezeigt wird. Ein RSS-Feed enthält, wie eingangs im Zusammenhang der Terminologie erwähnt (siehe Seite 38), einige XML-Elemente, die die aktuellsten Einträge des Blogs mit einigen zusätzlichen Metadaten aufführen.

Folgende RSS-Versionen sind verfügbar:

`/feeds/index.rss2`
: Aktuelle Beiträge in RSS-Version 2.0. Diese Version wird aufgrund ihrer Vollständigheit bevorzugt, da sie die meisten Möglichkeiten bietet und den höchsten Verbreitungsgrad besitzt.

`/feeds/index.rss`
: Aktuelle Beiträge in RSS-Version 0.91.

`/feeds/index.rss1`
: Aktuelle Beiträge in RSS-Version 1.0.

`/feeds/atom03.xml`
: Aktuelle Beiträge in Atom-Version 0.3 (veraltet).

`/feeds/atom10.xml`
: Aktuelle Beiträge in Atom-Version 1.0. Diese Version liegt in ihren Möglichkeiten ungefähr gleichauf mit RSS 2.0 und wird in Zukunft sicherlich noch an Bedeutung gewinnen, da dieser Atom-Standard vom W3C-Gremium[1] empfohlen wird.

`/feeds/index.opml`
: Aktuelle Beiträge im OPML-Format. Dies beinhaltet nur Artikelübersichten und dient mehr der Übersicht als einem tatsächlichen Feed-Format.

[1] Das W3C-Gremium entscheidet in einer großen Arbeitsgruppe über die Standardisierung von Internet-Formaten wie HTML und anderem.

Es gilt zu beachten, dass die URLs für die RSS-Feeds von Besuchern immer aufgerufen werden können, selbst wenn Sie das Seitenleisten-Plugin nicht aktiviert haben. Daher ist es auch grundsätzlich irrelevant, welche Versionen Sie persönlich bevorzugen, da Ihre Besucher die notwendige Version eigenständig beziehen können.

Neben den Einträgen können auch aktuelle Kommentare und Trackbacks zu einem Blog bezogen werden:

`/feeds/comments.rss2`
: Kommentare im RSS-2.0-Format.

`/feeds/trackbacks.rss2`
: Trackbacks im RSS-2.0-Format.

`/feeds/comments_and_trackbacks.rss2`
: Kommentare und Trackbacks im RSS-2.0-Format.

Diese Feeds können auch durch Änderung der Dateiendung in einem anderen Format ausgegeben werden: `.rss2` für RSS 2.0, `.rss1` für RSS 1.0, `.rss` für RSS 0.91 und `.atom` für Atom 1.0.

Um nur einen Feed von einer speziellen Kategorie oder einem Autor abzurufen, sind folgende URLs möglich (jeweils auch mit oben genannten Dateiendungen):

`/feeds/categories/`*`Kategoriename.rss2`*
: Aktuelle Beiträge der Kategorie *`Kategoriename`*.

`/feeds/authors/`*`Autorname.rss2`*
: Aktuelle Beiträge des Autors *`Autorname`*.

RSS-Feeds liegen von Serendipity in mehreren Versionen vor und werden allesamt auf die Ausgabedatei `rss.php` umgeleitet. Diese Datei stellt den Feed dar und kann einige optionale Parameter aufnehmen. Ein kleiner Teil dieser Optionen wird durch Einstellungen des Seitenleisten-Plugins „Blog abonnieren“ beeinflusst, die ab Seite 207 aufgeführt sind.

`rss.php` kann somit auch direkt von einem Browser aufgerufen werden, da sie übersichtlichere Paramter ermöglicht. Diese Parameter können jeweils miteinander verbunden werden. Um mehrere Parameter anzugeben, ruft man die URL mittels `rss.php?parameter1=wert1&parameter2=parameter3&...` auf. Bitte beachten Sie, dass Sie aus technischen Gründen die folgenden Parameter ausschließlich beim Aufruf der `rss.php`-Datei anhängen können, nicht aber beim Aufruf der sprechenden Feed-URLs wie `/feeds/index.rss2`.

`rss.php?version=`*`version`*
: wobei *version* folgende Werte haben kann: *0.91*, *1.0*, *2.0*, *atom0.3*, *atom1.0* und *opml1.0*. Individuelle Feeds, die von Ihnen erstellt werden können (siehe Seite 490), werden ebenfalls einer eindeutigen Version zugeteilt und können durch diesen Parameter später aufgerufen werden. Ohne Angabe der Version wird das RSS-2.0-Format gewählt.

`rss.php?category=1`
: wobei die Zahl der jeweiligen Nummer einer Kategorie des Blogs entspricht. Um einen RSS-Feed für Einträge mehrerer Kategorien zu bündeln, können die Kategorie-Nummern mittels Semikolon (;) hintereinander aufgeführt werden. Ohne Angabe werden Artikel aller Kategorien dargestellt.

`rss.php?viewAuthor=1`
: wobei die Zahl der jeweiligen Nummer eines Autors des Blogs entspricht. Um einen RSS-Feed für Einträge mehrerer Autoren zu bündeln, können die Autor-Nummern mittels Semikolon (;) hintereinander aufgeführt werden. Ohne Angabe werden Artikel aller Autoren dargestellt.

`rss.php?type=typ`
: Anstelle von *typ* kann *comments* (Kommentare), *content* (Beiträge), *trackbacks* (Trackbacks) und *comments_and_trackbacks* (Kommentare und Trackbacks) eingetragen werden, was dafür sorgt, dass nur Artikel des gewünschten Typs im RSS-Feed eingebunden werden. Ohne Angabe werden nur Artikel angezeigt.

`rss.php?nocache=true`
: Diese Variable sorgt dafür, dass *Conditional GET* für RSS-Feeds deaktiviert wird (siehe 3.6 auf Seite 87).

`rss.php?cid=Artikel-ID`
: Falls als Inhaltstyp eines RSS-Feeds ein anderer als *content* gewählt wurde, kann man die Anzeige der Kommentare und Trackbacks auf einen bestimmten Artikel einschränken. Die Nummer (*ID*) dieses Artikels wird als Wert des Parameters angegeben. Solche RSS-Feeds werden oft verwendet, damit Besucher die Kommentare zu einem Artikel verfolgen können, an dessen Diskussion sie teilgenommen haben.

`rss.php?all=true`
: Falls dieser Parameter gesetzt wird, enthält der RSS-Feed nicht nur die letzten 15 aktuellen Artikel, sondern alle verfügbaren. Der RSS-Feed kann dadurch sehr groß werden, daher ist die Verwendung dieses Parameters nur für den Export der Artikeldatenbank gedacht.

`rss.php?fullFeed=true`
: Falls dieser Parameter gesetzt ist, enthält der RSS-Feed den vollständigen Artikel (anstelle nur des Teasers). Diese Option muss vom Betreiber des Blogs jedoch gezielt aktiviert werden, da durch die Aktivierung der „Diebstahl“ von Artikeln stark vereinfacht wird (siehe Seite 207).

`rss.php?forceLocal=true`
: Das Plugin „Blog abonnieren“ unterstützt die Möglichkeit, dass RSS-Feeds zu einem Dienstleister namens *Feedburner* weitergeleitet werden. In so einem Fall würde der Aufruf der eigenen RSS-URLs immer direkt zu der Seite des Dienstleisters führen und man hätte keine Möglichkeit mehr, den *echten* RSS-Feed aufzurufen. Sollte diese

Umleitungsoption also gesetzt sein, können Sie mittels des Parameters *forceLocal* dafür sorgen, dass Sie *nicht* umgeleitet werden. Der Parameter zeigt jedoch nur Wirkung, wenn Sie als Redakteur am Blog angemeldet sind.

3.6.1 Caching von RSS-Feeds

Da RSS-Feeds von den RSS-Readern Ihrer Besucher relativ häufig aufgerufen werden (meist halbstündlich), ist es wichtig, dort so wenig wie möglich Redundanz für die Datenübertragung zu verursachen.

Bei 200 Abonnenten Ihres RSS-Feeds und einem größeren Artikelbestand kommt es durchaus vor, dass die XML-Datei des Feeds 100kb und größer ist. Bei dieser Benutzerzahl würden Sie pro Stunde 4MB (das sind im Monat gut 3GB!) an Daten übertragen. Wenn Sie durchschnittlich nur alle drei Tage einen neuen Artikel schreiben, wäre ein Großteil dieser Datenmenge unnötig übertragen worden.

Um dieses Problem zu lösen, wurde im HTTP-Standard ein Caching[2]-Mechanismus vorgesehen. Dabei überträgt der RSS-Reader (und auch jeder Webbrowser) den Zeitpunkt des letzten Abrufs an den Server. Der Server vergleicht, ob sich seit diesem Zeitpunkt etwas am RSS-Feed geändert hat. Falls das nicht der Fall ist, wird eine leere Antwort an die Software zurückgeschickt, und es werden keine Daten übertragen.

Wenn sich etwas verändert hat, kann der Server lediglich die seit dem letzten Aufruf erneuerten Daten übermitteln und spart so nochmals an Übertragungszeit (und damit auch Server-Performance).

Dieser Vorgang wird als *Conditional GET* bezeichnet und ist in Form einer verbindlichen *RFC 2616* (Richtlinie des HTT-Protokolls) festgehalten. Dennoch gibt es leider einige Clients, die das Caching nicht vollständig implementieren und mit derartig ausgelieferten RSS-Feeds Probleme haben. Solche Probleme können sich darin äußern, dass ein RSS-Reader ständig bereits bekannte Einträge des RSS-Feeds als neu ansieht oder womöglich gar keine neuen Artikel empfangen kann.

Um bei solchen *kaputten* Clients dennoch einen Serendipity-Feed korrekt abrufen zu können, ist der eingangs erwähnte Parameter `nocache=true` an den Aufruf der `rss.php`-URL anzuhängen.

Im Zusammenhang mit dem globalen Caching unterstützt Serendipity die Auslieferung von Artikeln ab einem bestimmten Datum. Da üblicherweise RSS-Feeds nur die letzten 15 Artikel enthalten und man möglicherweise einmal für eine Weile den RSS-Feed nicht empfangen kann[3], könnte es also passieren, dass man Artikel schlichtweg verpasst. Bei der Übermittlung einer Angabe des zuletzt gesehenen Artikels wird dies vermieden, und Serendipity würde in so einem Fall auch alle weiteren Artikel über dem 15. ausliefern.

[2] Ein Cache ist ein Zwischenspeicher oder Puffer, der einmal erstellte Daten erneut ausgeben kann, anstatt sie neu zusammenzustellen.

[3] Vielleicht hat man ja einmal sein Notebook oder die UMTS-Karte im Urlaub vergessen . . . ;-)

Diese Auslegung des RFC-Standards verursacht in manchen RSS-Readern und vor allem Aggregatoren wie der Software *Planet* jedoch Probleme. In der Konfiguration des Blogs wird daher die Option angeboten, das *strikte RFC2616* zu veranlassen (siehe Seite 166). Zwei weitere Variablen (siehe Seite 179) dienen darüber hinaus der Feineinstellung von Caching-Inhalten.

Sobald diese Option aktiviert ist, wird Serendipity, wie andere Software auch, stur nur die letzten 15 Artikel ausliefern und somit das Feature der *Urlaubsschaltung* deaktivieren.

3.7 Suche

Der Permalink für die Suche folgt dem Schema `http://www.example.com/serendipity/search/Su`
Mehrere Begriffe werden dabei durch den Pfadtrenner `/` getrennt. Die Pfadkomponente `P` für die gewünschte Seite der Übersicht kann angehängt werden.

Volltextsuche wird bei Einsatz der MySQL-Datenbank mit dem „Fulltext“-Suchmechanismus ausgeführt. Dieser Mechanismus funktioniert nur bei MySQL-Versionen ab 4.0, und es ist erforderlich, dass ein `FULLTEXT INDEX` auf die Spalten `title,body,extended` der Tabelle `serendipity_entries` erstellt wurde. Dies erledigt die Serendipity-Installationsroutine üblicherweise. Bei manchen Providern (besonders beim Einsatz von Confixx) kann es jedoch passieren, dass Ihr Datenbankbenutzer keine `INDEX`-Rechte hat. In einem solchen Fall würden Sie bei der Suche nach Artikeln eine Fehlermeldung erhalten, die Sie dazu anleitet, den Index korrekt zu erstellen. Notfalls müssen Sie Ihren Provider bitten, Ihnen die notwendigen Rechte dafür einzuräumen.

MySQL unterstützt die Suche mittels *BOOLEAN*-Operatoren. Hier können Sie Wörter mit einem Anführungszeichen einschließen und ein + oder - vorstellen, um Teilwörter ein- oder auszuschließen. Auch kann das Zeichen * als Platzhalter für beliebige Zeichen verwendet werden. Details zu der MySQL-Volltextsuche schlagen Sie bitte in der MySQL-Dokumentation nach.

Abbildung 3.6: Volltextsuche

Zwei wichtige Besonderheiten beim Einsatz von MySQL gibt es noch zu beachten: Die Suche funktioniert nur, wenn die Suchergebnisse nicht einen Großteil aller vorhandenen Einträge ausgeben. Damit verhindert MySQL, dass eine nutzlose Suche alle Einträge zurückliefern würde. Gerade wenn Sie also erst zwei Artikel geschrieben haben und nach einem Begriff suchen, wundern Sie sich nicht, wenn Sie keine Ergebnisse erhalten. Sobald die Datenbank eine vernünftige Datenbasis enthält, wird die Suche wie gewünscht funktionieren.

Die zweite Besonderheit ist, dass MySQL nur Wörter mit standardmäßig mindestens drei Buchstaben findet und dass ein Wort kein *Stoppwort* wie *und* enthält. Eine Suche nach *DB* würde also nicht funktionieren. Dies wird durch die MySQL-Option `ft_max_word_len` (siehe MySQL-Dokumentation) definiert.

Beim Einsatz von SQLite oder PostgreSQL wird die Suche mittels einer Zeichenteilkette durchgeführt. Eine Suche nach *risch* würde also sowohl Artikel mit dem Wort *Frischfleisch* als auch *betrügerisch* finden. Diese ist zwar langsamer als die erwähnten MySQL-Möglichkeiten, benötigt dafür aber weniger Rahmenbedingungen, um zu funktionieren.

Kapitel 4

Backend

Nachdem Sie nun über die möglichen Ansichten von Serendipity im Besucherbereich Bescheid wissen, ist es an der Zeit, das Backend zu erkunden.

4.1 Login

Das Backend ist geschützt durch einen Login. Sie erreichen die Administrationsoberfläche unter

```
http://www.example.com/serendipity/serendipity_admin.php
```

Auf dieser Seite müssen Sie Ihren Benutzernamen und das Passwort eintragen, das Sie bei der Installation angegeben haben. Standardmäßig ist der Benutzername `John Doe` und das Passwort `john`.

Unterhalb der beiden Eingabefelder befindet sich die Auswahlbox **Daten speichern**. Wenn Sie diese Option aktivieren, wird auf Ihrem Computer ein Cookie gespeichert, der Sie eindeutig im System identifiziert, und Sie können sich in Zukunft ohne Passwort einloggen.

Dieser Cookie enthält eine eindeutige Kennzeichnung sowie ein Ablaufdatum, Ihr Benutzername oder Passwort wird nicht gespeichert. Wenn das Ablaufdatum überschritten ist, wird der alte Cookie automatisch als ungültig markiert und ein neuer Cookie ausgestellt. Diese Funktionsweise stellt sicher, dass ein Angreifer aus einem gestohlenen Cookie keine Nutzerinformationen extrahieren und mit einem veralteten Cookie keinen Login ausführen kann.

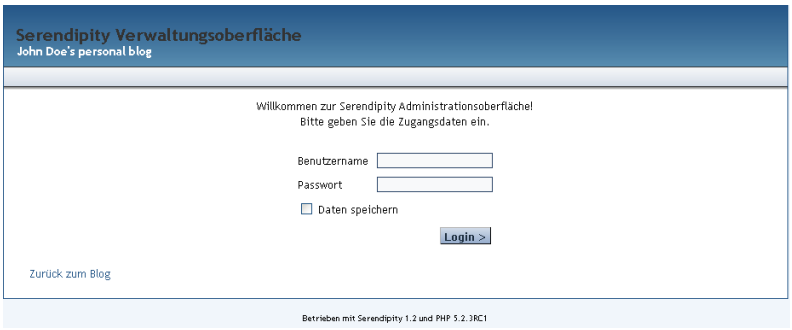

Abbildung 4.1: Login-Bildschirm

Trotz dieser Sicherungsmethoden sollten Sie die Option **Daten speichern** niemals auf einem Computer aktivieren, zu dem auch weniger vertrauenswürdige Benutzer Zugriff haben. Trotz aller Bequemlichkeit ist diese Option vor allem in Internetcafes tabu.

Sofern Ihr Server die *HTTPS*-Verschlüsselung unterstützt, sollten Sie die Administrationsoberfläche immer über `https://...` aufrufen. Erst diese Verschlüsselung schützt sie vor dem Ablauschen Ihrer Logindaten während der Übermittlung.

4.1.1 Mögliche Fehler beim Login

Das häufigste Problem beim Login ist ein PHP-Server, bei dem die Sessions nicht, wie im Kapitel 1.3 auf Seite 26 beschrieben, korrekt eingerichtet sind. Dies kann sich darin äußern, dass Sie sich nach jedem Klick neu einloggen müssen.

Ältere Serendipity-Versionen verursachten zudem ähnliche Probleme, wenn die URL `http://localhost/..` lautete. Viele Browser akzeptieren keine Login-Cookies, wenn der Servername nicht mindestens zwei Punkte enthält. Auch sollten Sie generell prüfen, ob Ihr Browser möglicherweise Cookies ablehnt oder eine Antiviren-Software auf Ihrem Rechner Session-Cookies blockiert.

Auch Umlaute im Passwort oder Benutzernamen können dazu führen, dass Ihr Passwort nicht korrekt abgeglichen werden kann. In manchen Fällen kann es helfen, wenn Sie in Ihrem Browser alle Cookies löschen, die in Verbindung mit Ihrer Blog-URL gespeichert sind, dann den Browser neu starten und sich so neu einloggen können.

4.1.2 Passwort vergessen

Serendipity speichert Ihr Passwort aus Sicherheitsgründen nicht im Klartext in der Datenbank ab. Sollte man daher sein Passwort vergessen haben, kann man dieses nicht einfach irgendwo nachschlagen, sondern muss das Passwort neu setzen.

Da Serendipity keine Umgehungsmechanismen des Logins (in Ihrem Interesse) ermöglicht, ist ein Neusetzen des Passwortes nur über die Datenbank möglich.

Dazu muss ein Tool wie phpMyAdmin benutzt werden, um die Tabelle `serendipity_authors` zu bearbeiten. Diese Tabelle enthält die Benutzernamen und Passwörter aller angelegten Benutzer im System – nach der Installation ist dies üblicherweise nur der Hauptbenutzer.

Um nun ein Passwort für diesen Benutzer neu zu setzen, muss man sich den Inhalt der Tabelle anzeigen lassen (phpMyAdmin: **Datensätze anzeigen**) und die Zeile bearbeiten (phpMyAdmin: Klick auf das Stift-Icon). In der Bearbeitungsmaske dieses Datensatzes findet man das Feld `password`, in das man das neue, gültige Passwort einsetzen muss. Dabei ist zu beachten, dass für das Passwort ein sogenannter `MD5-Hash`[1] eingetragen werden muss, und nicht das Passwort im Klartext. MySQL bietet hierfür eine Spaltenfunktion namens **MD5** an, die man auswählt und dann das gewünschte Passwort in den Inhaltsbereich einträgt – beim Speichern wird MySQL dann die Funktion `MD5` anwenden und den Klartext verschlüsseln. Alternativ kann man einen MD5-Hash für das gewünschte Passwort auch mit zahlreichen Online-Generatoren im Internet erzeugen und mittels der Zwischenablage in den Datensatz einfügen.

Alle anderen Spalten des Datensatzes sollten Sie unverändert lassen.

Sollte Ihnen dies alles zu komplex sein, können Sie stattdessen auch den folgenden Code in Ihrem Serendipity-Verzeichnis als `fixpass.php` speichern und per `http://www.example.com/serendipity/fixp` aufrufen:

```
<?php
include 'serendipity_config.inc.php';
$authorid = 1;
$neues_passwort = 'neues_passwort';

if ($authorid > 0) {
    serendipity_db_query("UPDATE {$serendipity['dbPrefix']}authors
                             SET password = '". md5($neues_passwort)."'
                           WHERE authorid = ". $author_id);
} else {
    serendipity_db_query("UPDATE {$serendipity['dbPrefix']}authors
                             SET password = '".md5($neues_passwort)."'");
}
?>
```

Das Script enthält zwei Variablen: `$authorid` und `$neues_passwort`. Diese Variablen müssen Sie auf Ihre Gegebenheiten anpassen. Tragen Sie in die Variable `$neues_passwort` das neue Passwort (in Anführungszeichen gesetzt) ein. Die Variable `$authorid` müssen Sie entweder auf die ID des Autors[2] setzen, oder Sie tragen dort eine `0` ein; dadurch werden die Passwörter *aller* Benutzer auf das neue Passwort gesetzt.

[1] Ein MD5-Hash ist ein Einwegmechanismus, der eine Zeichenkette analysiert und eine spezielle Prüfsumme dazu erzeugt. Diese Prüfsumme ist für alle Zeichenketten reproduzierbar, aber von der Prüfsumme kann nicht auf die Originalzeichenkette zurückgeschlossen werden.

[2] Diese entspricht von 1 an aufwärts zählend der Anlegungsreihenfolge der Autoren; Sie können sie mittels phpMyAdmin oder Ähnlichem in der Datenbanktabelle `serendipity_authors` nachschauen.

4.1.3 XSRF/CSRF-Schutz

Bei allen Zugriffen auf das Backend führt Serendipity eine Prüfung durch, ob Sie berechtigt sind, die gewünschte Aktion (das Löschen von Beiträgen etc.) durchzuführen.

Eine im Internet derzeit verbreitete Angriffsart ist die sogenannte *Cross-Site Request Forgery* (*XSRF, CSRF*). Dabei enthält eine fremde Seite ein verstecktes JavaScript. Wenn der Angreifer die Adresse Ihres Blogs kennt und Sie die fremde Webseite besuchen, könnte diese über das versteckte JavaScript administrative Aktionen in Ihrem Blog in Ihrem Namen ausführen. Sozusagen eine bösartige, von Ihnen ungewollte Fernsteuerung Ihres Blogs. Denn für Ihr Blog erscheint es so, als würden Sie selbst die Aktion bewusst ausführen.

Um derartigem Missbrauch vorzubeugen, setzt Serendipity zweierlei Methoden ein.

Die wirkungsvollste ist, dass jede Aktion im Backend durch einen sogenannten *Token* geschützt wird. Dieser Token wird anhand Ihrer Logindaten automatisch und zufallsgeneriert erstellt, daher kann er von böswilligen Angreifern nicht ohne Eindringen in Ihren eigenen Rechner (oder die Netzwerkverbindung) ausgelesen werden. Wenn eine fremde Webseite also Ihren Browser fernsteuern möchte, scheitert die Aktion daran, dass der gültige *Token* fehlt.

Eine derartige Aktion quittiert Serendipity dann mit der Meldung:

```
Ihr Browser hat keinen gültigen HTTP-Referrer übermittelt. Dies kann
entweder daher kommen, dass Ihr Browser/Proxy nicht korrekt konfiguriert
ist, oder dass Sie Opfer einer Cross Site Request Forgery (XSRF)
waren, mit der man Sie zu ungewollten Änderungen zwingen wollte. Die
angeforderte Aktion konnte daher nicht durchgeführt werden.
```

Wenn Sie diese Meldung erhalten, obwohl Sie eine Aktion veranlasst haben, kann dies darauf hinweisen, dass Ihr Browser den Token nicht korrekt übermittelt hat. Ein Token wird in einem Browser-Cookie gespeichert, daher muss Ihr Browser Cookies zwingend akzeptieren. Weiterhin kann die Fehlermeldung erscheinen, wenn Sie Ihr Backend nicht unter der URL aufrufen, unter der Sie sich eingeloggt haben[3]. Im Zweifelsfall kann es helfen, wenn Sie sich komplett aus dem Backend ausloggen, im Browser sämtliche Cookies des Blogs löschen und sich neu einloggen.

Die zweite Sicherheitsmethode besteht darin, dass Serendipity den *HTTP-Referrer* prüft. Bei jedem Aufruf, den Ihr Browser tätigt, wird an den Server die URL der vorausgehenden Webseite übermittelt und in einer sogenannten HTTP-Kopfzeile namens *Referrer* (*Verweis*) gespeichert. Anhand dieses Wertes kann Serendipity erkennen, ob Sie vor dem Aufruf Ihres Serendipity-Backends möglicherweise eine fremde Webseite besucht haben. Im regulären Betrieb von Serendipity kann es niemals vorkommen, dass Sie eine administrative Aktion ausführen, ohne dass der HTTP-Referrer Ihrer Blog-URL entspricht.

Wenn dieser Fall eintritt, gibt Serendipity ebenfalls die oben genannte Fehlermeldung aus. Da die *HTTP-Referrer*-Kopfzeile jedoch laut Spezifikation kein Pflichtfeld ist und auch gewisse

[3] Beispielsweise auch, wenn Sie sich über `https://www.example.com/serendipity/` eingeloggt haben, aber über `http://example.com/serendipity/` eine Aktion aufrufen.

private Informationen über Sie ausliefert, können Sie in vielen Browsern die Übermittlung dieser Variable unterdrücken. Dann kann keine Webseite herausfinden, welche Seite Sie vorher besucht haben – und auch Serendipity kann dies nicht mehr erkennen.

Am einfachsten ist es, dass Sie in Ihrem Browser die Übermittlung des HTTP-Referrers für Ihr Serendipity-Blog aktivieren. Wenn dies nicht möglich ist, können Sie Serendipity nach wie vor betreiben, werden jedoch die oben genannte Warnmeldung sehen.

Im Gegensatz zur Sicherheitsprüfung durch den *Token* wird die Prüfung des *HTTP-Referrers* jedoch standardmäßig die auszuführende Aktion *nicht* abbrechen, damit der Betrieb von Serendipity auch mit derartig konfigurierten Browsern funktionieren kann. Wenn Sie in Ihrem Browser die Übermittlung des HTTP-Referrers zulassen, können Sie als weitere Sicherheitsoption die globale Variable `$serendipity['referrerXSRF']` aktivieren – in so einem Fall wird eine Aktion im Backend abgebrochen, wenn der HTTP-Referrer fehlt (siehe Seite 175).

4.2 Übersicht

Nach dem Login befinden Sie sich auf der Übersichtsseite des Backends.

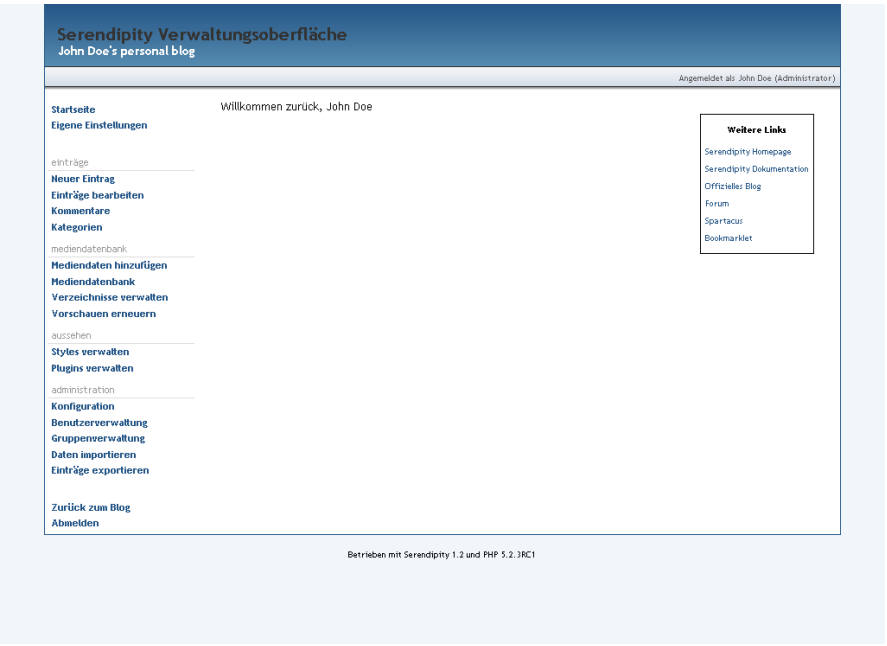

Abbildung 4.2: Verwaltungsoberfläche

Der Kopfbereich zeigt dabei den Namen des Blogs an und in einer kleinen Info-Zeile darunter den Namen und Benutzerrang des aktuell eingeloggten Benutzers.

Im Menü der linken Seite sehen Sie als Administrator alle Untermenüpunkte, mit denen sich Serendipity bedienen lässt. Auf der rechten Seite sind einige Links aufgeführt, mit denen Sie zu speziellen weiterführenden Seiten rund um Serendipity geführt werden.

Der eigentliche Inhaltsbereich gibt nur einen kleinen Begrüßungstext aus, und der Fußbereich der Seite stellt die aktuelle Versionsnummer von Serendipity und PHP dar.

Das Menü selbst ist in mehrere Abschnitte unterteilt:

- Ein Anfangsblock führt zur Übersichtsseite sowie zu den Einstellungen des aktuellen Benutzers.
- **Einträge** bietet Möglichkeiten zum Bearbeiten von Artikeln.
- **Mediendatenbank** gibt Zugriff auf hochgeladene Dateien.
- **Aussehen** verwaltet das Template und die eingesetzten Plugins des Blogs.
- **Administration** umfasst alle Funktionen, die den Zugriff auf das Blog und dessen Daten regulieren.
- Der abschließende Block enthält die Links **Zurück zum Blog**, um in das Frontend zu gelangen, und den Menüpunkt **Abmelden**, um sich aus dem Backend vollständig abzumelden.

Einige Plugins geben zudem an einigen Stellen im Menü ihre eigenen Menüpunkte aus. Diese zusätzlichen Menüpunkte führen dann zur jeweiligen Funktionalität eines Plugins.

4.3 Eigene Einstellungen

Die **Eigenen Einstellungen** ermöglichen den Zugriff auf alle Einstellungen, die lediglich für den aktuellen Redakteur von Belang sind. Jeder Benutzer kann somit seine eigenen, individuellen Einstellungen in diesem Bereich vornehmen.

Der Bereich ist zweigeteilt in **Persönliche Einstellungen** und **Voreinstellungen für neue Einträge**. Im ersten Abschnitt legen Sie, wie bei der Installation bereits geschehen, den Benutzernamen, das Passwort und den Loginnamen fest.

Damit später Ihr Passwort nur von autorisierten Personen geändert werden kann, muss für dessen Änderung auch stets das alte Passwort angegeben werden. Wenn Sie das Passwort nicht ändern wollen, tragen Sie daher in dem Eingabefeld **Passwort** nichts ein. Beachten Sie bitte, dass einige Browser (z. B. Firefox) die sogenannte Automatische Vervollständigung für Passwort-Felder unterstützen und möglicherweise daher automatisch das zuletzt verwendete Passwort im dafür vorgesehenen Feld eintragen. Sollten Sie also bereits vorausgefüllte Sternchen in dem Eingabefeld vorfinden, hat diese Ihr Browser eingefügt und nicht Serendipity.

Würden Sie nun direkt auf den Button **Speichern** am Ende der Seite klicken, bekämen Sie eine Fehlermeldung von Serendipity, da Ihr Browser das Feld **Altes Passwort** nicht eingetragen hat.

Nach diesen Login-relevanten Feldern folgen weitere Einstellungen:

E-Mail

Hier tragen Sie Ihre E-Mail-Adresse ein. Diese wird von Serendipity an mehreren Stellen verwendet, um Sie über neu eingegangene Kommentare zu benachrichtigen oder auch Administratoren des Blogs über den Zugang neuer Benutzer zu informieren. Geben Sie die Adresse hier im Format `name@example.com` an, also ohne zusätzliche Formatierungen wie `"Ihr Name" <name@example.com>`.

Sprache

Die Übersetzung der Systemtexte des Backends und des Frontends richtet sich nach der Auswahl in diesem Feld. Beachten Sie dabei, dass die Sprache erst nach dem nächsten Klick innerhalb des Backends gewechselt wird und nur für Sie gilt. Die globale Sprache stellen Sie in der **Konfiguration** ein (siehe Abschnitt 4.7, Seite Konfiguration).

Die hier eingestellte Sprache gibt lediglich an, welche Übersetzung Serendipitys aktiviert wird. Sie sind durch die Festlegung dieser Option nicht daran gebunden, Ihre Artikel auch in der gesetzten Sprache zu verfassen.

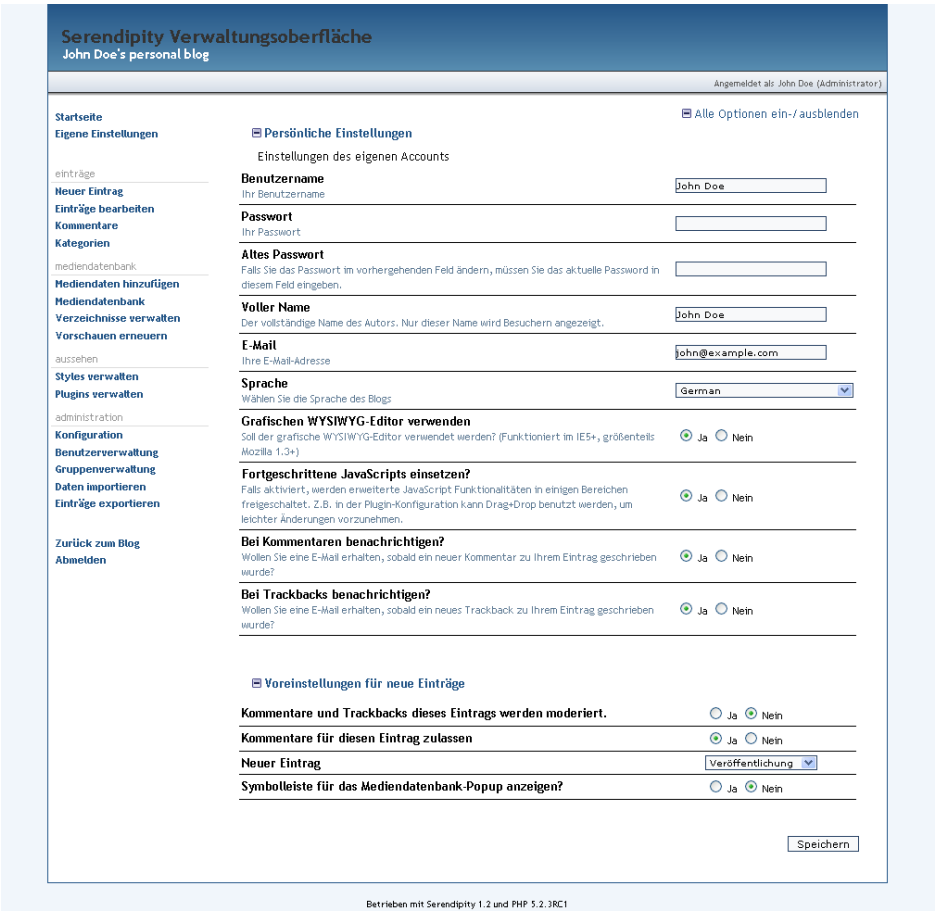

Abbildung 4.3: Eigene Einstellungen

Grafischen WYSIWYG-Editor verwenden

Serendipity liefert einen WYSIWYG-Editor (siehe Abschnitt 1.4.13 auf Seite 42). Dieser Editor stellt beim Erstellen eines Artikels für Ihr Blog erweiterte, einfache Textformatierungen zur Verfügung. So können Sie einen Blog-Artikel wie in einem Office-Programm erstellen und formatieren.

Wenn Sie dies wünschen, aktivieren Sie **Grafischen WYSIWYG-Editor verwenden**. Dieser wird jedoch nur in aktuellen Browsern unterstützt.

WYSIWYG-Editoren funktionieren leider in unterschiedlichen Browsern und je nach Einsatzzweck recht unterschiedlich – so auch die von Serendipity standardmäßig verwendete Komponente *HTMLArea*. Dieser Editor fügt leider gerade bei intensiven Kopier-/Einfüge-Operationen häufig problematische HTML-Konstrukte ein, die die Formatierung in Ihrem Blog durcheinanderbringen könnten. Beispielsweise sind dies falsche

Zeilenabstände, überflüssige Absätze bis hin zur falschen Platzierung von eingebundenen Bildern.

Leider wird Sie nur die Erfahrung lehren, wie ein WYSIWYG-Editor einzusetzen ist, ohne Fehler zu produzieren. Generell gilt die Empfehlung, Textformatierungen nur mit Bedacht einzusetzen und ab und an den WYSIWYG-Editor in den Quelltext-Modus zu versetzen, um zu prüfen, ob sich dort überflüssige, leere HTML-Konstrukte befinden.

Einige Plugins für Serendipity bieten zudem externe WYSIWYG-Editoren wie TinyMCE[4], FCKEditor[5] und Xinha[6] zur Einbindung an, die in ihrem Funktionsumfang durchaus variieren. Auch diese externen Editoren werden nur dann verwendet, wenn die Option **Grafischen WYSIWYG-Editor verwenden** aktiviert ist.

Fortgeschrittene JavaScripts einsetzen?
Grundsätzlich ist Serendipity ohne den Einsatz von JavaScript problemlos zu bedienen, also auch von textbasierten Browsern. Dennoch gibt es einige zusätzliche Features, die AJAX (siehe Seite 45) einsetzen. Wenn die Option **Fortgeschrittene JavaScripts einsetzen** aktiviert ist, werden die zusätzlichen Features aktiviert.

Aktuell wird diese Option nur an zwei Stellen ausgewertet: bei der Plugin-Konfiguration (um Plugins via *Drag and Drop* anzuordnen, siehe Kapitel 4.6.3 auf Seite 145) und für das Beschneiden eines Bildes.

Bei Kommentaren benachrichtigen?
Sobald ein Benutzer zu einem Ihrer Blog-Artikel einen neuen Kommentar verfasst, können Sie per E-Mail benachrichtigt werden.

Bei Trackbacks benachrichtigen?
Sobald ein Benutzer zu einem Ihrer Blog-Artikel ein neues Trackback geschickt hat, können Sie per E-Mail benachrichtigt werden.

Der zweite Abschnitt der **Eigenen Einstellungen** stellt **Voreinstellungen für neue Einträge** ein:

Kommentare und Trackbacks dieses Eintrags werden moderiert.
Diese Option stellt ein, ob bei einem neu erstellten Artikel standardmäßig Kommentare und Trackbacks moderiert werden sollen. Im Falle einer Moderation muss ein neuer Kommentar oder ein Trackback erst freigeschaltet werden, bevor er für Besucher angezeigt wird.

Kommentare für diesen Eintrag zulassen.
Mittels dieser Option wird festgelegt, ob Kommentare zu einem Artikel erlaubt sind.

[4] `http://tinymce.moxiecode.com/`
[5] `http://www.fckeditor.net/`
[6] `http://xinha.python-hosting.com/`

Neuer Eintrag
: Wenn Sie einen neuen Artikel erstellen, kann dieser entweder als **Veröffentlichung** oder **Entwurf** gespeichert werden. Nur veröffentlichte Artikel werden den Besuchern angezeigt, ein Entwurf ist nur für Redakteure sichtbar.

Symbolleiste für das Mediendatenbank-Popup anzeigen?
: Um eine Grafik oder Datei in einen Blog-Artikel einzufügen, kann man ein Popup-Fenster mit der Mediendatenbank Serendipitys öffnen. Da diese Oberfläche in diesem Fall nur zur Einfügung von Objekten benutzt wird, werden einige Optionen der Mediendatenbank (Bearbeiten, Umbenennen von Dateien und Weiteres) der Übersichtlichkeit halber ausgeblendet. Sollten Sie jedoch diese Funktionen auch gerne innerhalb des Popup-Fensters ausführen wollen, müssen Sie die Option **Symbolleiste für das Mediendatenbank-Popup anzeigen** aktivieren.

Um die vorgenommenen Änderungen an den Einstellungen zu speichern, müssen Sie auf den Button **Speichern** am Ende der Seite klicken.

In älteren Versionen von Serendipity befanden sich auf der Seite **Es können keine Einträge mehr veröffentlicht werden** einige weitere spezielle Konfigurationsparameter: **Zugriffsrechte**, **Gruppenzugehörigkeit**, **Rechte: Einträge veröffentlichen** und **Erstellung von Einträgen verbieten**. Diese Optionen sind in neuen Versionen nun in den Bereich **Administration** → **Benutzerverwaltung** verschoben worden, wo sie konzeptionell sinnvoller liegen. Bei Verwendung von Serendipity 1.2 und höher können Sie den folgenden Abschnitt also getrost ignorieren.

Über diese Optionen war es ehemals möglich, seinem eigenen Benutzer wichtige Grundrechte zu entziehen. So konnte es vorkommen, dass ein Administrator sich selbst (versehentlich) als Redakteur verzeichnete oder keine Gruppenmitgliedschaften auswählte. Beide Optionen führten dazu, dass man die Situation nur mittels Eingriff in die Datenbank wieder richten konnte.

Die Option **Rechte: Einträge veröffentlichen** konnte zudem dazu führen, dass sich ein Autor die Möglichkeit entzog, Einträge überhaupt veröffentlichen zu dürfen.

Weitaus schlimmer war die Aktivierung der Option **Erstellung von Einträgen verbieten**. Diese Option ist eigentlich für eingeschränkte Benutzerkonten gedacht, die im Backend überhaupt nichts tun dürfen. Benutzer, bei denen diese Option gesetzt ist, können im Menü außer dem Login, dem Logout und den eigenen Einstellungen auf nichts anderes zugreifen.

Sollte in einer älteren Version diese Fall eintreten, können Sie folgendes PHP-Script in Ihrem Serendipity-Verzeichnis als `fixpriv.php` abspeichern und mittels `http://www.example.com/serendi` aufrufen.

```
<?php
include 'serendipity_config.php';
serendipity_db_query("UPDATE {$serendipity['dbPrefix']}authors
                        SET userlevel     = 255,
                            right_publish = 1
```

```
                        WHERE authorid      = 1");

serendipity_db_query("UPDATE {$serendipity['dbPrefix']}config
                          SET value = 'false'
                        WHERE name  = 'no_create'
                          AND authorid = 1");
?>
```

Dieses Script ändert den Benutzer mit der ID *1* so, dass er wieder als Administrator mit vollen Rechten zählt. Sollten Sie die Rechte eines anderen Autors beheben wollen, müssen Sie in der Datenbanktabelle `serendipity_authors` nach der ID des gewünschten Autors suchen und ihn an den beiden Stellen in der `fixpriv.php` ändern.

4.4 Einträge

Der Bereich **Einträge** enthält sämtliche Menüpunkte, die der Erstellung und dem Bearbeiten von Blog-Artikeln dienen.

4.4.1 Neuer Eintrag

Der Menüpunkt **Neuer Eintrag** stellt nach der Einrichtung des Blogs üblicherweise die meistgenutzte Funktionalität zur Verfügung: die einfache Erstellung eines Artikels. Dahinter verbirgt sich die Eintrags-Maske, die in einem übersichtlich strukturierten Bereich alle Optionen für einen Artikel darstellt.

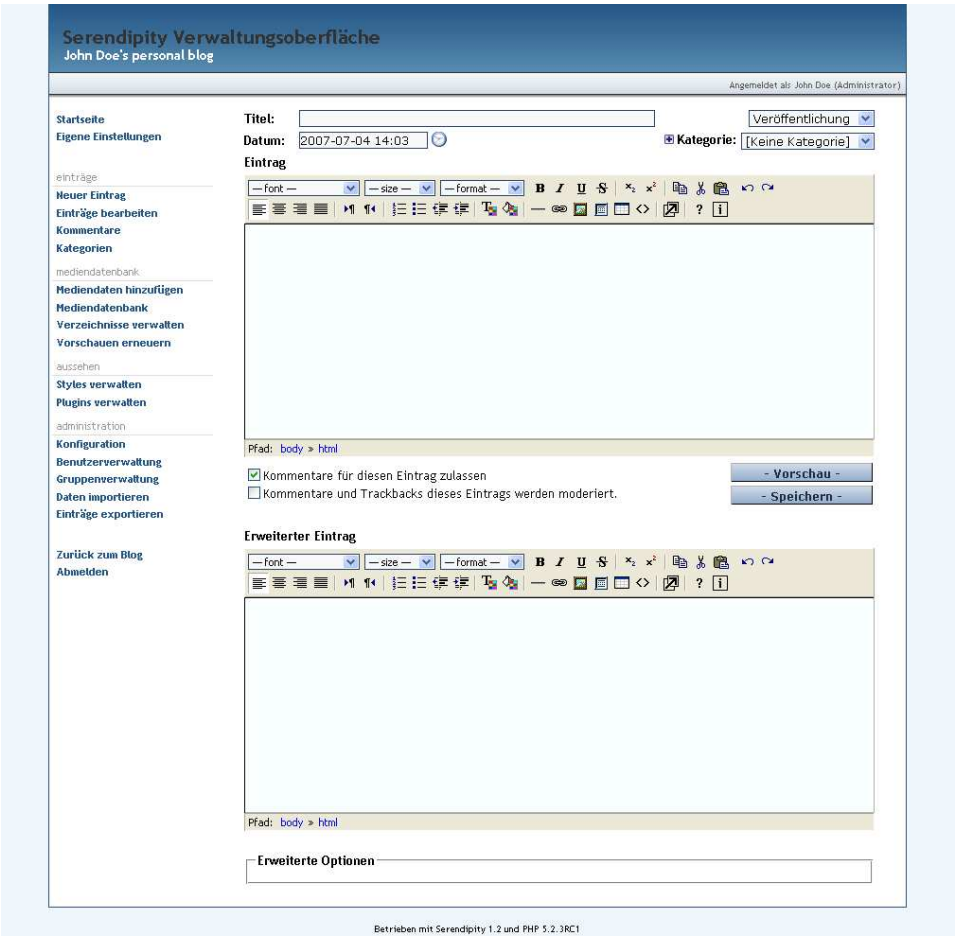

Abbildung 4.4: Einträge: Neuer Eintrag (aktivierter WYSIWYG-Editor)

Die jeweiligen Eingabefelder sind:

Titel

In diesem Feld wird der Titel des Artikels eingetragen, so wie er auch im Frontend dargestellt werden soll. Im Titel können Sonderzeichen und Umlaute enthalten sein, jedoch können keine HTML-Tags dort eingetragen werden.

Artikelmodus

Im Auswahlfeld rechts neben dem Titel kann bestimmt werden, ob der Artikel beim Speichern **veröffentlicht** werden soll oder nur als **Entwurf** abgelegt wird. Die Standardeinstellung richtet sich nach der Festlegung in den **Eigenen Einstellungen**. Je nach den Rechten Ihres Redakteurs kann es sein, dass dieses Auswahlfeld nur die Option

Entwurf zulässt. In diesem Fall muss ein Chefredakteur Ihre Artikel veröffentlichen.

Datum

Das Datum wird in dem Feld unterhalb des Artikeltitels festgelegt. Es muss dabei in einem maschinenlesbaren Format gespeichert werden: *Jahr-Monat-Tag Stunde:Minute*. Um die aktuelle Uhrzeit Ihres Rechners dort einzufügen, können Sie auf das Uhr-Symbol rechts neben dem Eingabefeld klicken. Dies ist in solchen Fällen hilfreich, wenn Sie einen Artikel überarbeiten (oder einen Artikel im Entwurfsmodus veröffentlichen wollen) und dabei die Uhrzeit aktualisieren möchten.

Wenn Sie hier einen zukünftigen Zeitpunkt eintragen, wird Serendipity einen derart veröffentlichten Eintrag in der Übersicht nicht anzeigen. Sobald jedoch der eingetragene Zeitpunkt erreicht wurde, wird der Artikel ohne Ihr eigenes Zutun automatisch dargestellt. Somit können Sie dieses Eingabefeld auch für eine zeitgesteuerte Veröffentlichung nutzen.

Kategorie

Rechts neben der Datumseingabe befindet sich ein Ausklappmenü, das standardmäßig auf **Keine Kategorie** gesetzt ist. Wenn Sie das Feld aufklappen, sehen Sie eine Liste aller erstellten Kategorien in hierarchischer Gliederung.

Während eine einzelne Kategorie durch normale Auswahl des Ausklappmenüs für den Artikel festgelegt wird, können Sie einem Artikel zusätzlich mittels Klick auf das +-Symbol links neben diesem Ausklappmenü auch mehrere Kategorien zuweisen. Nach einem Klick auf das Symbol verändert sich das Ausklappmenü in ein mehrzeiliges Auswahlfeld. Mit gedrückter *(Strg/Apfel)*-Taste und einem linken Mausklick können Sie dabei mehrere Kategorien markieren und mit einem zweiten Klick eine Markierung auch wieder aufheben. Alle ausgewählten Kategorien werden dabei vom Browser markiert dargestellt.

Eintrag

Den größten Platz dieser Seite nimmt das Feld für den eigentlichen Artikeltext ein. Je nachdem, ob Sie den *WYSIWYG*-Editor in den **Eigenen Einstellungen** aktiviert haben, sieht dies etwas unterschiedlich aus.

Eintrag mit deaktiviertem WYSIWYG-Editor

Bei deaktiviertem *WYSIWYG*-Editor sehen Sie ein Feld, in das direkt HTML-Code eingetragen wird. Wenn Sie einfachen, unformatierten Text schreiben wollen, können Sie diesen hier einfach einfügen. In HTML werden Umbrüche nicht durch einen üblichen Zeilenumbruch mittels Eingabe-Taste angegeben, sondern mittels des HTML-Tags `<br />`. Damit Sie aber Artikel trotz HTML-Tags bequem verfassen können, ist standardmäßig bei der Serendipity-Installation ein Plugin namens *Textformatierung: NL2BR* aktiviert worden. Dieses kümmert sich beim Speichern Ihrer Artikel darum, automatisch den gewohnten Zeilenumbruch in einen HTML-Zeilenumbruch umzuwandeln.

Rechts oberhalb des Eingabebereichs für den **Eintrag** befindet sich eine kleine Symbolleiste, auf der mehrere Buttons verteilt sind. Damit können Sie einige Formatierungsoptionen auf Ihren Artikeltext anwenden. Geben Sie dazu einen kleinen Text in **Eintrag** ein und markieren Sie ein Wort daraus per Maus.

Ein Klick auf den Button **I** formatiert ein Wort *kursiv*, ein Klick auf **B** formatiert es *fett*, und ein Klick auf **U** unterstreicht das gewählte Wort.

Mittels des Buttons **Zitat** können Sie einen größeren gewählten Textbereich als Zitat markieren. Gerade bei Blog-Einträgen ist die Form des Zitierens sehr gebräuchlich.

Die letzten drei Buttons dienen der Einbindung von externen Dateien und benötigen daher keine vorherige Textauswahl. Ein Klick auf einen der Buttons **img**, **Mediendatenbank** oder **URL** wird an der Stelle, wo sich der Cursor im Text befindet, das gewünschte Objekt einbinden.

img bindet ein externes Bild ein. Bei einem Klick auf den Button werden Sie über ein Popup-Fenster aufgefordert, die URL des Bildes einzugeben. Tragen sie die URL vollständig mit `http://`-Präfix ein. Nach dem Klick auf **OK** wird automatisch der benötigte HTML-Code in Ihren Eintrag eingefügt – so können Sie sich auch direkt merken, wie ein Bild bei Gelegenheit auch manuell eingefügt werden kann.

Mediendatenbank öffnet beim Anklicken ein neues Popup-Fenster, in dem ein Bild aus der eigenen Mediendatenbank eingefügt werden kann. Eine detaillierte Beschreibung dieses Popups folgt auf Seite 107.

URL öffnet ein Eingabefenster, um einen Hyperlink zu einer Webseite einzutragen. Geben Sie dort die URL wieder vollständig mit `http://`-Präfix ein. Nach dem Klick auf **OK** können Sie zusätzlich die Beschreibung des Hyperlinks eingeben, wie er später dem Benutzer dargestellt wird. Sie können hierbei auch vor dem Klick auf den Button **URL** einen Text in Ihrem Artikel mit der Maus markieren, dieser wird dann beim Einfügen eines Hyperlinks automatisch als Beschreibungstext eingesetzt. Zuletzt fordert Sie der Einfügedialog noch auf, einen *title/tooltip* zu vergeben. Der Text, den Sie dort eintragen, wird später beim Darüberfahren mit der Maus für den Besucher in einem Info-Popup angezeigt.

Eintrag mit aktiviertem WYSIWYG-Editor

Ist in Ihren **Eigenen Einstellungen** der WYSIWYG-Editor aktiviert, sehen Sie einen Eingabebereich, der aus Microsoft Word oder OpenOffice bekannte Bedienelemente abbildet. Die meisten der Symbole werden auf einen markierten Text im Artikel angewendet. Wenn Sie mit der Maus darüberfahren und etwas warten, wird ein Info-Popup die jeweilige Beschreibung des Icons anzeigen. Im Einzelnen bedeuten sie Folgendes:

- **font** formatiert einen ausgewählten Textbereich mit der im Auswahlfeld gewählten Schriftart.
- **size** formatiert einen ausgewählten Textbereich mit der im Auswahlfeld gewählten Schriftgröße.

- **format** stellt ein, welchem Texttyp ein ausgewählter Textbereich entspricht. Überschriften werden später auch als HTML-Überschriften gesetzt.
- **B** formatiert einen ausgewählten Textbereich fett.
- **I** formatiert einen ausgewählten Textbereich kursiv.
- **U** formatiert einen ausgewählten Textbereich unterstrichen.
- **S** formatiert einen ausgewählten Textbereich durchgestrichen.
- Die beiden **x**-Zeichen mit hoch- und tiefgestellter **2** können einen ausgewählten Textbereich entsprechend positionieren.
- Im letzten Bereich der oberen Symbolleiste befinden sich die Funktionen für die Zwischenablage. Ein markierter Text kann hiermit in die Zwischenablage **kopiert** oder **ausgeschnitten** und von dort in den Text **eingefügt** werden. Die beiden Pfeile **zurück** und **vor** können eine vorherige Aktion rückgängig machen oder wiederherstellen.
- Der erste Bereich der unteren Symbolzeile ermöglicht es, den Textfluss eines gewählten Absatzes zu verändern: **linksbündig**, **zentriert**, **rechtsbündig** und **Blocksatz**. Beachten Sie beim Blocksatz, dass dies aufgrund fehlender Browser-Unterstützung für Wortumbrüche selten empfehlenswert ist.
- Die beiden Symbole mit dem Paragraphen-Symbol können einen markierten Textbereich in eine bestimmte Textrichtung (**links nach rechts** oder **rechts nach links**) fließen lassen. Dies ist hauptsächlich für arabische Sprachen interessant.
- Im nächsten Bereich befinden sich Icons, die den jeweiligen markierten Text in eine **nummerierte** oder **geordnete** Listenaufzählung formatieren können. Die Buttons rechts daneben dienen zur **Einrückung** eines Textes.
- Die Textfarbe und Hintergrundfarbe eines markierten Textbereichs stellen Sie über die beiden Icons mit dem **T**-Symbol bzw. dem **Farbeimer** ein. Bei einem Klick darauf öffnet sich ein Farbauswahlfenster.
- Der Button mit dem horizontalen Strich fügt ein **Trennzeichen** in den Text an der aktuellen Cursorposition ein.
- Der Button mit dem **Kettensymbol** fügt an der aktuellen Cursorposition im Text einen Hyperlink ein. Bei einem Klick auf den Button werden Sie in einem Popup-Fenster aufgefordert, das Linkziel und die Linkbeschreibung einzutragen. Etwaiger vormarkierter Text wird hierbei als Beschreibung vorausgewählt.
- Das Symbol mit dem **Bilderrahmen** ermöglicht das Einfügen eines externen Bildes. Bei einem Klick hierauf öffnet sich ein Dialog, in dem Sie die URL eines Bildes sowie weitere Optionen eintragen können. Diese sind **Alternate text** (für eine manuell vergebene Bildbeschreibung), **Alignment** (für die Ausrichtung des Bildes), **Border thickness** (für die Randstärke eines Bildes), **Horizontal spacing** (für den horizontalen Abstand in Pixel des Bildes zum Text), **Vertical spacing** (für den vertikalen Abstand in Pixel des Bildes zum Text). Bei einem Klick auf **OK** wird das gewählte Bild in den Text eingefügt.

- Die komfortablere Art, ein Bild einzufügen, führt über die integrierte Mediendatenbank von Serendipity. Das Symbol hierfür befindet sich rechts neben dem Bilderrahmen und soll eine stilisierte **Bilderübersicht** in einer Liste darstellen. Ein Klick hierauf öffnet das Mediendatenbank-Popup, das auf Seite 107 eingehender erklärt wird.
- Über den Button **Tabelle** lässt sich eine HTML-Tabelle einbinden. Dies ist leider relativ komplex gelöst und erfordert viel Fingerspitzengefühl bei der Einrichtung einer Tabelle, da Spalten- und Zeilenanzahl vorher festgelegt werden müssen. Eine Tabelle kann nach ihrem Einfügen im Eingabetext selber gefüllt werden.
- Der letzte Button **¡¿** hat eine ganz besondere Bedeutung: Er schaltet vom normalen WYSIWYG-Modus in den HTML-Modus um. Sobald der HTML-Modus aktiviert ist, kann man sozusagen einen Blick *hinter die Kulissen* werfen und prüfen, wie der Editor die eigenen Eingaben in HTML umsetzt. Wenn man sich mit HTML etwas auskennt, kann man hier möglicherweise manuell einige Fehler beheben, die der WYSIWYG-Editor bei einigen Operationen einfügen könnte. Während der HTML-Modus aktiv ist, sind alle weiteren Optionen des WYSIWYG-Editors so lange ausgeblendet, bis der HTML-Modus durch erneuten Klick wieder deaktiviert wird.
- Der Button rechts neben dem HTML-Modus ermöglicht das Öffnen eines Popup-Fensters, in dem man den Text des Artikels in einem größeren Bereich bearbeiten kann. Diese Ansicht muss man mit erneutem Klick auf den Button wieder schließen, um die Änderungen in das Serendipity-Fenster zu übernehmen.
- Mittels **?** kann man eine Hilfe-Seite öffnen, die die (englische) Dokumentation der verwendeten WYSIWYG-Komponente HTMLArea öffnet.
- Der letzte Button **!** öffnet eine kleine Informationsseite von HTMLArea.

Im Textbereich können Sie wie gewohnt Text eintippen. Wenn Sie die *(Enter)*-Taste drücken, wird ein neuer Absatz begonnen, wenn Sie *(Shift)+(Enter)* drücken, wird ein einfacher Zeilenumbruch eingefügt.

Beachten Sie, dass Sie bei aktiviertem WYSIWYG-Editor das Textformatierungs-Plugin *Textformatierung: NL2BR* (siehe Seite 255) besser deinstallieren sollten. Andernfalls könnte es zu doppelten Zeilenumbrüchen kommen oder bei Tabellen zu überflüssigen Zeilenumbrüchen vor/nach der Tabelle. Alternativ können Sie das NL2BR-Plugin auch gezielt nur für einzelne Artikel deaktivieren, indem Sie es mithilfe des Plugins *Erweiterte Eigenschaften von Artikeln* (siehe Seite 262) in den **Erweiterten Optionen** des jeweiligen Artikels aus der Liste auswählen.

Kommentar-Optionen

Unterhalb des Eingabebereichs für den Text gibt es zwei Optionen, mit denen man steuern kann, ob Kommentare zu diesem Artikel zugelassen sind, und wenn ja, ob diese Kommentare nach dem Abschicken durch den Besucher erst von einem Redakteur freigeschaltet werden müssen. Die Voreinstellung dieser beiden Optionen wird in den **Eigenen Einstellungen** vorgenommen.

Aktionen

Rechts unterhalb des Eingabebereichs befinden sich zwei Aktions-Buttons. Der Button **Vorschau** lädt die Seite neu und zeigt oberhalb der Erfassungsmaske eine Voransicht des gerade erstellten Artikels an. Die Voransicht ist nur eine vage Annäherung, wie der Artikel später im Blog aussehen wird, kann aber dennoch als grobe Richtlinie dienen, um dessen Formatierung zu prüfen.

Mittels des Buttons **Speichern** wird ein Artikel im System gespeichert. Sollte vor einem Klick auf diesen Button der Computer abstürzen, sind die Einträge unwiderruflich verloren. Daher empfiehlt es sich sehr, einen längeren Artikel erst in den **Entwurfs-Modus** zu setzen und zwischendurch häufiger zu speichern, oder alternativ den Text in einem normalen Schreibprogramm vorzutippen.

Was beim Speichern eines Artikels geschieht, erfahren Sie auf Seite 111.

Erweiterter Eintrag

Der Artikeltext, den Sie im Bereich **Eintrag** verfasst haben, wird später im Frontend vollständig auf den Übersichtsseiten angezeigt. Dieser Text stellt somit einen *Teaser*[7] dar.

Der vollständige Text eines Artikels kann im Bereich **Erweiterter Eintrag** verfasst werden. Der hier eingegebene Text wird später im Frontend erst auf der Detailseite eines Artikels angezeigt und ist auch standardmäßig nicht im RSS-Feed enthalten.

Bei der Ansicht der Detailseite eines Artikels wird der normale Artikeltext zusätzlich angezeigt. Verfassen Sie daher Ihre Beiträge so, dass die beiden Textbereiche ineinander übergehen.

Der Bereich **Erweiterter Eintrag** ist identisch formatiert wie der normale **Eintrag**. Sie können im Nicht-WYSIWYG-Modus diesen erweiterten Bereich mittels des Icons + ein- und ausklappen.

Erweiterte Optionen

Unterhalb des erweiterten Eintrags befindet sich ein Block mit dem Titel **Erweiterte Optionen**. Hier werden diverse Optionen von eingebundenen Plugins angezeigt, und auch eigene Plugins können an dieser Stelle eingebunden werden. Die angebotenen Optionen an dieser Stelle können Sie der jeweiligen Erklärung der Plugins in den folgenden Kapiteln entnehmen.

Mediendatenbank-Popup

Das Mediendatenbank-Popup ist ein Fenster, das sich an mehreren Stellen Serendipitys öffnen lässt, vor allem beim Einfügen von Bildern bei der Artikel-Erstellung.

[7] Ein Teaser ist ein Kurztext, der dem Leser einen Vorgeschmack auf die ausführliche Fassung geben soll.

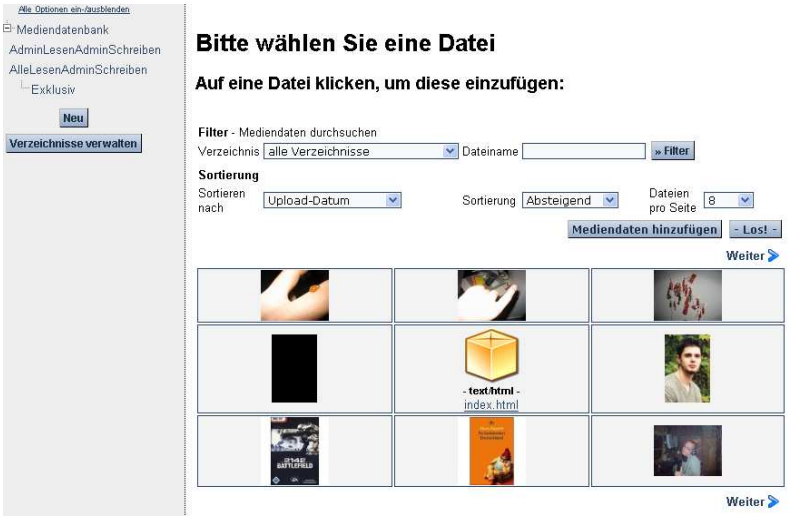

Abbildung 4.5: Mediendatenbank-Popup zum Einfügen einer Datei

Das Popup-Fenster enthält eine vereinfachte Ansicht der Mediendatenbank Serendipitys. Abhängig von der Option **Symbolleiste für das Mediendatenbank-Popup anzeigen** in den **Eigenen Einstellungen** wird Ihnen jedoch zusätzliche Funktionalität angeboten, die im Kapitel 4.5 auf Seite 124 detailliert beschrieben wird.

Solange das Popup-Fenster geöffnet ist, sollten Sie im Ursprungsfenster von Serendipity keine Änderungen vornehmen. Sollten Sie dies tun, kann das Popup-Fenster ein ausgewähltes Bild nicht mehr an den ursprünglich vorgesehenen Platz stellen.

Das Fenster ist in zwei Bereiche aufgeteilt. Auf der linken Seite befindet sich eine Übersicht über die vorhandenen Verzeichnisse in der Mediendatenbank. Diese Struktur entspricht der des `uploads`-Verzeichnisses auf Ihrem Server.

Da sich Verzeichnisse beliebig verschachteln lassen, können Unterverzeichnisebenen über einen Klick auf ein +-Symbol geöffnet und danach auch wieder durch einen Klick auf - geschlossen werden. Diese Art der Verzeichnisnavigation kennen Sie sicher auch aus dem Windows Explorer oder ähnlichen Dateiverwaltungs-Programmen auf Ihrem Computer.

Oberhalb der Verzeichnisse können Sie durch einen Klick auf **Alle Optionen ein-/ausblenden** alle Unterverzeichnisebenen auf einmal aus- bzw. einklappen. Je nachdem, wie viele Unterverzeichnisse Sie besitzen, kann die Ansicht sehr groß werden – in so einem Fall können Sie die üblichen Browser-Elemente zum Vergrößen/Verkleinern des Popup-Fensters benutzen und die Scrollbalken einsetzen.

Unterhalb der Verzeichnisliste befinden sich zwei Buttons **Neu** und **Verzeichnisse verwalten**, mit denen man neue Unterverzeichnisse anlegen kann. Wenn Sie nur ein Bild in einen Artikel einfügen möchten, ist dies meist nicht notwendig, daher lesen Sie bitte die Beschreibung dieser Möglichkeiten im Kapitel 4.5 ab Seite 124.

Ein Klick auf einen beliebigen Verzeichnisnamen im linken Bereich beeinflusst die Darstellung des Popup-Fensters auf der rechten Seite. Dabei stellt die Übersicht dort alle Dateien des gewählten Verzeichnisses dar.

Standardmäßig sehen Sie auf der rechten Seite die Bilder aller Verzeichnisse in chronologischer Reihenfolge, also unabhängig von einem gewählten Verzeichnis. In dieser Dateiübersicht sehen Sie jeweils Vorschaubilder für alle hochgeladenen Dateien. Auch können dort Nicht-Bilddateien (Word-Dokumente, PDF-Dateien, ZIP-Archive . . .) angezeigt werden.

Wenn Sie eine bestimmte Datei suchen, werden Ihnen die Filteroptionen der Mediendatenbank sehr hilfreich sein. Diese sind ebenfalls detailliert im Kapitel 4.5 ab Seite 133 beschrieben.

Da wir das Popup-Fenster nur aufgerufen haben, um ein Bild auszuwählen, dient die Dateiübersicht der einfachen Auswahl jenes Bildes. Ein Klick auf eines der Vorschaubilder führt zu einer Folgeseite, in der Sie bestimmen können, wie das Bild eingefügt werden soll. Abhängig vom Kontext, in dem Sie das Popup-Fenster aufgerufen haben, kann es sein, dass die Folgeseite nicht erscheint, sondern das Objekt direkt in die aufrufende Seite eingetragen wird. Auch wenn Sie eine einfache Datei anstelle eines Bildes einfügen wollen, wird die Folgeseite nicht erscheinen, sondern die Datei wird direkt in den Artikel eingebunden.

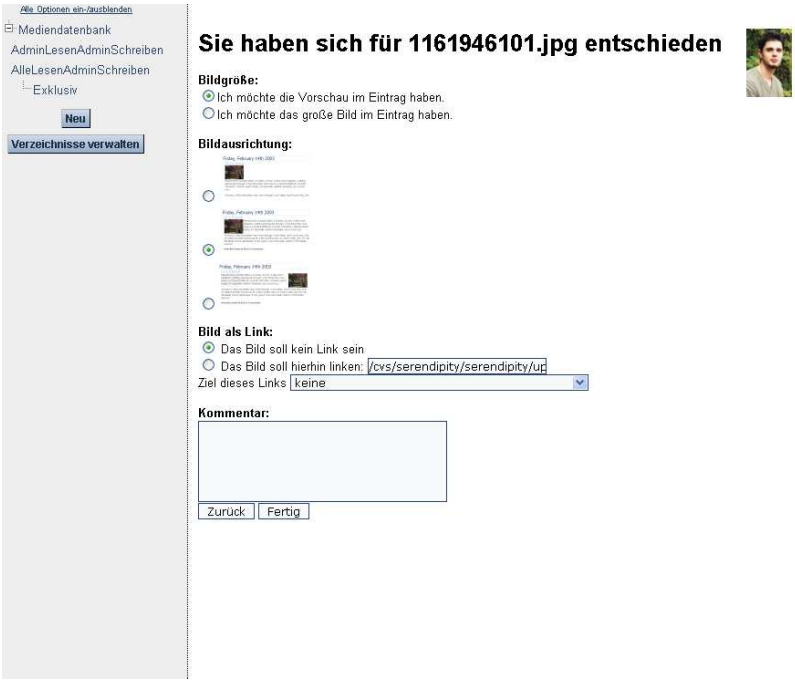

Abbildung 4.6: Mediendatenbank-Popup zur Formatierung einer eingefügten Datei

Auf dieser Folgeseite sehen Sie im oberen Bereich, für welche Datei Sie sich entschieden haben. Die kleine Vorschaugrafik wird ebenfalls dargestellt.

Es folgen nun einige Bereiche, in denen Sie das Layout des eingefügten Bildes bestimmen:

- Bei dem Bereich der **Bildgröße** legen Sie fest, ob nur das kleine Vorschaubild eingefügt werden soll oder das Bild in seiner Originalgröße.
- Über die **Bildausrichtung** bestimmen Sie, wie das Bild im Text des Artikels arrangiert wird. Dabei wird das Layout durch eine kleine Grafik visuell dargestellt. Die erste Option richtet ein Bild oberhalb eines Absatzes, also für sich allein stehend aus. Die zweite Möglichkeit bettet ein Bild so ein, dass es vom folgenden Text linksbündig umflossen wird. Die letzte Möglichkeit bindet ein Bild so ein, dass es innerhalb des Textes rechtsbündig dargestellt wird.
- Im Abschnitt **Bild als Link** können Sie zusätzlich einen Link auf ein Bild legen. Bei einem Klick auf ein Bild wird dann die eingetragene URL aufgerufen. Standardmäßig wird der Dateipfad zu der großen Version eines Bildes in dem Eingabefeld **Das Bild soll hierhin linken:** eingefügt. Über die Option **Ziel dieses Links** kann man einstellen, wie dieser Link im Browser geöffnet wird. **Popup-Fenster (JavaScript)** wird das Bild in einem Fenster öffnen, das sich der Bildgröße genau anpasst. **Eigenständige Seite** wird eine spezielle Frontend-Seite öffnen, die ausschließlich Ihr Bild im umgebenden Layout und mögliche Meta-Daten eines Bildes anzeigt. Die Option **Popup-Fenster (target=_blank)** nutzt eine einfachere Version eines Popup-Fensters, die auch ohne aktiviertes JavaScript des Besuchers ein neues Fenster öffnen kann.
- Zuletzt haben Sie noch die Möglichkeit, zu einem Bild eine Bildunterschrift in das Feld **Kommentar** hinzuzufügen. Dieses wird dann im Artikel später dem Bild zugehörig angezeigt.

Wenn Sie abschließend auf **Fertig** klicken, wird das Bild an die aufrufende Stelle Serendipitys zurückgeliefert und mittels HTML-Code eingebunden. Dort können Sie es gegebenenfalls auch noch innerhalb des Eintrages verschieben und anpassen.

Sämtliche Einfügeoperationen des Mediendatenbank-Popups benötigen JavaScript. Sollte das Einfügen bei Ihnen also fehlschlagen, prüfen Sie, ob Sie JavaScript im Browser korrekt aktiviert haben. Auch mögliche Popup-Blocker oder die Firefox-Erweiterung *NoScript* könnten den erfolgreichen Aufruf der Mediendatenbank verhindern.

Wenn Sie gerne ein Bild einfügen möchten, das Sie noch nicht in die Mediendatenbank hochgeladen haben, dann müssen Sie dafür nicht extra auf den Menüpunkt **Mediendaten hinzufügen** im Hauptmenü klicken. Stattdessen ermöglicht es das Mediendatenbank-Popup in der Bildübersicht, direkt, mittels Klick auf den Button **Mediendaten hinzufügen** eine neue Datei von Ihrem Computer hochzuladen. Diese Datei wird dann direkt für die Folgeseite zur Bestimmung der Einfügungsoptionen übernommen, was Ihnen einiges an Klickarbeit abnimmt.

Details zum Hochladen von Dateien finden Sie im Kapitel 4.5 auf Seite 125, die auch für das Mediendatenbank-Popup gelten.

Speichern eines Eintrags

Wenn Sie einen Beitrag speichern, wird Serendipity in einem eigenen Bereich einige Aktionen durchführen. Zuerst wird der Artikel vollständig in der Datenbank gespeichert. Danach können etwaige Plugins ausgeführt werden, die individuelle Funktionen auf den Artikel anwenden und ihn umwandeln. So könnte ein Plugin z. B. den neuen Artikel an Google und andere Suchmaschinen wie Technorati übermitteln, oder der Text könnte auf korrekte Syntax geprüft werden.

Sollte der Artikel als Veröffentlichung gespeichert sein, wird in einem weiteren Schritt der Artikeltext automatisch nach allen enthaltenen Hyperlinks durchsucht. Jeder gefundene Hyperlink wird daraufhin von Serendipity aufgerufen und geprüft, ob ein Trackback (siehe Seite 433) an diese Adresse gesendet werden soll. Für jede gefundene URL wird auf dem Bildschirm eine Nachricht ausgegeben, ob ein Trackback an diese URL geschickt werden konnte.

Auch etwaige Fehlermeldungen werden Ihnen beim Speichern an dieser Stelle angezeigt. Der Prozess endet mit einer Meldung Serendipitys, dass der Eintrag gespeichert wurde. Falls der Artikel direkt veröffentlicht wurde, können Sie die URL mit diesem Artikel direkt anklicken.

Wenn Serendipity lediglich den Entwurf eines Beitrags gespeichert hat, wird keine Trackback-Analyse durchgeführt, und Serendipity meldet nach dem Speichern, dass der Entwurf eines Artikels gespeichert wurde.

Sollten Sie nach dem Speichern eines Artikels keine solche Meldung erhalten, bedeutet dies möglicherweise, dass der Webserver Fehler verursacht hat. In diesem Fall sollten Sie die Fehler-Logfiles des Servers prüfen, da hierfür meist Netzwerk-Verbindungsprobleme oder Firewalls zuständig sind. In so einem Fall müssten Sie möglicherweise die Trackback-Funktionalität deaktivieren (siehe Seite 441).

Um vorab Probleme mit verlorenen Artikeltexten zu vermeiden, können Sie die auf Seite 276 beschriebenen Maßnahmen einsetzen.

Nach dem Speichern landen Sie erneut auf der Seite zur Bearbeitung des Artikels, um gegebenenfalls Änderungen vorzunehmen und den Artikel erneut zu speichern.

4.4.2 Einträge bearbeiten

Um einen geschriebenen Beitrag später zu bearbeiten, benutzen Sie den Hauptmenüpunkt **Einträge bearbeiten**. Bei einem Klick auf diesen Menüpunkt wird eine blätterbare Liste aller geschriebenen Einträge dargestellt.

Bei einem Klick auf den jeweiligen Eintrag gelangt man auf die Seite, in der man einen **Neuen Eintrag** erstellen kann. Die Felder sind dabei vorausgefüllt mit den jeweiligen Daten

des gewählten Eintrags, und ein Speichern erstellt keinen neuen Eintrag, sondern überarbeitet den bestehenden.

Die Liste der Einträge stellt in der ersten Zeile den Titel des Artikels dar. Sollte der Artikel noch nicht veröffentlicht worden sein, wird das Wort *Entwurf:* vorangestellt. Wenn Sie mit der Maus über einen Beitragstitel fahren, wird ein Info-Popup angezeigt, in dem die Artikel-ID angezeigt wird. Diese ID kann an einigen Stellen des Serendipity-Backends eingetragen werden, unter anderem auch am Ende der Seite zum schnellen Bearbeiten einer bekannten Artikelnummer.

Rechts neben dem Titel wird die Erstellungszeit des Artikels angezeigt. Wenn rechts daneben das **Uhrsymbol** steht, heißt dies, dass der Artikel seit seiner Erstellung mindestens einmal überarbeitet wurde. Wenn Sie mit der Maus über dieses Icon fahren (oder klicken), zeigt ein Info-Popup den Zeitpunkt der letzten Aktualisierung an.

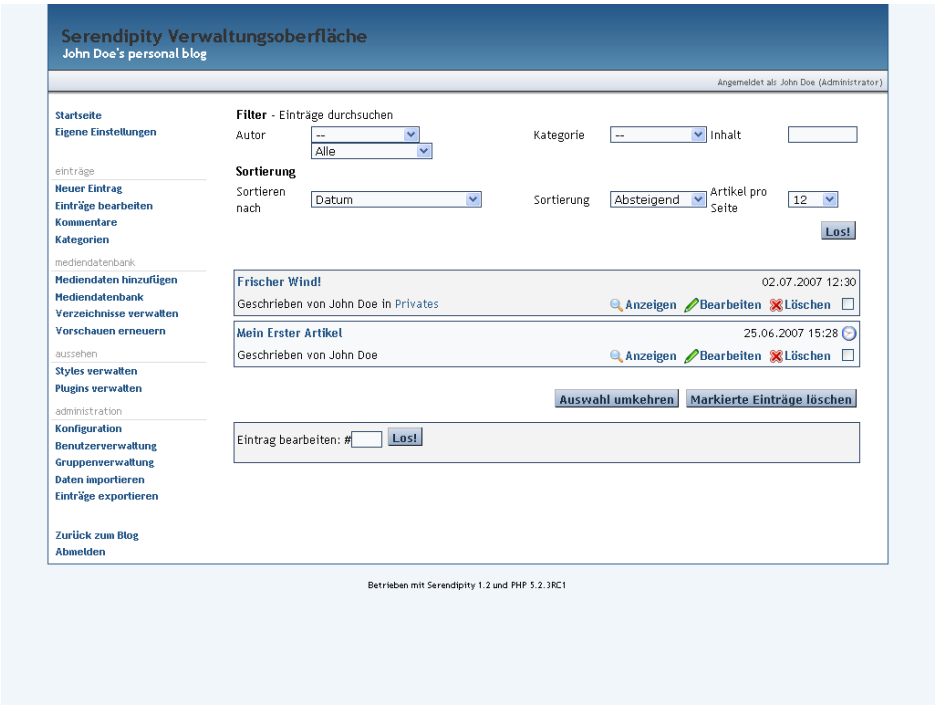

Abbildung 4.7: Einträge: Einträge bearbeiten

Die Zeile unterhalb des Titels gibt an, wer der Eigentümer eines Artikels ist. Bei Bearbeitung eines Artikels durch andere Redakteure ändert sich der Eigentümer nicht. Sollte ein Artikel in einer oder mehreren Kategorien zugeordnet sein, werden alle Kategorienamen nach dem Autornamen aufgelistet. Ein Klick auf den Kategorienamen ruft die Frontend-Ansicht der jeweiligen Kategorie auf.

Innerhalb der Box, die den Artikel anzeigt, werden auf der rechten Seite drei Buttons darge-

stellt. Der erste Button mit einer *Lupe* zeigt bei einem Klick abhängig vom Veröffentlichungsstatus entweder die **Ansicht** oder **Vorschau** des Artikels im Frontend in einem neuen Fenster an. Der zweite Button stellt mit dem *Stift* die Möglichkeit bereit, einen Artikel zu bearbeiten (wie bei einem Klick auf den Artikeltitel). Der Button mit *rotem Kreuz* löscht einen Artikel unwiderruflich. Zur Sicherheit werden Sie beim Löschen gefragt, ob Sie die Aktion wirklich ausführen wollen.

Als Letztes wird eine Auswahlbox neben dem **Löschen**-Button angezeigt. Hier können Sie mehrere Artikel auswählen, um anschließend am Seitenende durch den Klick auf **Markierte Einträge löschen** diese Artikel gesammelt zu löschen.

Der Button **Auswahl umkehren** wird jeden gewählten Artikel deselektieren und jeden deselektierten Artikel wieder auswählen. Wenn Sie also einmal alle Artikel bis auf einen löschen wollen, markieren Sie nur diesen einen Artikel, klicken auf **Auswahl umkehren**, wodurch automatisch die anderen Artikel ausgewählt werden, und können danach löschen.

Am Ende der Seite finden Sie eine Eingabebox, in der Sie die Artikel-ID eingeben können, um einen Eintrag direkt zu bearbeiten. Alle Artikel in Serendipity werden bei 1 beginnend automatisch durchnummeriert. Wenn ein Artikel einmal gelöscht werden sollte, verändert sich die ID aller anderen Artikel nicht, daher können ggf. Lücken entstehen.

Am Anfang der Seite sehen Sie mehrere Filtermöglichkeiten, die die Anzeige der Artikel beinflussen. Standardmäßig werden Ihnen die chronologisch aktuellsten Artikel angezeigt, die Liste ist über die Buttons **Weiter** und **Zurück** ober- und unterhalb der Artikel blätterbar.

Folgende Filteroptionen sind verfügbar:

Autor
: Das Auswahlfeld neben der Option **Autor** schränkt die Darstellung der Artikel auf den ausgewählten Autor ein.

Artikelmodus
: Unterhalb des Autors gibt es ein weiteres Auswahlfeld, mit dem Sie auswählen können, ob nur **Entwürfe**, **Veröffentlichungen** oder beide Artikelarten angezeigt werden sollen.

Kategorie
: Um nur Artikel einer gewissen Kategorie anzuzeigen, können Sie im Auswahlfeld **Kategorie** die gewünschte Auswahl treffen.

Inhalt
: In das Feld **Inhalt** können Sie ein beliebiges Suchwort eingeben, das in einem Artikeltext (oder dem erweiterten Artikeltext) vorhanden sein muss. Die Suche berücksichtigt dabei auch Teilworte und ignoriert die Klein- und Großschreibung, wenn Sie also `miet` eingeben, werden auch Artikel angezeigt, die *vermieten* oder *Mietbüro* enthalten.

Sortieren nach

Die Sortierung der gefilterten Artikel bestimmen Sie durch das Auswahlfeld **Sortieren nach**. Folgende Sortierungsmöglichkeiten stehen zur Verfügung: **Datum**, **Veröffentlichung/Entwurf**, **Autor** (Autorname, alphabetisch), **Kategorie** (Kategoriename, alphabetisch), **Zuletzt aktualisiert**, **Titel** (Artikeltitel, alphabetisch) und die **ID** eines Artikels.

Sortierung

Die Rangfolge der Sortierung (**Absteigend** oder **Aufsteigend**) können Sie durch das Auswahlfeld **Sortierung** einstellen. Dadurch bestimmen Sie, ob die Liste vom *aktuellsten/ersten* Eintrag bis zum *ältesten/letzten* Eintrag sortiert wird oder umgekehrt.

Artikel pro Seite

Standardmäßig enthält die Übersicht die letzten 12 Artikel. Um eine größere Übersicht zu ermöglichen, können Sie die **Artikel pro Seite** auf die Werte 12, 16, 15, 50 oder 100 stellen. Beim Blättern wird die jeweilige Folgeseite ebenso viele Artikel darstellen.

Ein Klick auf **Los!** führt die gewünschte Filterung aus und zeigt die daraus resultierende Artikelauswahl in der eingetragenen Sortierung. Bei Angabe mehrerer Filterkriterien werden diese alle miteinander *UND*-verkettet, d. h. es müssen alle Filterbedingungen zutreffen, damit ein Artikel in der folgenden Liste aufgeführt wird.

Die zuletzt eingestellten Filter- und Sortierungsoptionen werden in einem Cookie gespeichert und beim nächsten Aufruf des Menüs erneut angewendet. Wenn Sie also einmal eine chronologische Auswahl erwarten würden, prüfen Sie zuerst, ob die Filtereinstellungen dies verhindern.

4.4.3 Kommentare

Ähnlich wie die Übersichtsseite **Einträge bearbeiten** enthält die Seite **Kommentare** eine Übersicht aller übermittelten Kommentare und Trackbacks Ihrer Besucher.

Von dieser Oberfläche aus können Sie Kommentare löschen, freischalten, überarbeiten oder auch nach speziellen Kommentaren suchen.

Der Inhaltsbereich zeigt die Kommentare in einzelnen Boxen untereinander an. Direkt oberhalb dieser Boxen befindet sich ein Informationstext, der die Anzahl der dargestellten Kommentare und die Nummer der aktuellen Seite angibt. Die Möglichkeit zum Blättern mittels **Weiter** und **Zurück** befindet sich sowohl oberhalb als auch unterhalb der Kommentarauflistung.

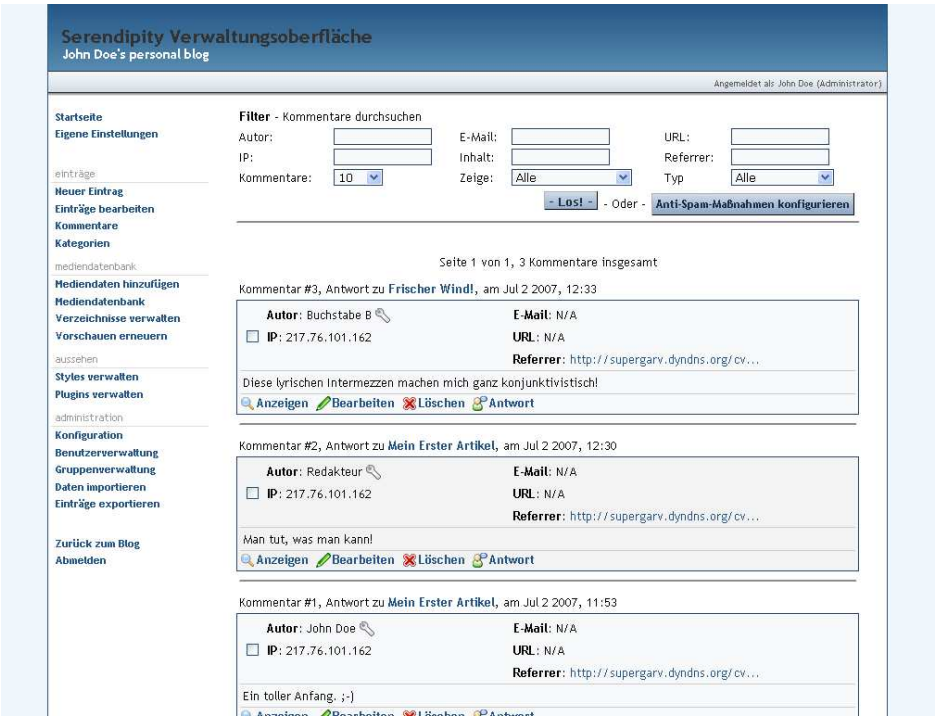

Abbildung 4.8: Einträge: Kommentare

Jede Kommentarbox enthält folgende Angaben:

Überschrift
: Oberhalb jeder Box steht eine kleine Informationszeile, die angibt, ob die Box einen Trackback oder einen Kommentar darstellt, zu welchem Artikel der Kommentar gehört und wann er erstellt wurde. Der Bezug zu einem Artikel ist dabei klickbar und führt ins Frontend Ihres Blogs.

  Nicht freigeschaltete Kommentare und Trackbacks werden durch eine auffällige gelbe Markierung vom Rest abgehoben.

Autor
: Der Name des Verfassers eines Kommentars wird hinter dem Punkt **Autor** dargestellt. Wenn das *Spamblock*-Plugin (siehe Seite 239) installiert ist, wird neben dem Namen ein Symbol mit **Schraubstock** angezeigt. Ein Klick auf dieses Symbol wird in Zukunft Kommentare des gleichnamigen Autors in eine *Blacklist* einfügen und abweisen. Wenn der Schraubstock rot eingefärbt erscheint, wird der Name bereits gefiltert, und ein erneuter Klick hebt den Filter wieder auf.

E-Mail
: Die E-Mail-Adresse eines Kommentators wird neben **E-Mail** angezeigt. Auch hier

kann bei aktiviertem *Spamblock*-Plugin die E-Mail-Adresse in einen Filter aufgenommen werden.

IP

In der zweiten Zeile der Kommentarbox wird die IP-Adresse des Kommentators angezeigt. Eine IP-Adresse identifiziert einen Benutzer im Internet beim Aufruf einer Seite. Sollte Sie einmal rechtswidrige Inhalte in einem Kommentar auffinden, kann eine Strafverfolgungsbehörde mit Hilfe dieser IP möglicherweise den wahren Täter aufspüren. Meist werden für derartige Straftaten aber offene *Proxies* eingesetzt, die die eigene IP-Adresse des Täters verschleiern, so dass er sozusagen nur über einen Deckmann auf Ihren Server zugreift.

URL

Die Homepage eines Kommentators wird unterhalb seiner E-Mail-Adresse angezeigt. Ein Klick hierauf öffnet die Homepage direkt in einem neuen Browserfenster. Auch die URL kann als Filter für das *Spamblock*-Plugin durch Klick auf das **Schraubstock**-Symbol gesetzt werden.

Referrer

Wenn ein Besucher Ihre Webseite aufruft, übermittelt sein Browser üblicherweise die URL der zuletzt aufgerufenen Webseite. Wenn ein Besucher über eine Suchmaschine den Weg zu Ihnen findet, gibt der sogenannte *Referrer* Auskunft darüber, nach welchen Begriffen er gesucht hat. Möglicherweise ist es also für Sie interessant zu erfahren, von welcher Webseite ein Besucher zu Ihnen kam und dann einen Kommentar hinterlassen hat.

Genau diese vorherige Webseite wird in der Kommentarbox beim Punkt **Referrer** eingebunden, ein Klick darauf öffnet die entsprechende Webseite.

Wenn ein Besucher keine vorherige Webseite besucht hat, ist in diesem Feld meistens die URL des kommentierten Artikels enthalten.

Kommentar

Der eigentliche Text, den der Besucher als Kommentar (oder Trackback) hinterlassen hat, wird im Hauptteil der Box angegeben. Damit zu lange Texte nicht den Rahmen sprengen, werden sie nach 160 Zeichen mit einem . . . abgekürzt.

Symbolleiste

Unterhalb jedes Kommentars befindet sich eine Symbolleiste mit Buttons. Die Buttons richten sich dabei danach, ob das Objekt ein Kommentar oder ein Trackback ist, ob es freigeschaltet oder moderiert ist und ob der Text länger als 160 Zeichen ist oder nicht.

Bei noch nicht freigeschalteten Beiträgen erscheint der Button **Bewilligen** als Erstes. Ein Klick darauf wird den gewählten Beitrag freischalten. Ist ein Kommentar bereits freigeschaltet, erscheint der Button **Moderieren**, um den Kommentar erneut zu verstecken.

Bei Beiträgen, die länger als 160 Zeichen sind, wird der Button **ganz anzeigen** eingeblendet. Per Klick wird der vollständige Kommentartext angezeigt, ein erneuter Klick reduziert die Ansicht wieder auf die gekürzte Fassung.

Danach folgen Buttons, die immer verfügbar sind. **Anzeigen** öffnet bei einem Klick ein neues Fenster, in dem der Kommentar im Kontext des Artikels im Frontend dargestellt wird.

Bearbeiten öffnet eine Oberfläche, in der ein Kommentar von Ihnen bearbeitet und redaktionell verändert werden kann.

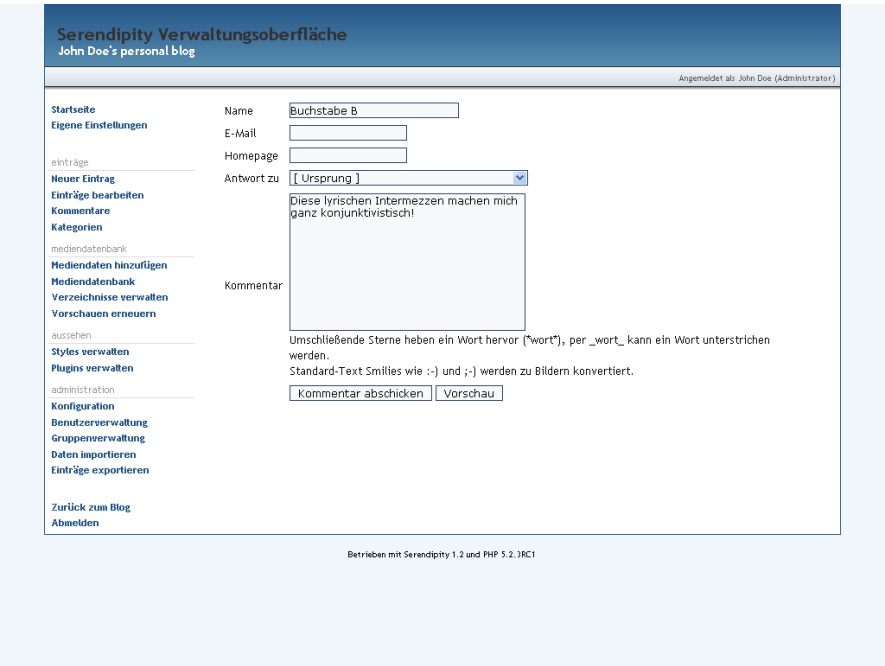

Abbildung 4.9: Einträge: Kommentar bearbeiten

In dieser Oberfläche können Sie wie beim Kommentieren im Frontend die bekannten Felder ändern, die mit den Eingaben des Besuchers vorausgefüllt sind. Ein Klick auf **Kommentar abschicken** speichert die von Ihnen überarbeitete Fassung des Kommentars. Achten Sie bei der Moderation von Kommentaren darauf, dass dies möglicherweise von Besuchern als Zensur aufgefasst werden könnte. Überarbeiten Sie also einen Kommentar wirklich nur dann, wenn Sie entweder darauf hinweisen oder es unabdingbar ist.

Der Button **Löschen** entfernt einen Kommentar vollständig aus der Datenbank. Wenn sich bereits andere Kommentare auf einen zu löschenden Kommentar beziehen (also im Frontend verschachtelt darunter dargestellt werden), wird ein Kommentar beim Löschen erst nur gekürzt. Der Text wird ausgeschnitten, aber die Grunddaten des Kommentars bleiben intakt, um die Verschachtelung nicht durcheinander geraten zu lassen. Erst wenn ein derartiger Kommentar ein zweites Mal gelöscht wird, werden die Bezüge zerstört und der Kommentar komplett entfernt.

Die Symbolleiste stellt als Letztes den Button **Antwort** zur Verfügung. Dies ermöglicht Ihnen, direkt zu einem Popup-Fenster zu springen, um den betreffenden Text direkt mit einer Antwort kommentieren zu können. Das Eingabefeld **Antwort zu** ist dabei korrekt mit dem betreffenden Kommentar vorausgefüllt, ebenso Ihr Name und Ihre E-Mail-Adresse.

Innerhalb jeder Kommentarbox befindet sich eine Ankreuzbox. Wenn diese ausgewählt wird, kann man die markierten Kommentare über den Button **Markierte Kommentare löschen** entfernen. Der Button **Auswahl umkehren** dreht die Markierungen um, so dass vorher ausgewählte Kommentare nun nicht mehr gewählt sind und stattdessen alle anderen. So lässt sich besonders leicht Spam herausfiltern, da eine ganze Seite mit dargestellten Kommentaren durch einen Klick auf **Auswahl umkehren** markiert und gelöscht werden kann.

Ebenfalls analog zu der Oberfläche **Einträge bearbeiten** kann die Darstellung der Kommentarübersicht nach bestimmten Kriterien gefiltert werden:

Autor
: In das Feld **Autor** können Sie den Namen eines Kommentators eingeben, nach dessen Kommentaren Sie suchen möchten. Dabei wird ein Autorenname unabhängig von Groß- und Kleinschreibung nach Teilwörtern durchsucht. Eine Suche nach `ann` wird also sowohl Kommentare von *Hanni* als auch von *Nanni* darstellen.

E-Mail
: Um nur Kommentare von Benutzern mit einer bestimmten E-Mail-Adresse anzuzeigen, können Sie deren Adresse in diesem Eingabefeld eintragen. Hierbei gilt dasselbe Suchmuster wie bei der Suche nach **Autor**.

URL
: Um nur Kommentare von Benutzern mit einer speziellen **URL**-Adresse anzuzeigen, können Sie diese URL eintragen. Hierbei gilt dasselbe Suchmuster wie bei der Suche nach **Autor**.

IP
: Um nur Kommentare von Benutzern mit einer speziellen **IP**-Adresse anzuzeigen, können Sie diese hier eintragen. Hierbei gilt dasselbe Suchmuster wie bei der Suche nach **Autor**. Bei der Angabe von IPs empfiehlt es sich daher, immer die vollständige Zahlenreihe einzugeben und nicht nur Teile davon.

Inhalt
: Hier können Sie Kommentare nach dem Auftreten bestimmter Wörter im Inhaltstext des Kommentars durchsuchen. Es gilt dasselbe Suchmuster wie bei der Suche nach **Autor**.

Referrer
: Die URL der Webseite, auf der sich ein Kommentator zuletzt befunden hat, kann in dem Eingabefeld **Referrer** eingetragen werden. Auch hier können Teilwörter wie bei der Suche nach **Autor** eingetragen werden.

Kommentare
: Das Auswahlfeld **Kommentare** ermöglicht es, mehr als nur 10 Kommentare pro Seite anzuzeigen. Die Auswahl reicht hier von 10, 20 und 50 bis hin zu allen Kommentaren. Achten Sie darauf, dass bei der Anzeige aller Kommentare die HTML-Seite sehr groß werden kann und Ihren Browser sehr lange beschäftigen könnte.

Zeige
: Über dieses Auswahlfeld kann man einstellen, ob man in der Kommentarübersicht noch freizuschaltende Kommentare/Trackbacks sehen möchte, nur freigeschaltete oder Kommentare/Trackbacks beiden Typs.

Typ
: Das Auswahlfeld schränkt ein, ob man nur **Kommentare**, nur **Trackbacks** oder beide Sorten von Kommentaren dargestellt bekommen möchte.

Diese Filteroptionen werden durch einen Klick auf **Los!** gesetzt und beim Blättern der Folgeseiten auch angewendet. Falls Sie das *Spamblock-Plugin* verwenden, können Sie übrigens durch einen Klick auf **Anti-Spam-Maßnahmen konfigurieren** direkt zu dessen Konfigurationsoberfläche springen.

Kommentarbenachrichtigungen

Wenn in Ihrem Blog ein Besucher einen neuen Kommentar hinterlässt, wird abhängig von den Einstellungen des Artikels und Ihrer **Eigenen Einstellungen** eine E-Mail an Sie gesendet, um Sie darüber zu benachrichtigen.

Innerhalb der E-Mail sehen Sie dann eine Übersicht über die Identität des Kommentators, den Kommentartext und einen Link zu dem Artikel in Ihrem Frontend.

Wenn Beiträge moderiert werden, müssen Kommentare von Ihnen erst freigeschaltet werden. Dazu dienen die drei Links am Ende der E-Mail: **Ansehen**, **Löschen** und **Bewilligen**. Diese Links führen direkt ins Frontend zu dem gewünschten Artikel und werden den Kommentar anzeigen, löschen oder freischalten. Dies funktioniert jedoch nur, wenn Sie sich vorher im Backend eingeloggt haben oder noch einen Login-Cookie besitzen. Andernfalls könnte jeder Besucher Ihres Blogs selbständig Kommentare freischalten.

Natürlich können Sie den Kommentar auch von der Kommentar-Verwaltungsoberfläche aus freischalten oder auch löschen.

Sobald ein Kommentar freigeschaltet wird, sucht Serendipity danach, ob sich andere Kommentatoren desselben Beitrags mit der Option **Bei Aktualisierung dieser Kommentare benachrichtigen** eingetragen haben. An alle diese Personen wird dann eine E-Mail verschickt, die sie über den Eingang eines neuen Kommentars informiert. Innerhalb der E-Mail befindet sich für jeden Empfänger die Möglichkeit, einen derart *abonnierten* Artikel auch wieder abzubestellen.

4.4.4 Kategorien

Im Bereich **Kategorien** können die für Artikel gültigen Kategorien erstellt und verändert werden.

Auf der Übersichtsseite sieht man die Auflistung aller verfügbaren Kategorien in ihrer hierarchischen Ordnung. Für jede Kategorie gibt es eine einzelne Zeile, und Kategorien können unendlich tief ineinander verschachtelt werden, um Unterkategorien zu ermöglichen.

Pro Zeile gibt es zwei Buttons zum **Bearbeiten** und **Löschen** einer Kategorie. Rechts daneben befindet sich ein (funktionsloses) Symbol für eine Kategorie sowie den Namen einer Kategorie. Eingerückt dahinter steht die Beschreibung einer Kategorie, und abschließend für jede Zeile sehen Sie den Namen des Eigentümers einer Kategorie.

Der Eigentümer einer Kategorie bestimmt im späteren Verlauf, welche Redakteure Artikel für diese Kategorie erstellen dürfen. Gewähren gewisse Rechtekonstellationen (siehe Kapitel 25 auf Seite 185) keinen Zugriff, wird die Kategorie auf dieser Übersichtsseite für unbefugte Redakteure nicht angezeigt und ist auch beim Erstellen eines Beitrags nicht auswählbar. Wenn an der Stelle des Eigentümers *Alle Autoren* steht, bedeutet dies, dass die Kategorie keinen speziellen Eigentümer besitzt, sondern für alle Redakteure zur Verfügung steht.

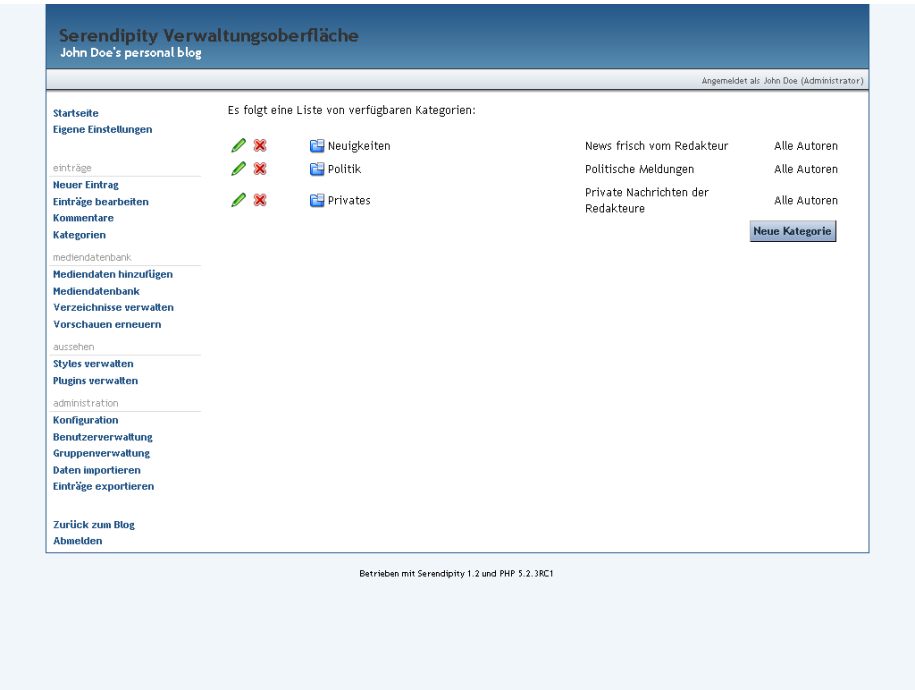

Abbildung 4.10: Einträge: Kategorien

Am Ende der Seite führt der Button **Neue Kategorie** zu der Erstellungsmaske für eine neue Kategorie.

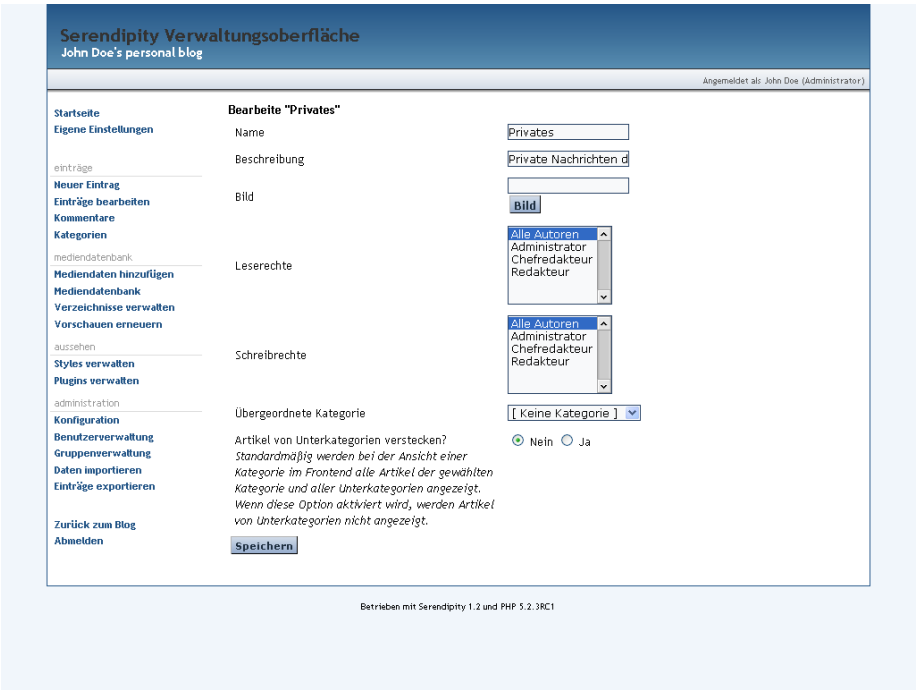

Abbildung 4.11: Einträge: Kategorien: Neue Kategorie

Sowohl der Klick auf **Neue Kategorie** als auch das **Bearbeiten** einer Kategorie führen zu derselben Maske. Dort können folgende Daten erfasst werden:

Name

Der Name einer Kategorie wird in das Feld **Name** eingetragen; sie wird später im Frontend jeweils mit diesem Namen dargestellt. Sonderzeichen und Leerzeichen sind erlaubt. Die Zeichenlänge ist grundsätzlich unbeschränkt, aber sehr lange Kategorienamen könnten zu problematischen Zeilenumbrüchen in Auswahlboxen führen.

Beschreibung

Die Beschreibung einer Kategorie wird an einigen Stellen im Backend und als Meta-Beschreibung zur Kategorie im Frontend angezeigt. Hier können Sie auch längere Beschreibungen eintragen.

Bild

Wenn Sie einen Artikel in einer Kategorie veröffentlichen, können Sie ein Bild mit dieser Kategorie verbinden. Dieses Bild wird dann bei der Darstellung eines Artikels im Frontend im Inhaltsbereich ausgegeben und ermöglicht den Besuchern eine einfache Assoziation des Textes mit einem Thema.

Über den Button **Bild** können Sie das Mediendatenbank-Popup (siehe Seite 107) auf-

rufen und die gewünschte Bilddatei einfügen. Alternativ tragen Sie in das Eingabefeld **Bild** eine vollständige URL mit dem Bildziel ein.

Leserechte

Wenn Sie einen Artikel in einer Kategorie verfassen, kann es sein, dass nicht jeder Besucher der Webseite Einträge aus dieser Kategorie lesen soll.

Über das Auswahlfeld **Leserechte** können Sie diejenigen Benutzergruppen wählen, die später im Frontend die Befugnis haben, einen Artikel zu lesen. Mehrere Gruppen können mit gedrückter *(Strg/Apfel)*-Taste und einem Mausklick gewählt werden.

Die Sondergruppe **Alle Autoren** wird benutzt, wenn eine Kategorie von jedem Besucher (also nicht eingetragenen Redakteuren) aufgesucht werden darf. Sobald eine Einschränkung auf eine Benutzergruppe eingerichtet wurde, muss ein Redakteur dieser Gruppe(n) sich erst ins Backend eingeloggt haben, bevor er im Frontend die Artikel lesen kann. Alle derart geschützten Artikel sind auch im normalen RSS-Feed nicht mehr vorhanden.

Die Option der Einschränkung von Leserechten kann durch die globale Serendipity-Konfigurationsoption **Leserechte auf Kategorien anwenden** (siehe Kapitel 4.7, Seite 155) ausgehebelt werden. Um im Seitenleisten-Plugin *Kategorien* (Seite 203) ebenfalls nur die Kategorien anzuzeigen, für die man Leserechte besitzt, muss in der Konfiguration dieses Plugins die Option **Quelle der Kategorien** auf **Derzeitiger Autor** eingestellt werden.

Schreibrechte

Analog zu den Leserechten können bei Serendipity die **Schreibrechte** einer Kategorie vergeben werden.

Nur die ausgewählten Benutzergruppen werden später die Möglichkeit haben, einen Artikel für diese Kategorie zu schreiben. Sobald eine Einschränkung des Schreibrechts auf mindestens eine Benutzergruppe vorgenommen wird, wird für die Kategorie der aktuelle Redakteur als *Eigentümer* der Kategorie vermerkt und in der Kategorieübersicht dargestellt. Die spezielle Option **Alle Autoren** bedeutet, dass der Schreibzugriff auf die Kategorie nicht eingeschränkt ist.

Übergeordnete Kategorie

Kategorien können beliebig verschachtelt werden. Um eine Kategorie einer Oberkategorie zuzuordnen, muss diese Kategorie im Auswahlfeld **Übergeordnete Kategorie** ausgewählt werden. In diesem Auswahlfeld sind alle bisher angelegten Kategorien hierarchisch (durch Leerzeichen eingerückt) dargestellt.

Beim Bearbeiten einer Kategorie kann diese somit leicht (mitsamt allen untergeordneten Kategorien) an einen anderen Punkt des Kategoriebaums *eingehängt* werden.

Artikel von Unterkategorien verstecken?

Wenn Sie im Frontend eine Kategorie zur Ansicht ausgewählt haben, werden standardmäßig alle Einträge dieser Kategorie und auch alle Einträge in den zugehörigen

Unterkategorien dargestellt. Dies ermöglicht dem Besucher, dass er bei der Auswahl von Oberkategorien nicht jede Unterkategorie einzeln anklicken muss.

Wenn Serendipity jedoch eher als Content-Management-System eingesetzt wird, ist dieses Verhalten recht untypisch und häufig nicht gewünscht. Daher können Sie für jede Oberkategorie die Option **Artikel von Unterkategorien verstecken** aktivieren. Daraufhin werden bei Auswahl einer derart konfigurierten Kategorie im Frontend ausschließlich die Artikel angezeigt, die auch in exakt dieser Kategorie eingetragen wurden.

Ein Klick auf **Erstellen** (für neue Kategorien) bzw. **Speichern** schließt die Erstellungs-/Bearbeitungsmaske und speichert die Änderungen.

Wenn Sie eine Kategorie löschen möchten, wird Serendipity Sie fragen, was mit Einträgen geschehen soll, die dieser Kategorie bisher zugeordnet waren. Mittels eines Auswahlfeldes können Sie festlegen, ob diese Einträge stattdessen einer anderen bestehenden Kategorie zugeordnet werden sollen. Ohne eine Neuzuordnung werden Artikel nicht gelöscht, sondern sind dann einfach keiner Kategorie mehr zugeordnet.

Einige Plugins benötigen zur Konfiguration die Kategorie-ID. Eine ID identifiziert eine Kategorie eindeutig und bleibt auch gleich, wenn Sie eine Kategorie einmal umbenennen. Diese ID wird üblicherweise weder im Frontend noch im Backend wirklich deutlich angezeigt; sie ist jedoch meist Bestandteil der URL (z. B. `/categories/17-Generelles`).

Um die ID ohne Nachschlagen in der Datenbank oder Durchsuchen des HTML-Quellcodes zu ermitteln, können Sie in der Kategorie-Übersicht im Backend-Bereich **Kategorien** mit der Maus über das Stift-Symbol fahren. Wenn Sie dann kurz warten, wird Ihr Browser unterhalb der Mausposition ein kleines Fenster anzeigen, in dem die ID der Kategorie steht.

4.5 Mediendatenbank

Serendipity kann von Ihnen hochgeladene Bilder und Dateien in einer eigenständigen Datenbank, der *Mediendatenbank*, verwalten.

Die Mediendatenbank besteht aus zwei Komponenten. Zum einen ist das ein Unterverzeichnis namens `uploads` im Serendipity-Stammverzeichnis. Dort werden die Dateien, die Sie von Ihrer eigenen Festplatte aus hochladen, abgespeichert. Zum anderen ist das eine Tabelle in Ihrer serverseitigen Datenbank, die sogenannte *Meta-Informationen* über die hochgeladenen Dateien speichert.

Diese Meta-Informationen enthalten Angaben über den Typ einer Datei, wann und von wem die Datei hochgeladen wurde sowie etwaige Beschreibungen der Datei.

Serendipity stellt nur die Dateien der Mediendatenbank dar, die auch in dieser Datenbanktabelle verzeichnet sind. Dateien, die lediglich manuell mittels FTP-Programm in das Verzeichnis `uploads` auf den Server geladen werden, sind dort vorerst nicht enthalten.

Aufgrund dieser zwei unterschiedlichen Komponenten ist es wichtig, dass Serendipity diese ständig miteinander *synchronisiert*, denn sonst könnte es passieren, dass Ihnen Dateien angezeigt werden, die gar nicht existieren – oder Ihnen könnten bereits hochgeladene Dateien fehlen.

Im Dateisystem der Mediendatenbank können sowohl Bilder (JPEG, GIF, PNG und weitere) verwaltet werden als auch beliebige andere Dokumente (MP3, AVI, ZIP, DOC). Diese Dateien können zur besseren Strukturierung auch in beliebig verschachtelten Unterverzeichnissen abgelegt werden.

Unterschiedliche Schreib- und Leserechte auf Bildordner können über die Meta-Informationen der Mediendatenbank verwaltet werden.

Bilder können in der Mediendatenbank speziell bearbeitet werden. So kann man die Bilder auf dem Server vergrößern/verkleinern, und Serendipity kann kleine Vorschaubilder (*Thumbnails*) einer Grafik erstellen. Für diesen Automatismus muss Ihr Webserver entweder *gdlib* oder *ImageMagick* unterstützen (siehe Kapitel 1.3 auf Seite 26). Trotz dieser rudimentären Grafikbearbeitungsmöglichkeiten ersetzt Serendipity nicht die Nachbearbeitung eines Bildes mit Programmen wie *GIMP* oder *Adobe Photoshop*. Ein Bild, das direkt von einer Digitalkamera kommt, ist üblicherweise viel zu groß für die Darstellung im Internet. Daher sollten Sie Bilder von vornherein auf eine angemessene Größe bringen. Statistisch liegen die verbreitetsten Auflösungen im Internet unter 1280 x 1024 Pixel, daher macht eine Datei mit einer höheren Auflösung als dieser nur in besonderen Fällen Sinn und würde nur zu viel Speicherplatz benötigen.

4.5.1 Mediendaten hinzufügen

Um Ihre Mediendatenbank mit Daten zu füllen, müssen Sie die Dateien in dieser Datenbank *anmelden*. Um eine Datei komfortabel über die Oberfläche hochladen zu können, gibt es den Menüpunkt **Mediendaten hinzufügen**.

Auf dieser Seite können Sie zwischen zwei Varianten wählen, von welcher Quelle eine Datei eingestellt werden soll.

Abbildung 4.12: Mediendatenbank: Mediendaten hinzufügen

Die erste Variante ist der Download einer Datei, die bereits im Internet unter einer URL verfügbar ist. Wenn Sie also auf einer Webseite eine Bilddatei sehen, können Sie über Ihren Browser (meist mittels eines Rechtsklicks auf die Grafik) die URL dieses Bildes heraussuchen und in das Feld **URL zum Download angeben** eintragen.

Achten Sie bitte beim Download einer Grafikdatei aus dem Internet immer darauf, dass die Bilder/Dateien urheberrechtlich geschützt sein könnten und die Einbindung einer solchen Datei in Ihrem Blog rechtlich nicht erlaubt ist. Stellen Sie daher immer sicher, dass Sie das Recht haben, eine Datei zu vervielfältigen.

Unterhalb der Eingabebox zum Download einer Datei aus dem Internet befindet sich das Auswahlfeld **Download-Methode**. Wenn Sie hier **Bild auf diesem Server speichern** auswählen, bedeutet das, dass die Daten der eingetragenen URL heruntergeladen und auf dem eigenen Webserver gespeichert werden. Wenn Sie ein solches Bild später einbinden, wird es von den Besuchern von Ihrem eigenen Webserver heruntergeladen und verursacht auf Ihrem Webserver Kosten für den aufgekommenen *Traffic* (Datenverkehr). Die zweite Möglichkeit stellt die Option **Nur zum Quellserver linken** dar. Ist diese Option gewählt, wird die von Ihnen eingetragene Datei nicht wirklich heruntergeladen, sondern nur ein Verweis auf den Zielserver gespeichert. Wenn Sie eine derartige Datei später einbinden, werden die Daten dann vom fremden Webserver geladen – dies nennt man *hotlinking*. Dies hat den Vorteil, dass auf Ih-

rer eigenen Seite kein erhöhter Datenverkehr entsteht und dass, formaljuristisch gesehen, die Einbindung solcher Bilder urheberrechtlich unterschiedlich gehandhabt wird, da Sie das Bild nicht eigenständig anbieten.[8] Der Nachteil einer solchen Einbindung ist jedoch, dass, wenn einmal der verwiesene Server nicht mehr betrieben oder die Datei entfernt wird, ein solches Bild natürlich auch in Ihrem Artikel nicht mehr angezeigt werden kann.

Als zweite Variante zum Hinzufügen einer Datei können Sie diese von Ihrer eigenen Festplatte auswählen. Wenn Sie auf den Button **Durchsuchen** neben der Eingabebox **Datei zum Hochladen angeben** klicken, öffnet sich ein Dateiauswahldialog Ihres Betriebssystems, und Sie können eine Datei aus Ihrer Verzeichnisstruktur auswählen.

Sie können immer nur eine dieser beiden Varianten des Hinzufügens wählen. *Entweder* Sie laden eine Datei aus dem Internet, *oder* Sie laden sie vom eigenen Computer. Wenn Sie beide Eingabefelder ausfüllen, wird nur die Datei aus dem Internet geladen und die selbst hochgeladene Datei ignoriert. Wenn Sie also sowohl eine Datei aus dem Internet als auch eine vom eigenen Computer hochladen wollen, müssen Sie dies nacheinander tun.

Wenn Sie eine Datei von der eigenen Festplatte hochladen, können Sie noch weitere Details bestimmen. Im Eingabefeld **Datei speichern mit dem Namen** können Sie einen Dateinamen für die hochgeladene Datei vergeben. Standardmäßig wird hier der Originalname der Datei von Ihrer Festplatte eingetragen. Im Feld **In diesem Verzeichnis ablegen** können Sie das Unterverzeichnis auswählen, in dem die Datei später gespeichert wird. Unterverzeichnisse werden über den Mediendatenbank-Menüpunkt **Verzeichnisse verwalten** (siehe Kapitel 14 auf Seite 136) erstellt.

Wenn Sie mehr als eine Datei hochladen möchten, können Sie auf den Button **Mehr Bilder hinzufügen** klicken. Jeder Klick auf diesen Button stellt ein zusätzliches Eingabefeld für eine auszuwählende Datei zur Verfügung, und in jedem dieser Blöcke können Sie danach auch den Dateinamen und das Zielverzeichnis bestimmen.[9] Beachten Sie bitte, dass Sie nicht zu viele Dateien in einem Schritt hochladen sollten. Je nach Einstellung des Webservers (siehe Kapitel 1.3.2 auf Seite 33) dürfen Sie ein gewisses Limit (meist 4MB) nicht überschreiten.

Nachdem Sie also die Dateien ausgewählt haben, haben Sie zwei Möglichkeiten der Speicherung. Ein Klick auf **Los!** speichert die Datei und erstellt ggf. automatisch kleine Voransichten. Wenn Sie jedoch auf **Los und Eigenschaften angeben** klicken, werden Sie auf einer Folgeseite gebeten, zu einer Datei optionale Meta-Informationen anzugeben. Diese Meta-Informationen können Sie auch später noch nachtragen oder überarbeiten, wenn sie in der Mediendatenbank bereits gespeichert sind.

4.5.2 Mediendaten: Probleme beim Upload

Scheitert das Speichern einer neuen Datei, liegt dies meist an falschen Zugriffsrechten (Kapitel 2.2.2 auf Seite 53). Stellen Sie sicher, dass Ihr Webserver Schreibzugriff auf den Ordner

[8] Diesen Hinweis genießen Sie bitte mit Vorsicht. Im Zweifelsfall sollten Sie den Eigentümer einer Datei immer um Erlaubnis bitten oder mit der Hilfe eines Fachanwalts den Sachverhalt klären.

[9] Diese Funktion benötigt aktiviertes JavaScript und wird von allen gängigen Browsern unterstützt.

`uploads` besitzt. Auch eine falsche Einstellung durch den PHP `SafeMode` kann Schreibprobleme verursachen.

Wenn eine Datei aufgrund der Größenbeschränkung des Webservers nicht hochgeladen werden kann, wird Serendipity darüber einen Hinweis geben.

Einige Webserver filtern zudem spezielle Dateien, um Sicherheitsprobleme zu verhindern. Wenn Sie also merkwürdige Fehlermeldungen erhalten, sollten Sie Webserver-Module wie `mod_security` oder `suhosin` in Betracht ziehen und gemeinsam mit dem Serverprovider anpassen.

Beim Upload großer Dateien kann es zudem auch passieren, dass Ihre Verbindung aufgrund der großen Transferdauer entweder sehr lange dauert oder sogar abbricht. In so einem Fall sollten Sie besonders große Dateien lieber mittels FTP-Programm auf den Server laden.

4.5.3 Mediendaten: Eigenschaften angeben

Wenn Sie beim Hochladen einer Datei auf den Button **Los und Eigenschaften angeben** geklickt oder innerhalb der Mediendatenbank-Übersicht auf das **Schraubstock**-Symbol einer Datei geklickt haben, landen Sie auf einer Seite, in der Sie die Eigenschaften einer Datei überarbeiten können.

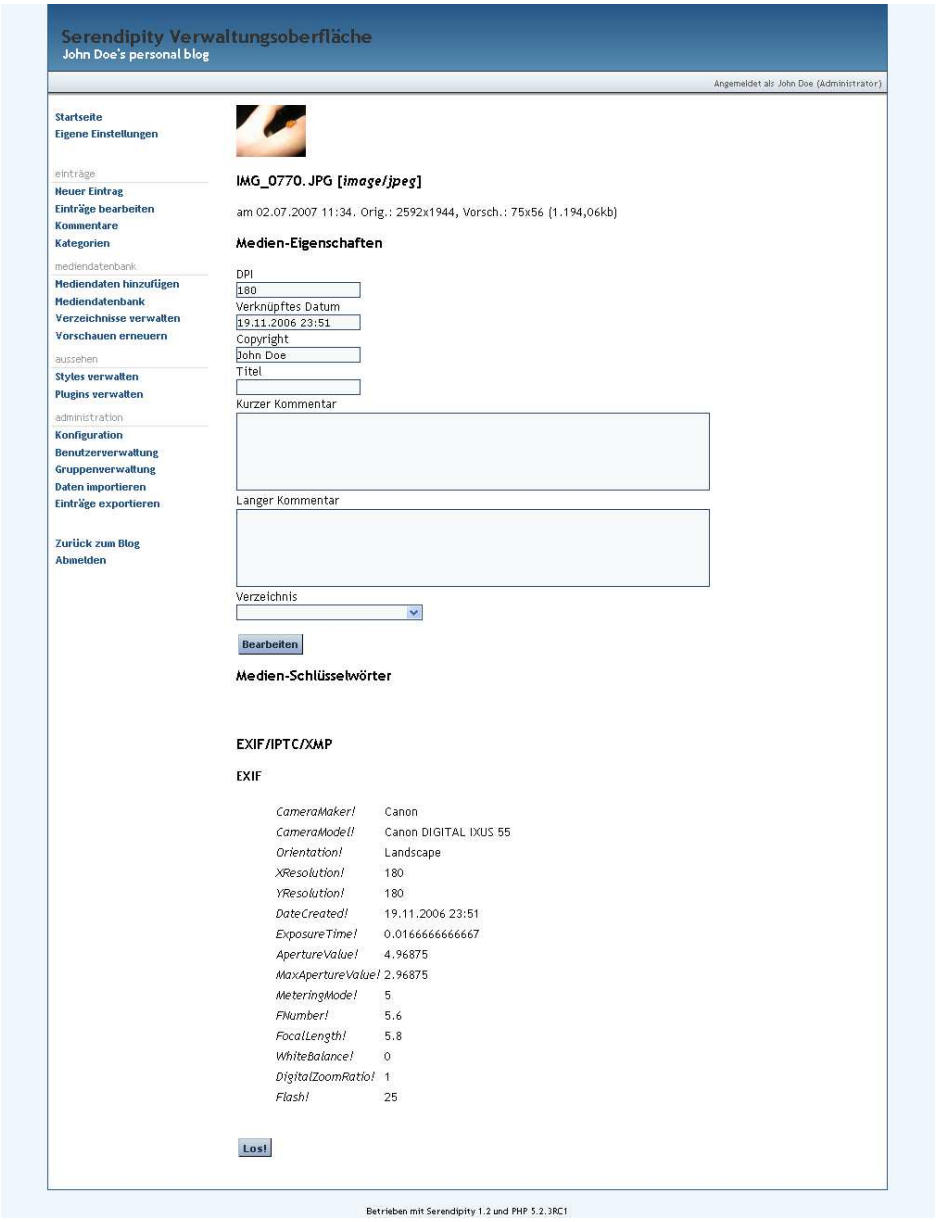

Abbildung 4.13: Mediendatenbank: Eigenschaften angeben

Diese Seite stellt die betreffenden Meta-Informationen einer Datei dar. Als Erstes wird dabei entweder das Vorschaubild einer Grafikdatei oder ein Dateityp-Icon von Nicht-Grafikdateien angezeigt.

Unterhalb dieses Icons sehen Sie den Namen und den Typ der gewählten Datei. Direkt darunter wird aufgeführt, wer die Datei zu welchem Datum hochgeladen hat und wie groß die Datei ist. Im Falle einer Grafikdatei wird hier zudem die Auflösung der Grafik und die Auflösung des kleineren Vorschaubildes angezeigt.

Es folgt der Block mit den **Medien-Eigenschaften**. Auch hier werden abhängig vom Dateityp gewisse Zusatzinformationen angezeigt. Bei Bildern wird die **DPI**-Zahl[10] angegeben. Bei Video- und Audiodateien wird die Länge der Datei dargestellt. Serendipity versucht die DPI-Zahl und die Spieldauer einer Datei automatisch auszulesen, jedoch kann dies bei den vielen unterschiedlichen Dateiformaten manchmal misslingen. Daher können Sie diese Angaben auch manuell korrigieren und in die Eingabefelder eintragen.

Für alle Dateitypen werden die folgenden Felder angezeigt:

Verknüpftes Datum
: Jede Datei kann einen Zeitstempel enthalten, wann sie erstmals gespeichert wurde. Bei Bildern, Videos oder Musikstücken entspricht dies oft dem Aufnahmedatum. Diese Information versucht Serendipity anhand der Datei automatisch zu erkennen. Sollten Sie das Datum ändern wollen, können Sie es (beliebig formatiert) hier eintragen.

Copyright
: Das Feld **Copyright** kann Urheberrechtsinformationen für die jeweilige Datei enthalten.

Titel
: Der **Titel** einer Datei wird vom Redakteur manuell festgelegt.

Kurzer Kommentar

Langer Kommentar Eine kurze sowie eine ausführliche Beschreibung einer Datei können Sie in diesen beiden Eingabefeldern eintragen.

Verzeichnis
: Das Auswahlfeld **Verzeichnis** bestimmt, in welchem Verzeichnis eine Datei gespeichert wird. Wenn Sie später eine Datei verschieben wollen, können Sie das zugeordnete Verzeichnis an dieser Stelle ändern.

  Bei der Änderung werden die Datei sowie das Vorschaubild in das gewünschte Verzeichnis auf dem Server physikalisch verschoben, der alte Speicherort existiert dann also nicht mehr. Serendipity versucht automatisch alle Ihre Artikel anzupassen, die das verschobene Bild referenziert haben.[11] Prüfen Sie also bitte sicherheitshalber, ob alle

[10] Die DPI-Zahl (Dots Per Inch) eines Bildes bestimmt, wie fein die Datei beim Ausdruck gerastert wird. Je höher die Zahl ist, desto mehr Pixel passen auf ein Inch und desto feiner wird eine Grafik dargestellt. Druckgrafiken haben meist 300 DPI und Dateien, die nicht für den Druck vorgesehen sind, 72 DPI.

[11] Dies funktioniert nur bei Einsatz des MySQL-Datenbanksystems und nicht bei PostgreSQL oder SQLite, da dafür reguläre Ausdrücke zur Datenbankabfrage eingesetzt werden müssen, die nur bei MySQL verfügbar sind.

Ihnen bekannten Verweise auf die Datei nach dem Verschieben noch stimmen. Generell empfiehlt es sich, nachdem Sie eine Datei bereits eingebunden haben, den Speicherort der Datei nicht mehr zu verändern.

Bearbeiten

Der Button **Bearbeiten** öffnet ein spezielles Popup-Fenster, in dem eine Originalgrafik beschnitten werden kann. Dieses Fenster zeigt zwar bereits korrekt eine Originaldatei und die Oberfläche zum Beschneiden an, die Funktionalität dazu ist aber in Serendipity zum jetzigen Zeitpunkt noch nicht implementiert. Sehen Sie dies also als einen Vorgeschmack auf Dinge, die noch kommen werden.

Diese Felder können später von Redakteuren in der Mediendatenbank eingesehen und durchsucht werden und stehen auch in speziellen Galerie-Ansichten (siehe Seite 687) für normale Besucher zur Verfügung. Sie können auch individuell weitere Meta-Informationsfelder für Dateien erfassen. Dies wird über die Option **Medien-Eigenschaften** in der Serendipity-Konfiguration (siehe Kapitel 4.7 ab Seite 155) eingestellt.

Der Abschnitt *Medien-Schlüsselwörter* kann für eine Datei vorher festgelegte Schlüsselwörter angeben. Die verfügbaren Schlüsselwörter werden dabei ebenfalls an zentraler Stelle der Serendipity-Konfiguration (s. o.) über die Option **Medien-Schlüsselwörter** festgelegt.

Jedes Schlüsselwort kann über die Aktivierung der links daneben stehenden Auswahlbox aktiviert werden. Schlüsselwörter machen es später leichter, eine Datei mit bestimmten Kriterien über eine Suche in der Mediendatenbank wieder aufzufinden.

Am Ende der Seite befindet sich im Abschnitt **EXIF/IPTC/XMP** eine von der Datei abhängige Liste an Datei-Angaben. Das *EXIF*-Format stellt spezielle Angaben wie das Erstellungsdatum, Angaben über die verwendete Digitalkamera, Blendenwerte, involvierte Software und Weiteres innerhalb einer Datei zur Verfügung und kann von Serendipity daher ausgelesen werden. Da diese Angaben fest in der Datei gespeichert sind, können sie von Ihnen nicht bearbeitet werden, sondern dienen nur der Information. Moderne Digitalkameras speichern diese Informationen in den erstellten Bilddateien automatisch ab.

Sollte eine Datei über die Mediendatenbank im Frontend speziell eingebunden werden (siehe Seite 688), können Sie in einem weiteren Abschnitt alle Verweisquellen aufgelistet sehen. Die dort aufgeführten Webseiten stellen also Internet-Seiten dar, auf denen das aktuelle Bild dargestellt wird. So können Sie leicht herausfinden, ob ein Bild überhaupt eingesetzt wird oder ob Sie es problemlos löschen/verändern können.

4.5.4 Mediendatenbank: Übersicht

Über den Menüpunkt **Mediendatenbank** wird die zentrale Übersicht aller in der Mediendatenbank eingetragenen Dateien aufgerufen.

Diese Seite haben Sie möglicherweise auch schon im Kapitel 4 auf Seite 107 gesehen, dort jedoch noch ohne Bearbeitungsoptionen für jede Datei.

Die Übersicht besteht aus einem Kopfbereich zur Suche in der Mediendatenbank und darunter einer zweispaltigen Ansicht aller von Ihnen hochgeladenen Dateien.

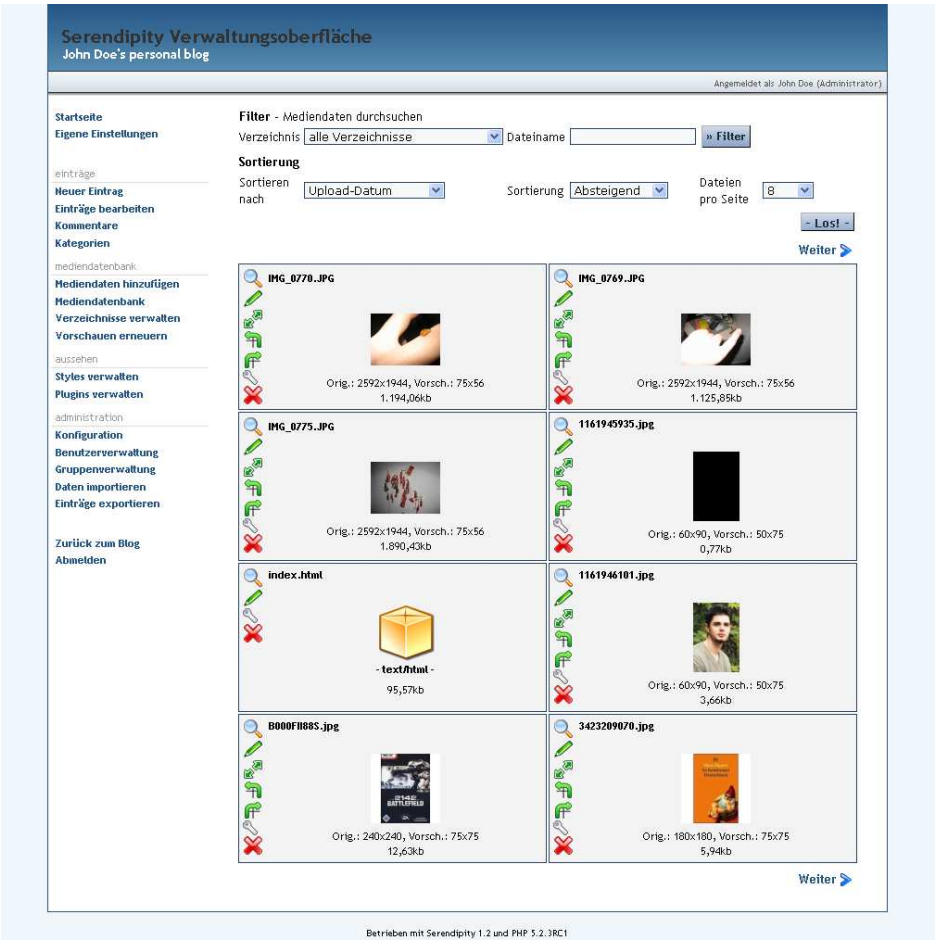

Abbildung 4.14: Mediendatenbank: Übersicht

Die Seite stellt standardmäßig jeweils acht Dateien pro Seite dar und kann über die Buttons **Weiter** und **Zurück** (in einem Bereich vor und nach der Dateiübersicht) geblättert werden.

In dem zweispaltigen Bereich wird pro Box ein Objekt aus der Mediendatenbank dargestellt, mit den jeweiligen Informationen und Optionen dieses Objekts:

Symbolleiste

Die **Symbolleiste** am linken Rand jeder Infobox stellt die möglichen Funktionen für eine Datei dar. Abhängig davon, ob eine Datei ein Bild oder etwas anderes darstellt, bietet die Symbolleiste unterschiedliche Optionen an.

Die **Lupe** öffnet eine Ansicht der Originaldatei in einem separaten Popup-Fenster. Bei Bildern wird hierin passgenau das Originalbild angezeigt, bei anderen Dateitypen wie PDF-Dokumenten wird das damit assoziierte Programm auf Ihrem Computer gestartet.

Das **Stift**-Icon ermöglicht Ihnen, eine Datei umzubenennen. Ein Popup-Fenster wird Sie nach dem neuen Dateinamen fragen. Beim Umbenennen der Datei müssen Sie darauf achten, dass, sofern die Datei bereits in Blog-Artikeln referenziert wird, Sie den Namen der Datei dort ebenfalls anpassen müssen, damit er dem neuen entspricht. Beim Umbenennen einer Datei kann nur der Stammname verändert werden und nicht die Dateiendung.

Bei Bilddateien sehen Sie nun drei weitere Buttons: Das Symbol der **Zwei Pfeile** ermöglicht es, ein Bild zu vergrößern oder zu verkleinern. Bei einem Klick darauf öffnet sich eine Seite, die das Originalbild und dessen Originalauflösung darstellt, und Sie können in einer Texteingabebox die neue Auflösung eintragen. Die Auswahlbox **Proportionen beibehalten** kann hier aktiviert werden und bewirkt, dass das Seitenverhältnis einer Datei bei der Verkleinerung beibehalten wird. Die beiden weiteren Buttons in der Symbolleiste (**Rotation nach links**, **Rotation nach rechts**) drehen ein Bild im oder gegen den Uhrzeigersinn jeweils um 90 Grad. Sie können daher ein im Querformat hochgeladenes Bild ins Hochformat umwandeln und umgekehrt.

Die letzten beiden Buttons sind nun wieder für jeden Dateityp identisch. Der **Schraubenschlüssel** öffnet die Medien-Eigenschaften einer Datei (siehe Seite 128). Das Icon mit dem **roten Kreuz** kann eine Datei aus der Mediendatenbank und vom Server löschen. Wenn die zu löschende Datei in einem Blog-Artikel referenziert wird, kann sie dort dann selbstverständlich nicht mehr angezeigt werden.

Dateiname
: Der Name einer Datei (mit Dateiendung) wird rechts neben der Symbolleiste angezeigt.

Eigentümer
: Direkt unterhalb des Dateinamens finden Sie den Namen des Autors, der dieses Bild hochgeladen hat.

Vorschaubild/Icon
: Bei Bildern sehen Sie in der Mitte der Übersichtsbox ein Vorschaubild. Bei anderen Dateien kann hier ein Symbol für das jeweilige Dateiformat stehen.

Dateigröße
: Unterhalb des Vorschaubilds sehen Sie die Dateigröße (in Kilobyte) und ggf. die Auflösung einer Bilddatei.

Üblicherweise richten sich die dargestellten Dateien nach deren chronologischem Hochlade-Zeitpunkt. Das erste dargestellte Bild ist also das, was zuletzt hochgeladen wurde – ganz gleich, in welchem Verzeichnis die Datei abgelegt wurde. Diese Reihenfolge und die Kriterien, nach denen Dateien angezeigt werden, können Sie im Kopfbereich anpassen:

Filter: Verzeichnise

Im Auswahlfeld **Verzeichnis** können Sie wählen, aus welchem Unterverzeichnis Dateien dargestellt werden sollen.

Filter: Dateiname

Wenn Sie die Mediendatenbank nach einem ganz speziellen Dateinamen durchsuchen wollen, können Sie diesen in dem Eingabefeld **Dateiname** eintragen. Dabei wird die Suche später auch nach Teilwörtern und ohne Berücksichtigung der Groß- und Kleinschreibung ausgeführt. So könnten Sie beispielsweise nach allen PDF-Dateien suchen, indem Sie dort `.pdf` eintragen.

Filter

Die beiden vorangestellten Pfeile vor dem Button **Filter** deuten darauf hin, dass ein Klick einen Unterbereich im Filtermenü ausklappt. Dieser belegt recht viel Bildschirmplatz und wird standardmäßig ausgeblendet. Sie finden innerhalb dieses eingeblendeten Bereichs zahlreiche Filtermethoden, die Sie ausfüllen können, um Dateien nach bestimmten Kriterien zu suchen.

Alle ausgefüllten Filterkriterien werden dabei miteinander so verkettet (*UND-Verkettung*), dass nur Dateien angezeigt werden, auf die *jedes* Kriterium zutrifft. Wenn Sie daher nach Dateien suchen wollen, die mehreren unterschiedlichen Kriterien entsprechen könnten, müssen Sie dafür jeweils erneut eine Suche ausführen.

Filter: Medien-Schlüsselwörter

In das Eingabefeld **Medien-Schlüsselwörter** können Sie, mit Semikolon getrennt, mehrere Schlüsselwörter eingeben, die auf ein Bild zutreffen müssen. Rechts neben dem Eingabefeld sehen Sie die Liste aller verfügbaren Schlüsselwörter, die Sie auch direkt durch Anklicken in das Eingabefeld übernehmen können.

Filter: Upload-Datum

Um nur Dateien anzuzeigen, die in einem gewissen Zeitraum eingestellt worden sind, können Sie das Start- und Enddatum in das Feld **Upload-Datum** eintragen. Das Dateiformat kann dabei der Syntax `Tag.Monat.Jahr` oder `Jahr-Monat-Tag` oder `Monat/Tag/Jahr` entsprechen – die Angabe einer Uhrzeit ist nicht möglich. Wenn Sie nur das Startdatum eintragen, werden alle Dateien seit diesem Zeitpunkt bis heute angezeigt. Ist nur das Enddatum ausgefüllt, werden alle Dateien bis zu diesem Zeitpunkt angezeigt.

Filter: Dateiname

Die Filterung nach einem Dateinamen konnten Sie bereits oberhalb des Filter-Bereichs einstellen, aber er wird hier der Vollständigkeit halber nochmals aufgeführt.

Filter: Autor

Wenn Sie nach Dateien suchen, die ein bestimmter Autor hochgeladen hat, können Sie diesen aus dem Auswahlfeld **Autor** aussuchen.

Filter: Dateiendung
Wollen Sie nach einem Dateinamen *und* einer Dateiendung suchen, reicht es nicht aus, nur das Feld **Dateiname** auszufüllen, da Sie dort ja nur nach einer einzigen Teilzeichenkette suchen. Daher können Sie zusätzlich im Feld **Dateiendung** nach einer Endung suchen und **Dateiname** für den eigentlichen Dateityp verwenden. Um z. B. nach allen PDF-Dateien mit der Zeichenkette `hund` im Dateinamen zu suchen, tragen Sie `pdf` in **Dateiendung** ein und `hund` in **Dateinamen**. So werden `Schäferhund.pdf` wie auch `Hundekuchen.pdf` gefunden. Hätten Sie für den Dateinamen nur `hund.pdf` eingetragen, wäre nur `Schäferhund.pdf` gefunden worden!

Filter: Dateigröße
Die **Dateigröße** kann ähnlich wie das Upload-Datum in einem `von-bis`-Bereich angegeben werden. So können Sie alle Dateien zwischen `X` und `Y`kb Größe anzeigen.

Filter: Bildbreite

Filter: Bildhöhe Auch die **Bildbreite** und **Bildhöhe** lassen sich hier in Pixeln in einem minimalen und maximalen Bereich angeben. Wenn Sie also nach besonders kleinen oder großen Bildern suchen wollen, können Sie diese Einschränkung hier vornehmen.

Filter: DPI
Die DPI-Angaben einer Bilddatei lassen sich im Eingabefeld **DPI** filtern. Sollte man also nach einer druckfähigen Bilddatei suchen, könnte man hier `300` angeben.

Filter: Laufzeit
Bei einer Video- oder Audio-Datei könnte man im Eingabefeld **Laufzeit** den Bereich der Spieldauer eintragen. Die Angabe erfolgt dabei in Sekunden.

Filter: Verknüpftes Datum
Wenn Sie nach einer Datei suchen, die zu einem speziellen Zeitpunkt erstellt wurde und in den Meta-Daten einer Datei festgehalten wurde, können Sie im Eingabefeld **Verknüpftes Datum** danach suchen. Die zeitliche Einschränkung wird im selben Format vorgenommen wie bei der Filterung nach **Upload-Datum**.

Filter: Copyright
Suchen Sie Dateien mit einem speziell hinterlegten Urheber, können Sie im Eingabefeld **Copyright** danach suchen. Teilwörter werden dabei auch akzeptiert. Achten Sie darauf, dass das Copyright ein vom Redakteur optional ausgefülltes Informationsfeld ist und daher nicht zwingend bei jeder Datei verfügbar ist.

Filter: Titel, Kurzer Kommentar, Langer Kommentar
Die letzten drei Filterfelder **Titel**, **Kurzer Kommentar** und **Langer Kommentar** können ebenfalls Teilwörter enthalten, nach denen Sie in den entsprechend vom Redakteur ausgefüllten Meta-Feldern suchen können.

Die verfügbaren Filter-Felder der Meta-Informationen lassen sich mittels der Option **Medien-Eigenschaften** in der Serendipity-Konfiguration (siehe Kapitel 4.7 ab Seite 155) einstellen. Alle dort festgelegten Felder können als Filterkriterium dienen.

Unterhalb des Filter-Bereichs können Sie die Sortierung der Anzeige beeinflussen:

Sortieren nach
: Über das Auswahlfeld **Sortieren nach** stellen Sie ein, wie die Bilder innerhalb der Übersicht sortiert werden. Zur Auswahl steht das **Upload-Datum**, der **Dateiname** (alphabetisch), der **Autorname** (alphabetisch), die **Dateiendung** (alphabetisch), die **Bildbreite** oder **Bildhöhe** und zuletzt die Sortierung nach allen konfigurierten Meta-Feldern.

Sortierung
: Die Sortierungsreihenfolge (**absteigend** oder **aufsteigend**) wird über das Auswahlfeld **Sortierung** festgelegt.

Dateien pro Seite
: Die letzte Option der Suche gibt an, wie viele **Dateien pro Seite** in der Übersicht angezeigt werden sollen. Die Auswahl ist auf **8, 16, 50** oder **100** Dateien begrenzt.

Ein Klick auf den Button **Los!** wird die gewünschten Filterkriterien auswerten und danach alle Suchergebnisse in der gewohnten blätterbaren Übersicht darstellen.

4.5.5 Verzeichnisse verwalten

Ähnlich wie die Verwaltung der Kategorien (siehe Kapitel 8 auf Seite 120) können Schreib- und Leserechte von Mediendatenbank-Dateien verwaltet werden.

Hinter dem Menüpunkt **Verzeichnisse verwalten** sehen Sie eine Übersicht aller Unterverzeichnisse der Mediendatenbank (`uploads`-Verzeichnis im Serendipity-Stammverzeichnis).

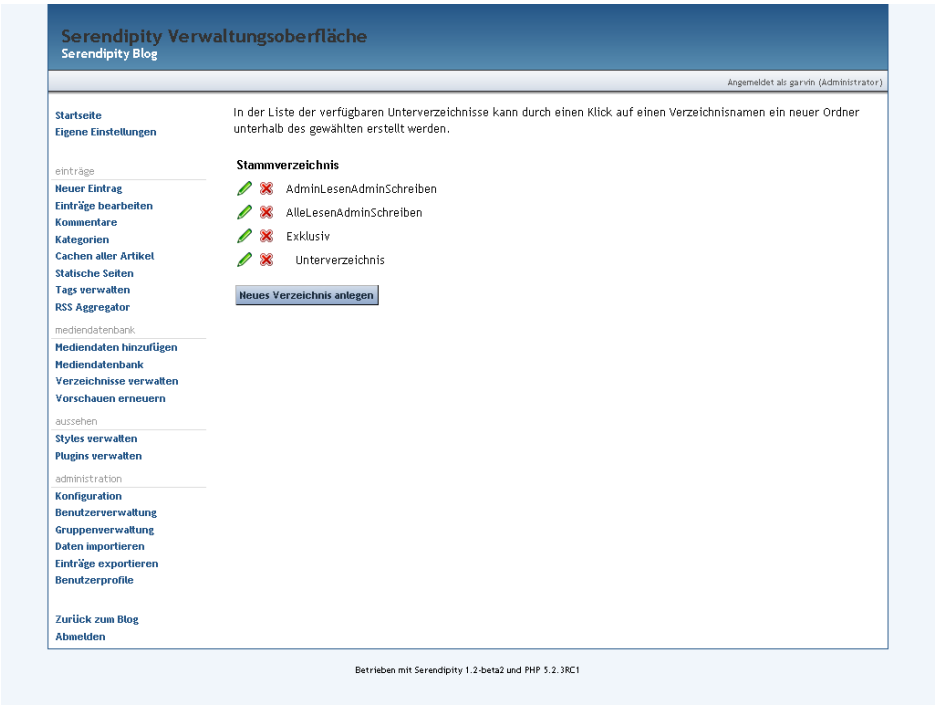

Abbildung 4.15: Mediendatenbank: Verzeichnisse verwalten

Neben allen Verzeichnissen, die jeweils wie im Dateisystem verschachtelt untereinander angezeigt sind, sehen Sie zwei Buttons zum **Bearbeiten** und **Löschen** eines Verzeichnisses.

Ein Klick auf den Button **Neues Verzeichnis anlegen** öffnet eine Seite, in der Sie den **Namen** eines neuen Verzeichnisses sowie dessen übergeordnetes **Stammverzeichnis** aus einem Auswahlfeld wählen können. Wenn Sie danach auf **Verzeichnis anlegen** klicken, wird Serendipity versuchen, das neue Verzeichnis zu erstellen. Damit dies erfolgreich durchgeführt werden kann, müssen die Zugriffsrechte (siehe Seite 53) korrekt eingerichtet sein.

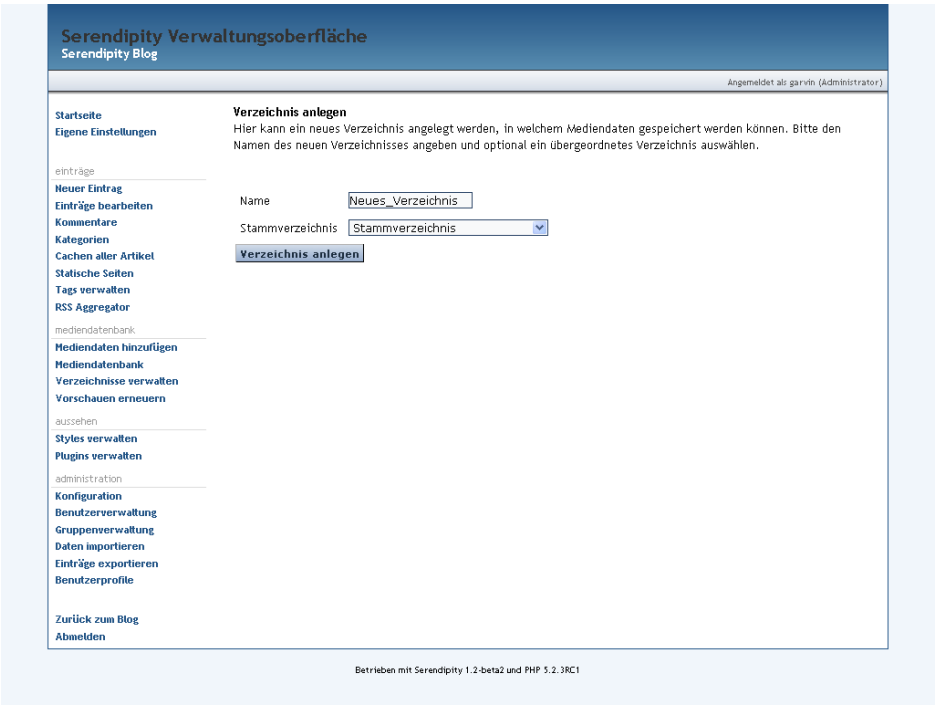

Abbildung 4.16: Mediendatenbank: Verzeichnisse verwalten: Neues Verzeichnis anlegen

Ein erstelltes Verzeichnis kann bearbeitet werden, indem Sie auf das **Stift**-Icon klicken. In dieser Maske sehen Sie zum einen den **Namen** des Verzeichnisses und zum anderen zwei größere Auswahlfelder für die **Leserechte** und die **Schreibrechte**.

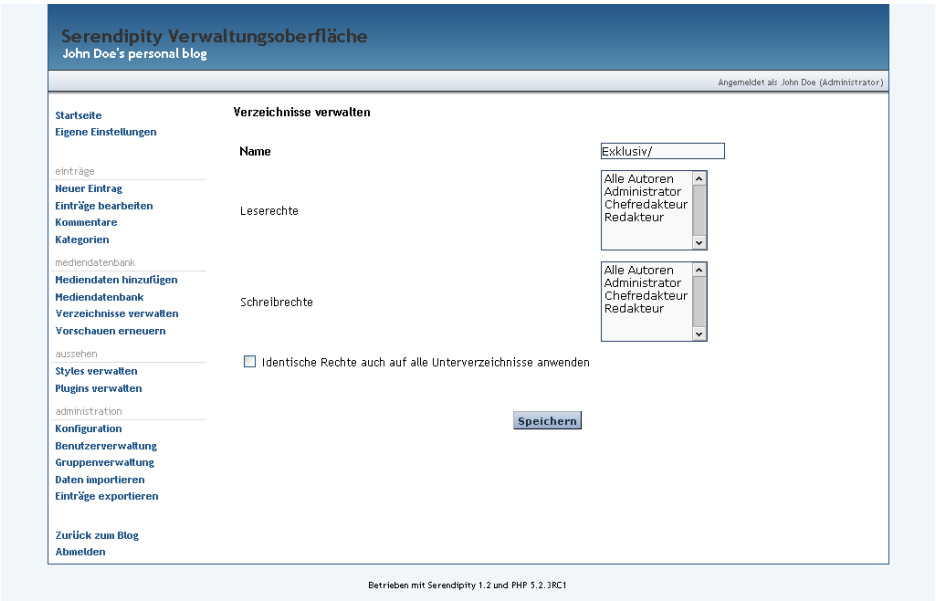

Abbildung 4.17: Mediendatenbank: Verzeichnisse verwalten: Verzeichnis bearbeiten

Diese Zugriffsrechte bestimmen ausschließlich die Zugriffsmöglichkeiten innerhalb Serendipitys. Wenn Ihre Redakteure auch per FTP Zugriff auf die Dateien der Mediendatenbank haben, entzieht sich dies Serendipitys Zugriffsmöglichkeiten.

Die Zugriffsrechte in der Mediendatenbank werden auch nicht herangezogen, wenn eine in einem Artikel eingebundene Datei im Frontend angezeigt wird. Der Link für eine eingebundene Datei zeigt direkt auf den Dateinamen und kann außerhalb von Serendipity über den normalen Webserver heruntergeladen werden. Serendipity selbst ist also beim Download der Datei nicht involviert und kann daher auch keine Zugriffsrechte auswerten.[12] Einzig das Mediendatenbank-Popup und weitere Funktionalitäten für Redakteure werten später die Zugriffsrechte aus und zeigen entsprechend der Rechte eines Redakteurs die ihm verfügbaren Optionen an.

Hat ein Redakteur kein Schreibrecht auf ein Verzeichnis, gilt dies auch für alle darin enthaltenen Dateien. Solche Dateien könnte ein Redakteur daher später nicht löschen. Wenn ein Redakteur auch keine Leserechte in einem Verzeichnis besitzt, wird er derartige Bilder in der Mediendatenbank-Ansicht erst gar nicht angezeigt bekommen.

Die Lese- und Schreibrechte können jeweils für mehrere Benutzergruppen definiert werden, indem Sie die *(Strg/Apfel)*-Taste gedrückt halten und anklicken. Die Option **Alle Autoren** gibt ein Verzeichnis für alle Redakteure frei.

Alle Lese- und Schreibrechte gelten ausschließlich für das aktuell bearbeitete Verzeichnis.

[12] Um Bilder *nicht* über den vollständigen URL-Pfad auszugeben und so Zugriffsrechte abzufragen, gibt es jedoch einen Trick, der auf Seite 688 erklärt wird.

Etwaige Unterverzeichnisse müssen alle separat konfiguriert werden. Dadurch ist es später möglich, zwar ein Stammverzeichnis für Redakteure zu sperren, aber Unterverzeichnisse gezielt wieder zugänglich zu machen. Wenn Sie im Nachhinein bei einer bestehenden Verzeichnisstruktur Rechte vergeben, möchten Sie vielleicht gerne diese Rechte auch für alle Unterverzeichnisse identisch vergeben. Dafür müssen Sie das Ankreuzfeld **Identische Rechte auch auf alle Unterverzeichnisse anwenden** aktivieren.

Um ein Verzeichnis zu löschen, klicken Sie auf das **rote Kreuz**. Sie gelangen danach auf eine Unterseite, die Sie nochmals fragt, ob Sie das Verzeichnis inklusive aller Dateien löschen (sowohl in der Mediendatenbank als auch im Dateisystem) wollen. Wenn Sie einmal Dateien mittels FTP in das zu löschende Verzeichnis (oder eines seiner Unterverzeichnisse) hochgeladen haben, kann es sein, dass sich darin Dateien befinden, die die Mediendatenbank noch nicht synchronisiert/importiert hat. Standardmäßig löscht Serendipity jedoch nur alle Dateien, die in der Mediendatenbank auch verzeichnet sind. Wenn Sie die Option **Alle Dateien dieses Verzeichnisses löschen** aktivieren, wird Serendipity auch alle dem System unbekannten Dateien zu löschen versuchen. Auch hier gilt wieder, dass Serendipity nur Dateien löschen kann, auf die es auch Schreibzugriff hat.

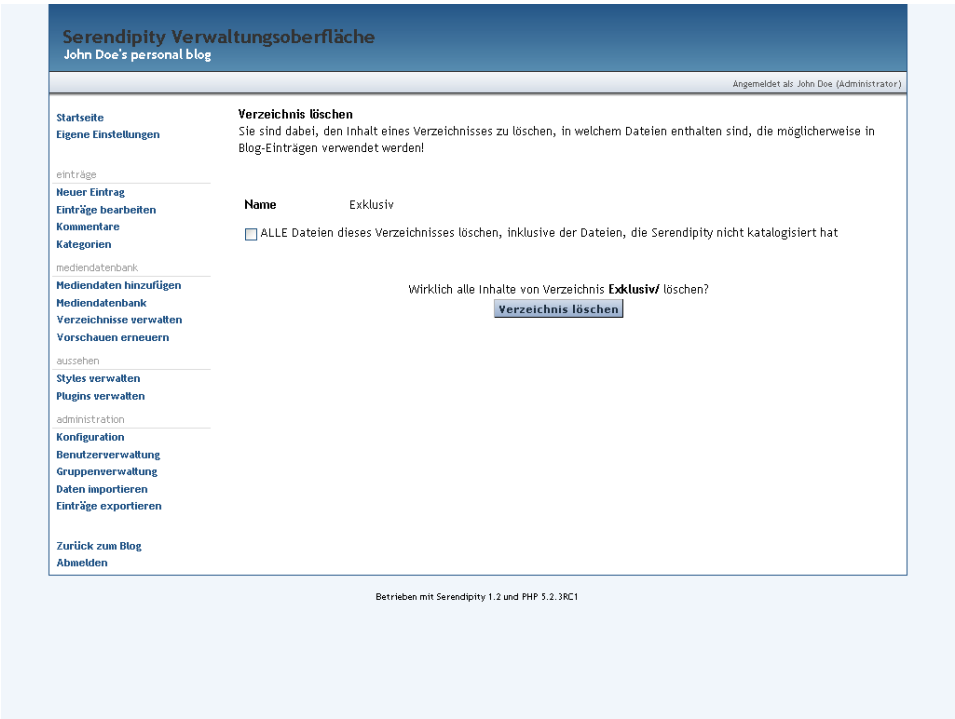

Abbildung 4.18: Mediendatenbank: Verzeichnisse verwalten: Verzeichnis löschen

4.5.6 Vorschauen erneuern

Wie am Anfang des Kapitels 4.5 auf Seite 124 erklärt, muss Serendipity zwischen den Dateien innerhalb der Verzeichnisstruktur auf dem Server und einer Datenbanktabelle mit Meta-Informationen synchronisieren.

Wenn Sie also Dateien mittels FTP in das Mediendatenbank-Verzeichnis `uploads` hineinkopieren, wird Serendipity davon erst einmal nichts mitbekommen. Damit Serendipity einen Abgleich zwischen Dateien auf dem Server und der Mediendatenbank durchführen kann, nutzen Sie den Menüpunkt **Vorschauen erneuern**.

Wenn Sie auf diesen Menüpunkt klicken, wird Serendipity alle Dateien und Unterverzeichnisse des `uploads`-Verzeichnisses durchgehen und prüfen, ob die Datei bereits (oder noch) in der Mediendatenbank vorhanden ist. Etwaige gelöschte Dateien auf dem Server werden daraufhin aus der Mediendatenbank entfernt und neue Dateien importiert.

Für jede gelöschte oder neu hinzugefügte Datei wird daraufhin ein Hinweis auf die durchgeführte Aktion ausgegeben. Die Meldung `Fertig (xx Bilder synchronisiert)` am Ende des Vorgangs gibt genaue Information darüber, wie viele Dateien bearbeitet wurden.

Fehlerhafte und nicht lesbare Dateien werden ebenfalls in dieser Ausgabe angezeigt, so dass Sie diese Dateien möglicherweise manuell via FTP löschen oder verändern müssen.

Wenn Sie besonders viele Dateien und Verzeichnisse in der Mediendatenbank gespeichert haben, kann die Ausführung dieser Funktion möglicherweise sehr lange dauern oder gar zu Server/PHP-Timeouts führen. Wenn dies passiert, sollten Sie entweder überlegen, alte Dateien in der Mediendatenbank zu löschen, oder Sie können alternativ in der Serendipity-Konfiguration die automatische Synchronisierung der Mediendatenbank aktivieren (siehe Seite 172). Sobald die automatische Synchronisierung aktiviert ist, werden extern hochgeladene Dateien beim Betrachten der Mediendatenbank automatisch importiert. Dies beansprucht weniger Zeit, da bei einem etwaigen Verbindungsabbruch an der Stelle fortgefahren werden kann, bei der zuletzt abgebrochen wurde.

Ein weiteres Feature der Funktion **Vorschauen erneuern** ist, dass alle Vorschaugrafiken von Bildern überprüft und ggf. neu erstellt werden. Falls Sie also versehentlich eine Vorschaugrafik namens `.serendipityThumb` auf dem Server gelöscht haben, kann diese Datei neu erstellt werden. Wenn Sie in der Serendipity-Konfiguration (siehe Seite 170) die Auflösung der Vorschaugrafiken einmal verändert haben (standardmäßig 110 Pixel), könnten sämtliche Vorschaubilder mit der neuen Auflösung neu berechnet werden.

Die automatische Synchronisierung kann Änderungen in der Vorschaubild-Auflösung nicht automatisch durchführen. Wenn also Ihre Mediendatenbank zu groß geworden ist, um die Ausführung der Funktion **Vorschauen erneuern** erfolgreich zu beenden, müssen Sie die Vorschaubilder leider manuell mittels anderer Bildverwaltungsprogramme verkleinern.

4.6 Aussehen

Der Menübereich **Aussehen** fasst die Möglichkeiten zusammen, um Ihr Blog zu individualisieren. Zum einen können Sie hier das Template Ihres Blogs wählen, zum anderen Ihre eingesetzten Plugins verwalten.

4.6.1 Styles verwalten

Bei Serendipity sind die Begriffe *Style*, *Theme* und *Template* synonym verwendet. Alle drei Begriffe bezeichnen eine Sammlung von Dateien, die das Layout Ihres Frontends (und auch in begrenztem Maße des Backends) bestimmen.

Alle Templates werden im Verzeichnis `templates` verwaltet. Dort befindet sich pro Template ein Unterverzeichnis, das eine Sammlung von Dateien enthält (siehe Kapitel 9.4.2 ab Seite 489).

Um eigene oder fremde Templates in Serendipity einzubinden, müssen sich die Template-Dateien in diesem Unterverzeichnis befinden. Erst dann kann Serendipity über den Menüpunkt **Styles verwalten** auf diese Verzeichnisse zugreifen.

Eine Template-Datei stellt dabei bestimmte Elemente (Einträge, Übersichten, Kommentare ...) in einfachem HTML-Format dar. Innerhalb der Template-Datei (`*.tpl`) werden Platzhalter verwendet, die bei der Ausführung Serendipitys durch eigentliche Inhalte ersetzt werden. Für diese Platzhalter-Logik verwendet Serendipity eine Software-Bibliothek namens *Smarty*[13]. Smartys Aufgabe ist es, die Template-Dateien nach Platzhaltern zu durchsuchen, zu *kompilieren* (in PHP-Code umwandeln) und danach auszugeben. Dabei kann Smarty weitaus mehr, als nur Variablen zu ersetzen: Smarty bietet Kontrollstrukturen (*IF-Abfragen*), Schleifen und PHP-Funktionsaufrufe an. Dies alles kann von Template-Entwicklern benutzt werden, um ganz individuelle Darstellungen des Frontends zu erreichen, und wird in Kapitel 9.3 ab Seite 480 ausführlich erklärt.

Ein Klick auf den Menüpunkt **Styles verwalten** stellt eine Liste aller verfügbaren Templates dar. Dabei werden auch kleine Vorschaubilder zu den Themes untereinander angezeigt, die einen Eindruck vermitteln, wie Ihr Frontend später aussehen könnte.

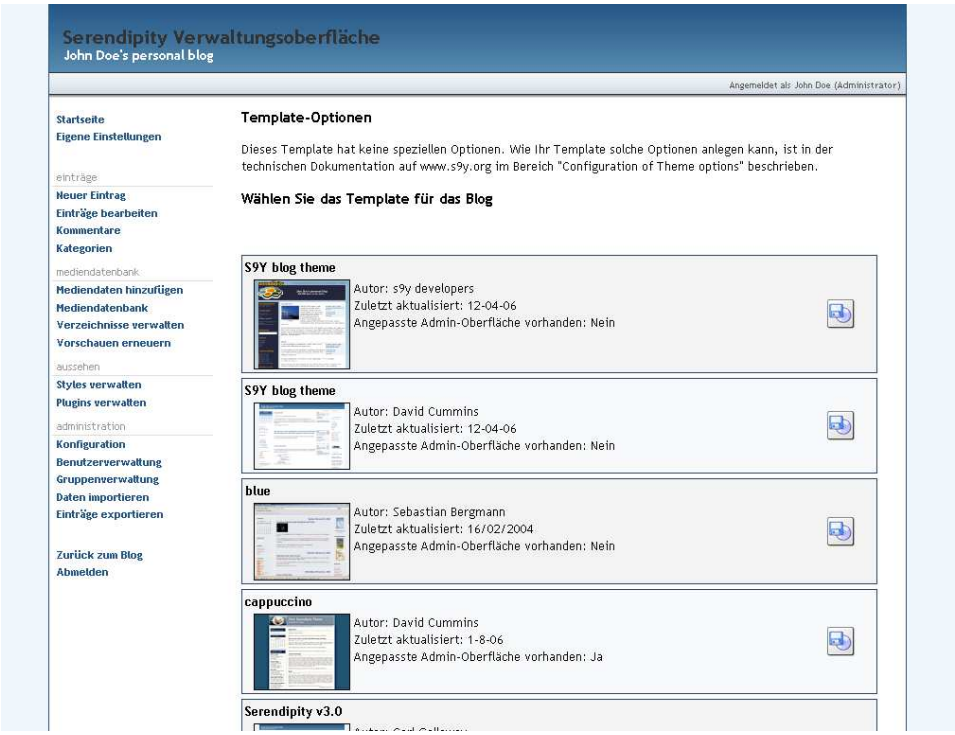

Abbildung 4.19: Aussehen: Styles verwalten

Jede Template-Box enthält dabei als Überschrift den Namen des Templates und Informationen über den Autor, das Erstellungsdatum und ob eine **Angepasste Admin-Oberfläche vorhanden** ist. Zu einigen Templates wird Ihnen auch ein großes Vorschaubild angeboten,

[13] `http://smarty.php.net/`

wenn Sie auf die kleine Vorschaugrafik klicken.

Um ein Template zu aktivieren, klicken Sie einfach auf den Button mit dem **Diskettensymbol** rechts daneben. Daraufhin wird das Template aktiviert, und wenn Sie das Frontend nun öffnen (**Zurück zum Blog**), werden Sie das neue Layout betrachten können. Je nach eingesetztem Browser könnte es notwendig sein, dass Sie den Browser neu starten oder den Browsercache leeren müssen, bevor die Änderungen am Layout bei Ihnen sichtbar sind. Dies könnte erforderlich sein, da das Design zum Teil von der geladenen Stylesheet-Datei bestimmt wird, die aus Geschwindigkeitsgründen von vielen Browsern zwischengespeichert wird.

4.6.2 Template-Optionen

Seit Serendipity 1.1 besteht für Templates die Möglichkeit, eigene Optionen zu konfigurieren. Das bedeutet, dass ein Template individuell entscheiden kann, was für zusätzliche Möglichkeiten und Darstellungsarten Sie gerne aktivieren möchten.

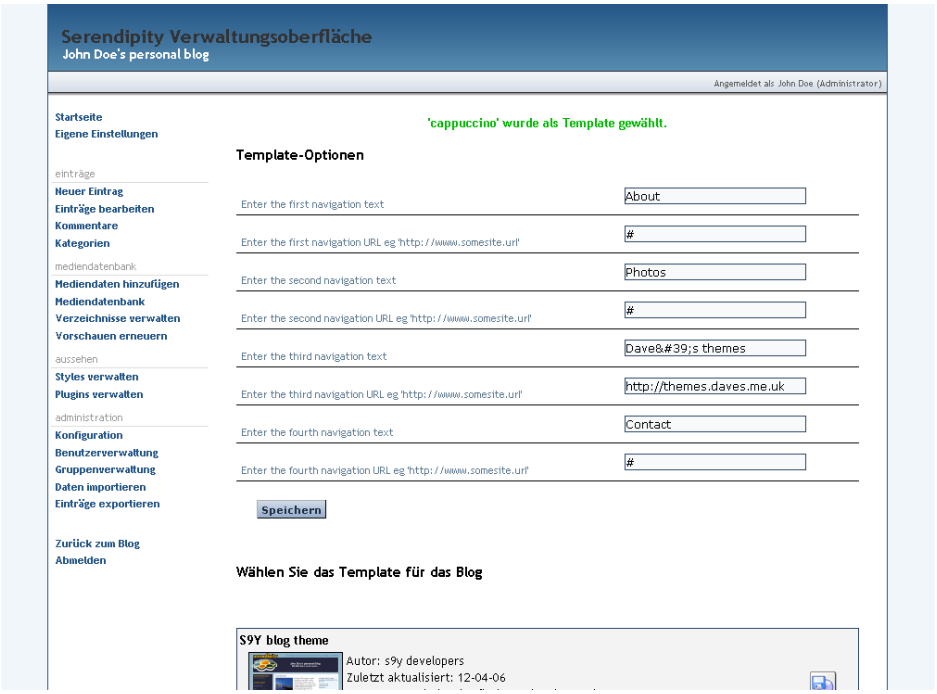

Abbildung 4.20: Aussehen: Template-Optionen des Themes Cappucino

Wenn Sie ein Template ausgewählt haben, das diese Optionen bereitstellt, sehen Sie einen zusätzlichen Bereich im ersten Abschnitt des **Styles verwalten**-Menüpunktes. Nur wenige der von Serendipity mitgelieferten Templates unterstützen diese Optionen. Das Template `cappucino` von David Cummins bietet z. B. diese Template-Optionen an, um den Inhalt

der zusätzlichen Navigations-Links im Frontend einstellen zu können.

Ist dieses Template gewählt, so befinden sich im oberen Bereich jeweils zwei Eingabeboxen pro Menüpunkt. Über diese Eingabeboxen können Sie den jeweiligen Link eines Menüpunkts im Frontend bestimmen und festlegen, wie der Text des Menüpunktes lauten soll.

Andere Templates, die via Spartacus (siehe Seite 265) bezogen werden können, bieten Template-Optionen für weitaus fortgeschrittenere Konfigurationsmöglichkeiten, die Sie z. B. zwischen mehreren Farbvarianten wählen lassen oder auch bestimmen, wie die Inhaltsbreite festgelegt wird.

4.6.3 Besondere Templates

Bei Serendipity gibt es eine kleine Zahl besonderer Templates, die Sie nur in Sonderfällen aktivieren sollten. Dies sind:

`PHP Engine (Dev)`
: Das Template `PHP Engine (Dev)` benutzt eine vom Standard abweichende Template-Engine. Anstelle von Smarty kommt hier PHP-Syntax zum Einsatz. Dieses Template ist nur für Entwickler gedacht und führt zu Fehldarstellung des Frontends, wenn es ohne weitere Modifikationen eingesetzt wird.

`XML Engine (Dev)`
: Auch das Template `XML Engine (Dev)` benutzt eine vom Standard abweichende Template-Engine. Hier kommt XSLT zum Einsatz, und auch dieses Template kann ohne weitere Anpassung die Darstellung des Frontends unbenutzbar machen. Für beide genannten Templates gilt, dass der Namenszusatz *Dev* für *Developer* (Entwickler) steht und daher auch nur von dieser Zielgruppe benutzt werden sollte.

`Newspaper Blog`
: Dieses Template wird eher aus historischen Gründen mitgeliefert und stammt aus einer Zeit, in der Serendipity-Templates noch nicht mittels Smarty-Syntax erstellt wurden. Dementsprechend erhalten Sie die Warnung:

  Hinweis: Das aktuelle Template verwendet eine ältere Methode der HTML-Erzeugung. Falls möglich, bitte das Template auf die Nutzung von Smarty optimieren.

  wenn das Template aktiviert wurde. Im Grunde genommen soll dieses Template also nur demonstrieren, dass auch ganz alte Templates mit Serendipity noch grundsätzlich funktionstüchtig sind.

4.6.4 Plugins verwalten

Das Herzstück des Serendipity-Systems zur Erweiterung des Blogs ist der Menüpunkt **Plugins verwalten**.

Als *Plugin* bezeichnet man eine eigenständige PHP-Datei, die zusätzliche Funktionalität in das System „stöpselt“. Das Aktivieren eines Plugins erfolgt komfortabel über eine Verwaltungsoberfläche, Sie müssen also selber keine Dateien bearbeiten oder Kenntnisse des Systems haben.

Ein Plugin hat zudem den Vorteil, dass bestimmte Funktionen so einfach optional nachgerüstet werden können. Für viele Benutzer ist möglicherweise ein Umfrage-Modul interessant, während andere Benutzer damit gar nichts anfangen können. Da liegt es nahe, diese optionale Funktionalität so loszulösen, dass nur diejenigen die Komponente installieren, die sie auch benötigen. Das hält das Grundsystem schlank und macht es einfacher wartbar. Durch die Auslagerung von Code in Plugins können auch kleine Bestandteile des Blogs einzeln aktualisiert werden. Wenn eine neue Version einer Komponente veröffentlicht wird, müssen Sie also nicht direkt das ganze Blogsystem aktualisieren.

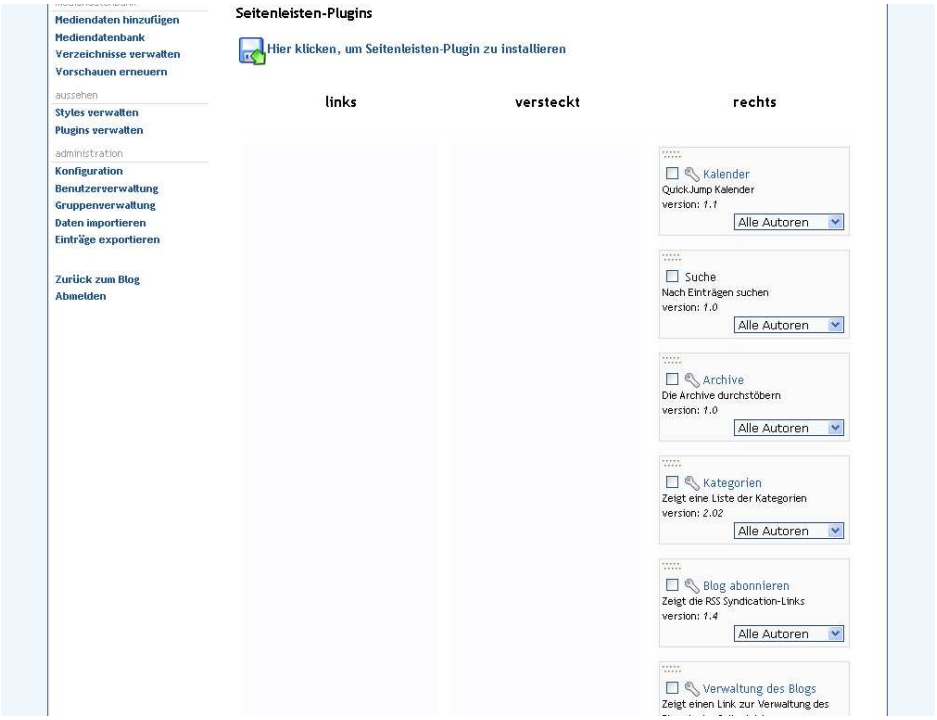

Abbildung 4.21: Aussehen: Plugins verwalten

Bei Serendipity gibt es zwei verschiedene Arten von Plugins. *Seitenleisten-Plugins* stellen Inhalte und Funktionen in der Seitenleiste eines Blogs im Frontend dar. *Ereignis-Plugins* sind interne Funktionen von Serendipity, die Anpassungen des Kernsystems ermöglichen und sich nicht nur im Frontend, sondern auch im Backend auswirken können. Ein Seitenleisten-Plugin sieht man daher immer an einer festgesetzten Stelle im Frontend, während ein Ereignis-Plugin eher *im Hintergrund* arbeitet.

Nach einem Klick auf **Plugins verwalten** sehen Sie zuerst eine Liste aller Seitenleisten-Plugins. Pro Plugin gibt es eine einzelne Inhaltsbox. Die Seitenleisten-Plugins sind in verschiedene Bereiche unterteilt: *links*, *rechts* und *versteckt*. Diese Bereiche entsprechen also der Positionierung des Seitenleisten-Plugins im Frontend, auf der linken oder rechten Seite. Dabei gilt es zu beachten, dass einige Templates möglicherweise nur eine Seitenleiste anbieten, und manchmal sind beide Seitenleisten auch rechts oder links angegliedert. Abhängig vom Template kann es auch sein, dass hier noch weitere Plugin-Bereiche dargestellt werden.

Der Bereich *versteckt* ist dabei ein Sonderbereich. Alle Plugins, die dort aufgeführt sind, werden später im Frontend nicht dargestellt.

Innerhalb der Pluginbox sind folgende Informationen/Einstellungen enthalten:

Drag and Drop-Rahmen

Jede einzelne Pluginbox kann mittels *Drag'n'Drop* in ihrer Position verändert werden. Das heißt, sowohl die Reihenfolge der Plugins untereinander als auch die Wahl der linken oder rechten Seitenleiste kann bestimmt werden. Dazu klicken Sie mit der Maus auf den Anfassbereich in der linken oberen Ecke jeder Pluginbox (der Cursor verwandelt sich dann in einen Positionierungspfeil), halten die Maus gedrückt und verschieben dann die Box an die gewünschte Position. Dabei wird der jeweilige Bereich, über dem Sie sich befinden, mit einem Rahmen hervorgehoben. Sobad Sie die Maustaste loslassen, wird das Plugin in den gerade hervorgehobenen Bereich eingefügt.

Diese Funktionalität wird mittels JavaScript eingebunden. Sollte Ihr Browser dies nicht unterstützen, können Sie über das Menü **Eigene Einstellungen** die Option deaktivieren, dass **Fortgeschrittene Javascripts** eingesetzt werden sollen. Sobald dies geschehen ist, sehen Sie anstelle der Anfasser für jedes Plugin eine Auswahlbox, mit der Sie angeben, ob das Plugins links, rechts oder versteckt platziert wird. Die Reihenfolge kann in diesem Falle dann über zwei eigenständige Pfeil-Buttons verändert werden.

Es ist wichtig, dass Sie nach der Änderung der Positionierung auf den Button **Speichern** klicken. Erst wenn Sie diesen Button anklicken, werden die Änderungen tatsächlich gespeichert. Sollten Sie also einmal nach einer längeren Sortierungsaktion unzufrieden mit dem Ergebnis sein, können Sie die Seite einfach neu laden oder verlassen, und die Änderungen werden verworfen.

Markierungsbox
: Neben jedem Plugin-Titel ist eine Auswahlbox dargestellt, die angekreuzt werden kann. Um ein oder mehrere Plugins zu löschen, müssen Sie die Boxen der jeweiligen Plugins aktivieren und danach auf den Button **Markierte Plugins entfernen** klicken.

Schraubstock-Symbol
: Einige Plugins bieten spezielle Konfigurationsoptionen an. Jedes Plugin mit solchen Optionen zeigt ein **Schraubstock**-Symbol an, auf das Sie klicken können.

Plugin-Titel
: Der Name eines Plugins wird im Zentrum der Pluginbox angezeigt. Plugins, die eigene Konfigurationseinstellungen anbieten, kann man mit einem Klick auf deren Namen (oder das Schraubstock-Symbol) bearbeiten.

Plugin-Beschreibung
: In der Plugin-Beschreibung wird die Funktionalität eines Plugins kurz beschrieben.

Plugin-Version
: Da Plugins öfter aktualisiert werden und mit neuen Versionen Fehler behoben oder neue Funktionen angeboten werden, ist es ab und zu interessant zu erfahren, welche Version eines Plugins man gerade einsetzt. Diese Versionsnummer wird unterhalb der Plugin-Beschreibung angezeigt.

Plugin-Eigentümer
: Zuletzt gibt ein Auswahlfeld den Eigentümer eines Plugins an. Bei einigen Plugins ist nur der hier festgelegte Eigentümer (oder ein Administrator) autorisiert, Änderungen an der Konfiguration eines Plugins durchzuführen. Ob ein Plugin solche Einschränkungen festlegt, ist dem Plugin selbst überlassen. Das Plugin `HTML Klotz` ist ein Beispiel hierfür.

  Diese Einstellung regelt *nicht*, wer den Inhalt eines Plugins im Frontend sehen darf! Wenn solche Beschränkungen vorgenommen werden sollen, muss das Plugin *Seitenleisten ein/ausklappen* (siehe Seite 298) verwendet werden.

Um bestimmte Plugins zu entfernen, markieren Sie diese und klicken auf **Markierte Plugins entfernen**. Um Änderungen an dem Eigentümer eines Plugins zu speichern, müssen Sie das Plugin markieren und danach auf **Speichern** klicken. Für Änderungen an der Positionierung oder Reihenfolge eines Plugins klicken Sie einfach nur auf **Speichern**, eine Auswahl der beteiligten Plugins ist in diesem Fall nicht notwendig.

Ein neues Seitenleisten-Plugin können Sie installieren, indem Sie den Link **Hier klicken, um Seitenleisten-Plugin zu installieren** benutzen.

Analog zu der Auflistung der Seitenleisten-Plugins werden die Ereignis-Plugins dargestellt. Da diese im Frontend keinem *Bereich* entsprechen, sind sie nicht weiter unterteilt und gliedern sich nur in **Aktiv** und **Inaktiv**. Ein aktives Ereignis-Plugin wird bei jedem Seitenaufruf von Serendipity in den Speicher geladen und ausgeführt. Wenn Sie ein Plugin vorübergehend deaktivieren wollen, können Sie es einfach in den Bereich **Inaktiv** verschieben.

Die Reihenfolge von Ereignis-Plugins kann in einigen Situationen wichtig sein. Die Liste dieser Plugins wird vom ersten bis zum letzten in dieser Reihenfolge ausgeführt; dabei kann die Ausgabe des vorigen Plugins miteinbezogen werden. Gerade bei Textformatierungs-Plugins, die gewisse Umformatierungen Ihrer Einträge vornehmen, ist diese Reihenfolge von Interesse. Wenn bspw. ein Textformatierungs-Plugin wie *Textformatierung: Smilies* ausgeführt wird und danach ein Plugin wie das *Textformatierung: Wiki-Markup*, dann könnte dies dazu führen, dass die HTML-Formatierung der Smilies durch das Wiki-Markup-Plugin zerstört wird. Richtig wäre es daher, das Wiki-Markup-Plugin als eines der ersten Textformatierungs-Plugins anzuwenden. Bei welchen Fällen Sie auf die Reihenfolge zu achten haben, wird bei der Beschreibung des jeweiligen Plugins in diesem Buch erwähnt.

Ereignis-Plugins können ähnlich wie Seitenleisten-Plugins über **Hier klicken, um Ereignis-Plugin zu installieren** neu hinzugefügt werden.

4.6.5 Fehler bei Plugins

Je länger die Liste der Ereignis-Plugins, desto mehr Arbeitsspeicher benötigt Serendipity auf dem Webserver. So kann es passieren, dass nach der Installation eines neuen Ereignis-Plugins der Arbeitsspeicher ausgereizt ist.

Es kann durchaus passieren, dass zu viele Plugins die fehlerfreie Ausführung des Frontends oder Backends verhindern. In so einem Fall müssen Sie also Ereignis-Plugins löschen oder Ihren Provider bitten, den Arbeitsspeicher (siehe Kapitel 1.3.2 auf Seite 33) zu vergrößern.

Sollten Sie nicht mehr auf die Plugin-Verwaltungsoberfläche gelangen, können Sie ein Plugin auch entfernen, indem Sie das entsprechende Verzeichnis im `plugins`-Unterverzeichnis löschen. Sobald Serendipity eine angeforderte Plugin-Datei nicht mehr findet, wird das entsprechende Plugin einfach temporär deaktiviert. Sie werden es dann trotzdem noch in der dann wieder aufrufbaren Plugin-Verwaltung sehen können, dort wird dann jedoch nur eine Meldung wie

```
Fehler! serendipity_event_xxx
```

zu sehen sein. Ein solches Plugin können Sie jedoch wie gewohnt löschen.

Abgesehen von der absichtlichen Löschung eines Plugins kann es auch einmal vorkommen, dass Sie die obige Meldung bei Plugins sehen, die eigentlich fehlerfrei funktionieren sollten. Diese Meldung kann auch angezeigt werden, wenn eine Plugin-Datei für Serendipity nicht lesbar ist oder womöglich fehlerhaft via FTP hochgeladen wurde. Prüfen Sie daher, ob diese Dateien auch dem richtigen Zustand (Dateigröße und Dateirechte) entsprechen.

Auch nach einem Serverumzug kann es vorkommen, dass früher funktionierende Plugins plötzlich nicht mehr aufrufbar sind. In diesem Fall finden Sie in der Datenbanktabelle `serendipity_plugins` eine Auflistung aller aktivierten Plugins inklusive deren Dateipfad. Diesen Pfad müssen Sie möglicherweise auf neue Gegebenheiten anpassen.

Manchmal kann es sein, dass nach der Installation mehrere Instanzen eines Plugins aktiviert wurden. Sie können daher doppelte Plugins leicht wieder aus der Verwaltungsoberfläche löschen.

Wenn Sie eine Plugin-Datei selbständig editieren, kann es passieren, dass Sicherungskopien eines Plugins mit der Dateiendung `.bak` oder `.php` von Ihrem Editor im jeweiligen Plugin-Verzeichnis gespeichert werden. Diese Dateien können die Plugin-Verwaltung von Serendipity stören und müssen daher unbedingt gelöscht werden. Auch ganze Kopien eines Verzeichnisses unter einem zweiten Namen dürfen niemals im `plugins`-Unterverzeichnis gespeichert werden, verschieben Sie derartige Sicherheitskopien immer an einen von Serendipity unabhängigen Speicherort!

4.6.6 Neue Plugins installieren

Wenn Sie ein neues Ereignis- oder Seitenleisten-Plugin installieren, wird Serendipity Ihnen die verfügbaren Plugins auf einer Plugin-Übersicht darstellen. Dort wird jedes Plugin, das im Serendipity-Verzeichnis vorhanden ist, aufgelistet. Auch bereits installierte Plugins werden daher an dieser Stelle dargestellt, da es einige Arten von Plugins gibt, die mehrfach installiert (genauer: instanziert) werden können.

Serendipity wird mit einer kleinen Auswahl an verfübaren Plugins ausgeliefert, die nicht alle von Anfang an aktiviert sind. Sollten Sie weitere Plugins benötigen, können Sie diese entweder vom Server `http://spartacus.s9y.org/` manuell herunterladen, automatisch mittels des `Spartacus`-Plugins installieren oder ein selbst entwickeltes Plugin auswählen. Wenn Sie aus dem Internet ein Serendipity-Plugin als ZIP-Datei heruntergeladen haben, müssen Sie dieses in die Unterverzeichnisstruktur von `plugins` entpacken. Serendipity kann mit der ZIP-Datei selber nichts anfangen, und auch wenn eine Plugin-Datei versehentlich nur ins Stammverzeichnis kopiert werden sollte, kann dies dazu führen, dass es später in der Plugin-Übersicht nicht aufgeführt wird.

Plugins müssen also in zwei Schritten für Serendipity verfügbar gemacht werden. Zum einen muss ein Plugin in einem Unterverzeichnis des `plugins`-Verzeichnisses gespeichert werden, damit Serendipity das Plugin überhaupt sehen kann. Im zweiten Schritt muss dann das hochgeladene Plugin für Serendipity über die Plugin-Verwaltung *aktiviert* werden.

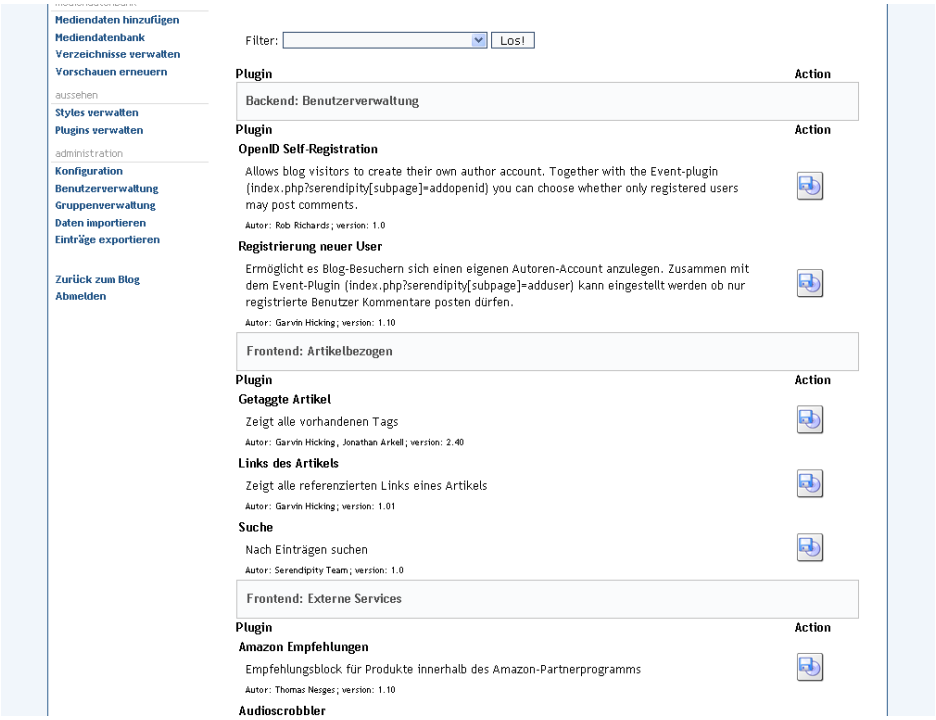

Abbildung 4.22: Aussehen: Plugins verwalten: Neues Plugin installieren

In der Plugin-Übersicht für die Installation sehen Sie pro Zeile jeweils einen Plugin-Namen. Die Plugins sind unterteilt in verschiedene Gruppen, die Sie speziell auch über das Auswahlfeld **Filter** am Seitenanfang anwählen können.

Unter jedem Plugin-Namen wird eine Beschreibung des Plugins angezeigt, sowie der Autor und die Versionsnummer des Plugins. Rechts neben der Beschreibung sehen Sie entweder einen Button zur Installation des gewünschten Plugins oder die Information, dass ein Plugin **bereits installiert** ist.

Wenn Sie auf den Installations-Button klicken, werden Sie ggf. direkt zur Konfiguration des gerade gewählten Plugins weitergeleitet. Dort werden Ihnen die Standardoptionen bereits vorausgefüllt, Sie müssen also nicht sofort die vollständige Konfiguration aufrufen, sondern können die Konfiguration auch jederzeit später vervollständigen. Nach diesem Vorgang ist ein Plugin installiert und kann wie gewohnt über den Menüpunkt **Plugins verwalten** verschoben und konfiguriert werden.

4.6.7 SPARTACUS

SPARTACUS steht für das *Serendipity Plugin Access Repository Tool And Customization/Unification System*. Trotz des zugegebenermaßen konstruierten Akronyms wird deutlich, dass Spartacus ein System ist, das Plugins und Templates zur Verfügung stellt.

Spartacus besteht aus zwei Komponenten: Zum einen der zentrale Server `http://spartacus.s9y.org`, auf dem sämtliche verfügbaren offiziellen Plugins vorhanden sind und auch für Nicht-Serendipity-Benutzer übersichtlich dargestellt werden. Zum anderen wird bei Serendipity ein gleichnamiges Ereignis-Plugin mitgeliefert, das direkt auf diesen zentralen Server zugreifen und Plugins herunterladen kann. Das Plugin stellt einige Anforderungen an den Server, auf dem Sie Serendipity installiert haben:

- Der Webserver darf nicht durch eine Firewall blockiert werden. Er muss Zugriffe auf `netmirror.org`, `sourceforge.net` und `spartacus.s9y.org` auf Port 80 ermöglichen.
- Die PHP-Konfiguration muss den Zugriff auf Netzwerk-Sockets erlauben.
- Damit Serendipity Dateien selbständig herunterladen und auf dem Server speichern kann, muss Schreibzugriff auf die Verzeichnisse `plugins` und `templates` gewährt werden.

Auf sehr vielen Webservern schlägt die erste Bedingung fehl, was dazu führt, dass keine PHP-Anwendung *nach draußen* zugreifen kann. Gerade bei großen Providern ist aus Sicherheitsgründen ein derartiger Zugriff nicht gestattet. In so einem Fall können Sie Spartacus also leider nicht einsetzen und müssen die Plugins manuell vom Server `http://spartacus.s9y.org/` beziehen. In der neuesten Serendipity-Version besteht behelfsweise die Möglichkeit, Dateien direkt über Ihren FTP-Zugang auf den Webserver hochzuladen – dadurch werden die PHP-Beschränkungen umgangen. Die Zugangsdaten müssen Sie in der Konfiguration des Spartacus-Plugins festlegen.

Wenn das Spartacus-Plugin auf solchen blockierten Servern installiert wird, kann das zu einer Reihe von Problemen führen: von der vollständigen Fehlfunktion der Plugin-Installationsoberfläche bis hin zu weißen Seiten beim Klick auf ein zu installierendes Plugin.

Sollten Sie also beim Installieren von Plugins eine leere Seite erhalten, oder Ihr Browser gibt an, die Datei `serendipity_admin.php` müsse lokal auf der Festplatte gespeichert werden, oder die Installation eines Plugins führt zu langen Ladezeiten, dann können Sie sich sicher sein, dass Spartacus auf Ihrem Server nicht gestartet werden kann. Eine Fehlermeldung, die in diesem Zusammenhang auch auftreten kann, ist:

```
Access Denied in /var/www/example.com/serendipity/bundled-libs/PEAR.php
at line 848
```

Dies ist eine generische Fehlermeldung, die bedeutet, dass die für den Download zuständige PEAR-Komponente keine Verbindung zu einem Server aufnehmen konnte.

Möglicherweise kann Ihr Webserver auf fremde Server zugreifen, wenn Sie einen Proxy-Server benutzen. Um einen Proxy für Spartacus einzustellen, bietet das Plugin `Trackbacks kontrollieren` (siehe Seite 283) eine Möglichkeit, den Proxy-Server zu konfigurieren. Der dort eingestellte Proxy-Server gilt dann für alle Serendipity-Funktionalitäten und aktuellen Plugins.

Wenn Sie ein älteres Plugin nutzen, das sich nicht von der im Trackback-Plugin konfigurierten Proxy-Einstellung beeindrucken lässt, können Sie notfalls auch manuell in der Datei `bundled-libs/HTTP/Request.php` in Zeile 150 folgende Variablen setzen:

```
$this->_proxy_host = null;
$this->_proxy_port = null;
$this->_proxy_user = null;
$this->_proxy_pass = null;
```

Beachten Sie, dass diese Änderung einer zentralen Bibliothek nicht gerade die *feine englische Art* ist, da die lokale Bibliothek auf einigen Servern durch eine zentrale PEAR-Bibliothek übergangen werden könnte.

Aufgrund der aufgeführten potenziellen Probleme ist das Spartacus-Plugin standardmäßig nach der Installation nicht aktiviert. Sie müssen es daher über den Menüpunkt **Klicken Sie hier, um Ereignis-Plugin zu installieren** erst auswählen und aktivieren. In der Konfiguration des Spartacus-Plugins können einige wichtige Optionen festgelegt werden, siehe Seite 265.

Kann Spartacus auf dem Server eingesetzt werden, fügt es sich nahtlos in die Serendipity-Oberfläche ein. Wenn Sie sich mit aktiviertem Plugin dann die Plugin-Verwaltung zur Installation eines neuen Plugins ansehen, wird das Plugin automatisch aus dem Internet eine aktuelle Liste der verfügbaren Plugins laden und Ihnen darstellen. Bei Klick auf ein Plugin wird Spartacus automatisch alle benötigten Dateien herunterladen und Sie über diesen Download benachrichtigen.

Plugins, die mittels Spartacus installiert werden, stellen als Aktions-Button ein CD-Symbol mit Download-Pfeil dar. Außerdem wird bei diesen Plugins zusätzlich in der Informationszeile angegeben, dass sie mittels Spartacus heruntergeladen werden.

Sollte der Download eines Plugins mittels Spartacus einmal fehlschlagen, können Sie über FTP das entsprechende Plugin-Verzeichnis einfach komplett löschen und die Neuinstallation probieren.

Auch die Aktualisierung von Plugins kann mittels Spartacus komfortabel vorgenommen werden. Anstatt über die Spartacus-Webseite nach neuen Versionen zu suchen, können Sie über die Plugin-Verwaltung die Menüpunkte **Neue Versionen von Seitenleisten-Plugins** und **Neue Versionen von Ereignis-Plugins** anklicken. Dort werden Sie dann alle aktualisierbaren Plugins aufgelistet bekommen und können auf den **roten Aktualisierungsbutton** klicken, um ein Upgrade durchzuführen.

Leider gibt es bei einem Upgrade in der Plugin-Oberfläche keinen Hinweis, welche Dinge sich bei einem Plugin verändert haben. Als Faustregel gilt, dass nur große Versionssprünge in einem Plugin tiefgreifende Änderungen mit sich bringen. Die meisten Updates sind Bugfixes

oder kleinere neue Features. Da Plugin-Entwickler oft faule Leute sind, ist die Dokumentation der Updates jedoch meist dürftig, und Sie müssen sich die Aktualisierungen manuell in der Spartacus-Versionsverwaltung ansehen.[14] Details zu dieser Webseite finden Sie im Kapitel 10.7 ab Seite 637.

Auch bei Plugins werden Updates immer unter großer Beachtung der Rückwärts-Kompatibilität durchgeführt. Sie müssen also nicht fürchten, dass sich bei einem Update eines Plugins etwas für Sie ändern könnte oder zu Bruch geht – diese Gefahr ist relativ gering. Lediglich wenn Sie selbst Änderungen an den Plugin-Dateien gemacht haben, müssen Sie diese sichern und nach dem Update erneut einfügen. Die Konfigurationsoptionen Ihrer Plugins bleiben bei einer Aktualisierung weiterhin bestehen.

Wenn Sie mehrere Plugins auf einmal aktualisieren wollen, müssen Sie diese nacheinander anklicken. Bei einem Browser, der *Tabs* (Registerkarten) unterstützt, könnten Sie einfach jeden Link mit einem Klick der mittleren Maustaste auf das Plugin-Symbol in einem solchen separaten Fenster öffnen. Das spart Ihnen etwas Zeit, um nicht jedes Mal zurück zur Verwaltungsoberfläche gehen zu müssen.

Das Upgrade eines Plugins können Sie auch ausführen, wenn Sie wie gewohnt die Übersicht aktivierbarer Plugins durchgehen. Auch hier sollten Ihnen die roten Aktualisierungs-Buttons ins Auge stechen.

Oberhalb jeder Übersichtsseite werden Sie einige Meldungen von Spartacus sehen, die Sie darüber informieren, dass gerade eine Übersichtsdatei oder Weiteres aus dem Internet geladen wird. Diese Meldungen helfen Ihnen, im Fall von Zugriffsproblemen die Schreibrechte zu prüfen.

Die XML-Dateien, die Spartacus herunterlädt, werden in dem temporären Verzeichnis `templates_c` gespeichert. Damit das Plugin nicht bei jedem Zugriff auf Ihre Plugin-Verwaltung diese Dateien neu lädt, wird Spartacus die Dateien zwischenspeichern und nur alle 24 Stunden auf Aktualisierung prüfen. Aufgrund dieser Zwischenspeicherung kann es also passieren, dass neue Plugins im Spartacus-Archiv erst mit maximal 24 Stunden Verzögerung dargestellt werden. Wenn Sie unbedingt forcieren wollen, dass Spartacus den Zwischenspeicher leert, dann können Sie über die Plugin-Konfiguration von Spartacus einfach auf den Button **Speichern** klicken. Dadurch werden die temporären Dateien alle gelöscht. Auf Wunsch könnten Sie die Dateien natürlich auch mit einem FTP-Programm löschen.

Alle Inhalte des Spartacus-Systems können Sie auch ohne Serendipity unter `http://spartacus.s9y.org/` ansehen und dort Plugins als ZIP-Datei manuell herunterladen.

Die Plugins von Spartacus werden, ebenso wie Serendipity selbst, öffentlich über ein Versionskontrollsystem gepflegt. Alle vorgenommenen Änderungen werden darin protokolliert. Details finden Sie in Kapitel 10.7 auf Seite 637.

[14] `http://php-blog.cvs.sourceforge.net/php-blog/additional_plugins/`

4.7 Administration

Im Menübereich **Administration** befinden sich alle Möglichkeiten, die Optionen und Benutzer Ihres Blogs zu bearbeiten. Auch der Export und der Import von Blog-Artikeln wird über diesen Bereich ausgeführt.

4.7.1 Konfiguration

Hinter **Konfiguration** finden Sie alle Einstellungen des Blogs, die Sie auch bei der Installation im *Fortgeschrittenen*-Modus festlegen können.

Der Konfigurationsbereich ist dabei in mehrere Bereiche untergliedert, um etwas Struktur in die Oberfläche zu bringen. Jeder Bereich kann dabei über den +-Button aus- und eingeklappt werden. Um alle Bereiche auf einmal auszuklappen, klicken Sie auf den Link **Alle Optionen ein-/ausblenden**.

Datenbankeinstellungen

Die **Datenbankeinstellungen** bestimmen den Zugang zur zentralen Serendipity-Datenbank und deren möglichen Optionen.

Hier müssen immer die aktuell gültigen Zugangsparameter zur Datenbank eingetragen werden. Diese Datenbankinformationen werden zusätzlich in der Datei `serendipity_config_local.inc.php` gespeichert. Wenn sich also einmal die Zugangsdaten ändern, könnte es sein, dass Sie manuell zuerst diese Datei (siehe auch Kapitel 2.2.6 auf Seite 64) überarbeiten müssen, bevor Sie sich wieder ins Backend einloggen können.

Abbildung 4.23: Administration: Konfiguration

Datenbanktyp

In diesem Auswahlfeld sind die verfügbaren Datenbanktypen aufgeführt, zu denen Serendipity Verbindung aufnehmen kann. Die Liste der verfügbaren Typen richtet sich nach der Konfiguration Ihres PHP-Servers. PHP muss für jeden Datenbankserver eine sogenannte *Client-Library* eingebunden haben, um auf die Funktionen des jeweiligen Datenbanktyps zugreifen zu können. Wenn Sie sich sicher sind, dass auf Ihrem Server ein MySQL-Server läuft, muss das nicht automatisch bedeuten, dass PHP darauf auch zugreifen kann. Zahlreiche Linux-Distributionen bieten die Client-Libraries in eigenen Paketen an, bei Debian beispielsweise `php-mysql`.

Serendipity kann auf folgende Client-Libraries zugreifen: `mysql` (MySQL 3, eingeschränkt auch 4 und 5), `mysqli` (ab MySQL 4), `postgres` (PostgreSQL), `PDO`

`postgres` (spezielle PHP5-Variante der PostgreSQL-Library), `sqlite` und `sqlite3`.

SQLite ist ein Sonderfall einer Datenbank, da es ohne einen Datenbankserver auskommen kann. Ab PHP5 ist SQLite fester Bestandteil von PHP und somit immer verfügbar. Die Vorteile von SQLite sind eine sehr hohe Performance beim Lesen von Datenbanken und die Erstellung einer einzelnen Datenbankdatei, von der man sehr leicht ein Backup aufnehmen kann. Der Nachteil ist, dass SQLite nicht alle Datenbank-Operationen mittels SQL-Syntax unterstützt und gerade bei komplexeren Datenbankabfragen doch eher langsam wird.

Am besten getestet ist Serendipity bei Verwendung der MySQL-Datenbank. Die Kernfunktionen Serendipitys laufen auch problemlos mit allen anderen Datenbanktypen, aber einige Plugins könnten spezielle MySQL-Funktionen einsetzen.

Wenn Sie den Datenbanktyp bei einer aktiven Serendipity-Installation ändern, richtet Serendipity *keine* neue Datenbank auf dem gewünschten System ein! Alle Datenbankeinstellungen richten sich nach dem *Ist-Zustand*, Serendipity geht also davon aus, dass Sie mit diesen Optionen die aktuellen Gegebenheiten abbilden, und nicht, wie Sie es in Zukunft gerne hätten. Um Serendipity auf eine andere Datenbank umzustellen, müssen Sie die Migration mithilfe der Dokumentation des Datenbankservers leider selbständig vornehmen.

Datenbank Servername
Der Servername der Datenbank kann entweder eine IP-Adresse (z. B. `127.0.0.1`), ein lokaler Hostname (`localhost`) oder ein Internet-Hostname (`www.example.com`) sein. Wird für SQLite nicht benötigt.

Datenbank Username
Der Benutzername für den Datenbankserver. Wird für SQLite nicht benötigt.

Datenbank Passwort
Das Passwort für den Benutzer des Datenbankservers. Wird für SQLite nicht benötigt.

Datenbankname
Der Name der Datenbank, in der Serendipity installiert wurde. Bei Verwendung von SQLite wird hier der Dateiname der `.db`-Datenbank gespeichert.

Datenbank-Präfix
Jede Serendipity-Tabelle in einer Datenbank muss einen vordefinierten Präfix besitzen, damit sich die Tabellennamen von möglicherweise gleichnamigen Tabellen anderer Anwendungen in derselben Datenbank abheben können. Standardmäßig wird hier `serendipity_` eingetragen. Wenn Sie den Tabellenpräfix nachträglich ändern, müssen Sie die bestehenden Tabellen manuell in der Datenbank umbenennen, Serendipity führt dies nicht selbständig aus.

Persistente Verbindungen nutzen
Bei jedem Aufruf einer Seite von Serendipity wird eine Verbindung zur Datenbank

aufgebaut. Im „Lebenszyklus“ einer PHP-Anwendung wird diese Datenbankverbindung nach der Ausführung automatisch wieder geschlossen. Um etwas Zeit zu sparen, können PHP-Anwendungen sogenannte *Persistent Connections* (*Ständige Verbindungen*) nutzen. Diese Option wird in der PHP-Dokumentation[15] ausführlicher beschrieben und bewirkt – kurz gesagt –, dass mehrere PHP-Aufrufe dieselbe Datenbankverbindung nutzen können, ohne diese ständig neu auf- und abzubauen.

Generell birgt diese Option einige Gefahren, da bei Datenbankproblemen eine Verbindung nicht mehr automatisch neu aufgebaut wird und möglicherweise viele Zugriffe auf die Webseite das Ressourcenlimit an Verbindungen schneller erschöpfen könnten. Daher sollte diese Option nur dann aktiviert werden, wenn Sie sich mit ihren Vor- und Nachteilen auskennen.

Datenbank-Zeichensatzkonvertierung aktivieren

Diese Option gilt nur für MySQL-Datenbanktypen und bringt Serendipity dazu, beim Verbinden mit der Datenbank anzugeben, welchen Zeichensatz das Blog benutzt. Gerade bei aktuellen MySQL-Versionen ab 4.1 kann die falsche Verwendung von Zeichensätzen zu Fehlern führen, siehe Kapitel 2.2.6 ab Seite 67.

Sollte Ihr Blog bei der Darstellung von Sonderzeichen Probleme machen, könnte die Umstellung dieser Option Abhilfe schaffen.

Pfade

Im Bereich **Pfade** tragen Sie einige Optionen zur Festlegung der URLs ein. Bitte beachten Sie, dass Sie an dieser Stelle die derzeit gültigen Pfade eintragen müssen, die mit Ihrer Installation übereinstimmen.

Wenn Sie eine Serendipity-Installation auf dem Server verschieben wollen, müssen Sie dies vorher manuell (mittels FTP-Programm oder Ähnlichem) erledigen und nachträglich die geänderten Pfade an dieser Stelle eintragen.

Bitte achten Sie darauf, dass *alle* Pfadangaben immer mit einem Schrägstrich enden müssen. Etwaige fehlerhaft eingestellte Pfade können dazu führen, dass Links in Ihrem Blog nicht aufgerufen werden können und zu leeren Seiten führen. Auch wenn Ihre Seite unformatiert aussieht und keine Grafiken darstellt, kann das seine Ursache in einem falsch konfigurierten Pfad haben.

Voller Pfad

Diese Option enthält den vollständigen Pfad der Serendipity-Installation auf dem Server. Dieser Pfad entspricht also dem eigentlichen Dateisystem (z. B. `/var/www/example.com/serend` und nicht dem Schema der URL (`/serendipity/`). Bitte stellen Sie sicher, dass der Pfad immer einfache Schrägstriche (/) verwendet und nicht den Backslash (\). Auf

[15] `http://de2.php.net/manual/en/features.persistent-connections.php`

Windows-Systemen können Sie den Backslash einfach mit einem normalen Schrägstrich ersetzen, und lassen Sie das Laufwerk (`C:/`) am Anfang weg.[16]

Wenn möglich, sollten Sonderzeichen in den Verzeichnispfaden vermieden werden, und der Pfad muss *immer* mit einem Schrägstrich enden.

Upload-Pfad

Unterhalb des im **Voller Pfad** eingetragenen Verzeichnisses befindet sich der Ordner, in dem die Dateien der Mediendatenbank gespeichert werden sollen (üblicherweise `uploads`). Wenn Sie Ihre Mediendateien in einem anderen Verzeichnis speichern wollen, können Sie den gewünschten Speicherort relativ zum vollen Pfad hier eintragen.

Sollte das Upload-Verzeichnis außerhalb des Serendipity-Verzeichnisses liegen, dürfen Sie hier nicht einfach einen absoluten Pfad eintragen, sondern müssen mittels `../../` die jeweiligen Verzeichnisebenen höher wechseln. Wenn Sie also Serendipity im Verzeichnis `/var/www/example.com/serendipity/` gespeichert haben und Ihr Upload-Verzeichnis sich in `/var/uploads/` befindet, müssen Sie für den **Upload-Pfad** `../../../uploads/` eintragen. Wenn Sie dieses Verzeichnis umkonfigurieren, müssen Sie bereits hochgeladene Dateien manuell in das neue Verzeichnis verschieben.

Relativer HTTP-Pfad

Der **Relative HTTP-Pfad** gibt die Pfadkomponente der URL an, in der Serendipity installiert wurde. Dabei muss der HTTP-Pfad sämtliche Verzeichnisnamen hinter dem Domain-Namen enthalten. Wäre Serendipity unter `http://example.com/~john/serendipity/` installiert, müsste der relative HTTP-Pfad den Eintrag `/~john/serendipity/` enthalten.

Relativer Template-Pfad

Um Grafiken und Stylesheets der Templates anzuzeigen, muss Serendipity auf den Stammordner zugreifen, in dem sich die Templates befinden. Der **Relative Template-Pfad** gibt dabei die notwendige Verzeichnisstruktur (relativ zum HTTP-Pfad) an.

Sie sollten diese Option möglichst nicht ändern, da an zahlreichen Stellen in Serendipity der Template-Pfad `templates/` fest vorgesehen ist. Eine Änderung macht nur Sinn, wenn Sie mit symbolischen Links auf dem Webserver arbeiten, die unterschiedliche Pfadangaben benötigen.

Relativer Upload-Pfad

Wenn Sie den Upload-Pfad wie im Beispiel oben ändern, wird sich auch dessen URL ändern. Die Option **Relativer Upload-Pfad** gibt dabei den Verzeichnisnamen des Speicherorts der Mediendateien an, relativ zur Stamm-URL. Sollte sich das Serendipity-Verzeichnis auf einem anderen VirtualHost oder einer ganz abweichenden Pfad-Struktur

[16] Den Laufwerksbuchstaben bräuchten Sie nur, wenn Ihr *DocumentRoot* nicht auf demselben Laufwerk liegen würde wie das Verzeichnis von Serendipity. Da Serendipity aber immer innerhalb des *DocumentRoot* liegen muss, kann dieser Fall effektiv nicht eintreten.

befinden, kann dies nur funktionieren, wenn Sie einen symbolischen Link[17] zum Ziel-Upload-Verzeichnis erstellen.

URL zum Blog

Im Eingabefeld **URL zum Blog** tragen Sie die vollständige URL (mit Protokoll und Domainnamen) ein. Dabei können Sie, falls verfügbar, auch gerne `https://` benutzen, wenn Ihr Server dies anbietet.

Sollte Ihr Blog unter mehreren URLs erreichbar sein[18], dann müssen Sie lediglich eine dieser URLs hier konfigurieren und die Option **HTTP-Hostnamen automatisch erkennen** aktivieren (siehe unten).

Auch hier gilt: Wenn Sie Serendipity auf eine andere URL verschieben möchten, müssen Sie die Verzeichnisse und Dateien manuell verschieben und können erst danach diese Konfigurationsoption den neuen Gegebenheiten anpassen.

HTTP-Hostnamen automatisch erkennen

Wenn Ihr Blog unter mehreren URLs erreichbar ist, können Sie Serendipity zwar unter allen drei URLs aufrufen, aber alle Verweise innerhalb des Blogs werden ausschließlich auf die in **URL zum Blog** konfigurierte URL verweisen.

Wenn Sie gerne möchten, dass für jeden Besucher die URL weiterbenutzt wird, die er zum Aufrufen verwendet hat, dann können Sie die Option **HTTP-Hostnamen automatisch erkennen** aktivieren.

Diese Funktion kann jedoch nur dann korrekt funktionieren, wenn sich die Pfadkomponente der URL niemals ändert. Sollte Ihr Blog aber unter den URLs `http://example.com/serendip` und
auch `http://serendipity.example.com/` verfügbar sein, dann wäre dies nicht der Fall. In so einem Fall können Sie nur eine der beiden URLs für den korrekten Betrieb von Serendipity wählen. Mögliche Lösungen dieses Problems finden Sie im Kapitel 2.2.6 auf Seite 65.

Index-Datei

Diese Option legt fest, welche zentrale Datei Serendipity benutzt, um jeden Seitenaufruf dorthin umzulenken.

Üblicherweise sollten Sie diese Option nur ändern, wenn Sie Serendipity in einer anderen Webseite einbetten (siehe Option **Eingebettete Nutzung von Serendipity aktivieren** auf Seite 679).

In der Datei `.htaccess` wird die hier konfigurierte Datei als `ErrorDocument` und `DirectoryIndex` eingebunden. Sollten Sie einmal eine eigene Datei hier an-

[17] Symbolische Links sind häufig auf Linux-Servern zu finden und ermöglichen einen Datei- oder Verzeichnisverweis zu völlig anderen Verzeichnissen. Symbolische Links können nur mit direktem Zugriff zum Dateisystem eines Servers erstellt werden und sind daher meist Administratoren bzw. Shell-Benutzern vorbehalten.

[18] Z. B. `http://example.com/serendipity/`, `https://www.example.com/serendipity/` und `http://blog.example.com/serendipity/`

geben, müssen Sie in dieser Datei sicherstellen, dass das Serendipity-Framework über `index.php` eingebunden wird.

Sie sollten die Option nicht zweckentfremden, um eine eigene HTML-Seite als Startseite für Serendipity festzulegen. Dies würde dazu führen, dass auch andere URLs von Serendipity nicht mehr zugänglich wären. Eine individuelle Startseite für Serendipity können Sie mittels des Plugins *Statische Seiten* einstellen, siehe Kapitel 7.2.5 ab Seite 410.

Permalinks

Sämtliche *Permalinks*[19] können im gleichnamigen Konfigurationsbereich eingestellt werden.

Bei den Permalinks ist es sehr wichtig, dass Sie genau wissen, in welchem Rahmen die Pfade verändert werden können. Es ist bei jedem Permalink wichtig, dass ein festes Verzeichnis-Präfix (wie `archives`, `categories` und andere) für jeden Permalink-Typen definiert wird, damit Serendipity beim Aufruf einer URL erkennen kann, welche Unterseite angefordert wird. Sie können also nicht einfach einen Pfadteil weglassen und Eintrags-URLs ohne vorangestellten Pfad konfigurieren, da Serendipity dann keine Pfad-Zuordnung treffen kann.

Jeder konfigurierte Pfad muss exakt in der Permalink-Struktur übernommen werden, Sie können also **Pfad zu den Einträgen** nicht auf `eintraege` setzen und dann für die Permalink-Struktur `artikel/%id%-%title%.html` einsetzen. In beiden Fällen müssen Sie entweder `eintraege` *oder* `artikel` benutzen. Vermeiden Sie möglichst Sonderzeichen (also auch Leerzeichen) in der URL.

Ein Hinweis noch zu dem Parameter `%id%`, den Sie bei mehreren Permalinks aufgeführt sehen. Obwohl dies später im Frontend als sprechende URL möglicherweise nicht *sexy* aussieht, birgt die Verwendung von `%id%` einen großen Performancevorteil. Denn sobald diese ID enthalten ist, kann Serendipity ohne weitere Datenbankabfrage herausfinden, welche Inhalte es anzeigen muss.

Bei der Umstellung von Permalinks ist darauf zu achten, dass dadurch die vorher gültigen URLs nicht mehr aktiv sind! Sollte eine Suchmaschine Ihre Seiten also bereits indiziert haben, kann eine Umstellung der Permalinks bedeuten, dass die Suchmaschine sämtliche Ihrer Seiten aus dem Index wirft und neu bewerten muss. Legen Sie sich daher möglichst früh auf Ihr gewünschtes URL-Muster fest.

Die Seiten, die sich hinter den jeweiligen Permalinks befinden, sind ausführlich im Kapitel 2 ab Seite 77 beschrieben.

Permalink-Struktur für die Artikel-URLs

Hier wird die Struktur des Permalinks eingetragen, die beim Aufrufen eines einzelnen Artikels verwendet wird. Platzhalter dienen jeweils dazu, dynamische Teile in die URL zu übernehmen:

[19] Permalinks sind sprechende URLs (siehe Seite 168), die im Frontend eindeutig zu den jeweiligen Funktionalitäten führen.

`%id%`
: für die ID eines Artikels.

`%title%`
: für den Titel eines Artikels.

`%day%`
: für den Tag, an dem der Artikel verfasst wurde.

`%month%`
: für den Monat, in dem der Artikel verfasst wurde.

`%year%`
: für das Jahr, in dem der Artikel verfasst wurde.

Diese (und auch alle folgenden) Permalink-Strukturen können weitere beliebige Pfadkomponenten enthalten.

Permalink-Struktur für Autoren-URLs
Für die Darstellung der Einträge eines Autors wird dieser Permalink verwendet. Folgende Variablen sind verfügbar:

`%id%`
: für die ID eines Autors.

`%realname%`
: für den Benutzernamen eines Autors, wie er üblicherweise bei Artikeln angezeigt wird.

`%username%`
: für den Login-Namen eines Autors.

`%email%`
: für die E-Mail-Adresse eines Autors.

Permalink-Struktur für Kategorie-URLs
Dieser Permalink stellt Einträge pro Kategorie dar. Verfügbare Variablen sind:

`%id%`
: für die ID einer Kategorie.

`%name%`
: für den Namen einer Kategorie.

`%description%`
: für die Beschreibung einer Kategorie (Vorsicht bei langen URLs durch lange Beschreibungen!).

`%email%`
: für die E-Mail-Adresse eines Autors.

Permalink-Struktur für RSS-Kategorien-Feed URLs
Die Artikel einer Kategorie für einen RSS-Feed können über diese konfigurierte URL aufgerufen werden. Es sind dieselben Variablen wie für den normalen Kategorie-Permalink verfügbar.

Permalink-Struktur für RSS-Autoren-Feed URLs
Die Artikel eines Autors für einen RSS-Feed werden in diesem Permalink festgehalten. Es gelten dieselben Variablen wie für die normalen Autoren-Permalinks.

Pfad zu den Einträgen
Der erste Verzeichnisname für alle Artikelübersichten, standardmäßig `archives`.

Pfad zu den Archiven
Der Verzeichnisname für die Archivübersicht, in der die Artikel nur chronologisch dem jeweiligen Monat zugeordnet sind.

Pfad zu den Kategorien
Der Verzeichnisname für die Artikelübersicht einer Kategorie.

Pfad zu den Autoren
Der Verzeichnisname für die Artikelübersicht eines Autors.

Pfad zum Abbestellen/Löschen/Bewilligen von Kommentaren
Wenn Sie als Autor oder Administrator über den Eingang eines neuen Kommentars per E-Mail benachrichtigt werden, können Sie innerhalb der E-Mail auf Links klicken, die den Kommentar freischalten oder löschen. Für Personen, die neue Kommentare bei einem Artikel abonniert haben, gibt es zusätzlich einen Link, der dieses Abonnement wieder aufhebt. Alle drei Pfadvariablen können über diese Konfigurationsfelder angepasst werden.

Pfad zu den RSS-Feeds
Der Verzeichnisname für alle RSS-Feeds.

Pfad zu einem externen Plugin
Ereignis-Plugins können eigenständige, neue Unterseiten in Ihrem Blog zur Verfügung stellen. Diese Unterseiten können unabhängig vom Serendipity-Layout von einem Plugin ausgegeben werden; unter anderem werden die Anti-Spam-Grafiken vom Spamblock-Plugin derart ausgegeben.

Dafür benutzen die Plugins das Unterverzeichnis `plugin`. Sie sollten dieses Verzeichnis möglichst nicht ändern, da es mehrere Plugins gibt, die immer fest auf das Verzeichnis `plugin` zugreifen und möglicherweise nicht mehr funktionieren, wenn Sie den Verzeichnisnamen ändern!

Pfad zur Administration
Der Pfad, den Sie aufrufen, um das Backend (die Administrationsoberfläche) hervorzubringen.

Pfad zur Suche
Der Pfad zur Suche nach Begriffen in Artikeltexten.

Pfad zu Kommentaren
Der Pfad zur Übersicht von Kommentaren zu Artikeln.

Generelle Einstellungen

Weitere globale Einstellungen befinden sich im Bereich **Generelle Einstellungen**.

Abonnieren von Einträgen erlauben?
Wenn Sie es Ihren Besuchern erlauben wollen, dass sie per E-Mail über Kommentare benachrichtigt werden, muss diese Option aktiviert sein. Ein Benutzer kann Kommentare jedoch nur abonnieren, wenn er selber zu einem Beitrag kommentiert hat.

Blog-Titel

Der **Blog-Titel** legt die Überschrift Ihres Blogs fest. Hier können Sie auch Sonderzeichen einfügen, jedoch keinen HTML-Code oder Grafiken. Eine derartige Sonderdarstellung müssen Sie über Anpassungen in Ihrem Template vornehmen (siehe Seite 489).

Blog-Beschreibung

Die Blog-Beschreibung wird unterhalb des Blog-Titels im Frontend angezeigt. Auch hier können Sie Sonderzeichen einfügen, aber keinen HTML-Code.

E-Mail-Adresse des Blogs

Wenn Serendipity E-Mails an Autoren oder Kommentatoren schickt, muss eine E-Mail-Adresse für den Absender eingetragen werden. Da Serendipity E-Mails über den Webserver verschickt, muss die hier eingetragene E-Mail-Adresse auch vom Mailserver als gültig anerkannt werden. Ansonsten könnte der Mailserver Ihre E-Mail als Spam deklarieren und verwerfen. Meist können an dieser Stelle nur E-Mail-Adressen eingetragen werden, die auch auf demselben Server gehostet werden.

Sollte Serendipity einmal keine Mails an Sie verschicken, prüfen Sie, ob Sie hier eine gültige Adresse eingetragen haben und ob Ihr Webserver auch in der Lage ist, E-Mails mittels *sendmail*[20] zu versenden. Auf Windows-Servern muss ein Mailserver in der `php.ini`-Datei eingetragen werden.

Sprache

An dieser Stelle können Sie die Sprache der Oberfläche des Blogs wählen. Serendipity zeigt dann alle eigenen Meldungen in dieser Sprache an.

Im Gegensatz zur Spracheinstellung im Menü **Eigene Einstellungen** wird hier nur die Sprache für Besucher des Blogs eingestellt. Ihre persönliche Spracheinstellung „überschreibt“ diesen Wert.

Zeichensatz-Auswahl

Serendipity ermöglicht die Darstellung aller Meldungen und Ihrer eigenen Artikel in verschiedenen Zeichensätzen. Alle Sprachen können im Zeichensatz *UTF-8* abgebildet werden, daher ist dieser Zeichensatz mittlerweile der Standard im Internet. Einige andere Sprachen, besonders Chinesisch, verwenden jedoch eigene, nationale Zeichensätze zur korrekten Darstellung. Wenn Sie in dem Auswahlfeld dieser Option **Nationaler Zeichensatz** wählen, wird dieser spezielle nationale Zeichensatz benutzt (abhängig von der jeweils gewählten Sprache).

Wenn Ihr Blog bereits Artikel enthält, wird die Umstellung dieser Option dazu führen, dass die Sonderzeichen Ihrer Artikel falsch angezeigt werden. Sie müssen daher die Konvertierung in diesem Fall manuell vornehmen, wie im Kapitel 2.2.6 ab Seite 67 beschrieben.

[20] Sendmail ist ein Unix-Mailbefehl, der von PHP nur genutzt werden kann, wenn dieses Paket auf dem Server installiert ist.

Kalendertyp

Für einige Kulturen gibt es möglicherweise unterschiedliche Arten der Kalender-Darstellung. Für Serendipity wurde daher der persische Kalender als Alternative zum bekannten gregorianischen Kalender eingebaut und kann hier ausgewählt werden.

Sprache des Browsers eines Besuchers verwenden

Wenn diese Option aktiviert ist, wird Serendipity die Sprache des Frontends abhängig von der im Browser des Besuchers eingestellten Sprache auswählen. So können Sie es Ihren Besuchern erleichtern, die umgebenden Texte zu verstehen, wenn sie Ihre Artikel in der Landessprache nicht lesen können. Bitte beachten Sie, dass diese Sprachauswahl die Artikel nicht automatisch übersetzen kann.

Sollen persönliche Plugin-Rechte für Benutzergruppen aktiviert werden?

Serendipity ermöglicht es, pro Benutzergruppe zu definieren, welche Plugins von dieser Gruppe im Backend ausgeführt werden dürfen. Über die Definition in der **Gruppenverwaltung** kann verhindert werden, dass bestimmte Autoren zum Beispiel Zugriff zu dem Plugin *Statische Seiten* haben.

Da es in Blogs relativ selten vorkommt, dass Sie den Zugriff derart strikt eingrenzen müssen, ist diese Option standardmäßig nicht aktiviert und muss daher bewusst von Ihnen eingeschaltet werden. Das Einschalten dieser Option kann die Performance von Serendipity spürbar negativ beeinflussen, da für jede Ausführung eines Plugins nun erst geprüft werden muss, ob Zugriff darauf erlaubt ist.

Wenn Sie also auf diese Flexibilität verzichten können, sollten Sie die Option **Sollen persönliche Plugin-Rechte für Benutzergruppen aktiviert werden** weiterhin auf **Nein** stellen.

Design und Optionen

Optionen, die das Aussehen des Blogs betreffen, sind im Bereich **Design und Optionen** zusammengefasst.

Anzahl der Artikel auf der Startseite

Standardmäßig zeigt Serendipity die letzten 15 Artikel auf der Startseite. Über die Einstellung **Anzahl der Artikel auf der Startseite** können Sie diese Anzahl kontrollieren, die auch für alle Übersichtsseiten angewendet wird.

Einträge im Feed

Analog zu der Anzahl der Artikel auf der Startseite bestimmt die Option **Einträge im Feed**, wie viele Artikel in Ihrem RSS-Feed standardmäßig enthalten sind.

Eine hohe Einstellung dieser Zahl stellt Ihren Besuchern zwar mehr Einträge dar, vergrößert aber auch die Dateigröße des RSS-Feeds (und damit den Traffic) erheblich.

Strikte RFC2616 RSS-Feed Kompatibilität

Wie im Kapitel 3.6 auf Seite 87 erklärt, können RSS-Feeds bei Ihren Besuchern zwischengespeichert werden. Dieses Caching macht möglicherweise bei einer kleinen Anzahl von Software-Programmen Probleme. Sollte die benutzerseitige Deaktivierung des Cachings (siehe erwähntes Kapitel) nicht ausreichen, können Sie die Option **Strikte RFC2616 RSS-Feed Kompatibilität** aktivieren.

Serendipity verhält sich dann absolut standardgemäß, verzichtet jedoch auf das zusätzliche Feature der fallweisen Artikelauslieferung abhängig vom letzten Besuchsdatum.

Zwei weitere Einstellungen, die das Caching beeinflussen, können Sie auf Seite 179 nachschlagen.

GZIP-Kompression verwenden

Wenn Serendipity den HTML-Code des Frontends an den Browser Ihrer Besucher ausgibt, wird dabei eine Menge an Klartext-Daten übertragen. Klartext lässt sich mittels ZIP-Kompression relativ stark bündeln und bietet sich daher zum Einsparen von Traffic an. Gerade bei Besuchern mit niedriger Bandbreite (Benutzer von Modems, GPRS-Verbindungen) kann die GZIP-Komprimierung einige Vorteile bringen.

Leider benötigt das Packen mit GZIP-Kompression einiges an CPU-Leistung. Daher muss man abwägen, ob der eigene Server eher einen Engpass bei der CPU-Leistung oder bei der Trafficmenge hat. Zwar wird eine Aktivierung der GZIP-Kompression weniger Daten übertragen, aber die Performance von Serendipity wird dadurch sinken.

Hier müssen Sie eine Entscheidung treffen. Da Bandbreite heutzutage ein selteneres Problem ist, empfehle ich, zugunsten der Servergeschwindigkeit diese Option zu deaktivieren.

Popups für Kommentare, Trackbacks usw. verwenden?

Blendet Serendipity im Frontend einen Link auf Kommentare und Trackbacks eines Artikels ein, können diese Links entweder auf die vollständige Artikelansicht zeigen oder auch ein Popup-Fenster mit der gewünschten Darstellung öffnen.

Wenn Sie diese Popups benutzen wollen, aktivieren Sie die Option **Popups für Kommentare, Trackbacks usw. verwenden**. Beachten Sie jedoch, dass Popups bei vielen Internet-Nutzern verpönt sind und es daher empfehlenswert ist, Ihren Besuchern die Wahl zu lassen, ob sie etwas als Popup öffnen möchten oder nicht. Jeder moderne Browser bietet dazu das Öffnen von Links in einem neuen Fenster mit einfachem Mausklick der mittleren Taste an.

Eingebettete Nutzung von Serendipity aktivieren?

Wenn Sie Serendipity in eine andere Anwendung einbetten wollen, können Sie die Ausgaben Serendipitys zentral speichern und in der PHP-Anwendung weiterverarbeiten. Damit diese Einbindung gelingt, darf Serendipity dann keine HTML-Kopf- und -Fußzeilen senden.

Die Aktivierung dieser Option bietet genau dies: Serendipity sendet dann nur den eigentlichen Inhalt des Blogs, der von der fremden Anwendung im Zusammenspiel mit

der Option **Index-Datei** (siehe Seite 160) genutzt werden kann.

Diese Option war besonders für ältere Serendipity-Versionen notwendig. Seit es Smarty-Templates gibt, kann man diese Templates auch leicht so anpassen, dass keine Kopf- und Fußzeilen mehr gesendet werden müssen. Schlagen Sie im Kapitel 10.9 ab Seite 673 nach, um eine Anleitung zur Einbettung von Serendipity zu erhalten.

Sollte Ihr Serendipity-Blog ein merkwürdiges Layout aufweisen, ist womöglich die Aktivierung dieser Option „schuld“ daran.

Externe Links klickbar?

Für die Seitenleisten-Plugins `Top Exits` und `Top Referrer` kann man mittels dieser Option einstellen, ob in diesen Plugins dargestellte Links mittels HTML klickbar dargestellt werden oder nicht.

Eigentlich gehört diese Option mittlerweile in die Konfigurationsseite der beiden Plugins. Aus historischen Gründen ist die Option jedoch in der globalen Konfiguration enthalten und wird möglicherweise in zukünftigen Versionen verschwinden.

Referrer-Tracking aktivieren?

Wenn Sie die Option **Referrer-Tracking** aktivieren, wird bei jedem Aufruf einer Serendipity-Seite ausgewertet, von welcher Seite ein Besucher kam. Diese Daten können dann in den Statistiken verwendet werden und zeigen Ihnen, von welchen Webseiten häufig zu Ihnen verlinkt wird.

Leider wurde diese Option im Laufe der Zeit sehr oft durch Spammer missbraucht und zeigt nur noch in seltenen Fällen wirklich nutzbare Daten an.

Geblockte Referrer

Um ein wenig Kontrolle auf Referrer-Spammer auszuüben, können Sie in diesem Eingabefeld mehrere Stichwörter eintragen, die Sie mittels *;* voneinander trennen. Sobald eines dieser Stichwörter in der URL eines Besuchers vorkommt, wird diese URL nicht mit in die Referrer-Statistik übernommen.

URL-Formung

Die Option **URL-Formung** ist zuständig, um *sprechende URLs* zu aktivieren (siehe Seite 30). Ob Serendipity URLs hübsch formatieren kann, hängt davon ab, ob Ihr Webserver dies unterstützt.

Die Unterstützung können Sie ausprobieren, indem Sie testweise einfach die Methoden zur URL-Formung aktivieren und die Konfiguration abspeichern. Besuchen Sie danach Ihr Frontend und prüfen Sie, ob der Link zu einer Artikel-Detailseite noch funktioniert.

Wenn Sie Fehlermeldungen erhalten oder dies nicht klappt, müssen Sie die URL-Formung wieder deaktivieren. Die Administrationsoberfläche können Sie immer unter `http://www.example.com/serendipity/serendipity_admin.php` auch im Fehlerfall aufrufen. Möglicherweise müssen Sie dazu die Datei `.htaccess` löschen, wenn bei Ihrem Webserver sonst alle Aufrufe fehlschlagen.

Folgende Arten der URL-Formung sind möglich:

Disable URL Rewriting
Bei dieser Methode werden sprechende URLs deaktiviert. Ein Link sieht dann aus wie: `http://www.example.com/serendipity/index.php?/archives/1-MeinArtikel.htm`
Diese URLs mögen für Sie zwar *sprechend* aussehen, aber für eine Suchmaschine wie Google sind sie es nicht und können daher nicht zur Aufwertung der Auffindbarkeit dienen (siehe Kapitel 47).

Use Apache ErrorHandling
Auf vielen Apache-basierten Webservern ist diese Methode einsetzbar. Mittels eines kleinen Tricks werden Fehlerseiten dazu benutzt, um die eigentlich nicht existierende virtuelle Seite aufzurufen. Eine URL sieht dann aus wie: `http://www.example.com/serendipity/archives/1-MeinArtikel.html`
Der Nachteil dieser Methode ist, dass für jeden URL-Aufruf in Ihrem Apache Fehler-Logfile ein Eintrag erscheint, was man als *unsexy* bezeichnen könnte.

Use Apache mod_rewrite
Die performanteste Methode der URL-Formung stellt **Use Apache mod_rewrite** dar. Sie produziert identische URLs wie das Apache ErrorHandling, ist aber flexibler und erzeugt keine Fehler-Logfile-Einträge.

Zeitunterschied des Servers
Wenn Ihr Webserver in einer anderen Zeitzone als Ihr eigener Computer steht, kann es zu einem Zeitversatz kommen. Ihr Server würde Artikel, die Sie um 15:00 Uhr Ortszeit erstellen, je nach Serverzeit möglicherweise für 03:00 Uhr morgens auszeichnen.

Um diesen Zeitversatz zu korrigieren, können Sie in dem Eingabefeld **Zeitunterschied des Servers** eintragen, wie viele Stunden Zeitunterschied zwischen Ihrem Computer und dem Server liegen. Um diesen Zeitunterschied herauszufinden, wird die aktuelle Serverzeit im Text neben der Eingabebox für Sie angezeigt.

Sie können halbe Stunden mit einer Angabe wie `1.5` eintragen. Negative Zeitunterschiede geben Sie mit einem vorangestellten `-` an.

Zukünftige Einträge zeigen
Üblicherweise wird Serendipity nur Artikel darstellen, deren Uhrzeit nicht vor der aktuellen Serverzeit liegt. So können Sie zukünftige Einträge bereits verfassen, ohne dass Sie zum jeweiligen Zeitpunkt manuell etwas freischalten müssen.

Wenn Sie jedoch auch zukünftige Artikel anzeigen wollen (zum Beispiel für ein Blog mit in der Zukunft stattfindenden Ereignissen), können Sie die Option **Zukünftige Einträge zeigen** aktivieren.

Leserechte auf Kategorien anwenden
Serendipity unterstützt die Möglichkeit, Artikel in Kategorien mit einem Leseschutz zu versehen. So können Sie einer Kategorie Leserechte nur für bestimmte Benutzergruppen zuweisen, und alle anderen Benutzergruppen (insbesondere anonyme Besucher des Frontends) können dann Artikel in dieser Kategorie nicht lesen. Dies ermöglicht es, Serendipity auch als CMS einzusetzen.

In vielen Blogs ist ein derartiger Zugriffsschutz jedoch nicht notwendig, da alle Artikel von allen Besuchern gelesen werden sollen. Ist dies der Fall, können Sie die Option **Leserechte auf Kategorien anwenden** gerne auf **Nein** setzen. Dies wird die Geschwindigkeit der Darstellung im Frontend positiv beinflussen, da weitaus weniger Datenbankabfragen ausgeführt werden müssen, um die Artikelübersicht zu erzeugen.

Bildkonvertierung

Der letzte Bereich, **Bildkonvertierung**, legt einige Optionen für die Mediendatenbank und die Vorschaubild-Erzeugung fest.

ImageMagick zur Skalierung verwenden

Wenn Sie auf Ihrem Webserver die Software ImageMagick[21] installiert haben, können Sie diese Software zur Erstellung und Konvertierung von Vorschaubildern benutzen.

Ist ImageMagick nicht aktiviert, kann Serendipity auch die PHP-Bibliothek `gdlib` einsetzen. ImageMagick bietet den Vorteil, dass auch Vorschaubilder von PDF-Dateien erzeugt werden können. Bis auf diesen Vorteil sind gdlib und ImageMagick gleichwertig.

Pfad zur convert ImageMagick-Datei

In diesem Eingabefeld müssen Sie den vollständigen Pfad zur ImageMagick-Datei auf dem Webserver eintragen. ImageMagick kann nur ausgeführt werden, wenn der Webserver-Benutzer für diese Datei Zugriffsrechte gewährt. Gerade bei aktiviertem Safe Mode ist dies selten der Fall. Deaktivieren Sie ImageMagick, falls Fehlermeldungen bei der Vorschaubild-Erzeugung auftreten.

Thumbnail-Endung

Als Thumbnail bezeichnet man die Vorschaugrafiken, die Serendipity von einer Bilddatei automatisch erzeugt.

Jede dieser Vorschaugrafiken enthält standardmäßig den Namenszusatz `serendipityThumb`. Die Vorschaudatei zum Bild `logo.jpg` wird also `logo.serendipityThumb.jpg` heißen und im selben Verzeichnis angelegt werden.

Diesen Dateinamen können Sie über die Option **Thumbnail-Endung** selber festlegen. Wenn Sie den Dateinamen ändern und bereits Dateien in der Mediendatenbank vorhanden sind, müssen Sie im Menü **Mediendatenbank** auf **Vorschauen erneuern** klicken.

Thumbnailgröße

Jede Grafikdatei wird standardmäßig auf maximal 110 Pixel Breite oder Höhe für die Vorschaugrafik verkleinert. Der jeweils größere Wert (Breite oder Höhe) wird dabei auf 110 Pixel verkleinert und der jeweils kleinere Wert unter Berücksichtigung der Bildproportionen errechnet.

[21] `http://www.imagemagick.org`

Serendipity zeigt die Vorschaubilder in der Mediendatenbank stets in der hier konfigurierten Größe an. Wenn Sie also größere (oder kleinere) Vorschaubilder bevorzugen, können Sie die maximale Größe unter **Thumbnailgröße** einstellen. Auch hier müssen Sie auf **Vorschauen erneuern** klicken, wenn Sie diesen Wert ändern, obwohl Sie schon Dateien in die Mediendatenbank eingestellt haben.

Maximale Dateigröße für den Upload
Wenn Sie den Upload von Dateien künstlich einschränken wollen, können Sie die maximale Dateigröße einer in die Mediendatenbank hochgeladenen Datei über das Feld **Maximale Dateigröße für den Upload** einstellen.

Beachten Sie, dass Sie diesen Wert nicht größer einstellen können als die ebenfalls limitierenden Konfigurationsoptionen `upload_max_filesize`, `post_max_size` und `max_input_time` der `php.ini` (siehe Kapitel 1.3.2 auf Seite 33).

Maximale Breite eines hochgeladenen Bildes

Maximale Höhe eines hochgeladenen Bildes Abgesehen von der Dateigröße können Sie ein hochgeladenes Bild auch auf eine maximale Auflösung festlegen. Wenn das Bild eine dieser Dimensionen überschreitet, wird es nicht akzeptiert. So können Sie verhindern, dass Redakteure Bilder hochladen, die für den Einsatz im Internet nicht geeignet sind.

Automagische Synchronisation der Mediendatenbank
Da Serendipity, wie im Kapitel 4.5 ab Seite 124 beschrieben, den Dateibestand des Upload-Verzeichnisses mit einer eigenen Datenbanktabelle abgleicht, kann es hier zu Unterschieden kommen.

Serendipity kann bei jedem Aufruf der Mediendatenbank prüfen, ob möglicherweise Dateien hinzugekommen oder gelöscht worden sind. Dazu werden alle Verzeichnisse und Dateien auf dem Server mit der Datenbank abgeglichen. Serendipity optimiert den Zugriff dabei, indem es diese Aktion nur ausführt, wenn sich etwas an der Menge der Dateien und Unterverzeichnisse getan hat.

Dieser Vorgang kann bei einer großen Mediendatenbank möglicherweise zu Ressourcen-Engpässen führen. In so einem Fall müssen Sie die **automagische Synchronisation** deaktivieren.

Wenn der automatische Synchronisationsvorgang eine neue Datei findet oder eine alte Datei löscht, wird dies direkt innerhalb der Mediendatenbank dargestellt.

Dynamische Bildgrößenanpassung erlauben
Üblicherweise kann Serendipity von einem Bild entweder nur die Originalgrafik oder die Vorschaugrafik zurückliefern.

Ein Teil der Mediendatenbank kann aber auch (manuell) über das Frontend angesprochen und benutzt werden, um Grafiken auch in jeder anderen beliebigen Dateigröße auszugeben (siehe Seite 687).

EXIF/JPEG Metadaten übernehmen?
Wenn Sie ein Bild in die Mediendatenbank von Serendipity hochladen, kann Serendipity automatisch die Binärdaten des Bildes auswerten und in der Mediendatenbank speichern. In solchen Binärdateien können sogenannte `EXIF`-Daten[22] gespeichert werden,

[22] `http://www.exif.org/`

die z. B. von Digitalkameras automatisch eingefügt werden und Informationen über Belichtungszeit und Aufnahmezeitpunkt enthalten.

Serendipity kann diese Metadaten innerhalb der Mediendatenbank anzeigen. Da das Auslesen der Binärdaten einiges an Server-Ressourcen beanspruchen kann, könnte dies möglicherweise zu sehr großen Datenmengen in der Datenbank (dort werden die Metadaten zwischengespeichert) oder auch dazu führen, dass das Hochladen von Dateien fehlschlägt. Deaktivieren Sie in diesem Fall die Option **EXIF/JPEG Metadaten übernehmen**.

Medien-Eigenschaften

Zu jeder Datei in der Serendipity-Mediendatenbank können Sie eine selbst definierte Menge von möglichen Metadaten eintragen.

Standardmäßig greift Serendipity dabei auf die Felder `DPI` (nur bei Bildern), `Laufzeit` (nur bei Video/Audio), `Datum`, `Copyright`, `Titel`, `Kurzer Kommentar` und `Langer Kommentar` zurück.

Diese verwendeten Felder werden in der Eingabebox **Medien-Eigenschaften** festgelegt. Die Eingabebox mag für Sie etwas kryptisch erscheinen, ist aber für Serendipity ein sehr schneller Weg, ohne umständliche Zusatzoberfläche Ihre Eingaben zu erfassen.

Alle gewünschten Felder werden dabei mit einem `;`-Zeichen getrennt: `DPI;DATE;COPYRIGHT;TITLE;COMMENT1;`

Jedes Feld kann einen beliebigen Namen haben, der aber keine Sonderzeichen und Leerzeichen beinhalten darf. Wie der Name eines Feldes später in der Mediendatenbank angezeigt wird, richtet sich ebenfalls nach diesem Feld. Serendipity sucht dabei in den Sprachdateien (siehe Kapitel 10.4, Seite 599) nach einer definierten Konstante `MEDIA_PROPERTY_`*`DPI`*. Ist diese Konstante definiert, wird die darin definierte Bezeichnung in der Mediendatenbank angezeigt. Gäbe es keine solche Konstante, würde der Begriff selbst dargestellt werden. Wenn Sie also das Feld `QUALITAET` am Ende einfügen, wird in der Mediendatenbank auch `QUALITAET` angezeigt. Um eine eigene, klarere Bezeichnung zu wählen, müssen Sie die Konstante selbst definieren, wie im angesprochenen Kapitel erwähnt.

Nun kann jedes aufgeführte Feld noch einige Optionen aufweisen, die jeweils durch das `:`-Zeichen getrennt werden. Vier Optionen sind verfügbar:

`:IMAGE`
: wenn das vorangehende Feld nur für Bilddateien benutzt werden soll.

`:VIDEO`
: wenn das vorangehende Feld nur für Videos benutzt werden soll.

`:AUDIO`
: wenn das vorangehende Feld nur für Sounddateien benutzt werden soll.

`:MULTI`
: wenn eine Medien-Eigenschaft einen längeren Eingabetext anstelle nur einer einzeiligen Eingabebox zulassen soll.

Beispielsweise möchten Sie gerne für Bilder ein Eingabefeld *Bildqualität*, für Videos *Bewegungsqualität* und für alle Dateien pauschal ein großes Eingabefeld *Einsatzzweck* speichern. Dafür würden Sie folgende Konfiguration vornehmen:

```
Bildqualitaet:IMAGE;Bewegungsqualitaet:VIDEO;Einsatzzweck:MULTI
```

Medien-Schlüsselwörter

Jeder Datei in der Datenbank können Sie über die Eigenschaftsoberfläche einer Liste von selbst definierten Schlüsselwörtern zuordnen. Dabei können mehrere Schlüsselwörter einer einzelnen Datei zugewiesen werden, und die Mediendatenbank kann später auch nach diesen Schlüsselwörtern durchsucht werden.

Schlüsselwörter werden fest in der Eingabebox **Medien-Schlüsselwörter** vorgegeben. Alle möglichen Schlüsselwörter müssen Sie mit einem Semikolon voneinander trennen. Ein einzelnes Schlüsselwort darf in diesem Fall auch Sonderzeichen und Leerzeichen enthalten.

Über den Button **Testen & speichern** können Sie die geänderte Konfiguration sichern. Falls Sie Änderungen an den Datenbankparametern vorgenommen haben, wird Serendipity die Datei `serendipity_config_local.inc.php` neu speichern. Bei Änderungen an der Permalink-Struktur oder den Pfaden wird zusätzlich die Datei `.htaccess` erneut erstellt.

Durch einen Klick auf diesen Button können Sie daher auch eine versehentlich gelöschte `.htaccess`-Datei wieder neu erzeugen.

Profi-Einstellungen

Abgesehen von diesen menügesteuerten Einstellungsoptionen besitzt Serendipity noch einige *versteckte* Optionen. Diese legt man in den Dateien `serendipity_config_local.inc.php` und `serendipity_config.inc.php` fest. Dafür müssen Sie also die beiden genannten Dateien mit einem Editor öffnen und ändern. Innerhalb der Datei sind einige Werte festgelegt, die sich alle auf unterschiedliche `$serendipity[...]`-Variablen beziehen.

Diese Variablen sind im Folgenden aufgeführt. Einige von ihnen werden über die Datei `serendipity_config.inc.php` vorbelegt, die Sie jedoch im Bedarfsfall über die eigene Konfigurationsdatei `serendipity_config_local.inc.php` überschreiben können.

`$serendipity['versionInstalled']`

In dieser Variable wird die aktuell installierte Serendipity-Versionsnummer gespeichert. Diese müssen Sie höchstens dann manuell anpassen, wenn Sie ein gescheitertes Update neu ausführen wollen!

`$serendipity['dbName']`

`$serendipity['dbPrefix']`

`$serendipity['dbHost']`

`$serendipity['dbUser']`

`$serendipity['dbPass']`

`$serendipity['dbType']`

`$serendipity['dbPersistent']` Enthält die Zugangsdaten und Parameter der Datenbankverbindung, mit der Serendipity installiert ist. Diese Variablen können normal über die Serendipity-Konfiguration eingestellt werden und sollten daher in dieser Datei nur geändert werden, wenn Sie die Datenbank oder den Server gewechselt haben und nicht mehr auf Ihre Serendipity-Installation zugreifen können.

`$serendipity['noautodiscovery']`
Wenn Serendipity nicht versuchen soll, automatische Trackbacks an URLs zu schicken, die Sie in Ihren Artikeln angegeben haben, können Sie die Variable `$serendipity['noautodiscovery']` auf `true` setzen. Standardmäßig ist die Variable überhaupt nicht gesetzt und daher deaktiviert (`false`).

`$serendipity['pingbackFetchPage']`
Wenn Serendipity einen Pingback (siehe Seite 433) empfängt, bedeutet dies, dass ein fremdes Blog sich auf Ihr Blog bezieht, aber kein Trackback senden möchte. Ein Pingback ist somit eine reduzierte Form des Trackbacks.

Serendipity kann ab Version 1.3 Pingbacks vollständig auswerten. Da ein Pingback üblicherweise nur die URL des fremden Blogs enthält, kann es im Gegensatz zu einem Trackback keine Textauszüge mitliefern. Somit würden Sie nur die URL des fremden Blogs in Ihrem Artikel auffinden.

Um Textauszüge zu beziehen, kann Serendipity die fremde URL öffnen und dort die ersten Zeichen des Textes mit einbinden. Damit dies erfolgen kann, muss die Variable `$serendipity['pingbackFetchPage']` auf `true` gesetzt werden.

Standardmäßig ist diese Variable nicht gesetzt, da das Öffnen einer fremden URL zu einem Performanceverlust des Blogs führen kann und auch DDoS-Angriffe[23] erleichtert.

`$serendipity['pingbackFetchPageMaxLength']`
Bei empfangenen Pingbacks wird standardmäßig nur eine beschränkte Anzahl an Zeichen der sendenden Blog-Seite abgerufen. Diese Zeichen werden später verwendet, um den Inhalt des Pingbacks zu füllen. Die Anzahl der Zeichen, die ausgelesen werden, können Sie über die Variable `$serendipity['pingbackFetchPageMaxLength']` steuern, standardmäßig sind dies 200 Zeichen.

[23] Rasche Zugriffe in Folge von mehreren automatisierten Browsern, die sämtliche Ressourcen Ihres Webservers aufbrauchen oder durch manipulierte URLs Timeouts erzeugen können.

`$serendipity['referrerXSRF']`

Serendipity verfügt über zwei Methoden zur Sicherung des Blogs gegen *XSRF-Angriffe* (siehe Seite 93). Wenn Sie die Variable `$serendipity['referrerXSRF']` auf `true` setzen, wird Serendipity bei einer fehlenden HTTP-Referrer-Kopfzeile (vom Browser übermittelt) eine angeforderte Aktion nicht ausführen. Diese Option können Sie aktivieren, wenn Sie Ihr Blog gegen zusätzliche Attacken sichern möchten; Sie müssen aber sicherstellen, dass alle Redakteure des Blogs in ihrem Browser die Übermittlung von HTTP-Referrern aktiviert haben und auch kein etwaiger Proxyserver diese Kopfzeilen herausfiltert.

Ist der Wert auf `false` gesetzt, wird Serendipity eine Warnmeldung ausgeben, sobald Ihr Browser oder Proxyserver bei der Ausführung einer Aktion keinen HTTP-Referrer ausliefert.

`$serendipity['expose_s9y']`

Standardmäßig ist diese Variable auf `true` gesetzt. Dies bedeutet, dass Serendipity einige HTTP-Header an den Browser übermittelt, die darauf hindeuten, dass Serendipity auf dem Server eingesetzt wird. Wenn Sie dies aus Sicherheitsgründen vermeiden wollen, können Sie die Variable auf `false` setzen.

`$serendipity['useHTTP-Auth']`

Diese Variable ist standardmäßig auf `true` gesetzt und Serendipity ermöglicht damit eine HTTP-Authentifikation. Diese HTTP-Authentifikation benutzt die Abfrage eines Benutzernamens und Passworts und ist Teil des HTT-Protokolls. Ihr Browser zeigt daher für die Autorisation ein Popup-Fenster an, das den Benutzernamen und das Passwort abfragt. Die eingegebenen Daten dienen dann für den Login eines Serendipity-Benutzers und können alternativ zum Login über das Backend dienen.

Der große Vorteil der HTTP-Autorisation ist, dass man sich auch ohne Interaktion des Benutzers mit dem Aufrufen einer URL einloggen kann:

```
http://John+Doe:john@www.example.com/serendipity/
```

Benutzername und Passwort müssen der URL vorangestellt werden. Besonders hilfreich ist diese Art der Authentifikation für RSS-Feeds. Da diese von externen Programmen (RSS-Readern) aufgerufen werden, können Sie häufig nicht auf Ihren vorhandenen Login ins Backend zugreifen. Ohne diesen Login kann Serendipity nicht zuordnen, welcher Benutzergruppe man angehört, und so würde man eventuell lesegeschützte Artikel nicht lesen können. Gibt man als URL für den RSS-Reader jedoch diese authentifizierte URL an, kann der Login sozusagen durchgereicht werden.

Wenn Sie das Authentifikations-Popup des Browsers verwenden wollen, können Sie an jede URL den Parameter `http_auth=true` anhängen, also zum Beispiel:

```
http://www.example.com/serendipity/index.php?http_auth=true
```

Die HTTP-Authentifikation kann nur bei Apache-Webservern benutzt werden, bei denen PHP als Modul (nicht als CGI) eingebunden wurde. Diese Art der Authentifikation kann man häufig auch mittels einer `.htaccess`-Datei herbeiführen:

```
AuthType Basic
AuthName "Authorisation: User erforderlich"
AuthUserFile /etc/passwd
require valid-user
```

Sollten Sie manuell derartige Kommandos eingebaut haben, um das Frontend von Serendipity zu schützen, wird Serendipity diese automatisch auch zum Login in das Backend verwenden. Daher müssten die Benutzernamen, die Sie zum HTTP-Auth-Login verwenden, identisch sein mit den Zugangsdaten der Serendipity-Benutzerdatenbank. Sollte dies nicht der Fall sein, müssen Sie die Variable `$serendipity['useHTTP-Auth']` auf `false` setzen und damit die HTTP-Authentifikation im Backend unterbinden.

Einen Benutzernamen und ein Passwort, das Sie zum Login einsetzen wollen, können Sie auch notfalls an jede URL mit den Parametern `http_auth_user` und `http_auth_pw` anhängen:

```
http://www.example.com/serendipity/index.php?http_auth_user=John+D
oe&http_auth_pw=john
```

Bei einem derartigen Aufruf müssen Sie darauf achten, dass Ihr Benutzername und Passwort im Klartext an den Webserver übermittelt werden und dabei möglicherweise in Zugriffs-Logfiles auftauchen könnten. Nutzen Sie daher diese Art des Logins nur in vertrauenswürdigen Umfeldern und wenn die obige Variante mittels Apache-PHP-Modul nicht möglich ist.

Die sicherste Methode zum Login bleibt jedoch nach wie vor der Zugriff über das `HTTPS`-Protokoll.

`$serendipity['use_PEAR']`

PEAR ist eine Sammlung von PHP-Bibliotheken (siehe Kapitel 1.3.3 auf Seite 36), die Serendipity für einige Funktionalitäten benötigt. Serendipity liefert diese Bibliotheken zwar mit, aber da sie oft auf Servern vorhanden sind und besser gewartet werden, bevorzugt Serendipity standardmäßig die Server-Bibliotheken.

Dies kann jedoch möglicherweise zu Problemen führen, wenn Ihr Server ganz alte PEAR-Bibliotheken anbietet oder es zu Zugriffsrechtsproblemen kommt. In so einem Fall können Sie die Variable `$serendipity['use_PEAR']` auf `false` setzen und damit erzwingen, dass Serendipity seine eigenen Bibliotheken benutzt.

`$serendipity['CacheControl']`

Falls diese Variable auf `false` gesetzt ist, wird Serendipity spezielle HTTP-Header senden, die es dem Browser ermöglichen, die Seiten des Blogs lokal zwischenzuspeichern. Dies beschleunigt den Aufruf von bereits besuchten Blog-Seiten ungemein.

Wenn Sie jedoch ein Blog führen, in dem es häufig Änderungen im Minutentakt gibt, kann das Zwischenspeichern dazu führen, dass Ihre Besucher ältere Inhalte sehen und nicht merken, dass es neue Daten gibt.

Standardmäßig ist die Variable auf `true` gesetzt und verbietet daher dieses Caching zugunsten stets aktueller Inhalte.

`$serendipity['version']`
: Im Gegensatz zu `$serendipity['versionInstalled']` gibt diese Variable nicht an, welche Version gerade *aktiviert* ist, sondern welcher Version die Dateien auf dem Server entsprechen. Dieser minimale Unterschied ist dann wichtig, wenn Sie gerade eine neue Serendipity-Version hochgeladen haben, denn Serendipity wird aufgrund eines Unterschieds in den Versionsnummern der beiden genannten Variablen erkennen, ob und wenn ja, welches Update ausgeführt werden soll. Ändern Sie diese Versionsnummer daher nicht eigenständig.

`$serendipity['production']`
: Wenn diese Variable auf `true` gesetzt ist, befindet sich Serendipity im *produktiven Einsatz*. In diesem Fall werden dann einige Fehlermeldungen *nicht* ausgegeben, die Serendipity in einer Testumgebung ansonsten darstellen würde.

  Um Fehler leichter zu bemerken, ist diese Variable bei allen Beta-Versionen und Snapshots von Serendipity standardmäßig deaktiviert (`false`), damit Fehlermeldungen ausgegeben werden. In allen finalen Versionen ist sie standardmäßig aktiviert.

`$serendipity['allowDateManipulation']`
: Standardmäßig ist diese Variable auf `true` gesetzt und ermöglicht es dadurch einem Redakteur, eine beliebige Uhrzeit für die Veröffentlichung seines Artikels einzutragen. Wenn Sie diese Variable auf `false` setzen, wird immer nur die aktuelle Uhrzeit für einen Artikel eingetragen, und Redakteuren (auch Administratoren!) ist es nicht erlaubt, das Datum zu verändern. Diese Option ist daher bei solchen Blogs sinnvoll, die streng auf *ehrliche* Zeitangaben setzen.

`$serendipity['max_last_modified']`

`$serendipity['max_fetch_limit']` Diese beiden Variablen beziehen sich auf das Caching der RSS-Feeds (siehe Seite 87).

: Wenn ein Artikel von Ihnen überarbeitet wird oder einen neuen Kommentar erhält, ändert dies nichts an dem Veröffentlichungsdatum Ihres Blog-Artikels. Ein RSS-Feed würde daher Änderungen an einem Artikel nicht an die Benutzer weiterreichen. Da so ein Verhalten unerwünscht ist, sorgt Serendipity bei jeder Aktualisierung (auch bei Kommentaren) eines Artikels dafür, dass ein Aktualisierungsdatum auf den aktuellen Zeitpunkt gesetzt wird. Der RSS-Feed richtet sich nach diesem Aktualisierungsdatum und zeigt trotz aktiviertem RSS-Caching daraufhin die neuen Einträge an.

  Da RSS-Reader Einträge eindeutig anhand einer ID oder ihrer URL identifizieren, kann das Programm den aktualisierten Artikel erneut einlesen und Sie über Änderungen (und neue Kommentare) informieren.

  Die Variable `$serendipity['max_last_modified']` legt nur fest, wie alt ein Artikel maximal (relativ zum aktuellen Zeitpunkt) sein darf, damit Änderungen am Artikel in RSS-Feeds als *neu* angesehen werden. Schließlich interessiert es einen Leser Ihres RSS-Feeds sicher nicht, wenn Sie einen zwei Jahre alten Artikel überarbeiten oder jemand den ersten Eintrag Ihres Blogs kommentiert. Diese Variable enthält eine Angabe in Sekunden, die der Abstand zwischen Veröffentlichungsdatum und aktuellem Zeitpunkt maximal betragen darf. Ist ein Artikel älter als diese Zeitangabe, wird er nicht als *aktualisiert* markiert. Standardmäßig ist `$serendipity['max_last_modified']` auf 604800 Sekunden[24] gesetzt.

[24] Das entspricht sieben Tagen. Für Programme ist es einfacher, einheitlich mit Sekunden zu arbeiten, auch wenn die Angabe fürs menschliche Auge etwas merkwürdig wirken mag.

Die zweite Variable `$serendipity['max_fetch_limit']` legt fest, wie viele Artikel ein Benutzer, der Ihren RSS-Feed seit langem nicht mehr aufgerufen hat, maximal empfangen darf. Standardmäßig ist der Wert auf `50` Artikel gesetzt, um nicht zu viel Bandbreite zu verschwenden und potenziellen Missbrauch einzudämmen.

`$serendipity['trackback_filelimit']`

Wenn Serendipity ein Trackback zu einem Artikel schicken will, muss es die URL zu dem fremden Artikel öffnen. Dies macht Serendipity mit allen URLs, auf die Sie in einem Artikel verweisen, um automatisch herauszufinden, zu welchen URLs ein Trackback gesendet werden soll. Wenn Sie jedoch auf große Dateien (Videos, Audio-Dateien) verweisen, muss Serendipity auch diese URL vollständig abrufen. Das kann dann dazu führen, dass Serendipity sehr lange braucht, um Trackbacks auszuwerten, und währenddessen auch viel Bandbreite/Traffic aufbraucht.

Sollten Sie Trackbacks aber nicht pauschal deaktivieren wollen (siehe Seite 175), können Sie die maximale Dateigröße (in Bytes) in der Variable `$serendipity['trackback_filelimit']` eintragen. Das Standardlimit steht derzeit auf `150`kb. Bitte beachten Sie, dass Serendipity nicht immer von vornherein beim Besuch einer URL weiß, wie groß der Inhalt sein wird. Daher kann es auch passieren, dass URLs mit größeren Inhalten zwangsweise aufgerufen werden. In diesem Fall hilft das Plugin `Trackbacks kontrollieren` (siehe Seite 283), mit dem Sie Trackbacks gezielt auf spezielle URLs einschränken können.

`$serendipity['use_iframe']`

Wenn Serendipity einen Artikel speichert, führt es diese Aktionen in einem separaten Bereich der Webseite aus, den Serendipity oberhalb der Artikelmaske einbettet. Ein derart eingebetteter Bereich nennt sich *iframe* und kann von allen halbwegs modernen Browsern angezeigt werden.

Bei Browsern auf Handys und PDAs könnte diese Technik jedoch möglicherweise Probleme machen, daher können Sie die Variable `$serendipity['use_iframe']` auf `false` setzen, um Serendipity ohne diese Technik nutzen zu können.

Wenn iframes deaktiviert sind, können mögliche Fehler beim Speichern eines Artikels (hauptsächlich Trackback- und Plugin-Fehler) jedoch eher dazu führen, dass der gesamte Artikel nicht gespeichert wird. Daher sollten Sie iframes nur im größten Notfall deaktivieren und vorzugsweise lieber auf einen anderen Browser zurückgreifen. Beachten Sie auch, dass das Deaktivieren der iframes global für alle Redakteure gilt.

`$serendipity['languages']`

In dieser Variable speichert Serendipity die Liste aller für Besucher und Redakteure verfügbaren Sprachen. Die dort aufgeführten Kürzel müssen eine entsprechende Sprachdatei `lang/serendipity_lang_XX.inc.php` aufweisen, damit Sie die Sprache auch wirklich benutzen können. Wenn Sie also eine neue Übersetzung von Serendipity (z. B. für Plattdeutsch) erstellen wollen, können Sie die neue Sprache zum einen in diese Variable miteintragen und zum anderen die entsprechende Sprachdatei im `lang`-Unterverzeichnis erstellen.

Wenn Sie die Liste der verfügbaren Sprachen für Redakteure einschränken wollen, können Sie die überflüssigen Sprachen aus der Variablenliste entfernen.

`$serendipity['autolang']`
: Die Sprache, die Serendipity standardmäßig bei der Installation benutzt (wenn keine andere Sprache vom Browser präferiert wurde), wird in der Variable `$serendipity['autolang']` gespeichert. Das dort eingetragene Kürzel (standardmäßig `en` für `englisch`) muss in der Liste der Variable `$serendipity['languages']` enthalten sein.

`$serendipity['defaultTemplate']`
: Templates müssen nur die Dateien in ihrem eigenen Verzeichnis mitliefern, die von den Standarddateien abweichen. Dadurch können Templates klein gehalten werden und ihre Features vom Standard-Template abhängig machen.

  Wenn eine Template-Datei eines Themes nicht vorhanden ist, wird die Datei im Standard-Theme benutzt. Eben jenes Standard-Theme definieren Sie in der Variable `$serendipity['defaultTemplate']` (standardmäßig `carl_contest`, *Serendipity 3.0*).

  Weiterhin gilt als letzter Ausweg immer das Verzeichnis `default` im Template-Verzeichnis.

`$serendipity['skip_smarty_hooks']`
: Wenn der Wert dieser Variable auf `true` gesetzt wird, können Smarty-Templates selbständig keinerlei Ereignis-Plugins aufrufen (siehe auch Seite 563).

`$serendipity['skip_smarty_hook']`
: Statt den Aufruf aller Ereignis-Plugins mittels `$serendipity['skip_smarty_hooks']` zu verbieten, kann dieses Array eine Liste von Ereignissen (`Hooks`) enthalten, die ein Smarty-Funktionsaufruf innerhalb der Template-Dateien *nicht* aufrufen darf. Jeder Array-Schlüssel muss hierbei dem Namen des zu verbietenden Hooks enthalten.

Achten Sie darauf, dass, wenn Sie Änderungen in der Datei `serendipity_config.inc.php` vornehmen, diese Änderungen bei einem Update von Serendipity überschrieben werden und Sie sie neu eintragen müssen. Einzig Änderungen in der Datei `serendipity_config_local.inc.php` (sofern sie am Ende der Datei vorgenommen wurden) behält Serendipity bei einem Update bei.

4.7.2 Benutzerverwaltung

In der Benutzerverwaltung Serendipitys können Sie die Redakteure und ihre Zugriffsmöglichkeiten verwalten.

Auf dieser Übersichtsseite sehen Sie eine Liste aller Redakteure, auf die Sie Zugriff haben.

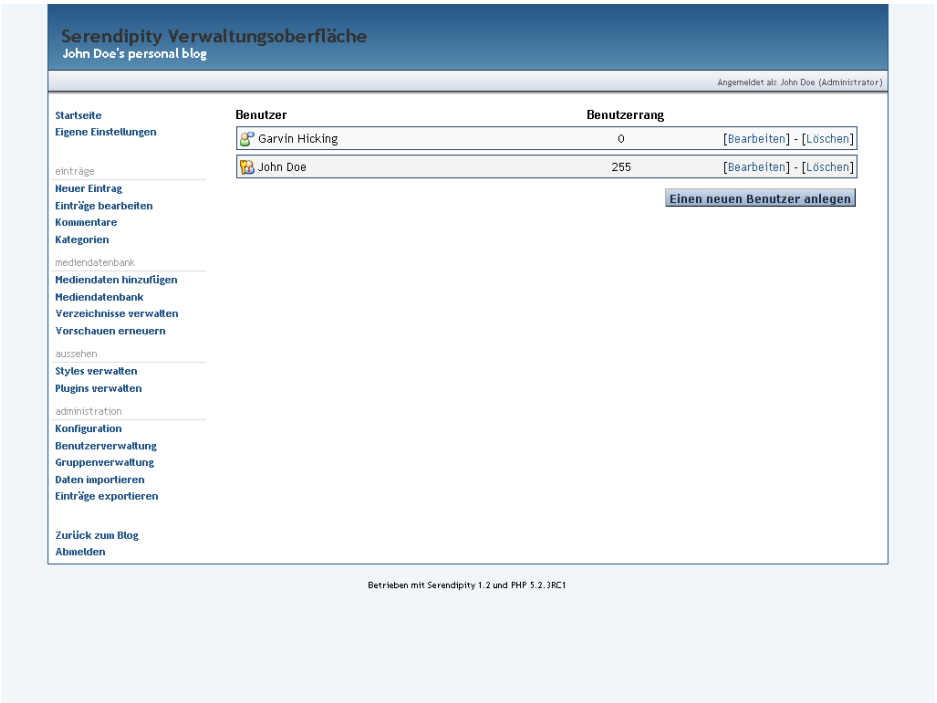

Abbildung 4.24: Administration: Benutzerverwaltung

In Serendipity werden Zugriffsrechte aufgrund von zwei Eigenschaften geregelt. Die eine ist der globale *Benutzerrang/Userlevel*, der der Zahl 0 (einfacher Redakteur), 1 (Chefredakteur) oder 255 (Administrator) entspricht. Die zweite ist die Zugehörigkeit zu einer Benutzergruppe, wobei jede Benutzergruppe flexible Rechte besitzen kann.

Die festen Benutzerlevel wurden in Serendipity 0.6 eingeführt und mittlerweile in fast allen Bereichen durch die Eingliederung in eigenständige *Benutzergruppen* abgelöst, die den Zugriff viel kleinstufiger regeln können. Dennoch gilt der Benutzerrang nach wie vor als ein Kriterium für einige zusätzliche Plugins. Sollte ein Redakteur keiner Gruppe zugehörig sein, regelt sein Benutzerrang alle ihm zur Verfügung stehenden Zugriffsmöglichkeiten.

Grundsätzlich empfiehlt es sich daher weiterhin, neue Redakteure auch einem Benutzerrang grob zuzuordnen. Bestimmen Sie dabei, ob ein Redakteur volle Rechte zu einem Blog haben soll (Administrator), ob er als Chefredakteur über anderen Redakteuren stehen soll oder ob er nur einen ganz schlichten Redakteur zur Texterfassung darstellen soll.

Ein Administrator sieht in der Übersicht alle Redakteure. Ein Chefredakteur sieht hier nur noch andere Chefredakteure und normale Redakteure. Normale Redakteure wiederum sehen nur andere normale Redakteure.

In der Benutzerübersicht wird dem Benutzerrang entsprechend ein kleines Symbol dem Namen vorangestellt. Es folgen der Benutzerrang und zwei Links zum **Bearbeiten** und **Löschen** eines Benutzers.

Über den Button **Einen neuen Benutzer anlegen** können Sie einen neuen Benutzer anlegen, die Oberfläche ähnelt dabei den Optionen bei der Bearbeitung eines Redakteurs. Beachten Sie beim Bearbeiten eines Redakteurs, dass Ihre Möglichkeiten zum Ändern der Daten von Ihren Rechten abhängig sind, daher wird Ihnen unter Umständen beim Speichern eines Benutzers das Recht verwehrt, diese Änderung durchzuführen.

Einen neuen Benutzer anlegen oder einen Benutzer bearbeiten

Die Oberfläche zum Bearbeiten/Anlegen eines Redakteurs sieht größtenteils exakt so aus wie die Seite der **Eigenen Einstellungen** (siehe Kapitel 2 ab Seite 96). Hier können Sie einen Benutzer so bearbeiten, als wären es Ihre eigenen Einstellungen. Tun Sie dies bitte nur mit Vorsicht, denn üblicherweise sollten die Redakteure ihre Einstellungen selbständig vornehmen.

Das **Alte Passwort** eines Benutzers muss an dieser Stelle nur eingegeben werden, wenn der Benutzer Ihrem eigenen Redakteurs-Account entspricht. Andernfalls kann ein Administrator bzw. befugter Benutzer Änderungen auch ohne Passwortangabe vornehmen und so auch ein neues Passwort setzen.

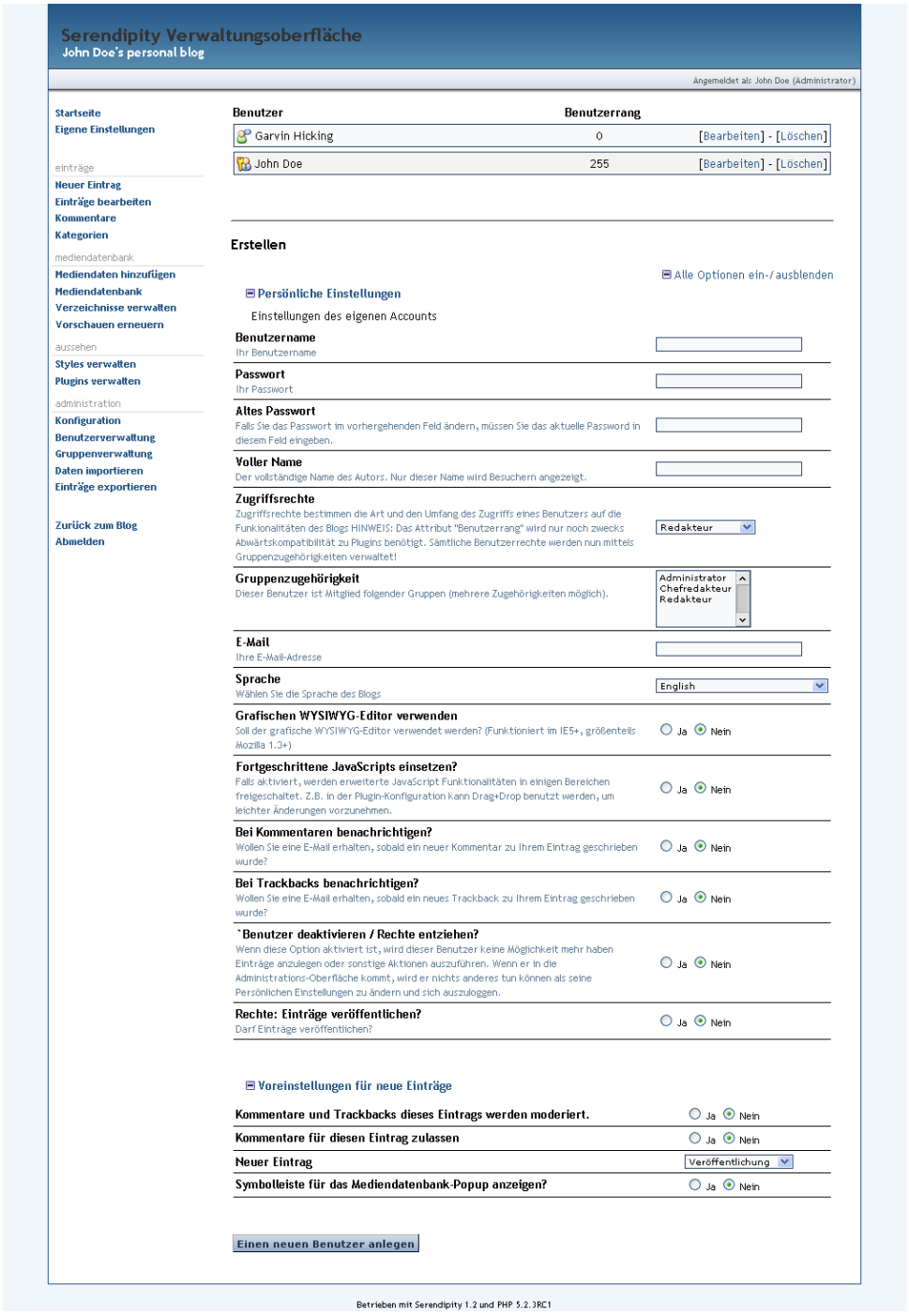
Serendipity Verwaltungsoberfläche
John Doe's personal blog
Angemeldet als John Doe (Administrator)
Startseite
Eigene Einstellungen
einträge
Neuer Eintrag
Einträge bearbeiten
Kommentare
Kategorien
mediendatenbank
Mediendaten hinzufügen
Mediendatenbank
Verzeichnisse verwalten
Vorschauen erneuern
aussehen
Styles verwalten
Plugins verwalten
administration
Konfiguration
Benutzerverwaltung
Gruppenverwaltung
Daten importieren
Einträge exportieren
Zurück zum Blog
Abmelden
Benutzer
Benutzerrang
Garvin Hicking
0
[Bearbeiten] - [Löschen]
John Doe
255
[Bearbeiten] - [Löschen]
Erstellen
Alle Optionen ein-/ausblenden
Persönliche Einstellungen
Einstellungen des eigenen Accounts
Benutzername
Ihr Benutzername
Passwort
Ihr Passwort
Altes Passwort
Falls Sie das Passwort im vorhergehenden Feld ändern, müssen Sie das aktuelle Password in diesem Feld eingeben.
Voller Name
Der vollständige Name des Autors. Nur dieser Name wird Besuchern angezeigt.
Zugriffsrechte
Zugriffsrechte bestimmen die Art und den Umfang des Zugriffs eines Benutzers auf die Funktionalitäten des Blogs HINWEIS: Das Attribut "Benutzerrang" wird nur noch zwecks Abwärtskompatibilität zu Plugins benötigt. Sämtliche Benutzerrechte werden nun mittels Gruppenzugehörigkeiten verwaltet!
Redakteur
Gruppenzugehörigkeit
Dieser Benutzer ist Mitglied folgender Gruppen (mehrere Zugehörigkeiten möglich).
Administrator
Chefredakteur
Redakteur
E-Mail
Ihre E-Mail-Adresse
Sprache
Wählen Sie die Sprache des Blogs
English
Grafischen WYSIWYG-Editor verwenden
Soll der grafische WYSIWYG-Editor verwendet werden? (Funktioniert im IE5+, größtenteils Mozilla 1.3+)
Ja Nein
Fortgeschrittene JavaScripts einsetzen?
Falls aktiviert, werden erweiterte JavaScript Funktionalitäten in einigen Bereichen freigeschaltet. Z.B. in der Plugin-Konfiguration kann Drag+Drop benutzt werden, um leichter Änderungen vorzunehmen.
Ja Nein
Bei Kommentaren benachrichtigen?
Wollen Sie eine E-Mail erhalten, sobald ein neuer Kommentar zu Ihrem Eintrag geschrieben wurde?
Ja Nein
Bei Trackbacks benachrichtigen?
Wollen Sie eine E-Mail erhalten, sobald ein neues Trackback zu Ihrem Eintrag geschrieben wurde?
Ja Nein
`Benutzer deaktivieren / Rechte entziehen?
Wenn diese Option aktiviert ist, wird dieser Benutzer keine Möglichkeit mehr haben Einträge anzulegen oder sonstige Aktionen auszuführen. Wenn er in die Administrations-Oberfläche kommt, wird er nichts anderes tun können als seine Persönlichen Einstellungen zu ändern und sich auszuloggen.
Ja Nein
Rechte: Einträge veröffentlichen?
Darf Einträge veröffentlichen?
Ja Nein
Voreinstellungen für neue Einträge
Kommentare und Trackbacks dieses Eintrags werden moderiert.
Ja Nein
Kommentare für diesen Eintrag zulassen
Ja Nein
Neuer Eintrag
Veröffentlichung
Symbolleiste für das Mediendatenbank-Popup anzeigen?
Ja Nein
Einen neuen Benutzer anlegen
Betrieben mit Serendipity 1.2 und PHP 5.2.3RC1

Abbildung 4.25: Administration: Benutzerverwaltung: Neuer Redakteur

Abweichend von den **Eigenen Einstellungen** sind in dieser Oberfläche zusätzlich die folgenden Optionen aufgeführt:

Zugriffsrechte
: Die **Zugriffsrechte** entsprechen der groben Einordnung eines Benutzers in einen Benutzerrang. Wählen Sie aus dem Auswahlfeld, welchem Rang der Benutzer am ehesten entspricht.

  Achten Sie darauf, dass Sie, wenn Sie Ihren eigenen Benutzer bearbeiten, Ihre eigenen Rechte beschränken könnten. Wenn Sie versehentlich Ihren Benutzerrang auf *Redakteur* stellen, sind Ihre Rechte erstmal eingeschränkt. Um in diesem Fall Ihren Benutzerrang zu reaktivieren, müssen Sie das Script `fixpriv.php` auf Seite 100 ausführen.

Gruppenzugehörigkeit
: Die Gruppenzugehörigkeit des gewählten Redakteurs wird über das Mehrfachauswahlfeld **Gruppenzugehörigkeit** gesteuert. Ein Redakteur kann dabei Mitglied in mehreren Gruppen sein, um alle Rechte jeder einzelnen Gruppe zu vereinen.

  Achten Sie auch hier darauf, dass, wenn Sie sich (versehentlich) als Mitglied der Gruppe entfernen, Sie möglicherweise Ihren Zugriff auf Serendipity zerstören.

Benutzer deaktivieren/Rechte entziehen
: Eine globale Option, um einem Benutzer sämtliche Zugriffsrechte außer dem Einloggen zu verbieten, stellt der Punkt **Benutzer deaktivieren** dar. Ist der Wert **Ja** aktiviert, ist ein Benutzer sozusagen gesperrt.

  Dies ist besonders nützlich, wenn Sie Benutzer anlegen wollen, die eigentlich nur im Frontend Artikel lesen, aber im Backend keinerlei Möglichkeiten nutzen sollen. Über das Plugin **Freie Benutzer-Registrierung** können Sie Besuchern ermöglichen, sich selbst einen Redakteurs-Account anzulegen (Details siehe Seite 395) – auch bei solchen Redakteuren wird es oft gewünscht sein, den Zugriff stark einzuschränken.

  Achten Sie darauf, nicht versehentlich für sich selbst diese Option zu aktivieren. Auch dies können Sie nur mittels des `fixpriv.php`-Scripts rückgängig machen.

Rechte: Einträge veröffentlichen
: Die Option **Rechte: Einträge veröffentlichen** bestimmt, ob es einem Redakteur erlaubt ist, einen Artikel als *Veröffentlichung* zu speichern. Ist diese Option nicht aktiviert, kann ein Benutzer lediglich *Entwürfe* speichern. Diese Entwürfe müssen dann von einem höherrangigen Redakteur veröffentlicht werden.

Beim Erstellen eines neuen Benutzers achten Sie bitte auf die Voreinstellungen, da z. B. die Option **Rechte: Einträge veröffentlichen** standardmäßig auf **Nein** steht. Möglicherweise entspricht dies nicht der Einstellung, die Sie beabsichtigen.

4.7.3 Gruppenverwaltung

Ähnlich wie bei der Verwaltung der Benutzer sehen Sie in der Übersicht der **Gruppenverwaltung** alle angelegten Benutzergruppen.

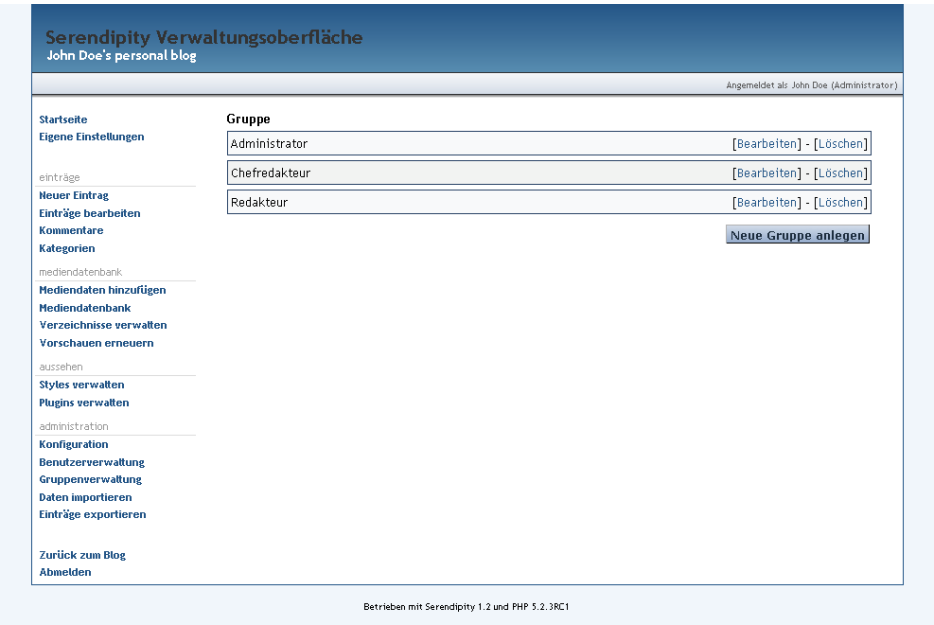

Abbildung 4.26: Administration: Gruppenverwaltung

Sie können von dort aus entweder eine **Neue Gruppe anlegen**, bestehende Gruppen **Bearbeiten** oder **Löschen**. Beim Löschen einer Gruppe müssen Sie vorsichtig sein, dass Sie damit bestehenden Redakteuren nicht den Zugriff auf das Backend vollständig entziehen.

Standardmäßig werden in Serendipity drei Benutzergruppen angelegt: **Administrator**, **Chefredakteur** und **Redakteur**. Jede dieser Gruppen ist so eingerichtet, dass sie den Zugriff für die Mitglieder dieser Gruppen so einschränkt, wie es zu erwarten wäre: Redakteure dürfen nur Artikel anlegen und eigene Artikel bearbeiten, Chefredakteure dürfen zusätzlich Plugins und das Aussehen des Blogs verwalten und Einträge anderer Redakteure überarbeiten. Administratoren dürfen natürlich alles.

Jede Gruppe kann eine große Menge an kleinstufigen Rechten festlegen, die Sie in der Detailseite einer neuen oder bestehenden Gruppe ankreuzen können.

Eine Gruppe erstellen oder bearbeiten

Die Detailseite einer Gruppe listet alle Rechte auf. Sie können dabei nur diejenigen Rechte mittels einer Auswahlbox auswählen, die Sie selber aufgrund Ihrer Gruppenmitgliedschaft

besitzen.

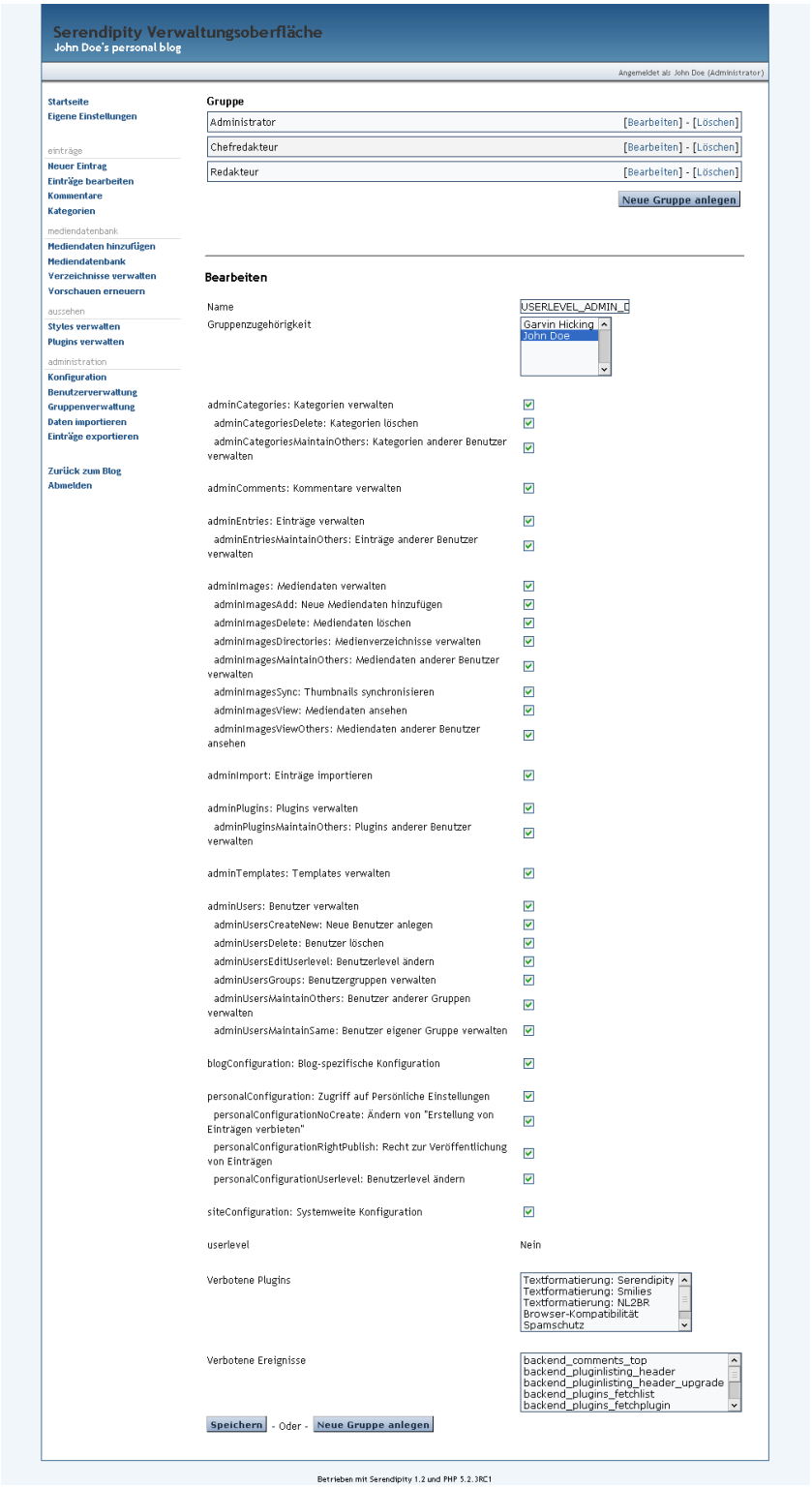
Serendipity Verwaltungsoberfläche
John Doe's personal blog
Angemeldet als John Doe (Administrator)
Startseite
Eigene Einstellungen
einträge
Neuer Eintrag
Einträge bearbeiten
Kommentare
Kategorien
mediendatenbank
Mediendaten hinzufügen
Mediendatenbank
Verzeichnisse verwalten
Vorschauen erneuern
aussehen
Styles verwalten
Plugins verwalten
administration
Konfiguration
Benutzerverwaltung
Gruppenverwaltung
Daten importieren
Einträge exportieren
Zurück zum Blog
Abmelden
Gruppe
Administrator [Bearbeiten] - [Löschen]
Chefredakteur [Bearbeiten] - [Löschen]
Redakteur [Bearbeiten] - [Löschen]
Neue Gruppe anlegen
Bearbeiten
Name
USERLEVEL_ADMIN_D
Gruppenzugehörigkeit
Garvin Hicking
John Doe
adminCategories: Kategorien verwalten
adminCategoriesDelete: Kategorien löschen
adminCategoriesMaintainOthers: Kategorien anderer Benutzer verwalten
adminComments: Kommentare verwalten
adminEntries: Einträge verwalten
adminEntriesMaintainOthers: Einträge anderer Benutzer verwalten
adminImages: Mediendaten verwalten
adminImagesAdd: Neue Mediendaten hinzufügen
adminImagesDelete: Mediendaten löschen
adminImagesDirectories: Medienverzeichnisse verwalten
adminImagesMaintainOthers: Mediendaten anderer Benutzer verwalten
adminImagesSync: Thumbnails synchronisieren
adminImagesView: Mediendaten ansehen
adminImagesViewOthers: Mediendaten anderer Benutzer ansehen
adminImport: Einträge importieren
adminPlugins: Plugins verwalten
adminPluginsMaintainOthers: Plugins anderer Benutzer verwalten
adminTemplates: Templates verwalten
adminUsers: Benutzer verwalten
adminUsersCreateNew: Neue Benutzer anlegen
adminUsersDelete: Benutzer löschen
adminUsersEditUserlevel: Benutzerlevel ändern
adminUsersGroups: Benutzergruppen verwalten
adminUsersMaintainOthers: Benutzer anderer Gruppen verwalten
adminUsersMaintainSame: Benutzer eigener Gruppe verwalten
blogConfiguration: Blog-spezifische Konfiguration
personalConfiguration: Zugriff auf Persönliche Einstellungen
personalConfigurationNoCreate: Ändern von "Erstellung von Einträgen verbieten"
personalConfigurationRightPublish: Recht zur Veröffentlichung von Einträgen
personalConfigurationUserlevel: Benutzerlevel ändern
siteConfiguration: Systemweite Konfiguration
userlevel
Nein
Verbotene Plugins
Textformatierung: Serendipity
Textformatierung: Smilies
Textformatierung: NL2BR
Browser-Kompatibilität
Spamschutz
Verbotene Ereignisse
backend_comments_top
backend_pluginlisting_header
backend_pluginlisting_header_upgrade
backend_plugins_fetchlist
backend_plugins_fetchplugin
Speichern - Oder - Neue Gruppe anlegen
Betrieben mit Serendipity 1.2 und PHP 5.2.3RC1

Abbildung 4.27: Administration: Gruppenverwaltung: Gruppe bearbeiten

Wenn Sie als Mitglied der Gruppe **Chefredakteur** eine neue Gruppe erstellen, können Sie keine Rechte zur Administration des Blogs vergeben – das wäre natürlich eine einfache Art, sich die Blog-Herrschaft anzueignen.

Bei solchen nicht vergebbaren Rechten sehen Sie also nur den aktuellen Wert des jeweiligen „gesperrten“ Rechtes.

Folgende Rechte stehen zur Verfügung:

Name
: Der Name einer Gruppe kann von Ihnen frei gewählt werden und darf auch Sonderzeichen enthalten. Bei den von Serendipity vordefinierten Gruppen steht an dieser Stelle nicht der jeweilige Gruppenname, sondern etwas in der Art `USERLEVEL_CHIEF_DESC`. Dies wird später bei der Darstellung durch den jeweiligen „echten“ Namen ersetzt, damit bei anderssprachigen Redakteuren deren Gruppenzugehörigkeit nicht auf einen deutschen Begriff festgelegt wird.

Gruppenzugehörigkeit
: In dieser Auswahlbox werden alle verfügbaren Redakteure dargestellt. So können Sie einer neuen Gruppe relativ leicht bestehende Redakteure hinzufügen, indem Sie sie einfach in dem Mehrfachauswahlfeld markieren.

  Einen Benutzer können Sie auf zwei Arten aus einer Gruppe entfernen: Entweder Sie heben die Zugehörigkeit in diesem Mehrfachauswahlfeld auf, oder Sie bearbeiten den jeweiligen Benutzer und heben dort die Gruppenzuordnung auf. Mehrere Redakteurszuordnungen können Sie mit gedrückter *(Strg/Apfel)*-Taste und der linken Maustaste vornehmen.

adminCategories, adminCategoriesDelete, adminCategoriesMaintainOthers
: Wenn das Recht **adminCategories** aktiviert ist, kann ein Redakteur Kategorien verwalten (neue Kategorien anlegen und bestehende Kategorien bearbeiten). Besitzt er das Recht **adminCategoriesDelete**, darf er auch Kategorien löschen.

  Dabei ist der Zugriff nur auf selbst angelegte Kategorien möglich. Erst wenn ein Benutzer auch das Recht **adminCategoriesMaintainOthers** besitzt, darf er auch Kategorien anderer Benutzer verwalten.

adminComments
: Damit ein Redakteur die Kommentarübersicht aufrufen kann, muss er das Recht **adminComments** besitzen. Ohne dieses Recht darf er nur Kommentare zu von ihm geschriebenen Artikeln (mittels der E-Mail-Benachrichtigung) freischalten.

adminEntries, adminEntriesMaintainOthers
: Nur wenn ein Redakteur das Recht **adminEntries** besitzt, darf er Einträge im Blog erstellen. Dabei darf er nur selbst erstellte Artikel im Nachhinein überarbeiten. Wenn

er das Recht **adminEntriesMaintainOthers** besitzt, darf er zusätzlich auch auf Artikel anderer Redakteure zugreifen.

Ob ein Benutzer überhaupt Einträge erstellen und veröffentlichen darf, wird zusätzlich individuell pro Benutzer in dessen Einstellungen (siehe Seite 182) festgelegt.

adminImages und weitere
: Um auf die Mediendatenbank zuzugreifen, benötigt ein Redakteur das Recht **adminImages**. Weitere feinstufige Rechte kontrollieren, was der Redakteur innerhalb der Mediendatenbank durchführen darf.

  Mittels **adminImagesAdd** darf er neue Mediendateien hochladen. Das Recht **adminImageDelete** ermöglicht es ihm, Mediendateien auch wieder zu löschen. Neue Unterverzeichnisse kann er mit dem Recht **adminImagesDirectories** anlegen. Um Vorschaubilder erneut zu erstellen, benötigt er das Recht **adminImagesSync**.

  Die Mediendatenbank selbst kann ein Redakteur zum Einbinden von Dateien nur aufrufen, wenn er das Recht **adminImagesView** besitzt. Um auch die Dateien anderer Redakteure anzusehen, benötigt er das Recht **adminImagesViewOthers**.

  Um auch die Mediendaten anderer Redakteure bearbeiten/löschen zu können, ist das Recht **adminImagesMaintainOthers** erforderlich.

adminImport
: Das Recht **adminImport** regelt, ob ein Redakteur die Einträge fremder Blogsysteme importieren darf.

adminPlugins
: Besitzt ein Redakteur das Recht **adminPlugins**, kann er Seitenleisten- und Ereignis-Plugins installieren. Dabei wird der Redakteur als Eigentümer eines Plugins aufgeführt und kann daraufhin nur seine eigenen Plugins auch konfigurieren. Über das Recht **adminPluginsMaintainOthers** kann ein Redakteur auch die Konfiguration fremder Plugins ändern.

adminTemplates
: Um ein anderes Template zu aktivieren und Template-Optionen zu konfigurieren, benötigt der Redakteur das Recht **adminTemplates**.

adminUsers und weitere
: Ob ein Redakteur die Benutzerverwaltung aufrufen darf, wird mit dem Recht **adminUsers** eingestellt. Neue Benutzer darf er mit dem Recht **adminUsersCreateNew** anlegen, Benutzer löschen mit **adminUsersDelete**.

  Den Benutzerrang darf er nur verändern, wenn er das Recht **adminUsersEditUserlevel** besitzt. Ansonsten darf er nur Redakteure niederen Ranges anpassen.

  Die Gruppenverwaltung darf der Benutzer mit dem Recht **adminUsersGroups** anlegen. Besitzt er das Recht **adminUsersMaintainSame**, darf er gleichrangige Benutzer bearbeiten, und wenn er das Recht **adminUsersMaintainOthers** besitzt, darf er auch alle anderen Benutzer verwalten.

blogConfiguration

Wenn der Redakteur das Recht **blogConfiguration** besitzt, darf er Änderungen an der globalen Konfiguration des Blogs vornehmen. Darunter fallen jedoch keine Änderungen der Datenbankkonfiguration oder systemnaher Einstellungen.

personalConfiguration, personalConfigurationNoCreate

Die **Eigenen Einstellungen** darf ein Redakteur nur mit dem Recht **personalConfiguration** aufrufen.

Das Sonderrecht, einen Benutzer zu sperren, darf ein Redakteur nur ausüben, wenn er das Recht **personalConfigurationNoCreate** besitzt. Analog dazu darf er die Sondervariablen zum Veröffentlichungsrecht von Artikeln und den Benutzerrang nur bearbeiten, wenn er über die Rechte **personalConfigurationRightPublish** und **personalConfigurationUserlevel** verfügt.

siteConfiguration Die systemweite Konfiguration (mit Einstellungen der Datenbank und anderer systemnaher Optionen) darf ein Redakteur nur verändern, wenn er über das Recht **siteConfiguration** verfügt.

userlevel Die vordefinierten Gruppen von Serendipity können mit einem Benutzerrang verkettet werden. Dies ist nicht veränderbar und lediglich für die vordefinierten Gruppen von Interesse. Daher sehen Sie an dieser Stelle auch in allen Fällen nur ein **Nein**.

Verbotene Plugins Ist die globale Option **Sollen persönliche Plugin-Rechte für Benutzergruppen aktiviert werden?** in der **Konfiguration** des Blogs (Abschnitt **Generelle Einstellungen**) aktiviert, werden Sie an dieser Stelle eine Liste aller installierten Plugins sehen. Sollte die Option (wie standardmäßig der Fall) deaktiviert sein, sehen Sie hier nur einen Hinweis, der auf die Konfigurationsoption Bezug nimmt.

In dem Mehrfach-Auswahlfeld können Sie alle Plugins auswählen, auf die eine Benutzergrupe *keinen* Zugriff hat. So können Sie Redakteure von der Benutzung einiger Plugins gezielt ausnehmen.

Verbotene Ereignisse Die Ereignis-Plugins von Serendipity werden an speziellen Stellen des Systems ausgeführt. Diese Stellen nennen sich *Ereignisse* und tragen alle einen individuellen Namen. Plugins können sich zu einem *Ereignis* einklinken und dann beliebige Aktionen durchführen.

Wenn man ein Plugin nicht zentral verbieten will, kann über dieses Mehrfachfeld gezielt nur eine Menge von gewünschten Ereignissen für die Benutzergruppe blockiert werden.

Die Namen der Ereignisse lassen auf deren Bedeutung schließen, sind aber nur für erfahrenere Benutzer von Belang. Auch von Plugins selbst eingebundene Ereignisse sind in dieser Liste enthalten.

Eine Übersicht über die üblichen Serendipity-Ereignisse sehen Sie im Abschnitt 10.8.4 ab Seite 659.

Wenn Sie eine neue Gruppe anlegen wollen, sehen Sie am Ende der Eingabemaske den Button **Neue Gruppe anlegen**.

Sollten Sie eine bestehende Gruppe bearbeiten, können Sie die vorgenommenen Änderungen über den Menüpunkt **Speichern** übernehmen. Auch hier sehen Sie dann den Button **Neue Gruppe anlegen**. Dieser dient dazu, eine ganz neue Gruppe mit den konfigurierten Möglichkeiten zu erstellen. Sie können diese Methode also nutzen, wenn Sie eine Gruppe erstellen wollen, die einer bestehenden Gruppe ähnelt. Bearbeiten Sie dann diese Gruppe als Vorlage, nehmen Sie die Änderungen vor und klicken Sie auf **Neue Gruppe anlegen**, um dies durchzuführen.

4.7.4 Daten importieren

Serendipity kann von anderen Blogsystemen Daten importieren. Über den Menüpunkt **Daten importieren** finden Sie eine Auswahl des betreffenden Systems, von dem Sie importieren wollen.

Abhängig vom gewählten System können nur Teile der Daten dieses Systems importiert werden, und da es bei fremder Software öfter zu Änderungen der Datenbankstruktur kommt, kann die exakte Versionsnummer beim Import sehr wichtig sein.

Sollte Ihre Blogsoftware in der Liste nicht aufgeführt sein, ist ein Import zu Serendipity dennoch nicht unmöglich. Fragen Sie in diesem Fall im Serendipity-Forum unter `http://board.s9y.org/` einfach einmal nach, dort wird man sicher helfen können. Grundsätzlich müssen beim Import lediglich Daten auf Datenbankbasis eingelesen und in einem neuen Format gespeichert werden. Wenn Sie über SQL-Kenntnisse verfügen, könnten Sie Ihren eigenen Importer so relativ einfach entwickeln. Details zur Datenbankstruktur von Serendipity finden Sie ab Seite 623.

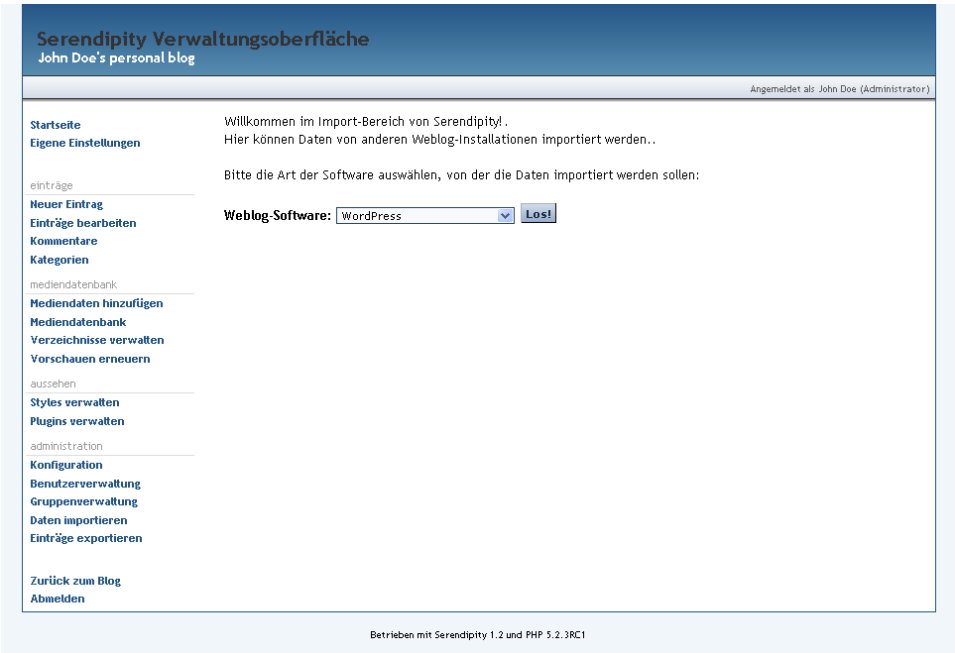

Abbildung 4.28: Administration: Daten importieren

Nachdem Sie das Quell-Blogsystem gewählt haben, klicken Sie auf **Los!**. Auf der Folgeseite werden Ihnen abhängig vom gewählten System einige Optionen präsentiert.

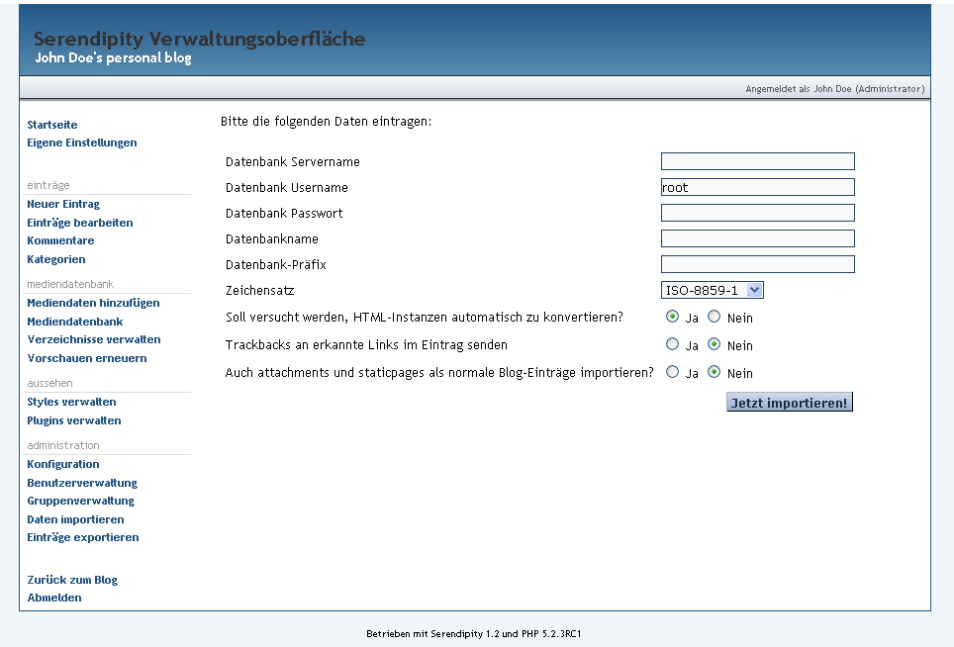

Abbildung 4.29: Administration: Daten importieren: WordPress

Meist ist es wichtig, dass die Datenbank des Zielsystems sich auf demselben Server wie das Serendipity-Blog befindet, um auf die Datensätze zugreifen zu können. Sie sollten auf jeden Fall mittels einer Software wie phpMyAdmin vor dem Import ein Backup sowohl der Serendipity-Datenbank als auch des Quell-Blogs machen.

Der Import-Vorgang in Serendipity kann beliebig oft durchgeführt werden und führt bei jedem neuen Aufruf zum erneuten Import. So kann es also passieren, dass bei mehreren Importversuchen die Artikel mehrfach importiert werden. Nach jedem fehlgeschlagenen Import sollten Sie daher die möglicherweise bereits erstellten Einträge, Kategorien und Autoren wieder löschen.

Folgende Blogsysteme stehen zum Import bereit:

WordPress

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel, Statische Seiten.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Die Option **Zeichensatz** gibt an, in welchem Zeichensatz WordPress seine Artikel gespeichert hat. Dies ist in neuen WordPress-Versionen meist UTF-8, und ISO-8859-1 in älteren.

In WordPress-Artikeln werden oft Sonderzeichen mit HTML-Syntax maskiert. Aus einem *Ä* wird so ein *Ä*. Serendipity bevorzugt jedoch die *echten* Sonderzeichen.

Wenn Sie die Option **Soll versucht werden, HTML-Instanzen automatisch zu konvertieren** aktivieren, werden derartige Sonderzeichen wieder korrekt umgewandelt.

Sollten Sie beim Import der Einträge später einmal merkwürdige Sonderzeichen erhalten, probieren Sie einmal diese beiden Optionen aus.

Mit der Option **Trackbacks an erkannte Links im Eintrag senden** können Sie erzwingen, dass alle Trackbacks zu den importierten Einträgen nochmals geschickt werden. Üblicherweise ist dies unerwünscht, daher ist die Voreinstellung auf **Nein** gesetzt.

WordPress kann neben Artikeln auch sogenannte **attachments** und **staticpages** speichern. Wenn Sie die Option **Auch attachments und staticpages als normale Blog-Einträge importieren** aktivieren, werden diese Sonderseiten auch als Serendipity-Artikel importiert.

WordPress PostgreSQL

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Eine Abspaltung von WordPress stellt **WordPress PostgreSQL** dar. Dieses läuft mit einer PostgreSQL- statt MySQL-Datenbank und kann mit identischen Optionen (Datenbankeinstellungen, Sonderzeichen, Trackbacks) wie **WordPress** importiert werden.

b2Evolution 0.9.0.11 Paris

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

LifeType

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

bBlog 0.7.4

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Blogger

Importiert werden: Autoren, Kommentare, Artikel.

Konfigurationsoptionen: Blogger.com-Exportdatei, Passwort für neue Autoren, Zeichensatz, Einteilung erweiterter Artikel.

`blogger.com` stellt einen recht komplizierten Fall des Imports dar. Dieses System läuft nicht auf dem eigenen Webserver, sondern wird von Google bereitgestellt. Daher gibt es auch keinen direkten Datenbankzugriff zum System.

Der Blogger-Import besteht aus einer ausführlichen Beschreibung, wie man sein Blog bei Blogger.com konfigurieren muss, um eine Exportdatei herzustellen. Diese Datei kann dann Serendipity wiederum importieren und auslesen.

Leider sind die Anweisungen auf die alten Templates von Blogger.com bezogen und können bei der neuen Blogger-Oberfläche nicht ohne Umstellung angewendet werden.

Notfalls müssen Sie daher Ihre Einträge von Blogger mittels des **Allgemeinen RSS-Imports** importieren.

boastMachine 3.0

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Geeklog 1.3.11

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

LiveJournal

Importiert werden: Nur Artikel.

Konfigurationsoptionen: XML-Quelldatei, zugeordnete Kategorie, Artikelstatus.

Ähnlich wie Blogger.com ist LiveJournal ein Online-Dienst, der Ihnen keinen direkten Zugriff auf Ihre geschriebenen Artikel ermöglicht.

Jedoch ermöglicht es LiveJournal, eine XML-Datei der Einträge zu exportieren. Diese können Sie auf Ihren Webserver hochladen und dem Importer benennen.

Leider können über diese XML-Datei nur die Artikel importiert werden und keinerlei andere Informationen.

MovableType

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: MovableType-Datensätze, Zeichensatz, Debugging, Trackbacks.

MovableType speichert seine Daten (in älteren Versionen) in einem für Serendipity nicht zugänglichen Datencontainer. Von der MovableType-Administrationsoberfläche aus können Sie jedoch eine Exportdatei erzeugen. Diese Exportdatei können Sie dann auf den Serendipity-Server hochladen und den Pfad zu dieser Datei beim Importvorgang eintragen.

Die MoveableType-Datensätze liegen leider in einem relativ „chaotischen“ Format vor, und daher kann es leicht zu defekten Importen kommen. Daher gibt es eine Debugging-Option und die Möglichkeit, trotz defekter Datei lesbare Einträge zu importieren.

Nucleus

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Pivot

Importiert werden: Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Pivot-Datensätze.

Pivot speichert seine Dateien in einer zentralen PHP-Datei. Den Pfad zu dieser Datei müssen Sie beim Importvorgang angeben. Alle importierten Artikel werden dem aktuellen Serendipity-Benutzer zugeordnet.

pMachine Pro 2.4

Importiert werden: Autoren, Passwörter, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

sunlog 0.4.4

Importiert werden: Autoren, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Sunlog bietet zwar diverse Zugriffsrechte für seine Redakteure, diese sind jedoch inkompatibel zu Serendipity. Dabei werden beim Import nur die Stammdaten der Redakteure übernommen, und alle Redakteure werden als Administratoren übernommen. Da Sunlog seine Passwörter in einem für Serendipity nicht lesbaren Format speichert, müssen die Passwörter aller Autoren erneut vergeben werden. Standardmäßig wird ein Redakteur dabei mit dem Passwort *sunlog* angelegt.

Textpattern 1.0rc1

Importiert werden: Autoren, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Da Textpattern seine Passwörter in einem für Serendipity nicht lesbaren Format speichert, müssen die Passwörter aller Autoren erneut vergeben werden. Standardmäßig wird ein Redakteur dabei mit dem Passwort *txp* angelegt.

phpNuke

Importiert werden: Autoren, Kategorien, Kommentare, Artikel.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

Voodoopad

Importiert werden: Inhalte.

Konfigurationsoptionen: VoodooPad-Datendatei.

Voodoopad ist eigentlich kein Blogsystem, sondern eher ein Offline-Wiki. Der Importer kann exportierte Daten von Voodoopad übernehmen und als statische Seiten importieren. Daher wird das Serendipity-Plugin *Statische Seiten* benötigt (siehe Seite 410).

Im Importer geben Sie den Pfad zu der XML-Datei an, und alle Datensätze darin können daraufhin importiert werden.

phpBB

Importiert werden: Autoren, Kategorien, Postings.

Konfigurationsoptionen: Zugangsdaten, Zeichensatz, HTML-Maskierung, Trackbacks.

phpBB ist ein Forensystem. Was in einem Blog die Artikel und Kategorien sind, wird in einem Forum als Bereiche und Postings definiert. Serendipity kann alle eingetragenen Benutzer des Forums importieren und als Serendipity-Redakteur übernehmen. Dabei wird auch erkannt, ob ein Forenbenutzer Administratorrechte hat, und er erhält diese dann auch in Serendipity.

Die einzelnen Foren und Unterforen werden in Serendipity als Kategorien und Unterkategorien übernommen. Der jeweils erste Eintrag in einem Foren-Thread wird als Blog-Artikel importiert, alle folgenden Postings des Threads werden als Blog-Kommentar importiert.

Allgemeiner RSS-Import

Importiert werden: Nur Artikel.

Konfigurationsoptionen: RSS-Feed URL, Artikelstatus, Zugeordnete Kategorie, Zeichensatz, Erweiterter Eintrag.

Als letzte Möglichkeit des Artikel-Imports gilt der **Allgemeine RSS-Import**. Hiermit können Sie einen RSS-Feed in Ihr eigenes Blog importieren. Dabei werden die importierten Artikel dem aktuell eingeloggten Serendipity-Benutzer zugeschrieben.

Da in einem RSS-Feed nur sehr wenige Daten vorhanden sind, können nur Artikelstammdaten (Veröffentlichungszeitpunkt, Titel, Text) importiert werden. Sollte eine Kategoriezuordnung im RSS-Feed enthalten sein, versucht der Importer eine gleichnamige Kategorie im Blog zu finden und zuzuordnen. Schlägt das fehl, wird die im Importer festgelegte Standard-Kategorie zugeordnet.

In einigen RSS-Feeds kann neben den Artikel-Teasern auch ein erweiterter Artikeltext enthalten sein, der ebenfalls importiert wird. Wenn Sie die Option **Füge den gesamten importierten Text in das einzelne Text-Feld ein** auf **Ja** setzen, wird sämtlicher verfügbarer Inhalt in das Hauptfeld des Blog-Artikels übernommen. Andernfalls versucht Serendpity, den Teaser-Text und vollständigen Text des RSS-Artikels sinnvoll aufzuteilen.

Als Sonderfall des RSS-Imports dient das WPXRSS-Format. Dieses wird von WordPress-Blogs eingesetzt und enthält außer Artikeln auch Informationen zu Kategorien, Kommentaren und Redakteuren. Das WPXRSS-Format funktioniert nur auf Webservern mit PHP5-Unterstützung.

Bitte beachten Sie, dass dieser Importvorgang nicht für den regelmäßigen Import eines RSS-Feeds gedacht ist. Dafür gibt es das `RSS Aggregator`-Plugin, siehe Seite 338.

4.7.5 Einträge exportieren

Hinter dem Menüpunkt **Einträge exportieren** befindet sich nur ein einzelner Button: **Vollständigen RSS-Feed exportieren**.

Dieser Button ist lediglich ein Link auf Ihren RSS-Feed im Frontend, dem der Parameter `all=1` angehängt ist.

Als Ergebnis dieser Exportaktion wird Ihr Browser also einen RSS-Feed mit allen Ihren Einträgen anzeigen. Diesen RSS-Feed können Sie speichern und in andere Anwendungen importieren.

Bitte beachten Sie unbedingt, dass der RSS-Feed nur einen ganz kleinen Teil Ihres Blogs enthält, nämlich nur den Beitragstext sowie die Überschriften Ihrer Artikel und einige zusätzliche Informationen wie den Autor, die zugeordnete Kategorie und die Veröffentlichungszeit. *Nicht* enthalten sind in diesem RSS-Feed die erweiterten Artikelinhalte, Benutzerinformationen, alle verfügbaren Kategorien und noch vieles mehr. Daher ist ein RSS-Export definitiv nicht als Backup geeignet.

Um die Daten von Serendipity in ein anderes Blogsystem zu überführen, müssen Sie daher auf manuelle Importwege zurückgreifen (siehe Kapitel 8) oder auf die Entwicklung eines Import-Moduls des Fremdsystems für Serendipity hoffen. Serendipity selbst bietet eine Reihe von Import-Modulen für andere Blogsysteme an. Häufig sind die Datenbankstrukturen fremder Systeme nicht so weit entfernt voneinander, so dass mit etwas SQL-Kenntnis ein Import vorgenommen werden kann.

4.8 Bookmarklet

Der Punkt **Bookmarklet** ist eigentlich kein eigenständiger Menüpunkt im Backend. Vielmehr sehen Sie diesen Menüpunkt im Kasten **Weitere Links** auf der **Startseite** des Backends.

Der Link zum Bookmarklet führt dabei auf Ihre eigene Seite und enthält einen JavaScript-Code. Dieser Link ist nicht zum Anklicken gedacht, sondern Sie sollten ihn als Lesezeichen/Favorit in Ihrem Browser speichern.

Wechseln Sie nun auf eine fremde Webseite, über die Sie in einem eigenen Blog-Artikel gerne berichten wollen. Markieren Sie dort einen Textteil, auf den Sie sich beziehen wollen, und wählen Sie in Ihrem Browser das Lesezeichen **Bookmarklet** aus. Sie werden nun auf Ihr Blog geleitet, und die URL der fremden Seite wird automatisch in Ihren Artikeltext eingefügt. So können Sie komfortabel und ohne eigenes Zutun schnell Artikel verfassen.

Kapitel 5

Seitenleisten-Plugins

Serendipity bietet eine große Menge von Plugins an. Einige davon werden direkt bei Serendipity mitgeliefert, andere sind im externen Spartacus-Archiv (siehe Seite 265) verfügbar.

Plugins sind bei Serendipity in *Seitenleisten-* und *Ereignis-Plugins* unterteilt. Wie im Kapitel zur Plugin-Verwaltung (siehe Seite 145) erwähnt, binden sich Seitenleisten-Plugins zur Ausgabe in den Rahmenbereich Ihres Blogs ein. Ereignis-Plugins hingegen dienen der funktionellen Erweiterung Ihres Blogs.

Auf den folgenden Seiten finden Sie eine Übersicht der am häufigsten genutzten Plugins und aller mitgelieferten Plugins. Aus Platzmangel können wir leider nicht auf jedes Plugin eingehen, verzeihen Sie also bitte, falls Ihr Lieblings-Plugin nicht beschrieben wird. Die meisten Plugins sind trotz kurzer Beschreibung leicht zu benutzen und zu konfigurieren. Sollte es darüber hinaus Fragen zu der Benutzung eines Plugins geben, wird man Ihnen im Serendipity-Forum[1] sicherlich helfen.

Einige Plugins bestehen aus einer Kombination von Seitenleisten- und Ereignis-Plugins. Sie werden als *Gekoppelte Plugins* bezeichnet und in einem eigenen Kapitel (ab Seite 385) behandelt.

Plugins können häufig mehr als einmal installiert werden. Gerade bei Seitenleisten-Plugins wie dem HTML-Klotz macht dies Sinn, damit man beliebig viele Inhalte im Blog anzeigen kann. Wenn ein Plugin nicht mehrfach installiert werden kann, erscheint es in der Plugin-Liste mit dem Hinweis „bereits installiert“. Bei Ereignis-Plugins ist es hingegen seltener notwendig, mehr als eine Instanz des Plugins zu installieren. Jedes mehrfach installierte Plugin kann individuell konfiguriert werden, da jede Instanz unabhängig agiert.

Die Konfigurationsoptionen der jeweiligen Plugins sind hier auf die essentiellen Parameter beschränkt und daher nicht vollständig aufgeführt. Im Buch nicht beschriebene Optionen sind jedoch in der Konfigurationsoberfläche eines Plugins aufgeführt und werden dort auch grob

[1] `http://board.s9y.org/`

beschrieben oder sind meistens selbsterklärend.

Leider sind nicht alle Plugins bisher ins Deutsche übersetzt, daher wird bei einigen Beschreibungen und Plugin-Namen im Folgenden auf das englische Original zurückgegriffen.

Damit Sie ein Seitenleisten-Plugin innerhalb der Verzeichnisstruktur besser finden, geben wir in der Übersicht neben dessen Namen auch den Klassennamen der Datei an.

5.1 Standardmäßig aktivierte Plugins

Seitenleisten-Plugins werden im Frontend Ihres Blogs angezeigt und können dort beliebige Inhalte darstellen oder einbinden. Im Gegensatz zu Ereignis-Plugins kann ein Seitenleisten-Plugin ausschließlich Ausgaben in der Seitenleiste vornehmen und keine weitere Kernfunktionalität zur Verfügung stellen. Die folgenden Seitenleisten-Plugins werden bei Serendipity mitgeliefert und sind standardmäßig aktiviert. Einige besitzen keine eigene Plugin-Datei im Unterverzeichnis `plugins`, sondern sind aus Performancegründen in `include/plugin_internal.inc.ph` zusammengefasst.

5.1.1 Kalender serendipity_calendar_plugin

Der Kalender zeigt einen blätterbaren Kalender eines Monats in der Seitenleiste an. Einzelne Tage des Kalenders können dabei hervorgehoben werden und zeigen dem Besucher an, dass an diesem Tag ein Blog-Artikel verfasst wurde. Der aktuelle Tag wird ebenfalls hervorgehoben.

In den Konfigurationsoptionen können Sie den Tag des Wochenanfangs festlegen und die Anzeige von Artikeln zu einem Tag auf eine spezielle Kategorie beschränken.

Die Option **Plugin-Schnittstelle aktivieren** können Sie aktivieren, wenn Sie anderen Plugins erlauben wollen, Tage im Kalender hervorzuheben. Das Plugin *Mein Kalender* kann diese Option beispielsweise benutzen, um Ihre persönlichen Termine anzufügen.

Zuletzt können Sie einstellen, welche Kategorie als Ausgangspunkt der im Kalender angezeigten Blog-Artikel verwendet wird. Standardmäßig werden alle Kategorien dafür benutzt, Sie können aber auch eine spezielle Kategorie auswählen. Wenn der Besucher die Blog-Artikel selbständig auf eine spezielle Kategorie eingrenzt und das Plugin als Ausgangspunkt **Alle Kategorien** verwendet, so wird automatisch die aktuelle Kategorie als Datenquelle zur Kalenderdarstellung eingesetzt.

Die Darstellung erfolgt mittels der Template-Datei `plugin_calendar.tpl`, deren Variablen auf Seite 556 erklärt werden.

5.1.2 Suche serendipity_quicksearch_plugin

Das Plugin *Suche* ist ein simples Seitenleisten-Plugin, das eine Sucheingabe-Box anzeigt. Dort können Besucher Suchwörter eintragen, um Artikel zu durchsuchen. Weitere Hinweise zur Volltextsuche finden Sie auf Seite 88.

Dieses Plugin bietet keine Konfigurationsmöglichkeiten an.

5.1.3 Archive serendipity_archives_plugin

Das Plugin *Archive* zeigt eine Übersicht der letzten Monate an. Ein Klick auf den jeweiligen Monat öffnet die Artikelübersicht für den gewünschten Zeitraum.

In der Konfiguration des Plugins können Sie neben der Überschrift des Plugins auch angeben, ob das Plugin Monate, Wochen oder Tage gliedern soll. Die **Anzahl der Einträge im Archiv** gibt an, wie viele Ausgaben im Plugin erfolgen sollen.

Wenn Sie die Option **Anzahl der Einträge pro Kategorie anzeigen** aktivieren, wird hinter jeder Ausgabe des Plugins in der Seitenleiste die Anzahl von Einträgen eingeblendet, die Sie in diesem Zeitraum erstellt haben. Achten Sie beim Aktivieren dieser Option darauf, dass diese die Geschwindigkeit des Plugins stark verringert. Das Auslesen der Anzahl aller Artikel pro Zeitraum ist eine komplexe Datenbankabfrage.

Wenn Sie die Option **Hide archives link when no entries were made in that timespan (requires counting entries)** aktivieren, können in der Ausgabe der Monatsarchive diejenigen Monate übergangen werden, in denen Sie keine Blog-Artikel geschrieben haben. Damit dies klappt, müssen Sie die Option **Anzahl der Einträge pro Kategorie anzeigen** ebenfalls aktivieren.

5.1.4 Kategorien serendipity_categories_plugin

Dieses Plugin blendet die im Blog verfügbaren Kategorien ein. Wenn Sie Ihr Blog als eine Art CMS betreiben, wird dieses Seitenleisten-Plugin daher oft als Menü benutzt, um auf die jeweiligen Unterbereiche zu gelangen.

Ein Klick auf die im Plugin dargestellte Kategorie ruft dabei jeweils die Artikelübersicht dieser Kategorie auf.

Konfigurationsoptionen sind:

Quelle der Kategorien
: Wie im Kapitel 11 auf Seite 123 erwähnt, können für Kategorien besondere Leserechte

gelten, so dass nur eingeloggte Mitglieder einer Benutzergruppe Einträge dieser Kategorie lesen können.

Standardmäßig stellt das Kategorien-Plugin alle Kategorien dar, die im Blog angelegt sind – ganz gleich, ob dafür besondere Leserechte gelten, oder nicht. Wenn ein Besucher auf eine solche Kategorie klickt, kann es daher sein, dass er darin keine Artikel angezeigt bekommt. Deshalb ist es möglicherweise wünschenswert, wenn einem Leser auch nur die Kategorien angezeigt werden, zu denen er auch Leserechte besitzt.

Dafür ist das Auswahlfeld **Quelle der Kategorien** vorgesehen. Dort wählen Sie eine Benutzergruppe aus, so dass nur Kategorien angezeigt werden, die für die gewählte Benutzergruppe sichtbar sind. Standardmäßig steht diese Option auf **Alle Autoren**. Sie müssen die Sonderoption **Derzeitiger Autor** wählen, wenn die Kategorieliste sich nach den Rechten des aktuell eingeloggten Besuchers richten soll.

Zeige nur Kategorien unterhalb…

Da Kategorien in Serendipity schachtelbar sind, kann es recht komplexe Kategorie- bzw. Menüstrukturen abbilden. Möglicherweise soll in der Seitenleiste aber nur ein Teil dieser Struktur angezeigt werden. Daher können Sie im Auswahlfeld **Zeige nur Kategorien unterhalb…** festlegen, ab welcher Oberkategorie die Unterrubriken angezeigt werden.

Die gewählte Oberkategorie verstecken

Wenn Sie im obigen Konfigurationsfeld eine Kategorie ausgewählt haben, bestimmt die Option **Die gewählte Oberkategorie verstecken**, ob der Name dieser Oberkategorie angezeigt (**Nein**) oder versteckt (**Ja**) wird.

XML-Button

Für die Artikel jeder einzelnen Kategorie kann ein RSS-Feed abonniert werden. Dazu wird neben jedem Kategorienamen ein kleines Bild eingeblendet, auf das ein Benutzer klicken kann, um den RSS-Feed der Kategorie zu abonnieren.

Dieses Bild stellt üblicherweise ein universelles RSS-Icon dar, das von zahlreichen Blogsystemen und Browsern einheitlich benutzt wird. Sie können aber gerne in dem Konfigurationsfeld **XML-Button** ein abweichendes Bild eintragen. Dazu müssen Sie den vollständigen HTTP-Pfad (alles hinter `http://www.example.com/`) eingeben.

Wenn Sie kein Bild einbinden wollen, tragen Sie hier `none` ein.

Sortierung

Zwei Auswahlfelder untereinander werden bei der Konfigurationsoption **Sortierung** dargestellt. Das obere Auswahlfeld bestimmt, nach welchem Kriterium die Liste der Kategorien sortiert werden soll.

Da Kategorien innerhalb Serendipity noch nicht beliebig sortiert werden können, entscheidet also ausschließlich diese Sortierungsreihenfolge die Reihenfolge der Darstellung in der Seitenleiste. Standardmäßig werden die Kategorien nach dem Namen der Kategorie alphabetisch sortiert. Weitere Möglichkeiten sind eine Sortierung nach Beschreibung oder der ID einer Kategorie. Auch kann eine Sortierung nach *keinem* Kriterium ausgewählt werden, in diesem Fall werden die Kategorien nach Erstellungszeit in der Datenbank sortiert.

Das zweite Auswahlfeld gibt dabei an, ob die Kategorien absteigend (also von A bis Z oder 0 bis 9) oder aufsteigend (von Z bis A bzw. 9 bis 0) sortiert werden sollen.

Wenn Sie eine ganz individuelle Auflistung von Kategorien haben möchten, gibt es zwei Möglichkeiten. Die erste Möglichkeit wäre, einfach eine manuelle HTML-Liste der Kategorien zu erstellen und als *HTML-Klotz Seitenleisten-Plugin* (Seite 216) einzufügen. Dadurch verlieren Sie natürlich die anderen dynamischen Möglichkeiten des Plugins.

Die zweite Möglichkeit wäre, das Feld **Beschreibung** einer Kategorie zu missbrauchen, indem Sie nicht einen Text, sondern eine Zahl darin aufschreiben. Durch die Eingabe einer Zahl für die Reihenfolge der Kategorien können Sie in der Sortierung die Reihenfolge beeinflussen. Das bedeutet jedoch auch, dass Sie dann keine normale textliche Beschreibung der Kategorie mehr nutzen können. Da diese Beschreibung sonst aber nur für Redakteure gilt, ist dies möglicherweise zu verkraften.

Besuchern erlauben, mehrere Kategorien gleichzeitig darzustellen?

Wenn ein Besucher in der Darstellung des Plugins auf eine einzelne Kategorie klickt,

sieht er alle Einträge dieser Kategorie. In manchen Fällen möchte ein Besucher aber gerne Inhalte mehrerer Kategorien auf einmal ansehen.

Wenn Sie die Option **Besuchern erlauben, mehrere Kategorien gleichzeitig darzustellen** aktivieren, wird neben jeder Kategorie ein Ankreuzfeld angezeigt. Der Besucher kann alle gewünschten Kategorien markieren und danach auf den Button **Los** klicken, um die Auswahl zu setzen.

Ist die Option deaktiviert, wird diese Auswahlbox nicht mehr angezeigt. Dennoch kann ein Besucher spezielle URLs aufrufen, in denen er mehrere Kategorien auf einmal ansehen kann (siehe Seite 78). Die Deaktivierung der Option nimmt dem Benutzer also höchstens die komfortable Möglichkeit zu einer solchen Übersicht und beeinflusst die Darstellungsweise des Plugins.

Kategorien verstecken, die nicht Teil des vom Besucher gewählten

Kategoriebaums sind Wenn Sie eine verschachtelte Kategoriestruktur eingerichtet haben, stellt diese eine Baumstruktur dar. Dabei sind jeweils einer Oberkategorie eine oder mehrere Unterkategorien zugeordnet.

Wenn Sie Serendipity als eine Art CMS einsetzen, bei dem die Kategorien Menüpunkten entsprechen, wünschen Sie sich möglicherweise, dass bei einem Klick auf eine Unterkategorie die anderen „parallelen“ Menüpunkte nicht mehr angezeigt werden.

Stellen Sie sich folgenden Kategorienbaum vor: Oberkategorien A, B und C. Jede Kategorie besitzt zwei Unterkategorien: A1 und A2, B1 und B2 sowie C1 und C2. Wenn der Besucher nun auf die Unterkategorie A1 klickt, würden Sie gerne nur die Kategorien A und A2 anzeigen. Die Kategorien und Unterkategorien von B und C sollen verschwinden. Genau zu diesem Zweck dient die Option **Kategorien verstecken...** Wenn diese Option aktiviert ist, werden die nicht zum gewählten Kategoriebaum passenden restlichen Kategorien nicht mehr in der Ausgabe des Plugins angezeigt. Sie müssten dann zuerst wieder auf die Startseite zurück (oder auf andere Menüpunkte klicken), um die ausgeblendeten Kategorien wieder zu sehen.

Anzahl der Einträge pro Kategorie anzeigen

Wenn Sie möchten, dass die Anzahl der Artikel in einer Kategorie in der Ausgabe des Plugins neben dem Kategorienamen angezeigt wird, können Sie die Option **Anzahl der Einträge pro Kategorie anzeigen** aktivieren.

Beachten Sie auch hier, dass die Aktivierung die Geschwindigkeit des Plugins stark beeinflusst, da das Zählen aller Artikel der Datenbank einige Zeit beansprucht.

Wenn Sie Einträge zu mehr als einer Kategorie zuweisen, wird dieses Plugin (leider) für jede zugewiesene Kategorie den Artikel erneut zählen. Daher kann es passieren, dass die angezeigte Artikelzahl höher ist als die in Wirklichkeit vorhandenen Artikel.

Smarty-Templating aktivieren

Damit Sie dieses Plugin möglichst komfortabel anpassen können, gibt es die Template-Datei `plugin_categories.tpl`. Nur wenige Seitenleisten-Plugins verfügen über

ein eigenes Smarty-Template, da meistens Änderungen per CSS bereits vollkommen ausreichen. Aus Performancegründen wird diese Template-Datei daher auch standardmäßig nicht benutzt. Wenn Sie also ein eigenes Template für das Aussehen der Kategorie-Auflistung benutzen wollen, müssen Sie die Konfigurationsoption **Smarty-Templating aktivieren** auf `Ja` setzen.

Die eingesetzten Variablen dieser Datei werden auf Seite 558 erklärt.

5.1.5 Blog abonnieren serendipity_syndication_plugin

Das Plugin stellt die verschiedenen Optionen zum Abonnieren Ihres Blogs mittels RSS-Feed dar. Dabei wird in der Seitenleiste für jedes gewünschte Format ein Link eingebunden, den Ihre Besucher in ihr RSS-Programm übernehmen können.

Neben dem Titel des Plugins in der Seitenleiste sowie der Aktivierung der zahlreichen Feed-Formate bietet das Plugin folgende Konfigurationsoptionen:

Volle Einträge mit erweitertem Text im RSS-Feed einbinden
: Ein RSS-Feed enthält üblicherweise nur den Text eines Artikels, den Sie im Feld *Eintrag* erfasst haben. Der *Erweiterte Eintrag* ist nicht Bestandteil der RSS-Feeds, und somit muss ein Benutzer Ihr Blog besuchen, um den vollständigen Artikel lesen zu können. Dies ist häufig gewünscht, um mehr Besucher auf die Webseite zu „locken“. Um Ihren Besuchern mehr Komfort zu bieten, können Sie aber auch den vollständigen Artikel im RSS-Feed eintragen.

  Mit dieser Konfigurationsoption können Sie dieses Verhalten beeinflussen. Wenn Sie die Option **Client** aktivieren, kann, wie im Abschnitt 3.6 auf Seite 86 erwähnt, der Leser selbst bestimmen, ob er einen vollständigen RSS-Feed haben will oder nicht. Dazu muss er Ihren RSS-Feed über die Datei `rss.php?fullFeed=true` abonnieren. Dies ist natürlich relativ versteckt, so dass Sie Ihre Besucher über diese mögliche Variante informieren müssten.

FeedBurner Feed
: Der Webservice `http://www.feedburner.com/` ist ein Angebot von Google, das Statistiken über die Auslieferung Ihres RSS-Feeds und einige weitere Möglichkeiten der Einbettung bietet. Bei einem FeedBurner-Feed kann der Aufruf des RSS-Feeds im Browser direkt hübsch formatiert sein, statt als für den Besucher merkwürdiger XML-Datensatz zu erscheinen. FeedBurner ermöglicht auch die Einbindung von Werbung und verwandten Tags.

  Bei FeedBurner müssen Sie sich separat anmelden und dem Service die URL Ihres RSS-Feeds mitteilen. FeedBurner wird dann regelmäßig diese URL aufrufen, Ihren RSS-Feed einlesen, ihn konvertieren und danach unter einer eigenen FeedBurner-URL (die Sie konfigurieren können) anbieten.

Damit die Statistikzählung aktiviert wird, müssen Ihre Besucher künftig diesen FeedBurner-Feed abonnieren und nicht den üblichen Serendipity-Feed. Dazu dient die Einstellung **Feedburner Feed** im Plugin. Wenn diese Variable auf **Ja** steht, wird der Link zum Feedburner-Feed in Ihrer Seitenleiste angezeigt. In diesem Fall sollten Sie alle anderen Feed-Formate deaktivieren, da die Besucher ja nur noch den FeedBurner-Feed aufrufen sollen.

Wenn Sie die FeedBurner-Option lediglich auf **Ja** gestellt haben, bleiben die URLs zu Ihren alten RSS-Feeds weiterhin intakt, sie werden lediglich versteckt. Wenn Sie Ihre Besucher aber in Zukunft auch von diesen verstecken URLs fernhalten wollen, stellen Sie die FeedBurner-Option auf **Erzwingen**. Dadurch wird jeder Aufruf des Serendipity-RSS-Feeds automatisch auf die FeedBurner-Feeds umgeleitet.

Zu welchem FeedBurner-Feed Serendipity verweist, stellen Sie am Ende der Konfigurationsoptionen ein.

E-Mail-Adresse einbinden?

Standardmäßig muss ein RSS-Feed zu jedem Autor eine E-Mail-Adresse anzeigen. Dies wird leider gegenwärtig häufig von SPAM-Robotern missbraucht, um E-Mail-Adressen im Internet zu sammeln. Daher können Sie über die Option **E-Mail-Adresse einbinden** einstellen, ob Ihre echte E-Mail-Adresse oder eine vorgetäuschte E-Mail-Adresse (`nospam@example.com`) eingebunden wird.

Beachten Sie, dass, wenn Sie im Anti-Spam-Plugin (Seite 239) die gleichnamige Option aktiviert haben, diese ebenfalls die Darstellung der E-Mail-Adressen beeinflusst.

In abgegrenzten Blogs (im Intranet zum Beispiel) macht die Aktivierung dieser Option noch Sinn, beim Einsatz im Internet ist es jedoch empfehlenswert, die Option auf **Nein** zu setzen.

Feld „managingEditor“

Der RSS 2.0-Feed ermöglicht die Einbindung eines optionalen Feldes, das die E-Mail-Adresse des leitenden Redakteurs eines Blogs enthalten kann. Abonnenten des RSS-Feeds können sich so leicht an den inhaltlich Verantwortlichen wenden. Auch hier gelten die obigen Bedenken, dass enthaltene E-Mail-Adressen oft Spammern zum Opfer fallen könnten.

Feld „webMaster“

Analog zu einem inhaltlich Verantwortlichen kann diese Option eine E-Mail-Adresse des technisch Verantwortlichen enthalten.

Feld „ttl“ (time-to-live)

RSS-Feeds müssen von RSS-Readern regelmäßig abgerufen werden, damit neue Artikel gefunden werden können. Üblicherweise wird ein solcher RSS-Reader alle 30 oder 60 Minuten (je nach Einstellung Ihrer Besucher) Ihre RSS-Feeds abrufen. Sie können über das Feld **ttl** (*Gültigkeitsdauer*) innerhalb des RSS-Feeds für RSS-Reader vorschlagen, wie lange Ihr Feed üblicherweise gültig ist. Diesen eingestellten Wert

können RSS-Programme dann als Richtlinie für die Aktualisierungshäufigkeit benutzen.

Der **ttl**-Wert wird in Minuten angegeben, wenn Sie also üblicherweise nur einen Artikel pro Tag online stellen, können Sie ihn auf 1440 stellen (entspricht 24 Stunden). Damit können Sie dann möglicherweise einiges an Traffic auf Ihrem Webserver sparen, wenn sich die RSS-Reader daran halten.

Feld „pubDate“

Ein RSS-Feed kann ein optionales **pubDate**-Feld enthalten, das Auskunft über den Erstellungszeitpunkt des letzten Artikels gibt. Dieser Wert wird von vielen RSS-Readern benutzt, um Aktualisierungen festzustellen, daher raten wir von der Deaktivierung dieser Option ab.

XML-Button

Neben jedem RSS-Feed wird in der Seitenleiste eine kleine Grafik angezeigt, die standardmäßig dem Standardsymbol für RSS-Feeds entspricht. Wenn Sie eine eigene Grafik benutzen wollen, können Sie den Pfad zu der Grafik in dem Feld **XML-Button** eintragen. Die Grafik muss sich dabei in Ihrem Template-Verzeichnis befinden, und der hier eingetragene Pfad muss relativ zu Ihrem Template-Verzeichnis sein. Wenn die Datei also in `/var/www/example.com/serendipity/templates/meinTemplate/bilder/MeinRSS.gif` liegt, tragen Sie im Konfigurationsfeld `bilder/MeinRSS.gif` ein.

Bild für den RSS-Feed

Das RSS 2.0-Format ermöglicht es, ein eigenes Logo für den Feed einzutragen. Diese Grafik kann dann von RSS-Readern in der Titelzeile oder an anderen Stellen angezeigt werden. Wenn Sie hier nichts eintragen, wird standardmäßig das Serendipity-Logo eingebunden.

Tragen Sie in diesem Feld die vollständige URL zu einem Bild ein, also etwa `http://www.example.com/MeinLog`
Diese Grafikdatei darf maximal 144 Pixel breit und 400 Pixel hoch sein. Die genauen Maße des Bildes müssen Sie in den beiden darunterliegenden Feldern **Breite des Bildes** und **Höhe des Bildes** eintragen.

FeedBurner-ID, Titel, Bildunterschrift und Bild

Die letzten vier Konfigurationsoptionen betreffen Sie nur, wenn Sie den oben erwähnten FeedBurner-Service benutzen. Als **ID** müssen Sie die im FeedBurner dargestellte ID Ihres Feeds eintragen, sie entspricht dem Namen, den Sie in der URL `http://feeds.feedburner.com/`*`ID`* sehen.

Wenn Sie den Feedburner-Feed aktiviert haben, wird ein Link dorthin im Seitenleisten-Plugin angezeigt. Dieser Link wird begleitet von einem Bild und einem Linktitel. Im Feld **FeedBurner Titel** können Sie diesen Titel vergeben, der neben der Grafik dargestellt wird. Die Grafikdatei müssen Sie mit einer vollständigen URL wie `http://www.example.com/feedburner.jpg` eintragen. Wenn Sie dieses Feld leer lassen, sehen

Sie eine Standardgrafik, die den Besucherzähler darstellt. Im Feld **FeedBurner Bild-Unterschrift** können Sie einen Text hinterlegen, der angezeigt wird, wenn Sie mit der Maus über die FeedBurner-Grafik fahren. Dort können Sie beispielsweise eine kurze Beschreibung eintragen, warum Sie FeedBurner benutzen.

5.1.6 Verwaltung des Blogs serendipity_superuser_plugin

Mit dem Plugin *Verwaltung des Blogs* binden Sie einen einfachen Link im Frontend ein, auf den Sie als Redakteur klicken können, um ins Backend zu gelangen. Das Plugin erkennt dabei automatisch, ob Sie schon eingeloggt sind, und gibt dementsprechend entweder einen Link zum Login oder direkt zum Backend aus.

Die einzige Konfigurationsoption des Plugins gibt an, ob Sie das **HTTPS** zum Login benutzen möchten. Ihr Server muss HTTPS (siehe Terminologie auf Seite 48) unterstützen und ein gültiges Zertifikat für den Server besitzen, auf dem Serendipity installiert ist. Nur bei der Benutzung von HTTPS ist gewährleistet, dass Ihr Benutzername und Passwort nicht im Klartext (sondern verschlüsselt) an den Server übertragen werden und niemand Ihre Logindaten abfangen kann.

5.1.7 Powered by serendipity_plug_plugin

Ein ebenfalls recht simples Plugin ist *Powered by*. Es gibt das Logo von Serendipity mit einem Link zur Webseite `http://www.s9y.org/` aus und erlaubt es Ihren Besuchern, zu sehen, welche großartige Software hinter Ihrem Blog steckt.

Die Konfigurationsoptionen des Plugins ermöglichen eine Feineinstellung, ob man nur das Logo oder nur den Text anzeigen möchte.

5.2 Weitere mitgelieferte Plugins

Neben den standardmäßig aktivierten Plugins können Sie natürlich auch weitere Seitenleisten-Plugins installieren. Es folgt eine Liste der Seitenleisten-Plugins, die bei Serendipity standardmäßig mitgeliefert werden.

5.2.1 Fremder RSS/OPML-Blogroll Feed serendipity_plugin_remoterss

Dieses leicht kryptisch klingende Plugin bietet eine recht einfache Möglichkeit, die RSS-Feeds fremder Seiten einzubinden.

So können Sie in Ihrer Seitenleiste beispielsweise RSS-Feeds von Newstickern einbinden oder auch die RSS-Feeds von diversen Webservices wie del.icio.us oder last.fm einbinden, die Daten von diesem Webserver anzeigen können.

Das Plugin lässt sich auch mehrfach installieren, so dass Sie beliebig viele RSS-Feeds einbinden können.

Abgesehen von RSS-Feeds sind auch sogenannte OPML-Blogrolls möglich. OPML-Dateien werden von vielen RSS-Readern erstellt und können Linklisten enthalten, welche Blogs Sie lesen. Wenn diese OPML-Datei dann in Ihrer Seitenleiste eingebunden wird, können Sie so relativ leicht automatisch Ihre Lieblingsblogs auflisten.

Das RSS-Seitenleisten-Plugin agiert grundsätzlich wie ein eigenständiger RSS-Reader: Ihr Blog wird in einem definierten Zeitraum die RSS-Feeds selbständig abholen, einbinden und für einen gewissen Zeitraum zwischenspeichern, um Bandbreite zu sparen. Damit das Plugin funktionieren kann, gelten dieselben Rahmenbedingungen wie für Spartacus (siehe Seite 265): Ihr Server darf nicht von einer Firewall für ausgehende Verbindungen blockiert werden.

Pro installiertem RSS-Feed-Plugin kann eine Vielzahl an Optionen eingestellt werden:

Feed-Titel
: Im Feld **Feed-Titel** stellen Sie ein, wie die Überschrift des RSS-Feeds innerhalb der Seitenleiste lauten soll.

Typ des Feeds
: In diesem Auswahlfeld legen Sie das Format des fremden Feeds fest. Sie können zwischen RSS und OPML wählen, wobei für RSS-Feeds alle Versionen von 0.91 bis 2.0 unterstützt werden, und von OPML die (einzig existierende) Version 1.0.

RSS/OPML-URI
: Die Quell-URL des RSS-Feeds (oder des OPML-Feeds) müssen Sie in das Feld **RSS-OPML-URI** eintragen. Geben Sie dort eine vollständige URL an, also beispielsweise `http://www.example.com/serendipity/rss.php?version=2.0`.

RSS-Zielelement
: In einem RSS-Feed stehen mehrere Felder zur Verfügung, die das Seitenleisten-Plugin darstellen kann. Im Feld `title` wird beispielsweise der Titel eingebunden, das Feld `content` enthält den Artikelinhalt. Manchmal kann es jedoch sein, dass die Felder für den Inhalt `content:encoded` oder `body` lauten.

  Damit das Seitenleisten-Plugin flexibel eingesetzt werden kann, müssen Sie den Namen des Feldes angeben, das Serendipity in der Seitenleiste darstellen soll. Die Liste der verfügbaren Felder können Sie in der Spezifikation des jeweiligen RSS-Formats nachlesen[2]. Sie können beliebig viele Feldinhalte anzeigen lassen, indem Sie mehrere Felder mittels `,` voneinander trennen.

[2] `http://www.rssboard.org/rss-2-0` für RSS 2.0; weitere Links finden Sie auf `http://de.wikipedia.org/wiki/RSS`

Damit nicht zu viel Platz verschwendet wird, gibt Serendipity üblicherweise nur den Titel (`title`) eines RSS-Feeds an.

Smarty-Templating aktivieren
Die Darstellung des Feeds in der Seitenleiste kann über die Smarty-Template-Datei `plugin_remoterss.tpl` gesteuert werden, wenn die Option **Smarty-Templating aktivieren** gewählt wurde. Andernfalls benutzt das Plugin für seine Ausgaben die integrierte HTML-Darstellung.

Ein Beispiel für eine Template-Datei liegt im Verzeichnis `plugins/serendipity_plugin_remoters` Die
dort verfügbaren Variablen sind auf Seite 560 dokumentiert.

Grundsätzlich sollten Sie die Templating-Option nur dann benutzen, wenn Sie an der Standardausgabe Anpassungen vornehmen wollen, die Sie nicht bereits mittels CSS-Anpassung oder Konfiguration des Plugins beeinflussen können.

Anzahl der Einträge
Standardmäßig zeigt das Seitenleisten-Plugin alle Einträge/Artikel eines RSS-Feeds. Wenn Sie aber die Anzahl auf eine gewisse Zahl begrenzen möchten, können Sie diese hier eintragen.

RSS-Link verwenden
Jeder Artikel eines RSS-Feeds besitzt üblicherweise einen Link, der zu der Quellseite des Artikels im Original-Blog (oder Webservice) führt. Falls die Option **RSS-Link verwenden** aktiviert ist, wird dieser Link in der Ausgabe des Plugins dargestellt.

HTML-Ausgabe escapen
Es ist möglich (und erlaubt), dass ein Artikel eines (fremden) RSS-Feeds HTML-Code enthält. Dies ermöglicht z. B. Fettungen und eigene HTML-Links in den dargestellten Artikeln.

Bei der Darstellung fremder RSS-Feeds könnte dies jedoch zu Sicherheitsproblemen führen, denn immerhin kann so auch möglicherweise bösartiges JavaScript in Ihrem Blog ausgeführt werden, das bei Ihren Besuchern Logindaten oder anderes ausspäht (dies nennt man *XSS-Angriff*).

Bei RSS-Feeds, deren Quelle Sie nicht absolut vertrauen, sollten Sie die Option **HTML-Ausgabe escapen** unbedingt aktivieren. Wenn Sie jedoch RSS-Feeds von vertrauenswürdigen Webservices einbinden (wie etwa Ihre Bilderliste von Flickr), dann werden Sie diese Option deaktivieren müssen, weil sonst keine Bilder angezeigt würden.

Display Date, Datumsformat
Jeder Artikel eines RSS-Feeds besitzt üblicherweise ein Veröffentlichungsdatum. Wenn Sie dieses zusätzlich unterhalb des RSS-Artikels anzeigen wollen, stellen Sie die Option **Display Date** auf **Ja**.

Die Option **Datumsformat** bezieht sich auf die Formatierung des Veröffentlichungsdatums. Die Variablen, die Sie hier benutzen können, sind auf `http://de3.php.net/strftime` zusammengefasst.

Zeichensatz

Jeder RSS-Feed enthält Artikel in einem speziellen Zeichensatz. Ein Großteil der RSS-Feeds ist im **UTF-8**-Zeichensatz verfasst, aber einige Feeds (gerade deutsche) enthalten die Artikel im **ISO-8859-1**-Zeichensatz.

Damit Serendipity fremde Artikel später im Blog mit korrekten Sonderzeichen darstellen kann, müssen Sie Serendipity über die Option **Zeichensatz** mitteilen, welchen Zeichensatz der RSS/OPML-Feed benutzt. Sie können dies manuell relativ leicht herausfinden, indem Sie die fremde URL aufrufen und in der XML-Ausgabe nach einer Angabe mit dem Titel **charset** in der ersten Zeile der Datei suchen.

Leider benutzt Serendipity eine ältere RSS-Bibliothek zum Einlesen der RSS-Feeds, daher wird Ihnen diese Aufgabe derzeit noch nicht vom System abgenommen.

Link-Target

Wenn Sie bei einem Artikel Hyperlinks zulassen (Option **RSS-Link verwenden**), wird ein Link standardmäßig im selben Browser-Fenster wie Ihr Blog geöffnet. Über das HTML-Attribut namens `target` kann man jedoch auch ein neues Browser-Fenster öffnen, wenn man bei der Option **Link-Target** ein `_blank` einträgt.

Bitte beachten Sie, dass ein solches `target=_blank` dazu führt, dass ein Besucher einen Link nicht so einfach *nicht* in einem neuen Fenster öffnen kann und dass ihn dies möglicherweise stört. Neue XHTML-Standards empfehlen daher, auf die Benutzung von `target` im Optimalfall zu verzichten und dem Besucher die volle Kontrolle zu überlassen, wie er einen Link in seinem Browser öffnen möchte.

Wann wird der Feed aktualisiert

Theoretisch würde bei jedem Besuch Ihres Blogs der fremde RSS-Feed neu aufgerufen und angezeigt. Da dies viel Bandbreite und auch viel Zeit kosten würde, speichert das Plugin die Ausgaben des fremden Feeds für einen gewissen Zeitraum. Diesen Zeitraum können Sie über die Option **Wann wird der Feed aktualisiert** selber vorgeben, indem Sie hier eine Dauer (in Sekunden) eintragen. Standardmäßig wird ein Feed für drei Stunden gecached, bevor ein neuer Besucher Ihrer Webseite ihn neu erzeugt.

Bei besonders häufig aktualisierenden RSS-Feeds können Sie dieses Limit natürlich heruntersetzen.

Bullet Image

Falls Sie gerne die einzelnen Artikel eines fremden RSS-Feeds voneinander trennen wollen, können Sie in dem Feld **Bullet Image** eine beliebige vollständige URL einer Grafikdatei eintragen. Diese Grafik wird dann nach jedem Artikel angezeigt.

Textformatierung(en) durchführen

Wenn die Option **Textformatierung(en) durchführen** aktiviert ist, wird die Ausgabe des RSS-Feeds an alle bei Ihnen installierten Textformatierungs-Plugins weitergereicht (siehe Seite 237). Diese können beliebige Umwandlungen vornehmen, die dann auch die Darstellung des RSS-Feeds beeinflussen.

5.2.2 Event-Ausgabe Wrapper serendipity_plugin_eventwrapper

Das Plugin **Event-Ausgabe Wrapper** ist ein recht spezielles Plugin, da es die Grenze zwischen Seitenleisten-Plugin und Ereignis-Plugin etwas aufweicht.

Eigentlich ist es ausschließlich Seitenleisten-Plugins möglich, eine Ausgabe in der Seitenleiste vorzunehmen. Ereignis-Plugins können nur auf andere Bereiche und Funktionalitäten Serendipitys Einfluss nehmen, was wiederum Seitenleisten-Plugins nicht erlaubt ist.

Seltene Fälle von Ereignis-Plugins bieten jedoch auch die Möglichkeit an, eine spezielle Ausgabe in der Seitenleiste vorzunehmen. Diese Möglichkeit wurde eher in frühen Versionen Serendipitys benutzt (z. B. vom Plugin „Link List“ oder „Worte ersetzen“). Inzwischen nutzen fast alle Plugins die einfachere Möglichkeit, ein zugehöriges Seitenleisten-Plugin (siehe 6.5.6 ab Seite 385) zu bestimmen und dieses konzeptionelle Problem für Sie als Benutzer einfacher zu lösen.

Wenn dieses Ausgabe-Wrapper-Plugin von einem anderen benötigt wird, finden Sie einen Hinweis dazu in dem jeweiligen Plugin.

In der Konfiguration dieses Plugins können Sie das **Quell-Ereignis-Plugin**, das eingebunden werden soll, über ein Auswahlfeld bestimmen. Den Titel des Plugins in der Seitenleiste können Sie ebenfalls in einem Eingabefeld eintragen.

Wenn Sie im Frontend keine Ausgaben zu dem gewählten Ereignis-Plugin auffinden, bedeutet das lediglich, dass das gewählte Plugin keine Ausgaben anbietet. Sie können also z. B. zwar das Plugin *Textformatierung: Smilies* auswählen, da dieses aber den *Event-Ausgabe Wrapper* nicht unterstützt, werden Sie keine Ausgabe sehen.

Für Techniker ist erwähnenswert, dass alles, was dieses Plugin tut, darin besteht, die `generate_content()`-Methode eines Ereignis-Plugins aufzurufen und die Ausgaben dieser Funktion an die für Seitenleisten-Plugins übliche (gleichbenannte) Methode `generate_content()` weiterzureichen. Aus Performance-Gründen (und Separierungsgründen) gibt es keinen Event-Hook, um ein Seitenleisten-Plugin zu *emulieren*.

5.2.3 Aktuelle Einträge serendipity_plugin_recententries

Das Plugin *Aktuelle Einträge* zeigt in der Seitenleiste eine kurze Übersicht der neuesten Artikel in Ihrem Blog an. Dies ermöglicht eine konzentrierte Übersicht der Artikel an zentraler Stelle, die ein Besucher auch dann sieht, wenn er gerade die Detailseite eines Artikels anschaut.

Neben dem üblichen Seitenleisten-Titel und der Anzahl an Artikeln, die eingebettet werden sollen, bietet das Plugin folgende besonderen Optionen:

Angezeigte Einträge überspringen

Wenn Sie die Option **Angezeigte Einträge überspringen** auf **Alle Anzeigen** stellen, zeigt die Seitenleiste stets die letzten aktuellen Einträge. Dies kann jedoch zu einer Dopplung führen, da Sie ja auf der Startseite des Frontends bereits die letzten 15 Artikel anzeigen.

Wenn Sie also nur die aktuellsten Artikel anzeigen wollen und dabei die auf der Startseite bereits angezeigten Einträge überspringen wollen, wählen Sie die Option **Einträge auf der Hauptseite überspringen**.

Datumsformat
Zu jedem Artikel wird das Erstellungsdatum angezeigt. Wie dieses formatiert wird, tragen Sie im Feld **Datumsformat** ein. Die zur Verfügung stehenden Variablen finden Sie auf `http://de3.php.net/strftime`.

Kategorie
Standardmäßig zeigt das Plugin die Artikel aller Kategorien chronologisch sortiert an. Wenn Sie jedoch in der Seitenleiste nur die letzten Einträge in einer speziellen Kategorie darstellen wollen, können Sie diese spezielle Kategorie in dem Mehrfach-Auswahlfeld angeben. Mehrere Kategorien können Sie hier auch wählen, indem Sie mit gedrückter *(Strg/Apfel)*-Taste auf die Kategorien klicken.

Die Sonderoption **Übergeordnete Kategorie** bedeutet, dass als Quelle für die Liste der aktuellen Einträge die Kategorie verwendet wird, die der Besucher im Blog gerade aufgerufen hat.

Show Random Articles
Wenn Sie eine zufällige Auswahl von Artikeln darstellen wollen, können Sie die Option **Show Random Articles** aktivieren. Bei jedem Seitenaufruf werden dann zufällig andere Artikel in der Seitenleiste angezeigt.

5.2.4 Autoren serendipity_authors_plugin

Das Plugin *Autoren* stellt in der Seitenleiste eine Liste aller im Blog eingetragenen Redakteure dar. Ein Klick auf einen Autor führt dann zu der Übersicht der von ihm geschriebenen Artikel.

Die Konfigurationsoptionen des Plugins sind:

XML-Button
Für jeden Autor steht ein eigener RSS-Feed-Link zur Verfügung, über den Einträge dieses Autors abonniert werden können. Die Grafik muss sich dabei in Ihrem Template-Verzeichnis befinden, und der hier eingetragene Pfad muss relativ zu Ihrem Template-Verzeichnis sein.
Wenn die Datei also in `/var/www/example.com/serendipity/templates/meinTemplate/bilder/Me`

liegt, tragen Sie im Konfigurationsfeld `bilder/MeinRSS.gif` ein.

Ermöglicht Besuchern, Einträge mehrerer Autoren gleichzeitig darzustellen
Ähnlich wie bei der Liste aller Kategorien, kann in der Liste aller Autoren vom Besucher eine Auswahl getroffen werden, welche Artikel er dargestellt sehen will. Üblicherweise kann er mit dem Klick auf den Link eines Redakteurs nur die Einträge eines einzelnen Redakteurs sehen.

Ist die Option **Ermöglicht Besuchern, Einträge mehrerer Autoren gleichzeitig darzustellen** aktiviert, wird neben jedem Redakteur ein Ankreuzfeld angezeigt, das der Besucher für alle gewünschten Redakteure markieren kann, um so die Übersicht der Artikel dieser Redakteursgruppe lesen zu können.

Anzahl der Artikel neben dem Autor-Namen anzeigen
Wenn Sie diese Option aktivieren, wird in Klammern hinter dem Redakteursnamen die Anzahl der von ihm verfassten Artikel angezeigt. Bitte beachten Sie, dass die Aktivierung dieser Zählung relativ datenbankintensive Abfragen durchführen muss und daher möglicherweise zu Geschwindigkeitseinbußen führen kann.

Nur Autoren mit mindestens X Beiträgen anzeigen
Möglicherweise möchten Sie nur aktive Redakteure des Blogs aufführen. Daher können Sie in dem Eingabefeld eine Mindestzahl an Beiträgen eintragen, die ein Redakteur geschrieben haben muss, um in der Seitenleiste angezeigt zu werden. Für diese Option gelten dieselben Performance-Bedenken wie bei der vorherigen. Sollten Sie eine von beiden aktivieren, wird die Aktivierung der anderen Option jedoch darüber hinaus keine weiteren Einbußen zur Folge haben.

5.2.5 HTML-Klotz serendipity_html_nugget_plugin

Das Plugin namens *HTML-Klotz* ist wohl das meistunterschätzte Plugin – sobald man es einmal einzusetzen weiß, ist es aus einem Blog nicht mehr wegzudenken. Denn mit diesem Plugin können Sie beliebiges HTML und beliebiges JavaScript in Ihre Seitenleiste einfügen.

Dieses Plugin lässt sich beliebig oft installieren und kann daher flexibel positioniert werden. Der Einsatz eines HTML-Klotzes macht zudem Dateiänderungen auf dem Server unnötig, denn alles kann über die Plugin-Verwaltung eingestellt werden.

Pro HTML-Klotz gelten folgende Konfigurationsoptionen:

Titel
Die Überschrift des HTML-Klotzes, die in der Seitenleiste angezeigt wird. Wenn Sie den Titel leer lassen, wird kein Titel angezeigt. Dies ist besonders dann hilfreich, wenn Sie ein JavaScript einfügen wollen, das keine Ausgabe besitzt und sonst nur zur Anzeige eines leeren Klotzes führen würde. Der Titel eines Plugins wird zusätzlich in

der Plugin-Verwaltung mit ausgegeben, damit Sie mehrere HTML-Klötze voneinander unterscheiden können.

Zusätzlicher Informationstext, der auf der Plugin-Oberfläche

dargestellt wird In diesem Eingabefeld können Sie eine kleine Beschreibung angeben, die in der Plugin-Verwaltung neben jedem HTML-Klotz angezeigt wird. Dies ist besonders hilfreich in Fällen, bei denen ein HTML-Klotz ohne Titel angelegt wurde.

Inhalt

In diesem großen Eingabefeld tragen Sie den HTML-Code oder das JavaScript ein, das Sie in der Seitenleiste darstellen wollen. Wenn Sie den WYSIWYG-Editor in den **Eigenen Einstellungen** aktiviert haben, wird hier die vom Eintrags-Editor bekannte Oberfläche angezeigt.

Beachten Sie, dass der WYSIWYG-Editor intern bereits HTML-Code speichert. Würden Sie in ein WYSIWYG-Editor-Fenster HTML-Code eintragen, würde dies später im Frontend „gedoppelt“ angezeigt werden. Für eigenes HTML oder JavaScript gilt daher, entweder den WYSIWYG-Editor vorübergehend zu deaktivieren oder im Quelltext-Modus des Editors zu arbeiten.

Textformatierung(en) durchführen

Die Ausgabe des Seitenleisten-Plugins wird auch stark von der Option **Textformatierung(en) durchführen** beeinflusst. Ist diese Option aktiviert, wird alles, was Sie im HTML-Klotz eingetragen haben, mit denselben Textformatierungs-Plugins (siehe Seite 237) verarbeitet, wie Sie es von ihren Artikeln kennen.

Besonders wenn Sie JavaScript oder eigenen HTML-Code in einem HTML-Klotz einbinden wollen, müssen Sie unbedingt darauf achten, diese Option auf `Nein` zu setzen. Andernfalls könnten destruktive Textformatierungen auf Ihren Code angewendet werden, was dann zu einer Fehldarstellung führen würde.

Wo soll dieses Plugin angezeigt werden

Mit diesem Auswahlfeld legen Sie fest, an welcher Stelle ein HTML-Klotz angezeigt werden soll: **Überall**, **Nur Artikelübersicht** oder **Nur Artikel-Detailansicht**.

Speziellere Möglichkeiten, ein Seitenleisten-Plugin vor bestimmten Benutzern zu verstecken, bietet das Plugin *Seitenleisten verstecken* (siehe Seite 298).

5.2.6 Kommentare serendipity_plugin_comments

Wenn Ihre Besucher Kommentare zu einem Beitrag hinterlassen, können Sie als Redakteur diese relativ leicht über den Backend-Menüpunkt **Kommentare** einsehen oder werden sogar per E-Mail darüber benachrichtigt. Normale Besucher haben diese Möglichkeit nicht und

können daher entweder den RSS-Feed der letzten Kommentare abonnieren oder müssten jeden Blog-Artikel individuell aufrufen, um sich über neue Kommentare zu informieren.

Daher bietet das Seitenleisten-Plugin *Kommentare* eine einfache Möglichkeit, die letzten Kommentare aller Artikel anzuzeigen und so für den Besucher übersichtlich einzubinden.

Folgende Konfigurationsoptionen bietet das Plugin, abgesehen vom Seitenleisten-typischen Titel:

Zeilenumbruch
: In der Seitenleiste steht meist nur begrenzt Platz zur Darstellung von Text zur Verfügung. Gerade Kommentare können jedoch die zur Verfügung stehende Breite übermäßig strapazieren. Daher bietet die Option **Zeilenumbruch** die Möglichkeit, den Platzbedarf einzuschränken, indem nach einer festgelegten Anzahl von Buchstaben ein harter Zeilenumbruch eingefügt wird.

  Der Zeilenumbruch wird dabei auch Wörter mittendrin trennen, um das Limit nicht zu überschreiten. Dies ist notwendig, damit überlange Wörter Ihrer Besucher (wie z. B. „Coooooooooooooooooooool!“) nicht das Layout Ihrer Seite sprengen.

  Leider bereitet der Zeilenumbruch gerade dann Probleme, wenn Besucher lange Links posten, die dann auseinandergebrochen werden. Sollte dies bei Ihnen häufig vorkommen, können Sie die Buchstabenanzahl des Zeilenumbruchs auf eine hohe Zahl wie 999 setzen und dann den Zeilenumbruch dem Browser überlassen. Dabei empfiehlt es sich dann, mittels CSS mögliche Layoutprobleme zu verhindern. Fügen Sie dazu die Zeile

  ```
  .plugin_comment_body { overflow: scroll }
  ```

  am Ende der Datei `style.css` Ihres gewählten Templates hinzu (siehe Kapitel 470). Dies wird den Browser dazu anweisen, überlange Wörter durch Scrollbalken innerhalb der Seitenleiste anzuzeigen und nicht die ganze Seitenleiste an die Wortbreite anzupassen.

Zeichen pro Kommentar
: Um nicht zu viel Platz zu beanspruchen, werden immer nur die ersten 120 Zeichen eines Kommentars in der Seitenleiste angezeigt. Dies reicht meistens, um sich als Besucher einen Überblick zu verschaffen und bei Interesse den vollständigen Kommentar in der Artikel-Detailseite zu lesen. Im Eingabefeld **Zeichen pro Kommentar** können Sie diese Zeichenanzahl vorgeben.

Anzahl an Kommentaren
: Üblicherweise zeigt das Plugin die letzten 15 Kommentare. Diese Anzahl können Sie mit dem Eingabefeld **Anzahl an Kommentaren** verändern.

Datumsformat
: Zu jedem Kommentar wird das Erstellungsdatum mit angezeigt. Wie dieses formatiert wird, tragen Sie im Feld **Datumsformat** ein. Die zur Verfügung stehenden Variablen finden Sie auf `http://de3.php.net/strftime`.

Typ
: Das Seitenleisten-Plugin kann sowohl Kommentare als auch Trackbacks anzeigen. Über das Auswahlfeld **Typ** bestimmen Sie diese Art der Anzeige. So können Sie beispielsweise auch zwei Kommentar-Seitenleisten-Plugins installieren: eines, das die Trackbacks anzeigt, und eines, das die Kommentare anzeigt.

Kommentatoren URL anzeigen bei...
: Über dieses Auswahlfeld können Sie bestimmen, ob die Homepage der Person, die den Kommentar hinterlassen hat, in der Seitenleiste angezeigt werden soll. Sie können wählen, ob die Homepage niemals, nur bei Kommentaren, nur bei Trackbacks oder bei beidem angezeigt wird.

5.2.7 Top Exits serendipity_topexits_plugin

Mittels des Ereignis-Plugins *Links verfolgen* (siehe Seite 258) können Sie dafür sorgen, dass alle Links in Beiträgen speziell maskiert werden. Wenn ein Besucher dann auf einen solchen Link klickt, kann Serendipity dies statistisch erfassen.

Einen Link, der auf eine fremde Seite führt und damit das Blog verlässt, nennt man *Exit-Link*. Diese formatiert Serendipity dann wie `http://www.example.com/serendipity/exit.php?url_id=17&entry_`

Anhand der Statistik können Sie also leicht herausfinden, welche Links besonders beliebt bei Ihren Besuchern sind. Eben diese Statistik der beliebtesten Links kann das Seitenleisten-Plugin *Top Exits* anzeigen.

Bitte beachten Sie, dass, falls Sie das oben genannte Ereignis-Plugin nicht installiert haben, Serendipity keine Statistik führt und daher das Plugin auch keine Daten anzeigen kann.

Die Konfigurationsoptionen sind:

Wie viele Elemente sollen angezeigt werden
: Üblicherweise werden nur die zehn beliebtesten Links in der Seitenleiste angezeigt. Mit dieser Option können Sie die Vorgabe verändern.

Top Exits/Referrers als Link anzeigen
: Wenn Sie diese Option auf **Ja** setzen, werden die beliebtesten Links in der Seitenleiste so angezeigt, dass die Besucher direkt darauf klicken können, um zu der jeweiligen genannten Seite zu gelangen. Ist die Option auf **Nein** gestellt, werden die Seiten nur textlich aufgeführt, und ein Besucher müsste die URL manuell eingeben.

  Auch in der globalen Serendipity-Konfiguration (siehe Seite 168) können Sie dieses Verhalten beeinflussen – wenn diese Vorauswahl genutzt werden soll, müssen Sie **Standard** als Option einstellen.

  Diese Option ist sehr hilfreich, weil Top-Exits seltener von Spammern genutzt werden können – denn schließlich sind die hinterlegten Links meist nur von Redakteuren

in das System eingetragen. Trickreiche Spammer können jedoch die Statistik-Routine von Serendipity missbrauchen, so dass ihre Seiten auch in die Statistik aufgenommen werden, ohne dass Sie in einem Ihrer Artikel darauf verweisen.

Kalenderintervall
Meist macht die Statistik der beliebtesten Links nur Sinn, wenn sie sich auf einen gewissen Zeitraum beschränkt. Besucher interessiert womöglich nicht, welche Seiten seit Eröffnung Ihres Blogs am häufigsten besucht wurden, sondern eher, welche Links der letzten Zeit interessant waren. Dafür dient die Option **Kalenderintervall**. Hier tragen Sie den Zeitraum (in Tagen) ein, der vom aktuellen Besuchsdatum aus gerechnet für die Datenbasis der Links herangezogen wird.

5.2.8 Top Referrer serendipity_topreferrers_plugin

Ähnlich wie das *Top Exits*-Plugin zeigt Ihnen das *Top Referrer*-Plugin eine Liste an URLs an. Diesmal sind es jedoch nicht die Links, die Ihre Besucher angeklickt haben, sondern die Webseiten, von denen aus Besucher zu Ihrem Blog gekommen sind.

Um dies auszuwerten, bietet jeder Browser die Möglichkeit, die zuletzt besuchte Seite im Webseitenaufruf mit zu übertragen. Manche Browser oder auch Proxies Ihrer Besucher verschleiern oder entfernen diese Angabe aus Datenschutzgründen. Genauso ist es für Spammer möglich, beliebige URLs als Quellseite vorzutäuschen, daher hat in heutigen Zeiten diese Statistik oft nur noch ideellen Wert.

Für die Darstellung der Referrer gelten dieselben Optionen wie für das Top-Exits-Plugin, beziehen sich dabei jedoch auf die Ausgabe der verweisenden URLs und nicht der beliebtesten Links.

5.2.9 Shoutbox serendipity_plugin_shoutbox

Besucher können zu Artikeln Ihres Blogs relativ leicht kommentieren. Wenn sie aber eher abstrakt und Artikel-ungebunden ihre Meinung über Ihr Blog äußern wollen, ist dies meist nur über ein Kontaktformular oder Gästebuch möglich.

Eine einfache Alternative dazu bietet das *Shoutbox*[3]-Seitenleisten-Plugin. Es zeigt in der Seitenleiste eine Eingabebox an, in der Besucher direkt Text eingeben können, der danach unterhalb der Shoutbox für alle Besucher angezeigt wird.

Als Administrator können Sie direkt im Frontend unpassende Kommentare löschen. Die Shoutbox zeigt kein Archiv von vergangenen Einträgen an, sondern immer nur eine konfi-

[3] Eine Shoutbox ist die sprichwörtliche Seifenkiste, auf die man sich gerne einmal stellt, wenn man seine Meinung in die Welt hinausposaunen möchte.

gurierbare Anzahl von Kommentaren Ihrer Benutzer. Daher kann dies auch oft als eine Art Echtzeit-Chatbox zweckentfremdet werden.

Beachten Sie, dass dieses Plugin keinen besonderen Schutz gegen Spam bietet.

Für die Konfigurationsoptionen gelten dieselben Möglichkeiten wie bei dem *Kommentar*-Plugin (Seite 217).

Das Plugin erstellt die Datenbanktabelle `serendipity_shoutbox` mit den Feldern `id` (fortlaufender Primärschlüssel), `timestamp` (Datum des Kommentars), `ip` (IP des Kommentators) und `body` (Text).

5.2.10 Links des Artikels serendipity_plugin_entrylinks

Das Plugin *Links des Artikels* wird nur auf Artikel-Detailseiten angezeigt und bleibt in der Übersichtsseite unsichtbar. Ruft man eine Detailseite auf, wird in dieser Box eine Übersicht aller im Artikel enthaltenen Links angezeigt.

Die relevanten Konfigurationsoptionen sind:

Top Exits
: Ist diese Option aktiviert, werden in dem Seitenleisten-Plugin alle Links des Beitrags angezeigt.

Top Referrer
: Wenn Sie diese Option aktivieren, werden alle auf die Artikel-Detailseite verweisenden Webseiten aufgeführt.

Anzahl eingehender Links
: Mit dieser Option bestimmen Sie, wie viele Links der beiden Darstellungsmöglichkeiten (Exits, Referrer) jeweils angezeigt werden sollen.

Reihenfolge eingehender Links
: Die verwendeten Links (Exits, Referrer) können entweder nach **Häufigkeit** oder **Datum** sortiert werden. Dies zeigt also entweder die neuesten oder die populärsten Links am Anfang.

5.2.11 Geschichte serendipity_plugin_history

Mit dem Seitenleisten-Plugin *Geschichte* können Sie im Frontend auf ältere Artikel Ihres Blogs hinweisen, indem Sie eine (von mehreren Optionen abhängige) Liste darstellen.

Intro, Outro
: Sie können vor (*Intro*, Einführung) und nach (*Outro*, Nachwort) der Auflistung der älteren Artikel einen beliebigen Text in die Ausgabe des Seitenleisten-Plugins aufnehmen, um Ihren Besuchern zu erklären, warum Sie eine Liste älterer Einträge aufführen. So könnten Sie als Intro z. B. den Text *„In einer Galaxie, weit weit entfernt . . . “* den Artikeln voranstellen. Obwohl hierfür nur eine kleine Eingabezeile vorgesehen ist, können Sie hier beliebig langen Text (auch mit HTML-Formatierung) eintragen.

Überschriftenlänge
: Wenn Ihre Artikel öfter besonders lange Überschriften enthalten, würden diese den Rahmen möglicherweise sprengen. Für diesen Fall können Sie die Zeichenlänge eines Artikeltitels einschränken.

Vorgefertigter Zeitrahmen
: Standardmäßig zeigt das Plugin Einträge an, die genau vor einem Jahr geschrieben wurden. Ist kein solcher Artikel vorhanden, zeigt das Plugin keinen Artikel an.

  Wenn Sie einen anderen Zeitraum wünschen, können Sie das Auswahlfeld **Vorgefertigter Zeitrahmen** auf **Anderer** stellen und die darunterliegenden Optionsfelder anpassen.

Mindestalter
: Wenn Sie einen eigenen vorgefertigten Zeitrahmen eingestellt haben, tragen Sie in dieses Feld ein, wie alt ein Artikel mindestens sein muss (in Tagen gemessen), um in der Ausgabe des Plugins berücksichtigt zu werden.

Höchstalter
: Analog zu dem Mindestalter eines Artikels können Sie hier das maximale Alter eines Eintrages angeben.

  Wenn Sie als Mindestalter beispielsweise `12` eintragen und als Höchstalter `20`, würden Sie am 30.06.2007 alle Artikel sehen, die zwischen dem 18.06.2007 und dem 10.06.2007 verfasst wurden.

Anzahl
: Beschränkt die Anzahl der Einträge, die in der Seitenleiste angegeben werden. Wenn Sie mehr als die hier festgelegte Anzahl an Einträgen im definierten Zeitraum erstellt haben, werden also nur die ersten Artikel davon angezeigt.

Ganze Einträge
: Standardmäßig zeigt das Plugin nur die Überschriften der Artikel des gewählten Zeitraums an. Wenn Sie diese Option auf **true** stellen, wird auch der Beitragstext mit angezeigt.

Datum anzeigen
: Wenn Sie diese Option auf **true** stellen, wird auch das Datum eines Artikels angezeigt.

Show author's name
Den Autorennamen eines Artikels können Sie anzeigen, indem Sie die Option **Show author's name** auf **true** stellen.

Datumsformat
Sofern Sie die Option **Datum anzeigen** aktiviert haben, können Sie hier das Datumsformat einstellen. Die zur Verfügung stehenden Variablen finden Sie auf `http://de3.php.net/strftime`.

5.3 Auswahl externer Plugins

Über die mitgelieferten Plugins hinaus finden Sie auch eine große Auswahl an Plugins über `http://spartacus.s9y.org/`. Eine Auswahl an häufig genutzten Seitenleisten-Plugins finden Sie auf den folgenden Seiten.

5.3.1 Externe PHP-Anwendung serendipity_plugin_externalphp

Oft kann es vorkommen, dass Sie eine bestehende kleine PHP-Anwendung in Ihrer Seitenleiste einbinden wollen, also beispielsweise einen Shoutbox-Ersatz, ein Zufallszitat oder auch eigenen Counter-Code.

Grundsätzlich ist es empfehlenswert, solche Anwendungen mit einem eigenen Seitenleisten-Plugin einzubinden (siehe Seite 642 ff.).

Für Fälle, in denen Ihnen das zu kompliziert erscheint, können Sie das Plugin *Externe PHP-Anwendung* installieren. Dieses ermöglicht Ihnen, mit einem PHP-`include`-Befehl ein anderes PHP-Script auf Ihrem Server aufzurufen.

Bitte beachten Sie, dass dieses Plugin ein großes Sicherheitsrisiko darstellt. Daher dürfen nur Administratoren (entsprechend dem Benutzerrang eines Redakteurs) ein derartiges Plugin einbinden und konfigurieren.

Eine Einbindung mittels dieses Plugins muss nicht unbedingt bei jedem beliebigen PHP-Script funktionieren. Wenn das fremde PHP-Script z. B. Datenbankverbindungen kappt, globale Namensräume überschreibt oder mit von Serendipity genutzten Funktionsnamen kollidiert, kann es zu Problemen kommen. In einem solchen Fall müssen Sie sich leider an den Autor des fremden Scripts wenden.

Für Programmierer gilt: Das Plugin setzt den `include`-Befehl innerhalb einer Klassenmethode (`generate_content()`) ein. Globale Variablen können daher nicht auf die übliche Weise angesprochen werden, sondern müssen entweder vorher mittels `global $variable` in den Namensraum importiert oder zentral via `$GLOBALS['variable']` angesprochen werden.

In der Konfiguration des Plugins tragen Sie den vollständigen Pfad zu einer Datei ein. Benutzen Sie dafür die Verzeichnisstruktur der Datei auf dem Server, also beispielsweise `/var/www/example.com/Script.php`. Wenn Ihr Server die PHP-Option `allow_url_fopen` aktiviert hat, können Sie anstelle eines Dateisystem-Pfades auch eine URL eintragen. Wenn Sie eine URL einbinden, wird lediglich die Ausgabe dieser URL im Plugin dargestellt – das Script muss also für eine derartige Verwendung vorgesehen sein.

Die Konfigurationsoption **Textformatierung(en) durchführen** gibt an, ob die Ausgabe des fremden PHP-Scripts an Textformatierungs-Plugins von Serendipity weitergereicht werden soll, um die Ausgabe genauso zu formatieren wie einen redaktionellen Artikel. Die Textformatierungen richten sich dabei nach den von Ihnen installierten Plugins, also z. B. der Umwandlung von Smiley-Zeichen in echte Grafiken und weitere (siehe Seite 237).

5.3.2 Flickr badge, FLICKR Sidebar-Plugin serendipity_plugin_flickrbadge, serendipity_plugin_flickr

Der Webservice `http://www.flickr.com/` ermöglicht es seinen Benutzern, eigene Bildergalerien komfortabel zu verwalten. Zudem bietet Flickr dabei zahlreiche Einbindungsmöglichkeiten über eine Schnittstelle an.

Serendipity besitzt zwei Seitenleisten-Plugins, mit denen man seine eigenen Bilder bei Flickr in der Seitenleiste anzeigen kann. Dafür muss Ihr Webserver natürlich ausgehende Verbindungen zum Flickr-Webserver erlauben, darf also nicht durch eine Firewall blockiert werden.

Flickr badge ist ein aktuelleres Plugin, das PHP5 auf Ihrem Webserver benötigt und sonst keine weiteren Voraussetzungen besitzt.

Das *FLICKR Sidebar-Plugin* ist auch auf PHP4-Webservern benutzbar, dafür müssen Sie aber selbständig eine fremde Bibliothek herunterladen. Diese Bibliothek nennt sich `PEAR::Flickr_API`[4]. Laden Sie sich das ZIP-Paket der Bibliothek herunter, entpacken Sie die Datei `API.php` aus dem Unterverzeichnis `pear/Flickr/` und laden Sie sie in Ihr Serendipity-Verzeichnis `bundled-libs` hoch. Danach muss sich also auf Ihrem Server eine Datei `bundled-libs/Flickr/API.ph` in Ihrem Serendipity-Stammverzeichnis befinden. Diese wird dann von Ihrem Plugin später eingebunden und verwendet, um mit Flickr zusammenzuarbeiten. Wenn Ihnen diese Datei fehlt oder Sie sie in ein falsches Verzeichnis hochgeladen haben, erhalten Sie eine Fehlermeldung in der Seitenleiste, und das Plugin wird nicht funktionieren.

[4] `http://code.iamcal.com/php/flickr/readme.htm`

In den Konfigurationsoptionen beider Plugins müssen Sie den Flickr-API-Schlüssel eintragen. Wenn Sie bei Flickr eingeloggt sind, können Sie diesen unter `http://www.flickr.com/services/api/keys/` einfach beantragen und später in der Konfigurationsoberfläche eintragen.

Das *Flickr badge*-Plugin ermöglicht weiterhin, die Anzahl der Bilder insgesamt und die Anzahl der dargestellten Bilder pro Zeile festzulegen. Dabei werden die chronologisch aktuellsten Bilder ausgewählt. Das Plugin erstellt selbständig einen Zwischenspeicher (*Cache*) der letzten Bilder, der stündlich aktualisiert wird.

Im *FLICKR Sidebar*-Plugin können Sie einige weiterreichende Optionen einstellen, die die Größe und das Format der Vorschaubilder regeln sowie die Möglichkeit geben, die Dauer der Zwischenspeicherung selbst zu bestimmen.

Welches Plugin Sie wählen, ist hauptsächlich von den Rahmenbedingungen Ihres Webservers abhängig: Wer über PHP5 verfügt und keine externen Bibliotheken herunterladen möchte, sollte das *Flickr badge*-Plugin verwenden. Wer etwas mehr Flexibilität bei der Bildgröße benötigt, über kein PHP5 verfügt oder den höheren Installationsaufwand nicht scheut, dem sei das *FLICKR Sidebar*-Plugin ans Herz gelegt.

Zuletzt bietet Flickr aber auch noch einen RSS-Feed oder einen JavaScript-Schnipsel an, die Sie auch mittels des Plugins *Fremder RSS/OPML-Blogroll Feed* (Seite 210) oder des HTML-Klotzes (Seite 216) anzeigen können. Weitere Hinweise zu den Einbindungsmöglichkeiten finden Sie auf der Flickr-Seite unter `http://www.flickr.com/services/feeds/`.

5.3.3 Social Bookmarks-Plugin serendipity_plugin_socialbookmarks

Das *Social Bookmarks*-Plugin ermöglicht es Ihnen, Ihre bei externen Anbietern geführten Lesezeichen in Ihrem Blog darzustellen. Es gibt zahlreiche Webservices (del.icio.us, ma.gnolia, furl.net, linkroll, Mister Wong), bei denen eine Community untereinander ihre Lieblings-Internetseiten oder Neuigkeiten tauschen und veröffentlichen kann. Häufig gibt es für Ihren Browser eigenständige Symbolleisten oder Extensions, mit denen Sie komfortabel einen Link zu Ihrem Profil bei einem derartigen Dienstleister hinzufügen können. Der Vorteil (abgesehen von der Community) dabei ist, dass Sie Ihre Lesezeichen so einfach von einem beliebigen Rechner im Internet aufrufen können.

Wenn Sie die Lesezeichen in Ihrem Blog einbinden, können Sie Ihre Besucher darüber informieren, welche Seiten Sie gerade interessant finden.

Die meisten Webservices bieten eigenständige JavaScripts an, die Sie auf Ihrer Webseite einbinden können. Da dies bei der Benutzung mehrerer Services leicht unübersichtlich werden kann, nimmt Ihnen das Serendipity-Plugin einiges an Arbeit ab.

Das Plugin können Sie beliebig oft installieren, und es muss jeweils immer für einen gewünschten Webservice konfiguriert werden. Damit das Plugin die Daten der Webservices per RSS-Feed einlesen kann, darf Ihr Webserver nicht von einer Firewall bei ausgehenden Verbindungen blockiert werden.

Folgende Konfigurationsoptionen stehen zur Auswahl:

Service
: In dem Auswahlfeld *Service* müssen Sie auswählen, welchen Webservice Sie ansprechen wollen. Vom gewählten Service sind einige der weiteren Konfigurationsoptionen abhängig.

Benutzername
: Bei jedem der verfügbaren Webservices haben Sie einen eindeutigen Benutzernamen ausgewählt oder zugewiesen bekommen. Diesen müssen Sie im Feld **Benutzername** eintragen, um die Links des gewählten Benutzers anzuzeigen.

  Selbstverständlich müssen Sie hier nicht unbedingt Ihren eigenen Benutzernamen eintragen. Wenn Sie die Links einer anderen Person besonders wertvoll finden, können Sie auch deren Lesezeichen anzeigen.

Anzahl der darzustellenden Links
: Das Plugin kann höchstens die Anzahl der Lesezeichen einbinden, die im RSS-Feed des jeweiligen Benutzers vorhanden sind. Abhängig vom Service sind dies meist die letzten 30 Links. Im Feld **Anzahl der darzustellenden Links** können Sie die Anzahl der ausgegebenen Links daher weiter einschränken. Ein höherer Wert ist prinzipiell nutzlos und kann nicht mehr Links als die verfügbare Anzahl anzeigen.

Wann soll der Feed aktualisiert werden
: Damit das Plugin nicht bei jedem Seitenaufruf des Besuchers neu ausgeführt werden muss, können die Ausgaben zwischengespeichert werden. Im Feld *Wann soll der Feed aktualisiert werden* tragen Sie die Anzahl der Stunden ein, für die ein Feed zwischengespeichert werden kann. Erst nach Ablauf dieser Zeit wird der Webservice erneut angefragt.

Zeig „mehr“-Verknüpfung
: Da der RSS-Feed meistens nicht alle verfügbaren Links beinhaltet, können Sie Ihren Besuchern einen Link zu den weiteren Lesezeichen des dargestellten Profils anbieten. Wenn Sie diese Option aktivieren, wird dieser „mehr“-Link eingebettet.

Tags anzeigen
: Jedem Lesezeichen können Sie bei der Erfassung auf den Seiten des Webservices eine Liste von Stichwörtern zuweisen, die das Lesezeichen identifizieren. Mit Hilfe dieser *Tags* ist es leicht möglich, andere Lesezeichen weiterer Benutzer anzuzeigen, die ähnliche Stichwörter benutzt haben.

  Wenn Sie die Option **Tags anzeigen** aktivieren, werden Ihnen die zugewiesenen Schlüsselwörter jedes Lesezeichens mit angezeigt. Andernfalls würden Sie nur den Link selbst sehen.

Feed-Einstellungen
: Üblicherweise wertet das Plugin den RSS-Feed aus, der die aktuellsten Lesezeichen enthält. Sie können aber auch eine andere Quelle benutzen. **Neueste Lesezeichen aller**

Benutzer zeigt die aktuellsten Lesezeichen des Webservices an, die unabhängig vom Benutzer gerade erfasst werden. Bei fast allen Webservices werden mehrere hundert Lesezeichen pro Minute erfasst, daher hat diese Option eher wenig Informationswert.

Populärste Lesezeichen der letzten 24 Stunden ist daher meist sinnvoller, wenn Sie neue, beliebte Lesezeichen aller Benutzer des Webservices anzeigen wollen.

Speziell für den del.icio.us-Webservice gibt es die Option **Meine Tagcloud**. Dabei wird anstelle der Liste der Lesezeichen nur eine Liste aller vergebenen Schlagwörter angezeigt. Die Schlagwörter sind dabei in ihrer Schriftgröße proportional zur Häufigkeit, mit der Sie sie eingesetzt haben. Besonders groß formatierte Schlagwörter vergeben Sie also sehr häufig, die sehr klein geschriebenen Schlagwörter nur selten.

Vorschaugrafiken anzeigen (ma.gnolia)
Der Webservice ma.gnolia bietet eine kleine Vorschaugrafik der Webseite an, die Sie als Lesezeichen gespeichert haben. Wenn Sie die Option **Vorschaugrafiken anzeigen** aktivieren, wird diese Grafik anstelle des Lesezeichen-Texts angezeigt, um dem Besucher bereits einen optischen Vorgeschmack auf die Seite geben zu können.

Zusätzliche Parameter für die Einstellung „Meine Tagcloud“
Wenn Sie für die **Feed-Einstellungen** die Option **Meine Tagcloud** gewählt haben, können Sie einige Parameter für die Formatierung der Schlagwortliste vornehmen. Eine Auflistung aller verfügbaren Parameter finden Sie unter `http://del.icio.us/help/tagrolls`.

5.3.4 AdSense serendipity_plugin_google_adsense

Google AdSense[5] ist ein Angebot von Google, mit dem Sie Werbeanzeigen von Googles Werbepartnern auf Ihrer eigenen Seite einbinden können.

Dabei ruft der Besucher (also NICHT Ihr Server selbst!) ein JavaScript auf, das eine Liste von Werbeanzeigen (meist abhängig vom Inhalt Ihres Blogs) darstellt.

Für jeden Klick Ihrer Besucher auf eine solche Werbung erhalten Sie von Google etwas Geld, das Ihnen ab Erreichen eines Mindestbetrags überwiesen wird.

Google wählt dabei die Anzeigen der Werbepartner nach meist seriösen Kriterien aus. Sie sollten sich jedoch gut überlegen, ob Sie auf Ihrem Blog tatsächlich Werbung einbinden wollen. Bei deutschen Blogs führt dies zum einen oft dazu, dass Ihr Blog dadurch als *kommerzielles Angebot* eingestuft wird und Sie unter diesem Gesichtspunkt für die Inhalte Ihres Blogs verantwortlich sind. Etwaige Verstöße gegen das Urheberrecht werden bei kommerziellen Angeboten weitaus härter geahndet, als es bei privaten Blogs der Fall ist.

Wenn Ihr Blog tatsächlich kommerzieller Natur ist, ist auch hier der Einsatz von Werbung mit Vorsicht zu genießen. Da Sie nur eingeschränkte Kontrolle über die angezeigte Werbung ha-

[5] `http://www.google.com/adsense`

ben, könnte es leicht passieren, dass Werbung der Konkurrenz auf Ihren Seiten eingebunden wird.

Für die Einbindung von Werbung spricht natürlich die finanzielle Verdienstmöglichkeit – wägen Sie jedoch stets die Nachteile ab und informieren Sie sich am besten vorher ausführlich über Google AdSense.

Um das JavaScript von Google AdSense einzubinden, bietet Google selbst bereits ein Tool zur Erzeugung des Codes an. Diesen Code könnten Sie dann per Kopieren & Einfügen in einen HTML-Klotz (siehe Seite 216) übernehmen. Der Vorteil des Google-Tools ist, dass Sie dort die Farbe und andere Optionen zum Aussehen der Werbung einstellen können.

Das *Google AdSense*-Seitenleisten-Plugin vereinfacht die Einbindung etwas, indem Sie keinen Code selbst einfügen müssen. Stattdessen tragen Sie bei der Konfiguration des Plugins lediglich Ihre AdSense-ID ein und wählen das grobe Anzeigenformat sowie die Anzahl der gewünschten Werbeanzeigen. Leider ist es aufgrund der Lizenzbedingungen von Google nicht erlaubt, dass das Plugin Ihnen dieselben Formatierungsmöglichkeiten wie das Google-Tool anbietet.

Auch vom *Google AdSense*-Plugin können Sie mehrere verschiedenartig konfigurierte Plugins installieren und frei in der Seitenleiste verteilen. Bei der Verwendung mehrerer Werbeblöcke können Sie die Konfigurationsoption **Kanal (Channel)** dafür benutzen, unterschiedlich bei Google konfigurierte Werbe-Themenkanäle anzuzeigen. So könnten Sie in der linken Seitenleiste nur Werbung zu Lifestyle-Produkten anzeigen und in der rechten Seitenleiste ein anders konfiguriertes Plugin nur mit Technikprodukten. Gehen Sie jedoch sparsam mit Werbeblöcken um, da Sie sonst Ihre Besucher vergraulen.

5.3.5 Show Entries in sidebar serendipity_plugin_showentries

Bereits mittels der Plugins *Geschichte* (Seite 221) und *Aktuelle Einträge* (Seite 214) können Sie in der Seitenleiste auf ältere Artikel verweisen.

Mit beiden Plugins ist es jedoch nicht möglich, individuelle Formatierungen durchzuführen oder die Artikel komplett anzuzeigen. Hier schafft das Plugin *Show Entries in sidebar* Abhilfe.

Es ist besonders dann sinnvoll, wenn Sie eine Artikelliste nach eigenen Vorgaben gestalten wollen oder wenn Sie kurze Einträge einer speziellen Kategorie anzeigen wollen. Stellen Sie sich vor, Sie schreiben Artikel in der Kategorie *Meine Einkäufe* und fassen dort in kurzen Sätzen zusammen, welche Lebensmittel Sie heute gekauft haben. In der normalen Artikelübersicht würden solche Einträge womöglich ziemlich leicht übersehen werden, daher möchten Sie diese Einkäufe in der Seitenleiste einbinden.

Ein weiterer Einsatzzweck des Plugins wird in der Plugin-Beschreibung erwähnt: *Good for Sidebar Moblogging*. Moblogging bezeichnet dabei die Möglichkeit, einen Blog-Artikel via E-Mail zu verfassen und unterwegs an das eigene Blog zu schicken. Das Blog kann dann

mittels Popfetcher-Plugin (siehe Seite 343) die E-Mails von einem Server abrufen und automatisch importieren und erspart es Ihnen dadurch, einen Artikel mittels eines Browsers erstellen zu müssen. Derart erstellte Artikel könnten Sie mit dem hier beschriebenen Plugin **Show Entries in sidebar** komfortabel und abgesetzt in der Seitenleiste anzeigen.

Auch dieses Plugin ist beliebig oft installierbar und kann daher für mehrere Seitenleisten-Positionen eingebunden werden.

Das Plugin formatiert die Ausgabe mittels Smarty-Template-Datei `plugin_showentries.tpl`. In dieser Datei können Sie auf alle Eintrags-Variablen (`array $entries` → `array $entry.*`) zugreifen und so mit HTML beliebige Layouts umsetzen. Details zu Smarty-Templates finden Sie ab Seite 480.

Das Plugin bietet folgende Konfigurationsoptionen:

Show how many entries
: Diese Einstellung legt fest, wie viele Artikel in der Seitenleiste angezeigt werden sollen.

Hide entries already displayed on frontpage
: Je nachdem, welche Einträge das Plugin darstellt, kann es passieren, dass diese Einträge auch bereits in der Artikel-Übersichtsseite aufgeführt sind.

  Wenn Sie die Option **Hide entries already displayed on frontpage** aktivieren, werden solche Einträge nicht nochmal in der Seitenleiste angezeigt.

Enter category ID to show
: Im Gegensatz zu vielen anderen Plugins können Sie die Quellkategorie für die darzustellenden Einträge nicht aus einem Auswahlfeld wählen, sondern müssen die ID/Nummer einer Kategorie manuell in ein Feld eingeben. Für das Plugin ist diese Eingabemethode um einiges flexibler, da Sie mehrere Kategorie-IDs einfach mittels Semikolon trennen können, um so die Einträge mehrerer Kategorien einzubinden.

  Die ID einer Kategorie können Sie über die Administrationsoberfläche **Kategorien** ermitteln (siehe Seite 124).

Show entry title, Show extended entry
: Mit diesen beiden Optionen stellen Sie ein, ob der Titel und/oder der erweiterte Eintrag eines Artikels in der Seitenleiste angezeigt werden sollen. Das Smarty-Template `plugin_showentries.tpl` wertet diese Optionen (`bool $showext`, `bool $showtitle`) aus, daher können Sie eine Einstellung auch direkt in der Smarty-Template-Datei vornehmen.

5.3.6 Unified Sidebar Image Display serendipity_plugin_imagesidebar

Zahlreiche Plugins im Serendipity-Plugin-Archiv ermöglichen die Einbindung von Bildern in der Seitenleiste.

Das Plugin *Unified Sidebar Image Display* versucht diese verschiedenen Plugins unter einem Dach zu vereinen. Anstatt für jede einzelne Galerie-Software ein eigenständiges Plugin zu installieren, können Sie mit diesem Plugin direkt mehrere mögliche Bildquellen auswählen: *Menalto Gallery*[6], *Coppermine Gallery*[7], *Zooomr*[8], aber auch die integrierte *Serendipity-Mediendatenbank*.

Bitte beachten Sie, dass für die gerade genannten Galerie-Anwendungen auch, wie erwähnt, eigenständige Plugins bestehen. Diese sind jedoch älter und nur noch aus historischen Gründen in der Plugin-Datenbank. Ein Umstieg auf das einheitliche *Unified Sidebar Image Display*-Plugin ist daher immer zu empfehlen.

Nachdem das Plugin installiert wurde, müssen Sie in der Konfiguration erst festlegen, welche Software als Quelle für Ihre Bilder herangezogen werden soll. Abhängig von der gewählten Option werden dann weitere Konfigurationsmöglichkeiten angezeigt, sobald Sie auf den Button **Speichern** klicken.

Menalto Gallery Url

Die *Menalto Gallery* bietet in ihrer Installation bereits ein kleines PHP-Script namens `block-random.php` an, mit dem Sie ein Bild der Galerie anzeigen können. Je nach Version der Menalto Gallery (1.x oder 2.x) heißt dieses Script anders, daher müssen Sie vor allem darauf achten, dem Serendipity-Plugin die richtige Gallery-Version mitzuteilen.

Wenn Sie die Menalto Gallery in ein Unterverzeichnis von Serendipity installiert haben, kann es möglicherweise zu Problemen mit der URL-Umformung von Serendipity kommen. Wenn Sie die Option `mod_rewrite` in der Serendipity-Konfiguration gewählt haben, sollten Sie eine `.htaccess`-Datei im Gallery-Unterverzeichnis erstellen. Diese sollte den Inhalt

```
RewriteEngine Off
```

haben. Mit einer solchen Datei können Sie Serendipity dazu bringen, seine eigene URL-Formung im Gallery-Unterverzeichnis nicht anzuwenden. Sicherlich sinnvoller ist in den meisten Fällen jedoch, wenn Sie Gallery in ein eigenständiges Verzeichnis oberhalb des Serendipity-Verzeichnisses installieren.

URL oder Pfad zur Gallery-Installation
Hier müssen Sie die vollständige URL eintragen, die zu Ihrer Gallery-Installation führt.

[6] `http://gallery.menalto.com`
[7] `http://coppermine-gallery.net`
[8] `http://www.zooomr.com`

Das Plugin fügt hier standardmäßig eine URL zum Blog ein, die Ihnen aber nur als Anhaltspunkt dienen soll.

Anzahl der Zufallsfotos
Mit der Option **Anzahl der Zufallsfotos** legen Sie fest, wie viele Bilder Ihrer Galerie angezeigt werden sollen. Die Bilder werden dabei vom Gallery-Script zufällig ausgewählt.

Rotate image time
Jede Darstellung eines Zufallsbildes aus der Galerie wird für einen gewissen Zeitraum zwischengespeichert, damit in diesem Zeitraum alle Besucher dasselbe Zufallsbild angezeigt bekommen. Wie lange dieser Zeitraum (in Minuten) ist, stellen Sie über die Option **Rotate image time** ein. Die Zahl `30` würde also bewirken, dass jeweils eine halbe Stunde lang dasselbe Zufallsbild angezeigt wird.

Welche Version der Gallery-Software benutzen Sie?
In diesem Auswahlfeld müssen Sie die Versionsnummer der installierten Gallery-Anwendung eintragen, da sich die Art der Einbindung danach richtet.

Wenn Sie über eine Version höher als 2.0 verfügen, können Sie die folgenden Konfigurationsoptionen nutzen:

Picture to display
Gallery 2.x ermöglicht es Ihnen, nicht nur ein Zufallsbild anzuzeigen, sondern auch ein Bild der zuletzt hochge yladenen Dateien (**Recent**), der meistangesehenen Bilder (**Most viewed**) oder auch eines gezielten Bildes (**Specific**).

Album ID to show
Standardmäßig zeigt das Plugin die Bilder egal welcher Unteralben an. Wenn Sie in dieses Eingabefeld die ID eines Albums eintragen, wird nur dieses Unteralbum berücksichtigt. Die ID eines Albums können Sie in der Gallery-Oberfläche herausfinden.

Maximum Width of Image
Das Plugin zeigt das gewünschte Bild als Vorschaugrafik direkt so an, wie es die Gallery-Software ausgibt. Wenn Sie eine andere Bildgröße benötigen, können Sie die maximale Breite dieses Bildes im Feld **Maximum Width of Image** eintragen. Dies bewirkt, dass das jeweilige Bild von Ihrem Browser auf diese Größe umgerechnet wird. Der Browser lädt dabei das große Bild herunter, daher kann diese Option dazu führen, dass das Bild zum einen vom Browser qualitativ minderwertig verkleinert wird und zum anderen auch mehr Bandbreite verbraucht.

Kurzum: Sie sollten lieber mit der Standardgröße der Gallery-Vorschaubilder vorlieb nehmen.

Link Target
Ein Klick auf das dargestellte Bild führt direkt in die Gallery-Anwendung und öffnet sie

im gleichen Browser-Fenster wie Ihr Blog. Wenn Sie dafür gerne ein neues Browser-Fenster öffnen wollen, können Sie im Feld **Link Target** das Schlüsselwort `_blank` eintragen.

Which details should be displayed
Zu jedem dargestellten Bild kann Gallery von sich aus Meta-Informationen anzeigen. Welche dieser Felder dargestellt werden, können Sie über das Eingabefeld **Which details should be displayed** regeln. Tragen Sie dort eine Liste aller gewünschten Schlüsselwörter ein, mit einem Komma voneinander getrennt. Alle verfügbaren Felder werden neben dem Eingabefeld erwähnt: `title` (Bildtitel), `date` (Veröffentlichungsdatum), `views` (Anzahl der Besucher des Bildes), `owner` (Eigentümer des Bildes), `heading` (Bildüberschrift).

Coppermine Database

Wenn Sie als **Image Source** die Quelle **Coppermine Database** auswählen, kann das Plugin direkt auf die MySQL-Datenbank der Coppermine-Gallery zugreifen und deren Bilder anzeigen.

Die Konfigurationsoptionen sind:

Server, Prefix, Database, Username, Password
Diese Optionen müssen die Datenbank-Zugangsdaten zu der Coppermine-Datenbank enthalten.

URL
In der Variable **URL** müssen Sie die volle URL zu Ihrer Coppermine-Installation eintragen.

Gallery Name
Das Seitenleisten-Plugin bindet einen Link zu Ihrer Coppermine-Installation ein. Die Beschreibung dieses Links geben Sie über die Option **Gallery Name** ein.

Usergroup
Die Coppermine-Galerie kann Zugriffsrechte zu Bildern abhängig von Benutzergruppen setzen. So können Sie private und öffentliche Bilder voneinander abgrenzen.

Das Coppermine-Plugin muss sich also einer bestehenden Coppermine-Benutzergruppe zuordnen, damit es weiß, welche Bilder es darstellen darf. Wenn Sie im Feld **Usergroup** den Namen einer Gruppe eintragen, die Sie bei Coppermine für Mitglieder Ihrer Familie vergeben haben, dann könnte das Plugin auch solche Bilder anzeigen. Wenn Sie dem Plugin den Namen einer Gastgruppe mitgeben, würden diese Bilder fehlen.

Standardmäßig gibt das Plugin einfach alle Bilder aus, daher ist die Benutzergruppe **Everybody** voreingestellt.

Size
Die Konfigurationsoption **Size** legt fest, wie groß das Vorschaubild (in Pixel, egal ob Höhe oder Breite) sein soll, das vom Plugin in der Seitenleiste angezeigt wird.

Thumbnails
Diese Option gibt an, wie viele Bilder angezeigt werden sollen.

Type
Standardmäßig zeigt das Plugin die aktuellsten Bilder (**Most Recent**). Über die Auswahlfelder der Konfigurationsoption **Type** können Sie jedoch auch die beliebtesten Bilder (**Most Viewed**) oder zufällige Bilder (**Random Images**) anzeigen.

Album Link
Wenn Sie diese Option aktivieren, wird unter jedem Bild ein Link zur Coppermine-Galerie eingebunden.

Gallery Link URL
Am Ende der Bilderübersicht können Sie einen Link einfügen, den Sie mit diesem Parameter angeben. Der Link zeigt üblicherweise direkt zu Ihrer Galerie, jedoch können Sie auch zu anderen Stellen verweisen.

Resolve Non-Images
Coppermine ermöglicht es Ihnen, auch Videos oder andere Dateien anstelle von Bildern zu verwalten. Wenn Sie die Option **Resolve Non-Images** aktiviert haben, versucht das Plugin solche Dateien ebenfalls in der Seitenleiste anzuzeigen.

Zooomr Plugin

Im Gegensatz zu den beiden eingangs erwähnten Mediendatenbanken werden die Bilder bei dieser Datenquelle nicht von Ihrem eigenen Server bezogen, sondern vom zentralen Zooomr-Server aus eingebunden.

Zooomr stellt dabei einen RSS-Feed zur Verfügung, der Ihre aktuellen Bilder enthält. Diesem RSS-Feed können Sie eine Reihe von Optionen mitgeben, die Sie auf der Zooomr-Homepage erfahren können.

In den Konfigurationsoptionen dieses Plugin-Bereichs können Sie ebenfalls die Größe der Vorschaubilder, die Anzahl der gezeigten Bilder und die Verlinkungs-Art verändern.

Media Library (Serendipity)

Natürlich ist es auch möglich, Bilder Ihrer Serendipity-Mediendatenbank in der Seitenleiste anzuzeigen. Hierfür dient die Option **Media Library** in den Einstellungen des Plugins. Folgende Konfigurationsoptionen sind dabei möglich:

Limit output to only hotlinked images
Wenn Sie diese Option aktivieren, werden nur die Bilder der Mediendatenbank angezeigt, die von fremden Servern stammen.

Zur Erinnerung: Serendipity kann Dateien auf Ihren Server hochladen, aber auch auf Dateien fremder Server verweisen und diese Grafiken oder Dateien trotzdem in der Datenbank vorhalten (*Hotlinking*).

Hotlink limiting keyword
Wenn Sie die Option **Limit output to only hotlinked images** aktivieren, können Sie über das Eingabefeld **Hotlink limiting keyword** ein spezielles Schlüsselwort vergeben. Daraufhin zeigt das Plugin nur die Bilder, bei denen das Schlüsselwort in der URL der Datei vorkommt.

Wenn Sie also einen Hotlink auf die Dateien `http://google.com/logo.jpg` und `http://yahoo.com/logo.jpg` in der Mediendatenbank gespeichert haben und als Schlüsselwort **google.com** eintragen, werden nur Bilder vom Google.com-Server angezeigt. Würden Sie stattdessen `logo.jpg` eintragen, könnten Sie beide Bilder sehen.

Pick a default directory
Mit diesem Auswahlfeld können Sie ein Stammverzeichnis der Mediendatenbank auswählen, von dem Sie Bilder darstellen wollen.

Output images strictly
Wenn Sie diese Option aktivieren, zeigt das Plugin nur die Bilder an, die in dem oben konfigurierten Verzeichnis abgespeichert worden sind. Nur wenn Sie die Option **Output images strictly** deaktiviert haben, werden auch die Bilder in möglichen Unterverzeichnissen des gewählten Verzeichnisses dargestellt.

Rotate image time
Jede Darstellung eines Zufallsbildes aus der Mediendatenkbank wird für einen gewissen Zeitraum zwischengespeichert, damit in diesem Zeitraum alle Besucher dasselbe Zufallsbild angezeigt bekommen. Über die Option **Rotate image time** stellen Sie diesen Zeitraum (in Minuten) ein. Die Zahl `30` würde also bewirken, dass jeweils eine halbe Stunde lang dasselbe Zufallsbild angezeigt wird.

Number of images to display
Hiermit steuern Sie die Anzahl der angezeigten Bilder.

Image width
Serendipity erzeugt Vorschaubilder in der konfigurierten Breite/Höhe (siehe Serendipity-Konfiguration auf Seite 170). Daher zeigt auch das Seitenleisten-Plugin standardmäßig die Bilder in genau dieser Breite an.

Wenn Sie größere Vorschaubilder in der Seitenleiste sehen wollen, können Sie den Browser anweisen, die Bilder selbständig zu vergrößern (oder zu verkleinern). Dies

führt jedoch immer zu einem Qualitätsverlust, da die Datei nicht aufgrund der Originalgröße reduziert wird, sondern immer nur das Vorschaubild gestaucht oder gestreckt werden kann. Wenn Sie hier die Zahl `0` eintragen, zeigt das Plugin ein Bild mit der vollen zur Verfügung stehenden Breite in der Seitenleiste an.

Behavior of image link

Mit der Option **Behavior of image link** könenn Sie einstellen, was bei einem Klick des Besuchers auf ein Bild passieren soll. Entweder kann dann das große Bild auf derselben Seite angezeigt werden (**In Page**), oder das Bild kann in einem Popup-Fenster erscheinen (**Pop Up**).

Eine weitere Möglichkeit ist, bei einem Klick auf ein Bild eine fest vorgegebene URL aufzurufen (die beispielsweise zu einer statischen Seite oder anderem führt). Wenn Sie die Option **URL** auswählen, müssen Sie erst die Plugin-Konfiguration speichern (über den Button **Speichern**) und können dann im Eingabefeld unter diesem Auswahlfeld die URL eintragen.

Eine besondere Option stellt die Variante **Try to link to related entry** dar. Wenn Sie diese Variante wählen, versucht das Plugin, zu jedem Bild einen Blog-Artikel zu finden, in dem das Bild eingesetzt wurde. Wenn ein Blog-Artikel gefunden wird, führt der Link dann zu diesem Beitrag.

Eine weitere Option namens **User-Galerie** steht hier zur Verfügung, wenn Sie das Serendipity-Plugin *serendipity_event_usergallery* (siehe Seite 358) installiert haben. Bei der Auswahl dieser Variante können Sie nach dem Speichern des Plugins den Permalink der User-Galerie eintragen.

Enter any text (or html) you would like placed before/after the picture

Am Ende der Konfigurationsoptionen finden Sie zwei große Texteingabebereiche. Hier können Sie einen Text eintragen, der vor und nach der Bilderübersicht in der Seitenleiste angezeigt wird.

5.3.7 Category Tree Menu serendipity_plugin_category_dhtml_menu

Das Standard-Seitenleisten-Plugin zur Darstellung von Kategorien zeigt diese einfach untereinander aufgelistet an. Da vielfach die Kategorie-Übersicht als eine Art Menüstruktur des Blogs benutzt wird, ist diese Darstellung möglicherweise nicht immer die beste.

Das Plugin *Category Tree Menu* bietet daher eine dynamische Kategorieansicht an. Auf oberster Ebene werden nur alle Hauptkategorien angezeigt, alle Unterkategorien werden ausgeblendet und nur durch Klick auf eine Hauptkategorie wieder angezeigt. So kann man, wie von Dateimanagern gewohnt, Kategorien platzsparend ein- und ausklappen.

Dieses Plugin benötigt die externe *PEAR::HTML_TreeMenu*-Bibliothek. Diese müssen Sie entweder als zentrale PEAR-Bibliothek über den Server-Provider installieren oder manu-

ell installieren. Dazu müssen Sie das PEAR-Paket herunterladen[9] und entpacken. Laden Sie die Datei `HTML_TreeMenu-1.2.0/TreeMenu.php` anschließend ins Serendipity-Verzeichnis `bundled-libs/HTML/` hoch. Kopieren Sie dann noch die Datei `TreeMenu.js` und das ganze Verzeichnis `images` in Ihr Plugin-Verzeichnis `plugins/serendipity_plugin_category_dhtml_menu`.

In den Konfigurationsoptionen des Plugins müssen Sie nun den vollständigen Pfad zu `TreeMenu.js` wie auch zum `images`-Unterverzeichnis eintragen. Bei der Standardkonfiguration im Beispiel dieses Buches wäre das also

```
/serendipity/plugins/serendipity_plugin_dategory_dhtml_menu/images/
```

und

```
/serendipity/plugins/serendipity_plugin_dategory_dhtml_menu/TreeMenu.js
```

Nur wenn diese beiden Pfade korrekt eingetragen sind, werden Sie später im Frontend eine Darstellung der Kategorien sehen. Bei einer Fehlkonfiguration ist das Seitenleisten-Plugin leer, und Ihr Browser zeigt möglicherweise eine JavaScript-Fehlermeldung an.

Die Verwendung des *Category Tree Menu* hat einen großen Nachteil gegenüber dem vorhandenen *Kategorien*-Plugin: Es zeigt immer pauschal alle Kategorien und Unterkategorien an und kann auch keine Leserechte zu einer Kategorie auswerten. Auch die Darstellung mehrerer Kategorien ist mit dem Plugin nicht möglich. Daher müssen Sie sich entscheiden, ob diese Möglichkeiten für Sie kritisch sind. Vielleicht findet sich in zukünftigen Versionen dieses Plugins ein Entwickler, der beide Plugins miteinander kombiniert?

[9] `http://pear.php.net/package/HTML_TreeMenu`

Kapitel 6

Ereignis-Plugins

Ereignis-Plugins dienen der Erweiterung von Funktionen im Frontend wie auch im Backend. Serendipity setzt innerhalb seines Quellcodes an zahlreichen strategisch wichtigen Stellen sogenannte *Event-Callbacks* bzw. *Event-Hooks*. An dieser Stelle des Codes führt Serendipity also nacheinander alle installierten Ereignis-Plugins aus, so dass jedes Plugin an dieser Stelle die Möglichkeit erhält, seine eigenen Funktionen auszuführen. Die Reihenfolge, in der die Ereignis-Plugins in der Plugin-Verwaltung aufgeführt sind, bestimmt dabei auch die Ausführungsreihenfolge.

Bei den Textformatierungs-Plugins ist die Ausführungsreihenfolge besonders wichtig. Ein Textformatierungs-Plugin kann bei Artikeltexten und auch Kommentartexten dazu benutzt werden, Text- oder Formatierungsänderungen durchzuführen. Daher reichen solche Plugins von einfachen Wortersetzungen (`s9y` könnte z. B. überall durch `Serendipity` ersetzt werden) über Smiley-Ersetzungen (`:-)` wird zu einem grinsenden Gesicht) bis zu speziellen Plugins, die Quelltexte in beliebigen Programmiersprachen korrekt einrücken und darstellen.

Stellen Sie sich ein Textformatierungs-Plugin vor (*Plugin A*), das das Wort `*doof*` in eine besondere Grafik verwandelt. Ein zweites *Plugin B* kümmert sich darum, dass alle Wörter, die von Sternchen (`*`) umgeben sind, im Artikel später fett geschrieben werden. Wenn Plugin A vor Plugin B platziert wird, sehen Sie nach der Umwandlung des Wortes `*doof*` wie gewünscht eine schöne Grafik. Wäre Plugin B jedoch das zuerst ausgeführte Plugin, würden Sie in der Ausgabe nur ein fettes Wort `doof` sehen. Denn nachdem Plugin B alle Wörter mit umgebenden Sternchen umgewandelt hat, kann das Smiley-Plugin das Wort `*doof*` nicht mehr finden.

Die Plugin-Reihenfolge ergibt sich also meist abhängig von Ihrem persönlichen Einsatz des Plugins, daher können keine allgemeinen Aussagen getroffen werden, welches Plugin an welcher Stelle stehen muss.

Textformatierungs-Plugins werden also immer erst bei der Ausgabe angewendet und bieten Ihnen den Komfort der einfacheren Eingabe. Wer möchte schon jedesmal den kompletten

HTML-Code für eine Smiley-Grafik einfügen?

Ihre Eingaben werden ganz unabhängig von der späteren Formatierung in der Datenbank gespeichert – wenn Sie also einen Artikel später überarbeiten, werden sämtliche Formatierungsänderungen dort noch nicht angezeigt, da sie nur bei der Anzeige angewendet werden und nicht beim Speichern der Rohdaten. Das bedeutet auch, dass sämtliche von einem Textformatierungs-Plugin vorgenommenen Änderungen nicht mehr ausgeführt werden, wenn Sie dieses Plugin löschen. Seien Sie also vorsichtig beim Löschen von Plugins und prüfen Sie, ob dadurch möglicherweise alte Artikel nicht mehr so dargestellt werden, wie Sie es erwarten.

Eine weitere Besonderheit von Textformatierungs-Plugins ist, dass alle diese Plugins eine Konfigurationsoption anbieten, um einzustellen, auf welche Ausgabefelder eine Formatierung angewendet wird. So können Sie einstellen, ob eine Formatierung nur auf normale Artikeltexte, erweiterte Artikeltexte, Kommentare von Besuchern oder HTML-Klötze (siehe Seite 216) angewendet wird. Im Eingabefeld für Kommentare von Besuchern kann jedes Textformatierungs-Plugin eigene Hinweise einblenden, wie deren Syntax zu benutzen ist.

Mehrere Ereignis-Plugins bieten die Möglichkeit, beliebige Inhalte innerhalb Ihres Blogs einzubinden: eigenständige HTML-Seiten (statische Seiten), Gästebücher, Kontaktformulare und weitere. Alle diese Plugins haben einen Satz an Konfigurationsoptionen gemeinsam, auf die Sie immer wieder stoßen werden. Damit diese nicht immer für jedes Plugin wiederholt werden müssen, werden Sie hier erklärt:

Permalink

In das Feld **Permalink** tragen Sie die URL ein, unter der Sie später die Ausgaben des jeweiligen Plugins wollen. Hier müssen Sie den vollständigen HTTP-Pfad eintragen, der zu dieser URL führt. Standardmäßig wird das Feld vorausgefüllt mit `/serendipity/pages/pluginname.html`, was Ihrem Pfadnamen und einem virtuellen Pfad entspricht. Im virtuellen Pfadnamen können Sie eine beliebige Struktur einsetzen. Dabei müssen Sie lediglich darauf achten, dass die URL mit `.html` endet, keine Sonderzeichen enthält und keinem bereits vorhandenen Permalink entspricht. Wichtig ist, dass diese Variable immer nur eine Pfadangabe enthalten darf, niemals eine vollständige URL wie `http://www.example.com/...`. Auch muss der Stammpfad zum Serendipity-Blog beibehalten bleiben, eine Eingabe wie `/anderes_blog/pages/pluginname.html` wäre ungültig.[1]

Wenn Sie in Ihrem Blog einmal die URL-Umformung (siehe Seite 168) (de-)aktivieren oder Ihr Blog in ein Unterverzeichnis verschieben, kann es sein, dass Sie die konfigurierten Permalinks aller Plugins ebenfalls anpassen müssen. Bei deaktivierter URL-Umformung muss ein Permalink einen Wert wie `/serendipity/index.php?/pages/pluginname.html` enthalten. Erst bei der Pfadangabe hinter `index.php?` dürfen Sie eine beliebige URL eintragen, der Pfad davor muss der Konfiguration Ihres Servers entsprechen.

[1] Eine derartige URL wäre ungültig, da der Aufruf dieser URL außerhalb der Serendipity-Installation läge. Serendipity kann nur virtuelle Verzeichnisse unterhalb seiner eigenen Verzeichnisstruktur verwalten.

URL-Titel der Seite, Seitentitel, Pagetitle oder URL shorthand name
Alternativ zu der Konfiguration eines Permalinks bieten alle Plugins auch einen *URL-Titel der Seite* oder *URL shorthand name* an. Mithilfe dieser Variable kann man die Ausgaben eines Plugins auch ohne Permalink ansehen, indem man

```
http://www.example.com/serendipity/index.php?serendipity[subpage]=
Seitentitel
```

aufruft. Dies ist wichtig, da in manchen Webserver-Konfigurationen ein Permalink nicht mit Formulardaten beschickt werden kann.[2]

Tragen Sie für den URL-Titel einer Seite ausschließlich Namen ohne Sonderzeichen, Umlaute und Leerzeichen ein. Beinahe alle Plugins weisen den Wert der `serendipity[subpage]`-Variable der Smarty-Variablen {`$staticpage_pagetitle`} zu. Über diese Variable kann man individuell in den Templates bei speziellen Plugins unterschiedliche Template-Ausgaben erreichen (siehe Seite 480).

Als Artikel formatieren
Wenn Sie die Option **Als Artikel formatieren** aktivieren, wird Serendipity die Ausgabe eines Plugins so darstellen, als sei dies ein normaler Blog-Artikel. Um die Ausgabe herum wird also das übliche HTML-Konstrukt erzeugt, das Serendipity standardmäßig ausgibt (mit Seitenüberschrift und Datum).

Wenn Sie diese Option deaktivieren, werden die Ausgaben 1:1 weitergereicht und in den Inhaltsbereich von Serendipity eingefügt. Das heißt, dass Serendipity sich in jedem Fall um die Ausgabe des HTML-Kopf- und -Fußkonstruktes sowie der Seitenleisten-Plugins kümmert, aber sonstige Überschriften nicht ausgibt.

6.1 Standardmäßig aktivierte Plugins

Die im Folgenden behandelten Ereignis-Plugins werden bei Serendipity mitgeliefert und sind standardmäßig aktiviert.

6.1.1 Spamschutz serendipity_event_spamblock

Das Plugin, das den Rekord für die höchste Anzahl von Konfigurationsoptionen hält, ist sicherlich das *Spamschutz*-Plugin. Erschrecken Sie nicht davor, sondern gehen Sie die Möglichkeiten ruhig nacheinander durch. Die Voreinstellungen des Plugins sind für gewöhnliche Zwecke bereits recht gut geeignet.

[2] Konkret ist dies bei der Nutzung der URL-Umformungsmethode `Apache ErrorHandling` nicht möglich. Eine URL wie `/serendipity/pages/seite.html?variable=inhalt` würde keine GET-Variable `$_GET['variable']` erzeugen, die die Plugins teilweise jedoch benötigen, um abhängig von der Anfrage des Besuchers bestimmte Inhalte anzuzeigen.

Anhand der Fülle verschiedener Optionen wird eines recht deutlich: Der Kampf gegen Spam (unerwünschte Werbung) ist extrem müßig und mit viel „Wenn“ und „Aber“ verbunden. Einen optimalen Schutz gegen Spam kann Ihnen auch dieses Buch leider nicht geben, dafür ändern sich die Techniken der Spammer zu häufig.

Als probates Mittel hat sich die Verwendung des *Akismet*-Services erwiesen. Dieser zentrale Server führt schwarze Listen von Kommentarspam-Fällen und kann von vielen Blogsystemen genutzt werden, darunter Serendipity, WordPress und MoveableType. Auch die Aktivierung von Captchas reduziert Kommentarspam sehr deutlich, bringt aber auch Usability-Einschränkungen mit sich.

Dafür ermöglichen es Ihnen aber viele der aufgeführten Optionen, flexibel auf neue Gegebenheiten eingehen zu können. Daher sollten Sie das Folgende aufmerksam durchlesen, um im Bedarfsfall schnell auf Spam reagieren zu können.

Das Spamschutz-Plugin wird jeweils aufgerufen, wenn in Ihrem Blog jemand einen Kommentar oder ein Trackback hinterlässt. Das Plugin prüft daraufhin, was der Benutzer übermittelt hat, und führt nacheinander mehrere Tests aus. Sobald einer der aktivierten Tests einen Kommentar als Spam markiert, wird der Artikel speziell vorgemerkt. Das Plugin merkt sich daraufhin, aufgrund welcher Maßnahme ein Artikel als Spam identifiziert wurde. Denn wie bei der Spam-Filterung bei E-Mails kann es leicht vorkommen, dass auch gültige Nachrichten aufgrund einer Filterregel als Spam klassifiziert worden sind. Daher ist es wichtig, die Filter von vornherein nicht zu „scharf“ einzustellen und ab und zu die Logdateien nach falsch eingestuftem Spam durchzugehen.

Wenn ein Kommentar als Spam identizifiert ist, können zwei Dinge geschehen: Entweder er wird komplett verworfen, oder er wird moderiert. Moderierte Kommentare müssen erst von einem Redakteur freigeschaltet werden. Komplett verworfene Kommentare werden erst gar nicht in der Datenbank gespeichert, und der Redakteur wird von einem solchen Kommentar nie etwas erfahren. Nur auf zu moderierende Kommentare wird ein Redakteur möglicherweise per E-Mail hingewiesen (abhängig von den Einstellungen des Redakteurs bezüglich E-Mail-Benachrichtigungen, siehe Seite 99).

Bei vielen Anti-Spam-Optionen können Sie einstellen, ob beim Zutreffen einer einzelnen Regel die Nachricht verworfen oder moderiert werden soll. Sobald die erste Regel zutrifft, die eine Nachricht verwirft, ist die Ausführung des Anti-Spam-Plugins beendet. Daher ist es durchaus möglich, dass ein Kommentar von einer Regel erst als „moderiert“ eingestuft wird, dann aber eine Folge-Filterregel den Kommentar doch „abweist“. Das Abweisen hat also Priorität vor der Moderation.

Wenn Sie einen Artikel bereits mit aktivierter Option **Kommentare und Trackbacks dieses Eintrags werden moderiert** erstellt haben, kann das Anti-Spam-Plugin dennoch Kommentare abweisen. Ein Kommentar kann dann jedoch niemals ohne Moderation veröffentlicht werden. Wenn Sie die Option **Kommentare für diesen Eintrag zulassen** bei einem Artikel deaktiviert haben, ist das Anti-Spam-Plugin für diesen Artikel irrelevant, da sämtliche Kommentare abgewiesen werden.

Notfall-Blockade von Kommentaren

Wenn gerade eine Spam-Welle über Sie hereinbricht, können Sie diese Option aktivieren, um im Blog keinerlei Kommentare oder Trackbacks anzunehmen. Weitergehende Blockademaßnahmen können Sie auf Seite 441 nachlesen.

Empfohlene Einstellung: *Nein*

Spamblock für Autoren deaktivieren

Häufig können Sie den angemeldeten Autoren des Blogs vertrauen, dass sie keine Spam-Einträge in Ihrem Blog vornehmen. Daher können Sie mit der Einstellung **Spamblock für Autoren deaktivieren** dafür sorgen, dass alle eingeloggten Redakteure bei Kommentaren von keinerlei Anti-Spam-Maßnahmen betroffen sind (Einstellung **Alle Autoren**). Eingeloggte Autoren sehen also keine Captcha-Grafiken und werden nicht anderweitig von Filterungen behelligt. Wenn Sie das Auswahlfeld auf **keine** stellen, unterscheidet das Plugin nicht zwischen anonymen und eingeloggten Besuchern.

Ansonsten können Sie in diesem Feld gezielt Benutzergruppen auswählen. Wenn Sie das Plugin *Freie Benutzer-Registrierung* benutzen, ist dies besonders sinnvoll, um „anonyme“ Redakteure dennoch von den „echten“, manuell erzeugten Redakteuren unterscheiden zu können. Achten Sie dann einfach darauf, dass die Benutzergruppe der Redakteure, die über das Registrierungs-Plugin hinzugefügt werden, nicht im Auswahlfeld markiert ist.

Empfohlene Einstellung: *Alle Autoren*

Trackback IP Validierung

Wenn Sie von einem fremden Blog ein Trackback erhalten (siehe Seite 433), dann wird dieses Trackback von einem bestimmten Server aus gesendet. In einem Trackback selbst ist die URL des fremden Blogs enthalten.

Im üblichen Fall entspricht die IP-Adresse des Servers, der das Trackback sendet, genau der IP-Adresse der URL, die im Trackback enthalten ist.

Bei Spam sieht dies anders aus: Die Trackbacks werden von infizierten Computersystemen aus verschickt, und die beworbene URL entspricht meist der von Spam-Seiten und liegt auf einem völlig anderen Server.

Das Antispam-Plugin kann solche Trackbacks automatisch abweisen, wenn die IPs des sendenden Servers und des beworbenen Servers nicht übereinstimmen.

Die Aktivierung dieser Option ist sehr empfehlenswert, da es im täglichen Einsatz de facto nicht zu einer fehlenden Übereinstimmung bei echten Trackbacks kommen kann. Lediglich in Randfällen wie Servern, die Trackbacks über Proxy-Server verschicken müssen, kann dies gültige Trackbacks verwerfen. Wägt man den positiven Nutzen des Schutzes vor ungültigem Spam dagegen ab, ist dieser Nachteil jedoch leicht zu verschmerzen.

Empfohlene Einstellung: *abweisen*

Keine doppelten Kommentare erlauben

Üblicherweise sollte es in einem Blog nie dazu kommen, dass identische Kommentare mehrfach vorkommen. Daher können Sie Kommentare sperren, die bereits in der

Datenbank vorhanden sind, wenn Sie die Option **Keine doppelten Kommentare erlauben** auf **Ja** stellen.

Viele Spammer variieren ihre Inhalte durch zufällig erzeugte Buchstaben und sind daher von dieser Einstellung nicht betroffen. Dennoch kann diese Anti-Spam-Maßnahme auch helfen, doppelte profane Kommentare wie „super!“ zu unterbinden.

Empfohlene Einstellung: *Ja*

Kommentare abweisen, die als Text nur den Artikeltitel enthalten

Eine beliebte Form des Kommentar-Spams stellte einmal die Methode dar, bei einem Kommentar einfach den Titel Ihrer eigenen Artikel zu übernehmen. Anstelle eines Links zu einem fremden Artikel erhielten Spammer ihren Nutzen dadurch, dass deren Benutzername oder Homepage auf eine fremde URL verwies.

Wenn Sie diese Option aktivieren, werden solche Kommentare verboten. In der freien Wildbahn treten derartige Spam-Kommentare kaum noch auf, daher ist es empfehlenswert, die Option aus Geschwindigkeitsgründen besser zu deaktivieren.

Empfohlene Einstellung: *Nein*

IP-Block Intervall

Jeder Besucher des Blogs besitzt eine eigene IP, die den Benutzer im Internet identifiziert. Üblicherweise hat somit jeder Besucher der Webseite eine eindeutige IP, die sich jedoch bei jeder neuen Einwahl des Benutzers ins Internet unterscheiden wird. Mit einer so gewonnenen IP-Adresse kann man versuchen, Kommentare von dort nach gewissen Kriterien zu unterbinden.

IP-Adressen sind jedoch leider keine verlässliche Maßnahme. Wenn ein Besucher einen sogenannten *Proxy*[3] benutzt, werden seine Zugriffe über die IP-Adresse dieses Servers durchgeführt. Der Proxy verschleiert also die wahre Identität des Besuchers. Da große Unternehmen wie T-Online und auch AOL für ihre Benutzer zentrale Proxy-Server benutzen, könnte man also beim Blocken einer solchen IP nicht nur den Zugriff für eine einzelne Person verhindern, sondern für alle Besucher, die diesen Proxy verwenden.

Kurzum, Sie sollten der IP-Adresse kein absolutes Vertrauen schenken. Dennoch kann es helfen, Missbrauch auf gewisse Weise einzudämmen. Die Option **IP-Block Intervall** ermöglicht es, dass eine eindeutige IP nur einmal einen Kommentar eintragen darf, und dann erst wieder nach Ablauf des konfigurierten Zeitraums. Somit ist es nicht mehr möglich, mehrere hundert Kommentare pro Sekunde abzusetzen.

In Blogs passiert es selten, dass Benutzer (auch die Benutzer eines Proxies) im Minutentakt gewünschte Kommentare hinterlassen. Daher ist ein Zeitraum von einer Minute hier durchaus empfehlenswert.

Die Abweisung eines Kommentars mittels dieser Option erfolgt erst bei der Abarbeitung eines bereits übermittelten Kommentars. Das heißt, der eigentliche Zugriff auf das Blog wird anhand dieser Option nicht eingeschränkt.

[3] Ein Proxy ist eine Art Webserver, der die Inhalte fremder Webseiten zwischenspeichern kann, um so Traffic zu reduzieren oder Inhalte leichter zu filtern.

Empfohlene Einstellung: *60*.

CSRF-Schutz aktivieren?

Die meisten Spam-Kommentare erfolgen, indem ein Spam-Roboter automatisch Ihre URLs aufruft und einen Kommentar überträgt. Meist macht sich der Roboter nicht die Mühe, vorher (wie ein normaler Blog-Besucher) die eigentliche Webseite aufzurufen. Dieses Vorgehen kann man sich zur Abwehr zunutze machen: Man speichert auf der Seite, von der aus das Kommentarformular eingebunden wird, einen Zufallswert. Nur wenn dieser Zufallswert gültig mit dem Kommentar vom Browser des Besuchers übermittelt wird, kann man davon ausgehen, dass ein menschlicher Besucher auch vorher den Artikel aufgerufen hat und dass nicht einfach etwas automatisiert übertragen wird.

Die Aktivierung dieser Option hat leider einen gravierenden Nachteil. Denn damit der eingegebene Zufallswert auf dem Server zur Prüfung zwischengespeichert werden kann, muss eine Session (siehe Terminologie, Seite 49) für den Besucher angelegt werden. Eine Session ist jedoch von Serendipity nur nutzbar, wenn der Browser des Besuchers Cookies annimmt (siehe Hinweis Seite 49).

Weiterhin bringt diese Option einen Schutz vor *CSRF*[4]. Dies führt zu Angriffen, bei denen ein böswilliger Benutzer Sie dazu bringen könnte, ungewollt Kommentare zu verfassen oder sogar freizuschalten.

Empfohlene Einstellung: *Ja*.

Captchas aktivieren

Captchas (siehe Seite 44) sind kleine Grafiken mit schwer zu lesenden Zeichenkombinationen. Damit ein Besucher einen Kommentar schreiben darf, muss er die dargestellten Zeichen in eine Box abtippen, um sich dem System gegenüber als „Mensch“ auszuweisen.

Captcha-Grafiken müssen eine gewisse Komplexität aufweisen, damit sie tatsächlich nicht von Computern überlistet werden können. Dies kann dazu führen, dass es auch den Besuchern immer schwerer fällt, die Captchas zu entziffern. Dennoch bietet diese Maßnahme eine der effektivsten Möglichkeiten, automatisierten Spam zu unterbinden.

Mit der Option **Stärkere Captchas** können Sie die Komplexität der Grafiken noch weiter erhöhen, indem weitere Zufallsmuster eingefügt werden. Diese Option ist mehr für den „Spam der Zukunft“ vorgesehen und sollte derzeit glücklicherweise noch nicht erforderlich sein.

Um Captchas anzeigen zu können, sollte Ihr Server die PHP *gdlib* anbieten. Andernfalls werden die Zeichenfolgen mit Grafikdateien zusammengebaut, was von Spam-Robotern um einiges einfacher zu umgehen ist. Damit Ihre Besucher Captchas ausfüllen können, müssen sie in ihrem Browser Cookies annehmen. Wie Sie die Captcha-Grafiken anpassen können, ist auf Seite 251 ausgeführt.

Empfohlene Einstellung: *Ja*.

[4] *Cross Site Request Forgery* nennt man den Versuch eines fremden Benutzers, Ihren Browser fernzusteuern und damit Aktionen auszulösen, die Sie selbst gar nicht ausführen möchten.

Captchas nach wie vielen Tagen erzwingen

Da Captchas gerade für sehbehinderte Menschen große Probleme schaffen, möchte man diese Grafiken so selten wie möglich erforderlich machen.

Da Spammer mit Vorliebe alte Artikel in Ihrem Blog als Ziel nehmen (da diese Artikel bei Suchmaschinen bereits besser vertreten sind), sind häufig neue Artikel des Blogs nicht betroffen.

Dies können Sie ausnutzen, indem Sie Captchas nur dann einblenden, wenn ein Artikel ein gewisses Alter erreicht hat. Das Mindestalter tragen Sie in Tagen ein. Wenn Sie *0* eintragen, werden Captchas auch bei aktuellen Artikeln direkt eingebunden.

Empfohlene Einstellung: *14*.

Hintergrundfarbe des Captchas

Die Captcha-Grafik wird unterhalb des Eingabeformulars für einen Kommentar eingebunden. Je nach Design Ihres Blogs möchten Sie die Hintergrundfarbe dieser automatisch erzeugten Grafiken sicher anpassen. Daher können Sie bei dieser Option eine Farbe im RGB-Format eintragen. Drei Zahlen repräsentieren hier, von einem Komma voneinander getrennt, den Farbwert von *Rot, Grün* und *Blau*. Wer sich an die Farbenlehre erinnern kann, weiß, dass man mit diesen drei Primärfarben jede andere vom Monitor darstellbare Farbe abbilden kann. Um diese Farbwerte leicht herauszufinden, können Sie kleine Tools wie etwa den Colorpicker[5] benutzen.

Empfohlene Einstellung: *255,255,255* (Weiß)

Kommentarmoderation nach wievielen Tagen erzwingen

Bei besonders alten Artikeln ist es sehr selten, dass Besucher noch gewünschte Kommentare hinterlassen. Blogs sind meistens so tagesaktuell, dass ältere Artikel schnell an „Kommentarwert“ verlieren. Dies können Sie also ausnutzen, um Kommentare zu alten Artikeln von vornherein auszusortieren.

Wenn Sie hier die Zahl `0` eintragen, wird diese Möglichkeit der automatischen Moderation deaktiviert. Eine Zahl wie `60` bewirkt, dass Artikel älter als zwei Monate nicht ohne Ihre Zustimmung kommentiert werden können.

Empfohlene Einstellung: *60*

Was soll mit auto-moderierten Kommentaren passieren?

Üblicherweise werden Kommentare zu Artikeln, die Sie mit der Option **Kommentarmoderation nach wievielen Tagen erzwingen** gefiltert haben, in der Datenbank gespeichert, aber nicht freigeschaltet.

Über die Option **Was soll mit auto-moderierten Kommentaren passieren** können Sie mit solchen Kommentaren auch noch härter verfahren und Kommentare vollständig **abweisen**.

Empfohlene Einstellung: **moderieren**

[5] `http://www.pagetutor.com/colorpicker/index.html`

Trackbackmoderation nach wievielen Tagen erzwingen

Ähnlich der Einstellung für Kommentare können auch Trackbacks nach einem gewissen Zeitraum automatisch moderiert werden.

Empfohlene Einstellung: *60*

Was soll mit auto-moderierten Trackbacks passieren?

Ähnlich der Einstellung für Kommentare können die durch automatische Moderation erfassten Trackbacks entweder moderiert oder verworfen werden.

Empfohlene Einstellung: *moderieren*

Behandlung von per API übermittelten Kommentaren

Es gibt bei Serendipity grundsätzlich zwei Möglichkeiten, als Benutzer eine Mitteilung zu einem Blog-Artikel zu hinterlassen. Zum einen sind das Kommentare, die direkt im Blog eingetragen werden. Hierbei ist also eine Interaktion des Benutzers notwendig, Captchas und andere Anti-Spam-Maßnahmen können in den Prozess eingebunden werden.

Anders ist dies bei Trackbacks. Derartige „Kommentare“ werden von einem automatischen System verschickt und empfangen, daher können dort keine Captchas zwischengeschaltet werden. Solche automatische Methoden bezeichnet man daher als API (siehe Terminologie auf Seite 40). Sie können derartige Kommentare, die immer von Computern übermittelt werden, gezielt behandeln: Entweder werden solche Kommentare **abgewiesen**, oder sie müssen von Ihnen **moderiert** werden.

Wenn Sie automatisch übermittelte Kommentare gleichrangig wie via Formular eingetragene Kommentare behandeln wollen, können Sie die Option **keine** wählen.

Da das Aktivieren der automatischen Moderation meist eine große E-Mail-Flut von Benachrichtigungen mit sich bringt und man gültige von ungültigen Trackbacks schwer trennen kann, macht eine pauschale Moderation von Trackbacks selten Sinn. Nur wenn Sie auf Trackbacks vollständig verzichten wollen, sollten Sie die Option auf **abweisen** stellen.

Empfohlene Einstellung: *keine*

Trackback-URLS prüfen

Der Sinn eines Trackbacks ist, dass ein fremdes Blog Sie darauf hinweisen kann, dass es sich auf einen Ihrer Artikel bezieht. Daher muss ein Trackback immer die URL des fremden Blogs enthalten, auf dem sich jemand zu Ihrem Artikel äußert.

Bei Spam ist das meistens nicht der Fall – die URL, die das Trackback bewirbt, enthält meistens nur besondere Angebote zum preiswerten Bezug von Viagra oder tolle Vorschläge, wie man reich werden kann. Für die Spammer wäre es ein hoher Aufwand, sich dort tatsächlich auf Ihren Blog-Artikel zu beziehen.

Diese Nachlässigkeit der Spammer können Sie ausnutzen und mit der Option **Trackback-URLs prüfen** einstellen, ob alle Trackback-URLs daraufhin geprüft werden sollen, ob sie die URL Ihres Artikels enthalten.

Die Aktivierung dieser Option birgt jedoch zwei Gefahren: Wenn ein Spammer eine ungültige URL oder eine URL mit besonders vielen Daten hinterlässt, dann wird Ihr Webserver diese Seite gnadenlos trotzdem aufrufen und viel Traffic verursachen. Da die URL-Prüfung ausgeführt wird, während das Trackback gespeichert wird, kann dies die Dauer der Prüfung sehr in die Länge ziehen und Ihren Server stark auslasten. Zusätzlich darf bei der Aktivierung dieser Option der Webserver nicht von einer Firewall blockiert werden, damit fremde URLs überhaupt aufgerufen werden können.

Die zweite Gefahr ist, dass Sie möglicherweise Trackbacks abweisen, die eigentlich gültig wären. Manche Blogsysteme senden ein Trackback an Ihr Blog nämlich bereits, bevor der eigene Artikel veröffentlicht wurde. Das Blogsystem würde also erst dann seinen Artikel mit dem Link zu Ihnen online stellen, wenn das Trackback an Sie abgearbeitet wurde. Da Sie aber bereits bei der Annahme des Trackbacks die Existenz der fremden URL voraussetzen, würde das Trackback fehlschlagen.

Empfohlene Einstellung: *Nein*

Erforderliche Anzahl an Links für Moderation

Viele Spammer nennen in ihren Kommentaren einfach zahlreiche Links und hoffen, dass Ihre Besucher diese Links anklicken. Es ist üblicherweise ungewöhnlich, dass Kommentatoren Ihrer Artikel dutzende von Links an Sie schicken, daher können Sie in diesem Optionsfeld eine Anzahl von Links angeben, bei deren Erreichen ein Kommentar automatisch moderiert werden soll.

Empfohlene Einstellung: *7*

Erforderliche Anzahl an Links für Abweisung

Die Anzahl der erforderlichen Links im oberen Eingabefeld dient nur der Moderation eines Kommentars. Wenn Sie ab einer gewissen Anzahl von Links aber einen Kommentar direkt abweisen wollen, können Sie dies im Feld **Erforderliche Anzahl an Links für Abweisung** eintragen.

Empfohlene Einstellung: *13*

Wortfilter aktivieren

In den folgenden großen Eingabefeldern können Sie Zeichenketten eintragen. Sobald eine dieser Zeichenketten in einem Kommentar vorkommt („Blacklisting“), kann ein Kommentar speziell behandelt werden.

Wie diese Behandlung ausfallen soll, stellen Sie über die Option **Wortfilter aktivieren** ein. Ob Sie diese auf **moderieren** oder **abweisen** stellen, hängt davon ab, wie „scharf“ Sie Ihre Wortfilter einstellen. Wenn Sie beispielsweise das Wort `arsch` filtern, dann ist das erstmal ein guter Gedanke. Vielleicht denken Sie aber nicht daran, dass ja diese Zeichenkette auch bei dem Wort „marschieren“ vorkommen könnte und in diesem Fall keineswegs filterungswürdig wäre. Hätten Sie den Wortfilter hier auf **abweisen** gestellt, wäre ein Kommentar mit diesem Wort vollständig verloren, und der Besucher wundert sich womöglich, warum sein Kommentar nicht angenommen wurde.

Daher ist es ganz wichtig, dass Sie die folgenden Wortfilter so eng wie möglich fassen, um gültige Buchstabenkonstellationen nicht auch zu verhindern. Für obiges Beispiel wäre daher der Einsatz von ␣arsch␣ eher zu empfehlen. Durch die zusätzlichen Leerzeichen würde der Filter nur noch dann aktiv werden, wenn das Wort isoliert vorkommt.

Empfohlene Einstellung: *abweisen*

Wortfilter für URLs, Autorennamen, Inhalt und E-Mail-Adressen

Auf jedes Feld, das ein Kommentator ausfüllt, können Sie einen Wortfilter anwenden.

Die zu filternden Wörter müssen mit dem Semikolon (`;`) getrennt werden. Wenn Sie am Zeilenende angelangt sind, können Sie vor oder nach diesem Semikolon auch gerne einen Zeilenumbruch (*(Return)*-Taste) zur besseren Lesbarkeit einfügen.

Jedes eingetragene Wort wird dabei einzeln auf das Vorhandensein im Kommentar geprüft. Trifft eines der Wörter zu, wird der Kommentar entsprechend der Wortfilter-Einstellung entweder moderiert oder abgewiesen.

Sämtliche Wortfilter werden als „reguläre Ausdrücke“[6] interpretiert. Dies ist eine spezielle Syntax, in der Sie auch Platzhalter verwenden können. Reguläre Ausdrücke sind ein Thema für sich, aber hier ein Beispiel für einen Wortfilter-Ausdruck:

```
\@[^\s]+\.jp\s
```

Dieser Ausdruck würde einen Kommentar dann sperren, wenn jemand eine E-Mail-Adresse angibt, die zu einer japanischen Domain führt. Ein Kommentar wie „Mail me at shyguy@yahoo.jp“ würde also blockiert werden.

Aufgrund des Einsatzes von regulären Ausdrücken gibt es bei den Wortfiltern Besonderheiten für Zeichen, die bei regulären Ausdrücken besonders genutzt werden. Solche Zeichen muss man, wenn sie im Wort vorkommen sollen, *escapen*. Dazu stellt man einfach den Backslash (\) vor eines der folgenden Sonderzeichen:

```
@ [ ] ( ) { } . ? * ^ $ | + -
```

Wenn Sie also die große Flexibilität von regulären Ausdrücken nicht nutzen wollen, können Sie die Wortfilter auch ganz einfach als Wortliste verwenden. Einzig beim Auftreten eines Sonderzeichens müssen Sie daran denken, den Backslash voranzustellen.

Wenn Sie ungültige reguläre Ausdrücke eingeben, kann dies zu PHP-Fehlermeldungen bei der Kommentarabgabe führen oder sogar dazu, dass keinerlei Kommentare mehr gespeichert werden können. In diesem Fall sollten Sie versuchen, alle Wortfilter zu löschen. Wenn es danach wieder klappt, können Sie nach und nach die ursprünglichen Zeichenketten wieder einfügen und so herausfinden, welche Regel falsch war.

Empfohlene Einstellung: *Keine Empfehlung möglich*

[6] `http://de.wikipedia.org/wiki/Regul%C3%A4rer_Ausdruck`

URL-Filterung anhand der blogg.de Blacklist aktivieren

Der Blog-Provider `blogg.de` hat in der Vergangenheit eine Liste von URLs geführt, die von Spammern benutzt wurden. Serendipity kann einen Kommentar mit dieser schwarzen Liste abgleichen und bei einem Treffer nach Wunsch **moderieren** oder **abweisen**.

Aufgrund des hohen Pflegeaufwands ist die Blacklist dieses Providers seit längerer Zeit nicht mehr aktiv, daher ist die Aktivierung dieser Option zum jetzigen Zeitpunkt zwecklos. Möglicherweise wird die Blacklist irgendwann wieder aktiviert, daher wird diese Option weiterhin angeboten.

Empfohlene Einstellung: *keine*

Akismet API Key

Akismet ist ein zentraler Web-Service, der eine Schnittstelle für Blogsysteme wie WordPress, Serendipity und MoveableType anbietet. Ein Kommentar wird vollständig an den Service übermittelt, der Service überprüft den Kommentar und sieht nach, ob er Spam-Kriterien aufweist.

Da sehr viele Systeme Spam an diesen Service melden und die Datenbank bereits sehr groß ist und aktiv gepflegt wird, kann Akismet relativ verlässlich entscheiden, ob ein Kommentar Spam darstellt oder nicht.

Wenn Sie den Akismet-Dienst benutzen wollen, benötigen Sie einen sogenannten API-Key, den Sie mit der Anmeldung[7] erhalten. Sie müssen ihn dann im Spamschutz-Plugin im Konfigurationsfeld **Akismet API Key** eintragen.

Die Abfrage des Akismet-Servers kann nur klappen, wenn Ihr Webserver nicht von einer Firewall am Verbindungsaufbau gehindert wird.

Beachten Sie bei der Benutzung von Akismet, dass es auch hier zu falsch erkanntem Spam kommen kann und dass in besonders geschützten Intranet-Blogs Kommentare so an einen zentralen Server übertragen werden, über den Sie keine Kontrolle haben. Blogs, bei denen Sie schützenswerte Inhalte hinterlegen, sollten Sie daher nicht mit Akismet betreiben.

Empfohlene Einstellung: *Akismet-Schlüssel beantragen!*

Behandlung von Akismet-Spam

Mit dieser Option legen Sie fest, wie von Akismet erkannter Spam behandelt werden soll. Sie können derartigen Spam entweder **moderieren** oder **abweisen**.

Der große Komfort von Akismet und die relativ geringe Fehlerquote ermöglichen es, diese Option auf **abweisen** einzustellen. So erhalten Sie wirklich nur noch Moderationshinweise von Kommentaren, bei denen die Einstufung als Spam aufgrund anderer Spamschutz-Kriterien nicht eindeutig war.

Empfohlene Einstellung: *abweisen*

[7] `http://akismet.com`

E-Mail-Adressen bei Kommentatoren verstecken

Wird ein Kommentar bei einem Blog-Artikel angezeigt, sehen Sie auch die Informationen zu demjenigen, der den Kommentar verfasst hat. Dabei kann (je nach Template) auch die E-Mail-Adresse angezeigt werden, die wiederum Spam-Roboter in Ihrem Blog sammeln könnten, um den Kommentatoren mit Werbenachrichten zu belästigen.

Um dies zu verhindern, können Sie die E-Mail-Adresse Ihrer Besucher mit der Option **E-Mail-Adressen bei Kommentatoren verstecken** stets ausblenden, bzw. automatisch mit der Dummy-Adresse `nospam@example.com` ersetzen.

Als Administrator können Sie die echte E-Mail-Adresse nach wie vor in der Backend-Kommentar-Oberfläche einsehen.

Empfohlene Einstellung: *Ja*

Auf ungültige E-Mail-Adressen prüfen?

Grundsätzlich ist es möglich, dass Kommentatoren in Ihrem Blog eine ungültige E-Mail-Adresse eintragen. Gerade Spammer können dies benutzen, um statt einer E-Mail-Adresse einfach eine Homepage zu übertragen.

Das Spamschutz-Plugin kann dies in gewissem Maße einschränken und eine (grobe) Prüfung durchführen, ob eine E-Mail-Adresse ein gültiges Muster besitzt (also *`user@domain.land`*).

Wenn Sie diese Option aktivieren, kann Serendipity jedoch nicht prüfen, ob die E-Mail-Adresse auch tatsächlich existiert. Fantasie-Adressen mit gültigem Muster können also so nicht abgefangen werden.

Empfohlene Einstellung: *Ja*

Pflichtfelder

Wenn ein Kommentator die Formularfelder ausfüllt, um einen Kommentar zu hinterlassen, gibt es eine Reihe von optionalen Feldern. Serendipity lässt Kommentare zu, bei denen lediglich der Kommentartext ausgefüllt wird und ansonsten alle Angaben anonym sind.

Wenn Sie dies verhindern wollen, können Sie Pflichtfelder im Kommentarformular definieren. Alle Pflichtfelder tragen Sie dabei in dem Eingabefeld ein. Folgende Felder stehen zur Verfügung: **name** (Name des Kommentators), **email** (E-Mail-Adresse), **url** (Homepage), **comment** (Kommentartext). Ein weiteres Feld ist **replyTo**, das angibt, auf welchen vorausgehenden Kommentar sich ein Benutzer bezieht. Dieses Feld als Pflichtfeld zu bestimmen macht meistens keinen Sinn, weil man einen Besucher dadurch zwingt, sich auf einen existierenden Kommentar zu beziehen.

Wenn Sie Pflichtfelder definieren, müssen Sie Ihre Besucher auch darüber in Kenntnis setzen. Schlagen Sie dazu im Kapitel 9.6.2 auf Seite 548 nach, wie Sie Anpassungen am Kommentarformular vornehmen können.

Empfohlene Einstellung: *comment*

Block bad IPs via HTaccess?

Eine sehr experimentelle Option stellt **Block bad IPs via HTaccess** dar. Innerhalb einer `.htaccess`-Datei können Sie bei Apache-Webservern Regeln definieren, um Besucher mit einer bestimmten IP-Adresse abzulehnen.

Diese Besucher können dann das gesamte Blog nicht mehr aufrufen und belasten dabei glücklicherweise auch das System nur noch minimal – denn Serendipity kommt selbst gar nicht mehr zum Zuge, um den Aufruf dieses Besuchers zu bearbeiten, da der Webserver ihn bereits zuvor abgewiesen hat.

Derartige Verbote können mittels des Befehls `Deny From` *`IP-Adresse`* vorgenommen werden. Dieses Kommando stellt eine Blacklist von ein oder mehreren IP-Adressen dar und könnte auch manuell in der `.htaccess`-Datei eingestellt werden.

Das Spamschutz-Plugin ermöglicht jedoch eine interessantere Art der Einbindung. Jedesmal, wenn beim Prüfen eines Kommentars dieser abgewiesen wird, merkt sich das Spamblock-Plugin die IP der Person, die diesen Kommentar hinterlassen hat. So entsteht eine Liste, mit der man herausfinden kann, von welchen IPs in letzter Zeit Spam geschickt wurde. Sobald dieser Tabelle eine IP hinzugefügt wird, aktualisiert das Plugin die `.htaccess`-Datei und sperrt den Zugriff für alle IPs, die in den letzten zwei Tagen dort eingetragen worden sind.

Daher kann ein Besucher mit einer „verdächtigen“ IP-Adresse frühestens nach zwei Tagen erneut probieren, einen Kommentar zu hinterlassen.

Der große Vorteil dieser Sperrungsart ist, dass Spammern so schnell die Server-Ressourcen entzogen werden können. Die Nachteile sind jedoch, dass zum einen nur Apache-Webserver mit dieser Option arbeiten können und man zum anderen möglicherweise einen zu großen Benutzerkreis blockiert (siehe Anmerkungen zur Option **IP-Block Intervall** auf Seite 242). Benutzen Sie die Option daher nur, wenn Sie wissen, was sie bewirkt.

Empfohlene Einstellung: *Nein*

Protokollierung von fehlgeschlagenen Kommentaren

Wenn ein Kommentar oder Trackback moderiert oder abgewiesen wird, bemerken Sie dies als Redakteur möglicherweise gar nicht. Daher ermöglicht das Spamschutz-Plugin, solche Meldungen entweder in der Datenbank oder in einer Datei zu speichern.

Dieses Protokoll können Sie dann von Zeit zu Zeit prüfen, um herauszufinden, ob Ihre Anti-Spam-Einstellungen in der Gesamtheit noch Wirkung haben.

Wenn Sie das Protokoll in einer **Einfachen Datei** speichern, müssen Sie einen Speicherplatz angeben. Sie können diese Datei dann von Zeit zu Zeit herunterladen und auf dem Server wieder löschen, damit sie nicht zu groß wird.

Für die Auswertung von Protokollen ist die Speicherung in einer **Datenbank** sicher empfehlenswerter. Die Datenbanktabelle `serendipity_spamblocklog` können Sie mit einem Programm wie *phpMyAdmin* komfortabel ansehen und nach bestimmten Kriterien filtern.

Fortgeschrittene Benutzer können die Datenbanktabelle nutzen, um sich kleine Scripte zu schreiben, die täglich oder wöchentlich Statusberichte per E-Mail verschicken oder auch im Blog anzeigen. Auch diese Datenbanktabelle sollten Sie von Zeit zu Zeit leeren, um nicht zu viele alte Daten vorzuhalten.

Speicherplatz für das Logfile
Wenn Sie die Protokollierung in eine einfache Datei aktiviert haben, müssen Sie hier den vollständigen Dateisystem-Pfad auf dem Server eintragen, wo das Logfile gespeichert werden soll. Der Webserver muss Schreibrechte zu diesem Verzeichnis haben, daher können Sie z. B. auch Ihr `uploads`-Verzeichnis dafür verwenden.

Unterhalb der Konfigurationsoptionen bindet das Plugin noch eine kleine Vorschaugrafik der Captchas ein. Dort können Sie sehen, wie ein Captcha für einen Besucher aussieht, und gegebenenfalls Änderungen der Hintergrundfarbe vornehmen.

Die Art der Captcha-Grafiken richtet sich danach, ob Ihr Webserver das PHP-Modul *gdlib* unterstützt. Ohne *gdlib* gibt das Spamblock-Plugin PNG-Grafiken im Verzeichnis `plugins/serendipity_event_spamblock` direkt aus. Für jeden Buchstaben gibt es eine eigene `captcha_`*`zeichen`*`.png`-Datei, die Sie mit einem Bildbearbeitungsprogramm anpassen können.

Wenn *gdlib* vorhanden ist, gilt: Die verwendeten Schriftarten der Captchas können Sie nur mit etwas Aufwand verändern. Das Plugin kann mit beliebigen TTF-Schriftdateien, wie sie Windows mitliefert, umgehen. Diese Dateien liegen im Verzeichnis `plugins/serendipity_event_spamblock` und sind standardmäßig auf die vier Schriftarten Vera, VeraSE, Chumbly und 36daysago beschränkt. Das Auswahlkriterium der Schriftarten war, dass diese möglichst nicht maschinenlesbar sein sollen. Wenn Sie lieber eine eigene, besser lesbare Schriftart benutzen wollen, können Sie die vorhandenen TTF-Dateien mit anderen ersetzen und müssen dabei die Dateien lediglich umbenennen.

Das Plugin wählt zufällig eine der vier verfügbaren Schriftarten aus. Wenn Sie mehr als vier Schriftdateien benutzen wollen, müssen Sie die Plugin-Datei `plugins/serendipity_event_spamblock/serendipi spamblock.php` öffnen und bearbeiten. Suchen Sie nach der Zeile

```
$fontfiles = array('Vera.ttf', 'VeraSe.ttf', 'chumbly.ttf',
'36daysago.ttf');
```

Dort müssen Sie (jeweils in Anführungszeichen) den Namen der Schriftdatei kommasepariert eintragen.

Etwas oberhalb dieser Codezeile findet sich auch die Angabe, wie groß die Captcha-Grafik ist:

```
$width = 120;
$height = 40;
```

Mit den beiden Zahlen `120` (Breite) und `40` (Höhe) können Sie die Bildgröße anpassen.

Die Schriftgröße müssen Sie über eine weitere Variable anpassen, die ebenfalls in der Nähe der obigen Stellen erscheint:

```
$size  = mt_rand(15, 21);
```

Diese Zeile bewirkt, dass das Plugin zufällig eine Schriftgröße zwischen 15 und 21 für jeden einzelnen Buchstaben benutzt. Sie können diese beiden Zahlen auf den gewünschten Wert erhöhen (z. B. 21 und 26), um größere Schriften anzuzeigen.

Das Plugin verzichtet bei der Ausgabe der Schriften auf einige Zeichen, die häufig missverständlich aussehen. Die Zahlen 1, 5, 6 und 8 sowie die Buchstaben I, O und S werden daher ausgelassen. Wenn Sie weitere Buchstaben ausschließen wollen, können Sie auch diese in der Spamschutz-Datei ändern. Suchen Sie dafür folgende Code-Stelle und passen Sie sie an:

```
function random_string($max_char, $min_char) {
    $this->chars = array(2, 3, 4, 7, 9);
    // 1, 5, 6 and 8 may look like characters.
    $this->chars = array_merge($this->chars, array('A','B','C','D','E'
,'F','H','J','K','L','M','N','P','Q','R','T','U','V','W','X','Y','Z'));
    // I, O, S may look like numbers
```

6.1.2 Browser-Kompatibilität serendipity_event_browsercompatibility

Eines der simpelsten Ereignis-Plugins ist das Plugin namens *Browser-Kompatibilität*. Es wurde damals erstellt, um ein gravierendes Problem des Microsoft Internet Explorer 6 zu beheben. Dieser Browser konnte standardmäßig keine Grafiken mit mehrstufiger transparenter Hintergrundfarbe anzeigen, während alle anderen am Markt verfügbaren Browser damit kein Problem hatten.

Leider war (und ist) dieser Browser stark verbreitet, aber Serendipity sollte dennoch in der Lage sein, Icons mit transparentem Hintergrund in der Oberfläche anzuzeigen. So sind beispielsweise die Grafiken des Uhr-Symbols oder des Hammers transparente Grafiken, die bei jeder Hintergrundfarbe eingebunden werden können.

Mit Hilfe dieses Plugins konnte eine einfache Regel in die Ausgabe von Serendipity aufgenommen werden, so dass die Grafiken dargestellt werden können. Ursprünglich war geplant, auch weitere Browser-Unterschiede mit diesem Plugin auszubügeln. Dies war jedoch seither nie wieder nötig.

Wenn in Zukunft Version 7 des Internet Explorers noch größere Verbreitung hat, wird dieses Plugin also nicht mehr erforderlich sein. Auch heutzutage ist es eigentlich nur dann notwendig, wenn Sie die Administrationsoberfläche mit Internet Explorer 6 ohne merkwürdige Hintergrundfarben einsetzen wollen.

Einen gravierenden Nachteil hat dieses Plugin jedoch. Das Plugin sorgt dafür, dass der Browser eine PNG-Grafik intern als Hintergrundbild interpretiert und eine unsichtbare GIF-Grafikdatei darüberlegt. Das klappt nur dann einwandfrei, wenn ein Bild im HTML-Code so eingebunden ist, dass die *width*- und *height*-Angaben des *¡img¿*-Tags vorhanden sind. Fehlt diese Angabe, wird die PNG-Grafik womöglich gestaucht oder in die Länge gestreckt. Wenn Sie also später PNG-Grafiken in Ihrem Blog hinzufügen, achten Sie immer darauf, dass die width/height-Angaben vorhanden sind. Oder benutzen Sie alternativ das GIF- oder JPEG-Grafikformat, wenn die Transparenz nicht wichtig ist. Eine weitere Möglichkeit bestünde natürlich darin, das Plugin *Browser-Kompatibilität* zu entfernen und darüber hinwegzusehen, dass Besucher mit dem alten Internet Explorer 6 möglicherweise eine pinke Hintergrundfarbe statt Transparenz sehen.

Das Plugin besitzt keine Konfigurationsoptionen.

6.1.3 Textformatierung: Serendipity serendipity_event_s9ymarkup

Das Textformatierungs-Plugin *Serendipity* bietet einige ganz einfache Umwandlungen für Ihre Artikeltexte an. Diese Umwandlungen sind an alte Mailbox-Hervorhebungsmöglichkeiten angelehnt und stammen noch aus einer Zeit, in der Formatierung via HTML gänzlich unbekannt war. Daher haben sich viele alte Hasen an derlei Formatierung gewöhnt und können damit leichter umgehen als mit HTML-Code:

- `*Wort*` formatiert ein Wort fett.
- `_Wort_` unterstreicht ein Wort.
- `^Wort^` setzt ein Wort hochgestellt.
- `@Wort@` setzt ein Wort tiefgestellt.

Aufgrund dieser besonderen Konventionen kann das Serendipity-Textformatierungs-Plugin besonders bei Artikeln mit Quellcode-Inhalten Probleme verursachen. Wenn Sie in einem Artikel PHP-Code zitieren, könnten die oben aufgeführten Sonderzeichen ungewollte HTML-Formatierungen auslösen.

Sie können dieses Problem meist beheben, indem Sie das Plugin im Bedarfsfall für einen einzelnen Beitrag deaktivieren (über die zusätzlichen Artikeloptionen des Plugins *Erweiterte Eigenschaften für Artikel*, siehe Seite 262) oder indem Sie es vollständig deinstallieren. Wenn Sie das Plugin in Zusammenhang mit anderen Quellcode-Hervorhebungs-Plugins benutzen, achten Sie möglichst darauf, dass das Serendipity-Textformatierungs-Plugin als letztes Textformatierungs-Plugin in der Plugin-Liste aufgeführt wird.

6.1.4 Textformatierung: Smilies serendipity_event_emoticate

Das wohl gebräuchlichste Textformatierungs-Plugin nennt sich *Smilies*. Es ersetzt typische Smiley-Zeichenketten wie :-) durch einen grafischen Smiley.

Wenn Sie die Konfigurationsoptionen des Plugins aufrufen, sehen Sie eine Liste aller verfügbaren Smiley-Umwandlungen. Auch Besucher können, falls gewünscht, in ihren Kommentaren auf die Smiley-Grafiken zurückgreifen.

Vielfach wird den Serendipity-Smileys vorgeworfen, dass sie ziemlich hässlich aussähen. Zum einen ist das natürlich Geschmackssache, aber zum Teil kann man das auch nicht ganz von der Hand weisen. Da Serendipity so freizügig wie möglich lizensiert wurde, können nur Smiley-Grafiken mit Serendipity ausgeliefert werden, die derselben Lizens unterliegen. Die meisten Smileys, die im Internet zu haben sind, sind jedoch entweder nur kommerziell oder inkompatibel lizensiert oder von den Seiten ohne Befugnis eingebaut. Da die Smiley-Grafiken auch mit einem Serendipity-Template zusammen gebündelt werden können, gibt es auch einige (wenige) Templates mit eigenen Grafikdateien: *GreenMile*, *kamouflage* und *truth*.

Als Betreiber des Blogs können Sie aber glücklicherweise die Grafiken einfach ersetzen. Das Plugin versucht Ihnen das so einfach wie möglich zu machen, daher gibt es mehrere Möglichkeiten, eigene Smileys einzubinden.

Variante 1: Grafikdateien ersetzen

Die einfachste Methode ist, die Smiley-Grafikdateien von Serendipity durch eigene zu ersetzen. Diese Dateien befinden sich standardmäßig im Verzeichnis `templates/default/emoticon`
Die Dateien haben eine `.png`-Dateiendung, Sie können aber auch die Smiley-üblichen `.gif`-Dateien hochladen, wenn Sie in der Konfiguration des Smilie-Plugins danach als **Dateiendung** für Smileys auch `.gif` eintragen.

Sie können die eigenen Smiley-Dateien entweder direkt in das genannte Verzeichnis hochladen oder auch in ein gleichnamiges Unterverzeichnis eines eigenen Templates. Wenn Sie die Dateien in ein eigenes Template-Verzeichnis hochladen, werden die Grafiken automatisch von dort eingebunden und Sie haben den Vorteil, später die Grafiken zusammenhängend mit Ihrem Template archivieren oder verteilen zu können.

Variante 2: Smileys erweitern

Beim Ersetzen von Smileys können Sie natürlich keine neuartigen Smileys hinzufügen, sondern nur die bestehenden überarbeiten.

Um eigene zu erstellen, kann das Plugin eine Datei `emoticons.inc.php` auswerten. Diese Datei muss ein Array enthalten, in dem Smileys beschrieben und einer Grafik zugewiesen werden.

Standardmäßig würde eine Datei so aussehen:

```
<?php
$serendipity['custom_emoticons'] = array(
```

```
 ":'("  => serendipity_getTemplateFile('img/emoticons/cry.png'),
 ':-)'  => serendipity_getTemplateFile('img/emoticons/smile.png'),
 ':-|'  => serendipity_getTemplateFile('img/emoticons/normal.png'),
 ':-O'  => serendipity_getTemplateFile('img/emoticons/eek.png'),
 ':-('  => serendipity_getTemplateFile('img/emoticons/sad.png'),
 '8-)'  => serendipity_getTemplateFile('img/emoticons/cool.png'),
 ':-D'  => serendipity_getTemplateFile('img/emoticons/laugh.png'),
 ':-P'  => serendipity_getTemplateFile('img/emoticons/tongue.png'),
 ';-)'  => serendipity_getTemplateFile('img/emoticons/wink.png'),
);
?>
```

Pro Zeile sehen Sie jeweils den Text, der durch eine Grafik ersetzt werden soll, und danach auf der rechten Seite einen Aufruf, der auf die entsprechende Grafikdatei verweist.

Wenn Sie sich mit regulären Ausdrücken (siehe Seite 247) auskennen, können Sie auch Smileys mit solchen Ausdrücken definieren. Damit ist es leichter, diverse Plugin-Alternativen zu beschreiben, z. B. dass sowohl :-) als auch :) zur selben Grafik umgewandelt werden. Damit das Array `$serendipity['custom_emoticons']` diese regulären Ausdrücke nutzen kann, müssen Sie eine weitere Variable

```
$serendipity['custom_emoticons_regexp'] = true;
```

innerhalb der `emoticons.inc.php` definieren.

6.1.5 Textformatierung: NL2BR serendipity_event_nl2br

Das Plugin *NL2BR* ist ein ebenfalls recht simples Textformatierungs-Plugin. Es sorgt einfach dafür, dass von Ihnen eingegebene Zeilenumbrüche in einem Beitrag später in einen korrekten HTML-Zeilenumbruch (`<br />`) umgewandelt werden.

Das ist notwendig, weil der HTML-Standard normale Zeilenumbrüche nicht als solche erkennt. Auch mehrere hintereinander eingetragene Leerzeichen werden von HTML als ein einziges Leerzeichen zusammengefasst.

Wenn Sie also ohne dieses Plugin Beiträge schreiben, müssten Sie selber die korrekten HTML-Absätze (entweder mittels `<p>...</p>` oder `<br />`) einfügen. Je nachdem, ob Sie einen WYSIWYG-Editor einsetzen, tut dies der Editor auch bereits selbständig.

Wenn Sie also bei selbständiger Eingabe überflüssige Zeilenumbrüche in Ihren Artikeln haben, sollten Sie das *NL2BR*-Plugin deinstallieren.

Abhängig vom eingesetzten Template kann es sein, dass das Template die Abstände zwischen Absätzen (den `<p>`-Tags) mittels CSS-Formatierungen deaktiviert. Die eigentlich zu erwartenden Leerzeilen zwischen zwei Absätzen würden somit also unterdrückt werden. In älteren

Templates wurde dies hauptsächlich deshalb eingefügt, um doppelte Zeilenumbrüche in Verbindung mit dem *NL2BR*-Plugin zu vermeiden. Wenn Sie ein derartiges Template einsetzen, können Sie eine einfache CSS-Formatierung am Ende der `style.css`-Datei im entsprechenden `templates`-Unterverzeichnis einfügen:

```
.serendipity_entry p {
  margin: 1em;
}
```

Dadurch wird der Abstand (*margin*) zwischen Paragraphen auf eine relative Einheit gesetzt, die dem üblichen Absatzabstand entspricht.

Ein weiteres Problem kann auftreten, wenn Sie in einem Beitrag JavaScript oder anderweitigen Sourcecode platzieren möchten. Denn auch hier würde das Plugin relativ stur sämtliche Leerzeichen durch `<br />` umwandeln und dadurch JavaScript ungültig machen oder Ihre spezielle Sourcecode-Formatierung in einem `<blockquote>`-HTML-Konstrukt mit überflüssigen HTML-Zeilenumbrüchen stören. Es gibt mehrere Möglichkeiten (abgesehen vom Deinstallieren des NL2BR-Plugins), dieses Problem zu beheben:

- NL2BR-Plugin fallweise für einen einzelnen Artikel deaktivieren. Dies können Sie beim Erstellen eines Artikels im Abschnitt *Erweiterte Optionen* erledigen, wenn Sie das Ereignis-Plugin *Erweiterte Eigenschaften von Artikeln* installiert haben.

- In der Konfiguration des NL2BR-Plugins die Liste von geschützten HTML-Tags so ändern, dass keine Zeilenumbrüche zwischen ungewünschten Tags eingefügt werden. So können Sie z. B. `script` mit in die kommaseparierte Tagliste aufnehmen, damit das NL2BR-Plugin keine Zeilenumbrüche in JavaScript-Containern einfügt.

- Bei der Reihenfolge der Textformatierungs-Plugins darauf achten, dass das NL2BR-Plugin als letztes ausgeführt wird. Dadurch wird verhindert, dass die Zeilenumbrüche zu früh eingefügt werden und möglicherweise andere Textformatierungs-Plugins durcheinanderbringen.

6.2 Weitere mitgelieferte Plugins

Über die bereits vorinstallierten Ereignis-Plugins hinaus liefert Serendipity noch einige weitere Ereignis-Plugins mit, die Sie direkt über die Plugin-Verwaltung installieren können.

Beachten Sie, dass bei Textformatierungs-Plugins die Reihenfolge der Plugins für die endgültige Formatierung maßgeblich sein kann (siehe Seite 237).

6.2.1 Textformatierung: BBCode serendipity_event_bbcode

BBCode ist eine sehr weit verbreitete Möglichkeit, um einfachste Formatierungen (fett, kursiv, Hyperlinks) durchzuführen. Grundsätzlich könnte man sich natürlich auch die nicht viel komplizierteren HTML-Tags aneignen, aber aus mehreren Gründen sollte man darauf verzichten.

BBCode wird seltener von den Redakteuren selbst genutzt, sondern meist von Besuchern, die ihre Kommentare im Blog mittels BBCode formatieren können. HTML stellt ein grundlegendes Sicherheitsrisiko dar, wenn beliebige Besucher derartige Formatierungen auf die Webseite setzen können. Denn in HTML kann man JavaScript einbinden, das ungewünschte, sicherheitsrelevante Funktionen auslösen kann – so könnten möglicherweise Logindaten ausgespäht werden, oder der Inhalt der Seite könnte von gewitzten Benutzern vollständig abgeändert werden. Kurzum, als Betreiber eines Blogs sollte man es vermeiden, HTML-Kommentare zuzulassen. Serendipity verhindert dies aus Sicherheitsgründen vollständig, daher ist die einzige Möglichkeit zur Formatierung von Kommentaren die Benutzung von Standards wie BBCode. Denn diese Formatierungen können problemlos und sicher in HTML umgesetzt werden. BBCode ist aufgrund seiner hohen Verbreitung so allgegenwärtig, dass die Kommentatoren meist Kenntnis davon haben. Nicht nur Blogs, sondern auch Internetforen oder auch E-Mail-Programme „sprechen“ oft BBCode.

Auch bei der Artikelerstellung kann BBCode hilfreich sein – die Syntax, um eine HTML-Auflistung zu erstellen, ist wesentlich unkomplizierter als entsprechender HTML-Code.[8] BBCodes sind immer von eckigen (statt bei HTML spitzen) Klammern umgeben. Um ein Wort zu fetten, würde man `[b]Wort[/b]` verwenden, Bilder kann man mittels `[img]http://www.example.com/bild.jpg[/img]` einbinden und Hyperlinks via `[url]http://www.example.com[/url]`. Die erwähnten Auflistungen kann man mittels `[list] [*]Punkt [*]Punkt ... [/list]` formatieren.

In den Konfigurationsoptionen des Plugins können Sie zudem einstellen, ob von BBCode formatierte Links standardmäßig in einem neuen Fenster geöffnet werden sollen.

6.2.2 Textformatierung: Textile serendipity_event_textile

Textile ist in Grundzügen ähnlich zu BBCode, benutzt jedoch eine leicht andersartige (Wiki-ähnliche) Syntax, die manche Personen bevorzugen. Auf eckige Klammern wird zugunsten von Formatierungen wie `"Mein Blog":http://www.example.com/serendipity/` verzichtet. Darüber hinaus bietet Textile eine weitaus höhere Abstraktion als BBCode. Während BBCode so einfach wie möglich gehalten ist, bietet Textile eine Flexibilität, mit der man fast ganz auf HTML verzichten kann.[9]

[8] Eine Auflistung von BBCodes gibt `http://de.wikipedia.org/wiki/BBcode`.

[9] Unter `http://thresholdstate.com/articles/4312/the-textile-reference-manual` finden Sie die vollständige Textile-Syntax.

Textile ist eine relativ komplexe Bibliothek, die auch mehr Ressourcen verbraucht als das BBCode-Plugin. Daher wird sie meist eher von Redakteuren benutzt als von Kommentatoren.

Da Textile auch weitaus mehr Formatierungsmöglichkeiten für beliebiges HTML enthält, sollten Sie sich gut überlegen, ob Sie diese Flexibilität auch den Kommentatoren anbieten wollen, die so möglicherweise das Layout innerhalb der Kommentare durcheinanderbringen könnten.

In den Konfigurationsoptionen des Plugins können Sie einstellen, ob die Textile-Bibliothek in Version 1.0 oder 2.0 genutzt werden soll.

6.2.3 Textformatierung: Wiki serendipity_event_textwiki

Eine sehr verbreitete und beliebte Form der Textauszeichnung stellt das sogenannte *Wiki markup* dar. Diese Formatierungsart hat aufgrund des Wikipedia-Booms hohe Verbreitung gefunden. Da Wikis erschaffen wurden, um auch Leuten ohne HTML-Kenntnisse die Möglichkeit zu bieten, gemeinsam formatierte Texte zu erfassen, ist die Syntax dieses Plugins ebenfalls einfach gehalten.

Serendipity benutzt hierfür das mitgelieferte PEAR Text::Wiki-Paket, um die Wiki-Syntax in HTML umzuformen. Dieses PEAR-Paket hat eine gewaltige Anzahl an Konfigurationsoptionen, mit denen Sie die möglichen Eingaben und Ausgaben des Plugins kontrollieren können. Alle Optionen entsprechen dabei den Optionen, die das PEAR-Paket bereitstellt.

Die vollständige Dokumentation der Syntax und der verfügbaren Optionen befindet sich auf `http://wiki.ciaweb.net/yawiki/index.php?area=Text_Wiki`.

Bei der Verwendung des Wiki-Plugins gilt es zu beachten, dass dieses Plugin sämtliche HTML-Formatierungen eines Artikels umwandelt. So können Sie als Redakteur keinerlei manuelles HTML mehr einfügen. Dieses Verhalten können Sie jedoch in der Konfiguration des Plugins abstellen (*Html: Ja*). Weiterhin können Sie über die Plugin-Optionen auch alle anderen Umwandlungsvarianten flexibel ein- oder ausschalten.

6.2.4 Textformatierung: Externe Links zählen serendipity_event_trackexits

Serendipity kann für Statistiken zählen, wie oft Ihre Besucher auf externe Links in Ihrem Blog geklickt haben. So ist es für Sie als Betreiber möglich, herauszufinden, welche von Ihnen genannten URLs für Ihre Besucher besonders interessant sind.

Sobald ein Besucher eine externe URL klickt, verlässt er Ihr Blog. Das bedeutet, dass Serendipity üblicherweise gar nicht herausfinden kann, wenn ein Besucher Ihr Blog verlässt, denn dann findet kein Aufruf der Seiten auf Ihrem Server mehr statt.

Um dies zu umgehen, müssen also fremde URLs so verändert werden, dass ein Klick darauf erst Ihrem Blog die angeklickte Seite mitteilt und dann erst die gewünschte Seite aufgerufen

wird. So ein Mechanismus nennt sich *Link Tracking*. Eine externe URL wie `http://www.google.de/` wird dazu speziell aufbereitet, so dass in Ihrem Artikel ein Link wie `http://www.example.com/serendipity/exit.php?entry_id=1&url_id=1` erscheint.

Die Datei `exit.php` erkennt aufgrund der URL-Variablen, welche Seite angefordert wurde. So kann in einer Datenbanktabelle die Anzahl der Klicks zu der gewünschten URL mitgezählt und dann der Browser des Besuchers zu der gewünschten Seite weitergeleitet werden.

Diese Methode ist die einzige Möglichkeit, externe Links nachzuverfolgen. Die automatische Umformung von Links in das notwendige Format nimmt das Plugin *Externe Links zählen* vor. Ist dieses aktiviert, wird jeder HTML-Code wie `<a href='http://www.example.com'>Google</a>` erfasst und umgeschrieben.

Dieses Umschreiben hat für Sie Vorteile in der Statistik, für den Benutzer jedoch mehrere Nachteile. Zum einen kann ein Besucher so eine URL nicht mehr eindeutig identifizieren. Ob er nach einem Klick darauf wirklich auf der Seite von Google landet, weiß er vorher noch nicht. Daher werden viele Besucher (zu Recht) misstrauisch, wenn sie auf derart merkwürdig formatierte Links stoßen. Den Besuchern wird dadurch auch sofort klar, dass sie statistisch „ausgespäht“ werden – und letztlich kann ein Besucher auch nicht einfach per Kopieren und Einfügen den Link übernehmen, sondern er muss ihn erst aufrufen, um herauszufinden, auf welcher URL er landet.

Überlegen Sie sich also beim Einsatz dieses Plugins gut, ob der statistische Nutzen für Sie ausschlaggebend ist. In älteren Versionen Serendipitys war dieses Plugin noch ein Standard-Plugin, wurde aber aufgrund der Kritik von Blog-Besuchern aus der Standardinstallation entfernt.

Wenn ein Besucher bei einem Kommentar seine Homepage angibt, kann auch diese Homepage umgeformt und in der Statistik nachverfolgt werden. Dies hat den Vorteil, dass potenzielle Spammer die angegebene Homepage nicht mit dem Suchmaschinenrang Ihres Blogs aufwerten können. In der Konfiguration des Plugins können Sie dieses Verhalten einstellen: Sie können die Nachverfolgung von Kommentatoren-Homepages entweder ausstellen (**keine**), Sie können das angebotene Link Tracking aktivieren (**Serendipity Exit-Tracking Routine**) oder auch auf eine Maskierung von Google (**Google PageRank Deflector**) zurückgreifen. Diese Methode ist ein ähnliches Vorgehen, wie von der NoFollow-Initiative[10] gefordert – man verbietet so den Kommentatoren, vom Suchmaschinen-Wert (dem *Google PageRank*) zu profitieren. Die berechtigte Kritik an diesem Vorgehen ist jedoch, dass man aufrichtige Kommentatoren ruhig durch die Verlinkung auf ihre Seiten belohnen soll. Ohne derartige Verlinkungen, die von Suchmaschinen berücksichtigt werden, wären die Blogs von heute bei weitem nicht so verbreitet und weniger relevant bei Suchergebnissen.

[10] `http://de.wikipedia.org/wiki/Nofollow`

6.2.5 Übliche XHTML-Fehler beseitigen serendipity_event_xhtmlcleanup

Da XHTML an XML angelehnt ist, besteht wenig Fehlertoleranz gegenüber Syntaxfehlern. Sollte ein Redakteur also versehentlich ungültiges XHTML produzieren, kann dies dazu führen, dass trotz barrierefreier Templates die Seite nicht mehr validiert.

Abgesehen von logischen Fehlern (HTML-Tags öffnen und sie nicht wieder schließen oder bei verschachtelten HTML-Tags die Reihenfolge vertauschen), kann das Serendipity-Plugin *Übliche XHTML-Fehler beseitigen* dabei helfen, einige gängige Fehlerquellen auszuräumen:

- Das Sonderzeichen & darf bei XHTML nur dann benutzt werden, wenn daraufhin ein HTML-Sonderzeichen folgt, also beispielsweise `>` (¿), `&euro;` (Euro-Symbol) oder andere. Wenn Sie jedoch das Zeichen & auch so in einem Artikel anzeigen wollen, müssen Sie `&` benutzen. Normalerweise geben Redakteure selten HTML-Sonderzeichen ein und wissen nicht von der Regel, das &-Zeichen nicht alleine zu setzen. Das Plugin kann sich daher darum kümmern, dass ein allein stehendes & immer zu `&` umgewandelt wird.
- Alle XHTML-Tags müssen immer ein schließendes Element verwenden. Ein `<p>` ist daher nur gültig, wenn auch ein schließendes `</p>` folgt. Das früher verbreitete HTML-Tag `<br>` musste, da es kein schließendes Element voraussetzt, zu `<br />` verändert werden. In den häufigsten Fällen wird das `/>`-Zeichen bei den Tags `<img>`, `<hr>` und eben jenem `<br>` vergessen. Das Plugin kann bei diesen Tags das fehlende `/` nachreichen.
- XHTML-Bilder-Tags (`<img>`) benötigen stets ein `alt`-Attribut, das sehbehinderten Benutzern beschreibt, was ein Bild darstellt. Dieses ALT-Tag wird aufgrund des höheren Aufwands von vielen Redakteuren jedoch vernachlässigt oder vergessen. Das Plugin kann sicherstellen, dass zumindest immer ein leeres `alt`-Attribut gesetzt wird. Wenn Sie die Option **Encode XML-parsed data** aktivieren, wird zudem sichergestellt, dass innerhalb eines Bild-Links keine ungültigen Sonderzeichen erscheinen können.
- Bei Webseiten im UTF-8-Format sind HTML-Sonderzeichen wie `&auml;` (ä) nicht erlaubt. Das Plugin kann solche Sonderzeichen in die korrekten UTF-8-Sonderzeichen umwandeln, wenn Sie die Option **Cleanup UTF-8 entities** aktivieren.

Wenn Sie Wert darauf legen, dass Ihre Seite dem XHTML-Standard entspricht, sollten Sie dieses Plugin installieren.

6.2.6 Wort-Ersetzer serendipity_event_contentrewrite

Das Plugin *Wort-Ersetzer* ist ein sehr komplexes Plugin, das es Ihnen ermöglicht, beliebige Wörter von Kommentaren oder Blog-Artikeln umzuwandeln. Sehr beliebt ist das Plugin daher

bei schreibfaulen Redakteuren oder Personen, die gerne mit Akronymen um sich werfen.

Jedes Mal, wenn Sie in einem Artikel beispielsweise die Zeichenfolge `s9y` verwenden, kann das Plugin dafür sorgen, dass dies umgewandelt wird zu `<a href='http://www.s9y.org/' title='s9y'>s9y</a>`. Man sieht also bereits, welches Einsparungspotenzial das Plugin bietet, solange Sie das zu Grunde liegende Wörterbuch gut pflegen.

Die Eingabemaske für diese Wörterbücher erreichen Sie über die Konfiguration des Plugins. In dieser Oberfläche können Sie beliebig viele Wörter erfassen, die jeweils eine *Quelle* und ein *Ziel* darstellen, die anhand der konfigurierten *Umformungsmaske* speziell umgeschrieben werden.

Jedes Mal, wenn Sie einen neuen **Titel** und eine neue **Beschreibung** eintragen und das Plugin speichern, wird dem Wörterbuch ein neues Wort hinzugefügt. Um ein Wort aus dem Wörterbuch zu entfernen, können Sie es in der Konfigurationsmaske einfach aus dem Eingabefeld löschen.

Bevor Sie jedoch eine *Quelle* und ein *Ziel* für die Wortersetzung eintragen, müssen Sie global festlegen, wie die beiden Eingaben behandelt werden sollen. Die *Quelle* legt das Wort fest, das Sie selbst in einem Artikel eintragen. Das *Ziel* ist später das Wort, das anstelle Ihrer Eingabe erscheinen oder ergänzt werden soll.

Mittels der **Umformungsmaske** legen Sie fest, wie die Wortquelle und das Wortziel später umformatiert werden sollen. Dies ist weitaus komfortabler, als jedes Mal den vollständigen Zieltext festzulegen. Stellen Sie sich vor, Sie würden analog zur Umformung des Wortes `s9y` auch das Wort `PHP` so umwandeln, dass ein Link darauf gesetzt wird. Als Umformungsmaske würde man hier Folgendes festlegen: `<a href='{ziel}' title='{quelle}'> {quelle}</a>`

Tragen Sie nun als **Neuer Titel** das Wort `s9y` ein, als **Neue Beschreibung** geben Sie `http://www.s9y.org/` ein. Speichern Sie das Plugin, und Sie können das nächste Wörterpaar `PHP` und `http://www.php.net` eintragen.

Die Umformungsmaske wird also in Zukunft bei jedem Vorkommen der eingetragenen *Quelle* das entsprechende *Ziel* heraussuchen und im Inhalt einsetzen.

Für jedes Plugin kann nur eine Umformungsmaske eingetragen werden. Sie können daher mehrere *Wort-Ersetzer*-Plugins installieren. So wäre es also auch denkbar, als **Umformungsmaske** nur `{ziel}` einzutragen, damit Sie z. B. das Wort `gh` (als *Quelle*) immer einfach nur mit `Garvin Hicking` (als *Ziel*) ersetzen.

Das **Rewrite-Zeichen** hat eine besondere Bedeutung. Wenn Sie im Wörterbuch des Plugins eine *Quelle* eintragen, die in gewöhnlichem Text vorkommen kann, dann möchten Sie nicht immer, dass das Wort ersetzt wird. Mit dem hier konfigurierten Sonderzeichen können Sie erreichen, dass ein Wort nur dann ersetzt wird, wenn am Ende das hinterlegte Sonderzeichen auftaucht. Wenn Sie also `*` als **Rewrite-Zeichen** eintragen, würde im obigen Beispiel nur `PHP*` und `s9y*` ersetzt werden. Eine Nennung ohne das Rewrite-Zeichen würde das Wort unangetastet lassen.

Das *Wort-Ersetzer*-Plugin können Sie vom *Ausgabe-Wrapper*-Plugin (siehe Seite 213) in der

Seitenleiste ausgeben lassen. Es zeigt Ihnen dann das Wörterbuch der eingetragenen *Quellen* und *Ziele*. Dieses Wörterbuch wird auch in der Konfigurationsmaske des Plugins am Ende der Seite eingebunden.

Während dieses Plugin relativ abstrakt gehandhabt wird, gibt es zwei besondere Plugins, die die Thematik der Wortersetzung anders handhaben: zum einen das *Glossary*-Plugin (`serendipity_event_glo` mit dem Sie eine einfache Wortliste hinterlegen können, die beim Auftauchen in einem Text hervorgehoben wird. Zum anderen das Plugin *Markup: RegexpMarkup*, mit dem Sie komplexe reguläre Ausdrücke zum Ersetzen von Wörtern definieren können. Letztgenanntes Plugin bietet so die Möglichkeit, automatisch Links mit vorangestelltem *http://* anklickbar zu machen oder auch alle Wörter mittels spezieller Formatierung (z. B. `[(Bismarck)]`) so umzuformen, dass ein Klick darauf den entsprechenden Eintrag in der Wikipedia öffnet.

6.2.7 Erweiterte Eigenschaften von Artikeln serendipity_event_entryproperties

Das Plugin *Erweiterte Eigenschaften von Artikeln* ist ein sehr mächtiges Ereignis-Plugin, das Ihnen eine große Vielfalt an Möglichkeiten eröffnet. Zum einen bietet dieses Plugin mehrere Optionen bei der Erstellung von Einträgen an, und zum anderen kann es dafür sorgen, Artikel zu cachen und somit schneller darzustellen.

Die Konfigurationsoptionen des Plugins sind:

Artikel cachen?

Wenn Sie mehrere Textformatierungs-Plugins benutzen, werden diese nacheinander bei jeder Darstellung eines Artikels erneut durchgeführt. Meistens verschwendet Serendipity bei dieser erneuten Ausführung einiges an Ressourcen, die Sie aber mit diesem Plugin einsparen können.

Wenn Sie die Caching-Option aktivieren, wird das Plugin beim Speichern eines Artikels die Ausgabe aller Textformatierungs-Plugins auswerten und speichern. Bei der Anzeige eines Artikels wird dann nur diese Version geholt – alle Textformatierungs-Plugins werden übersprungen. Wenn Sie einen Artikel überarbeiten, wird der Cache automatisch neu gefüllt.

Jedoch kann das Aktivieren des Cachings auch Probleme mit sich bringen. Wenn Sie den Inhalt eines Artikels direkt über die Datenbank bearbeiten oder einmal ein neues Textformatierungs-Plugin installieren, kann der Cache vom aktuellen Stand abweichen. Daher können Sie den Cache über den Menüpunkt **Einträge** → **Cachen aller Artikel** neu aufbauen lassen. Dies sollten Sie immer dann tun, wenn Sie ein Textformatierungs-Plugin entfernen oder hinzufügen.

Auch für das Caching ist die Reihenfolge der Ereignis-Plugins von entscheidender Bedeutung. Das Plugin kann nur die Ausgaben der Textformatierungs-Plugins cachen, die in der Reihenfolge vor diesem Plugin stehen. Alle danach aufgeführten Plugins werden weiterhin ausgeführt und nicht gecached.

Diese Tatsache können Sie ausnutzen, um Textformatierungs-Plugins hinter diesem Plugin zu positionieren, wenn sie ohne Caching ausgeführt werden sollen. Derartige Textformatierungs-Plugins gibt es jedoch wenige. Als Faustregel gilt, dass man ein Plugin nicht cachen sollte, wenn es die Ausgabe eines Artikels abhängig von anderen Daten macht. Wenn ein Plugin also beispielsweise die Farbe eines Links abhängig von der aktuellen Tageszeit verändert, wäre ein Caching denkbar ungeeignet.

Denken Sie also vor allem bei der Erstellung eines eigenen Plugins an diese Caching-Option, falls dessen Ausgaben Ihnen nicht korrekt vorkommen.

Leserechte auf Gruppen/Benutzer beschränken

Bevor Serendipity es ermöglichte, Leserechte auf Kategorie-Ebene zu vergeben, konnte man mit diesem Plugin pro Eintrag bestimmen, von wem er gelesen werden darf.

Diese besonders flexible Leserecht-Setzung bietet das Plugin nach wie vor an, wenn Sie die Option **Leserechte auf Gruppen beschränken** und/oder **Leserechte auf Benutzer beschränken** auswählen. Der Ressourcenbedarf bei der Datenbankabfrage ist relativ hoch, daher sollten Sie diese Optionen nur aktivieren, wenn Sie individuelle Leserechte tatsächlich benötigen.

Standard: Artikel können gelesen werden von

Wenn Sie die Beschränkung der Leserechte aktivieren, können Sie mit der Option **Standard: Artikel können gelesen werden von** festlegen, welche Leserechte ein neu erstellter Artikel standardmäßig besitzt. **Co-Autoren** bedeutet, dass jeder eingeloggte Besucher einen Artikel lesen darf.

Freie Felder

Die sogenannten *Freien Felder* (oder *Custom Fields*) bieten eine sehr praktische Möglichkeit, um beliebige weitere Eingabefelder zu einem Artikel auszufüllen.

In das große Eingabefeld **Freie Felder** können Sie eine Liste von kommaseparierten Feldnamen eintragen. Die Groß- und Kleinschreibung dieser Feldnamen ist später von Bedeutung, außerdem sollten Sie bei einem Feldnamen auf Leer- und Sonderzeichen verzichten.

Für jedes hier eingetragene Feld wird später beim Erstellen eines Artikels ein eigenständiges Eingabefeld eingebunden. Dort können Sie dann genauso wie beim *Artikeltext* oder *Erweiterten Eintrag* beliebigen Text eintragen.

Später können Sie die eingetragenen Felder an beliebigen Stellen in der Artikelausgabe des Frontends einbinden, indem Sie die Template-Datei `entries.tpl` bearbeiten. Weitere Informationen zum Einbau von Freien Feldern können Sie auf Seite 580 nachschlagen. Um Artikel mit bestimmten Eigenschaften innerhalb eines Templates darzustellen, können Sie den Parameter `entryprops` der Smarty-Funktion `serendipity_fetchPrintEntries` nutzen, wie auf Seite 572 beschrieben.

Wenn das Plugin konfiguriert und aktiviert ist, können Sie im Bereich **Erweiterte Optionen** beim Erstellen oder Bearbeiten eines Blog-Artikels einige Einstellungen tätigen:

Dauerhafte Artikel

Wenn ein Artikel als *Dauerhafter Artikel* (*Sticky*) markiert ist, wird er im Frontend immer als erster Artikel angezeigt. Ein Blog-Beitrag kann durch diese Markierung hervorgehoben werden und erscheint so außerhalb der üblichen chronologischen Übersicht. Oft wird dies für besonders wichtige Artikel verwendet, oder als eine Art Einführung zum Blog.

Nicht in Artikelübersicht zeigen

Die Aktivierung dieser Option bewirkt, dass ein Artikel nicht in der Artikelübersicht dargestellt wird. Er kann dann von einem Besucher nur gefunden werden, wenn er sich in der Ansicht der zugehörigen Kategorie des Eintrages befindet oder nach einem Artikel sucht.

Eintragsinhalt im RSS-Feed verstecken

Wenn Sie nicht wollen, dass ein Blog-Beitrag mit im RSS-Feed ausgeliefert werden soll, können Sie diese Option aktivieren.

Artikel können gelesen werden von:

Mit diesem Auswahlfeld können Sie festlegen, ob ein Artikel nur von eingeloggten Benutzern gelesen werden kann, nur von Ihnen selbst oder von allen Besuchern. Diese Option wird nur dann angezeigt, wenn Sie in den Optionen des Plugins die Beschränkung der Leserechte aktiviert haben.

Passwort

Sie können einen Artikel vor unbefugten Lesern schützen, indem Sie ein Passwort für einen Artikel vergeben. Der Besucher kann dann in der Artikelübersicht zwar den normalen Artikeltext wie gewöhnlich lesen, aber der *Erweiterte Eintrag* und die Detailansicht des Artikels können nur aufgerufen werden, wenn der Besucher das richtige Passwort in einer dargestellten Box einträgt.

Autor

Wenn ein Redakteur einen Beitrag erstellt, wird er als der Eigentümer des Artikels festgelegt. Der Artikel kann danach nur noch vom Eigentümer oder berechtigten Benutzergruppen gelesen werden.

In manchen Fällen möchten Sie den Eigentümer eines Artikels gerne ändern. Das können Sie mithilfe dieses Plugins tun. Nur Chefredakteure bzw. Administratoren haben (abgesehen vom derzeit eingetragenen Besitzer) die Befugnis, den Autoren zu verändern.

Der hier eingestellte Autor ist auch derjenige, der in der Artikelübersicht als Autor aufgeführt wird.

Disable Markup plugins for this entry

Standardmäßig werden alle installierten Textformatierungs-Plugins auf einen Artikel angewendet. In manchen Fällen kann es jedoch erforderlich sein, dass gewisse Textformatierungen nicht ausgeführt werden, beispielsweise wenn Sie HTML-Quellcode

oder JavaScript in einem Beitrag verwenden wollen und es durch ein möglicherweise installiertes Wiki-Formatierungs-Plugin zu Veränderungen darin käme.

Über das Auswahlfeld können Sie bestimmen, welche Textformatierungen auf den aktuellen Artikel *nicht* angewendet werden sollen. Alle nicht ausgewählten Plugins werden weiterhin ausgeführt.

Freie Felder

Abschließend folgt eine Liste aller in der Plugin-Konfiguration eingerichteten *Freien Felder*. Für jedes festgelegte Feld können Sie hier einen beliebigen Inhalt hinterlegen: HTML-Text, kurze Sätze – die Einsatzzwecke sind nur durch Ihre Fantasie begrenzt. Auch ist es möglich, einen Verweis auf eine Datei in der Mediendatenbank einzutragen – so könnten Sie beispielsweise einfach eine MP3-Datei mit einem Artikel verketten. Um eine Mediendatei leicht einzufügen, befindet sich hinter jedem freien Feld direkt ein Link zum Aufruf der Mediendatenbank. In diesem Popup können Sie wie beim Artikeltext gewohnt eine Datei zum Einfügen auswählen.

6.2.8 Spartacus serendipity_event_spartacus

Im Kapitel 22 auf Seite 151 wird das Plugin-System *Spartacus* beschrieben. Dies ist ein online verfügbares Archiv, in dem zahlreiche Plugins und Templates für Serendipity angeboten werden.

Damit Sie von Ihrem Blog aus leicht solche Plugins und Templates installieren können, müssen Sie das Ereignis-Plugin *Spartacus* installieren.

Das Plugin bindet sich in Serendipity an allen Stellen ein, wo Plugins (und Templates) zur Installation angeboten werden. Sobald das Plugin installiert ist, werden zusätzliche Funktionen aktiviert und die Inhalte des Plugin-Archivs eingebunden.

Da beim Betrieb von Spartacus einige Voraussetzungen erfüllt werden müssen (siehe erwähntes Kapitel), ist das Plugin standardmäßig nicht installiert.

Technisch funktioniert das Plugin so, dass es eine Paketdatei im XML-Format von einem festgelegten Internet-Server (*Mirror*) abruft. Diese Paketdatei (`package_event_de.xml` für Ereignis-Plugins mit deutscher Beschreibung, `package_sidebar_de.xml` für Seitenleisten-Plugins) wird auf Ihren eigenen Server heruntergeladen und im Verzeichnis `templates_c` gespeichert. Bei der Installation von Plugins wertet Spartacus diese XML-Daten aus und stellt sie für Sie dar. Sobald Sie nun ein Plugin installieren wollen, wird Spartacus vom konfigurierten Internet-Server die Dateien einzeln herunterladen und im `plugins`-Verzeichnis abspeichern.

Damit Spartacus die XML-Datei nicht jedesmal aufs Neue herunterladen und auslesen muss, wird die XML-Datei für einen bestimmten Zeitraum zwischengespeichert, und alle Informationen daraus werden in einer Datenbank hinterlegt.

Folgende Konfigurationsoptionen bietet Spartacus:

Enable the use of Spartacus for fetching plugins
Damit Spartacus sich in die Plugin-Verwaltung einbindet, muss diese Option aktiviert sein. Wenn Sie Spartacus also beispielsweise nur für die Verwaltung von Templates aktivieren möchten, können Sie dies gezielt einstellen.

Enable the use of Spartacus for fetching themes
Ähnlich wie für die Einbindung in die Plugin-Verwaltung müssen Sie Spartacus auch für das Herunterladen von Templates aktivieren, wenn Sie dies wünschen.

Enable remote plugin version information, Secret key to Remote plugin

version information Spartacus verfügt über eine Art *Fernwartungszugriff*. Unter einer speziellen URL können Sie eine Liste aller installierten Plugins einsehen und prüfen, ob für diese Plugins neue Versionen vorliegen. Diese Datei ist in einem sehr einfachen Format abgelegt und kann so leicht von Ihnen geparst oder regelmäßig z. B. via cronjob ausgelesen und per E-Mail versendet werden.

Diese Option ist nur für erfahrene Administratoren vorgesehen. Da die Ausgabe etwaigen böswilligen Besuchern des Blogs detaillierte Informationen über Ihr Blog geben kann, ist die Option standardmäßig deaktiviert.

Wenn Sie die Fernwartung aktivieren, sollten Sie den Namen der URL unbedingt ändern, so dass fremde Besucher den Namen zum Aufruf nicht raten können.

Datei/Mirror Speicherort (XML-Metadaten)
In diesem Auswahlfeld legen Sie fest, von welchem Internet-Server das Plugin Paketinformationen beziehen kann. Nur *Netmirror.org* ist derzeit in Betrieb. Für die Zukunft ist der Eintrag *s9y.org* vorgesehen, der jedoch noch brach liegt.

Datei/Mirror Speicherort (Downloads)
Die eigentlichen Plugin- und Template-Dateien bezieht das Plugin von dem hier ausgewählten Server. Hier haben Sie neben *Netmirror.org* auch noch die Alternative, den *SourceForge.net*-Server zu wählen. Der SourceForge-Server hat in der Vergangenheit jedoch öfter Ausfälle gehabt, daher sollten Sie möglichst *Netmirror.org* voreingestellt lassen.

Die beiden Server *s9y.org* und *berliOS.de* sind ebenfalls für zukünftige Erweiterungen vorgesehen und lassen sich derzeit nicht benutzen.

Eigentümer der heruntergeladenen Dateien, Zugriffsrechte der

heruntergeladenen Dateien/Verzeichnisse Spartacus läuft auf Ihrem Webserver als ein normaler PHP-Prozess. Alle Daten, die das Plugin auf die Festplatte Ihres Servers schreibt, gehören somit standardmäßig dem PHP-Benutzer, meist *nobody* oder *wwwrun*. Weitere Informationen zur Einrichtung von Benutzerrechten lesen Sie in Kapitel 2.2.2 ab Seite 53.

Je nach Konfiguration des Providers könnte es passieren, dass aufgrund dieser Eigentumsrechte eine von Spartacus heruntergeladene Datei für Sie mit FTP-Zugriffsrechten nicht mehr zu bearbeiten ist. Oft ist es jedoch gewünscht, per FTP die Dateien eines Plugins anzupassen/zu bearbeiten.

Daher bietet das Spartacus-Plugin die Möglichkeit, dass Sie hier den Namen Ihres FTP-Benutzers eintragen. Spartacus versucht dann, eine heruntergeladene Datei diesem Benutzer zu übertragen. Nicht alle Provider unterstützen diese `chown`-Kommandos!

Abgesehen von den Informationen über den Eigentümer eines Plugins können Sie auch gezielt Zugriffsrechte einer Datei und eines Verzeichnisses für andere Benutzer festlegen. Wenn Sie also auf Ihrem System die Dateien von Spartacus später nicht mehr verändern dürfen, können Sie über die hier festgelegten Zugriffsrechte Anpassungen vornehmen. Zugriffsrechte wie `0777` würden dafür sorgen, dass jeder Benutzer auf dem Server die Dateien lesen und schreiben darf.

Wenn Sie die Zugriffsrechte oder Eigentümer an dieser Stelle ändern, gilt dies nur für Dateien, die Spartacus in Zukunft herunterladen wird. Bereits auf dem Server befind-

liche Dateien werden nachträglich nicht verändert. Diese müssten Sie manuell z. B. mittels des *fixperm*-Scripts (siehe Seite 64) beheben.

FTP server address, username, password, serendipity directory
Seit Serendipity 1.3 bietet Spartacus die Möglichkeit, Dateien nicht nur intern direkt per PHP-Befehl auf dem Server zu speichern, sondern auch per FTP.

Dieser FTP-Upload umgeht das Problem, dass auf einigen Servern der PHP *SafeMode* aktiviert ist. Der *SafeMode* sorgt dafür, dass Dateien in Ihrem Stammverzeichnis des Webservers nur durch den FTP-Besitzer verändert werden dürfen. PHP selbst besitzt in diesem Fall häufig keinen Schreibzugriff.

Daher kann das Spartacus-Plugin, das über den PHP-Benutzer ausgeführt wird, auf derartigen Servern nicht korrekt ausgeführt werden, da es die heruntergeladenen Plugins nicht speichern kann. Durch Verwendung des FTP-Zugangs, den Sie auch zum Hochladen Ihrer Dateien verwenden, kann das Plugin die Dateien jedoch über diesen Umweg speichern.

In den Konfigurationsfeldern müssen Sie die Zugangsdaten Ihres FTP-Benutzers eintragen. Das **Serendipity directory** entspricht dabei dem relativen Verzeichnis von Serendipity. Bei einer FTP-Verbindung werden Sie standardmäßig in das Stammverzeichnis Ihres Webauftritts geleitet, zum Beispiel `/var/www/example.com/`. Wenn Serendipity im Unterverzeichnis `blog` installiert wird, müssen Sie diesen Pfad auch in der Konfiguration eintragen. Andernfalls würde das Spartacus-Plugin die heruntergeladenen Dateien in einem falschen Unterverzeichnis speichern.

6.2.9 Artikel mailen serendipity_event_mailer

Das Plugin *Artikel mailen* ermöglicht es, einen neuen Artikel nach der Veröffentlichung via E-Mail an einen (oder mehrere) Empfänger zu verschicken.

Grundsätzlich stellen die RSS-Feeds (siehe Seite 38) eine wesentlich einfachere Möglichkeit dar, Besucher über neue Einträge zu informieren. Im Gegensatz zu E-Mails können RSS-Feeds auf Wunsch und Initiative des Besuchers empfangen werden, daher spricht man hier von einem *Pull*-Dienst. Newsletter und E-Mails zählen zu den *Push*-Diensten, das bedeutet, dass Inhalte zu beliebigen Zeitpunkten an den Benutzer geliefert werden, und nicht etwa dann, wenn der Besucher dies auslöst.

Der Vorteil von RSS-Feeds liegt daher darin, dass ein Besucher sich aktiv über Ihren Blog informieren möchte, während eine eingehende Benachrichtigungs-E-Mail bei ihm möglicherweise gerade unpassend ankommt.

Andererseits kann man natürlich auch argumentieren, dass ein E-Mail-Hinweis komfortabel ist, um Besucher an Ihre Webseite zu erinnern, so dass sich der Benutzer „berieseln“ lassen kann, anstatt selber Initiative zu zeigen.

Wie immer man also zu RSS-Feeds vs. E-Mails steht, das Plugin ermöglicht genau diesen Mail-Versand. Standardmäßig wird eine E-Mail jedoch nur an eine zentrale Empfängeradresse versendet. Wenn Sie stattdessen mehrere Benutzer informieren wollen, können Sie mehrere Empfängeradressen, natürlich mit Kommas getrennt, auflisten. Sinnvoller wäre es in diesem Fall aber, potenziell interessierte Besucher in einer Liste zusammenzufassen. Eine neue Benachrichtigungsmail geht dann an eine zentrale Liste, und diese Liste wiederum leitet die E-Mail an alle eingetragenen Empfänger weiter. Dieses Konzept nennt man *Mailingliste* oder auch *Newsletter*.

Der Versand von Massen-E-Mails mittels einer PHP-Anwendung (wie Serendipity) ist aus Ressourcengründen nicht zu empfehlen. Daher bietet dieses Plugin keine konfigurierbare Mailingliste an, sondern Sie müssen sich selbst um die Einrichtung einer solchen kümmern. Das ist jedoch kein Problem, da zahlreiche kostenlose Anbieter am Markt so etwas komfortabel umsetzen: Google Groups (`http://groups.google.de`) und Yahoo Groups (`http://de.groups.yahoo.com`) sind die bekanntesten Vertreter. Dort können Sie schnell eine eigene Mailingliste erstellen. Die Dienstleister bieten auch kleine Code-Schnipsel an, die Sie in Ihrem Blog einbinden können, damit sich Besucher leicht in die Mailingliste eintragen können. Sie tragen dann lediglich die E-Mail-Adresse der Mailingliste in der Konfiguration des Plugins ein, und Yahoo bzw. Google erledigt den Rest für Sie. Natürlich können Sie auch eigene Mailinglisten-Software wie ezmlm (`http://www.ezmlm.org/`) oder Mailman (`http://www.gnu.org/software/mailman/mailman.html`) einsetzen, wenn Sie dies auf Ihrem Server nutzen können.

Bei der Erstellung eines Blog-Artikels sehen Sie im Bereich **Erweiterte Optionen** eine Oberfläche, in der standardmäßig die E-Mail-Adressen aufgeführt sind, an die der Artikel verschickt wird. Zudem können Sie die Auswahlbox **An alle Redakteure schicken** markieren, damit der Blog-Artikel an alle Redakteure des Blogs geht. Sobald Sie einen Artikel veröffentlichen, gibt das Plugin aus, an welche E-Mail-Adressen eine E-Mail geschickt wurde.

Die weiteren Konfigurationsoptionen des Plugins sind:

Inhalt
: Mit diesem Auswahlfeld können Sie festlegen, ob die E-Mails den vollständigen Arti-

keltext (bestehend aus **Eintrag** und **Erweitertem Eintrag**) enthalten sollen oder eine beliebige Kombination aus beidem. Wenn Sie bei einem ausführlichen Artikel also den erweiterten Eintrag weglassen wollen, geben Sie so Ihren Besuchern einen Anreiz, trotzdem Ihr Blog zu besuchen.

Mail-Empfänger
In dieses Feld tragen Sie den eingangs erwähnten E-Mail-Empfänger ein, oder die Adresse einer Mailingliste.

An alle Redakteure schicken
Legt die Standardeinstellung für die Auswahlbox **An alle Redakteure schicken** bei der Erstellung eines Blog-Artikels fest.

URL des Artikels mailen
Wenn Sie diese Option aktivieren, wird die URL Ihres Artikels mit in die E-Mail aufgenommen. Dies ist wichtig, damit die Leser einer E-Mail auch leicht zu Ihrem Blog gelangen können.

HTML entfernen
Artikeltexte können HTML-Formatierungen enthalten, die in einer E-Mail nicht sinnvoll dargestellt werden. Wenn Sie die Option **HTML entfernen** aktivieren, stellt das Plugin sicher, dass sämtliche HTML-Tags (mittels PHP `strip_tags()`-Befehl) aus der E-Mail entfernt werden.

Mögliche `<a>`- und `<img>`-Tags für Bilder und Hyperlinks werden dabei so entfernt, dass die referenzierte URL trotzdem im Text bestehen bleibt.

HTML-Paragraphen in Leerzeilen wandeln
Wenn Sie die Option **HTML entfernen** aktiviert haben, kann dies dazu führen, dass die Zeilenumbrüche verloren gehen, die bei der HTML-Ansicht eines Artikels für die Gliederung sorgen. Daher können Sie die Option **HTML-Paragraphen in Leerzeilen wandeln** aktivieren, damit nach einem HTML-Absatz (`</p>`) ein gewöhnlicher E-Mail-Zeilenumbruch eingefügt wird.

Kategorien
Für jede im Blog erstellte Kategorie sehen Sie ein zusätzliches Eingabefeld. Dort können Sie eine kategorieabhängige E-Mail-Adresse eintragen, damit ein Artikel dieser Kategorie nur an die E-Mail-Adressen geschickt wird, die für die Kategorie hinterlegt wurden.

E-Mails zu Artikeln, die keiner Kategorie zugeordnet sind, werden dann an die Standardadresse verschickt.

6.2.10 LiveSearch serendipity_event_livesearch

Das *LiveSearch*-Plugin ist eine Erweiterung des Seitenleisten-Plugins für die Suche (siehe Seite 202).

Sobald es installiert ist, wird ein kleines JavaScript in das Suchformular eingebunden. Wenn Sie als Besucher einen Begriff in das Suchformular eintragen, wird die LiveSuche alle zutreffenden Artikel in einer Aufklappbox unter dem Suchformular anzeigen, und Sie können einfach darauf klicken.

Diese Art der Suche erspart Ihnen bei aktiviertem JavaScript im Browser also, erst eine lange Artikelübersicht durchforsten zu müssen, bevor Sie den gewünschten Artikel finden.

6.2.11 Hebe Suchwörter hervor serendipity_event_searchhighlight

Wenn ein Besucher Ihr Blog mittels einer Suchmaschine gefunden hat und aufruft, wird von den Suchmaschinen der Suchbegriff übermittelt. Diesen Suchbegriff kann das Plugin ermitteln und in Ihren Beiträgen hervorheben. So kann ein Besucher dann komfortabel sehen, wo der ursprünglich benutzte Suchbegriff in Ihrem Artikel auftaucht.

Das Plugin kann Suchwörter folgender Suchmaschinen ermitteln und hervorheben: Google, Yahoo, Lycos, MSN, Altavista, AOL.de und AOL.com. Die Art der Hervorhebung können Sie mittels der CSS-Klasse `.serendipity_searchQuery` beeinflussen (siehe Kapitel 9.1 ab Seite 470).

Das Plugin setzt die Smarty-Sondervariable {`$smarty.SESSION.search_referer`} auf die URL der Suchmaschine, von welcher der aktuelle Besucher kam. Wenn der Besucher über eine Suchmaschine auf das Blog gelangte, ist weiterhin die Variable {`$smarty.SESSION.is_searchengine_visitor`} auf `true` gesetzt. So können Sie innerhalb Ihrer Template-Dateien individuelle Ausgaben darstellen, falls der Besucher über eine Suchmaschine auf Ihre Seite gelangte. Für solche Besucher sind z. B. einleitende Worte zu Ihrem Blog und weiteren interessanten Artikeln sehr hilfreich.

6.2.12 Karma serendipity_event_karma

Der Begriff *Karma* beschreibt ein spirituelles Konzept, wonach jede gute oder schlechte Tat sich in guten oder schlechten Erfahrungen niederschlagen wird. Gerade im Hinduismus oder bei Religionsgemeinschaften, die an die Wiedergeburt glauben, ist diese Vorstellung wichtig für die Handlungsweisen der Menschen.

Das Karma-Plugin wendet dieses Konzept auf Blog-Beiträge an: Ihre Besucher können zu jedem Beitrag abstimmen, wie „gut“ oder „schlecht“ sie diesen finden. Das kann Ihnen dann

Aufschluss darüber geben, wie gezielt Sie auf Ihre Besucher eingehen.

Das Plugin ermöglicht eine Bewertung in fünf Stufen: -- (sehr schlecht), - (schlecht), 0 (neutral), + (gut) und ++ (sehr gut). Die Abstimmungen können nur bei aktiviertem JavaScript durchgeführt werden, damit Suchmaschinen nicht auch abstimmen können.

Seit der neuen Version des Plugins mit Serendipity 1.3 können auch beliebige Grafiken für die Abstimmung eingebunden werden.

Zusätzlich bietet das Karma-Plugin auch „statistische Fähigkeiten". Es kann zählen, wie viele Besucher die Detailseiten eines Artikels angesehen haben. Dafür legt das Plugin eine eigenständige Datenbanktabelle an, in der die Abstimmungen und Statistiken gespeichert werden. Diese Statistiken können direkt in der Artikelansicht dargestellt werden und werden auch bei den Ausgaben des *Statistik*-Plugins (siehe Seite 387) berücksichtigt.

Die grafische Gestaltung der Abstimmungslinks kann via CSS und den entsprechenden Plugin-CSS-Klassen vorgenommen werden (siehe auch Seite 470).

Das Plugin bietet die folgenden Konfigurationsoptionen, aufgeteilt in drei Bereiche namens **Globales**, **Darstellung** und **Texte**:

Karmavoting aktivieren
: Wenn Sie den Besuchern die Abstimmungsmöglichkeit zu einem Artikel anbieten wollen, müssen Sie diese Option aktivieren. Sie können sie deaktivieren, wenn Sie beispielsweise nur die Aufrufstatistik erhalten möchten.

Nur erweiterte Artikel
: Die Abstimmungsmöglichkeit kann vom Plugin sowohl in der Artikelübersicht als auch in der Detailansicht eines Artikels angezeigt werden. Wenn Sie die Option **Nur erweiterte Artikel** aktivieren, kann ein Besucher nur in der Detailansicht eines Artikels abstimmen. Dies kann möglicherweise hilfreich sein, um Ihre Artikelübersichten einfacher zu strukturieren.

Maximaler Abstimmungszeitraum
: In diesem Eingabefeld legen sie fest, nach welchem Zeitraum seit der Veröffentlichung eines Artikels keine Abstimmung mehr zugelassen wird. Häufig interessiert Sie nur die Meinung Ihrer Besucher zu aktuellen Beiträgen, daher ist standardmäßig die Abstimmung über Artikel nur bis zu 7 Tage nach deren Veröffentlichung möglich.

Abstimmungszeitraum
: Mit der Option **Abstimmungszeitraum** legen Sie fest, wie lange die „Zwangspause" eines Besuchers dauert, die er nach einer Bewertung eines Beitrags abwarten muss, bis er über einen weiteren Artikel abstimmen darf. Standardmäßig sind dies 5 Minuten.

  Der Abstimmungszeitraum gilt nur für Artikel, die älter als das in der Option **Abstimmungszeitraum nach Veröffentlichung eines Artikels** festgelegte Alter sind.

Abstimmungszeitraum nach Veröffentlichung eines Artikels
: Wenn Sie das Karmavoting aktiviert haben, kann ein Besucher einen Artikel bewerten.

Nach jeder Bewertung erfolgt eine „Zwangspause“ für den Besucher, bis er einen neuen Artikel bewerten kann.

Diese Zwangspause gilt jedoch nicht für gerade erst veröffentlichte Artikel. Stellen Sie sich vor, Sie veröffentlichen drei neue Artikel, und ein Besucher müsste erst mehrere Minuten warten, bevor er alle drei neuen Artikel nacheinander bewerten kann. Dies wäre keine besonders sinnvolle Einschränkung. Dennoch macht eine Zwangspause später Sinn, damit ein böswilliger Besucher nicht einfach alle Artikel Ihres Blogs nacheinander schlecht bewertet.

In dem Eingabefeld **Abstimmungszeitraum nach Veröffentlichung eines Artikels** tragen Sie eine Zeitdauer in Minuten ein, die das Alter eines Eintrages festlegt, bei dem ein Benutzer ohne Zwangspause abstimmen darf. Standardmäßig sind dies 1440 Minuten, also ein Tag. Alle Artikel, die jünger als ein Tag sind, können dann ohne Pause von jedem Besucher nacheinander bewertet werden.

Minimale Anzahl an Stimmen für Darstellung
Wenn Sie die Ergebnisse der Abstimmung erst ab einer gewissen Anzahl an Stimmen anzeigen wollen, können Sie dies hier festlegen.

Aufrufstatistik aktivieren
Wenn Sie möchten, dass das Plugin jeden Aufruf eines einzelnen Artikels zählt, können Sie diese Option aktivieren. Wenn ein Artikel in einer Übersicht dargestellt wird, zählt dies nicht als Aufruf.

Aufrufstatistik auch für eingeloggte Benutzer
Falls diese Option deaktiviert wird, werden die Besuche eingeloggter Redakteure in der Aufrufstatistik nicht mitgezählt.

Minimale Besucheranzahl
Wenn Sie die Ergebnisse der Besucherzählung erst ab einer gewissen Anzahl an Besuchern anzeigen wollen, können Sie dies hier festlegen.

Zeigt die Top-Exit-Links des Blogs
Wenn Sie das Plugin *Externe Links zählen* installiert haben, kann jeder Klick auf eine in Ihren Artikeln genannte Webseite gezählt werden. Da diese Seitenaufrufe in der Datenbank gespeichert werden, können sie nicht nur mittels des Plugins *Top Exits* in der Seitenleiste eingebunden werden, sondern das Karma-Plugin kann auch auf diese Daten zugreifen. Wenn Sie die Option *Zeigt die Top-Exit-Links des Blogs* aktivieren, wird die Summe der Aufrufe aller in einem Beitrag genannten Links am Ende des Artikels dargestellt. Sinnvoll ist diese Möglichkeit vor allem bei Blogs, die pro Artikel immer nur einen oder zumindest ähnliche Links aufführen, da durch die Summenbildung die Individualität mehrerer Links verloren geht.

Protokollieren
Jede Abstimmung über einen Beitrag kann in der Datenbanktabelle `serendipity_karmalog` protokolliert werden, sofern diese Option aktiviert ist. Die Datenbanktabelle können Sie dann manuell mittels phpMyAdmin ansehen (und von Zeit zu Zeit löschen).

Konfigurationsbereich Darstellung
: In diesem Bereich können Sie über mehrere Konfigurationsoptionen bestimmen, ob der Abstimmungsbereich durch Text- oder Grafikdarstellung eingebunden werden soll und welchem grafischen Stil dieser entsprechen soll.

Konfigurationsbereich Texte
: Die einzelnen Texte für die Umfrage können von Ihnen frei eingetragen werden.

Datenbanktabellen

Die Datenbanktabelle `serendipity_karma` enthält die Abstimmungen zu jedem Blog-Artikel. Zu jedem Blog-Artikel gibt es nur einen Eintrag, d. h. die Werte der Abstimmung werden jeweils zu diesem Eintrag hinzuaddiert.

`entryid`
: enthält die ID des Blog-Artikels.

`points`
: enthält die aktuelle Punktzahl des Artikels.

`votes`
: enthält die Anzahl der abstimmenden Besucher.

`lastvote`
: enthält das Datum der letzten Abstimmung.

`visits`
: enthält die Anzahl an Besuchen des Artikels.

Damit alle Abstimmungen einzeln nachverfolgt werden können, wird bei aktivierter Logging-Option des Plugins die Datenbanktabelle `serendipity_karmalog` befüllt:

`entryid`
: enthält die ID des Blog-Artikels.

`points`
: enthält die abgegebene Punktzahl.

`ip`
: enthält die IP-Adresse des abstimmenden Besuchers.

`user_agent`
: enthält den verwendeten Browser des Besuchers.

`votetime`
: enthält das Datum, an dem diese Punktzahl abgegeben wurde.

6.2.13 Einträge ankündigen serendipity_event_weblogping

Nach der Veröffentlichung eines Artikels möchten Sie natürlich auch, dass Ihre Besucher diesen lesen. Damit Internet-Surfer auf neue Blog-Einträge aufmerksam werden (und auch Suchmaschinen neue Beiträge direkt aufnehmen), kann das Plugin *Einträge ankündigen* Verbindung zu Webservices aufnehmen.

Zur Benachrichtigung für neue Einträge existiert eine Standard-API, die sich „Ping“ nennt. Die API (`weblogUpdates.ping` und `weblogUpdates.extendedPing`) wird mittels XML-RPC-Schnittstelle aufgerufen. Damit Serendipity einen Artikel bei einem Webservice ankündigen kann, muss der Webservice diese Schnittstelle auch bereitstellen. Weiterhin muss Ihr Webserver in der Lage sein, ausgehende HTTP-Verbindungen zu anderen Internet-Servern herzustellen. Eine Firewall muss also entsprechend konfiguriert werden.

Sobald das Plugin installiert ist, sehen Sie in dem Abschnitt *Erweiterte Optionen* beim Erstellen eines Artikels ein Feld, in dem Sie auswählen können, zu welchen Webservices ein Ping gesendet werden soll. Jeden gewünschten Webservice können Sie dort ankreuzen, die Voreinstellungen können Sie über die Konfiguration des Plugins beeinflussen.

Abhängig von der Sprache des Blogs werden unterschiedliche Webservices angeboten. Für deutsche Blogs sind dies: Ping-o-matic, blo.gs, blogrolling.com, technorati.com, weblogs.com, ge.bloggt.org, Yahoo! und Google.

Jeder Dienst, den Sie über einen neuen Artikel informieren, wird beim Veröffentlichen des Artikels etwas Zeit in Anspruch nehmen. Wenn das Speichern eines Artikels sehr lange dauert oder zu Fehlern führt, kann dies also am Plugin *Einträge ankündigen* liegen. In diesem Fall sollten Sie die Anzahl der gepingten Services reduzieren oder das Plugin vollständig deaktivieren. Einige Webservices dienen als „Weiterleitung“ von Pings, z. B. der Dienst Ping-o-matic. Wenn Sie diesen benutzen, können Sie sich das Pingen anderer Dienstleister sparen, allerdings ist Ping-o-matic nicht immer vollständig funktionstüchtig[11]. Das Plugin wird automatisch die Webservices in der Liste abwählen, die bereits von einem solchen Meta-Service verwaltet werden.

In der Konfiguration des Plugins können Sie zusätzlich eigene Webservices eintragen, zu denen Sie Pings setzen wollen. Tragen Sie diese mit einem Komma voneinander getrennt in **Selbstdefinierte Ping-Services** ein. Da jeder Webservice unter einer speziellen URL aufrufbar ist, müssen Sie diese im Format `host.domain/pfad` (also z. B. `rpc.blogrolling.com/pinger/` für den Webservice von BlogRolling.com) im Eingabefeld eintragen. Die XML-RPC-Spezifikation ermöglicht entweder einen *einfachen* oder einen *erweiterten* Ping. Beim erweiterten Ping wird die URL Ihres RSS-Feeds an den Webservice übermittelt, der anhand des Feeds weitere Daten Ihres Blogs einlesen kann. Wenn ein selbst definierter Webservice diese Option unterstützt, können Sie ein `*` vor den Hostnamen setzen.

[11] Siehe `http://pingomatic.com/`

6.3 Auswahl externer Plugins

Über die mitgelieferten Plugins hinaus finden Sie auch eine große Auswahl an Plugins über `http://spartacus.s9y.org/`. Eine Auswahl an häufig genutzten Ereignis-Plugins finden Sie auf den folgenden Seiten.

6.3.1 Einträge automatisch sichern serendipity_event_autosave

Bei der Benutzung einer Web-Anwendung in Ihrem Browser kann es öfter als bei normalen Desktop-Anwendungen passieren, dass Ihre mühsame Arbeit verloren geht. Der Browser stürzt ab, möglicherweise weil Sie in einem anderen Fenster gerade eine fehlerhafte Webseite besucht haben; oder Sie haben aus Versehen die Taste oder den Button gedrückt, mit dem der Browser zurück zur vorherigen Seite wandert. In beiden Fällen wäre der mühsam geschriebene Blog-Artikel vermutlich verloren.

Das häufigste Problem bei Web-Anwendungen, in denen man lange Texte schreibt, ist, dass man fleißig eine Stunde oder länger damit verbringt, den Text perfekt zu formulieren. Während dieser Zeitspanne sendet Ihr Browser keine Daten zum Server, daher ist die Zeit, die Sie mit einem Artikel verbringen, für eine Web-Anwendung so, als würden Sie überhaupt nichts tun.

Wenn Sie nach einer Stunde des „Nichtstuns“ also plötzlich einen Artikel veröffentlichen wollen, muss Serendipity Sie erst einmal wiedererkennen. Dies erfolgt üblicherweise anhand der Session-Cookies (siehe Seite 49).

Session-Cookies sind jedoch nur für einen fest eingestellten Zeitraum gültig. Ist die Haltbarkeit eines Cookies überschritten, löscht PHP sämtliche Session-Daten. Für Serendipity bedeutet dies, dass Ihr Login nicht mehr erkannt werden kann und Sie sich daher neu einloggen müssen. Dies gilt natürlich nur, wenn Sie beim Login die Auswahlbox **Daten merken?** *nicht* aktiviert haben. Ist die Box aktiviert, kann Serendipity Sie auch nach dem Überschreiten der Session-Haltbarkeit automatisch neu einloggen.

Die Haltbarkeit einer PHP-Session wird in der PHP-Konfigurationsvariable `session.gc_maxlifetime` in Sekunden festgelegt. Per Default sind dies 1440 Sekunden, also 24 Minuten. Zwar kann man diesen Wert verändern, aber da es einige Besonderheiten zu beachten gilt (siehe `http://de2.php.net/manual/de/ref.session.php`), sollte man dies nicht unbedingt tun. Zumal die Standardeinstellung einen guten Mittelwert zwischen normaler Inaktivität und Sicherheitserwägungen darstellt. Je länger eine Session gilt, desto eher kann ein böswilliger Benutzer diese Daten möglicherweise ausspähen und sich unberechtigten Zugang verschaffen.

Daher sollten Sie bei der Benutzung von Serendipity eher entweder die Login-Option **Daten merken?** aktivieren oder bei einem ungewollten Logout folgenden Trick anwenden: Wenn Sie eine Aktion in Serendipity ausführen, sendet Ihr Browser die Daten (z. B. den Artikeltext) an den Server. Meldet Serendipity daraufhin, dass Sie sich neu einloggen müssen, dann sind diese an den Server gesendeten Daten noch im Zwischenspeicher des Browsers. Der Trick ist nun, dass Sie einfach ein weiteres Browser-Fenster öffnen und sich dort in die Serendipity-

Administrationsoberfläche einloggen. Nun haben Sie ein neues, gültiges Session-Cookie, mit dem Sie Serendipity wie gewohnt bedienen können. Gehen Sie nun zurück in das Browser-Fenster, bei dem das Speichern des Artikels fehlschlug, und laden die Seite erneut. Die meisten Browser tun dies, wenn Sie die Taste *(F5)* drücken oder auf den **Neu laden**-Menübutton klicken. Ihr Browser sollte Sie nun fragen, ob die Formulardaten (die Daten Ihres Artikels) erneut geschickt werden sollen. Bestätigen Sie dies, und Ihr ursprünglich übermittelter Artikel wird nun mit einem gültigen Session-Cookie korrekt gespeichert.

Diesen Trick können Sie nur nutzen, wenn Sie sich im Browser-Fenster noch nicht neu eingeloggt und das Ursprungsfenster noch nicht geschlossen haben.

Ein grundsätzlicher Tipp ist jedoch, dass Sie besonders lange Artikel im *Entwurfsmodus* speichern. Ab und zu klicken Sie dann einfach auf den Button **Speichern**, um einen Zwischenstand des Artikels in der Datenbank zu sichern. Wenn Ihr Browser abstürzt, können Sie so den letzten Stand direkt über Serendipity wieder einsehen. Ein anderer Tipp wäre, dass Sie besonders lange Artikel in einem Schreibprogramm wie OpenOffice vorschreiben und erst später per Kopieren & Einfügen in das Blog übernehmen. Wenn Sie das XML-RPC Plugin (siehe Seite 351) benutzen, können Sie zudem Schreibprogramme verwenden, die solche vorgeschriebenen Artikel automatisch zwischenspeichern und später an den Server senden.

Neben den aufgeführten Möglichkeiten gibt es auch ein Plugin namens *Einträge automatisch sichern* (*Autosave*), das Ihnen die Arbeit des regelmäßigen Speicherns teilweise abnehmen kann.

Dabei verwendet das Plugin ein spezielles JavaScript (AJAX), das in einem regelmäßigen Intervall den Artikel als Entwurf speichert. Wurde ein Artikel derart gespeichert, zeigt das Plugin einen Hinweis darauf in der Artikel-Erstellungsmaske an. Wenn Ihr Browser abstürzt, können Sie einen so gespeicherten Artikel als *Entwurf* in Ihrer Artikelübersicht fortsetzen.

Wenn Sie einen bereits veröffentlichten Artikel überarbeiten, wird eine Sicherungskopie davon angelegt. Somit werden Änderungen, die Sie ausführen, nicht automatisch am veröffentlichten Artikel dargestellt. Derartige Sicherungskopien, die nicht den ursprünglichen Artikel betreffen, nennt man *Schattenkopien*, da sie vor dem Benutzer versteckt werden. Solche Schattenkopien markiert das Plugin mit dem vorangestellten `[AUTOSAVED]` im Titel des Artikels. Die Schattenkopie eines Artikels finden Sie wie einen gewöhnlichen Artikel in der Übersicht, und sie wird beim Speichern des Originalbeitrags automatisch wieder gelöscht.

Wenn Sie nach einem Browser-Absturz eine Schattenkopie wiederherstellen wollen, bearbeiten Sie den Originalartikel (*nicht* die Schattenkopie) und klicken oberhalb der Artikelerfassung auf den Button **Der Eintrag hat Sicherungsdaten, zur Wiederherstellung hier klicken**. Dadurch wird die Schattenkopie in den Ursprungsartikel eingefügt.

Das Plugin besitzt zwei Konfigurationsoptionen. Zum einen können Sie einstellen, nach wie vielen Sekunden ein Artikel automatisch gespeichert wird. Stellen Sie dieses Intervall nicht zu gering ein, da ansonsten viel Redundanz und unnötige Serverlast entsteht.

Die zweite Option stellt den HTTP-Pfad zu dem Plugin ein. Da Serendipity Plugins aus beliebig verschachtelten Unterverzeichnissen laden kann, weicht Ihre Einstellung möglicherweise von dem Standard-Plugin-Pfad ab. Wenn das Plugin im Unterverzeichnis `plugins/spartacus_plugins/serendipity_event_autosave` gespeichert wäre, würde der einzutragende Pfad `http://www.example.com/plugins/serendipity_event_autosave` lauten.

6.3.2 Frei definierbare Permalinks zu Einträgen serendipity_event_custom_permalinks

Ein Blog-Artikel ist standardmäßig unter der URL `http://www.example.com/serendipity/archives` verfügbar. Wie genau die Pfadkomponenten benannt sind und welche zusätzlichen Komponenten (Datum des Artikels, mit oder ohne Artikeltitel oder ID etc.) in einem solchen Detail-Link (*Permalink*) erscheinen, können Sie in der Serendipity-Konfiguration im Bereich *Permalinks*[12] einstellen.

Für besonders interessante Artikel können Sie aber auch individuelle URLs für einen Beitrag vergeben, die von dieser Konvention abweichen. Diese URL kann dann zusätzlich zur „echten“ URL benutzt werden, um zu dem gewünschten Artikel zu führen. Diese Extra-URL bietet das Plugin *Frei definierbare Permalinks zu Einträgen* bei der Erstellung eines Artikels in einem Feld der *Erweiterten Optionen* an. Andere Blogsysteme nennen einen derartigen Permalink auch *slug*.

Dort tragen Sie den vollständig gültigen Pfad zu einem Artikel ein. Dieser Link wird später auch im Frontend anstelle der vordefinierten Struktur angezeigt. Ein Standardpfad wird in dieser Box vorausgefüllt dargestellt (z. B. `/serendipity/permalink/Mein-Blogartikel.html`). Wenn Sie den Pfad anpassen, dürfen Sie nur die Teile anpassen, die hinter der Stamm-URL von Serendipity (in unserem Beispiel `/serendipity/`) aufgeführt sind. Sie können beliebige gültige URL-Zeichen[13] als Permalink einsetzen. Die einzige Regel dabei ist, dass der Permalink auf `.htm` oder `.html` endet, damit Serendipity ihn korrekt erkennen kann. Auch dürfen Sie bereits bestehende interne Permalinks (wie `/serendipity/archives/1-Mein-Blogartikel.html`) nicht verwenden, da Serendipity dann eine URL nicht eindeutig einem Inhalt zuweisen könnte. Die verwendeten Verzeichnisnamen werden nur virtuell belegt, Sie müssen also keine entsprechende Verzeichnisstruktur auf Ihrem Server anlegen.

Das Plugin wird gerne dann verwendet, wenn man besonders lange Artikeltitel benutzt, die man in der URL aber lieber auf Stichworte eingrenzen möchte. Oder im umgekehrten Fall, wenn man aus Gründen der Suchmaschinenoptimierung einer URL mehr Stichworte mitgeben möchte, als der Artikeltitel enthält.

---

[12] Siehe Kapitel 23 ab Seite 161.

[13] Sonderzeichen und Leerzeichen müssen entsprechend der RFC 1738 (`http://www.faqs.org/rfcs/rfc1738.html`) konvertiert werden. Das URL-Sonderzeichen `;` würde daher als `%3B` umgeschrieben werden. Da dies eher unschön aussieht, sollten Sie auf derartige Sonderzeichen in Permalinks besser verzichten.

6.3.3 Sonderzeichen/Erweiterte Buttons für Non-WYSIWYG serendipity_event_typesetbuttons

Wenn Sie als Redakteur nicht den WYSIWYG-Editor beim Verfassen von Artikeln einsetzen, sind Sie gezwungen (oder eher in der glücklichen Lage), HTML-Code selbständig einzugeben. Serendipity bietet zwar zum Einfügen von Bildern/Links und für gängige Textformatierung (fett, unterstrichen, kursiv) eigenständige Buttons an, aber dies ist Ihnen möglicherweise noch nicht komfortabel genug.

Glücklicherweise erlaubt die Serendipity-Plugin-API, dass Ereignis-Plugins eigenständig Buttons oberhalb der Artikelmaske einbinden können. Das Plugin *Sonderzeichen/Erweiterte Buttons für Non-WYSIWYG* tut genau dies.

In der Plugin-Konfiguration können Sie einstellen, welche Buttons das Plugin darstellen soll. Damit die Artikelmaske nicht überfrachtet wird, sollten Sie hier nur die Buttons aktivieren, die Sie wirklich gerne (und häufig) nutzen möchten.

Eine sehr wichtige Konfigurationsoption des Plugins ist, ob Sie für die Sonderzeichen HTML oder XHTML benutzen wollen. Je nachdem, welchen dieser beiden Standards Sie für Ihre Webseite benutzen wollen, müssen die Sonderzeichen unterschiedlich eingebunden werden. Wenn Ihr Blog-Template das XHTML-Format benutzt (was bei fast allen Serendipity-Templates der Fall ist), würde ein HTML-Sonderzeichen unter Umständen dazu führen, dass Ihre Seite ungültigen Code enthält, der bei manchen Browsern zu Fehldarstellungen der Seite führen könnte.

Da sich XHTML bereits etabliert hat und von allen Browsern unterstützt wird, sollten Sie möglichst XHTML-Sonderzeichen einsetzen. Lediglich *alte Hasen*, die sich von ihren liebgewonnenen HTML-Tags einfach nicht trennen können, interessieren sich womöglich für die HTML-Variante der Sonderzeichen.

Die einzelnen Konfigurationsoptionen des Plugins sind:

Center Button aktivieren

Bindet einen Button ein, mit dem Sie ausgewählten Text im Artikel zentrieren können.
Dies geschicht mittels `<center>`-HTML-Tag oder `<div class="s9y_typeset_center">`-XHTML-Tag.

Strike-through Button aktivieren
Hiermit können Sie Texte als durchgestrichen auszeichnen, um z. B. eine Änderung in Ihren Artikeln hervorzuheben (`<del>` für HTML, `<s>` für XHTML).

Leerzeichen-Button aktivieren
Bei (X)HTML werden mehrere Leerzeichen immer zu einem einzelnen Leerzeichen zusammengefasst. Um mehrere Leerzeichen hintereinander in einem Artikel einzubinden, müssen Sie sogenannte *geschützte* Leerzeichen verwenden. Dieses wird durch das HTML-Sonderzeichen ` ` repräsentiert.

Kaufmännisches-Und-Button aktivieren
Das Kaufmännische Und (`&`) ist bei HTML ein Zeichen, das eine Formatierung für ein Sonderzeichen einleitet. Daher kann man ein `&`-Zeichen nicht einfach alleinstehend einsetzen, sondern muss es speziell maskieren. Der Button bindet dafür das Sonderzeichen `&#38;` ein.

Gedankenstrich-Button aktivieren
Der lange Gedankenstrich wird durch das Sonderzeichen `&#8212;` repräsentiert. Wenn Sie die Option **Use Named Entities** aktivieren, wird stattdessen das (gleichwertige) Sonderzeichen `&mdash;` eingebunden, das man sich zugegebenermaßen für eine manuelle Eingabe besser merken kann.

Kurzer Gedankenstrich-Button aktivieren
Analog zum langen Gedankenstrich gibt es auch einen kurzen Gedankenstrich, der `&#8211;` bzw. `&ndash;` entspricht.

Aufzählungszeichen-Button aktivieren
Das Aufzählungszeichen (ein mittig ausgerichteter Kreis) wird durch das Sonderzeichen `&#8226;` bzw. `&bull;` eingebunden.

Doppelte Anführungszeichen-Button aktivieren
Ein Wort kann in typografischen Anführungszeichen eingebunden werden, wenn Sie diesen Button aktivieren. Welche Anführungszeichen Sie verwenden möchten, können Sie mit der darunterstehenden Option festlegen. Abhängig von der Option **Use Named Entities** werden später auch für diese Anführungszeichen entweder numerische Codes oder die Namen der Sonderzeichen (`&laquo;, &raquo;` etc.) eingefügt.

Choose the type of double quote to use
Hier können Sie einen gewünschten Anführungszeichen-Typ (1 bis 8) auswählen. Welche Anführungszeichen möglich sind, wird anhand von Beispielen dargestellt.

Einfache Anführungszeichen-Botton aktivieren
Analog zu doppelten Anführungszeichen können Sie ein Wort auch in einfachen Anführungszeichen einschließen.

Choose the type of single quote to use
Wie bei den doppelten Anführungszeichen wählen Sie mit dieser Option den gewünschten Typ (1 bis 8) aus.

Apostroph-Button aktivieren, Use real apostrophe
: Der Apostroph (') ist wohl das im deutschen Sprachraum am häufigsten falsch eingesetzte Sonderzeichen. Oft sieht man stattdessen die gedrehte Variante (‘). Auch im HTML-Standard gibt es mehrere Möglichkeiten.

  Das Plugin verwendet entweder `&#39;`, `&#8217;` oder `&rsquo;`.

  `&#39;` wird benutzt, wenn Sie die Option **Use real apostrophe** aktivieren. Dies ist der „einzig wahre“ Apostroph.

  `&#8217;` wird eingesetzt, wenn Sie die beiden Optionen **Use real apostrophe** und **Use Named Entities** nicht aktiviert haben. Das daraus resultierende Sonderzeichen ist eigentlich ein einfaches Anführungszeichen, das wie ein hochgestelltes Komma aussieht.

  `&rsquo;` wird bei deaktiviertem **Use real apostrophe** und aktiviertem **Use Named Entities** angezeigt und sieht genauso wie `&#8217;` aus.

Akzent-Button aktivieren
: Das häufig als Apostroph missbrauchte Zeichen kann über diesen Button eingebunden werden (`&#x0301;`).

Schräger-Akzent-Button aktivieren
: Der umgekehrte Akzent wird über diesen Button als `&#x0300;` eingebunden.

XHTML-1.1-Standard verwenden
: Mit dieser Option bestimmen Sie, ob die Tags für Center und Strike-Through anhand des HTML- oder des XHTML-Standards eingefügt werden sollen.

Use Named Entities
: Wenn Sie diese Option aktivieren, werden anstelle der Namen einiger Sonderzeichen die Zahlencodes verwendet. Wenn Sie sich die Sonderzeichen nicht einprägen wollen, ist es empfehlenswert, diese Option zu deaktivieren. Die Zahlenwerte (wie `&#8211;`) können systemübergreifend besser von Browsern interpretiert werden als die benannten Sonderzeichen (wie `&ndash;`).

Custom HTML-Tags

Eine sehr flexible Option zum Hinzufügen von beliebigen Buttons in Ihre Eintragsmaske stellt die Option **Custom HTML-Tags** dar. Dort können Sie eine Liste von HTML-Codes einfügen, die durch Klick auf den entsprechenden Button einen gewählten Text mit dem gewünschten HTML-Code umgeben.

Ein klassisches Einsatzgebiet hierfür ist die Bereitstellung eigener Formatierungsvorlagen. Wenn Sie Ihren Redakteuren spezielle HTML-DIV-Container in der zentralen Stylesheet-Datei (siehe Kapitel 9.1 ab 470) angelegt haben, den Redakteuren aber nicht zumuten wollen, eigenes HTML zu schreiben, können Sie für die vorhandenen Container jeweils einen eigenen Button hinzufügen. Wenn Sie beispielsweise Zitate durch den HTML-Code

```
<div class="MyQuote">
 Errare humanum est
</div>
```

auszeichnen wollen, können Sie dieses `<div>`-Konstrukt als Button oberhalb der Texteingabemaske einbinden. Der Redakteur muss sein Zitat dann nur markieren, klickt auf den Button, und daraufhin wird der gewählte Text in das aufgeführte HTML-Konstrukt überführt.

Welche HTML-Tags Sie einbinden, ist völlig Ihren Wünschen überlassen. So könnten Sie dort auch einen Button für das Einfügen von JavaScript einbinden.

Die Liste der HTML-Tags tragen Sie in dem Konfigurationsfeld **Custom HTML-Tags** ein. Diese Liste muss in einem ganz speziellen Format vorliegen.

Zum einen werden mehrere Buttons mit dem Zeichen `|` (das *Pipe*-Symbol) voneinander getrennt. Der Übersichtlichkeit halber können Sie nach jedem `|` auch gerne einen Zeilenumbruch einfügen.

Jeder einzelne Button muss daraufhin drei Attribute besitzen: Wie der Button benannt wird, welches HTML-Tag bei einem Klick vor einem ausgewählten Text eingefügt wird und welches HTML-Tag nach der Textauswahl eingefügt wird. Diese drei Attribute werden von dem `@`-Zeichen voneinander getrennt.

Demnach wäre obiges Beispiel durch folgende Syntax umzusetzen:

```
Zitat@<div class="MyQuote">@</div>
```

Der Button wird also mit dem Text *Zitat* in der Symbolleiste oberhalb des Eintrags eingebunden. Bei einem Klick auf diesen Button wird dem Text, den Sie ausgewählt haben, ein `<div...>` vorangestellt und ein `</div>` angefügt.

6.3.4 Trackbacks kontrollieren serendipity_event_trackback

Wenn Serendipity Ihren Artikel speichert, durchsucht es den Eintrag nach allen Links. Dabei werden Trackbacks an automatisch erkannte Blogs verschickt (siehe Seite 433).

Manche Blogs binden jedoch leider ihre Trackback-URL in einem Artikel nicht standardkonform so ein, dass Serendipity sich dorthin automatisch verbinden kann. In einem solchen Fall benötigen Sie das Plugin *Trackbacks kontrollieren*, denn dieses Plugin ermöglicht es Ihnen, manuelle Trackback-URLs zu einem Eintrag hinzuzufügen.

Zusätzlich bietet das Plugin Ihnen die Möglichkeit, ausgehende Trackbacks global in Ihrem Blog zu deaktivieren. Dann sendet Serendipity beim Speichern Ihrer Artikel keinerlei Trackbacks, dies empfiehlt sich also besonders bei versteckten Blogs oder Intranet-Blogs, die mit der Außenwelt nicht in Kontakt treten sollen. Abgesehen von diesem Plugin regelt auch die Variable `$serendipity['noautodiscovery']` der Datei `serendipity_config_local.inc.php` die globale Deaktivierung von Trackbacks, wie auf Seite 175 beschrieben.

Ein weiteres praktisches Feature des Plugins ist, dass Sie in der Konfiguration einen Web-Proxy eintragen können. Ein Proxy ist ein Server, der als Bindeglied zwischen Webserver und dem Internet steht. Oft ist es einem Webserver aus Sicherheitsgründen nicht erlaubt, direkt auf andere Server zuzugreifen. Dies ist auch der häufigste Grund dafür, dass das Spartacus-Plugin (siehe Seite 265) nicht funktioniert. Wenn der Webserver jedoch auf einen Proxy-Server zugreifen darf, können die Zugriffe auf fremde Server über den Proxy geleitet werden.

Auch wenn Serendipity Trackbacks verschickt, muss es direkt auf einen Server zugreifen können. Damit Sie einen Proxy dazwischensetzen können, bietet das Trackback-Plugin die Möglichkeit, einen solchen Server einzutragen. Serendipity wird dann nicht nur beim Senden von Trackbacks, sondern schon beim Zugriff auf Spartacus diesen Proxy-Server benutzen, da fast alle URL-Aufrufe von Serendipity über denselben Mechanismus (die PEAR `HTTP::Request`-Bibliothek) ausgeführt werden.

Abgesehen von der Proxy-Konfiguration bietet das Plugin die Option **Disable the global use of trackbacks**, die bei Aktivierung dafür sorgt, dass keine Trackbacks verschickt werden. Da Serendipity üblicherweise auch Trackbacks zu Ihren eigenen Artikeln schickt, können Sie dies ebenfalls im Plugin über die Option **Send trackbacks to your own blog** deaktivieren.

Wenn Sie das Trackback-Plugin installiert haben und Trackbacks nicht global deaktiviert sind, sehen Sie in den *Erweiterten Optionen* beim Erstellen eines Artikels eine neue Eingabemaske. Dort werden bereits im Eintrag erkannte Links aufgeführt, und zusätzlich können Sie hier weitere URLs eintragen, zu denen Serendipity ein Trackback schicken soll. Mehrere Links werden durch ein Leerzeichen oder einen Zeilenumbruch getrennt.

Mitttels dreier Optionsfelder können Sie bestimmen, wie Trackbacks für den jeweiligen Eintrag gesendet werden sollen:

Trackbacks an erkannte Links im Eintrag senden

Diese Option bewirkt, dass Serendipity automatisch alle im Eintrag eingetragenen URLs auf Trackbacks prüft. Etwaige weitere, manuell eingefügte Links in der Box **Weitere Links für Trackbacks** werden zusätzlich geprüft.

Ausgehende Trackbacks deaktivieren
Ist diese Option ausgewählt, prüft Serendipity weder automatisch die Links innerhalb Ihres Beitrags, noch sendet es Trackbacks an die Liste weiterer, manuell eingefügter URLs.

Trackbacks nur an unten aufgeführte URLs schicken
Wenn Sie Trackbacks ausschließlich an die manuell eingetragenen Trackback-URLs senden wollen, wählen Sie diese Option. Dadurch wird die automatische Erkennung von Links in Ihrem Artikel deaktiviert.

Bitte achten Sie darauf, dass Serendipity bei jedem Speichern eines veröffentlichten Eintrags Trackbacks prüft. Wenn Sie also später mal einen Artikel überarbeiten, bei dem Sie ursprünglich die Option zum Versenden von Trackbacks deaktiviert hatten, müssen Sie daran denken, auch beim neuen Speichern diese Option zu deaktivieren.

Stellen wir uns also konkret folgendes Beispiel vor: Sie schreiben einen Artikel zu einem fremden Blog-Beitrag auf `http://example.com/wp/?p=15`. Dafür würden Sie folgenden HTML-Code in Ihrem Beitrag verwenden:

```
Hanni und Nanni <a href="http://example.com/wp/?p=15">berichten in ihrem
Blog</a> darüber, wie man von WordPress auf Serendipity umsteigen kann
und damit höchste Glückseligkeit erreicht. Sehr spannender Artikel, ich
habe es direkt ausprobiert und lasse gerade mein Shakra richtig entspannt
herumbaumeln.
```

Wenn Sie diesen Artikel speichern, greift Serendipity auf die genannte URL `http://example.com/wp/?p=1` zu. Wäre in dem Beitrag die Trackback-URL korrekt enthalten, würde Serendipity das Trackback direkt abschicken. In unserem Beispiel gehen wir aber davon aus, dass die automatische Erkennung fehlschlägt, da Hanni und Nanni leider ein Blog-Template benutzen, das die Trackback-URL nicht korrekt über RDF-Metadaten einbindet (siehe Trackback-Kapitel ab Seite 433).

Daher müssen wir uns nun auf die Jagd nach der Trackback-URL machen. Dazu rufen wir das Blog selber im Browser auf, und meist stößt man auf eine Angabe des Blogs zu einer manuellen Trackback-URL. Auch Serendipity gibt eine derartige URL auf den Artikel-Detailseiten für fremde Blogsysteme an. In unserem Fall wäre die gewünschte URL `http://example.com/wp/wp-trackback.php?id=15`. Diese URL notieren Sie sich und fügen sie in die Eingabebox **Weitere Links für Trackbacks** ein, speichern den Artikel dann erneut ab – und Serendipity wird nun das Trackback an diese genannte URL schicken, da es die automatische Erkennung nicht mehr benötigt.

Bei dem Beispiel ist es extrem wichtig, dass Sie die manuelle Trackback-URL *nicht* im Artikel erwähnen! Wenn Sie dies machen, würde Serendipity weiterhin denken, es müsste bei

dieser URL einen automatischen Trackback erkennen, der zwangsweise scheitern wird. Wie man in so einem Fall Trackbacks erneut vollständig ausführen kann, entnehmen Sie ebenfalls Kapitel 8.1 ab Seite 437.

6.3.5 Display RSS-Feed in Backend Overview serendipity_event_backendrss

Wenn Sie sich ins Backend einloggen, sehen Sie üblicherweise eine ziemlich leere Seite. Diesen Platz können Sie ausnutzen, um etwas gehaltvollere Informationen darzustellen, wie beispielsweise einen RSS-Feed.

Ähnlich wie das Plugin *Fremder RSS/OPML-Blogroll Feed* (siehe Seite 210) bietet das Plugin *Display RSS-Feed in Backend Overview* die Möglichkeit, beliebige RSS-Feeds einzubinden. Die Konfigurationsoptionen sind dabei ähnlich gestaltet:

Anzahl der Einträge
: Bestimmt, wie viele Einträge aus dem RSS-Feed angezeigt werden sollen. Die Zahl `0` entspricht allen Einträgen des Feeds. Sie können immer nur maximal so viele Einträge darstellen, wie der Feed enthält.

Feed-Titel
: Hiermit legen Sie die Überschrift des RSS-Feeds fest. Da Sie mehrere RSS-Feeds im Backend darstellen können, sollten Sie die Überschriften zur Abgrenzung eindeutig vergeben.

RSS/OPML-URI
: In diesem Eingabefeld tragen Sie die vollständige URL des RSS-Feeds ein, den Sie darstellen möchten.

Zeichensatz
: Die Option **Zeichensatz** legt fest, in welchem Zeichensatz der fremde RSS-Feed formatiert ist. Bei den meisten RSS-Feeds ist dies **UTF-8**.

Link-Target
: Der RSS-Feed zeigt nur die Überschriften der Beiträge an, ein Klick auf den jeweiligen Beitrag öffnet diesen standardmäßig in einem neuen Fenster (`_blank`). Wenn Sie stattdessen den Beitrag gerne im selben Browser-Fenster lesen möchten, können Sie hier ein anderes Ziel (z. B. `_self` oder leer) eintragen.

Wann wird der Feed aktualisiert
: Der RSS-Feed wird vom Plugin gecached, damit nicht bei jedem Aufruf des Backends der fremde Webserver kontaktiert wird. Tragen Sie in diesem Feld ein, wie lange ein Feed zwischengespeichert werden soll. Standardmäßig sind dies 3 Tage (10800 Sekunden).

Sie können diesen RSS-Feed in der Admin-Oberfläche gut dazu benutzen, den RSS-Feed vom offiziellen Serendipity-Blog[14] zu abonnieren. So können Sie direkt beim Login in Ihr Blog erkennen, ob es möglicherweise neue Versionen der Software gibt, in der z. B. Sicherheitslücken behoben worden sind.

6.3.6 HTML-Code für den head-Bereich (HTML-Kopf-Klotz) und HTML Nugget on Page serendipity_event_head_nugget, serendipity_event_page_nugget

Ähnlich wie der berühmte HTML-Klotz (siehe Seite 216) ermöglicht es das Plugin *HTML-Code für den head-Bereich*, beliebigen HTML-Code (oder JavaScript) in Ihr Blog einzubinden. Dieser HTML-Code wird dann im HTML-`<head>`-Bereich eingebunden. In diesem Bereich findet man üblicherweise Meta-Tags und seitenübergreifende JavaScripts. Google Analytics beispielsweise muss in diesen Kopf-Bereich eingefügt werden.

Grundsätzlich können Sie derartigen Code natürlich auch in die Template-Datei `index.tpl` einfügen. Der Nachteil dieser Methode wäre allerdings, dass Sie ihn bei jedem neuen Template neu einfügen müssen und nicht einfach das Template wechseln können. Auch müssen Sie zum Einfügen extra das FTP-Programm und einen Editor bemühen, während der HTML-Kopf-Klotz eine einfachere Eingabe über ein zentrales Plugin ermöglicht.

Zugleich ist dieses Plugin wohl das kleinste existierende Serendipity-Plugin, und daher sehr geeignet, um einmal in die Programmierweise von Plugins hineinzuschnuppern.

Komplexere Möglichkeiten zum Einfügen von *Klötzen* bietet das Plugin namens *HTML Nugget on Page* (`serendipity_event_page_nugget`). Dort kann man einen Klotz beliebig im Kopf-Bereich oder im Inhaltsbereich (Header, Footer, Seitenende) platzieren. Dieses Plugin ermöglicht auch die Benutzung der verfügbaren Textformatierungs-Plugins und lässt sich analog zum *HTML-Klotz* auch so konfigurieren, dass es nur auf Artikel-Detailseiten oder nur auf Artikel-Übersichtsseiten angezeigt wird.

6.3.7 Versioning of entries serendipity_event_versioning

Jedesmal wenn Sie einen Blog-Artikel neu speichern, wird die vorige Version überschrieben. Wenn Sie einen Beitrag versehentlich durch ein einziges Wort ersetzen und speichern, ist also sämtlicher voriger Text unwiderruflich verloren.[15]

Um dieses Problem zu umgehen, gibt es das Plugin *Versioning of entries*. Wie der englische Titel suggeriert, kümmert sich das Plugin um eine Versionskontrolle von Artikeln. Jedes Speichern eines Artikels mit unterschiedlichem Text führt dazu, dass das Plugin den Artikeltext

[14] `http://blog.s9y.org/rss.php`

[15] Es sei denn, Sie haben vorausschauend regelmäßig Backups angelegt.

(*Eintrag* und *Erweiterter Eintrag*) in einer zweiten Datenbanktabelle (`serendipity_versioning`) speichert. Diese Datensatzkopie wird mit dem aktuellen Datum und Autor verknüpft.

In der Konfiguration des Plugins können Sie einstellen, ob mehrere Versionen eines Artikels auch für Ihre Besucher im Frontend dargestellt werden sollen. Ein Besucher kann dann eine beliebige vorige Version aufrufen und so nachlesen, was sich zwischen den einzelnen Bearbeitungszuständen verändert hat.

Ein Redakteur hat im Abschnitt *Erweiterte Optionen* beim Erstellen oder Bearbeiten eines Artikels die Möglichkeit, eine beliebige vorhergehende Version wiederherzustellen und neu abzuspeichern. Die Zwischenversionen bleiben dabei intakt, da die Übernahme einer alten Version lediglich eine neue, aktuelle Version einführt.

Bitte beachten Sie, dass das Plugin außer dem Artikeltext keine anderen Veränderungen[16] speichern und auch in der jetzigen Version keinen Vergleich innerhalb der Versionen durchführen kann.

Datenbanktabelle

Die Datenbanktabelle `serendipity_versioning` enthält alle gespeicherten Versionen eines Artikels:

[16] Beispielsweise an *Erweiterten Optionen*, der Kategoriezuweisung oder auch dem Artikeltitel.

`id`
: enthält die fortlaufende Versionsnummer.

`entry_id`
: enthält die Artikel-ID der jeweiligen Version.

`version`
: enthält eine numerisch erhöhte Versionsnummer.

`body`
: enthält den Artikeltext dieser Version.

`extended`
: enthält den erweiterten Artikeltext dieser Version.

`version_date`
: enthält das Datum, an dem diese Revision angelegt wurde.

`version_author`
: enthält die ID des Redakteurs, der diese Revision erstellt hat.

6.3.8 Cronjob scheduler serendipity_event_cronjob

Der Begriff *Cronjob* ist die Kurzform für eine wiederkehrende (*chronologische*) Aufgabe und stammt ursprünglich von Unix-Betriebssystemen. Bei derartigen Betriebssystemen kann in beliebigen Zeit-Intervallen automatisch ein Programm oder eine Aktion ausgeführt werden.

Solche regelmäßigen Cronjobs sind auch im Blog-Umfeld von Interesse. Beispielsweise bietet Serendipity Plugins an, die regelmäßig RSS-Feeds importieren (Aggregator, siehe Seite 338) oder regelmäßig E-Mails als Blog-Artikel abrufen (Popfetcher, siehe Seite 343).

Eine solch regelmäßige Aktion kann Ihr PHP-Webserver üblicherweise nicht selbst ausführen. Serendipity ist keine Anwendung, die wie ein Betriebssystem ständig im Hintergrund läuft, sondern man muss sich das System eher wie einen Bären im Winterschlaf vorstellen. Der Bär ist nur dann wach, wenn gerade ein Besucher seine Höhle aufsucht und ihn mit großem Lärm aufweckt. Wenn der ungebetene Gast vom Bären *versorgt* wurde, kehrt wieder Ruhe ein und der Bär schläft weiter. Für unser Blog bedeutet das: Wenn kein Besucher oder Redakteur gerade das Blog aufruft, ist Serendipity inaktiv und kann keine Aktionen ausführen.

Daher kann Serendipity von sich aus nicht einfach einen Vorgang auslösen, sondern ist abhängig von Ihren Aktionen. Bei der Benutzung der genannten Plugins müssen Sie selber dafür sorgen, dass eine spezielle URL aufgerufen wird, die wiederum die eigentliche Funktionalität aufruft. Wenn Sie einen eigenen Webserver administrieren, könnten Sie diese URL in die Liste der Cronjobs (die *Crontab*) aufnehmen und automatisch aufrufen.

Jedoch ist diese Möglichkeit für viele Benutzer keine Option. Hier kommt das Plugin *Cronjob scheduler* ins Spiel. Das Plugin ermöglicht es Ihnen, unterstützte Plugins in einem gewünschten Intervall aufzurufen.

Dabei bedient sich das Plugin eines kleinen Tricks. Es bindet im Frontend eine unsichtbare Datei ein, die von jedem Besucher aufgerufen wird, ohne dass dieser das merkt. Die Datei prüft, wann der letzte automatische Aufruf stattfand, und startet gegebenenfalls die notwendigen Aktionen. Andernfalls tut das Plugin nichts und wartet auf den nächsten Aufruf.

Dieser Aufruf durch einen Besucher hat natürlich den Nachteil, dass man niemals exakt voraussagen kann, ob auch wirklich zum benötigten Zeitpunkt jemand das Blog besucht. Wenn ein Job alle fünf Minuten ausgeführt werden soll, aber statistisch nur alle zwei Stunden ein Besucher vorbeikommt, dann wird dieser Job faktisch auch nur alle zwei Stunden ausgeführt. In den meisten Fällen ist das kein Problem, da, wenn es früher keine Besucher gibt, die neu importierten Daten des Jobs sowieso nicht benötigt werden. Das Cronjob-Plugin wird zentral über die URL `http://www.example.com/serendipity/index.php?serendipity[cronjob]=true` aufgerufen. Dies ist die erwähnte *unsichtbare Datei*, die ein Besucher aufruft. Natürlich können Sie diese URL auch von einer Crontab aus alle fünf Minuten aufrufen, anstatt dies Ihren Besuchern zu überlassen. In so einem Fall können Sie das Plugin über die Konfigurationsoptionen so einstellen, dass die **visitor-based cronjobs** deaktiviert werden.

Welche Aktionen der Cronjob ausführt, stellen Sie nicht innerhalb des Cronjob-Plugins ein, sondern in dem jeweiligen Plugin. Bei jedem unterstützten Plugin können Sie in den Konfigurationsoptionen einstellen, in welchem Zeitraum (alle fünf Minuten, 30 Minuten, stündlich, halbtäglich, täglich, wöchentlich, monatlich) die Aktion ausgeführt wird. In der Konfiguration des Cronjob-Plugins sehen Sie eine Liste aller eingerichteten Cronjobs, und wann diese zuletzt ausgeführt worden sind.

Bei jedem Aufruf schreibt das Plugin einen kleinen Logfile-Eintrag in die Datenbanktabelle `serendipity_cronjoblog`. Diese können Sie jederzeit leeren (nicht löschen), wenn Sie Speicherplatz sparen wollen. Die Spalte `timestamp` enthält das Datum der Cronjob-Ausführung, `type` die Cronjob-Art und `reason` einen Informationstext zur Ausführung.

6.3.9 QuickNotes serendipity_event_adminnotes

Die Kommunikation unter Redakteuren läuft in einem Blog meist informell ab: Man spricht sich persönlich, via Telefon oder E-Mail ab – und Gerüchten zufolge ist es vielen Bloggern auch egal, was ihre Co-Autoren so treiben.

Das Plugin *QuickNotes* nimmt sich des Kommunikationsproblems an und verwandelt die Startseite des Backends in ein Nachrichtenzentrum. Dort können Chefredakteure Anweisungen an die Benutzer weiterreichen, Administratoren können über etwaige neue Plugins berichten und einfache Redakteure über ihre Artikel informieren.

Die Nachrichten arbeiten grundsätzlich gruppenorientiert. Eine Nachricht wird immer für eine festgelegte Autorengruppe eingeblendet, das heißt, dass Mitglieder anderer Gruppen

die Nachricht nicht sehen können. Notizen können zudem nur an die Gruppen geschrieben werden, in denen man Mitglied ist. Daher ist es unter Umständen für dieses Plugin notwendig, dass Sie als Administrator auch anderen Benutzergruppen beitreten (siehe Seite 191), um Nachrichten an diese zu verfassen.

In der Übersicht werden QuickNotes wie in folgender Abbildung dargestellt:

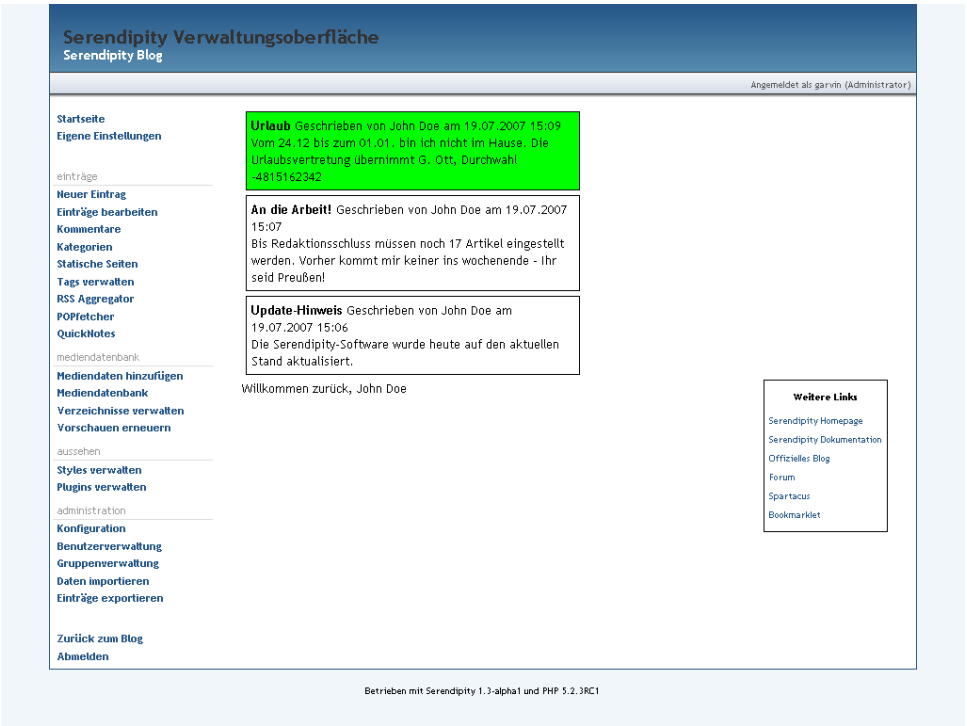

Abbildung 6.1: Beispiel für QuickNotes

Nachrichten, die Sie noch nicht gelesen haben, werden speziell hervorgehoben, bereits gelesene Nachrichten werden normal dargestellt. In jedem Kasten befindet sich der Titel einer Notiz, die Angabe des Herausgebers der Information und das dazugehörige Datum. Besonders lange Hinweise können nach einer gewissen Textmenge abgeschnitten werden, ein Klick auf den Button **Alle Optionen ein/ausblenden** zeigt den vollständigen Text an. Alle QuickNotes werden chronologisch sortiert, ihre Anzahl richtet sich nach der Konfiguration des Plugins.

Sie können eine neue Notiz anlegen, indem Sie auf den Menüpunkt **Einträge** → **QuickNotes** klicken. Dort sehen Sie eine Liste aller Notizen, chronologisch sortiert. Mittels der Buttons **Bearbeiten** und **Löschen** können Sie eine bestehende Notiz verändern, der Button **Neuer Eintrag** legt eine neue Notiz an.

Beim Anlegen einer neuen Notiz tragen Sie den **Titel** und den **Text** in den vorhergesehe-

nen Feldern ein. In dem Mehrfach-Auswahlfeld **Gruppenzugehörigkeit** markieren Sie alle Gruppen, für die die Nachricht bestimmt ist.

Die Konfigurationsoptionen des Plugins ermöglichen Ihnen weitere Einstellungen zu Menge und Darstellung der Notizen:

Sollen Autoren eigene Hinweise anlegen dürfen
: Wenn Sie diese Option nicht aktivieren, können nur Chefredakteure und Administratoren[17] QuickNotes anlegen. Bei aktivierter Option können auch normale Autoren Nachrichten an die Mitglieder ihrer Benutzergruppen senden.

Wie viele Elemente sollen angezeigt werden
: Tragen Sie in dieses Feld die Anzahl der QuickNotes ein, die auf der Startseite angezeigt werden sollen.

HTML-Formatierung erlauben
: Standardmäßig dürfen die QuickNotes (aus Sicherheitsgründen) nur normalen Text enthalten und keinen HTML-Code. So können Sie verhindern, dass böswillige Redakteure möglicherweise JavaScript einschleusen. Mit dieser Option können Sie gezielt HTML-Formatierungen auch nur für Benutzer mit Administrator-Rang zugänglich machen.

Textformatierung(en) durchführen
: Der Text einer QuickNote kann, wie auch bei Blog-Artikeln üblich, mit den installierten Textformatierungs-Plugins behandelt werden. Standardmäßig ist dies aktiviert. Wenn Sie jedoch den Text ohne weitere Umformatierung darstellen wollen, können Sie diese Option deaktivieren.

Nachrichten nach X Bytes kürzen
: Lange Notizen werden in der Übersicht nach einer festgelegten Anzahl an Zeichen gekürzt. Längere Notizen müssen dann erst durch einen Klick auf den Button **Alle Optionen ein/ausblenden** gezielt geöffnet werden und belegen so keinen wertvollen Bildschirmplatz.

Das Aussehen der Admin-Notes können Sie über das Admin-Stylesheet (siehe Seite 470) beeinflussen, indem Sie die CSS-Klassen `.serendipity_note` und `.note_new` der Plugin-Datei `plugins/serendipity_event_adminnotes/notes.css` entweder anpassen oder in die `admin/style.css`-Template-Datei übernehmen. Weitere HTML-Klassen bei der Ausgabe der Notizen sind verfügbar, diese können Sie im Bedarfsfall anhand des HTML-Quelltextes in Ihrem Browser einsehen und dann dem Stylesheet hinzufügen.

[17] Die Identifikation erfolgt anhand des Benutzerrangs, nicht der Zugehörigkeit zu den gleichnamigen Benutzergruppen.

Datenbanktabellen

QuickNotes werden in der Datenbanktabelle `serendipity_adminnotes` gespeichert:

`noteid`
: enthält die fortlaufende ID einer QuickNote.

`authorid`
: enthält die ID des Redakteurs, der die QuickNote erstellt hat.

`notetime`
: enthält die Erstellungszeit der QuickNote.

`subject`
: enthält den Betreff der QuickNote.

`body`
: enthält den Inhalt der QuickNote.

`notetype`
: gibt die Art der QuickNote an (derzeit nur `note` unterstützt).

`serendipity_adminnotes_to_groups` ist eine 1:n-Zuordnungstabelle, die erstellte QuickNotes den Empfänger-Benutzergruppen zuordnet. In Spalte `noteid` wird die ID der QuickNote gespeichert, `groupid` enthält die ID der jeweiligen Benutzergruppe.

6.3.10 Benutzerprofile serendipity_event_userprofiles

Bei Blogs, in denen mehrere Redakteure schreiben, ist es für den Besucher oft interessant, Details über einzelne Redakteure zu erfahren. Redakteure wiederum möchten möglicherweise gerne auch etwas mehr Informationen über sich preisgeben.

Viele Blogs regeln dies durch Einführungsartikel, in denen sich ein Redakteur vorstellt. Solche Artikel sind aber später möglicherweise nicht immer im Blickfeld eines Besuchers.

Deshalb bietet das Plugin *Benutzerprofile* eine Möglichkeit, mit der jeder Redakteur Details zu seiner Person in vorausgefüllte Felder (wie Wohnort, Land, Hobbies, Homepage etc.) eintragen kann. Diese Felder können Besucher des Blogs dann einsehen, wenn sie im Frontend auf den Namen eines Redakteurs klicken. Oberhalb der Übersicht der von dem jeweiligen Redakteur geschriebenen Artikel sehen sie dann eine Box mit den eingetragenen Profildaten.

Die Benutzerprofile verwaltet das Plugin, indem es sich in den Admin-Bereich **Eigene Einstellungen** einklinkt. Unterhalb der bekannten Einstellungen sehen Sie einen neuen Bereich, in dem die verfügbaren Redakteure aufgelistet sind, wobei Ihr Name standardmäßig hervorgehoben ist. Direkt darunter befinden sich die jeweiligen Profilfelder wie *Voller Name*, *Homepage*, *Hobbies* und so weiter. Dort kann der jeweilige Redakteur sämtliche Felder seiner

Datei ausfüllen und auf **Speichern** klicken. Bitte beachten Sie, dass aus technischen Gründen Geburtstage vor dem 01.01.1970[18] von dem Plugin leider nicht korrekt ausgewertet werden können.

Zusätzlich bindet das Plugin dieselbe Oberfläche auch unter dem eigenständigen Menüpunkt **Administration** → **Benutzerprofile** ein.

In dem Ausklappfeld der Benutzerprofile können Sie auch die Daten anderer Redakteure bequem einsehen. Als höherrangiger Chefredakteur oder Administrator können Sie außerdem alle Profildaten der niederrangigeren Redakteure bearbeiten. Profildaten von Redakteuren des gleichen Benutzerranges können Sie nur ansehen, aber nicht bearbeiten.

Wenn Sie auf den Button **Erweiterte Optionen** klicken, können Sie dort einstellen, welche Ihrer eingetragenen Profildaten für Besucher des Blogs angezeigt werden sollen. Alle anderen Daten sind nur für eingeloggte Redakteure über die Benutzerprofilauswahl im Backend zu sehen. So können Sie bestimmte Daten, wie Ihre Telefonnummer, nur den Redakteuren zugänglich machen. Auch werden im Frontend nur dann wirklich aktivierte Datensätze angezeigt, wenn Sie sie auch ausgefüllt haben. Wenn Sie also keine Hobbies eingetragen haben, wird das leere Feld einfach übersprungen. Standardmäßig sind alle Profilfelder deaktiviert, damit ein Redakteur nur auf seinen ausdrücklichen Wunsch seine Informationen veröffentlichen kann.

Bearbeiten Sie einen Datensatz, erscheint neben dem Ausklappfeld der Benutzerprofile ein weiterer Button **VCard-Datei erstellen**. Wenn Sie darauf klicken, wird eine VCard-Datei in der Mediendatenbank erstellt (`uploads/`*`Benutzername`*`.vcf`). Diese Datei können Sie dann herunterladen (`http://www.example.com/serendipity/uploads/`*`Benutzername`*`.vcf`) bzw.
später in Blog-Einträgen einbinden. Eine VCard-Datei ist ein standardisiertes Format für eine Visitenkarte und enthält die Profildaten des Redakteurs. VCards können zum Beispiel in E-Mail-Programme eingebunden werden.

Abgesehen von den Benutzerprofilen bietet das Plugin auch noch weitere Funktionalität an. Zum einen ist dies, dass für jeden Redakteur ein eigenes Bild für dessen Einträge eingebunden werden kann. Ein Besucher muss so nicht erst nach einem Autorennamen suchen, sondern kann anhand des Bildes leichter unterschiedliche Redakteure zuordnen.

Das jeweilige Bild muss dazu zuerst in der Konfiguration des Plugins mit der Option **Bild des Autoren im Eintrag zeigen** aktiviert werden. Die Bilder, die das Plugin einbindet, müssen sich in einem Unterverzeichnis `img` Ihres jeweiligen Template-Verzeichnisses befinden. Der Dateiname richtet sich nach dem vollständigen Namen des Redakteurs sowie der im Plugin konfigurierten **Dateiendung** (standardmäßig jpg). Damit der Redakteursname in einen gültigen Dateinamen umgewandelt werden kann, müssen Sie für den Dateinamen alle Sonderzeichen, Umlaute und Leerzeichen mit einem Unterstrich (_) ersetzen. Für den Redakteur *John Doe* müssen Sie also eine Datei wie `templates/default/img/John_Doe.jpg`

[18] Dieses Datum markiert Sekunde 1 der Unix-Zeitrechnung. Alle Unix-Server arbeiten intern mit sogenannten Timestamps, daher werden auch in vielen Datenbanken Timestamps zur Formatierung von Zeitpunkten verwendet, da damit leichter zu rechnen ist und sie sich einfacher in beliebige Zeitzonen umrechnen lassen.

erstellen. Beachten Sie dabei die Groß- und Kleinschreibung des Redakteursnamens!

Da diese Art der Bildereinbindung relativ komplex ist und ein Redakteur ohne FTP-Zugriff zum Blog keine Bilder einstellen kann, bietet das Plugin auch eine andere Möglichkeit für individuelle Redakteursbilder. Diese können nämlich alternativ über den Webservice `http://www.gravatar.c` eingebunden werden. Der Gravatar-Dienst ermöglicht es Ihnen, eine beliebige Grafikdatei hochzuladen, die mit einer E-Mail-Adresse verbunden wird. Die E-Mail-Adresse eines Redakteurs wird bei der Darstellung seiner Artikel ausgewertet und somit das Bild vom Gravatar-Server eingebunden. Das heißt, jeder Redakteur kann für seine E-Mail-Adresse einen Gravatar registrieren, und dieser wird dann ohne weiteres Zutun dargestellt. Damit dies klappt, muss in der Plugin-Konfiguration die Option **Gravatar-Bild bevorzugen** aktiviert werden. Einige weitere Optionen regeln zudem, welche Standard-Bilddatei bei Redakteuren angezeigt wird, die keinen Gravatar besitzen, und welche Größe die Bilder haben sollen.

Für weitere Möglichkeiten der Gravatar-Unterstützung bietet sich das Plugin **Avatar Plugin** an, das über Spartacus erhältlich ist. Dies bindet Gravatare auch für Besucher des Blogs ein.

Die Option **Anzahl der Kommentare zeigen** des Plugins stellt eine besondere Möglichkeit dar, mit der bei jedem Kommentar eines Benutzers angezeigt wird, wie viele Kommentare dieser insgesamt bereits im Blog hinterlassen hat. Diese Option ist besonders dann interessant, wenn Sie ein geschlossenes Blog führen, in dem Kommentare nur für registrierte Benutzer möglich sind. Solche Benutzer besitzen meist keine Redakteursrechte, sondern kommentieren lediglich Artikel. Mit der Aktivierung der genannten Option kann man als Besucher des Blogs so leicht nachverfolgen, welche Benutzer oft kommentieren. In dem Auswahlfeld dieser Option haben Sie die Wahl zwischen mehreren Optionen.

keine
: Diese Option deaktiviert die Einbindung von Kommentarzählern.

An Kommentartext anhängen
: Bei dieser Option wird die Anzahl der Kommentare in Klammern hinter dem jeweiligen Kommentar angezeigt. Wenn ein Benutzer also den Kommentar „*Das war sehr interessant!*“ schreibt, dann wird dahinter „*Kommentare (42)*“ eingefügt. Der Kommentartext wird innerhalb eines HTML-Containers mit der Klasse `.serendipity_commentcount` eingebunden, den Sie in der CSS-Datei Ihres Templates beliebig formatieren können. Standardmäßig ist dieser Kommentarblock rechtsbündig gesetzt.

Vor Kommentartext setzen
: Diese Option ist identisch zur vorangehenden, setzt den Kommentarzähler aber an den Anfang.

Eigenes Smarty Template
: Statt den Kommentartext zu modifizieren, können Sie die Anzahl der Kommentare auch an einer speziellen Position innerhalb der Kommentardarstellung einbinden. Kommentare werden mit der Smarty-Template-Datei `comments.tpl` dargestellt, dort können Sie die Variable `$comment.plugin_commentcount` an beliebiger Stelle setzen.

Das *Benutzerprofile*-Plugin ermöglicht es Besuchern des Blogs, eine Übersicht aller Benutzergruppen des Blogs einzusehen. Gerade bei Gruppenblogs ist die Aufteilung in Benutzergruppen von großem Interesse, wie man es auch von Foren her kennt. Über `http://www.example.com/serendipity/index.php?serendipity[subpage]=userprofiles` kann der Besucher diese Liste einsehen. Er kann über ein Auswahlfeld die jeweilige Benutzergruppe aussuchen und mit einem Klick auf **Los** alle Mitglieder der Gruppe anzeigen lassen. Standardmäßig werden der Benutzername, die Anzahl der Artikel des Redakteurs, der echte Name des Redakteurs und seine E-Mail-Adresse angezeigt.

Zusätzlich bietet das Benutzerprofile-Plugin zwei kleine Seitenleisten-Plugins an. Zum einen das gleichnamige Plugin *Benutzerprofile*, welches eine Liste aller verfügbaren Benutzer und Benutzergruppen anzeigen kann, und zum anderen das Plugin namens *Geburtstage von Redakteuren*, mit dem Sie die zukünftigen Geburtstage darstellen können.

Das Layout und die dargestellten Datensätze der Benutzerliste können Sie über die Smarty-Template-Datei `plugin_groupmembers.tpl` anpassen. Es gibt dort folgende Variablen, die für Smarty-Kenner wichtig sind:

`$userprofile_groups`: Array(`id, name`)
: Enthält eine Liste aller verfügbaren Gruppen.

  `$userprofile_groups.X.id` enthält die ID einer jeweiligen Gruppe, `$userprofile_groups.X.name` den Namen.

`$selected_group`: Int
: Enthält die ID der vom Besucher ausgewählten Gruppe. Falls nicht gesetzt, ist noch keine Gruppe gewählt.

`$selected_group_data`: Array (`id, name`)
: Enthält die ID und den Namen der vom Besucher ausgewählten Gruppe.

`$selected_members`: Array (`realname, username, authorid, email, userlevel, posts` ...) In diesem Array werden die Mitglieder der vom Besucher gewählten Gruppe gespeichert. Als Schlüssel des Arrays sind alle Tabellenfelder der `serendipity_authors`-Tabelle verfügbar sowie zusätzlich der Schlüssel `$selected_membmers.X.posts` für die Anzahl der vom Redakteur geschriebenen Artikel.

Die Darstellung der jeweiligen Benutzerprofile wird über die Template-Datei `plugin_userprofile.tpl` gesteuert. In dieser Datei sind die nötigen IF-Abfragen enthalten, die bestimmen, welche Datensätze angezeigt werden. Folgende Variablen sind verfügbar:

`$userProfile`: Array
: In diesem Array speichert das Plugin alle Angaben eines Benutzerprofils. Für jede Profilangabe besteht ein Schlüssel, dessen Wert die Eingabe des Benutzers enthält:

```
$userProfile.(city, street, country, url, occupation, hobbies,
yahoo, aim, jabber, icq, msn, skype, birthday)
```

`$userProfileProperties`: Array
: Dieses Array enthält eine Liste aller möglichen Konfigurationsoptionen eines Datensatzes:

```
$userProfileProperties.(show_city,
show_street, show_country, show_url, show_occupation, show_hobbies,
show_yahoo, show_aim, show_jabber, show_icq, show_msn, show_skype,
show_birthday, city, street, country, url, occupation, hobbies,
yahoo, aim, jabber, icq, msn, skype, birthday)
```

Zu jedem Schlüssel gibt es ein weiteres Array, in dem der Typ der Eigenschaft und dessen Beschreibung enthalten ist. `$userProfileProperties.show_city.type` enthält „boolean“, `$userProfileProperties.show_city.desc` enthält „Stadt anzeigen“.

`$userProfileLocalProperties`: Array
: Analog zum Array `$userProfileProperties` enthält dieses Array alle lokalen Benutzerdaten. Diese werden nicht im Benutzerprofil gespeichert, sondern stammen aus der Datenbanktabelle `serendipity_authors` und sind gleich benannt:

```
$userProfileLocalProperties.(realname, username, email)
```

Auch hier enthält der Array-Unterschlüssel `.desc` die Beschreibung einer Konfigurationsoption und `.type` den Typen.

`$userProfileTitle`: String
: Diese Variable enthält den Namen des Benutzerprofil-Plugins.

Die Standardausgabe des Plugins lässt sich weiterhin über die CSS-Klasse `.serendipityAuthorProfile` optisch anpassen.

Wenn Sie mehr als die festgelegten Profilfelder anzeigen wollen, können Sie das Plugin relativ leicht anpassen. Dazu müssen Sie `plugins/serendipity_event_userprofiles/serendipity_ev` editieren und die Variablen `$properties` und `$option_properties` ergänzen. Sie können dazu das bestehende Format mittels Kopieren & Einfügen übernehmen und lediglich die Benennung der Felder anpassen. Die neuen Variablen können daraufhin automatisch in der Profilverwaltung eingetragen werden; damit diese angezeigt werden können, müssen Sie noch die jeweiligen Variablen in der Template-Datei `plugin_userprofile.tpl` ergänzen.

Wenn Sie beispielsweise eine neue Profiloption namens „Lieblingsfarbe“ einbauen möchten, fügen Sie Folgendes dem Array `$properties` in der PHP-Datei des Plugins am Ende hinzu:

```
, 'farbe' => array('desc' => 'Lieblingsfarbe',
                   'type' => 'string')
```

Wenn Sie die Farbe als letztes Element einfügen (hinter `'birthday'`), müssen Sie darauf achten, dass hinter der Klammer nach `'date')` ein neues Komma eingefügt werden muss, damit PHP weiß, dass dies nicht das letzte Element einer Liste war. Daher ist in obigem Code-Beispiel das notwendige Komma enthalten. Wenn Sie die Liste fehlerhaft (ohne Komma oder mit doppelten Kommas) eintragen, wird das Plugin nicht mehr funktionieren. Sie müssen den Fehler dann erst aufspüren und korrigieren.

Nun können Sie bereits das neue Profilfeld `farbe` ausfüllen, aber noch nicht bestimmen, ob es angezeigt werden soll. Dazu müssen Sie noch das Array `$option_properties` anpassen und genauso wie vorher eine neue Zeile einfügen:

```
, 'show_farbe' => array('desc' => 'Lieblingsfarbe anzeigen',
                        'type' => 'boolean')
```

Der Redakteur kann jetzt auch bestimmen, dass seine Lieblingsfarbe angezeigt wird. Damit sie auch erscheint, fehlt nur noch eine neue Zeile in der `plugin_userprofile.tpl`-Datei, die Sie ebenfalls hinter der bereits bestehenden Ausgabe an IF-Abfragen einsetzen:

```
{if $userProfile.farbe and $userProfile.show_farbe == "true"}
    <dt>{$userProfileProperties.farbe.desc}</dt>
    <dd>{$userProfile.farbe}</dd>
{/if}
```

Dieser Codeschnipsel prüft, ob ein Redakteur die Lieblingsfarbe ausgefüllt hat und ob er diese auch darstellen möchte. Im folgenden HTML-Tag `<dt>` wird dann die oben eingetragene Beschreibung namens `Lieblingsfarbe` eingefügt, und im HTML-Tag `<dd>` erscheint die Lieblingsfarbe des Redakteurs.

Datenbanktabellen

Das Plugin erstellt die Datenbanktabelle `serendipity_profiles`, in der die jeweiligen Profileigenschaften zu jedem Autor gespeichert werden.

Jeder Datenbankeintrag steht dabei für eine Profileigenschaft, so dass beliebig viele und verschiedene Eigenschaften für einen Autor vergeben werden können. Das Feld `authorid` enthält dabei die ID des Redakteurs, `property` enthält den Namen der Profileigenschaft und `value` den jeweils zugeordneten Wert.

6.3.11 Seitenleisten ein-/ausklappbar machen serendipity_event_sidebarhider

Als *Seitenleiste* gelten in Serendipity die Bereiche des Frontends, in denen Ihre Seitenleisten-Plugins dargestellt werden. Meist sind sie links und/oder rechts neben den Artikeln eingebunden.

Mit dem Plugin *Seitenleisten ein-/ausklappbar machen* können Sie zwei Dinge für jedes Element der Seitenleiste anpassen. Zum einen können Sie Inhalte der jeweiligen Seitenleisten-Plugin-Container ausblenden, so dass sie der Benutzer durch einen Klick auf einen Button wieder einblenden kann. Dadurch sparen Sie einiges an Bildschirmplatz, jedoch müssen die Besucher dann gezielt Seitenleisten-Container ausklappen, um deren Inhalt zu sehen.

Zum anderen können Sie bestimmte Elemente in gewissen Fällen vollständig ausblenden.

Sobald das Plugin installiert wurde, bietet es einen neuen Menüpunkt im Backend-Bereich **Einträge** → **Seitenleisten verwalten** an. Zugleich ändert es etwas in der Darstellungsart der Seitenleisten in Ihrem Frontend – dort ist für jedes Seitenleisten-Plugin ein kleiner Link **-** bzw. **+** hinzugekommen. Dank eines JavaScripts kann ein Besucher nun auf diesen Link klicken, und das entsprechende Plugin wird ein- bzw. ausgeblendet, um weniger Platz zu belegen. Beachten Sie, dass diese Icons/Links nur dann verfügbar sind, wenn Ihr gewähltes Template die CSS-Klassen des Standard-Templates benutzt. Ohne diese übereinstimmende Benennung kann das JavaScript nicht aktiv werden (Details siehe Seite 477).

Über das neue Menü **Einträge** → **Seitenleisten verwalten** im Backend können Sie nun detailliert einstellen, wie die Seitenleisten-Plugins versteckt werden sollen. Wenn Sie diesen Bereich betreten, sehen Sie eine Oberfläche, die an die **Pluginverwaltung** angelehnt ist. Für jede Seitenleiste des Templates werden die dort eingerichteten Seitenleisten-Plugins in der eingestellten Reihenfolge angezeigt.

Standardmäßig sind alle Seitenleistenelemente ausgeklappt, das heißt, der Besucher muss sie von sich aus verstecken. Mithilfe der Oberfläche können Sie jedoch auch auswählen, welche Seitenleisten-Plugins standardmäßig versteckt sein sollen, indem Sie für jedes gewünschte Plugin die Auswahlbox **Versteckt** aktivieren.

Unterhalb dieser Auswahlbox befinden sich weitere Einstellungsmöglichkeiten für die Sichtbarkeit des jeweiligen Seitenleistenelements. So können Sie auch festlegen, ob ein Plugin nur für spezielle Benutzer angezeigt werden soll. Wenn Sie die Option **Jeder Besucher** auswählen, kann jeder Besucher das jeweilige Element sehen. Bei Aktivierung der Option **Nur Mitglieder** wird das Seitenleisten-Plugin nur Redakteuren angezeigt, die sich vorher im Backend eingeloggt haben. Die Option **Nur ich** legt fest, dass ein Seitenleisten-Plugin nur für den Redakteur angezeigt wird, der gerade im Backend die Änderungen vornimmt (also Sie selbst). Jeder Redakteur, der Rechte zum Erstellen von Einträgen hat, darf standardmäßig die Elemente der Seitenleisten verwalten. Wollen Sie dies nur einigen Redakteuren genehmigen, müssen Sie in der Gruppenverwaltung das Plugin über den Event-Hook `backend_sidebar_entries_event_display_sidebarhider` für andere Benutzergruppen als die gewünschten verbieten (siehe Seite 191).

Ein Seitenleisten-Element kann so konfiguriert werden, dass es nur für Mitglieder einer gewählten Benutzergruppe dargestellt wird. In der Mehrfach-Auswahlbox zum Konfigurationspunkt **Gruppe** müssen Sie daher alle Benutzergruppen wählen, für die ein Seitenleisten-Plugin angezeigt werden soll. Wählen Sie keine Gruppe aus, wird ein Plugin für alle Benutzergruppen angezeigt. Sobald Sie hier eine Gruppe auswählen, entspricht dies automatisch der Einstellung **Nur Mitglieder**, da nicht eingeloggte Benutzer keiner Gruppe angehören und somit ein Plugins niemals sehen.

Abgesehen vom Verstecken eines Plugins in Abhängigkeit vom Status eines Besuchers können Sie ein Seitenleisten-Plugin auch dann verstecken, wenn ein Besucher eine spezielle Kategorie des Blogs ansieht. Auch hier gilt wieder, dass Sie bei der Benutzung der **Nur Kategorien**-Option alle Kategorien auswählen müssen, bei denen ein Seitenleistenelement angezeigt werden soll. Die Sonderoption **Alle Kategorien** ist standardmäßig vorausgewählt, so dass ein Seitenleisten-Plugin immer angezeigt wird. **¡Front Page¿** bezieht sich auf die Standard-Artikelübersicht von Serendipity, in der ein Besucher noch keine besondere Einschränkung zu einer gewählten Kategorie getroffen hat.

Sobald ein Plugin über die Optionen **Nur Mitglieder/Nur ich/Gruppe/Nur Kategorien** ausgeblendet wird, kann ein Besucher im Frontend das Plugin (abhängig von seinem Status) nicht mehr sehen. Es wird also nicht einfach nur standardmäßig eingeklappt, sondern es wird vollständig aus der Ausgabe entfernt.

Diese Möglichkeit können Sie gezielt benutzen, um gewisse HTML-Klötze oder andere Seitenleistenelemente kategoriebezogen anzuzeigen oder bestimmte Informationen nur Redakteuren zugänglich zu machen.

Nachdem Sie Änderungen an der Sichtbarkeit der Plugins vorgenommen haben, müssen Sie diese über den Button **Änderungen am Layout speichern** aktivieren. Wenn Sie später einmal die Reihenfolge der Plugins über die übliche Plugin-Verwaltung ändern, müssen Sie das Layout der eingeklappten Seitenleisten-Plugins erneut anpassen. Die Seitenleistenelemente werden anhand ihrer Positionsindizes versteckt. Wenn Sie die Position des ersten und zweiten Plugins vertauschen, würde nach wie vor das erste Seitenleistenelement versteckt werden, anstelle des eigentlich logischen zweiten Plugins. Nur durch eine entsprechende Einstellung über die Layout-Verwaltung des *Seitenleisten-verstecken*-Plugins kann dies korrigiert werden.

Die Funktionalität des Ein- und Ausklappens eines Seitenleisten-Plugins kann über die globale Konfiguration des Plugins beinflusst werden. Wenn Sie diese JavaScript-Funktionalität nicht benötigen, sondern nur das Anzeigen/Verstecken von Seitenleisten-Plugins nutzen möchten, können Sie dies gezielt über die Option **Seitenleisten ein-/ausklappbar machen** einstellen.

Die weiteren Konfigurationsoptionen können Sie benutzen, um die Gestaltung des Ein- und Ausklappens an Ihr Template anzupassen. Damit dies überhaupt klappen kann, muss die Smarty-Template-Datei `sidebar.tpl` Ihres Templates von dem Standard-Template abgeleitet sein. Das JavaScript benötigt einen HTML-Container mit der CSS-Klasse `.serendipitySideBarItem`, in dem die jeweiligen Seitenleistenelemente eingebunden sein müssen. Jedes Seitenleistenelement muss den Titel des Plugins in einem HTML-Container mit der CSS-

Klasse `.serendipitySideBarTitle` einbinden, und den Inhalt mit der CSS-Klasse `.serendipitySideBarContent`. Schauen Sie sich die Datei `templates/default/sidebar.tpl` für ein Beispiel dieser Gestaltung an. Das JavaScript kann ebenfalls standardmäßig nur mit einer linken und einer rechten Seitenleiste umgehen, die jeweils in einem HTML-Container mit der ID `#serendipityLeftSideBar` und `#serendipityRightSideBar` eingebunden sind. Wenn Ihr Template andere Seitenleisten einbindet, müssen Sie gegebenenfalls die Datei `sidebarhider.js` des Plugins anpassen und die Funktion `sideBarHideRun()` mit Ihren eigenen Seitenleisten-Containernamen erweitern.

Sobald durch das JavaScript ein Seitenleisten-Element versteckt wird, weist es einen speziellen CSS-Stil zu, den Sie über die Konfigurationsoptionen des Plugins manuell beeinflussen können.

Anstelle der HTML-Links + und - können Sie auch beliebigen anderen HTML-Code einbinden, zum Beispiel eine Grafik oder einen ausführlichen Text wie `Verstecken` und `Anzeigen`.

Sollten Ihnen die Optionen dieses Plugins nicht ausreichen, um ein Seitenleisten-Plugin abhängig vom umgebenden Inhalt zu verstecken, müssen Sie dies manuell mittels Smarty-Syntax in der `sidebar.tpl` lösen. Dort können Sie auch beispielsweise Abfragen einfügen, so dass bestimmte Seitenleistenelemente für statische Seiten nicht angezeigt werden. Generelle Informationen zu diesen Anpassungsmöglichkeiten finden Sie ab Seite 480.

6.3.12 Externe PHP-Anwendung serendipity_event_externalphp

Oft werden Sie sich wünschen, Ihr Blog mit anderen, externen PHP-Anwendungen zu erweitern. Serendipity kann nicht alles, und gerade bei der Einbindung von Bildergalerien möchten viele Blog-Betreiber gerne ihr liebgewonnenes XYZ-Script einsetzen.

Serendipity versteht sich als ein Framework, das sowohl die Einbindung in andere Systeme erleichtern, als auch andere Systeme einbinden können soll. Viele Möglichkeiten der Einbindung eröffnen sich durch die geschickte Anwendung von Smarty-Templating (siehe Seite 480) oder von eigenen Ereignis-Plugins (siehe Seite 642).

Wem dies zu komplex ist, der kann mit dem kleinen Plugin *Externe PHP-Anwendung* ein PHP-Script *includen*[19]. Das Plugin darf nur von Administratoren konfiguriert werden, da die Einbindung fremder PHP-Anwendungen potenziell Sicherheitslücken öffnen kann, um beliebigen Code von Serendipity auszuführen. Benutzen Sie es daher mit Vorsicht und binden Sie nur vertrauenswürdige PHP-Anwendungen ein!

[19] Der Fachterminus *include* bezeichnet ein PHP-Konstrukt, das eine Datei mit PHP-Anweisungen aus einem anderen Script heraus einlesen und interpretieren kann.

Das Ziel des Plugins ist, dass Sie bei dem Aufruf einer konfigurierbaren URL die Inhalte eines fremden PHP-Scripts sehen können. Die Ausgabe dieses Scripts wird im Inhaltsbereich angezeigt. PHP-Anwendungen wie Counter-Scripte oder Dinge, die in der Seitenleiste angezeigt werden sollen, sind daher für dieses Plugin nicht geeignet und müssen stattdessen eher über die oben genannten Alternativen eingebunden werden. Scripte wie eine Bildergalerie, eine Google-Maps-Unterseite oder alle anderen Anwendungen mit eigener Ausgabe, die im Serendipity-Layout erscheinen sollen, sind Anwendungsziel dieses Plugins.

Wenn Sie das Plugin installiert haben, können Sie in der Konfiguration folgende Eigenschaften festlegen:

Permalink
: Globaler Permalink, um die Ausgabe des Plugins im Frontend sehen zu können. Details siehe Seite 238.

URL-Titel der Seite
: Globale Variable, die einen alternativen Permalink zur Ausgabe des Plugins im Frontend bereitstellt. Details siehe Seite 238.

Einzubindende Datei
: Dieses wichtige Feld darf nur von Administratoren ausgefüllt werden. Sie müssen hier den vollständigen Dateipfad zu der externen PHP-Anwendung eintragen, wie beispielsweise `/var/www/example.com/php_gallery/include.php`. An dieser Stelle dürfen Sie keine URL eintragen, da PHP-Includes aus Sicherheitsgründen nur aus Dateien auf demselben Server ausgeführt werden sollten. Wenn Sie den Inhalt einer fremden URL einbinden wollen, sollten Sie sich lieber das Plugin *WrapURL* (siehe Seite 302) ansehen.

  Es ist sehr wichtig, dass die fremde PHP-Anwendung sich an folgende Richtlinien hält: Sie sollte keine HTML Header und Footer setzen (da dies Serendipity bereits erledigt), sollte bestehende Datenbankverbindungen nicht kappen, und wenn sie spezielle Namensräume verwendet, muss sie darauf achten, die Variablen gegebenenfalls per `$GLOBALS[...]` anzusprechen. Das PHP-Include wird von Serendipity innerhalb einer Objektmethode ausgeführt und verfügt daher nicht über den üblichen globalen Namensraum.

Als Artikel formatieren
: Gibt an, ob die Ausgaben des Plugins wie ein Blog-Eintrag dargestellt werden sollen. Details siehe Seite 239.

Sobald Sie das Plugin entsprechend konfiguriert haben, können Sie die Ausgabe unter dem von Ihnen konfigurierten Permalink aufrufen und in Ihrem Blog über einen Seitenleisten-HTML-Klotz oder Ähnliches verlinken.

6.3.13 WrapURL serendipity_event_wrapurl

Ähnlich wie das vorausgehende Plugin kümmert sich das *WrapURL*-Plugin darum, einen fremden Inhalt in Ihrem Blog einzubinden.

Dabei nutzt dieses Plugin einen kleinen Trick namens *iframe*. Iframes werden von Browsern so genutzt, dass die Inhalte einer anderen Seite verschachtelt innerhalb einer Oberseite eingebunden werden. Der technische Vorteil dabei ist, dass man beliebige fremde (und auch eigene) Inhalte so auf einer Seite ohne *Layoutnarben* einbinden kann. Die Darstellung wirkt im Browser wie aus einem Guss, obwohl in Wirklichkeit zwei unabhängige Seiten geladen werden.

Der Vorteil besteht darin, dass man zwei unabhängige Anwendungen ohne viel technischen Aufwand miteinander verbinden kann. Die fremde Anwendung kann also eigene Kopf-/Fußzeilen und HTML-Konstrukte ausgeben und muss sich nicht um die Umgebung kümmern, in der die Anwendung eingebettet ist.

Das einzige layout-technische Problem des Plugins ist, dass der unsichtbare *Frame* der fremden Anwendung eine Mindesthöhe in Pixeln enthalten muss, da sich die Höhe des Frames in der Wirtsumgebung leider nicht dynamisch an die Seitenhöhe der Anwendung anpassen kann.

Folgende Optionen können für das Plugin eingestellt werden:

Headline
: Oberhalb der eingebundenen Zielseite kann Serendipity eine Überschrift im Layout des Blogs einbinden. Welche Überschrift benutzt wird, können Sie im Feld **Headline** einstellen.

Permalink
: Globaler Permalink, um die Ausgabe des Plugins im Frontend sehen zu können. Details siehe Seite 238.

URL shorthand name
: Globale Variable, die einen alternativen Permalink zur Ausgabe des Plugins im Frontend bereitstellt. Details siehe Seite 238.

The URL
: Im Feld **The URL** tragen Sie das Ziel des iframes ein. Diese URL entspricht der externen Webseite, die Sie einbinden wollen, und muss vollständig (also z. B. `http://www.example.com/g` eingetragen werden. Sie können auch vollständige URLs mit GET-Parametern einbinden: `http://www.example.com/users?id=15&output=html`

Height in Pixels
Ein iframe benötigt eine Höhenangabe innerhalb der Einbindung einer Webseite. Damit die fremde Webseite in Ihrem Blog-Layout möglichst vollständig eingebunden werden kann, müssen Sie eine maximale Höhe in Pixeln für den iframe eintragen. Der iframe wird dabei in eine Tabelle eingebunden, in der die Höhe der Tabellenzelle der von Ihnen festgelegten Pixelzahl entspricht. Der iframe selbst wird innerhalb dieser Zelle mit 100-prozentiger Höhe eingebunden, da man durch diese Technik die spätere Höhe relativ genau festlegen kann.

Append GET-Variables
Wenn Sie eine fremde Webanwendung über einen iframe einbinden, müssen Sie oft spezielle Parameter in der URL übergeben, beispielsweise für die Einbindung einer Bildergalerie: `http://www.example.com/galerie/index.php?album=Albumname`. Diese Variable müssten Sie jedoch eigentlich im Eingabefeld **The URL** festlegen und danach für jedes Album einer Galerie ein eigenes WrapURL-Plugin installieren. Das wäre ein gewaltiger Verwaltungsaufwand; besser wäre es, wenn Sie einfach die benötigte Variable `album` an das Wrap-URL Plugin übergeben, und dies leitet die Variable an den iframe weiter.

Dies erreichen Sie durch Aktivierung von **Append GET-Variables**. In diesem Fall können Sie Ihren WrapURL-Permalink per `http://www.example.com/serendipity/wpages/pagetitle.html?album=Albumname` bzw. `http://www.example.com/serendipity/index.php?serendipity[subpage]=alternativurl&album=Albumname` aufrufen und so die Variable `album` an den iframe weiterreichen.

Hide sidebars
Standardmäßig wird der Inhalt eines iframes innerhalb Ihres gewöhnlichen Blog-Templates eingebunden, daher bleiben auch die Seitenleisten-Plugins erhalten. Gerade dies macht Ihr Blog-Layout meistens aus, und so ist es durchaus gewünscht, das vollständige Layout zu sehen.

Wenn Sie die Option **Hide sidebars** aktivieren, können Sie die Anzeige der Seitenleisten-Plugins unterbinden.

6.3.14 Show links to services like Digg, Technorati, del.icio.us etc. related to your entry serendipity_event_findmore

Ein ganz wesentlicher Faktor von Blogs ist deren Popularität und Vernetzung untereinander. Durch die gegenseitige Verlinkung werden Blogs erst allgemein bekannt, und dies ist letztlich ausschlaggebend für die hohe Relevanz von Blog-Artikeln in Suchmaschinen wie Google.

Durch die Entstehung zahlloser Blogs wurde es im Laufe der Zeit immer schwieriger, einen Überblick über die Themen der *Blogosphäre* zu behalten. Welche Artikel werden gerade heiß

diskutiert, welche Themen sind in Blogs angesagt? Dieser Fragestellung nahmen sich mehrere Webservices an, die Blogs durchsuchen und mittels Automatismen analysieren. Aber auch durch Webservice-Communities wie `http://del.icio.us` wurde es schnell populär, beliebte Links zentral zu verwalten. Ein Leser kann nun einen interessanten Blog-Artikel als öffentliches Lesezeichen speichern, auf das die Mitglieder der Community Zugriff haben. Wenn diese Mitglieder einen Link auch informativ finden, übernehmen sie ihn ebenfalls als eigenes Lesezeichen, markieren oder bewerten ihn. So entsteht schnell eine qualitative und quantitative Bewertung aktueller Blog-Artikel. Übersichtsseiten der Social-Web-Communities stellen die populärsten Blog-Artikel übersichtlich dar und erhöhen so den Andrang auf beliebte Blogs noch weiter.

Die Teilnahme an so einer Community ist vergleichsweise einfach: einen Account erstellen, Browser-Plugins installieren oder über die jeweilige Webseite selbst die Lesezeichen mit anderen Benutzern teilen. Weitaus schwerer ist es, in der Community seine eigenen Blog-Artikel populär zu machen.

Damit das gelingt, muss man mehrere Dinge tun. Das Allerwichtigste ist, dass Sie für die Allgemeinheit oder ein Fachpublikum interessante Blog-Artikel verfassen, die Ihren Besuchern einen Anreiz bieten, einen Artikel überhaupt zu „bewerten“ oder zu „empfehlen“. Sobald Sie willige Besucher auf sich aufmerksam gemacht haben, müssen Sie diesen nun eine Möglichkeit bieten, Ihren Blog-Artikel in der jeweiligen Web-Community zu bewerten.

An dieser Stelle kommt das Plugin *Show links to services like Digg etc.* zum Tragen. Dieses bietet eine Einbindung fast aller verfügbaren Link-Services, so dass Sie unterhalb jedes Blog-Artikels einen kleinen Button des jeweiligen Services finden, der Ihre Besucher dorthin führt und Ihre Seite in der Community verbreitet. Grundsätzlich benutzen die meisten aktiven Besucher zwar ihre eigenen Browser-Tools zu diesem Zweck, aber durch Ihre Einbindung zu diesen Services motivieren Sie einen Besucher stärker, sich in der Community zu beteiligen.

Da es mittlerweile eine fast unüberschaubare Anzahl an Link-Communities gibt, bietet das Plugin für jeden Dienst einen einzelnen Button an. Folgende Services sind dabei standardmäßig enthalten: Del.icio.us, Digg, Bloglines, Technorati, Fark, MyWeb2, Furl, Reddit, BLinklist, Spurl, Newsvine, Simpy, Blogmarks, Wists, ma.gnolia, Mister Wong, addthis.

Es ist empfehlenswert, dass Sie in dieser großen Liste eine eigene Einschränkung bzw. Gewichtung vornehmen, um so nur die Link-Communities einzubinden, die für Ihre Zwecke Sinn machen. Bei deutschen Blogs sind beispielsweise Mister Wong, Digg und Del.icio.us sehr populär. Überwältigen Sie Ihre Besucher daher nicht unbedingt mit einer riesigen Liste von Verlinkungsmöglichkeiten, sondern überzeugen Sie durch eine sinnvolle Einschränkung. Schauen Sie sich ruhig jeden Link-Service an, damit Sie entscheiden können, welche Ihnen am meisten zusagen.

Sobald Sie das Plugin installiert haben, finden Sie die erwähnte Leiste an Link-Services unterhalb jedes Beitrags. Ein Klick auf einen Button ruft dabei die jeweilige Community auf und fügt dort einen Link zu Ihrem ursprünglichen Blog-Artikel mitsamt dem Titel des Beitrags hinzu.

Die eingebundenen Linkservices werden in einem HTML-Container mit der CSS-Klasse

`.serendipity_findmore` ausgegeben. Der Inhalt der Leiste richtet sich nach der Template-Datei `plugin_findmore.tpl`, die sich innerhalb des Plugin-Verzeichnisses `plugins/serendipity_event_findmore/` befindet.

In dieser Template-Datei sind sämtliche Link-Services übersichtlich untereinander aufgeführt. Jeder Linkservice lässt sich meist gleichartig einbinden und unterscheidet sich nur in der Benennung von Variablen. Diese Variablen werden mithilfe der Smarty-Platzhalter {`$entrydata.*`} später durch die Inhalte Ihres jeweiligen Blog-Artikels ersetzt. Folgende Smarty-Variablen stehen zur Verfügung:

{`$entrydata.url`}
: Enthält die vollständige URL des Blog-Artikels. Wenn eine URL an einen Linkservice übermittelt wird, müssen etwaige Sonderzeichen in der URL über einen Zusatz wie {`$entrydata.title|escape:url`} eingebunden werden (siehe Kapitel 9.3 ab Seite 480).

{`$entrydata.title`}
: Enthält den Titel des Blog-Artikels. Diese Variable wird im Template meist über den Befehl {`$entrydata.title|escape`} eingebunden. Das zusätzliche `|escape` sorgt dafür, dass Sonderzeichen im Artikeltitel durch HTML-Sonderzeichen ersetzt werden. `|escape:url` sorgt im Unterschied dazu dafür, dass Sonderzeichen durch URL-Sonderzeichen ersetzt werden.

{`$entrydata.path`}
: Enthält den HTTP-Pfad zu dem Bildverzeichnis des Plugins. Dort liegen für fast alle Link-Communities die jeweiligen Logos, damit sie in der Linkleiste grafisch eingebunden werden können.

Die Datei `plugin_findmore.tpl` können Sie nach Belieben anpassen, um beispielsweise überflüssige Services zu entfernen oder auch neue hinzuzufügen. Eine geänderte Template-Datei sollten Sie dabei möglichst in das Verzeichnis Ihres gewählten Templates kopieren, damit Sie bei einem Update des Plugins Ihre Anpassungen nicht verlieren.

Über die Konfiguration des Plugins können Sie komfortabel einstellen, welche Links zu welchen Services Sie darstellen möchten, falls das Bearbeiten der Template-Option für Sie zu umständlich ist.

Weiterhin bietet das Plugin folgende Konfigurationsoptionen:

Relative path to Findmore images
: In diesem Eingabefeld müssen Sie den vollen HTTP-Pfad eintragen, aus dem das Plugin die Logo-Grafiken für den jeweiligen Linkservice über die Template-Datei `plugin_findmore.tpl` bezieht. Standardmäßig sollte dieser Pfad bereits korrekt auf Ihr Plugin-Verzeichnis zeigen.

Display DiggCount badge
: Der Link-Dienst `Digg` bietet ein kleines JavaScript an, mit dem Ihre Besucher direkt

sehen können, wie oft die Links in Ihrem Blog-Artikel von Teilnehmern der Digg-Community positiv bewertet wurden. Diese kleine Anzeige mit einer Zahl für den jeweils bewerteten Link nennt sich *DiggCount badge*.

Das Plugin analysiert bei aktivierter Option *DiggCount badge* sämtliche Links zu Beiträgen auf `digg.com`, die Sie in Ihrem Artikel platziert haben. Sollten Sie die *Diggs* zu Ihrem eigenen Artikel anzeigen wollen, müssen Sie manuell Ihren eigenen Blog-Artikel im Artikeltext verlinken, damit das Plugin diesen in die Ausgabe einbeziehen kann. Für jeden Link, der in Ihrem Blog-Artikel zu `digg.com` zeigt, stellt das Plugin eine eigenständige Zählergrafik dar.

Wenn Sie die Option aktivieren, werden die Digg-Grafiken und das JavaScript in einem eigenständigen HTML-Container mit der CSS-Klasse `.serendipity_diggcount` angezeigt.

DiggCount placement
Wenn Sie *DiggCount badge* aktiviert haben, bestimmen Sie mit der Option **DiggCount placement**, wo dieser Zähler angezeigt werden soll. **Before entry** bindet den Zähler direkt am Anfang Ihres Blog-Artikels ein, **After entry** am Ende des Artikels und **After findmore links** nach der Linkleiste des Plugins.

Template-Profis haben möglicherweise bereits bemerkt, dass die Linkleiste der `plugin_findmore.tpl`-Template-Datei recht simpel gestrickt ist und lediglich die URL und den Titel eines Artikels enthalten muss. Diese Informationen können Sie daher auch ganz einfach in die Template-Datei `entries.tpl` einfügen und benötigen dieses Plugin höchstens noch für die Einbindung der *DiggCount badge*.

Der Vorteil der Einbindung in die `entries.tpl`-Datei besteht in einer höheren Ausführungsgeschwindigkeit der Templates, da nicht noch eine eigenständige Datei analysiert werden muss. Der Einfachkeit halber können Sie den Inhalt der `plugin_findmore.tpl`-Datei per Kopieren & Einfügen in die `entries.tpl`-Datei übernehmen und den HTML-Container `<div class="serendipity_findmore">` beliebig innerhalb der Smarty-Foreach-Schleife platzieren, in der ein Eintrag dargestellt wird. Sie müssen danach nur noch die Variable `$entrydata.url` mit `$entry.rdf_ident` ersetzen, `$entrydata.title` mit `$entry.title` und anstelle von `$entry.path` einen gültigen HTTP-Pfad zu den Linkservice-Bildern einsetzen.

6.3.15 Suchmaschinen-Sitemap Generator serendipity_event_google_sitemap

Wenn Sie den Inhalt Ihres Blogs über eine Suchmaschine auffinden wollen, muss die jeweilige Suchmaschine Ihr Blog *indizieren*. Bei diesem Vorgang ruft ein Suchroboter Ihr Blog auf, als wäre er ein normaler Besucher. Dann analysiert er Ihre Webseite, sammelt sämtliche enthaltenen Links und folgt diesen dann. Diesen Vorgang vergleicht man oft mit *crawling* oder

spidering, da ein Suchroboter mit diesem Mechanismus wie ein Insekt über die vollständige Webseite *kriecht*.

Dieser Vorgang kann mitunter sehr zeitaufwändig sein, da durch die Baumstruktur einer Webseite möglicherweise besonders viele Seiten aufgerufen werden müssen. Da der Suchroboter die Struktur einer Seite nicht wie ein Mensch erfassen kann, muss er stur sämtliche verfügbaren Unterseiten aufrufen und erkennt möglicherweise nicht, dass einige Seiten denselben Inhalt besitzen.

Gerade bei Weblogs gibt es zahlreiche Möglichkeiten, Blog-Artikel anzusehen: die globale Artikelübersicht, eine Übersicht pro Woche, pro Monat, pro Jahr, pro Kategorie, pro Autor und viele weitere. So entstehen für einige wenige Blog-Artikel bereits mehrere hundert Seitenaufrufe der Suchroboter.

Dies ist nicht unbedingt ein grundlegender Nachteil: Da Blog-Artikel so verschieden dargestellt werden, interpretiert eine Suchmaschine wie Google dies oft als eigenständige Seite und gewichtet daher die vorkommenden Wörter stärker, als wenn Sie nur eine einzelne Artikelübersichtsseite besäßen.

Der Nachteil entsteht erst dann, wenn der Suchroboter sehr oft Ihre Webseiten besucht und dadurch viel Traffic erzeugt – oder wenn er nicht oft genug die Seiten besucht und daher nicht der Aktualität Ihrer Beiträge entsprechen kann.

Für diese Fälle wurde von den Suchmaschinenbetreibern ein eigener Standard ins Leben gerufen. Dieser setzt am Prinzip einer *Sitemap* an. Eine Sitemap stellt sozusagen die Struktur Ihrer Webseite mit allen Unterseiten dar, wie eine Verzeichnisübersicht Ihrer Festplatte im vollständig ausgeklappten Zustand. Zusätzlich interpretieren Suchroboter auch noch die Datei `robots.txt` im Stammverzeichnis Ihrer Webseite. In dieser Datei können bereits Seiten von der Indizierung explizit ausgeschlossen und einige andere Optionen eingestellt werden. Weitere Informationen hierzu finden Sie auf `http://de.selfhtml.org/diverses/robots.htm`.

Eine Sitemap-Datei muss dabei im XML-Format[20] abgespeichert werden und alle gültigen URLs enthalten. Jede URL kann mehrere Eigenschaften enthalten: das Datum der letzten Änderung, eine Aktualisierungspriorität (Gewichtung) und eine Angabe, wie häufig sich das Dokument schätzungsweise ändert.

Damit Sie diese Sitemap-Datei nicht manuell erstellen müssen, wurde das Plugin *Suchmaschinen-Sitemap* entwickelt. Dieses erstellt die notwendige XML-Datei anhand Ihrer Datenbank automatisch.

Sobald das Plugin installiert ist, wird es sich bei der Veröffentlichung jedes neuen Artikels um die Aktualisierung der Sitemap-Datei kümmern. Dabei setzt es automatisch die korrekten Werte für die letzte Änderung an einem Artikel sowie die Gewichtung und Aktualisierungsfrequenz. Es enthält die URLs für:

- Startseite des Blogs, Gewichtung 0.6
- Archivübersicht des Blogs zzgl. Folgeseiten, Gewichtung 0.5

[20] Die vollständige Spezifikation des Formates können Sie auf `http://www.sitemaps.org/` nachschlagen.

- Artikel-Detailseiten sowie eventueller Alternativ-Permalinks, Gewichtung 0.7–0.8
- Kategorie-Übersichtsseiten, Gewichtung 0.4
- Autor-Übersichtsseiten, Gewichtung 0.2
- Archivübersicht nach Monat (Gewichtung 0.3), Monatsübersicht (Gewichtung 0.1), Archivübersicht nach Monat pro Kategorie (Gewichtung 0.1)
- RSS-Feed pro Kategorie, Gewichtung 0.0[21]
- Globaler RSS-Feed, Gewichtung 0.0
- etwaige statische Seiten, Gewichtung 0.7

Die XML-Datei `sitemap.xml` speichert das Plugin danach im Stammverzeichnis des Blogs, wo es von den Suchmaschinen gefunden werden kann. Damit die Datei erfolgreich geschrieben werden kann, müssen entweder Schreibrechte zu dieser Datei bestehen, oder das Stammverzeichnis muss für den PHP-Anwender beschreibbar sein (Verzeichnisrechte 0777, siehe Kapitel 2.2.2 ab Seite 53).

Das Sitemap-Format wird neben Google auch durch die Suchmaschinen-Roboter von MSN, Yahoo und Ask unterstützt.

In den Konfigurationsoptionen des Plugins können Sie einige Einstellungen vornehmen:

Updates melden
: Wenn Sie die Option **Updates melden** aktivieren, kann das Sitemap-Plugin eine Liste von Suchmaschinenbetreibern kontaktieren, um diesen mitzuteilen, dass sich Ihre Sitemap geändert hat. Daraufhin kann der jeweilige Suchroboter Ihre Seite neu indizieren.

  Das Plugin kann nur dann Updates melden (*pingen*), wenn Ihr Server nicht durch eine Firewall blockiert wird und ausgehende HTTP-Verbindungen zu anderen Webservern erlaubt.

URL-Liste für Pings
: Wenn Sie die Option **Updates melden** aktiviert haben, können Sie in diesem Feld eine Liste der URLs von Suchmaschinenbetreibern eintragen, an die das Plugin sich bei Aktualisierungen automatisch wenden soll. Voreingestellt sind im Plugin die URLs von Google und Ask.

  Welche URL Sie beim jeweiligen Suchmaschinenbetreiber benutzen können, erfahren Sie auf dessen Webseite. Mehrere Betreiber-URLs können Sie mit dem Semikolon `;` voneinander trennen.

[21] RSS-Feeds sollen nicht als Suchergebnisse bei den Suchmaschinen angezeigt, sondern nur als zusätzliche Alternativdarstellung eingebunden werden. Daher erhalten diese eine Gewichtung von 0.0.

Die sitemap.xml mit gzip packen

Da Ihre `sitemap.xml`-Datei bei vielen Blog-Einträgen recht groß werden kann, ermöglicht das Sitemap-Plugin, diese Datei mit ZIP-Kompression zu speichern. Diese Kompression funktioniert nur, wenn Ihre PHP-Version auf dem Webserver das ZIP-Modul eingebunden hat. Ist diese Option aktiviert, wird die Sitemap-Datei als `sitemap.xml.gz` gespeichert.

Wenn Sie Probleme bei der Erstellung einer Sitemap haben, empfiehlt es sich, diese Option zu deaktivieren. Dann können Sie mit einem beliebigen Editor die Datei `sitemap.xml` öffnen, um nach etwaigen Fehlermeldungen zu suchen.

URL-Typen

In dieser Liste können Sie festlegen, welche URLs in die Sitemap eingefügt werden sollen: **Feeds, Kategorien, Autoren, Permalinks, Archiv, Statische Seiten, Tag-Seiten**. Standardmäßig sind alle URL-Typen ausgewählt.

Feeds werden jedoch nur mit einer Gewichtung von 0 in die Sitemap aufgenommen, da die Inhalte eines RSS-Feeds derzeit nirgends als Suchergebnisse herangezogen werden und nur als alternative Informationsquelle zu Aktualisierungen dienen.

Je nach Suchmaschinenbetreiber müssen Sie Ihr Blog möglicherweise erst für die Benutzung von Sitemaps freischalten. Bei Google geschieht dies etwa über die URL `http://www.google.com/webmasters/tool`

Einige Suchmaschinen werten zudem die Datei `robots.txt` aus und suchen dort nach der Angabe Ihrer Sitemap. Dazu müssen Sie die Zeile `Sitemap: http://www.example.com/serendipity/sitemap.` in `robots.txt` aufnehmen.

6.3.16 Kategorie als Startseite serendipity_event_startcat

Die Einstiegsseite des Frontends Ihres Blogs zeigt üblicherweise die aktuellsten Artikel aller verfügbaren Kategorien an, sofern diese nicht durch Leserechte zugriffsbeschränkt sind.

Während dies bei üblich geführten Blogs genau das ist, was Ihre Leser erwarten, kann es zu Verwirrungen kommen, wenn Sie Serendipity eher als Content-Management-Software einsetzen.

In solchen Fällen ist es oft üblicher, auf der Startseite nur Artikel einer bestimmten Kategorie anzuzeigen. Wenn Ihre Besucher andere Artikel/Inhalte sehen wollen, müssen sie vorher aktiv die gewünschte Kategorie auswählen.

Genau dieses Verhalten einer besonderen *Startkategorie* können Sie mit dem Plugin *Kategorie als Startseite* erreichen. Das Plugin macht letztlich etwas ganz Einfaches: Es beschränkt die Ansicht der Startseite auf eine konfigurierte Kategorie. Dies entspricht dann der Ansicht, die ein Besucher erhält, wenn er im Frontend die konfigurierte Kategorie ausgewählt hätte.

Abseits von dieser Startkategorie bietet das Plugin jedoch auch noch weitere Möglichkeiten, die Sie über seine Konfigurationsoptionen festlegen können:

Ursprungskategorie
: In diesem Auswahlfeld können Sie festlegen, welche Kategorie Ihres Blogs für die Artikelübersicht herangezogen wird.

Versteckte Kategorie
: Anstatt in der Artikelübersicht nur die Artikel einer gewissen Kategorie anzuzeigen, können Sie die Logik auch umdrehen und gewisse Kategorien von der Artikelübersicht *ausschließen*. Darin enthaltene Artikel kann ein Besucher erst dann lesen, wenn er explizit die Kategorie-Übersicht der ausgeschlossenen Kategorie aufruft. Im Vergleich zum Fixieren einer Kategorie hat dies den Vorteil, dass Serendipity nicht stets eine Kategorie vorgibt.

Mehrere Ursprungskategorien
: Anstatt nur eine einzelne Kategorie als Startkategorie festzulegen, können Sie auch mehrere Kategorie-IDs mit einem `;` getrennt voneinander eintragen. Die Kategorie-IDs können Sie der Kategorieverwaltung (siehe Seite 124) entnehmen.

Mehrere versteckte Kategorien
: Analog zum Fixieren mehrerer Kategorien für die Startseite können Sie auch mehr als eine Kategorie auf der Startseite ausblenden. Tragen Sie auch hier die IDs mit einem `;` voneinander getrennt ein.

Gewählte Kategorie besucherseitig merken
: Abgesehen vom Anzeigen/Verstecken von Kategorien bietet dieses Plugin auch eine unabhängige Funktionalität, mit der ein Besucher die zuletzt gewählte(n) Kategorie(n) dauerhaft speichern kann. Beim nächsten Besuch des Blogs wird für ihn dann automatisch die zuletzt besuchte Kategorieansicht wiederhergestellt.

  Diese Funktionalität kann für Ihre Besucher möglicherweise verwirrend sein. Wenn man ein Blog neu im Browser öffnet, erwartet man nicht unbedingt, dass die zuletzt angesehene Kategorie reaktiviert ist, sondern rechnet damit, dass man die Startseite des Blogs sieht, um neue Artikel auch von anderen Kategorien lesen zu können. Stellen Sie also sicher, dass dieses Feature im Kontext Ihres Blogs wirklich Sinn macht, und weisen Sie Ihre Besucher möglichst gezielt mittels eines HTML-Klotzes oder Ähnlichem auf das Feature hin.

Sobald das Plugin eine Einschränkung der Kategorie vornimmt, hat dies auch Auswirkung auf den Standard-RSS-Feed. Dieser enthält dieselben Artikel wie die Artikelübersicht, versteckt also gegebenenfalls auch Artikel der nicht gewählten Kategorien. Um einen RSS-Feed aller gewählten Kategorien anzuzeigen, müssen Sie diesen über `http://www.example.com/serendipity/rss.php?serendipity[category]=all` aufrufen. Damit Ihre Besucher darüber auch Bescheid wissen, müssen Sie gegebenenfalls gesondert darauf hinweisen (in einem HTML-Klotz oder durch Anpassung des *Blog abonnieren*-Plugins).

Die Wahl einer Standardkategorie hat auch Auswirkungen auf die Anzeige des Kalender-Plugins in der Seitenleiste und auch möglicherweise auf das Kategorien-Seitenleisten-Plugin.

Abhängig von ihrer Konfiguration beziehen beide Plugins die gewählte Kategorie mit ein und zeigen nicht wie gewohnt die Daten aller Kategorien.

6.3.17 Eigenschaften/Templates von Kategorien serendipity_event_categorytemplates

Mithilfe des Plugins *Eigenschaften/Templates von Kategorien* können Sie jeder Kategorie des Blogs gewisse Eigenschaften und Konfigurationsoptionen zuweisen sowie individuelle Templates auswählen.

Diese Eigenschaften werden aktiv, wenn ein Besucher im Frontend die Artikelübersicht einer Kategorie aufruft. Auch wenn er die Detailansicht eines Blog-Artikels öffnet, der einer einzelnen Kategorie zugewiesen ist, werden die Eigenschaften aktiv. Wenn einem Artikel mehrere Kategorien zugewiesen sind, von denen aber nur eine Kategorie besondere Eigenschaften besitzt, wird diese automatisch für die Einzel-Detailansicht angewendet.

Am häufigsten wird das Plugin dazu eingesetzt, abhängig von der gewählten Kategorie unterschiedliche Templates für das Blog einzusetzen. So kann man mehrere Kategorien des Blogs grafisch klar voneinander unterscheiden. Durch diese Trennung kann man auch thematische Unter-Blogs einrichten, die für den Besucher voneinander unabhängig erscheinen, aber allesamt dieselbe Datenbank besitzen. Gerade für den Blog-Betreiber ist es weitaus komfortabler, mit einem einzigen Backend mehrere voneinander getrennt erscheinende Frontends zu verwalten.

Falls ein gewünschtes Template den Einsatz von Template-Optionen unterstützt (siehe Seite 499), können diese Optionen für das Template individuell für jede Kategorie gesetzt werden, in der das Template zum Einsatz kommt. Sie müssen daher das Template nicht in mehrere Unterverzeichnisse kopieren, um ein eigenes Template pro Kategorie zuzuweisen, sondern können die Template-Optionen direkt von der Oberfläche aus mit den Eigenschaften einer Kategorie aufrufen.

Mit zusätzlicher Hilfe der individuellen Lese- und Schreibrechte zu Kategorien kann man so sogar recht flexibel Mehr-Autoren-Blogs einrichten, die allesamt mit derselben Verwaltungsoberfläche administriert und bearbeitet werden können.

Das Plugin bietet folgende grundsätzliche Konfigurationsoptionen:

Password protection

Über diese Option können Sie pro Kategorie entscheiden, ob diese mit einem Passwort-Zugriffsschutz versehen werden soll.

Wenn ein Besucher später die Übersichtsseite einer geschützten Kategorie aufrufen will, muss er zuerst ein Passwort eintragen. Erst nachdem der Besucher dies erledigt hat, sieht er die Einträge einer derart geschützten Kategorie auch auf der globalen Übersichtsseite.

Die Aktivierung dieser Option verlangsamt die Datenbankabfragen für Übersichtsseiten etwas, daher sollten Sie sie nur aktivieren, wenn Sie passwortgeschützte Kategorien

dringend benötigen.

Standard: Sortierung

Standardmäßig sortiert Serendipity die Artikelansichten stets chronologisch. Die aktuellsten Artikel erscheinen zuerst, die ältesten Artikel am Ende. Ein Besucher sieht so die aktuellsten Artikel immer direkt auf den ersten Blick.

Bei einigen Blogs wünschen sich die Besitzer jedoch, dass ihr Blog in der richtigen Reihenfolge, von den ältesten zu den neuesten Artikeln gelesen wird. Gerade bei fiktiven Blogs, die eine Geschichte erzählen, macht dies Sinn.

Über das Eingabefeld **Standard: Sortierung** können Sie die Artikelsortierung festlegen, die für die generelle Artikelübersicht gilt. Beachten Sie, dass Sie auch eine Einstellung der Sortierung pro Kategorie vornehmen können, indem Sie die jeweiligen Eigenschaften einer Kategorie (über den Menüpunkt **Einträge** → **Kategorien**) bearbeiten.

In dem Eingabefeld müssen Sie eine SQL-Syntax eingeben. Als Sortierungskriterium kann man sämtliche Datenbank-Spaltennamen der Tabelle `serendipity_entries` heranziehen. Die eigentliche Reihenfolge gibt man über den Befehl `ASC` (ascending: aufsteigend) oder `DESC` (descending: absteigend) an. In den meisten Fällen müssen Sie daher an dieser Stelle nur `timestamp DESC` eingeben (für die übliche Sortierungsreihenfolge mit aktuellsten Artikeln am Anfang) oder alternativ `timestamp ASC` (älteste Artikel am Anfang). Weitere sinnvolle Möglichkeiten wären eine Sortierung nach Titel (`title DESC`) oder nach der Anzahl der Kommentare zu einem Artikel (`comments DESC`).

Abgesehen von diesen beiden zentralen Optionen bietet das Plugin seine Kernfunktionalität im Menüpunkt **Einträge** → **Kategorien** an. Bearbeiten Sie dort eine Kategorie, und Sie werden zusätzliche Felder dort sehen:

Artikel von Unterkategorien verstecken

Standardmäßig zeigt Serendipity bei der Artikelübersicht einer gewählten Kategorie im Frontend auch alle Einträge an, die in den möglichen Unterkategorien der gewählten Kategorie vorhanden sind. Gesetzt den Fall, Sie haben eine Hauptkategorie „Fernsehen“ eingerichtet und darin die Unterkategorien „Drama“, „Action“, „Comedy“ und „Lustspiele“. Wenn Sie im Frontend nun auf die Kategorie „Fernsehen“ klicken, sehen Sie auch alle Beiträge in den Unterkategorien. Erst wenn Sie auf die letzte Unterkategorie wie „Action“ klicken, sehen Sie keine Einträge der anderen Kategorien mehr.

In den meisten Fällen ist diese Variante bei der Artikeldarstellung für Besucher recht angenehm, und Sie müssen Artikel nicht zwangsläufig mehreren Kategorien zuweisen.

Wenn Ihre Kategoriestruktur jedoch nicht hierarchisch aufgebaut ist, könnte die Standardübersicht unpassend sein. Daher bietet das Plugin die Möglichkeit (Option **Artikel von Unterkategorien verstecken**), keine Artikel von Unterkategorien mehr anzuzeigen.

Die Einstellung sollte nur auf Kategorien angewendet werden, die Unterkategorien besitzen. Bei fehlenden Unterkategorien hat diese Einstellung natürlich keine Wirkung.

Wählen Sie das Template für das Blog

Bei dieser Option stehen zwei Eingabemöglichkeiten zur Verfügung. Entweder Sie füllen das Eingabefeld aus und tragen dort einen Template-Namen ein, oder Sie wählen aus dem Ausklappfeld den entsprechenden Template-Namen. Bei einer Auswahl im Ausklappfeld wird Ihre Eingabe im Textfeld ignoriert.

In dem Ausklappfeld stehen Ihnen sämtliche Templates zur Verfügung, die in Ihrem Blog im Unterverzeichnis `templates` eingerichtet wurden.

In das Eingabefeld tragen Sie nicht den Namen des Templates (wie beispielsweise „Serendipity 3.0“) ein, sondern den Verzeichnisnamen (`carl_contest`). Diese Eingabemöglichkeit bietet die Flexibilität, dass Sie in einem Haupt-Template-Verzeichnis auch Unterverzeichnisse mit eigenständigen Templates erstellen können, um so Ihre Verzeichnisstruktur aufgeräumter zu strukturieren. In einem solchen Fall können Sie auf Unterverzeichnisse mittels `mein_template/unter_template1` zugreifen. Tragen Sie nur Verzeichnisnamen ab der Verzeichnisebene `templates` ein.

Das ausgewählte Template wird im Frontend anstelle des Standard-Templates angezeigt, sobald der Besucher die Artikelübersicht (oder einen detaillierten Artikel) der Kategorie aufruft. Mit dieser Gestaltungsmöglichkeit können Sie Kategorien gezielt formatieren und eine Art „Themen-Blog“ einrichten, bei dem Ihre Besucher das Gefühl haben, jeweils eigenständige Blogs aufzurufen. Für Sie hat dies den Vorteil, dass Sie das komplette Blog weiterhin über eine einzelne Administrationsoberfläche verwalten können.

Sollte das gewählte Template Ihnen zusätzliche Template-Optionen anbieten, wird ein Button zum Festlegen dieser Optionen unterhalb der Template-Auswahl eingebunden. Ein Klick darauf führt zu den Optionen des Templates, die Sie von der normalen **Styles verwalten**-Funktionalität her bereits kennen (siehe Seite 142).

Beachten Sie dabei, dass die so festgelegten Template-Optionen nur dann gelten, wenn Sie die jeweilige zugehörige Kategorie aufrufen – die Template-Optionen für die Übersichtsseite legen Sie wie gewohnt getrennt davon fest.

Wenn Sie ein Template mit Template-Optionen einsetzen und diese für jede Kategorie unterschiedlich konfigurieren möchten (beispielsweise über ein individuelles Kopfbild für jede Kategorie), können Sie dies festlegen, ohne für jede Kategorie ein eigenes Template anlegen/kopieren zu müssen.

Zukünftige Einträge zeigen

In der globalen Konfiguration von Serendipity können Sie einstellen, ob Einträge mit einem zukünftigen Datum im Frontend bereits vor Erreichen dieses Datums angezeigt werden sollen (siehe Seite 169).

Diese Einstellung können Sie auch pro Kategorie festlegen. Die Option `Standard` entspricht dabei der Einstellung, die Sie global vorgenommen haben. `Nein` legt fest, dass zukünftige Einträge nicht gezeigt werden, bei `Ja` werden sie angezeigt.

Die Einstellungen pro Kategorie sind nur dann gültig, wenn Sie im Frontend gezielt die Kategorie-Übersicht aufrufen.

Anzahl der Artikel für die Startseite der Kategorie
Standardmäßig richtet sich die Anzahl der dargestellten Artikel in einer Kategorie nach der Einstellung in der globalen Serendipity-Konfiguration. Sie können jedoch auch fallweise pro Kategorie eine andere Artikelanzahl einbinden. Wenn Ihre Artikel in bestimmten Kategorien länger sind als gewöhnlich, könnten Sie die Anzahl der Artikel zur besseren Übersicht daher gezielt reduzieren.

Tragen Sie in das vorgesehene Eingabefeld die gewünschte Anzahl ein.

Sortierung
In den Konfigurationsoptionen des Plugins können Sie bereits die globale Sortierungsreihenfolge der Artikelübersichten einstellen. Pro Kategorie können Sie auch eine individuelle Sortierungsreihenfolge vorgeben. Welche Möglichkeiten Sie zur Eingabe nutzen können, wird bei den Konfigurationseinstellungen dieses Plugins auf Seite 312 beschrieben.

Die Konfigurationsoption **Globally set entry's category** sorgt dafür, dass beim Aufruf der Detailansicht eines Artikels, der einer einzelnen Kategorie zugeordnet ist, diese Kategorie als *aktuelle Kategorie* für die gesamte Blog-Darstellung verwendet wird. Diese Option ist vor allem dann hilfreich, wenn Sie Seitenleisten-Plugins so konfiguriert haben, dass sie ihre Ausgabe von der aktuellen Kategorie abhängig machen sollen. Da Detailseiten eines Artikels standardmäßig niemals eine Kategorie setzen, wären in so einem Fall bei deaktivierter Option die Seitenleisten ebenfalls ohne feste Kategorieabhängigkeit. Erst durch Aktivierung der Option sorgt das Plugin dafür, die globale Kategorievariable (`$serendipity['GET']['category']`) zu füllen.

Datenbanktabelle

Das Plugin speichert die Zuordnungen individueller Templates pro Kategorie in einer eigenen Datenbanktabelle `serendipity_categorytemplates`:

`categoryid`
enthält die Kategorie-ID, bei der ein eigenes Template verwendet werden soll.

`template`
enthält den Verzeichnisnamen des Templates, der für diese Kategorie eingesetzt werden soll.

`fetchlimit`
gibt an, wie viele Einträge für Artikel dieser Kategorie auf der Übersichtsseite angezeigt werden sollen.

`futureentries`
: gibt an, ob bei Darstellung der jeweiligen Kategorie Einträge gezeigt werden sollen, deren Datum in der Zukunft liegt.

`lang`
: legt die Sprache für die Darstellung dieser Kategorie fest (wird derzeit noch nicht ausgewertet).

`pass`
: legt ein Passwort fest, um die Übersichtsseite dieser Kategorie aufrufen zu können.

`sort_order`
: legt die Sortierungsreihenfolge der Artikel für die Übersichtsseite der jeweiligen Kategorie fest.

6.3.18 Gästebuch serendipity_event_guestbook

Mit dem *Gästebuch*-Plugin können Sie in Ihrem Blog einen Bereich einbinden, in dem Besucher sich in ein Gästebuch eintragen können. Die Kommentarfunktionen Serendipitys beziehen sich immer auf die jeweiligen Artikel, daher sind generelle Kommentare Ihrer Besucher besser in einem Gästebuch aufgehoben.

Es gibt zahlreiche PHP-Scripts zur Einbindung von Gästebüchern und auch zahlreiche kostenlose Webangebote, bei denen Sie Gästebücher eröffnen können. Das Gästebuch-Plugin ist möglichst simpel gehalten und hauptsächlich für eine nahtlose Einbindung in Serendipity vorgesehen.

Das Gästebuch können Sie im Frontend über einen frei von Ihnen vergebenen Permalink aufrufen. Diesen Link binden Sie am besten entweder in einen HTML-Klotz in der Seitenleiste ein oder fügen ihn fest in eine Navigation Ihres Templates ein (siehe Seite 489 und 469).

Ein Besucher sieht im Gästebuch die Einträge der vorherigen Gäste und kann dort über ein Formular seinen eigenen Eintrag hinterlassen. Wenn Sie im Blog eingeloggt sind, können Sie als Administrator zudem bestehende Einträge im Gästebuch direkt löschen.

Für das Gästebuch-Plugin stehende folgende Konfigurationsoptionen zur Verfügung:

Permalink
: Globaler Permalink, um die Ausgabe des Plugins im Frontend sehen zu können. Details siehe Seite 238.

Seitentitel
: Globale Variable, die einen alternativen Permalink zur Ausgabe des Plugins im Frontend bereitstellt. Details siehe Seite 238.

Überschrift
Tragen Sie in diesem Feld die **Überschrift** ein, die Sie später im Gästebuch sehen möchten.

Einführungstext
Um Ihren Besuchern eine Einführung in Ihr Gästebuch zu geben und sie zu einem Eintrag zu motivieren, können Sie in diesem Feld einen beliebigen **Einführungstext** eingeben. Dieser darf auch beliebiges HTML enthalten und wird später oberhalb der Gästebucheinträge angezeigt.

Gästebuch-Formular
Mit dieser Option legen Sie fest, ob das Formular für neue Einträge oberhalb oder unterhalb der vorhandenen Gästebucheinträge eingebunden werden soll.

Einträge pro Seite
Tragen Sie im Feld **Einträge pro Seite** eine Zahl ein, die festlegt, wie viele Gästebucheinträge auf einer einzelnen Seite angezeigt werden sollen. Weitere Einträge können dann durch eine Blätterfunktion aufgerufen werden.

Anzahl der Zeichen pro Zeile
Da Gästebucheinträge möglicherweise auch überlange Wörter enthalten, können Sie mit dieser Option festlegen, nach wie vielen Zeichen ein automatischer Zeilenumbruch erfolgen soll.

Als Artikel formatieren
Wenn Sie diese Option aktivieren, werden etwaige Textformatierungs-Plugins (Smilies, BBCode etc.) ausgeführt. Damit erscheint ein Gästebucheintrag dann mit denselben Formatierungsmöglichkeiten wie ein normaler Blog-Kommentar.

Send e-mail to admin
Wenn Sie bei jedem neuen Gästebucheintrag einen E-Mail-Hinweis erhalten wollen, können Sie diese Option aktivieren.

E-Mail-Adresse des Admin
Bei aktivierter Option **Send e-mail to admin** müssen Sie in dem Eingabefeld hier die gewünschte E-Mail-Adresse eintragen.

E-Mail-Adresse des Users, Homepage des Users
Das Gästebuchformular kann von einem Besucher mehrere Eingabefelder optional abfragen. Standardmäßig muss für einen Gästebucheintrag nur ein Text und ein Name angegeben werden. Über die Aktivierung der Optionen **E-Mail-Adresse des Users** und **Homepage des Users** können Sie zusätzlich auch weitere Felder abfragen.

Captcha-Schutz aktivieren
Um Spam im Gästebuch zu verhindern, kann für das Gästebuch ebenfalls der Captcha-Schutz (siehe Seite 239) aktiviert werden.

Datumsformat
: Tragen Sie im Feld **Datumsformat** ein, wie das Datum der Gästebucheinträge formatiert werden soll. Zur Verfügung stehen die Platzhalter, die in der PHP-Dokumentation[22] beschrieben werden.

Gästebuch-Hilfe
: Am Ende der Konfigurationsoptionen befindet sich ein kleiner Informationstext mit einem Link zu der README-Datei des Plugins. In dieser Datei ist eine kleine Anleitung zum Plugin enthalten, die einige technische Informationen enthält.

Nachdem Sie das Plugin installiert haben, müssen Sie noch einen Link zum Gästebuch-Plugin (der von Ihnen festgelegte **Permalink**) in Ihrem Blog hinzufügen. Den Link können Sie am besten über einen HTML-Klotz in der Seitenleiste einfügen oder auch durch den Einbau in eine mögliche Navigationsleiste Ihres Templates (siehe Kapitel 9.4.2, Seite 489, oder auch Kapitel 9.1 auf Seite 469).

Weiterhin verfügt das Plugin über ein mitgeliefertes Seitenleisten-Plugin. Damit können Sie die letzten Gästebucheinträge in der Seitenleiste darstellen. Die Konfigurationsoptionen sind dabei an das Plugin *Letzte Kommentare* (siehe Seite 217) angelehnt.

Die Ausgabe des Plugins erfolgt vollständig via Smarty-Templates. Die Datei `plugin_guestbook_form.tpl` enthält den Code zur Darstellung des Formulares für Eintragungen. Die Datei `plugin_guestbook_entries.tpl` wird zur Darstellung der einzelnen Gästebucheinträge verwendet.

Die dort gültigen Variablen sind:

`$staticpage_headline: String`
: enthält die Überschrift des Gästebuchs.

`$staticpage_pagetitle: String`
: enthält den Kurztitel des Gästebuchs.

`$staticpage_formorder: String`
: gibt an, ob das Eintragsformular oben (`top`) oder unten (`bottom`) angezeigt werden soll.

`$admin_delete: Bool`
: ist auf `true` gesetzt, wenn ein Administrator einen Gästebucheintrag löschen möchte.

`$admin_dodelete: Bool`
: ist auf `true` gesetzt, wenn ein Administrator eine Seite gelöscht hat.

`$admin_page: Int`
: enthält die Nummer der Seite, auf der ein Administrator eine Aktion ausführt.

`$admin_url: String`
: enthält die Ziel-URL des Gästebuchs.

[22] `http://de.php.net/strftime`

`$admin_target: String`
: enthält die Ziel-URL zum Löschen eines Gästebucheintrags.

`$base_url: String`
: enthält die URL des Blogs.

`$is_guestbook_url: String`
: enthält den Permalink zum Gästebuch-Plugin.

`$delete_sure: String`
: enthält einen Abfragetext, der benutzt werden kann, wenn ein Administrator einen speziellen Gästebucheintrag löschen möchte.

`$rip_entry: String`
: enthält einen Ausgabetext, wenn ein spezieller Gästebucheintrag gelöscht wurde.

`$is_show_mail, $is_show_url: Bool`
: sind auf `true` gesetzt, wenn im Eingabeformular die zusätzlichen Eingabefelder für die E-Mail-Adresse oder die Homepage eines Benutzers abgefragt werden sollen.

`$plugin_guestbook_preface: String`
: enthält den Einleitungstext des Gästebuch-Plugins.

`$plugin_guestbook_sent: String`
: enthält den Text, der angezeigt wird, wenn ein Benutzer einen Gästebucheintrag erstellt hat.

`$plugin_guestbook_action: String`
: enthält die URL zum Abschicken eines Gästebucheintrags.

`$plugin_guestbook_sname: String`
: enthält den Seitentitel der Gästebuch-URL.

`$plugin_guestbook_name, $plugin_guestbook_email,`

`$plugin_guestbook_url, $plugin_guestbook_comment: String` enthält die
: Benutzereingaben bei der Übermittlung eines Gästebuchkommentars.

`$plugin_guestbook_emailprotect: String`
: enthält den Text, der den Besucher über den Schutz der E-Mail-Adresse informiert.

`$plugin_guestbook_messagestack: Array,`

`$is_guestbook_message: Bool,`

`$error_occured: String, $guestbook_messages` enthält etwaige Fehlermeldungen.

`$plugin_guestbook_entry: Array (timestamp)`
: enthält Daten, die das Plugin zur Prüfung eines Gästebuch-Kommentars durch das Antispam-Plugin benötigt.

`$guestbook_paging: String`
: enthält HTML-Code für die Seiten-Blätterfunktion der Einträge.

`$guestbook_entry_paging: Bool`
: ist auf `true` gesetzt, wenn die Blätterungsfunktion aktiviert ist.

`$guestbook_entries: Array (email, name, homepage, body,`

`imgshorttime, timestamp, page, imgdelete)` enthält das Array mit den vorhandenen Gästebucheinträgen. `email, name, homepage, body` enthalten dabei die Stammdaten eines Gästebuchkommentars. `page` enthält die URL zum Löschen des jeweiligen Beitrags, `imgshorttime` enthält eine URL zur Grafik einer Uhr, `imgdelete` die URL zur Grafik eines Löschen-Icons. `timestamp` enthält das gemäß der Plugin-Konfiguration formatierte Datum eines Beitrags.

6.3.19 Kontaktformular serendipity_event_contactform

Das Plugin *Kontaktformular* dient dazu, in Ihrem Blog den Besuchern eine einfache Möglichkeit zu bieten, mit Ihnen in Kontakt zu treten. Dies ist besonders dann hilfreich, wenn Sie im Blog keine E-Mail-Adresse veröffentlichen möchten und bei der Kontaktaufnahme im Voraus bestimmte Daten abfragen möchten.

Ähnlich wie das Gästebuch-Plugin können Sie nach der Installation des Plugins eine eigenständige URL aufrufen, unter der das Kontaktformular eingebunden wird.

Das Plugin kann detailliert konfiguriert werden, um festzulegen, welche Angaben eines Besuchers Sie abfragen wollen. Dabei stehen Ihnen folgende Optionen zur Verfügung:

Permalink
: Globaler Permalink, um die Ausgabe des Plugins im Frontend sehen zu können. Details siehe Seite 238.

URL shorthand name
: Globale Variable, die einen alternativen Permalink zur Ausgabe des Plugins im Frontend bereitstellt. Details siehe Seite 238.

E-Mail-Adresse für Kontaktmails
: Hier tragen Sie die E-Mail-Adresse ein, bei der Kontaktanfragen eingehen sollen. Sie können mehrere Instanzen des Kontaktformular-Plugins benutzen, um unterschiedliche Kontaktarten für unterschiedliche Adressen/Formulare aufzusetzen. Abhängig vom

Mailserver[23] auf Ihrem Server können Sie mehrere E-Mail-Adressen mit Komma getrennt angeben. Alternativ können Sie eine Mailingliste einrichten, um Kontaktmails an mehrere Adressen zu verteilen.

Einführungstext
Der Einführungstext wird oberhalb des Kontaktformulars angezeigt. Hier können Sie beliebigen HTML-Code eintragen.

Dargestellter Text nach Übermittlung der Nachricht
Hier können Sie HTML-Code eintragen, der dem Benutzer nach Abschicken des Kontaktformulars angezeigt wird.

Als Artikel formatieren
Legt fest, ob das Kontaktformular im Layout normaler Blog-Einträge angezeigt werden soll, Details siehe Seite 239.

Use the dynamic tpl
Legt fest, welche Daten von einem Besucher beim Abschicken eines Kontaktformulars abgefragt werden sollen.

Standard legt als Pflichtfelder den Namen, die E-Mail-Adresse, die Homepage und einen Anfragetext fest.

Small Business fragt die Pflichtfelder Vorname, Nachname, E-Mail-Adresse und den Anfragetext ab. Im Vergleich zu **Standard** fehlt also die Homepage, aber dafür wird der Name in Vor- und Nachname aufgeteilt.

Detailed Form fragt zusätzlich zu den Feldern von *Small Business* auch noch eine postalische Adresse ab.

Als letzte Variante dient die Option **Custom**. Wenn Sie diese Option auswählen und auf den Button **Speichern** klicken, wird unterhalb des Auswahlfeldes ein detaillierter Erklärungstext und ein Eingabefeld eingeblendet. Dort können Sie individuell festlegen, welche Kontaktfelder im Formular eingebunden werden sollen.

Form field string
Die Syntax zur Eingabe der gewünschten Formularfelder sieht auf den ersten Blick sehr komplex aus, da sie in einem einzigen Texteingabefeld vorgenommen werden muss und nicht menügesteuert erfolgt. Dank dieser Eingabezeile haben Sie dafür eine sehr große Flexibilität in der Konfiguration der Felder.

Grundsätzlich müssen Sie für jedes Feld, das Sie im Kontaktformular darstellen wollen, einen Block bestehend aus vier Attributen eingeben. Mehrere Eingabefelder müssen Sie dabei mit dem Doppelpunkt (`:`) voneinander trennen. Die Attribute sind jeweils mit dem Semikolon (`;`) voneinander separiert.

Dazu ein kleines Beispiel. Wenn Sie nur ein Feld `Nachricht` und `Name` abfragen wollen, müssen Sie Folgendes im Eingabefeld eintragen:

[23] Der Mailserver muss mehrere Adressen im TO:-Header der E-Mail aufschlüsseln können.

```
Name;text:Nachricht;textarea
```

Es werden also zwei Elemente definiert, die jeweils nur zwei Attribute besitzen. Die vollständige Liste der Attribute kann an erster Stelle den Text `require` enthalten, um ein Feld zu einem Pflichtfeld zu machen. In obigem Beispiel wären beide Felder optional, könnten also vom Besucher leer gelassen werden. Um den Namen als Pflichtfeld zu definieren, wäre folgende Eingabe nötig:

```
require;Name;text:Nachricht;textarea
```

Als zweites Attribut[24] folgt der Name des Eingabefeldes (in unseren Beispielen `Name` und `Nachricht`).

Das dritte Attribut legt fest, wie das Eingabefeld dargestellt wird. Es stehen folgende Eingabetypen zur Verfügung:

`text`
: Einzeilige Text-Eingabebox

`checkbox`
: Ankreuzbox

`radio`
: Auswahlbox

`hidden`
: Verstecktes Feld

`password`
: Passwort-Feld

`textarea`
: Mehrzeiliges Eingabefeld

`select`
: Ausklappfeld

Als viertes und letztes Attribut können Sie angeben, welcher Standardwert in einem Formularfeld vorgegeben wird. Mit so einem Standardwert können Sie Ihren Benutzern verdeutlichen, welche Eingaben Sie erwarten:

```
require;Homepage;text;Ihre private Homepage-URL
```

Auch dieses letzte Attribut ist optional und kann weggelassen werden. Bei den Typen `checkbox` und `radio` kann als letztes Attribut `checked` angegeben werden, da eine Ankreuzbox nur entweder angekreuzt oder nicht angekreuzt sein kann. Bei dem Typ `select` kann stattdessen das Attribut `selected` eingetragen werden.

[24] Wenn Sie das Sonderattribut *require* nicht angegeben haben, verschiebt sich die Attribut-Reihenfolge um eins nach vorne.

Bei den beiden Typen `radio` und `select` gilt für das letzte Attribut eine leicht unterschiedliche Syntax. Für derartige Mehrfach-Auswahlfelder müssen Sie festlegen, welche Inhalte diese besitzen sollen. Diese werden mit einem weiteren Sonderzeichen, dem Pipe-Symbol (`|`) voneinander getrennt: `;Feld1,Wert1|Feld2,Wert2|Feld3,Wert3`. `FeldX` entspricht dabei der Angabe, die der Besucher in einer Auswahlbox sieht, während `WertX` dem entspricht, was als Wert später im Kontaktformular übermittelt wird. Ihre Eingaben für `FeldX` und `WertX` können also auch identisch lauten.

Um `radio`- und `select`-Felder zu verdeutlichen, konstruieren wir ein weiteres Beispiel. Der Besucher soll zum einen mittels Auswahlbox mitteilen, ob er ein Raucher ist. Dafür stehen ihm die Antwortmöglichkeiten *Militanter Nichtraucher*, *Raucher* und *Nicht von einem Schornstein zu unterscheiden* zur Verfügung. Vorausgewählt soll die erste Option sein. In der Kontaktmail soll zur besseren Übersichtlichkeit nur *nichtraucher*, *raucher* oder *potenziellerKunde* erscheinen.

Zum anderen soll der Besucher mittels Ausklappfeld festlegen, was sein Lieblingsessen ist. Hierfür soll er die Möglichkeiten *Pizza*, *Pommes*, *Blumen* und *Ich esse nichts das einen Schatten wirft* wählen können. Hier soll die letzte Option vorausgewählt sein, und diese Angabe soll ein Pflichtfeld sein. In der Kontaktmail sollen nur die Werte *pizza*, *pommes*, *blumen* und *nichts* erscheinen.

Dazu dient folgende Konfiguration (alles in einer Zeile einzugeben):

```
Raucher;radio;Militanter Nichtraucher,nichtraucher,checked|Raucher,
raucher|Nicht von einem Schornstein zu unterscheiden,potenziellerKu
nde:require;Lieblingsessen;select;Pizza,pizza|Pommes,pommes|Blumen,
blumen|Ich esse nichts das einen Schatten wirft,nichts,selected
```

Bei Eingabe dieser Konfigurationsmöglichkeiten müssen Sie stets darauf achten, dass Sie die richtigen Trennungszeichen verwenden, da es ansonsten zu unvorhergesehenen Auswirkungen kommen könnte. Ganz besonders gilt, dass Sie ein Komma (`,`) nicht innerhalb eines Auswahlfeld-Werts einsetzen dürfen, da dies als Trennzeichen verwendet wird.

Alle Einträge, die über das Kontaktformular-Plugin von Besuchern ausgefüllt wurden, können durch das Serendipity-Antispam-Plugin überprüft werden. Die Antispam-Methoden greifen dafür auf die Felder `name`, `url`, `comment` und `email` zu. Wenn Sie das Antispam-Plugin also sinnvoll integrieren wollen, sollten Sie darauf achten, bei einer dnyamischen Konfiguration der Felder die Bezeichner für Namen, URLs etc. mit exakt den genannten Bezeichnern zu verwenden.

Das Layout des Kontaktformulars kann durch die Smarty-Templatedateien `plugin_contactform.tpl` (für normale Kontaktformulare) und `plugin_dynamicform.tpl` angepasst werden. Dabei sind folgende Variablen vorhanden:

{`$is_contactform_error`}: Bool,

{`$plugin_contactform_error`}: String,

{`$comments_messagestack`}: String enthält etwaige Fehlermeldungen beim Absenden des Kontaktformulars.

{`$is_contactform_sent`}: Bool
gibt an, ob das Kontaktformular erfolgreich verschickt wurde.

{`$plugin_contactform_articleformat`}: Bool
gibt an, ob das Kontaktformular im Layout eines Blog-Artikels formatiert werden soll.

{`$plugin_contactform_name`}: String
enthält den Namen des Kontaktformulars.

{`$plugin_contactform_preface`}: String
enthält den Einleitungstext des Formulars.

{`$plugin_contactform_sent`}: String,

{`$plugin_contactform_message`}: String enthält den Text bei erfolgreichem Versand des Kontaktformulars.

{`$commentform_action`}: String
enthält das URL-Ziel des Formulars.

{`$commentform_sname`}: String
enthält den URL-Namen der gewählten Kontaktformular-Seite.

{`$commentform_name`}: String,

{`$commentform_url`}: String,

{`$commentform_email`}: String,

{`$commentform_comment`}: String enthält die Formulareingaben des Benutzers.

{`$commentform_dynamicfields`}: Array
enthält eine Liste der konfigurierten dynamischen Eingabefelder des Kontaktformulars.

Datenbanktabelle

Gästebucheinträge werden in der Datenbanktabelle `serendipity_guestbook` gespeichert:

`id`
enthält die fortlaufende ID eines Gästebucheintrags.

`ip`
: enthält die IP des Besuchers.

`name`
: enthält den Namen des Besuchers.

`homepage`
: enthält die Homepage des Besuchers.

`email`
: enthält die E-Mail-Adresse des Besuchers.

`body`
: enthält den Gästebucheintrag des Besuchers.

`timestamp`
: enthält die Uhrzeit, zu der der Gästebucheintrag erstellt wurde.

6.3.20 Diskussionsforum/phpBB-Kommentare serendipity_event_forum

Diskussionen in Blogs finden üblicherweise thematisch isoliert in den jeweiligen Blog-Artikeln statt. Im Gegensatz zu einem Diskussionsforum mit offenen Themenbereichen dient ein Blog üblicherweise der Einzelberichterstattung. Anstatt einer Diskussion gleichberechtigter Teilnehmer stellt ein Blog-Artikel meist die Meinung des jeweiligen Redakteurs dar, zu der sich Besucher dann äußern können.

Wenn ein Besucher eines Blogs ein Thema ansprechen und zur Diskussion bringen möchte, wäre dies in einem Blog-Artikel fehl am Platz. Daher stellt das Plugin *Diskussionsforum* eine Plattform innerhalb des Blogs bereit, wo sich Besucher unabhängig von den Blog-Artikeln austauschen können.

Generell können Sie etwas Derartiges auch durch die Installation einer separaten Foren-Software wie phpBB, FUDForum oder anderer auf Ihrem Webserver anbieten und in Ihrem Blog einfach einen Link auf eine solche separate Anwendung setzen.

Wenn Sie aber nur ein kleines Forum betreiben und dafür keine zweite Anwendung installieren und warten möchten, bietet sich jedoch das Foren-Plugin an. Dieses Plugin integriert sich in Serendipity und kann auch die Benutzerdatenbank Serendipitys weiterverwenden, um den Zugang gegebenenfalls nur auf registrierte Benutzer zu beschränken.

Das Forum lässt sich nach der Installation über `http://www.example.com/serendipity/index.php?` aufrufen. Im Gegensatz zu anderen Plugins bietet es keinen eigenständigen Permalink (siehe Seite 238) an, Sie können höchstens den Wert *forum* in der URL-Variable frei bestimmen.

Im Frontend des Forums können Besucher neue Themen erstellen und auf bestehende Themen antworten. Die Bedienung erfolgt dabei wie in jeder Foren-Software: Erst wählt der

Besucher in der Themenübersicht einen Bereich aus und kann dort die enthaltenen Beiträge ansehen. Über den Button **New Thread** kann der Besucher einen neuen Beitrag beginnen und dort eine Überschrift und seinen Beitrag erstellen. Abhängig von der Konfiguration des Foren-Plugins können Besucher oder Administratoren auch Dateianhänge zu einem neuen Thema in das Forum einstellen.

Zu jedem Beitrag kann ein Besucher (sofern er einen Redakteurs-Login zum Blog besitzt) die E-Mail-Benachrichtigung zu neuen Themen aktivieren. Administratoren können zudem direkt im Frontend Beiträge überarbeiten, löschen oder auch ganze Themenzweige sperren.

Das Backend des Plugins zur Einrichtung neuer Unterforen kann ein Redakteur über das Serendipity-Backend erreichen. Dort befindet sich der Menüpunkt **Einträge** → **Diskussions-Forum/phpBB-Kommentare**. In diesem Bereich werden alle bestehenden Foren aufgelistet, das neue Unterforum kann über den Button **Neues Forum hinzufügen** eingerichtet werden. Bestehende Unterforen können durch einen Klick auf den jeweiligen Namen bearbeitet werden.

Unabhängig von diesem eigenständigen Forum bietet das Plugin ein weiteres Feature an, um Ihr Blog mit einer phpBB-Foreninstallation zu verbinden. Mit Hilfe dieses Features können Sie die Kommentare zu Blog-Artikeln in einem unabhängigen Forum verwalten und somit von Serendipity loslösen. Dies macht besonders dann Sinn, wenn Sie über die von Serendipity angebotenen Kommentarmöglichkeiten hinaus oft besonders lange Diskussionen zu Artikeln führen möchten. Auch wenn Sie bereits ein phpBB-Forum mit bestehenden Diskussionen betreiben, können Sie dank des Plugins Serendipity als eine Art News-System einsetzen und sämtliche Diskussionen wie früher über phpBB führen.

Die Konfiguration des Plugins bietet folgende Optionen:

Seitentitel
: Der **Seitentitel** des Forums wird im Inhaltsbereich dargestellt und kann individuell festgelegt werden.

Kopfzeile/Beschreibung
: Unterhalb des Seitentitels können Sie eine **Kopfzeile/Beschreibung** definieren.

Statische URL
: Hier können Sie ein einfaches Wort eintragen, das für die URL verwendet wird, über die Sie das Forum aufrufen können. Standardmäßig ist hier *forum* eingetragen, und die URL für das Forum lautet somit `http://www.example.com/serendipity/index.php?serendipity[subpage]=forum`.

  Das Forum müssen Sie selbständig über einen HTML-Klotz oder das Bearbeiten Ihres Templates in das Blog einfügen, damit Ihre Besucher dorthin finden können. Achten Sie also darauf, den Link dorthin immer übereinstimmend mit der Konfiguration anzulegen. Das hier eingetragene Wort darf keine Sonder- und Leerzeichen enthalten.

Enable phpBB mirroring
: Anstatt Kommentare zu Blog-Artikeln durch Serendipity zu verwalten, möchten man-

che Blog-Betreiber lieber ein spezialisiertes Forensystem zur Diskussionsführung benutzen. Solche Forensysteme haben oft einfacher strukturierte Kommentarmöglichkeiten und sind zugänglicher für Benutzer, die sich mit Foren bereits auskennen, Blogs eher als redaktionelles System ansehen und ihre Diskussionen im „liebgewonnenen" System führen wollen.

Wenn Sie die Option **Enable phpBB mirroring** aktivieren, wird das Plugin sämtliche Kommentare, die im Blog angezeigt werden, aus einer bestehenden phpBB 2.x- oder 3.x-Installation (je nach gewähltem Wert dieser Option) beziehen. Wenn Sie neue Kommentare im Blog eintragen, werden diese automatisch in die phpBB-Forendatenbank eingefügt. Für Ihre Besucher erscheinen alle Kommentare also sowohl im Forum als auch im Blog, werden aber nur an der zentralen Stelle im Forum tatsächlich verwaltet. Auch wenn Sie im Blog einen neuen Artikel erstellen, wird das Plugin Ihren Artikel automatisch im Forum einstellen, damit er dort diskutiert werden kann.

Damit das Plugin diesen Abgleich durchführen kann, muss es auf die Datenbank von phpBB zugreifen können. Dazu dienen die Konfigurationsfelder `database username, password, server, name` und `table prefix`. Tragen Sie hier Daten einer bestehenden phpBB-Konfiguration ein. Sollten Sie phpBB noch nicht auf Ihrem Webserver installiert haben, müssen Sie dies nachholen.[25]

Ein phpBB-Forum kann Kommentare immer erst in einem Themenbereich (dem **forum**) anlegen. Diese Themenbereiche legen Sie in der phpBB-Installation an, jedes Forum erhält eine eigene ID. Das offizielle Serendipity-Forum wird beispielsweise mit phpBB betrieben, und der deutsche Bereich hat die ID `10` (`http://board.s9y.org/viewforum.php?f=10`). Diese ID des Zielforums müssen Sie in der Konfiguration des Plugins im Feld **(optional) phpBB target forum ID** eintragen.

Durch die Abgrenzung des Forums ist zudem gewährleistet, dass Sie in Ihrem phpBB-Forum auch andere, vom Blog unabhängige Bereiche anlegen können.

Wenn Sie einen Artikel im Blog erstellen, wird dieser mit der User-ID eingetragen, die Sie im Feld **(optional) phpBB target poster ID** konfiguriert haben. Diese User-ID muss einer im Forum bestehenden ID entsprechen, am besten legen Sie also im Forum einen eigenständigen Benutzer an, der den Namen Ihres Blogs trägt. So können die Forennutzer einfach herausfinden, welche Artikel „das Blog" erstellt hat.

Sämtliche Kommentare, die vom Blog aus erstellt werden, trägt das Plugin als anonyme Benutzer mit dem im Blog eingetragenen Benutzernamen direkt in die Datenbank ein. So kann zwar ein Kommentar nicht eindeutig einem bestehenden Forennutzer zugeordnet werden, aber durch die Beibehaltung der Benutzernamen bleibt dies eindeutig. Wenn ein Benutzer direkt über das Forum anstatt über das Blog kommentiert, kann die korrekte Benutzerzuordnung selbstverständlich beibehalten werden.

Absoluter Server-Pfad zum uploads-Verzeichnis

Hier tragen Sie den absoluten Server-Pfad zu dem Verzeichnis ein, in dem über das Forum hochgeladene Dateien gespeichert werden.

[25] Die Installationsanweisungen finden Sie unter `http://www.phpbb.com/`.

Pfad zu dem Plugin
Das Foren-Plugin bringt einige Grafikdateien mit, beispielsweise für die Buttons zur Erstellung neuer Beiträge. Damit diese Grafiken angezeigt werden können, muss das Plugin den vollständigen HTTP-Pfad hinter der Domainangabe kennen.

Standardmäßig ist hier der korrekte Pfad voreingetragen, er sollte daher nur geändert werden, wenn Sie Ihre Plugin-Verzeichnisse umbenannt haben.

Datumsformat
Das **Datumsformat** bestimmt, mit welcher Formatierung das Datum zu Diskussionsbeiträgen dargestellt wird. Standardmäßig ist hier die englische Syntax eingetragen, die deutsche Datumsformatierung wäre `d.m.Y`.[26]

Zeitformatierung
Analog zum Datumsformat tragen Sie hier das Format ein, mit dem eine Uhrzeit zu Diskussionsbeiträgen angezeigt werden soll.

Einträge pro Seite
Die **Einträge pro Seite** bestimmen, wie viele Diskussionsbeiträge in Übersichtsseiten angezeigt werden. Weitere Seiten sind blätterbar.

Hintergrundfarbe der Titelzeilen, 1. und 2. Hintergrundfarbe,

Farbe für Schriftzüge Über diese Konfigurationsfelder können Sie mehrere Farbeinstellungen vornehmen, die das Plugin zur Darstellung des Forums benutzt. Die Farben müssen dabei im HTML-typischen RGB-Format (Hex #RRGGBB) eingetragen werden.

[26] Die zur Verfügung stehenden Variablen sind auf `http://de.php.net/date` aufgeführt.

Spamblock-Plugin benutzen
: Um Spam zu verhindern, können alle Diskussionsbeiträge durch das Anti-Spam-Plugin behandelt werden. Dies führt zum Beispiel auch dazu, dass Captchas (siehe Seite 44) für jeden Beitrag erforderlich sind.

Sollen die Textformatierungs-Plugins benutzt werden
: Wenn Sie diese Option aktivieren, können die Diskussionsbeiträge von Benutzern genauso wie Kommentare zu Blog-Artikeln mit Ihren installierten Textformatierungs-Plugins (siehe Seite 237) bearbeitet werden.

Textformatierungs-Plugins für unregistrierten User deaktivieren
: Das Foren-Plugin kann eingeloggte Redakteure von anonymen Besuchern unterscheiden. Um unregistrierten Besuchern zu verbieten, z. B. Links in Diskussionsbeiträge einzufügen, können Sie gezielt diesen Besuchern das Benutzen der Textformatierungs-Plugins verbieten. Dazu müssen Sie die Option **Textformatierungs-Plugins für unregistrierten User deaktivieren** auf *Ja* stellen.

Datei-Upload für registrierte User, Datei-Upload für Gäste
: Bei der Erstellung eines Diskussionsbeitrags können Sie es registrierten Benutzern und auch Gästen erlauben, Dateien hochzuladen. Diese Dateien werden dem jeweiligen Beitrag angehängt und können von Besuchern des Forums heruntergeladen werden. Sinnvoll ist dies, um Ihre Besucher beispielsweise Screenshots oder Fotos anhängen zu lassen.

  Beachten Sie hierbei, dass die Dateien auf Ihrem eigenen Webserver gespeichert werden; möglicherweise kann dies von Ihren Besuchern missbraucht werden und Ihren Speicherplatz aufbrauchen. Je nach Einsatzzweck Ihres Forums kann es also empfehlenswert sein, Gästen keine Upload-Möglichkeiten zu erlauben.

Anzahl gleichzeitiger Datei-Uploads, Max. Anzahl Dateien in einem Posting
: Standardmäßig darf ein Forenteilnehmer nur drei Dateien pro Diskussionsbeitrag einstellen. Über das Feld **Max. Anzahl Dateien in einem Posting** können Sie aber auch mehrere Uploads zulassen.

  Jede Datei muss dabei vom Besucher einzeln hochgeladen werden, damit Ihr Webserver nicht überlastet wird. Über die Option **Anzahl gleichzeitiger Datei-Uploads** können Sie jedoch festlegen, dass auch mehr als nur eine Datei gleichzeitig hochgeladen werden kann.

Max. Anzahl Datei-Uploads pro User
: Sie können generell beschränken, wie viele Dateien ein Benutzer maximal hochladen darf. Nachdem ein Benutzer das Limit erreicht hat, darf er keine weiteren Dateien mehr einstellen.

  Diese Option wirkt ausschließlich für registrierte Benutzer. Da ein Gast für das Plugin immer als ein neuer Benutzer gilt, können Uploads für Gäste nicht in ihrer Anzahl beschränkt werden. Im Zweifelsfall sollten Sie daher die Option **Datei-Upload für Gäste** nicht aktivieren.

Benachrichtigungs-E-Mail: E-Mail-Adresse, Name

Wenn Ihre Forenteilnehmer die E-Mail-Benachrichtigung aktiviert haben, werden sie bei Antworten zu ihren abonnierten Themen via E-Mail benachrichtigt. Tragen Sie in die beiden Konfigurationsfelder ein, von welcher E-Mail-Adresse diese Benachrichtigungen stammen sollen und wie der Absendername lauten soll.

Viele Webserver erlauben den E-Mail-Versand nur von den Adressen, für die sie freigeschaltet sind, daher können Sie möglicherweise nicht jede beliebige E-Mail-Adresse an dieser Stelle eintragen.

Admin benachrichtigen

Wenn Sie die Option **Admin benachrichtigen** aktivieren, erhalten alle Administratoren des Blogs eine E-Mail, sobald neue Forenbeiträge von Teilnehmern erstellt wurden.

Das Foren-Plugin benutzt zur Darstellung der Inhalte weitestgehend Smarty-Template-Dateien. Diese befinden sich im Unterverzeichnis `plugins/serendipity_event_forum/templates`. Die Datei `boardlist.tpl` stellt die Forenübersichten dar, `threadlist.tpl` die Themenübersichten und `postlist.tpl` die jeweiligen Beiträge. Die Datei `newthread.tpl` stellt das Formular zum Erzeugen eines neuen Beitrags dar, `editform.tpl` das Formular zum Bearbeiten eines bestehenden Beitrags und `replyform.tpl` zum Antworten auf einen Beitrag. `deleteform.tpl` dient der Ausgabe zum Löschen von Beiträgen durch Administratoren.

Einige HTML-Codes zur Darstellung von Buttons und Übersichten werden leider auch in der Plugin-Datei `serendipity_event_forum.php` vorgenommen. Daher würde eine Dokumentation der notwendigen Variablen den Rahmen dieses Buches sprengen. Schauen Sie sich daher bitte zur Bearbeitung die jeweils bestehenden Dateien an. Der Administrationsbereich ist vollständig innerhalb des PHP-Codes des Plugins enthalten und kann derzeit nicht durch Smarty-Template-Dateien angepasst werden. Das Plugin unterscheidet Forenbenutzer nicht anhand ihrer Benutzergruppen, sondern behandelt alle registrierten Benutzer ausschließlich anhand ihres Benutzerranges (Userlevel).

Derzeit verfügt das Plugin über keinen aktiven Entwickler, es wäre für die Zukunft jedoch schön, den PHP-Code des Plugins etwas zu entschlacken, um neue Features leichter einbauen zu können. Sollten Sie sich angespornt fühlen, dem Plugin etwas Gutes zu tun, melden Sie sich bitte im Serendipity-Forum.

Datenbanktabellen

In der Datenbanktabelle `serendipity_dma_forum_boards` werden die jeweiligen Unterforen gespeichert:

`boardid`
: enthält die ID des jeweiligen Unterforums.

`name`
: enthält den Namen des Unterforums.

`description`
: enthält die Beschreibung des Unterforums.

`sortorder`
: enthält einen numerischen Wert zur Sortierung der Unterforen untereinander.

`threads`
: enthält die Anzahl der in diesem Unterforum vorhandenen Themen (*Threads*).

`posts`
: enthält die Anzahl der in diesem Unterforum geschriebenen Beiträge (*Posts*).

`views`
: enthält die Anzahl der Lesezugriffe zu diesem Unterforum.

`flag`
: gibt an, ob ein Unterforum deaktiviert wurde.

`lastauthorid`
: enthält die ID des Redakteurs, der zuletzt einen Beitrag verfasst hat.

`lastauthorname`
: enthält den Namen des Redakteurs, der zuletzt einen Beitrag verfasst hat.

`lastthreadid`
: enthält die ID des aktuellsten Themas.

`lastpostid`
: enthält die ID des aktuellsten Beitrags.

`lastposttime`
: enthält die Uhrzeit des aktuellsten Beitrags.

Die Tabelle `serendipity_dma_forum_threads` speichert die jeweiligen Themen im Forum:

`boardid`
: enthält die ID des Unterforums, in dem sich das Thema befindet.

`threadid`
: enthält eine fortlaufende ID dieses Themas.

`title`
: enthält den Titel des Themas.

`replies`
: enthält die Anzahl an Antworten zu dem Thema.

`views`
: enthält die Anzahl an Lesezugriffen zu diesem Thema.

`flag`
: gibt an, ob dieser Thread aktiv ist.

`notifymails`
: enthält eine Liste aller E-Mail-Adressen, die bei Eintreffen eines neuen Beitrags benachrichtigt werden sollen.

`announce`
: gibt an, ob das Thema eine Ankündigung ist.

`lastauthorid`
: enthält die ID des Redakteurs, der zuletzt einen Beitrag zu diesem Thema geschrieben hat.

`lastauthorname`
: enthält den Namen des Redakteurs, der zuletzt einen Beitrag zu diesem Thema geschrieben hat.

`lastpostid`
: enthält die ID des letzten Beitrags zu dem Thema.

Die Artikel werden in der Tabelle `serendipity_dma_forum_posts` gespeichert:

`threadid`
: enthält die ID des zugeordneten Themas.

`postid`
: enthält die fortlaufende ID des Beitrags.

`postdate`
: enthält das Erstellungsdatum des Beitrags.

`title`
: enthält den Titel des Beitrags.

`message`
: enthält den Inhalt des Beitrags.

`flag`
: gibt an, ob dieser Beitrag aktiv ist.

`authorid`
: enthält die ID des erstellenden Redakteurs.

`authorname`
: enthält den Namen des erstellenden Redakteurs.

`editcount`
: enthält die Anzahl an Bearbeitungen dieser Nachricht.

Zusätzliche Foren-Benutzer werden in der Tabelle `serendipity_dma_forum_users` gespeichert:

`authorid`
: enthält die ID eines Benutzers.

`posts`
: enthält die Anzahl an Beiträgen des Benutzers.

`visits`
: enthält die Anzahl an Besuchen des Benutzers.

`lastvisit`
: enthält das Datum des letzten Besuchs.

`lastpost`
: enthält die ID des letzten Beitrags dieses Benutzers.

`uploadids`
: enthält die IDs der von dem Benutzer hochgeladenen Dateien.

Benutzer können, abhängig von der Konfiguration des Plugins, selbständig Dateien hochladen. Diese werden in der Datenbanktabelle `serendipity_dma_forum_uploads` gespeichert:

`postid`
: enthält die ID des Beitrags, in dem eine Datei eingebunden wurde.

`authorid`
: enthält die ID des hochladenden Redakteurs.

`uploadid`
: enthält eine fortlaufende ID für die hochgeladene Datei.

`uploaddate`
: enthält das Datum, an dem die Datei hochgeladen wurde.

`filesize`
: enthält die Dateigröße.

`sysfilename`
: enthält den Dateinamen.

`realfilename`
: enthält den vollständigen Pfad zur Datei.

`dlcount`
: enthält die Anzahl an Downloads dieser Datei.

6.3.21 Downloadmanager serendipity_event_downloadmanager

Mittels der Serendipity-Mediendatenbank haben Sie als Blog-Redakteur die Möglichkeit, Dateien in Blog-Artikel einzubinden, damit Ihre Besucher diese herunterladen können.

Wenn Sie jedoch viele Dateien übersichtlich zum Download anbieten wollen, ist diese Methode mit zu viel manueller Arbeit verbunden. Aus diesem Grund wurde das *Downloadmanager*-Plugin entwickelt.

Mit diesem Plugin können Sie komfortabel einen Download-Katalog anlegen. Auf beliebige Unterrubriken verteilt können Sie hier Ihre Dateien sortieren und dem Besucher anbieten.

Sobald das Plugin installiert wurde, können Sie über den Menüpunkt **Einträge** → **Downloadmanager** Kategorien einrichten. Um eine neue Kategorie einzurichten, tragen Sie den Namen im Feld **Kategorie-Name** ein. Über das Ausklappfeld **Unter-Kategorie von** können Sie festlegen, ob die neu zu erstellende Kategorie unterhalb einer bestehenden angelegt werden soll.

Alle vorhandenen Kategorien werden am Ende der Verwaltungsoberfläche in einer Baumstruktur angezeigt. Dort können Sie bestehende Kategorien umbenennen, indem Sie einen neuen Namen eintragen und auf **Los!** klicken. Kategorien können Sie löschen, indem Sie auf das zugehörige Bild mit der Mülltonne klicken. Wenn ein Warndreieck über der Mülltonne angezeigt wird, ist dies ein Indikator, dass beim Löschen auch vorhandene Unterkategorien entfernt würden. Um eine Kategorie nur vorübergehend zu verstecken, können Sie auf den danebenstehenden Kreis klicken. Rechts neben dem Namen einer Kategorie sehen Sie zur Information die Anzahl der darin angelegten Dateien.

Um Dateien in eine bestehende Kategorie einzustellen, müssen Sie auf das Ordnersymbol links neben dem jeweiligen Kategorienamen klicken.

In der folgenden Oberfläche haben Sie insgesamt drei Möglichkeiten, Dateien zuzuordnen. Dabei greift das Plugin auf Unterverzeichnisse zu, die Sie in der Konfiguration des Plugins festlegen können. Diese Unterverzeichnisse sollten Sie mittels FTP manuell anlegen oder alternativ das bestehende `uploads`-Unterverzeichnis verwenden.

Die erste Zuordnungsmöglichkeit ist das Hochladen von Dateien über die Downloadmanager-Oberfläche. Hierfür klicken Sie auf den Link **Dateien hochladen...**, der sich neben dem Informationstext **Dateien in dieser Kategorie** befindet. Mittels dieser Oberfläche können Sie Dateien von Ihrer Festplatte auswählen und optional eine Beschreibung dieser Datei in der Eingabebox darunter eintragen. Sie können maximal fünf Dateien auf einmal hochladen.

Die maximale Dateigröße richtet sich nach der PHP-Konfiguration Ihres Webservers (siehe Seite 33).

Die zweite Zuordnungsmöglichkeit ist die Übernahme von Dateien, die Sie mittels FTP in das sogenannte *incoming-Verzeichnis* hochgeladen haben. Die Oberfläche zeigt alle in diesem Verzeichnis vorhandenen Dateien an, und Sie können durch einen Klick auf das Ordnersymbol mit dem Pfeil darin auswählen, welche Dateien Sie in die Downloadmanager-Kategorie übernehmen möchten. Die Datei wird daraufhin aus dem incoming-Verzeichnis in das Zielverzeichnis verschoben.

Als letzte Zuordnungsmöglichkeit können Sie Dateien auswählen, die Sie bereits in die Mediendatenbank von Serendipity hochgeladen haben. Ähnlich wie beim incoming-Verzeichnis können Sie hier eine bestehende Datei durch Klick auf das Ordnersymbol mit dem grünen Pfeil in die Downloadmanager-Kategorie übernehmen. Dabei wird die Datei aber nicht verschoben, sondern einfach kopiert.

Nachdem Sie über diese Mechanismen Dateien zugeordnet haben, werden diese in der Dateiübersicht am Anfang der Seite dargestellt. Neben Informationen zum Dateinamen, den durchgeführten Downloads von Besuchern und der Dateigröße können Sie eine Datei über das Mülltonnen-Symbol auch wieder löschen. Die Dateibeschreibung jeder Datei können Sie ändern, indem Sie auf den jeweiligen Dateinamen klicken. Dort können Sie auch eine Datei einer anderen Kategorie zuordnen, um sie zu verschieben. Über den Link **Zurück...** kehren Sie zur Kategorie-Übersicht zurück.

Den Downloadmanager können Ihre Besucher über den im Plugin konfigurierten Permalink aufrufen, standardmäßig `http://www.example.com/serendipity/index.php?serendipity[su ager`.

In dieser Oberfläche sehen die Besucher eine Verzeichnisstruktur ähnlich Ihrer Administrationsoberfläche. Dort können die Besucher beliebig navigieren und eingestellte Dateien herunterladen.

Der Zugriff auf die Datei erfolgt stets über das Downloadmanager-Plugin, die Datei wird also nicht direkt über das Verzeichnis heruntergeladen. Dadurch können Ihre Besucher nicht einfach beliebige Dateien herunterladen, und der Downloadmanager ist in der Lage, die Anzahl der Downloads zu zählen. Gleichzeitig bedeutet dieser technische Kniff jedoch auch, dass Ihre Besucher nur Dateien herunterladen können, die das Plugin komplett in den Speicher laden kann. Daher ist das Plugin für Dateien größer als das festgelegte PHP-Speicherlimit (meist 8MB) nicht geeignet.

Folgende Konfigurationsoptionen bietet das Plugin:

Seitentitel
: Der **Seitentitel** wird als Überschrift in der Downloadmanager-Übersicht für die Besucher eingeblendet.

Kopfzeile
: Die **Kopfzeile** dient als Unter-Überschrift für die Downloadmanager-Übersicht.

Statische URL
URL-Titel der Seite, mit dem Sie die URL `http://www.example.com/serendipity/index.php?serendip` `ager` aufrufen können. Details siehe Seite 238.

Permalink
Den alternativen Permalink zum Zugriff auf das Plugin können Sie in diesem Konfigurationsfeld eintragen. Details siehe Seite 238.

Pfad zu „incoming“
Das `incoming`-Verzeichnis dient der Speicherung von Dateien, die Sie via FTP hochladen, um diese in den Downloadmanager zu importieren. Das hier festgelegte Verzeichnis muss Schreibrechte für den Webserver besitzen (z. B. 0777).

Pfad zu „download“-Verzeichnis
Im Gegensatz zum temporären `incoming`-Verzeichnis werden die Dateien im `download`-Verzeichnis dauerhaft gespeichert und von dort aus an die Besucher ausgeliefert.

HTTP-path to plugin
Dieses Verzeichnis legt den HTTP-Pfad fest, der zu dem Downloadmanager-Plugin-Verzeichnis führt. Aus diesem Verzeichnis werden die Icons und Grafikdateien bezogen, die das Plugin zur Darstellung benötigt. Normalerweise müssen Sie an der Standardeinstellung des Plugins nichts ändern.

Icon Breite, Icon Höhe
Das Plugin liefert einige Icons für bekannte Dateierweiterungen (ZIP, PDF, Bilddateien etc.) mit, die in der Dateiübersicht für den Besucher angezeigt werden. Diese Dateien sind standardmäßig 18 Pixel breit und 20 Pixel hoch. Wenn Sie die Grafikdateien gerne gegen andere und größere Icons austauschen wollen, können Sie über die Optionen **Icon Breite** und **Icon Höhe** die neuen Ausmaße der Grafik festlegen.

Beachten Sie: Wenn Sie die Größe ändern, ohne die Grafikdateien auszutauschen, werden die Grafiken nur vom Browser vergrößert und dadurch möglicherweise grob aufgelöst dargestellt.

Datumsformat
Das Feld **Datumsformat** legt fest, wie die Datumsangabe von Dateien dargestellt werden soll. Standardmäßig ist hier das englische Format `Y/m/d, h:ia` voreingestellt. Das deutsche Format könnnen Sie mittels `d.m.Y H:i` einstellen. Weitere verfügbare Variablen der PHP `date()`-Funktion finden Sie unter `http://de.php.net/date`.

Versteckte Kategorien für registrierte User zeigen
Wenn Sie eine Kategorie des Downloadmanager-Plugins verstecken, werden die darin befindlichen Dateien normalen Besuchern nicht angezeigt. Wenn Sie die Option **Versteckte Kategorien für registrierte User zeigen** aktivieren, können diese Dateien allen Besuchern dargestellt werden, die im Blog als Redakteure angemeldet sind.

Über diese Option können Sie daher spezielle Downloads nur für ausgewählte Benutzer zulassen.

Dateinamen anzeigen

Ist die Option **Dateinamen anzeigen** aktiviert, werden die ursprünglichen Dateinamen der hochgeladenen Dateien angezeigt. Bei deaktivierter Option sehen die Besucher nur die Dateibeschreibung.

Anzahl der Datei-Downloads

Haben Sie die Option **Anzahl der Datei-Downloads** aktiviert, wird in der Kategorie-Übersicht für Besucher die Anzahl der Downloads jeder einzelnen Datei angezeigt. Bei deaktivierter Option können Besucher die Anzahl der Downloads jedoch nach wie vor in der Detailansicht einer Datei einsehen. Wollen Sie die Anzahl auch dort verstecken, müssen Sie die Smarty-Template-Datei `dlmanager.filedetails.tpl` bearbeiten.

Show filesize

Wenn Sie die Option **Show filesize** aktivieren, wird die Dateigröße einer Datei in der Kategorie-Übersicht für Besucher eingeblendet.

Datei-Datum anzeigen

Wenn Sie die Option **Show filesize** aktivieren, wird das Erstellungsdatum einer Datei in der Kategorie-Übersicht für Besucher eingeblendet.

Bezeichnung der Dateinamen, Dateigröße, Dateidatum und

Download-Felder In diesen Konfigurationsfeldern können Sie eintragen, wie die jeweiligen Felder in der Downloadmanager-Übersicht für Besucher benannt werden.

Die Ausgabe des Plugins erfolgt in der Besucheransicht teilweise mittels dreier Smarty-Template-Dateien. Die Datei `dlmanager.catlist.tpl` erstellt die Kategorie-Übersicht, die Datei `dlmanager.filelist.tpl` die Datei-Übersicht, und die Datei `dlmanager.filedetails.tpl` regelt die Ausgabe der Download-Seite einer Datei.

Leider sind alle drei Smarty-Dateien nur teilweise vom PHP-Code des Plugins losgelöst. Ein Großteil der HTML-Darstellungen wird nach wie vor über das Plugin ausgegeben und kann nicht mittels Smarty angepasst werden. Aufgrund der hohen Vermischung von Smarty- und PHP-Variablen ist eine Dokumentation der zur Verfügung stehenden Variablen an dieser Stelle zu komplex – bitte schauen Sie sich die bestehenden Dateien an, um die Verwendung der Variablen nachzuvollziehen.

Derzeit verfügt das Plugin über keinen aktiven Entwickler, es wäre für die Zukunft jedoch schön, den PHP-Code des Plugins etwas zu entschlacken, um neue Features leichter einbauen zu können. Sollten Sie sich angespornt fühlen, dem Plugin etwas Gutes zu tun, melden Sie sich bitte im Serendipity-Forum.

Datenbanktabellen

Die Datenbanktabelle `serendipity_dma_downloadmanager_files` enthält Verweise zu den im Downloadmanager eingebundenen Dateien:

`id`
: enthält eine fortlaufende, eindeutige ID einer Datei.

`catid`
: enthält die einer Datei zugeordnete Kategorie.

`timestamp`
: enthält das Hochladedatum der Datei.

`systemfilename`
: enthält den Dateinamen.

`realfilename`
: enthält den Dateinamen mitsamt seines Pfads.

`description`
: enthält die Beschreibung der Datei.

`filesize`
: enthält die Dateigröße (in Bytes).

`dlcount`
: enthält die bisherige Anzahl der Downloads dieser Datei.

In der Tabelle `serendipity_dma_downloadmanager_categories` werden die Kategorien des Downloadmanagers (in Nested-Set-Struktur[27]) gespeichert:

`node_id`
: enthält die ID der Kategorie.

`root_id`
: enthält die ID einer etwaigen Oberkategorie.

`payload`
: enthält den Namen der Kategorie.

`lft`
: enthält die ID der vorigen Kategorie.

`rgt`
: enthält die ID der nächsten Kategorie.

`hidden`
: gibt an, ob die Kategorie derzeit versteckt wird.

6.3.22 RSS Aggregator serendipity_event_aggregator

Als *Aggregator* bezeichnet man eine Software, die verschiedene RSS-Feeds (siehe Seite 38) zusammenführen (*aggregieren*) kann. RSS-Feeds sind die fundamentale Basis für *Content Syndication*, also den Vertrieb von Inhalten.

Serendipitys *RSS Aggregator*-Plugin kümmert sich darum, dass die Meldungen von beliebigen RSS-Feeds in Ihr Blog importiert werden können. Gewissermaßen lässt sich ein Aggregator mit den Borg vergleichen: Er assimiliert sämtliche Datenquellen und fügt sie einem Kollektiv hinzu.

Ein Blog, in dem unterschiedliche Feeds zusammengeführt dargestellt werden, bezeichnet man häufig als *Planet*. In den meisten Fällen werden solche Blogs ohne eigenständige redaktionelle Inhalte geführt, da andernfalls die Vermischung von eigenem und fremdem Inhalt für Besucher sehr verwirrend sein könnte.

[27] Siehe `http://www.dbazine.com/oracle/or-articles/tropashko4`

Besonders beliebt sind thematisch zusammengeführte Blogs. Wenn Sie als Besitzer einer Gärtnerei selbst kein Blog führen, aber den Besuchern Ihrer Webseite einen Mehrwert anbieten wollen, könnten Sie beispielsweise ein solches aggregiertes Blog einrichten, in dem Sie die RSS-Feeds anderer Partner-Gärtnereien oder Ihrer Lieferanten zusammenführen.

Das Plugin importiert Artikel fremder RSS-Feeds so, als seien es ganz normale Blog-Einträge. Daher können Sie als Administrator des Blogs später auch Änderungen an den Blog-Einträgen vornehmen und die jeweiligen RSS-Feeds auch eigenständigen Blog-Kategorien zuordnen.

Bei derartiger Zusammenführung ist es vor allem wichtig, dass Sie etwaige urheberrechtliche Fragen klären. Sie dürfen fremde RSS-Feeds nicht ohne Zustimmung des Betreibers auf Ihren Seiten einbinden.

Sobald das RSS Aggregator-Plugin installiert ist, können Sie über den Menüpunkt **Einträge** → **RSS Aggregator** die gewünschten Feeds einstellen.

In der Oberfläche sehen sie eine Liste, in der Sie einen neu zu importierenden RSS-Feed festlegen können. In das Feld **Feed name** tragen Sie einen beliebigen Namen ein, der Ihnen später die Identifizierung eines Feeds erleichert. Rechts daneben tragen Sie die URL des jeweiligen RSS-Feeds ein, inklusive aller potenziellen URL-Parameter. In dem Eingabefeld darunter können Sie zudem die URL der Homepage angeben. In dem Mehrfachauswahlfeld rechts daneben legen Sie fest, welcher Blog-Kategorie ein RSS-Feed zugeordnet werden soll. Jeder Beitrag des RSS-Feeds wird später im Frontend innerhalb der gewählten Kategorie dargestellt.

Ein besonderes Feld stellt die große Eingabebox **Filter-Ausdruck** dar. Dort können Sie einen regulären Ausdruck eintragen, der beim Importieren jedes Artikels des fremden RSS-Feeds ausgewertet wird. Nur wenn ein hier eingetragener Suchausdruck in einem Artikel vorhanden ist, wird der Artikel ins Blog importiert, ansonsten wird er verworfen. Der Filter wird dabei auf die Felder *Titel* und *Inhalt* angewendet. Dies ist hilfreich, wenn Sie beispielsweise von der Gärtnerei *Rosenrot*, die für ihre hervorragenden Blumenmesse-Berichte bekannt ist, nur die Artikel importieren wollen, die das Schlüsselwort *Messe* enthalten.

Da es sich bei dem Filter um einen regulären Ausdruck handelt, müssen Sie etwaige Sonderzeichen *escapen*, also einen Backslash (\) voranstellen.[28] Erweiterte Suchmuster wie `Messe(bericht|report|review)(` können ebenfalls verwendet werden, damit Begriffe wie *Messeberichte* oder *Messereviews* ebenfalls akzeptiert werden. Mehrere Filterbegriffe können Sie durch das Tilde-Zeichen ( ) voneinander trennen, die Suchwörter werden dann ODER-kombiniert. Wenn Sie also `Messe Rosen Stock` eintragen, wird jeder Artikel importiert, der die Wörter *Messe*, *Rosen* oder *Stock* irgendwo im Inhalt oder Betreff der RSS-Nachricht enthält.

Ein leerer Filter-Ausdruck importiert alle Einträge des RSS-Feeds. Beachten Sie dabei, dass der RSS-Aggregator nur das importieren kann, was der RSS-Feed anbietet. Der vollständige Artikeltext ist meist nicht in einem RSS-Feed enthalten und kann daher auch vom Plugin nicht ausgewertet/eingelesen werden!

[28] Unter `http://de3.php.net/manual/de/reference.pcre.pattern.syntax.php` sind solche Sonderzeichen aufgeführt.

Wenn Sie alle Felder eines neuen Datensatzes ausgefüllt haben, können Sie auf den Button **Los!** klicken, um den RSS-Feed zu speichern. Danach können Sie weitere Datensätze anlegen. Um einen bestehenden Datensatz zu löschen, müssen Sie lediglich den Feed-Namen leeren und auf **Los!** klicken.

Unterhalb jedes Datensatzes stellt das Plugin dar, wann der RSS-Feed zuletzt importiert wurde. Dieser Importvorgang muss in einem festen Intervall ausgeführt werden und wird gestartet, indem Sie die URL `http://www.example.com/serendipity/index.php?/plugin/aggrega` aufrufen. Dies können Sie entweder manuell über Cronjobs Ihres Servers lösen oder mittels des Serendipity Cronjob-Plugins (siehe Seite 288). Die URL können Sie auch manuell im Browser aufrufen, um den Importvorgang einmalig auszuführen. Ohne die Ausführung dieses Vorgangs können RSS-Feeds nicht regelmäßig importiert werden!

Anstatt jeden Datensatz eines zu importierenden RSS-Feeds mühsam von Hand einzupflegen, können Sie auch den Import einer *OPML-Datei* von der RSS-Aggregator-Oberfläche auslösen. Eine OPML-Datei kann von vielen RSS-Readern erstellt werden und enthält eine Liste abonnierter RSS-Feeds inklusive ihrer Homepage-Adresse und etwaiger Kategorisierung. Um diese OPML-Datei zu importieren, müssen Sie die Datei auf einen Server hochladen (beispielsweise Ihren eigenen) und die URL dieser Datei im Feld **OPML-Datei importieren** eintragen. Das Ankreuzfeld **Kategorien importieren** bestimmt, ob etwaige im RSS-Feed vorhandene Kategorisierungen übernommen werden sollen. Nicht vorhandene Kategorien, die in der OPML-Datei angegeben sind, legt das Plugin automatisch im Blog an. Über das Ankreuzfeld **Jeden Feed in seine eigene Kategorie einfügen** können Sie dafür sorgen, dass jeder von der OMPL-Datei importierte Feed seine eigene, gleichbenannte Kategorie erhält. Dies hilft, um später die Übersicht über die importierten RSS-Feeds zu behalten.

Im Gegenzug zum Import ermöglicht der RSS-Aggregator es auch, die von Ihnen angelegten RSS-Feeds komfortabel in einer OPML-Datei zu speichern (Export). Diese können Sie dann in einen RSS-Reader importieren. Ein Klick auf **Export OPML!** löst diesen Exportvorgang aus, die resultierende Datei können Sie aus Ihrem Browser heraus speichern.

Sobald Sie erstmalig den RSS-Aggregatorvorgang gestartet haben, werden alle Artikel im RSS-Feed eingelesen und in Ihr Blog importiert. Bei jedem späteren Vorgang werden nur noch neue Artikel importiert.

Standardmäßig wird jeder importierte Artikel in Ihrem Blog so eingetragen, als wäre er von Ihnen redaktionell erstellt. Aus urheberrechtlichen Gründen ist das jedoch nicht unproblematisch, daher sollten Sie darauf achten, den Originalautor und die URL des RSS-Feeds mit in die Darstellung des Blog-Eintrages einzubeziehen. Für diesen Zweck stellt das Plugin einige Smarty-Template-Variablen zur Verfügung, die Sie nach Belieben in die Datei `entries.tpl` Ihres gewählten Templates einsetzen können (siehe auch Seite 519). Die zur Verfügung stehenden Smarty-Variablen sind:

{`$entry.properties.ep_aggregator_feedname`}
: enthält den hinterlegten Namen des RSS-Feeds, aus dem der jeweilige Artikel stammt.

{`$entry.properties.ep_aggregator_feedurl`}

enthält die hinterlegte URL des Quell-RSS-Feeds.

{`$entry.properties.ep_aggregator_htmlurl`}
enthält die hinterlegte URL des Quell-Blogs.

{`$entry.properties.ep_aggregator_articleurl`}
enthält die hinterlegte URL des Artikels aus dem RSS-Feed.

{`$entry.properties.ep_aggregator_author`}
enthält den Originalautornamen des Artikels.

Empfehlenswert ist daher, wenn Sie in der Template-Datei `entries.tpl` ober- oder unterhalb der Ausgabe von {`$entry.body`} Folgendes einfügen:

```
<div class="serendipity_entry_body">
  {$entry.body}

  {if $entry.properties.ep_aggregator_feedurl}
  <p>Dieser Artikel wurde ursprünglich von
     {$entry.properties.ep_aggregator_author}
     im Blog <a href="{$entry.properties.ep_aggregator_articleurl}">
     {$entry.properties.ep_aggregator_feedname}</a>
     verfasst und hier aggregiert.
  </p>
  /if
</div>
```

Ein weiteres Beispiel, wie eine Änderung der `entries.tpl` aussehen kann, wird in der Datei `theme-patch.diff` des Aggregator-Plugins demonstriert.

Folgende Konfigurationsoptionen bietet das Plugin:

RSS Parser wählen
Das RSS-Aggregator-Plugin benötigt zum Importieren der RSS-Feeds eine Softwarebibliothek. Sie können auswählen, ob dafür die bei Serendipity mitgelieferte Bibliothek `Onyx` verwendet werden soll oder die vom Plugin bevorzugte Bibliothek `MagpieRSS`.

Onyx wird heutzutage nicht mehr weiterentwickelt, ist aber stabil und bei Serendipity mit allen notwendigen Funktionen zur Zeichensatzkonvertierung aufgewertet. Onyx ist zudem zur BSD-Lizenz kompatibel.

MagpieRSS unterstützt im Gegensatz zu Onyx nicht nur RSS-Formate, sondern auch das neuere Atom-Feedformat. MagpieRSS ist GPL-lizenziert.

Wenn Sie den Aggregator in einem nichtkommerziellen Projekt einsetzen, ist der Einsatz von MagpieRSS zu bevorzugen.

Artikel entfernen
Um die Artikeldatenbank des Blogs nicht unnötig immer größer werden zu lassen, kann das Aggregator-Plugin automatisch alle Artikel löschen, die älter als das hier festgelegte Limit (in Tagen) sind. Die Eingabe der Zahl 0 deaktiviert das automatische Entfernen.

Prüfsummen entfernen
Um zu erkennen, ob das Plugin bestimmte Artikel bereits importiert hat, verwaltet es intern eine Prüfsummenliste. Da diese Prüfsummen ebenfalls recht groß werden können, sollte man sie automatisch nach einer gewissen Zeit löschen (standardmäßig 90 Tage).

Sie sollten unbedingt darauf achten, dass Sie die Prüfsummen länger aufbewahren als die Artikel selbst – andernfalls könnte es zu doppelt importierten Artikeln kommen, da das Plugin möglicherweise nicht mehr zuordnen kann, ob ein Artikel bereits importiert wurde.

Aktualisierungen ignorieren
Wenn ein RSS-Artikel einmal importiert wurde, prüft Serendipity dennoch bei jedem neuen RSS-Importvorgang, ob sich ein derartiger Artikel möglicherweise verändert hat. Wenn der Original-Redakteur beispielsweise Rechtschreibfehler korrigiert hat, kann das Plugin so den Artikel auf Ihrem Blog ebenfalls korrigieren.

Wenn Sie die Option **Aktualisierungen ignorieren** aktivieren, wird das Plugin niemals bereits importierte Artikel nachträglich verändern.

Nicht mehr verkettete Einträge löschen
Wenn Sie die Option **Nicht mehr verkettete Einträge löschen** aktiviert haben und einen RSS-Feed aus dem Aggregator löschen, werden standardmäßig alle diesem Feed zugehörigen Artikel gelöscht.

Wenn Sie jedoch die alten Einträge aus historischen Gründen sichern wollen, sollten Sie diese Option deaktivieren.

Kommentare für diesen Eintrag zulassen
Da importierte RSS-Artikel üblicherweise nicht in einem aggregierenden Blog kommentiert werden (sondern nur im Original-Artikel), ist es sinnvoll, dass Kommentare zu derartigen Artikeln in Ihrem Blog nicht zugelassen sind.

Wenn Sie dennoch Kommentare zulassen wollen, müssen Sie die Option **Kommentare für diesen Eintrag zulassen** aktivieren.

Debugging-Output
Das Aggregator-Plugin kann bei Problemen mit dem Importvorgang detaillierte Statusmeldungen ausgeben. Um diese beim Importvorgang zu sehen, müssen Sie die Option **Debugging-Output** aktivieren.

Datenbanktabelle

Die Datenbanktabelle `serendipity_aggregator_feeds` enthält eine Liste der Feeds, die vom Aggregator gelesen werden sollen:

`feedid`
: enthält die fortlaufende ID eines Feeds.

`feedname`
: enthält den im Backend eingegebenen Namen des Feeds.

`feedurl`
: enthält die URL des Feeds.

`htmlurl`
: enthält die zugeörige Homepage der Seite.

`last_update`
: enthält das Datum der letzten Aktualisierung dieses Feeds.

`charset`
: enthält den Zeichensatz des Feeds.

In der Tabelle `serendipity_aggregator_md5` werden Hash-Werte gespeichert, die jeden Artikel mit einer eindeutigen Prüfsumme verbinden. Durch Abgleich dieser Datenbank kann beim Importieren der Feeds geprüft werden, ob ein Artikel bereits in der Datenbank vorliegt.

`entryid`
: enthält die ID des Blog-Artikels, für den die Prüfsumme erstellt wurde.

`md5`
: enthält den 32 Zeichen langen MD5-Hashcode des Artikeltexts.

`timestamp`
: enthält das Datum, an dem der MD5-Hash erstellt wurde.

In der Tabelle `serendipity_feedcat` wird gespeichert, welcher Blog-Kategorie (`categoryid`) der jeweilige Feed (`feedid`) zugeordnet sein soll.

6.3.23 POPfetcher serendipity_event_popfetcher

Ähnlich wie das *RSS Aggregator*-Plugin dient der *POPfetcher* dem Import von Artikeln in Ihr Blog.

Der *POPfetcher* kann automatisch einen E-Mail-Account mittels POP-Server abfragen und E-Mails zu dieser Adresse als Blog-Eintrag übernehmen. So können Sie unterwegs Artikel schreiben (*Moblogging*) oder auch bequem vom Handy/PDA ohne Zugriff auf das Serendipity-Backend das Blog mit Inhalten füllen.

Ähnlich wie das Aggregator-Plugin kann der POPfetcher sich systembedingt nicht selber aufrufen. Daher müssen Sie den E-Mail-Abruf entweder manuell ausführen oder über einen zentralen Cronjob. Auch das Serendipity Cronjob-Plugin (siehe Seite 288) unterstützt den POPfetcher zum regelmäßigen Abruf von E-Mail-Konten.

Der POPfetcher kann einen gezielten E-Mail-Account abrufen, der dem Plugin als *Empfängeradresse* dient. Obwohl das Plugin sich so konfigurieren lässt, dass es nur die E-Mails bestimmter Absender akzeptiert, sollten Sie die Empfängeradresse des Plugins streng geheim halten. Ansonsten könnte es möglicherweise fremden Personen gelingen, Artikel in Ihr Blog zu schleusen.

E-Mails können zudem Dateianhänge enthalten, die das Plugin automatisch in Artikeln und der Mediendatenbank einträgt.

Da der Importvorgang des POPfetchers automatisiert abläuft, lassen sich sämtliche Einstellungen ausschließlich über die Plugin-Konfiguration vornehmen:

Plugin-Methode, Name für externen Aufruf

Der POPfetcher kann auf zwei Arten dazu angewiesen werden, Ihr Postfach abzurufen: Zum einen die **interne** Methode. Hierbei müssen Sie als eingeloggter Redakteur auf den Menüpunkt **Einträge** → **POPfetcher** klicken, um den Abruf zu starten.

Wesentlich komfortabler ist jedoch die **externe** Methode. Hierbei rufen Sie eine URL auf, die Sie über die Option **Name für externen Aufruf** festgelegt haben. Standardmäßig wäre dies `http://www.example.com/serendipity/index.php?/plugin/popfetcher`. Die URL sollten Sie aus einer möglichst nicht zu ratenden Buchstabenfolge zusammensetzen, damit Ihre Blog-Besucher den Aufruf des POPfetchers nicht (böswillig) ausführen können. Das wäre zwar nicht weiter tragisch, da ein derart gestarteter Aufruf nur das von Ihnen festgelegte Postfach abruft, aber es könnte durchaus zu Performance-Problemen bei häufigem Aufruf kommen.

Die derart konfigurierte externe URL können Sie beispielsweise in einem Cronjob Ihres Servers automatisiert alle X Stunden aufrufen, beispielsweise mittels `wget 'http://www.example.c pity/index.php?/plugin/popfetcher'`.

Wenn Sie das Plugin über das Serendipity-Cronjob-Plugin ausführen, ist diese Einstellung irrelevant.

Autor

Die via E-Mail geschriebenen Artikel können vom Plugin nicht automatisch bestehenden Blog-Redakteuren zugeordnet werden. Daher müssen Sie über die Option **Autor** festlegen, welchem Blog-Redakteur die Artikel zugewiesen werden sollen. Das bedeutet, dass sämtliche via E-Mail eingelieferten Artikel immer demselben Autor gehören werden, egal von welcher E-Mail-Adresse diese stammen.

Andernfalls können Sie in dem Auswahlfeld den Wert **Lookup by email** wählen. Dann wird der Autor anhand seiner Absenderadresse zugewiesen, falls eine übereinstimmende Redakteursadresse in der Redakteursdatenbank vorliegt.

Mail Server, POP3 User, Passwort, POP3 port, APOP
Tragen Sie in diese Felder die Zugangsdaten zu Ihrem E-Mail-Postfach ein. Wenn Sie einen Server mit POP3 über SSL benutzen wollen, müssen Sie den POP3-Port 995 eintragen, und die PHP-Installation des Webservers muss das *openssl*-Modul aktiviert haben.

Wenn Ihr Mailserver APOP unterstützt, kann das Passwort bei aktivierter **APOP**-Option speziell verschlüsselt übertragen werden. Ohne APOP-Unterstützung wird ein E-Mail-Passwort im Klartext an den Server übertragen, was ein gewisses Sicherheitsrisiko darstellt.

E-Mail-Absender
Wenn der POPfetcher nur E-Mails von einem bestimmten Absender akzeptieren darf, können Sie diesen Absender im Feld **E-Mail Absender** eintragen. Hier müssen Sie die vollständige E-Mail-Adresse eintragen, wie sie im *From-Header* der E-Mail erscheint. Die meisten E-Mail-Programme geben solche Adressen in der Form `"Garvin Hicking" <mail@example.com>` ein; die Adresse muss daher auch in exakt dieser Notation im Eingabefeld eingetragen werden.

Kategorie
Die via E-Mail abgeholten Einträge können Sie einer festen Blog-Kategorie zuordnen. Dazu tragen Sie im Feld **Kategorie** die Kategorie-ID ein (siehe Seite 124).

Sie können auch pro E-Mail eine eigene Kategorie festlegen, indem Sie im Betreff der E-Mail den Namen (nicht die ID!) der gewünschten Kategorie in eckigen Klammern voranstellen. Mehrere Kategorienamen können Sie mittels Semikolon (`;`) voneinander trennen, um den Eintrag jeder aufgeführten Kategorie zuzuordnen.

Eine E-Mail mit dem Betreff `[Sport] FC Köln in Regionalliga abgestiegen` würde einen Blog-Artikel mit dem Betreff *FC Köln in Regionalliga abgestiegen* in der Kategorie *Sport* veröffentlichen.

Upload-Verzeichnis
Dateianhänge einer E-Mail können vom POPfetcher automatisch ausgegliedert und in einem Verzeichnis auf dem Webserver gespeichert werden. So können Sie beispielsweise Handyfotos leicht mittels E-Mail-Versand in Ihr Blog hochladen.

Das Zielverzeichnis für diese Dateien legen Sie mit **Upload Verzeichnis** fest. Als Stammverzeichnis dient hierbei das in der zentralen Serendipity-Konfiguration festgelegte Upload-Verzeichnis (`uploads`). Wenn Sie hier ein Unterverzeichnis vorgeben (`Urlaubsbilder/`), müssen Sie sicherstellen, dass das gewünschte Verzeichnis bereits existiert. Das Verzeichnis können Sie über den Menüpunkt **Mediendatenbank** → **Verzeichnisse verwalten** oder mittels FTP anlegen.

Blog

Nur wenn Sie die Option **Blog** aktivieren, wird der Inhaltstext einer E-Mail als Blog-Artikel angelegt. Wenn Sie die Option deaktivieren, werden nur Dateianhänge der E-Mail in der Mediendatenbank gespeichert.

Diese Option ist besonders dann sinnvoll, wenn Sie den POPfetcher nur dazu verwenden wollen, um Bilder mittels Ihres Handys automatisch in das Blog hochzuladen und sie erst später in einen Blog-Artikel einzufügen.

Use plaintext attachments as entry body

Einige E-Mail-Programme oder Handys verschicken die Texte einer E-Mail als Dateianhang. Damit diese Anhänge dekodiert und als Inhaltstext Ihrer Blog-Artikel verwendet werden können, müssen Sie die Option **Use plaintext attachments as entry body** aktivieren.

Bei deaktivierter Option wird ein derartiger Dateianhang genauso wie eine PDF- oder Bilddatei behandelt und nur ein Link zu der Mediendatenbank für diese Datei in den Blog-Artikel eingebunden.

First text attachment is entry body, the rest extended

Standardmäßig wird der POPfetcher alle Texte der E-Mail zusammenfassen und als Artikeltext für den Blog-Eintrag verwenden. Somit ist sämtlicher Text der E-Mail für den Besucher in der Blog-Übersichtsseite lesbar.

Wenn Sie jedoch längere E-Mails in den dafür vorgesehenen *erweiterten Eintrag* (siehe Seite 107) aufteilen wollen, können Sie die Option **First text attachment is entry body, the rest extended** aktivieren. Diese Option sorgt dafür, dass nur der erste Text-Dateianhang als Grundtext verwendet und alles Weitere im erweiterten Eintrag gespeichert wird.

Abgesehen von dieser Trennung durch Dateianhänge können Sie E-Mail-Texte auch mit einem manuellen Trennzeichen aufteilen, siehe die Option **Spezieller Text, der Text und erweiterten Eintrag einer E-Mail aufteilt**

Werbung entfernen

Der Versand von E-Mails über das Handy führt oft dazu, dass Mobilfunkbetreiber wie O_2, E-Plus und die Telekom Werbung in die E-Mail einfügen. Solche Werbung ist in Blog-Artikeln unerwünscht.

Die Option **Werbung entfernen** kann automatisch die Werbung aus Mails von T-Mobile und O_2 entfernen. Da die Art der Werbeeinbindung jedoch in Zukunft variieren könnte, ist diese Option möglicherweise nicht immer einsatzfähig.

Text nach speziellen Buchstaben abschneiden

Etwas wirksamer im Umgang mit zu entfernender Werbung ist die Option **Text nach speziellen Buchstaben abschneiden**. Tragen Sie hier eine beliebige Zeichenkette ein, die Sie auch später in Ihren E-Mails verwenden, und der POPfetcher wird alle Texte nach der konfigurierten Zeichenkette löschen.

Wenn Sie hier beispielsweise den Text *overandout* eintragen und Ihre Handy-E-Mail später ebenfalls mit *overandout* beenden, kann sämtliche Werbung, die der Mobilfunkbetreiber dahinter einfügt, einfach ignoriert werden.

Sie sollten hier nur Text eintragen, der sonst in Ihrer E-Mail nicht vorkommt, um Textverlust zu vermeiden.

Entferne alle HTML-Tags aus den Mails

Grundsätzlich können Sie auch HTML-E-Mails an den POPfetcher verschicken, der diese Formatierung auch ins Blog übernimmt. Da dies jedoch auch zu potenziellen Sicherheitslöchern durch JavaScript führen kann, sollten Sie die HTML-Tags mit dieser Option entfernen, wenn Sie die Inhalte einer E-Mail nicht völlig kontrollieren können.

Spezieller Text, der Text und erweiterten Eintrag einer E-Mail aufteilt

Wenn Sie eine längere E-Mail in Artikeltext und erweiterten Eintrag aufteilen wollen, können Sie ein beliebiges Trennzeichen formulieren. Sobald dieses im Text Ihrer E-Mail auftaucht, wird sämtlicher Folgetext in den *erweiterten Eintrag* des Blog-Artikels übernommen.

Achten Sie darauf, dass das hier festgelegte Wort einmalig in der E-Mail vorkommen sollte. Wenn Sie hier beispielsweise *xxx-TRENNER-xxx* festlegen, könnten Sie eine E-Mail wie diese schreiben:

```
Liebe Blog-Leser,

hier meldet sich eure Uschi vom Hauskatzenzüchterverein. Gerade
bin ich auf der Showbühne und schaue begeistert dem Auftritt der
Pussy Cat Dolls zu. Meinen ausführlichen Messebericht findet ihr,
wenn ihr auf "Weiterlesen" klickt!

xxx-TRENNER-xxx

Die schönste Katze der diesjährigen Züchtung, wenn nicht die
schönste Katze seit Bestehen unserer Hausmesse, war diesmal der
Kater "Giesbert" aus NRW ...
```

Define a string which indicates the text to capture

Wenn Ihre E-Mails besonders viel Werbung oder HTML-Dateianhänge enthalten, könnte der POPfetcher möglicherweise überfordert damit sein, alle Inhalte korrekt zu erkennen.

Als letzten Ausweg können Sie daher eine besondere Zeichenkette im Eingabefeld **Define a string which indicates the text to capture** festlegen. In Ihrer E-Mail können Sie dann diese Zeichenkette *vor* und *hinter* den zu bloggenden Text schreiben. Alles dazwischen wird vom POPfetcher als Text für den Blog-Artikel verwendet.

Auch hier sollten Sie sicherstellen, dass die Zeichenkette in Ihrer E-Mail einmalig ist.

Publizieren
Mit der Option **Publizieren** legen Sie fest, ob ein via POPfetcher eingestellter Artikel direkt veröffentlicht oder nur als Entwurf gespeichert wird.

Da aufgrund der vielen unterschiedlichen E-Mail-Formate nie sichergestellt werden kann, ob die spätere E-Mail exakt Ihren Vorstellungen entspricht, kann es unter Umständen besser sein, einen Artikel vor der Veröffentlichung im Blog durch einen Redakteur prüfen zu lassen.

Zwar verlieren Sie so die Möglichkeit, spontan unterwegs eine E-Mail online ins Blog zu stellen, aber Sie gewinnen so die absolute Kontrolle über die Inhalte Ihres Blogs.

Kommentare und Trackbacks dieses Eintrags werden moderiert,

Kommentare für diesen Eintrag zulassen Mit diesen beiden Optionen legen Sie fest, ob ein via POPfetcher veröffentlichter Artikel von Besuchern kommentiert werden darf und ob diese Kommentare standardmäßig moderiert werden sollen (siehe Kapitel 3 ab Seite 99).

Mail löschen
Wenn der POPfetcher eine E-Mail abgeholt hat, wird sie standardmäßig vom Server gelöscht, damit sie beim nächsten Importvorgang nicht erneut gelesen wird.

Zu Testzwecken ist es jedoch praktisch, unterschiedliche Plugin-Einstellungen mit derselben E-Mail auszuprobieren, um die optimalen Einstellungen zu finden. Während dieser Testphase sollten Sie E-Mails nicht vom Server löschen.

Timeout
Mit der Option **Timeout** legen Sie eine Zeitspanne in Sekunden fest, nach deren Ablauf der POPfetcher eine fehlgeschlagene Verbindung zum Mailserver abbrechen soll.

Das Plugin ist speziell auf die Verarbeitung von E-Mails der Provider O_2, T-Mobile, Sprint PCS, Cingular und Verizon optimiert. Dateianhänge mit den Endungen `pif, vbs, scr, bat, com, exe` werden vom POPfetcher ignoriert, da es sich dabei meist um E-Mail-Viren handelt. Diese Dateiliste ist in der Variable `$list_virus` in der Datei `serendipity_event_popfetcher.php` manuell erweiterbar.

Als Bilder erkennt das Plugin die Dateiendungen `gif, jpg, png, jpeg` (Variable `$list_imageext`) an, als Content-Type von Dateianhängen wird `jpg, jpeg, gif, png, x-png, jpeg` (Variable `$list_imagetype`) akzeptiert. Dateien mit der Endung `smil` (Variable `$list_ignore`) ignoriert das Plugin.

Um gegen neue Arten von E-Mail-Werbung oder problematische E-Mail-Formate vorzugehen, können Entwickler über die Variable `$debug_file` testweise eine E-Mail-Dumpdatei einlesen und deren Import implementieren. Sie können gerne im Serendipity-Forum nachfragen, wenn Sie Probleme mit dem Format Ihrer E-Mails haben.

6.3.24 Uses TinyMCE as WYSIWYG editor serendipity_event_tinymce

Serendipity wird standardmäßig mit dem WYSIWYG-Editor *HTMLArea* ausgeliefert. Dieser Editor läuft relativ problemlos mit aktuellen Browsern (Internet Explorer, Firefox, Safari, Konqueror), wird aber seit einiger Zeit nicht mehr aktiv weiterentwickelt.

An der Qualität verschiedener WYSIWYG-Editoren scheiden sich die Geister – grundsätzlich ist es ein beinahe unmögliches Ziel, vollständige WYSIWYG-Fähigkeiten browserübergreifend zu verwirklichen. Gerade zwischen Internet Explorer und Firefox unterscheiden sich die technischen Implementationen fast grundlegend, so dass ein überall gleich arbeitender WYSIWYG-Editor wohl noch lange ein Zukunftstraum bleiben wird.

Dennoch arbeiten viele Entwickler unbeirrt an diesem hehren Ziel und bringen häufige Updates ihrer Editoren heraus, die stetig stabiler werden. Das Kernproblem bei WYSIWYG-Editoren ist, dass diese häufig ungültigen HTML-Code produzieren, Sonderzeichen schlucken oder beim Verschieben von Bildern Abstürze hervorrufen. Davon bleibt auch der interne Serendipity-Editor leider nicht verschont.

Serendipity geht daher grundsätzlich den Weg, dass alternative WYSIWYG-Editoren eingebunden werden können. Dazu zählen TinyMCE[29], FCKEditor[30] und Xinha[31]. Für alle drei Editoren sind externe Plugins verfügbar.

Das schwierigste Unterfangen bei der Verwendung externer WYSIWYG-Editor-Plugins bei Serendipity ist, dass einigen die Unterstützung der Mediendatenbank fehlt oder die Integration in weitere Plugins (wie den HTML-Klotz oder Statische Seiten) nicht vorhanden ist. TinyMCE ist hier am weitesten fortgeschritten und wird daher an dieser Stelle exemplarisch beschrieben. Die Verwendung der anderen WYSIWYG-Editor-Plugins ist jedoch im Grundsatz gleich.

Um einen externen WYSIWYG-Editor wie TinyMCE verwenden zu können, müssen Sie drei Dinge durchführen: Erstens müssen Sie in Ihren persönlichen Einstellungen die Verwendung des WYSIWYG-Editors aktivieren und zweitens das Serendipity-Plugin installieren. Als Letztes müssen Sie den jeweiligen WYSIWYG-Editor manuell herunterladen und in das Plugin-Verzeichnis kopieren. Die WYSIWYG-Editoren selbst sind *nicht* Bestandteil des Serendipity-Plugins, da sie sich zum einen zu häufig ändern und zum anderen aufgrund ihrer Dateigröße den Rahmen eines Serendipity-Plugins sprengen würden.

Sobald Sie das TinyMCE-Plugin heruntergeladen haben, ist dieses im Verzeichnis `/plugins/serendipity_event_tiny` vorhanden. Besuchen Sie nun die TinyMCE-Downloadseite.[32] Laden Sie dort die aktuelle Version von TinyMCE herunter. Entpacken Sie danach die ZIP-Datei und laden Sie das entstandene Verzeichnis `tinymce` in Ihr Plugin-Verzeichnis auf dem Serendipity-Server.

Danach sollten Sie eine Datei wie `/plugins/serendipity_event_tinymce/tinymce/jscripts/tiny_mce/ti`

[29] `http://tinymce.moxiecode.com/`
[30] `http://www.fckeditor.net/`
[31] `http://xinha.webfactional.com`
[32] `http://tinymce.moxiecode.com/download.php`

besitzen. Lassen Sie sich von der leider tief verschachtelten Verzeichnisstruktur nicht irritieren.

Abgesehen von dem Basispaket benötigen Sie noch den TinyMCE compressor PHP. Dieser befindet sich ebenfalls auf der TinyMCE-Downloadseite. Auch diesen müssen Sie herunterladen und entpacken. Die beiden Dateien `tiny_mce_gzip.js` und `tiny_mce_gzip.php` laden Sie nun auf Ihren Serendipity-Server in das Verzeichnis `/plugins/serendipity_event_tinymce/tinymce/jscripts/tiny_mce` hoch.

Je nachdem, welche TinyMCE-Sprachversion Sie benutzen wollen, müssen Sie eventuell noch weitere Dateien herunterladen. Optional können Sie das TinyMCE-Tool namens *iManager* installieren, das eine eigenständige Mediendatenbank (inklusive Bildbearbeitung) ermöglicht. Die hierfür notwendigen Schritte werden in der Konfigurationsoberfläche des Serendipity-TinyMCE-Plugins aufgeführt.

Nachdem nun alle notwendigen Dateien erfolgreich hochgeladen wurden, können Sie TinyMCE das erste Mal benutzen. Rufen Sie dazu die bekannte Oberfläche zum Erstellen eines Eintrages auf.

Das TinyMCE-Plugin ist seinerseits stark modular aufgebaut. Alle Bestandteile des Editors können nach Belieben zusammengewürfelt werden. Viele der Einstellungsmöglichkeiten werden Ihnen über die Konfigurationsoptionen des Plugins angeboten:

iManager Tool benutzen
: Wenn Sie den iManager benutzen wollen, müssen Sie diesen nach der eigenständigen Installation über den Menüpunkt **iManager Tool benutzen** aktivieren.

Zusätzliche TinyMCE Plugins, Knopfleiste 1, 2 und 3
: In dieser Eingabebox können Sie die TinyMCE-Plugins eintragen, die Sie verwenden wollen. Auch die Reihenfolge der Buttons im TinyMCE-Plugin richtet sich nach dieser Eingabe.

  Wenn Sie beispielsweise mit dem TinyMCE-Plugin keine Änderung der Textrichtung vornehmen wollen, können Sie das Plugin **directionality** einfach aus der Liste entfernen. Eine vollständige Liste, welches Plugin welcher Funktionalität dient, finden Sie auf der Homepage von TinyMCE.

  Das Pipe-Sonderzeichen (`|`) können Sie in der Definition der Knopfleisten dazu benutzen, um einen Zeilenumbruch zu erreichen.

Relative URLs erzeugen
: TinyMCE versucht standardmäßig alle von Ihnen eingegebenen Hyperlinks in relative URLs umzuwandeln. Aus `http://www.example.com/serendipity/index.php` wird so `/serendipity/index.php`. Grundsätzlich hat dies den Vorteil, dass relative URLs beim Umzug Ihres Blogs auf einen anderen Server nicht geändert werden müssen.

  Bei der Benutzung der Mediendatenbank von Serendipity sind solche relativen URLs jedoch eher problematisch. Daher ist es eher zu empfehlen, die Option **Relative URLs**

erzeugen zu deaktivieren, um potenziellen Problemen bei der Einbindung von Podcasts, RSS-Feeds und Ähnlichem zu entgehen.

HTML verifizieren/korrigieren
TinyMCE kann ungültigen HTML-Code Ihrer Artikel automatisch korrigieren. Theoretisch ist dies eine tolle Funktionalität, praktisch scheitert die Korrektur jedoch oft an dem, was TinyMCE als gültig und ungültig erkennt. HTML-Tags, die TinyMCE nicht anerkennt, werden so kurzerhand entfernt.

Gerade bei der Einbindung von YouTube-Links oder OBJECT/EMBED-HTML-Tags kann dies dazu führen, dass gültiger HTML-Code von TinyMCE einfach stillschweigend entfernt wird. Noch problematischer wird das Vorgehen dadurch, dass abhängig von Ihrem Browser (Microsoft Internet Explorer oder Firefox) unterschiedliche HTML-Tags entfernt werden.

Auch hier empfehlen wir daher, die Option **HTML verifizieren/korrigieren** zu deaktivieren, wenn Sie stellenweise selbst HTML-Code eingeben.

Code säubern
Im Gegensatz zur Option **HTML verifizieren**, die sich nur um das Entfernen ungültiger HTML-Tags kümmert, kann die Option **Code säubern** TinyMCE dazu anweisen, eventuell ungültige HTML-Konstrukte zu beheben. Darunter fallen beispielsweise falsche Sonderzeichen, fehlende schließende Tags oder anderweitige ungültige Verschachtelung.

Dieser Säuberung kann man normalerweise getrost vertrauen, da sie tatsächlich der Gültigkeit Ihrer Artikel dienlich ist. Lediglich wenn Sie selber absolute Kontrolle über den HTML-Code haben wollen, sollten Sie diese Option deaktivieren. Dann jedoch wäre es generell eher zu empfehlen, vollständig auf WYSIWYG zu verzichten, um die größtmögliche Flexibilität zu erlangen.

Mozilla Rechtschreibhilfe
Seit einigen Versionen ermöglicht der Mozilla Firefox Browser die Rechtschreibprüfung Ihrer Eingaben in beliebigen Textfeldern. Da TinyMCE über eine eigene (optionale) Rechtschreibprüfung verfügt, deaktiviert TinyMCE standardmäßig die Prüfung der Mozilla Rechtschreibhilfe. Wenn Sie diese jedoch gerne verwenden möchten, können Sie die Option **Mozilla Rechtschreibhilfe** aktivieren.

Relativer HTTP Pfad des Plugins
Damit das Serendipity-Plugin den Quellcode des TinyMCE-Editors korrekt einbinden kann, benötigt es den relativen Pfad zum TinyMCE-Verzeichnis ab dem Serendipity-Stammverzeichnis.

Bei normalen Installationen können Sie hier stets den Standardwert `plugins/serendipity_event_tinymce/` beibehalten. Nur wenn Sie symbolische Links oder Ähnliches verwenden, ist eine Anpassung dieser Konfigurationsoption angebracht.

6.3.25 Einträge via XML-RPC erstellen serendipity_event_xmlrpc

Inzwischen haben Sie bereits gelernt, Artikel in Serendipity über das Backend, eine E-Mail und das POPfetcher-Plugin und auch über den Import von RSS-Feeds zu erstellen.

Wem das noch nicht genug ist, der kann sich einer weiteren beliebten Methode bedienen: der XML-RPC API.

Was dank Akronymen erstmal kryptisch klingt, ist recht simpel: Über eine technische Spezifikation kann eine beliebige Software auf Ihrem Computer (dem Client) mit Serendipity (dem Server) interagieren.

So können Sie einen Artikel mit der Software (dem sogenannten *Blog Client*) erstellen und dann an das Serendipity-Blog übermitteln. Durch derartige Software müssen Sie keinen Internet-Browser mehr benutzen.

Die Vorteile sind: Sie können Artikel auch offline erstellen und formatieren, können diese lokal (wie ein Office-Dokument) speichern und oft mit leistungsfähigeren WYSIWYG-Editoren, als es im Browser möglich ist, ein komplexeres Layout umsetzen. Blog-Editoren sind häufig optisch sehr simpel gehalten und sind somit für manche Zielgruppen zugänglicher als die Bedienung der browserbasierten Oberfläche Serendipitys.

Abhängig von Ihrem Betriebssystem gibt es mehrere Editoren: Blogdesk[33], BlogJet[34], Ecto[35] oder auch w.bloggar[36].

Als technische Spezifikation (die API) unterstützen viele der erwähnten Clients unterschiedliche Methoden. Auch wenn Serendipity nicht in jeder Software aufgeführt wird, können Sie es fast immer mit der manuellen Konfiguration der jeweiligen Software ans Laufen kriegen.

Das Serendipity-Plugin unterstützt sowohl die MoveableType-API (MT/metaWeblog) als auch die ältere Blogger API. Bei der manuellen Konfiguration zum Betrieb Ihres gewählten Blog-Clients müssen Sie meist folgende Werte angeben: Servername (z. B. `www.example.com`), Port (meistens 80) und letztlich die URL oder auch etwas wie *Page* oder *API-Endpoint*. Dort tragen Sie den HTTP-Pfad zu der Serendipity-Datei `serendipity_xmlrpc.php` ein – also z. B. `/serendipity/serendipity_xmlrpc.php`. Diese PHP-Datei dient der Kommunikation Ihres Blog-Client mit dem Serendipity-Plugin.

Oft können Blog-Clients auch automatisch anhand Ihrer Blog-URL erkennen, welche Zugangsparameter notwendig sind. Damit Sie sich im jeweiligen Blog-Client in Ihr Blog einloggen können, müssen Sie zudem Ihren Benutzernamen und das Passwort eingeben, mit dem Sie sich auch im Serendipity-Backend einloggen.

Im Blog-Client können Sie nun Einträge veröffentlichen, Kategorien verwalten und auch Bilder anhängen. Leider können wir aufgrund der Vielfalt an Blog-Clients hier keine gezielteren

[33] `http://www.blogdesk.org/`
[34] `http://www.codingrobots.com/blogjet/`
[35] `http://ecto.kung-foo.tv`
[36] `http://wbloggar.com`

Beschreibungen geben. Aber dies ist meist auch nicht notwendig, da die Hilfen zu den Blog-Clients oft sehr ausführlich sind und man Ihnen in deren Support-Foren gerne weiterhilft.

Bei etwaigen Problemen in der Kommunikation zwischen Client und Server können Sie über die Datei `plugins/serendipity_event_xmlrpc/serendipity_xmlrpc.inc.php` die Variable `$debug_xmlrpc` auf den Wert `2` setzen, um ein Debugging-Protokoll zu schreiben. In dieser Protokolldatei können Sie gegebenenfalls bereits selbständig Fehlermeldungen entdecken. Ansonsten hilft man Ihnen gerne im Serendipity-Forum.

Zuletzt seien jedoch auch die Nachteile von Blog-Clients erwähnt. Der größte Nachteil liegt darin, dass die universelle Schnittstelle darauf verzichten muss, erweiterte Optionen von Blogs anzubieten. Viele Serendipity-Plugins (beispielsweise das Tagging-Plugin oder die Freien Eigenschaften von Einträgen) können Zusatzfelder für einen Eintrag verwalten, die Sie aber über einen Blog-Client nicht eintragen können. Sie verlieren also einen Großteil an Flexibilität und müssen so möglicherweise nach wie vor das Serendipity-Backend aufsuchen.

Ein weiterer Nachteil ist die große Vielfalt an APIs und der Interpretationsspielraum der zahlreichen Blog-Clients, die unterschiedliche Funktionsaufrufe teilweise ganz unterschiedlich ansprechen.

Sie sollten sich daher bereits im Voraus überlegen, ob möglicherweise der Blog-Client nur für spezielle Redakteure eingesetzt werden soll. Natürlich hindert Sie nichts daran, den Blog-Client und das Serendipity-Backend fallweise einzusetzen. Ausführliche Artikel können Sie beispielsweise im Blog-Client vorschreiben und dann später erweiterte Optionen über das Serendipity-Backend vornehmen.

6.3.26 Easy Podcast serendipity_event_podcast

Mit zunehmender Bandbreite der Internetanschlüsse sind neben bebilderten Blog-Artikeln auch ganz andere multimediale Präsentationen möglich. Sehr populär sind dank der Verbreitung des Apple iPod die sogenannten *Podcasts* geworden. Dieses Kunstwort bezeichnet die Ausstrahlung (*cast*) von Audioaufnahmen zu den i*Pods*. Blogger können so ihre eigenen Hörspiele oder einfach nur Sprachaufzeichnungen als gewöhnliche MP3-Datei aufnehmen und in ihr Blog stellen. Von dort kann sie automatisch durch die iTunes-Software (und mittlerweile auch durch viele andere RSS-Reader) heruntergeladen und auf den mobilen MP3-Player übertragen werden.

Die Idee ist einfach: Ähnlich wie ein Blog-Leser regelmäßig Ihre Blog-Artikel durch RSS-Anwendungen beziehen kann, um diese Artikel zu lesen, soll er unterwegs Ihre Neuigkeiten auch über seinen MP3-Player hören können.

Innerhalb kurzer Zeit hat sich dank der gelangweilten Berufspendler so ein großer Absatzmarkt für persönliche Hörspiel-Blogs gebildet. Da iTunes und iPods leicht bedienbar sind, ist diese Verbreitungsweise auch für kommerzielle Betreiber recht interessant geworden. Sogar Tageszeitungen und TV-Magazine geben mittlerweile regelmäßig Podcasts heraus, und große Portale wie `http://www.podcast.de` führen Übersichten über Audio-Blogs.

Durch den Erfolg von YouTube ist auch Video-Blogging populärer geworden, wenngleich aufgrund der geringen Verbreitung von portablen Videoplayern deren Durchbruch im Blog-Sektor noch auf sich warten lässt.

Technisch gesehen ist der Vorgang des Podcastings relativ simpel: Sie müssen lediglich eine MP3-Audiodatei erstellen, diese in den RSS-Feed einbinden, und schon können Ihre Leser die Dateien automatisch beziehen. Das Serendipity-Plugin **Easy Podcast** macht diesen Vorgang für Sie recht einfach, da es die Einbindung in den RSS-Feed selbständig übernimmt.

Sobald Sie das Plugin installiert haben, können Sie eine MP3-Audiodatei einfach in die Serendipity-Mediendatenbank hochladen und in einen Blog-Artikel einbinden. Wenn Sie einen derartigen Blog-Artikel nun speichern, erkennt das Plugin automatisch angehängte MP3-Dateien, bindet einen Audioplayer in die Darstellung ein und fügt die MP3-Datei als Podcast in den RSS-Feed ein (sogenannte *Enclosure*).

Ihre Besucher können nun über Ihr Blog die MP3-Datei anhören. Abgesehen von Audiodateien unterstützt das Plugin auch noch weitere Formate, die jeweils abhängig vom Dateityp mit unterschiedlichen Browser-Applets eingebunden werden können: QuickTime (`3gp, mov, mp3, mqv, qt`),
Windows Media Player (`avi, mpg, mpeg, wmv`), Flash (`swf`), Audio (`mp3, ogg, m3u, pls, m4b`).

Grundsätzlich funktioniert das Podcast-Plugin dabei so, dass es Ihren Blog-Artikel nach Links zu allen unterstützten Dateiformaten durchgeht und abhängig von der Konfiguration des Plugins den benötigten Player einbindet. Zu jedem unterstützten Dateiformat wird zudem auch das notwendige Enclosure-Element in dem RSS-Feed hinzugefügt.

Das Podcast-Plugin bietet mehrere Konfigurationsoptionen an:

Player anzeigen
: Nur wenn Sie die Option **Player anzeigen** aktivieren, wird das Plugin automatisch Hyperlinks zu unterstützten Dateitypen so abändern, dass der entsprechende Audio-/Video-Player anstelle des Links eingebunden wird.

Player Größe anpassen, Breite, Höhe
: Wenn Sie diese Option aktivieren, versucht das Plugin die Bildgröße einer Videodatei herauszufinden. Die Breite und Höhe des eingebetteten Players wird dann an diese automatisch ermittelte Bildgröße angepasst. Ohne diese Größenanpassung wird der Player stets in der festgelegten Höhe und Breite (standardmäßig 200 x 200 Pixel) angezeigt.

  Um die Bildgröße eines Videos herauszufinden, muss die `getid3`-Bibliothek installiert sein.[37] Dieser Automatismus kann möglicherweise die Geschwindigkeit des Plugins spürbar verlangsamen, daher ist eine feste Player-Größe unter Umständen die bessere Wahl.

Ausrichtung
: Über das Ausklappfeld **Ausrichtung** legen Sie fest, wie der Player mit der Video- oder Audiodatei ausgerichtet werden soll (**links, rechts, zentriert, ohne Ausrichtung**).

Nur die erste Mediendatei an den Feed Eintrag hängen
: Laut Spezifikation unterstützt ein RSS-Feed nur eine einzelne Mediendatei pro Artikel. Daher können Sie diese Option aktivieren, um auch beim Anhängen mehrerer Mediendateien nur die erste davon in den Feed einzubinden.

Erweiterte Artikel-Attribute
: Hier tragen Sie eine Liste der *freien Artikeleigenschaften* ein, falls Sie das Plugin *Erweiterte Eigenschaften für Artikel* verwenden, um zusätzliche Podcasts oder Videocasts einzubinden (siehe unten).

Einbettung der Medien aus erweiterten Artikel Attributen
: In dem Ausklappfeld **Einbettung der Medien aus erweiterten Artikel Attributen** können Sie festlegen, ob ein Player mit der entsprechenden Datei oberhalb oder unterhalb des (erweiterten) Artikels angezeigt werden soll.

  Wenn Sie die Platzierung über die Template-Datei `entries.tpl` (siehe unten) vornehmen möchten, können Sie die Option **Nicht in den Artikel einfügen** beibehalten.

Quicktime, WindowsMediaPlayer, Flash, Audio Erweiterungen
: In diesen Feldern können Sie eintragen, welche Dateiendungen für welchen Player benutzt werden sollen.

  Der HTML OBJECT/EMBED-Code für die jeweiligen Player befindet sich in der Datei `serendipity_event_podcast.php` in den Konstanten `PLUGIN_PODCAST_QUICKTIMEPLAYER`,

[37] Diese können Sie von `http://www.getid3.org` herunterladen und in ein Unterverzeichnis namens `getid3` in das Serendipity-Verzeichnis `bundled-libs` entpacken.

PLUGIN_PODCAST_WMPLAYER, PLUGIN_PODCAST_FLASHPLAYER und PLUGIN_PODCAST_MP-3PLAYER.

Wenn Sie einen eigenen HTML-Player (z. B. für FLV-Dateien) einbinden wollen, können Sie diese Konstanten in der PHP-Datei anpassen.

Caching

Wenn Sie die Option **Caching** aktivieren, wird das Plugin zur automatischen Player-Erstellung die jeweiligen Audio-/Videodateien nur einmalig überprüfen und deren Metadaten zwischenspeichern.

Bei deaktiviertem Caching wird die Geschwindigkeit möglicherweise stark eingeschränkt, da die großen Videodateien jedesmal erneut geprüft werden.

Abgesehen von der automatischen Player-Einbindung bietet das Plugin eine Integration mit dem Plugin *Erweiterte Eigenschaften für Artikel* an. Mit Hilfe der *Freien Eigenschaften* können Sie einem Artikel ein Feld namens `Podcast` hinzufügen, das Sie später in dem Smarty-Template `entries.tpl` an gewünschter Stelle im Layout anzeigen können. So wird eine Audio- oder Videodatei immer an einer festen Stelle eingebunden, und ein Redakteur muss diese nicht innerhalb des Artikeltextes platzieren. Konkret funktioniert dies so:

1. Über die Konfiguration des Plugins *Erweiterte Eigenschaften für Artikel* fügen Sie den Feldnamen `Podcast` ein.

2. Wenn Sie nun einen neuen Artikel erstellen, sehen Sie in dem Abschnitt **Erweiterte Optionen** ein neues Eingabefeld namens **Podcast**. Dort können Sie entweder eine URL zu der MP3-Datei eingeben oder mittels des Buttons **Mediendatenbank** eine Datei aus der Mediendatenbank auswählen.

3. Nach dem Speichern eines Artikels ist nun ein Datenbankfeld mit der URL zu der Podcast-Datei angelegt worden. Diese Variable möchten Sie nun an einer festen Stelle des Artikel-Layouts ausgeben, beispielsweise immer am Ende des Artikels.

4. Dazu bearbeiten Sie die Datei `entries.tpl` Ihres Templates und fügen beispielsweise unterhalb von {`$entry.body`} Folgendes ein:

```
<div class="serendipity_entry_body">
  $entry.body

  if $entry.properties.ep_Podcast
  <p>MP3-Podcast:
    <object width="300" height="42">
    <param name="src" value="$entry.properties.ep_Podcast">
    <param name="autoplay" value="true">
    <param name="controller" value="true">
    <embed src="$entry.properties.ep_Podcast" autostart="true"
           loop="false"  width="300" height="42" controller="true">
```

```
    </embed>
    </object>
  </p>
  /if
</div>
```

5. Dieser HTML-Code sorgt dafür, dass die von Ihnen eingetragene Audiodatei mittels Browser-Applets abspielbar wird. Zusätzliche Attribute wie `width="300"` geben die Breite der Zeitleiste für die Audiowiedergabe an.

Diese Variante der Einbindung erfordert zwar mehr Anpassung, garantiert dafür aber auch ein einheitliches Layout. So können Sie auch eigene Videoplayer anstelle der QuickTime/WindowsMedia und anderen Player einbinden.

6.4 Ereignis-Plugins für Bilder

6.4.1 Lightbox/Thickbox/Graybox serendipity_event_lightbox

In Blog-Artikeln von Serendipity können Sie dank der Mediendatenbank relativ einfach ein Bild einfügen (siehe Seite 107). Ein solches Bild wird entweder in der vollen Größe oder als kleines Vorschaubild eingebettet, und ein Klick auf das Bild öffnet es meist in einem neuen Browserfenster.

Diese Art der Einbindung ist relativ unspektakulär. Daher wurden in letzter Zeit zahlreiche JavaScripts entwickelt, die Bildervorschauen und -galerien mit hübschen Überblendungseffekten einbinden lassen. Ein Klick auf ein Bild dunkelt dann den Hintergrund ab, das Bild öffnet sich in einem eigenständigen Bereich und passt sich in der Größe automatisch an. Auch ein Hin- und Herblättern zwischen mehreren zusammengehörigen Bildern ist so komfortabel möglich.

Mittlerweile gibt es viele JavaScripts, die diese Technik einsetzen: Lightbox[38], Lightbox plus[39], Thickbox[40] und Greybox[41]. Alle diese Bibliotheken bieten unterschiedliche Features und eine unterschiedliche Syntax. Sie sollten sich daher alle Varianten und Demos ansehen, um sich auf die für Sie beste zu beschränken. Da dies großteils Geschmackssache ist, kann man leider keines der Scripts pauschal empfehlen.

Welche der Bibliotheken Sie einsetzen, ist erst einmal nicht weiter wichtig. Das Serendipity-Plugin *Lightbox/Thickbox/Graybox* unterstützt alle aufgeführten Bibliotheken und liefert alle drei benötigten Dateien mit. Sie können sich also auch später für den Einsatz eines anderen Scripts entscheiden. Jedoch können Sie immer nur eine der Bibliotheken benutzen, das Plugin lässt sich nicht mehrfach installieren.

In der Plugin-Konfiguration müssen Sie daher lediglich die Bibliothek auswählen und dann den HTTP-Pfad zu dem Plugin-Verzeichnis Ihres Blogs eintragen. Standardmäßig entspricht dies dem Serendipity-Stammpfad zzgl. `/plugins`. Im Beispiel des Buches wäre es also `/serendipity/plugins/`. Den voreingestellten Pfad können Sie daher in den meisten Fällen beibehalten.

Sobald das Plugin aktiviert ist, wird es sich automatisch um die Links zu Ihren Bildern kümmern. Jedesmal, wenn ein Blog-Artikel dargestellt wird, sucht das Plugin nach vorhandenen `<a href...>`-Hyperlinks und fügt dort den Code ein, den die jeweilige Bibliothek benötigt, um beim Klick auf den Link direkt ein Bild anzuzeigen. Bei Lightbox (und dessen Varianten) wird einem Link das Attribut `<a rel="lightbox" href...>` eingesetzt, für Thickbox ist dies `<a rel="thickbox" href...>`, und Graybox benutzt die Syntax `<a rel="gb_image[]" href...>`. Natürlich können Sie auch manuell in den HTML-Code Ihrer Blog-Artikel die entsprechenden weiteren Möglichkeiten der Box-Bibliotheken einsetzen – lesen Sie dazu bitte die Dokumentation der jeweiligen Bibliothek.

Die Darstellung von großen Bildern in einem eingebetteten Popup kann nur funktionieren, wenn Sie in Ihrem Beitrag einen Link zu einem Bild eingebunden haben. Wenn Sie ein Bild aus der Mediendatenbank einfügen und davon nur das Vorschaubild ohne weitere Verlinkung auswählen, können die Bibliotheken nicht aktiv werden, da ihnen die notwendigen HTML-Daten fehlen.

6.4.2 Bildergalerie serendipity_event_usergallery

Mit dem Plugin *Bildergalerie* können Sie Bilder und Dateien aus Ihrer Mediendatenbank präsentieren. Dabei kann das Plugin so konfiguriert werden, dass nur spezielle Unterordner Ihrer Mediendatenbank eingesehen werden können. Die Darstellung der Galerie kann über Templates gesteuert werden, zahlreiche Konfigurationsoptionen ermöglichen eine individu-

[38] `http://www.huddletogether.com/projects/lightbox2/`
[39] `http://serennz.sakura.ne.jp/toybox/lightbox/`
[40] `http://jquery.com/demo/thickbox/`
[41] `http://orangoo.com/labs/GreyBox/`

elle Anpassung der Galerie.

Abgesehen von der Galerie-Übersicht bietet das Plugin auch einen eigenen RSS-Feed an, der Ihre aktuellsten Bilder (als Vorschaubild) enthält. Diesen können Sie über

```
http://www.example.com/rss.php?version=2.0&gallery=true&limit=A&picdir=B
&feed_width=C&head_title=D
```

Die einzelnen URL-Variablen sind:

`version`
: Legt die RSS-Feed-Version fest (siehe auch Seite 85).

`gallery`
: Wird benötigt, damit das Plugin seine Ausgaben einbinden kann.

`limit` (A, optional)
: Legt fest, wie viele Bilder im RSS-Feed enthalten sein sollen (standardmäßig 15).

`picdir` (B, optional)
: Legt den Pfad fest, aus dem die Bilder ausgelesen werden sollen. Standardmäßig werden alle Bilder aus allen Unterverzeichnissen herangezogen.

`feed_width` (C, optional)
: Gibt die gewünschte Größe der Vorschaubilder an. Standardmäßig wird die in den Konfigurationsoptionen festgelegte Zahl verwendet. Falls die URL-Option angegeben wird, hat diese jedoch Vorrang.

`hide_title` (D, optional)
: Wenn dieser Parameter angegeben wird, enthält der RSS-Feed keine Dateinamen oder Titel der Bilder, sondern lediglich die Bilddatei.

Um also die aktuellsten zehn Bilder des Verzeichnisses `Messe` in einer Größe von maximal 110 Pixeln Breite darzustellen, würden Sie folgende URL verwenden:

```
http://www.example.com/rss.php?version=2.0&gallery=true&limit=10&picdir=
Messe&feed_width=110
```

Die Darstellung dieses Feeds erfolgt über die üblichen Serendipity `feed*.tpl`-Dateien. Mit der auf Seite 490 vorgestellten Methodik können Sie daher auch ein ganz eigenes Feed-Template für den Galerie-Feed anlegen, indem Sie `http://www.example.com/rss.php?version=gallery...` verwenden und eine Template-Datei `feed_gallery.tpl` anlegen.

Folgende Konfigurationsoptionen bietet das Plugin:

Display name
: Legt den Titel der Galerie fest.

Anzahl der Spalten
Legt die Anzahl der Spalten für die Bildübersicht fest.

Subpage name for gallery view
Legt den *URL-Titel der Seite* (siehe Seite 238) fest, der benötigt wird, um die Galerie mittels einer speziellen URL aufrufen zu können.

Mache diese Seite zur Startseite für Serendipity
Mit Aktivierung dieser Option wird die Bildergalerie als Startseite Ihres Blogs dienen. In diesem Fall sollten Sie dafür sorgen, dass das Galerie-Ereignis-Plugin als eines der ersten Ereignis-Plugins positioniert wird, da andernfalls Konflikte mit anderen Plugins bei der Darstellung der Startseite auftreten können.

Permalink für die Anzeige der Galerie
Legt den *Permalink* fest, mit dem Sie die Galerie aufrufen können (siehe auch Seite 238).

Choose the gallery style
Mit diesem Ausklappfeld legen Sie fest, ob das Plugin zur Darstellung der Galerie eigene Template-Dateien mit eigener Formatierung (Wert `Thumbnail page`) oder die Standard-Mediendatenbankansicht von Serendipity (Wert `Media library`) verwenden soll. Nur bei der Option `Thumbnail page` sind die meisten der folgenden Konfigurationsoptionen verfügbar.

Pick a default directory
Hier können Sie ein Stammverzeichnis auswählen, das für die Galerie herangezogen wird. Besucher können nur auf Unterverzeichnisse des gewählten Verzeichnisses zugreifen.

Show a directory listing
Legt fest, ob eine Liste aller verfügbaren Verzeichnisse in der Galerie angezeigt werden soll.

Bilder pro Seite
Legt die Anzahl der Bilder pro Seite fest. Wenn Sie hier `0` eingeben, zeigt das Plugin alle verfügbaren Bilder auf einer einzigen Seite an. Andernfalls wird eine Möglichkeit zum Vor- und Zurückblättern für weitere Bilder eingebunden.

Reihenfolge der Bilder
Legt fest, in welcher Sortierungsreihenfolge (nach Dateiname oder Einstellungsdatum) die Bilder dargestellt werden.

Einleitungstext
Hier können Sie einen beliebigen (HTML)-Text eintragen, der auf jeder Seite der Galerie für Ihre Besucher angezeigt wird.

Display Single Image
: Mit dieser Option bestimmen Sie, ob beim Klick auf ein einzelnes Bild dieses als **Adaptive pop-up** (eigenes, korrekt skaliertes Fenster) oder innerhalb des Blog-Layouts (**Scaled to fit**) dargestellt wird.

Output images strictly
: Falls diese Option aktiviert wird, zeigt die Galerie ausschließlich Bilder des aktuellen Verzeichnisses an und ignoriert alle etwaigen Unterverzeichnisse.

Feste Bildgröße
: Legt die Standardbreite eines Bildes bei der Darstellung der Galerie fest. Standardmäßig wird die Voransichtsgröße der globalen Serendipity-Konfiguration übernommen.

Max. image width in page
: Legt die maximale Bildbreite fest, die ein Bild bei der Darstellungsart **Scaled to fit** einnehmen darf.

RSS-Feed image dimensions
: Legt fest, wie groß ein Vorschaubild bei der Einbindung innerhalb des RSS-Feeds sein soll. Diese Größe kann durch den URL-Parameter `feed_width` vom Benutzer übergangen werden.

Nur verlinkte Bilder im RSS-Feed
: Falls Sie diese Option aktivieren, enthält der RSS-Feed nur die Bilder, die Sie innerhalb Ihrer Blog-Beiträge auch tatsächlich verlinkt haben. Andernfalls wird stur das Verzeichnis der Mediendatenbank ausgelesen, und alle darin enthaltenen Bilder werden eingebunden.

Zeige exif-Tags
: Falls aktiviert, werden die EXIF-Metadaten einer Datei der Mediendatenbank dargestellt.

Exif data
: Wenn Sie die Darstellung der EXIF-Metadaten aktiviert haben, können Sie im Bereich **Exif data** festlegen, welche dieser EXIF-Daten tatsächlich angezeigt werden sollen. Sie können jeden der verfügbaren Werte gezielt aktivieren oder deaktivieren.

Show Media Properties
: Bei aktivierter Option können die vom Redakteur eingetragenen Metadaten (Copyright, Bildtitel . . .) zu den einzelnen Mediendateien angezeigt werden.

Media properties list
: Wenn Sie die Option **Show Media Properties** aktivieren, können Sie in diesem Eingabefeld festlegen, welche Metadaten Sie anzeigen wollen. Die hier verfügbaren Feldnamen richten sich nach dem, was Sie in der globalen Serendipity-Konfiguration eingetragen haben (siehe Seite 173).

Zeige einen Link zu den Einträgen/statischen Seiten, die auf

das Bild verlinken Wenn Sie diese Option aktivieren, wird in der Ansicht eines Bildes in der Galerie eine Liste angezeigt, die alle URL-Verweise auf dieses Objekt enthält. So können Sie sehen, von welchen Seiten aus ein Bild oder eine Datei eingebunden wurde. Dies kann nur ausgewertet werden, wenn Sie die Bilder mittels Serendipity-Wrapper ausgeben (siehe Seite 688).

Falls die Darstellung des Plugins mittels Template-Datei erfolgt (**Choose the gallery style** → **Media Library**), können Sie in den Dateien `plugin_usergallery.tpl` (Übersicht) und `plugin_usergallery_imagedisplay.tpl` (Einzelseite) auf folgende Smarty-Variablen zugreifen:

{`$staticpage_pagetitle`} (Zeichenkette)
: Enthält den Kurztitel der Galerieseite. Dies wird z. B. dazu benutzt, bestimmte HTML-Klassen mit einem eindeutigen Namen zu belegen, um sie individuell per CSS-Anweisungen formatieren zu können.

{`$const`} (Array)
: Enthält ein Array mit zusätzlichen Sprachvariablen der Galerie.

{`$plugin_usergallery_httppath`} (Zeichenkette)
: Enthält den URL-Pfad zur aktuellen Seite der Galerie.

{`$plugin_usergalleryhttppath_extend`} (Zeichenkette)
: Enthält den URL-Pfad zur aktuellen Seite der Galerie. Diese Variable kann zum Anhängen von weiteren URL-GET-Variablen verwendet werden, da sie mit einem ? oder & endet (je nach konfigurierter URL-Umformungsoption des Blogs, siehe Seite 168).

{`$plugin_usergallery_currentgal`} (Zeichenkette)
: Enthält den Namen des aktuellen Pfades der Mediendatenbank.

{`$plugin_usergallery_uppath`} (Zeichenkette)
: Enthält den Namen des übergeordneten Pfades des aktuell dargestellten Verzeichnisses der Mediendatenbank.

{`$plugin_usergallery_toplevel`} (Zeichenkette)
: Enthält den Wert `yes`, falls der Besucher das Stammverzeichnis der Mediendatenbank ansieht. Falls der Besucher ein Unterverzeichnis ansieht, enthält diese Variable den Wert `no`.

{`$plugin_usergallery_maindir_filecount`} (Zahl)
: Enthält die Anzahl an dargestellten Dateien des aktuellen Verzeichnisses.

{`$plugin_usergallery_subdirectories`} (Array)
: Enthält ein Array mit allen Unterverzeichnissen der Mediendatenbank. Jeder Array-Index enthält den Namen des jeweiligen Verzeichnisses, der Array-Wert enthält in einem Unter-Array mit dem Array-Schlüssel `filecount` die jeweilige Anzahl der Objekte in diesem Verzeichnis. Alle weiteren Array-Schlüssel richten sich nach der Liste auf Seite 540.

{`$plugin_usergallery_pagination`} (Boolean)
: Enthält den Wert `true`, wenn die Navigation zum Vor- und Zurückblättern der Galerieansicht dargestellt werden muss.

{`$plugin_usergallery_total_count`} (Zahl)
: Enthält die Gesamtzahl an Objekten in der Mediendatenbank.

{`$plugin_usergallery_total_pages`} (Zahl)
: Enthält die Gesamtzahl an Seiten, falls die Darstellung einer Galerie auf mehrere Seite aufgebrochen werden muss.

{`$plugin_usergallery_current_page`} (Zahl)
: Enthält die aktuelle Seitennummer, falls die Darstellung einer Galerie auf mehrere Seiten umbrochen werden muss.

{`$plugin_usergallery_next_page`} ,

{`$plugin_usergallery_previous_page`} (Zahl) Enthält die jeweils nächste und vorherige Seitennummer.

{`$plugin_usergallery_title`} (Zeichenkette)
: Enthält den Seitentitel der Galerie.

{`$plugin_usergallery_cols`} (Zahl)
: Enthält die gewünschte Anzahl an Spalten pro Galerieseite.

{`$plugin_usergallery_preface`} (Zeichenkette)
: Enthält die in den Konfigurationsoptionen hinterlegte Einführung (Text) zur Mediendatenbank.

{`$plugin_usergallery_fixed_width`} (Zahl)
: Enthält die maximale Breite eines Bildes.

{`$plugin_usergallery_image_display`} (Zeichenkette)
: Enthält den in den Konfigurationsoptionen festgelegten Wert, ob ein Bild bei Klick als Popup oder innerhalb derselben Seite angezeigt werden soll.

{`$plugin_usergallery_gallery_breadcrumb`} (Array)
: Enthält ein Array mit allen übergeordneten Pfaden des aktuellen Verzeichnisses der Mediendatenbank, so dass diese gezielt angesprungen werden können. Pro Array-Schlüssel enthält der Array-Wert den jeweiligen Verzeichnisnamen.

{$plugin_usergallery_dir_list} (Zeichenkette)
: Enthält den in den Konfigurationsoptionen festgelegten Wert, ob eine Verzeichnisliste eingeblendet werden soll.

{$plugin_usergallery_display_dir_tree} (Zeichenkette)
: Enthält den in den Konfigurationsoptionen festgelegten Wert, ob eine Verzeichnisnavigation eingeblendet werden soll.

{$plugin_usergallery_colwidth} (Zahl)
: Enthält einen gerundeten Wert, der abhängig von der konfigurierten Anzahl der Bilder pro Zeile die Spaltenbreite bei der Darstellung festlegt.

{$plugin_usergallery_limit_directory} (Zeichenkette)
: Enthält den Namen des aktuellen Verzeichnisses, von dem sämtliche Sonderzeichen entfernt wurden.

{$plugin_usergallery_images} (Array)
: Enthält ein Array mit allen Bildern der aktuellen Seite. Die Inhalte des Arrays richten sich nach den festgelegten Array-Schlüsseln (siehe Seite 543).

{$plugin_usergallery_nextid},

{$plugin_usergallery_previousid} (Zahl) Enthält die ID des jeweils vorherigen und nächsten Bildes.

{$plugin_usergallery_xtra_info} (Zeichenkette)
: Enthält eine HTML-Ausgabe mit den EXIF-Metadaten des Bildes.

{$plugin_usergallery_extended_info} (Zeichenkette)
: Enthält die HTML-Ausgabe mit in der Mediendatenbank festgelegten Medieneigenschaften einer Datei (z. B. Titel, Copyright etc.)

{$plugin_usergallery_file} (Array)
: Enthält das Array mit Bildinformationen, falls die Einzeldarstellung erfolgt. Die Inhalte des Arrays richten sich nach den Array-Schlüsseln (siehe Seite 543).

Bei der Darstellung mittels **Choose the gallery style** → **Media** greift das Plugin auf die Standard-Template-Dateien wie `media_choose.tpl` zurück (siehe Seite 535).

6.4.3 Erweiterte Optionen für Bildauswahl serendipity_event_imageselectorplus

Das Plugin *Erweiterte Optionen für Bildauswahl* dient mehreren Zwecken rund um die Bildverwaltung Serendipitys.

Zum einen ermöglicht es über eine sogenannte *QuickBlog*-Funktion, mit einem einfachen Klick ein hochgeladenes Bild direkt als neuen Blog-Artikel einzustellen. Dies ist besonders für Fotoblogs sehr praktisch, um etwas Klickarbeit zu sparen.

Weiterhin bietet das Plugin eine Möglichkeit, kleine Bilderserien in einen Blog-Eintrag einzufügen. Dies geschieht mittels einfachem XML-Code.

Das Plugin erweitert das Formular zum Hochladen eines Eintrags (falls Ihre PHP-Version dies unterstützt) um die Möglichkeit, auch ZIP-Archive mit Bildern/Dateien hochzuladen.

Zuletzt dient das Plugin dazu, die Optionen beim Einfügen einer Datei aus der Mediendatenbank in einen Blog-Eintrag zu erweitern. Seit Serendipity 1.1 sind diese Optionen jedoch bereits Bestandteil der offiziellen Version und werden somit innerhalb des Plugins nicht weiterverwendet. In Zukunft könnte das Plugin jedoch möglicherweise wieder eigenständige Optionen einbinden.

QuickBlog

Wenn Sie auf den Menüpunkt **Mediendatenbank** → **Mediendaten hinzufügen** klicken, bindet das Plugin dort einen neuen Bereich mit der Überschrift **QuickBlog** ein. Dort finden Sie mehrere Eingabefelder vor. Sobald Sie das Feld **Titel** ausfüllen, wird die hochgeladene Mediendatei automatisch in einen neuen Blog-Artikel eingebunden. Als Titel des Blog-Artikels wird der in dem Feld **Titel** eingegebene Inhalt verwendet.

Der so entstandene Blog-Artikel ist später wie ein ganz regulärer Artikel zu bearbeiten. Er enthält jedoch bereits sämtlichen HTML-Code, der für die Darstellung des Bildes notwendig ist.

Die Eingabefelder im **QuickBlog**-Bereich legen einen Titel für das Bild fest, einen zusätzlichen beschreibenden Text und die Zuordnung zur Blog-Kategorie.

Das Feld **Bildgröße** enthält eine Zahl (in Pixeln), die bestimmt, in welcher Größe das Bild in dem Quickblog-Artikel eingestellt wird.

Technisch gesehen wird ein via Quickblog hochgeladenes Bild mit speziellem Code in Ihren Eintrag eingebunden. Dieser Code enthält die ID zu einem Bild der Mediendatenbank und wird im normalen Eintragstext des Artikels hinterlegt:

```
<!--quickblog:4711-->
```

Wenn ein solcher Blog-Artikel angezeigt wird, sucht das Plugin automatisch nach dem Vorkommen derartiger Zeichenketten. Daraufhin liest es die Mediendatenbank aus und erstellt eine formatierte Darstellung für das Bild. Die Formatierung richtet sich nach der Template-Datei **quickblog.tpl** des Plugins. Der dort hinterlegte HTML-Code stellt das Bild, einen Link zum Bild sowie etwaige EXIF-Daten ein. Diese Template-Datei können Sie Ihren eigenen Wünschen anpassen und in Ihr jeweiliges Template-Verzeichnis kopieren.

Folgende Variablen stehen dort zur Verfügung:

{$quickblog.image}
: Enthält die URL zu dem Vorschaubild.

{$quickblog.fullimage}
: Enthält die URL zu dem Bild in Originalgröße.

{$quickblog.body}
: Enthält den etwaigen zusätzlichen Eintragstext des Blog-Artikels.

{$quickblog.exif}
: Enthält ein Array mit allen EXIF-Metadaten des Bildes. Der Array-Index richtet sich nach dem jeweiligen EXIF-Feldnamen, z. B. Focal_length oder COMMENT.0.

{$quickblog.exif_mode}
: Falls auf Ihrem Server das Programm **jhead** installiert ist, kann das Plugin dieses verwenden, um die EXIF-Daten eines Bildes zu lesen. In diesem Fall ist die Variable {$quickblog.exif_mode} auf den Wert jhead gesetzt.

  Ohne dieses Programm versucht das Plugin die PHP-Funktion exif_read_data zu verwenden. Dann ist {$quickblog.exif_mode} auf internal gesetzt.

  Wenn keine Datei gelesen werden konnte, enthält die Variable den Wert none.

Sie können die <!--quickblog:xxx-->-Anweisungen praktisch auch eigenständig in Artikel einfügen und in einen normalen Artikeltext einbetten. Bei der Darstellung wird dann lediglich die quickblog-Anweisung ausgetauscht, Ihr restlicher Text erscheint wie gewohnt.

XML-Bilderserien

Wenn Sie viele Bilder in einen Artikel einbinden wollen, kann das in mühsame Klickorgien ausarten. Das Plugin bietet Ihnen hier einen angenehmeren, wenn auch technischeren Weg, dasselbe Ziel zu erreichen.

Sie können eigenen (aber einfachen!) Code in Ihren Blog-Artikel einfügen, der an eine XML-Syntax angelegt ist. Diesen wertet das Plugin später automatisch aus, und bindet die gewünschten Bilder ein. Die Einbindung erfolgt anhand einer eigenständigen Template-Datei (plugin_mediainsert.tp so dass Sie hier die Formatierung beliebig anpassen können.

Der XML-Code sieht dabei wie folgt aus:

```
<mediainsert>
 <gallery name='Verzeichnisname' />
 <media type='single' name='bildname1' />
 <media type='single' name='bildname2' />
</mediainsert>
```

Sie können so viele `<mediainsert>`-Blöcke in Ihrem Eintrag einfügen, wie Sie wünschen. Innerhalb dieses Blocks müssen Sie das `<gallery>`-Element angeben. In dessen Attribut `name` tragen Sie den Namen des Verzeichnisses ein, aus dem Sie ein Bild darstellen wollen. Als Verzeichnisname muss hier nur der tatsächliche Verzeichnisname unterhalb Ihres `uploads`-Serververzeichnisses eingetragen werden. Wenn Sie also Ihre Bilder in `/var/www/example.com/serendipity/uploads/Urlaub/Mallorca2006` hochgeladen haben, tragen Sie `<gallery name='Urlaub/Mallorca2006/' />` ein. Der abschließende Schrägstrich ist erforderlich. Wenn Sie Bilder des Stammverzeichnisses darstellen wollen, können Sie `<gallery name='/' />` verwenden.

Daraufhin folgt eine beliebige Anzahl von `<media>`-Elementen. Als `type`-Attribut können Sie entweder `single` (ein einzelnes Bild), `gallery` (das komplette Verzeichnis) oder `range` (durchnummerierte Bilderreihe) eintragen.

Bei Bildern vom Typ `single` müssen Sie im Attribut `name` den Dateinamen (ohne Dateiendung) eintragen. Um also `uploads/Urlaub/Mallorca2006/dscf_001.jpg` darzustellen, können Sie `<media type='single' name='dscf_001' />` verwenden. Für jede Datei, die Sie anzeigen wollen, verwenden Sie eine solche XML-Zeile. Dabei entspricht die Reihenfolge Ihrer Eingabe der Reihenfolge, in der die Bilder später dargestellt werden.

Bildern des Typs `range` müssen Sie die Start- und Endnummer mitgeben. Wenn Sie die Dateien `dscf_1.jpg` bis `dscf_10.jpg` darstellen wollen, benutzen Sie `<media type='single' prefix='dscf_' start='1' stop='10'`. Achten Sie daher auch darauf, dass Ihre Dateien keine führenden Nullen bei der Durchnummerierung enthalten, da das Plugin andernfalls auf nicht existierende Dateinamen verweisen könnte.

Wenn Sie ein vollständiges Verzeichnis auslesen wollen, reicht die Zeile `<media type='gallery' />` aus. Es werden dann alle Dateien eingebunden, die im durch `<gallery name='Urlaub/Mallorca2006/' />` festgelegten Verzeichnis in die Mediendatenbank geladen wurden.

Der Code muss sich strikt an XML-Syntax halten. `<media type='gallery'>` wäre beispielsweise ungültig, da das Anführungszeichen nicht übereinstimmt und der abschließende Schrägstrich des XML-Elements fehlt. Wenn Sie diese Regeln nicht beachten, kann dies dazu führen, dass das Plugin eine Fehlermeldung darstellt.

Innerhalb der Template-Datei `plugin_mediainsert.tpl`, die zur Darstellung Ihrer Bilderliste verwendet wird, stehen folgende Variablen zur Verfügung:

{`$plugin_mediainsert_media`}
: Enthält ein Array mit allen Bilddateien. Als Array-Schlüssel stehen folgende Werte zur Verfügung:

  {`$plugin_mediainsert_media.X.name`}
  : Enthält den Dateinamen (ohne Datei-Endung) der Bilddatei.

  {`$plugin_mediainsert_media.X.extension`}
  : Enthält die Datei-Endung der Bilddatei.

{`$plugin_mediainsert_media.X.thumbnail_name`}
: Enthält den Namen (Suffix) der Vorschaudatei.

{`$plugin_mediainsert_media.X.realname`}
: Enthält den vollständigen Dateinamen.

{`$plugin_mediainsert_media.X.path`}
: Enthält den URL-Pfad zu der Bilddatei.

{`$plugin_mediainsert_media.X.comment1`}
: Kommentar zu einem Bild, wie in der Mediendatenbank festgelegt. Hierfür wird davon ausgegangen, dass Sie die Beschreibung im Datensatz `COMMENT1` gespeichert haben (siehe Konfigurationsoptionen der Mediendatenbank auf Seite 173). Falls kein Kommentar eingetragen wurde, wird der Dateiname verwendet.

{`$plugin_mediainsert_media.X.width, .height`}
: Größe des Originalbildes (Breite, Höhe).

{`$plugin_mediainsert_media.X.thumbwidth, .thumbheight`} Größe des Vorschaubildes (Breite, Höhe).

ZIP-Archive

Wenn Sie die Konfigurationsoption **ZIP archives unzipping** des Plugins aktiviert haben und Ihr Server mindestens PHP 5.1.0 einsetzt, könnnen Sie ein ZIP-Archiv mit mehreren Bildern in die Mediendatenbank hochladen. Die ZIP-Datei wird automatisch auf dem Server entpackt, und alle darin enthaltenen Dateien werden so hochgeladen, als hätten Sie diese einzeln eingestellt.

Bitte beachten Sie, dass das Entpacken großer ZIP-Dateien viele Ressourcen auf dem Server bindet und eventuell zu Timeouts oder anderen Performanceproblemen führen könnte. In diesem Fall müssen Sie die Dateien einzeln hochladen.

Konfigurationsoptionen

Die Konfigurationsoptionen des Plugins beinhalten lediglich die Möglichkeit, eine von der normalen Serendipity-Konfiguration abweichende Größe der Vorschaubilder einzutragen. Bei den üblichen Vorschaubildern in Serendipity wird die größte Seite (Höhe oder Breite) eines Bildes für die Verkleinerung herangezogen. Mittels der beiden Konfigurationsoptionen jedoch können Sie ganz gezielt ein Format für die Vorschaubilder festlegen, falls Sie z. B. die Proportionen eines Bildes nur horizontal statt vertikal beschränken wollen.

Weiterhin gibt Ihnen das Plugin die Möglichkeit, festzulegen, ob die Erkennung der XML-Bilderserien innerhalb des *Eintrags* oder *Erweiterten Eintrags* zugelassen werden soll. So können Sie kontrollieren, dass Bilderserien von Ihren Redakteuren z. B. nicht auf der normalen Übersichtsseite des Blogs eingesetzt werden.

6.5 Auswahl an Profi-Plugins

Serendipity bietet besonders für Entwickler eine größere Zahl an komplexen und flexiblen Plugins. Eine kleine Auswahl wird auf den folgenden Seiten vorgestellt. Auch für Einsteiger in die Weblog-Szene könnte es interessant sein, einmal über den Tellerand zu blicken, um zu sehen, welche Möglichkeiten sich eröffnen, wenn Sie sich tiefer in Serendipity einarbeiten.

6.5.1 Show entries via JavaScript serendipity_event_backend

Dieses Plugin bietet eine JavaScript-basierte Schnittstelle an, um Blog-Einträge auf unabhängigen Seiten über einen Hintergrundaufruf einzubinden.

Sobald das Plugin installiert ist, kann man eine spezielle URL aufrufen, die abhängig von bestimmten URL-Parametern Blog-Einträge mittels JavaScript ausgibt. Diese Art der Einbindung arbeitet ähnlich wie die Benutzung von GoogleAds oder anderen JavaScript-*Widgets*.

Anwendung findet dieses Plugin dann, wenn Sie die Inhalte Ihrer Blog-Artikel auf Webseiten einbinden wollen, auf denen Serendipity nicht installiert ist. Zwar könnten Sie dies auch durch die Einbindung des RSS-Feeds erledigen, aber dies ist oft mit Programmieraufwand verbunden. Auch das Einbinden des Serendipity-PHP-Frameworks (falls die einbindenden Webseiten auf demselben Server wie das Blog liegen) wäre eine mit zusätzlichem Aufwand verbundene Methode.

Standardmäßig können Sie nach der Installation des Plugins eine URL wie `http://www.example.com/serendipity/i` aufrufen. Diese URL gibt JavaScript-Befehle aus, die die Inhalte der letzten Blog-Einträge enthalten. Daher ist diese URL nicht zum Aufruf im Browser gedacht, sondern Sie sollten sie wie folgt in Ihre HTML-Seite einbauen:

```
<html>
 <body>
 Hier eine Liste meiner aktuellen Beiträge:
 <div id="backend">
   <script scr="http://www.example.com/serendipity/index.php?/plugin/
backend"></script>
 </div>
 </body>
</html>
```

Wenn Sie diese HTML-Datei als `test.htm` auf einem beliebigen Server speichern und aufrufen, sollten Sie auf dieser Seite die mittels JavaScript eingebundenen Blog-Einträge sehen können.

Dies können Sie beispielsweise benutzen, wenn Sie Webseiten betreiben, die nicht mit Serendipity betrieben werden (Werbeportale, Firmenseiten, private Seiten ihrer Mitarbeiter), und auf denen Sie gerne Blog-Artikel ausgeben möchten.

Um gezielt bestimmte Artikel auszulesen, bietet das Plugin mehrere URL-Parameter an. Standardmäßig werden nur Links zu den Blog-Einträgen sowie deren Titel angezeigt. Eine Ausgabe des Plugins sähe im JavaScript-Code so aus:

```
document.write('<a
href="http://www.example.com/serendipity/archives/1-isotopp-rock0rz.html
">Blog-Huldigung</a><br/>');
```

Um diese Ausgabe zu beeinflussen, können Sie die folgenden URL-Parameter mittels `http://www.example.com/serendipity/backend&variable1=wert1&variable2=wert2...` aufrufen – Variablen sind mit & voneinander getrennt:

`&category=`*X*
: Beschränkt die Ausgabe der Artikel auf eine spezielle Kategorie. *X* entspricht dabei dem Namen einer Kategorie.

`&categoryid=`*X*
: Beschränkt die Ausgabe der Artikel auf eine (oder mehrere) spezielle Kategorie(n). *X* entspricht dabei der ID einer Kategorie. Mehrere Kategorie-IDs können durch ein Semikolon (`;`) voneinander getrennt werden.

`&authorid=`*X*
: Beschränkt die Ausgabe der Artikel auf einen speziellen Autor. *X* enthält die ID des gewünschten Autors. Um mehrere Autoren einzubeziehen, können Sie deren IDs durch ein Semikolon (`;`) voneinander trennen.

`&num=`*X*
: Beschränkt die Anzahl der Artikel auf *X* Stück.

`&order=`*X*
: Gibt die Reihenfolge der Artikel an. *X* kann entweder `asc` (aufsteigend) oder `desc` (absteigend) enthalten.

`&showdate=`*X*
: Wenn der Wert `showdate=1` übermittelt wird, gibt das JavaScript das Datum eines Beitrages mit aus.

`&dateformat=`*X*
: Bei gesetztem `showdate`-Parameter bestimmt die Variable die Formatierung des Datums. Standardmäßig wird das Format `Y-m-d` (Jahr-Monat-Tag) benutzt. Gültige Platzhalter finden Sie in der PHP-Dokumentation zum `date()`-Befehl[42].

`&showtime=`*X*
: Wenn der Wert `showtime=1` übermittelt wird, gibt das JavaScript die Uhrzeit eines Beitrages mit aus.

[42] `http://www.php.net/date`

`&timeformat=`*X*
: Bei gesetztem `showtime`-Parameter bestimmt die Variable die Formatierung der Uhrzeit eines Eintrags. Standardmäßig wird das Format `g:ia` (Stunde:Minute am/pm) benutzt. Gültige Platzhalter finden Sie in der PHP-Dokumentation zum `date()`-Befehl. Sie können die Platzhalter zur Uhrzeit-Formatierung jedoch auch über die Variable `dateformat` mitbestimmen. Wenn Sie sowohl `dateformat` als auch `timeformat` übermitteln, werden in der Datumsausgabe des Plugins beide Platzhalter benutzt.

`&point=`*X*
: Die URL-Variable kann eine beliebige Zeichenkette (ohne HTML!) enthalten, die zur Trennung der einzelnen Artikel benutzt wird. So können Sie beispielsweise `point=*` benutzen, um vor jedem Artikel einen Stern auszugeben.

`&details=`*X*
: Wenn Sie die Variable `details=1` benutzen, bindet das Plugin nicht nur den Link des Artikels ein, sondern auch den vollständigen Blog-Artikeltext (ohne den erweiterten Eintrag).

Etwaige Sonderzeichen für die Werte der Variablen (beispielsweise Kategorienamen) müssen mit URL-Sonderzeichen ersetzt werden. Aus dem Kategorienamen *Thundercats und Mäuse* würde daher der Wert `Thundercats+und+M%E4use`. Die PHP-Funktion `urlencode` kann Variablen entsprechend kodieren, siehe `http://de2.php.net/urlencode`.

Wenn Sie also beispielsweise fünf Artikel der ersten Kategorie, aufsteigend sortiert, inklusive Artikeltext mit deutschem Datumsformat anzeigen wollen, müssen Sie folgende URL einbinden:

```
http://www.example.com/serendipity/index.php?/plugin/backend&categoryid=
1&details=1&num=5&order=asc&showdate=1&dateformat=d.m.Y+H:i
```

Wenn das JavaScript erfolgreich in Ihrer Webseite eingebunden ist, können Sie mittels der CSS-Klassen `.blog_body` (Artikelinhalt), `.blog_hr` (Artikeltrenner), `.blog_author` (Autorname), `.blog_date` (Artikeldatum), `.blog_title` (Artikeltitel), `.blog_point` (Artikeltrenner) und `.blog_link` (Hyperlink) die Formatierung der Ausgabe individuell beeinflussen.

Wollen Sie die Ausgabeformatierung generell ändern, können Sie die Datei `plugins/serendipity_event_backend/se`
`php` bearbeiten und am Ende der Datei die JavaScript `document.writeln`-Zeilen anpassen.

In den Konfigurationsoptionen des Plugins können Sie einstellen, über welchen Plugin-Pfad (`index.php?/plugin/`*`backend`*) das JavaScript aufgerufen wird. Anstelle von `backend` können Sie auch einen beliebigen anderen Pfad konfigurieren. Dieser Pfad sollte jedoch lediglich die Buchstaben A bis Z und Zahlen enthalten – Sonderzeichen, Unterstriche, Schrägstriche und andere könnten Probleme verursachen.

6.5.2 Textformatierung: Smarty Markup serendipity_event_smartymarkup

Serendipity erlaubt aus Sicherheitsgründen nur HTML und Klartext in Beiträgen und Seitenleisten-HTML-Klötzen. Bei freier Verwendung von PHP-Code könnten Redakteure beliebigen Code einschleusen, der sogar dazu führen könnte, dass die vollständige Serendipity-Datenbank gelöscht wird.

Dennoch reizt die Flexibilität, die man mit eigenem Programmiercode in einem Blog-Artikel erreichen könnte. Dank JavaScript und HTML kann man zwar bereits einiges umsetzen, aber nun mal nicht das, was man mittels PHP programmieren könnte: Einbindung von Umfragen in einen Artikel, Einlesen von Dateien auf dem Server, uhrzeitabhängige Einbindung unterschiedlicher Inhalte – grundsätzlich alles, soweit Ihre Vorstellungskraft reicht.

Das Plugin *Textformatierung: Smarty Markup* kann diesem Problem etwas Abhilfe schaffen. Es erlaubt zwar nicht, rohen PHP-Scriptcode in Ihre Einträge einzufügen, aber dafür können Sie damit Smarty-Funktionen benutzen.

Sobald Sie das Plugin installiert haben, können Sie beliebige Smarty-Syntax in Ihren Einträgen verwenden (siehe Smarty-Dokumentation ab Seite 480). Die Templates-Variablen des eigenen Artikels können über das Array {`$smartymarkup_eventData`} (z. B. {`$smartymarkup_eventD`
) aufgerufen werden.

Mittels der Smarty-Syntax können Sie auch Ihren eigenen PHP-Code als Smarty-Funktion schreiben und diesen dann in Artikeln aufrufen. Diese zusätzliche Hürde erfordert, dass Sie als Administrator via FTP Dateien auf dem Server anpassen können (konkret sind die eigenen Smarty-Funktionen in der Datei `config.inc.php` Ihres Templates einzufügen). Sobald ein Administrator PHP-Dateien anpassen kann, kann man auch davon ausgehen, dass er verantwortungsvoll mit eigenem PHP-Code umgehen kann.

Mit der Smarty-Syntax können Sie zusätzlich alle von Serendipity bereitgestellten Smarty-Funktionen ansprechen.

Ein großer Vorteil des Smarty-Markups ist, dass Sie diesen auch gezielt für statische Seiten (siehe Seite 410) freischalten können. Denn gerade in statischen Seiten könnten Zusatzfunktionen, die nur mittels PHP umzusetzen sind, sehr hilfreich sein.

In vielen Fällen reicht die Benutzung von Smarty-Funktionen bereits aus, so dass Sie kein eigenes Serendipity-Plugin entwickeln müssen. Nutzen Sie diese Flexibilität gezielt, um beispielsweise in HTML-Klötzen der Seitenleiste eigene PHP-Funktionen aufzurufen oder Ihr eigenes PHP-Framework in Serendipity zu integrieren.

In den Konfigurationsoptionen des Plugins können Sie gezielt einstellen, für welche Eingabemöglichkeiten (Eintrag, erweiterter Eintrag, Kommentare, HTML-Klotz, statische Seiten) die Smarty-Syntax zugelassen wird. Nur in besonderen Fällen sollten Sie Smarty-Funktionen auch für Benutzerkommentare freischalten – auch die Smarty-Syntax birgt Sicherheitsrisiken, die böswillige Besucher über Kommentare ausnutzen könnten.

6.5.3 Textformatierung: Eintragsdaten einfügen serendipity_event_includeentry

Das Plugin *Textformatierung: Eintragsdaten einfügen* bietet trotz Klassifizierung als *Textformatierungs-Plugin* zahlreiche und flexible Funktionen. Mithilfe dieses Plugins können Sie Daten bestehender Blog-Einträge in anderen Einträgen einbinden, einen Artikelpool (sog. *Template-Blocks*) zum Anhängen an einen Eintrag verwalten und auch Vorlagen für neue Beiträge erfassen. Alle drei Möglichkeiten sind voneinander unabhängig einsetzbar.

Nach der Installation des Plugins finden Sie einen neuen Menüpunkt **Einträge** → **Template-Blocks** im Backend. In diesem Bereich sehen Sie eine zweigeteilte Seite: Links können Sie Vorlagen für neue Blog-Einträge (*Templates*) verwalten, in rechten Teil können Sie den Artikelpool (*Blocks*) bearbeiten.

Um eine neue Artikelvorlage zu erstellen, müssen Sie im linken Bereich unter **Select Template** das Auswahlfeld auf **Neuer Eintrag** stellen und auf den Button **Los!** klicken. Danach können Sie in der Folgeseite einen Titel, einen Eintrag und den erweiterten Eintrag festlegen, den Sie als Artikelvorlage speichern wollen. Diese Vorlage können Sie beispielsweise benutzen, um immer einheitliche Blog-Artikel zu erstellen. So können Sie für Blog-Artikel zu Filmberichten bereits ausfüllbare HTML-Tags vorsehen, die die Links zu der Film-Homepage enthalten und eine grobe Gliederung von Überschriften (*Der Inhalt*, *Die Schauspieler*, *Mein Fazit*) vorbereiten. Auch wenn Sie spezielle HTML-Konstrukte für Ihre Blog-Artikel vorsehen, könnten Sie diese als Vorlage speichern, um denselben HTML-Code nicht manuell für jeden neuen Eintrag erstellen zu müssen.

Um einen neuen Blog-Artikel auf Basis einer Vorlage zu erstellen, stellen Sie unter **Einträge** → **Template-Blocks** das Auswahlfeld unter **Select Template** auf die gewünschte Vorlage und klicken dann **Use Template**. Sie gelangen auf die bekannte Oberfläche zum Erstellen eines Blog-Eintrages; die Felder *Titel*, *Eintrag* und *Erweiterter Eintrag* sind korrekt vorausgefüllt.

Über die Template-Blocks-Oberfläche können Sie Vorlagen auch bearbeiten oder löschen, indem Sie das Auswahlfeld auf die gewünschte Vorlage setzen und den Button **Los!** oder **Löschen** anklicken.

Ähnlich wie die Verwaltung der Vorlagen können Sie sogenannte *Blocks* erstellen (und auch nachträglich bearbeiten und löschen). Ein Block stellt dabei einen Datensatz dar, den Sie (wie auch Vorlagen) mit *Titel*, *Eintrag* und *Erweitertem Eintrag* ausfüllen können. Im Gegensatz zu einer Vorlage ist ein Block jedoch ein eigenständiger Datensatz, der in einen Blog-Artikel eingebunden werden kann.

Blöcke können Sie dazu verwenden, häufig wiederkehrende Elemente nahtlos an einen Artikel anzubinden. Wiederkehrende Elemente sind beispielsweise Werbeblöcke, Kontaktformulare, Disclaimer und andere rechtliche Hinweise. Grundsätzlich ist hier alles denkbar, was Sie mehr als einmal in einen Artikel einfügen wollen.

Die Einbindung solcher Blöcke erfolgt über die gewohnte Oberfläche zur Erstellung eines neuen Blog-Artikels. Dort finden Sie im Bereich **Erweiterte Optionen** einen Abschnitt na-

mens **Attach a static Block**. Darunter wird ein Auswahlfeld dargestellt, aus dem Sie einen vorhandenen Block auswählen können.

Der Inhalt eines Blocks wird dabei im Frontend unterhalb des Eintragstextes (mittels der Smarty-Variable {`$entry.display_dat`}) eingebunden. Über die Template-Datei `entries.tpl` können Sie die Positionierung dieser Smarty-Variable jedoch auch frei beeinflussen. Für flexiblere Seitengestaltungen können Sie zusätzlich den eingebundenen Block über die Variable {`$entry.entryblock`}) umpositionieren.

Bei der Bearbeitung eines Blocks können Sie zwei Sonderoptionen festlegen. Zum einen bestimmt die Option **Template (Smarty)**, welche Template-Datei zur Darstellung eines Template-Blocks benutzt wird. Diese Datei legt fest, wie der von Ihnen angelegte Template-Block innerhalb Ihres Blog-Artikels formatiert wird. Die Standarddatei `plugin_staticblock.tpl` ist recht einfach gehalten und gibt lediglich den Titel und Text mit HTML-Div-Containern aus. Pro Block können Sie eine eigene Template-Datei bestimmen, um so unterschiedliche Formatierungen für unterschiedliche Blöcke zu erreichen. Damit können Sie rechtliche Hinweise grafisch anders darstellen als beispielsweise Google-Ad-Werbungen.

Die zweite Sonderoption **Textformatierung auf Block anwenden** gibt an, ob der Text des jeweiligen Blocks durch andere Textformatierungs-Plugins (beispielsweise Smiley-Konvertierung) ausgewertet werden soll.

Wie eingangs erwähnt, ermöglicht das Plugin auch eine eigene Textformatierung, mit der Sie in Blog-Artikeln beliebige Inhalte von bestehenden anderen Artikeln einbinden können. Dies können Sie auslösen, indem Sie in einem Artikel die Zeichenkette `[s9y-include-entry:`*X*`:`*Y*`]` an gewünschter Stelle einfügen. Ersetzen Sie dabei *X* durch die ID des bestehenden Blog-Artikels und *Y* mit dem gewünschten Datenbankfeld, das Sie auslesen wollen.

Als Beispiel hierfür folgender Blog-Artikel:

```
In meinem Blog-Artikel namens "[s9y-include-entry:1:title]" schrieb ich
damals:

[s9y-include-entry:1:body]
```

Der Vorteil einer derartigen Einbindung ist, dass Sie einen älteren Blog-Artikel ändern können und sich dieser automatisch auch in dem referenzierenden Artikel ändert. Das einzig Umständliche an dieser Einbindungsart ist, dass Sie die ID eines Artikels von vornherein wissen müssen (diese ist meist Bestandteil des Permalinks Ihres Eintrags).

Um auf die *Erweiterten Eigenschaften* eines Artikels zuzugreifen, können Sie die Syntax `[s9y-include-entry:1:prop=ep_Video]` verwenden, um beispielsweise ein angehängtes Video (siehe Podcast-Plugin, Seite 353) einzubinden. Sämtliche Feldnamen (wie `body`) entsprechen hierbei den Datenbank-Spaltennamen der Tabelle `serendipity_entries` bzw. `serendipity_entryproperties`.

Auch Template-Blocks können Sie in einem Blog-Artikel mit einer ähnlichen Syntax einbinden. Dies funktioniert unabhängig von dem vorher beschriebenen Anhängen eines Template-Blocks an einen Artikel:

```
And now over to something completely different: Werbung!

[s9y-include-block:1:body]
```

Dies würde den Text des ersten Template-Blocks in den Artikel einfügen. Meist ist jedoch gewünscht, die vollständig geparste Template-Datei (standardmäßig `plugin_staticblock.tpl`) des Template-Blocks auszugeben. Dies erreichen Sie mittels:

```
[s9y-include-block:1:template]
```

Der besondere Feld-Bezeichner `template` löst also die Rückgabe des geparsten Templates aus. Andernfalls stehen die Feldnamen `title, body, extended, author, authorid, last_modified` und `timestamp` zur Verfügung.

Abgesehen von diesen zahlreichen Einbindungsarten bietet das Plugin auch noch eine Fülle von globalen Konfigurationsoptionen an:

Show random blocks, Kategorien
: Das Plugin ermöglicht es, zufällig einen beliebigen Block unterhalb eines Artikels einzufügen (unabhängig von möglicherweise manuell eingefügten Template-Blocks).

  Solche zufälligen Blöcke können Sie verwenden, wenn Sie beispielsweise einen großen Pool an Werbebannern und Ähnlichem angelegt haben. Aktivieren Sie die Option **Show random blocks**, wenn Sie diesen Zufallsblock verwenden wollen.

  Die Auswahl der Kategorien bezieht sich bei aktivierter Option darauf, dass Zufallsblöcke nur dann gezeigt werden, wenn der Besucher sich gerade im Frontend in einer speziellen Kategorie aufhält.

  Mithilfe dieser Einschränkung könnten Sie daher die Werbeblöcke nur in gewünschten Kategorien einblenden. Standardmäßig sind alle Kategorien aktiviert, daher werden Zufallsblöcke überall angezeigt. Der Eintrag **[Keine Kategorie]** im Auswahlfeld entspricht dabei der Übersichtsseite. Wenn Sie keine Zufallsblöcke in der Übersichtsseite anzeigen wollen, heben Sie die Auswahl dieser Option einfach auf. Mehrere Kategorien können Sie in dem Mehrfach-Auswahlfeld mit gedrückter *(Strg/Apfel)*-Taste und einem Mausklick aktivieren.

First entry, Skip entries
: Wenn Sie die Option **Show random blocks** aktiviert haben, können Sie mit der Option **First entry** festlegen, nach dem wievielten Artikel die Anzeige eines Zufallblocks beginnen soll.

  Wenn Sie hier die Zahl `2` eintragen, wird erst nach dem zweiten Artikel in der Übersicht eine Werbung angezeigt. Mit der Option **Skip entries** legen Sie fest, nach jedem wievielten Artikel ein zufälliger Block eingebunden werden soll. Die Zahl `2` würde bewirken, dass nur nach jedem zweiten Eintrag ein Block eingebunden wird.

Allow multiple blocks
Bei aktivierter Zufallsfunktion kann es durchaus vorkommen, dass ein zufälliger Block unter einem Artikel eingebunden wird, dem Sie bereits manuell einen Template-Block zugewiesen haben.

Wenn Sie die Option **Allow multiple blocks** aktivieren, wird ein Zufallsblock jederzeit angehängt. Bei deaktivierter Option wird der Zufallsblock ignoriert und nicht dargestellt.

Eintrag, Erweiterter Eintrag, Kommentar, HTML-Klotz
Über diese Felder legen Sie fest, in welchen Eingabemöglichkeiten das Plugin die Syntax `[s9y-include-entry]` anbieten soll. Grundsätzlich kann es ein Sicherheitsrisiko sein, wenn Sie die Syntax in Kommentaren von Besuchern zulassen.

Datenbanktabelle

Das Plugin erstellt die Datenbanktabelle `serendipity_staticblocks`. Darin werden die **Template-Blöcke** gespeichert:

`id`
enthält eine fortlaufende ID, die jeden Template-Block identifiziert.

`title`
enthält den Betreff eines Template-Blocks bzw. einer Artikelvorlage.

`type`
gibt an, ob der Eintrag der Datenbank einen Template-Block (`block`) oder eine Artikelvorlage (`template`) darstellt.

`body`
enthält den Eintragstext.

`extended`
enthält den erweiterten Eintrag.

`template`
enthält den Namen der Template-Datei zur Darstellung des Blocks.

`apply_markup`
legt fest, ob Textformatierungs-Plugins angewendet werden sollen.

`author`
enthält den Namen des erstellenden Redakteurs.

`authorid`
enthält die ID des erstellenden Redakteurs.

`last_modified`
: enthält das Datum der letzten Änderung an der Vorlage bzw. am Template-Block.

`timestamp`
: enthält das Erstellungsdatum der Vorlage bzw. des Template-Blocks.

6.5.4 Einfache Cached/Pregenerated Seiten serendipity_event_cachesimple

Serendipity ist aufgrund seiner hohen Flexibilität und Dynamik leider wenig ressourcenschonend. An vielen Stellen des Systems können Plugins eingreifen und abhängig vom eingeloggten Benutzer unterschiedliche Inhalte anzeigen. Das Plugin *Erweiterte Eigenschaften für Artikel* (siehe Seite 262) kann zwar die Ausführung der Textformatierungs-Plugins beschleunigen, gilt aber nicht für den kompletten Seitenaufruf.

Das Plugin *Einfache Cached/Pregenerated Seiten* ist daher ein recht einfaches Plugin, das in den generellen Seitenaufbau eines Serendipity-Blogs eingreift. Ein Seitenaufruf sieht folgende Reihenfolge[43] vor:

- Aufruf der Webseite durch den Besucher.
- Laden der Funktionsdateien von Serendipity (Dateien im `include`-Verzeichnis).
- Herstellung einer Datenbankverbindung.
- Laden der globalen Serendipity-Konfiguration aus der Datenbank.
- Laden der Plugin API und installierter Ereignis-Plugins.
- Laden der persönlichen Konfigurationsdaten bei eingeloggten Benutzern.
- Analyse der aufgerufenen URL.
- Zusammenstellung der Artikel/Daten aus Datenbanktabellen für die aufgerufene URL, parallel Ausführung etwaiger Plugins bei den jeweiligen Schritten.
- Aufruf des Smarty-Template-Frameworks.
- Laden und Kompilieren des eingestellten Templates.
- Darstellung des Templates.

[43] Dies ist eine verkürzte Darstellung. Die technischen Details finden Sie im Kapitel 10 ab Seite 591.

Obwohl dies relativ komplex aussieht, kann Serendipity einen derartigen Workflow typischerweise in weitaus weniger als einer Sekunde ausführen, so dass man pro Sekunde durchaus mehrere Besucher bedienen kann.

Das Caching-Plugin setzt nun bei Schritt 7 an und versucht für eine aufgerufene URLs alle Folgeschritte zu sparen. Dabei geht das Plugin davon aus, dass, wenn ein Besucher eine identische URL bereits vorher aufgerufen hat, dieselbe Seite auch für den aktuellen Besucher gilt, und gibt denselben Inhalt aus.

Daher speichert das Plugin bei jedem Seitenaufruf die ausgegebene Seite im Verzeichnis `templates_c` zwischen und gibt sie später aus. Erst nach Ablauf einer Stunde oder beim Veröffentlichen eines neuen Artikels wird der Zwischenspeicher erneut erzeugt. Das Plugin bietet zwei Konfigurationsoptionen: **Use seperate IE/Mozilla caches?** und **Force clients to maintain fresh copy**.

Die erste Option bewirkt, dass das Plugin für Internet Explorer und Mozilla-Browser unterschiedliche Zwischenspeicher erstellt. Das ist dann hilfreich, wenn Sie möglicherweise in Ihrem Template oder einem PHP-Code eine Browser-Erkennung durchführen. Da das Plugin in den Zwischenspeicher immer das übernimmt, was der erste Besucher durch seinen Aufruf sieht, würde im Zwischenspeicher sonst, unabhängig vom Browser dieses Besuchers, möglicherweise eine falsche Kopie landen.

Wenn Sie die Option **Force clients to maintain fresh copy** aktivieren, übermittelt das Plugin an den Browser des Besuchers eine Uhrzeit, zu der die aktuell zwischengespeicherte Ausgabe nicht mehr gültig ist und ersetzt wird (der sog. HTTP-*Expires*-Header). Dies hat den Vorteil, dass ein Browser dann selbständig entscheiden kann, ob er eine Webseite aus dem Browser-Cache anzeigt oder vollständig aus dem Blog abruft, und kann daher die Seitendarstellungsgeschwindigkeit bei den Besuchern noch weiter beschleunigen.

Ist das Caching-Plugin aktiviert, gibt es weitere HTTP-Header aus, die dafür sorgen, dass es einem Browser generell erlaubt ist, die Blog-Seiten zwischenzuspeichern. In gut besuchten Blogs könnte das dazu führen, dass neue Kommentare einem Besucher nicht direkt angezeigt werden, da dieser immer noch eine alte Version der Seite von seinem Browser bezieht. Daher gibt Serendipity ohne dieses Plugin immer der Aktualität den höchsten Stellenwert und erlaubt solches Caching standardmäßig nicht.

Obwohl das Plugin mit der Abkürzung einer Seitenerzeugung einiges an Arbeit sparen kann, gibt es dennoch zentrale Probleme:

- Eine Browser-Anfrage mit HTTP-POST-Formulardaten kann nicht zwischengespeichert werden. Die meisten Spammer verwenden jedoch diese Daten, und daher wird für fast keinen Blog-Aufruf durch einen Spammer das Caching-Plugin aktiv. Daher wird also nach wie vor viel an Ressourcen an Spammer verschwendet.
- Die Caching-Funktionalität wird von einem Plugin eingebunden. Damit ein Plugin überhaupt aktiv werden kann, muss Serendipity schon zahlreiche Aktionen ausführen, die bereits viele Ressourcen verschlingen.
- Durch das Caching einer vollständigen Seite verliert das Blog zahlreiche Features für

viele dynamische Plugins. Beispielsweise können Artikel, die nur von eingeloggten Besuchern gelesen werden können, nicht mehr dargestellt werden – denn für Serendipity gibt es nun nur noch einen einzigen Besucher, dessen Login-Status nicht mehr abgefragt werden kann.

Ein technisch sinnvolles Caching, das dennoch die Flexibilität von Serendipity beibehält, ist wohl nach heutigem Stand kaum möglich. Sollten Sie Interesse an der Ausarbeitung anderer Caching-Konzepte haben, werden Sie bei den Serendipity-Entwicklern sicher auf ein offenes Ohr stoßen, denn Mitarbeit in diesem Bereich wird stark benötigt.

6.5.5 Community Rating serendipity_event_communityrating

Das *Community Rating*-Plugin bietet die Möglichkeit, einen Blog-Artikel zu einem Meinungsbericht aufzuwerten. Wenn Sie einen Artikel über einen Film, ein Produkt oder über bekannte Personen schreiben, können Sie den Artikel gezielt über ein eigenes Feld bewerten und diese Wertung an zentraler Stelle anzeigen.

Außerdem können die eigenen Bewertungen auch anderen Blogs zur Verfügung gestellt werden, indem auf Meinungsbeiträge anderer Blogger zum selben Produkt hingewiesen wird.

Das Plugin agiert daher auf mehreren Ebenen und ist mitunter komplex einzurichten. Die folgenden Anweisungen sind daher eher für fortgeschrittene Anwender konzipiert, Anfänger sollten vorher das Kapitel 9.3 ab Seite 480 über das Smarty-Templating lesen.

In der Konfiguration des Plugins vergeben Sie eine (frei wählbare) Namensliste von Wertungsmöglichkeiten. Dabei ist es notwendig, ein Produkt später eindeutig zu identifizieren, um eine in der Community einheitliche Zuordnung herzustellen. Das Plugin schlägt daher standardmäßig die Wertungsmöglichkeiten bei der IMDb[44] und von Produkten bei Amazon[45] vor. Sie können hier grundsätzlich jeden Bewertungstyp eintragen, benötigen aber später eine zugehörige Webseite mit einer eindeutigen ID des bewerteten Produkts. Ein Bewertungstyp darf keine Sonderzeichen (Umlaute oder Leerzeichen) enthalten.

Sobald Sie freie Namen für gewünschte Produktanbieter gewählt haben, können Sie einen Blog-Artikel erstellen. Im Abschnitt *Erweiterte Optionen* sehen Sie für jeden konfigurierten Produktanbieter ein Eingabefeld **ID** und ein Eingabefeld **Rating**. In das ID-Eingabefeld tragen Sie die eindeutige ID ein, die das bewertete Produkt bei dem Anbieter identifiziert. Im Feld **Rating** bestimmern Sie ein Bewertungskriterium (Schulnoten von 1 bis 6, Punkte von 1 bis 10 oder sogar Sternchen von * bis ***). Sinnvoll ist die Vergabe von Zahlen, da diese mathematisch verarbeitet werden können.

Sobald der Artikel gespeichert wurde, ist das Produkt in der Datenbank mit seiner ID zu dem jeweiligen Produktanbieter mit einer Punktzahl verbunden. Sie können mittels Anpassung der eigenen Template-Dateien (siehe Kapitel 9.3 ab Seite 480) folglich an gewünschter Stelle

[44] Internet Movie Database, `http://imdb.com/`
[45] `http://www.amazon.de/`

(also auch in RSS-Feed-Template-Dateien!) eine Smarty-Funktion namens {`community-rating_show`} einbinden. Diese Funktion kann auch dafür sorgen, dass Punktzahlen in grafische Icons umgewandelt werden. Diesem Funktionsaufruf kann man folgende Parameter mitgeben:

`type`
: Der Parameter bestimmt den Produktanbieter, den Sie anzeigen wollen. Für jeden Produktanbieter (wie `IMDB` und `Amazon`) müssen Sie {`communityrating_show type="IMDB"`} und {`communityrating_show type="Amazon"`} einzeln einbinden.

  Für jeden verwendeten *type* muss es in Ihrem Template-Verzeichnis oder dem Verzeichnis des Community-Rating-Plugins eine Datei namens `communityrating_`*`type`*`.tpl` geben, die bestimmt, wie der jeweilige Block eines Produktanbieters formatiert wird.

`data`
: Der Parameter `data` muss das Array der freien Artikeleigenschaften (*Entry-Properties*) enthalten. Diese Variable wird von Serendipity bereitgestellt, daher können Sie einfach darauf zugreifen. Falls Sie den Aufruf der Produktbewertung unterhalb von {`$entry.body`} in der Datei `entries.tpl` einbinden, können Sie {`communityrating_show data=$entry.properties`} einfügen.

`url`
: Mit dieser Variable können Sie die URL eines fremden Blogs übergeben, bei dem das Community-Rating-Plugin ebenfalls zum Einsatz kommt. So können Sie von fest definierten Blogs deren Produktbewertung einbinden. Das Plugin kann nicht automatisch erkennen, welche fremden Meinungen Sie einblenden möchten, daher müssen Sie für jedes fremde Blog einen eigenen Aufruf der Smarty-Funktion benutzen:

  ```
  {communityrating_show
    url="http://example.com/serendipity/plugin/communityrating"
    data=$entry.properties
    who="garvin"
    type="IMDB"
  }
  ```

  Die URL entspricht dabei immer der Serendipity-Plugin-URL plus `/plugin/communityrating`. Bei Serendipity-Blogs ohne URL-Umformung wäre dies `index.php?/plugin/communityrating`.

  Der Aufruf dieser URL mit den Parametern für den gewünschten Typ und die Produkt-ID wird dann bei dem jeweiligen Blog die XML-Daten ausliefern, die das Plugin benötigt, um die fremde Bewertung einzubinden.

`who`
: Dieser Parameter wird in Zusammenhang mit dem Parameter *url* benötigt. Wenn ein fremder Produktbericht angezeigt wird, soll dieser üblicherweise mit einem anderen

Inhalt oder einem anderen Layout angezeigt werden als der eigene Produktbericht. Daher benutzt das Plugin in so einem Fall nicht mehr nur die Template-Datei `communitytyrating_IMDB.tpl`, sondern `communityrating_IMDB_`*`who`*`.tpl`, in der man gezielt auf das jeweilige fremde Blog Formatierungen vornehmen kann.

`path`

Dieser Parameter enthält den Pfad, in dem die Bewertungsbilder gespeichert werden. Die Bewertungsbilder müssen in einem Unterverzeichnis `img` liegen und der Benennung `star_`*`Typ_Zustand`*`.png` entsprechen. Für den Produktanbieter *IMDB* müssen die Dateien `img/star_IMDB_full.png`, `img/star_IMDB_half.png` und `img/star_IMDB_zero.png` existieren. Die *full*-Grafikdatei enthält ein Symbol, das einen vollen Bewertungspunkt repräsentiert. *half* gibt einen halben Bewertungspunkt aus und *zero* einen leeren.

Das Standard-Bewertungssystem ist darauf ausgelegt, dass Sie im *Rating*-Feld für jeden Artikel eine Zahl von 0 bis 10 vergeben. Jeweils zwei volle Punkte ergeben einen Stern, ein einzelner Punkt entspricht einem halben Stern, und fehlende Punkte werden mit leeren Sternen dargestellt. Es werden daher immer fünf Sterne angezeigt, die entsprechend des vergebenen Ratings gefüllt sind. Sie können innerhalb der Template-Dateien selbständig für die grafische Formatierung von Bildern sorgen, wenn Sie ein eigenes System benutzen möchten.

`escaped`

Wenn Sie den Parameter `escaped=true` setzen, wird die Smarty-Funktion die Rückgabe der Template-Datei im UTF-8-Format mit escapten HTML-Sonderzeichen anleiten. Setzen Sie diesen Parameter, wenn Sie die Smarty-Funktion innerhalb eines RSS-Feeds aufrufen.

Somit dient die zentrale Smarty-Funktion dazu, sowohl fremde als auch eigene Produktbewertungen einzubinden. Pro Aufruf kann immer nur eine Bewertung zu einem Produktanbieter von einem Benutzer eingebunden werden, daher sind mehrere Aufrufe der Smarty-Funktion üblich. Wenn fremde Meinungen eingebunden werden, speichert das Plugin diese Daten für eine Woche im Verzeichnis `templates_c` zwischen, damit die fremde URL nicht zu häufig aufgerufen wird und dort Serverlast erzeugt.

Jeder Aufruf der Funktion wertet die zugehörige Template-Datei `communityrating_XXX.tpl` aus. Innerhalb dieser Datei können Sie beliebige Smarty- und HTML-Syntax einsetzen. Es stehen folgende Smarty-Variablen zur Verfügung, die sich jeweils auf das betreffende Produkt des jeweiligen Produktanbieters beziehen:

`communityrating_images`

Diese Variable enthält den HTML-Code für die zusammengesetzte Sternchen-Grafik mit fünf Sternen, die anhand des Ratings gefüllt oder leer sind.

`communityrating_rating`

In dieser Variable ist das *Rating* des Produktes gespeichert. Hier kann Ihre individuelle

Produktbewertung stehen, falls Sie statt der Punktzahl von 0 bis 10 lieber Noten von 1 bis 6 vergeben.

`communityrating_type`
: Diese Variable enthält den angeforderten Produktanbieter (bspw. `IMDB` oder `Amazon`).

`communityrating_id`
: Die ID des bewerteten Produkts wird in dieser Variable gespeichert.

`communityrating_foreign_url`
: Wenn ein fremder Produktbericht angezeigt wird, enthält diese Variable die URL des fremden Blogs, wo Sie den bewerteten Artikel finden.

Sehen Sie sich die dem Plugin beiliegenden `communityrating_.*`-Dateien an, um zu sehen, wie die Ausgaben üblicherweise formatiert werden.

6.5.6 Microformats serendipity_event_microformats

Ähnlich wie das Community-Rating-Plugin ist *Microformats* eher ein Spielzeug für fortgeschrittene Benutzer. *Microformats* sind eine moderne Möglichkeit, um Metadaten zu bestimmten Objekten in einer HTML-Seite einzubinden. Objekte können Produkte, Termine, Events, Geburtstage, Filme und alles andere darstellen. Die Metadaten zu diesen Objekten ergeben sich daraus – sie können eine Produktmeinung, eine Bewertung, weiterführende Infos, kleine Bilder und vieles mehr enthalten. Auf der Webseite `http://microformats.org/` sind viele dieser Formate aufgeführt.

Das Serendipity-Plugin namens *Microformats* bindet zwei Microformats an: *hCalendar* und *hReview*. Für beide Formate gibt es Browser-Plugins, die die eingegebenen Metadaten auswerten und speziell darstellen können.

hCalendar ermöglicht die Darstellung von Terminen und bietet als Metadaten die Felder: Titel (*summary/title*), Ort (*Location*), URL, den Anfangs- und Endzeitpunkt des Termins und eine Beschreibung (*Description*).

hReview erlaubt die Bewertung beliebiger Produkte. Als Metadaten dafür stehen zur Verfügung: Produktname, Produkttyp (Auswahlfeld für *Produkt, Geschäft (Business), Event (Veranstaltung), Person, Ort, Webseite, URL*), URL des bewerteten Gegenstandes, eine Bewertung in Punkten, eine Zusammenfassung, eine vollständige Bewertung, der Zeitpunkt der Bewertung und der Name des Bewerters.

Darüber hinaus kann man das Plugin mit weiteren Microformats später relativ einfach aufrüsten. Der Vorteil von solchen Microformats liegt darin, dass die Metadaten von Programmen (auch Suchmaschinen) gelesen werden können und man so leicht eine universelle Produktdatenbank erstellen kann. Je mehr Benutzer solche Metadaten angeben, desto einfacher kann man in Zukunft z. B. alle Meinungsberichte zum Apple iPhone suchen und eine Durchschnittsbewertung bilden. Für Ihre Leser hat es den Vorteil, dass sie mittels Browser-Plugins schnell

Termine in ihre eigene Termindatenbank aufnehmen könnten. Die Möglichkeiten der Microformats stecken noch in den Kinderschuhen, und die Zukunft bringt hier womöglich noch viel mehr Flexibilität. Eine Initiative, dieses Format bei Bloggern populär zu machen, ist das *StructuredBlogging*-Projekt auf `http://structuredblogging.org/`.

Alle Microformats können die Metadaten im Frontend beliebig formatiert darstellen. Dies kann man mittels der Smarty-Template-Dateien beeinflussen, die das Plugin in einem Standardformat mitliefert.

Das Microformats-Plugin bindet die Erfassung der Metadaten in die Erstellungsmaske für einen Blog-Artikel im Bereich *Erweiterte Optionen* ein. Dort können Sie zwei Auswahlboxen aktivieren, die einem Artikel jeweils ein *hCalendar*- oder *hReview*-Objekt (oder beides) zuordnen. Wenn Sie eine solche Box aktivieren, werden die Eingabeboxen darunter eingeblendet.

Damit die Metadaten, die Sie eingetragen haben, später im Frontend dargestellt werden können, müssen Sie eine Smarty-Funktion an der gewünschten Stelle einfügen. Dies geschieht üblicherweise über die Template-Datei `entries.tpl` in der Nähe von `$entry.body`. Dort müssen Sie für jedes Microformat die Smarty-Funktion {`microformats_show`} aufrufen:

```
{microformats_show data=$entry.properties type="hReview"}
{microformats_show data=$entry.properties type="hCalendar"}
```

Der Aufruf dieser Funktion sorgt dafür, dass das Array mit den Eintrags-Metadaten (`$entry.properties`) an die Template-Datei des jeweiligen Microformats (`hCalendar.tpl` oder `hReview.tpl`) weitergereicht wird. Die jeweilige Template-Datei stellt dann die Metadaten in beliebiger Formatierung dar, die jeweils verfügbaren Variablen können Sie am einfachsten dieser Datei entnehmen.

Analog zum Community-Rating-Plugin lassen sich die Microformats in RSS-Feed-Template-Dateien einbinden, indem Sie dem Smarty-Funktionsaufruf den Parameter `escaped=true` hinzufügen (siehe Seite 381).

In den Konfigurationsoptionen des Microformat-Plugins können Sie Folgendes festlegen:

Subnode hinzufügen
: Nur wenn Sie diese Option aktivieren, bindet das Microformats-Plugin die XML-Metadaten ein, die Browser-Plugins benötigen, um Ihre eingegebenen Metadaten anzuzeigen. Die Einbindung erfolgt *inline*, was bedeutet, das XML wird direkt innerhalb der HTML-Seite eingebunden. Je nach Struktur Ihrer Webseite kann das jedoch dazu führen, dass die Webseite nicht mehr XHTML-konform ist. Die Struktur der XML-Metadaten wird ebenfalls über die Template-Dateien `hCalendar.tpl` bzw. `hReview.tpl` bestimmt. Wenn sich diese Microformats also in Zukunft kompatibler einbinden lassen, kann man die Art der Einbindung so überarbeiten.

Zeitzone
: Für *hCalendar*-Einträge ist die Zeitzone, auf die sie sich beziehen, sehr wichtig, damit

internationale Benutzer einen Termin in ihre Zeitzone umrechnen können. Stellen Sie daher im Auswahlfeld **Zeitzone** die korrekte Zone ein, in der Sie leben. Für Deutschland ist dies **+1 (CET)**.

Maximale Punktzahl
hReview-Einträgen können Sie bewerten. Die maximal von Ihnen vergebene Punktzahl müssen Sie hier eintragen. Beachten Sie dabei, dass die Punktzahl mit Dezimalstellen angegeben werden sollte (damit auch 2.5 Punkte möglich sind).

Punktabstände
Die Abstände Ihrer Bewertungen müssen Sie hier eintragen. Wenn Sie also nur ganze Punkte vergeben, tragen Sie 1.0 ein, andernfalls können auch Werte wie 0.5 für Halb-Punktsprünge eingetragen werden.

Zusammen mit dem Microformat-Plugin wird auch ein Seitenleisten-Plugin namens *Kommende Termine* mitgeliefert. Mit diesem können Sie von Ihnen eingetragene *hCalendar*-Termine in der Seitenleiste darstellen und auch manuell zusätzliche Termine (z. B. von `http://upcoming.org` eintragen.

Wenn Sie eigene Microformats einbinden wollen, müssen Sie dazu folgende Veränderungen vornehmen:

- Eigene `hMicroformat.tpl` erstellen.
- Die Plugin-Datei `microformatsBackend.inc.php` anpassen und dort einen Eingabe-Abschnitt für das neue Microformat einbinden.
- In der Plugin-Datei `serendipity_event_microformats.php` an einigen Stellen die Besonderheiten des Formats festlegen. Folgende Methoden müssen dazu angepasst werden: `getSupportedProperties()` (`$supported_properties`), `addProperties()` (`$supported_formats`), `case 'backend_preview'` (`$supported_formats`), `case 'backend_display'` (`$mf_exist, $itemtypes`).
- In der Smarty-Funktionsdatei des Plugins (`smarty.inc.php`) muss bei `microformats_serendipit` für das neue Format die Variable `$params['mf_type']` abgefragt werden.

Kapitel 7

Gekoppelte Plugins

Gekoppelte Plugins bestehen sowohl aus einem Seitenleisten-Plugin als auch aus einem Ereignis-Plugin. Beide Plugins können dabei auch im selben Verzeichnis liegen. Wenn ein Plugin ein anderes bedingt, kann bei der Installation des einen Plugins auch automatisch ein anderes zugehöriges Plugin installiert werden. Generell sieht Serendipity dabei das Ereignis-Plugin als das wichtigere an – wird dieses entfernt, wird auch das zugehörige Seitenleisten-Plugin (falls Sie es installiert haben) entfernt. Umgekehrt wird ein benötigtes Ereignis-Plugin automatisch installiert, wenn Sie ein zugehöriges Seitenleisten-Plugin installieren.

7.1 Standardmäßig verfügbare Plugins

In der folgenden Liste sind alle miteinander verkoppelten Plugins aufgeführt, die in der Serendipity-Distribution mitgeliefert werden. Keine dieser Plugins sind standardmäßig installiert, daher müssen Sie diese gezielt installieren.

7.1.1 Creative Commons, Creative Commons-Lizenz serendipity_plugin_creativecommons, serendipity_event_creativecommons

Wenn Sie in Ihrem Blog einen Artikel der Allgemeinheit zugänglich machen, liegen die Urheberrechte meist bei dem Redakteur des jeweiligen Artikels. In kommerziellen Blogs gehen die Nutzungsrechte dafür üblicherweise in Firmenbesitz über.

Die Thematik der Eigentums- und Nutzungsrechte von Blogs wird vielerorts ausgiebig diskutiert – als Fazit kann man sagen, dass Sie sich auch als privater Blogger Gedanken über die Verwertungsrechte Ihrer Artikel machen sollten.

Wenn Sie beispielsweise eine ausführliche Dokumentation zu einem Open-Source-Programm

schreiben, könnte an diesem Artikel großes Interesse der Allgemeinheit bestehen. Nun haben Sie als Verfasser zwei Möglichkeiten: Entweder Sie verbieten Fremden, Ihren Artikel weiterzuverbreiten, oder Sie erlauben die Verbreitung Ihres Textes in beliebigen Medien.

Die erste Variante hat für Sie den Vorteil, dass Sie Herr über Ihren Artikel bleiben und möglicherweise auch durch Werbeschaltung etwas Geld verdienen können. Auf der anderen Seite werden erfahrungsgemäß trotzdem viele Benutzer Ihren Artikel einfach weiterverbreiten, und Sie müssen Rechtsmittel gegen diese Nutzung einlegen.

Die zweite Variante, die Offenlegung Ihres Artikels, hilft der Verbreitung Ihres Textes. Wenn Sie hauptsächlich aus ideologischen Gründen daran interessiert sind, Informationen zu verbreiten („*Das Internet ist frei*“), ist dies sicher die beste Maßnahme, damit Ihr Text Gehör (oder *Geles*) findet. Andererseits könnten böswillige Nutzer Ihren Artikel auf kommerziellen Webseiten als Eigenleistung darstellen. Da Sie den Text aber zur freien Vervielfältigung freigegeben haben, sind Ihre Rechtsmittel bei derartiger Nutzung recht eingeschränkt.

Beide Varianten haben eines gemeinsam: Sie stellen eine Lizenzierung dar. Sie sollten sich stets für irgendeine Lizenz Ihrer Blog-Inhalte entscheiden, um klare Verhältnisse zu schaffen.

Es gibt zahlreiche Lizenzen im Internet: Die GPL, BSD, Mozilla-Lizenz und viele weitere sind grundsätzlich zwar eher für Software gedacht, können aber durchaus auch auf eigene Werke wie Blog-Artikel ausgeweitet werden.

Weil Artikel sich aber auch im deutschen Recht von Software unterscheiden, gibt es spezielle Lizenzen, die sich auf Texte und eigene Bilder spezialisiert haben. Die wohl verbreitetste und bekannteste ist die sogenannte *Creative Commons*-Lizenz[1]. Diese bietet ein bausatzartiges Modell, bei dem Sie sich Ihre Lizenz aus einigen Komponenten zusammenstellen können. Sie können auswählen, ob Ihr Text kommerziell eingesetzt werden darf, ob Ihr Urheberrecht jederzeit mit angegeben werden muss und ob auf Ihrem Artikel aufbauende Texte erstellt werden dürfen.

Sie können sowohl das gesamte Blog unter eine derartige Lizenz stellen, als auch ausgewählte Bereiche des Blogs. Um diese Kennzeichnung zu erleichtern, bietet Serendipity das *Creative Commons*-Plugin an.

In dem Ereignis-Plugin stellen Sie ein, für welche Lizenz Sie sich entschieden haben. In dem RSS-Feed Ihres Blogs und in den HTML-Metatags wird das Plugin daraufhin Lizenzhinweise einbinden, die der gewählten Lizenz entsprechen.

Das Seitenleisten-Plugin können Sie einbinden, um in der Seitenleiste für Besucher sichtbar anzuzeigen, welcher Lizenz ihr Blog unterliegt.

Grundsätzlich können Sie für Ihr Blog auch jede andere beliebige Lizenz einbinden. Dazu können Sie einen HTML-Klotz als Seitenleisten-Plugin einbinden und dort auf Ihre gewählte Lizenz hinweisen. Die HTML/RSS-Metadaten können Sie über die Template-Dateien `feed*.tpl` sowie die `index.tpl` ebenfalls manuell einpflegen.

[1] `http://de.creativecommons.org/`

7.1.2 Template dropdown, Template-Auswahl serendipity_plugin_templatedropdown, serendipity_event_templatechooser

Als Eigentümer des Blogs legen Sie üblicherweise fest, wie es auszusehen hat. Manchen Betreibern ist das aber relativ egal; sie legen eher Wert darauf, dass sich der Besucher mit dem Design identifizieren kann. Daher ist es in manchen Fällen angebracht, dem Besucher eine Wahl des Designs (*Template*) zu überlassen – besonders, wenn Sie Wert auf Barrierefreiheit legen.

Über das Seitenleisten-Plugin *Template dropdown* können Sie ein Ausklappfeld einbinden, das alle in Ihrem Blog verfügbaren Templates enthält. Der Besucher kann ein Template auswählen und danach das Blog im gewünschten Layout ansehen.

Das Seitenleisten-Plugin dient lediglich der Darstellung der verfügbaren Templates. Damit das Template erfolgreich aktiviert werden kann, muss das gekoppelte Ereignis-Plugin *Templateauswahl* installiert werden.

Das Ereignis-Plugin setzt einen Browser-Cookie, um die Template-Auswahl des Besuchers auch beim nächsten Besuch wiederherzustellen. Darüber hinaus können Sie, sobald das Ereignis-Plugin installiert ist, Ihr Blog mittels `http://www.example.com/index.php?user_template=`*`default`* aufrufen. Der Option `user_template` können Sie dabei den Namen des gewünschten Template-Verzeichnisses zuweisen.

Bitte beachten Sie: Wenn Sie eigenständige Anpassungen an einem Template vornehmen, um beispielsweise spezielle erweiterte Eigenschaften einzubinden, sind diese nur in dem jeweils von Ihnen modifizierten Template sichtbar. Wenn ein Besucher ein davon abweichendes Template gewählt hat, werden möglicherweise wichtige Änderungen bei ihm nicht korrekt dargestellt. Auch wenn Sie das Plugin *Eigenschaften/Templates von Kategorien* verwenden, sollten Sie eine freie Template-Auswahl nur in Sonderfällen zulassen.

7.1.3 Statistiken serendipity_plugin_statistics, serendipity_event_statistics

Einen sehr wichtigen Haushaltsgegenstand eines jeden Profi-Bloggers stellt der Bauchpinsel dar. Mit diesem Instrument lässt sich das Ego eines Bloggers vorzüglich bürsten, was letztlich die beruhigende Selbstbestätigung für den Betrieb eines Blogs gibt.

Trotz dieser etwas ironischen einleitenden Worte ist es für viele Blogger tatsächlich sehr wichtig, Statistikwerte über ihr Blog zu sammeln. Ob dies später als persönlicher Ansporn dient oder als Marketinginstrument der Geschäftsführung, ist dem Einzelnen überlassen.

Serendipity liefert ein Statistik-Plugin mit. Das Seitenleisten-Plugin kann für Besucher wertvolle Informationen darstellen, wie beispielsweise die aktuelle Anzahl an Artikeln, das Datum des aktuellsten Artikels, die Anzahl der Kommentare und die Anzahl der Besucher im

Monat und aktuell auf der Seite.

Das Ereignis-Plugin bindet eine ausführliche Statistik für Redakteure im Backend unter **Einträge** → **Statistiken** ein.

Abgesehen von diesem internen Plugin macht es durchaus Sinn, weitere Möglichkeiten der Statistikerhebung auf dem Webserver zu benutzen, allen voran die Analyse von Webserver-Logfiles mittels Programmen wie awStats[2], webalizer[3] oder Modlogan[4]. Auch das Google Analytics[5] Widget hat sich als extrem detailreich erwiesen (und es lässt sich einfach via JavaScript oder eigenständigem Serendipity-Plugin einbinden).

In der Konfiguration des Seitenleisten-Plugins können Sie gezielt einstellen, welche statistischen Daten Sie Ihren Besuchern darstellen wollen. Die Bezeichnung dafür können Sie frei vergeben. Als Platzhalter für die später dargestellte Zahl verwenden Sie `%s`. Damit das Plugin nicht jedes Mal erneut die Statistiken abfragen muss, werden die Ergebnisse temporär zwischengespeichert (`Caching`). Über die Option **Cache-Zeitlimit** können Sie einstellen, wie viel Zeit vergehen darf, bevor der Cache neu erstellt wird.

Die Anzahl der monatlichen und der aktuellen Besucher stellt das Plugin nur dar, wenn Sie im Ereignis-Plugin die Option **Erweiterte Besucherstatistiken** aktiviert haben. Als *aktueller Besucher* zählt in diesem Fall jeder Besucher, der in den letzten 15 Minuten auf das Blog zugegriffen hat.

Die Konfiguration des Ereignis-Plugins bietet folgende Optionen:

Anzahl Einträge
: Standardmäßig zeigt das Plugin im Menüpunkt **Einträge** → **Statistiken** immer 20 Datensätze pro statistischem Wert an. Wenn Sie mehr oder weniger Einträge sehen wollen, können Sie dies mit der Option **Anzahl Einträge** festlegen.

Erweiterte Besucherstatistiken
: Wenn Sie diese Option aktivieren, kann das Plugin jeden Zugriff eines Besuchers auf das Blog nachverfolgen (*visitor tracking*). Ein Besucher wird mittels eines Cookies eindeutig identifiziert, alle Folgezugriffe aktualisieren dann einen bestehenden Datensatz.

  Im Gegensatz zu den Statistikfunktionen des **Karma**-Plugins werden so globale Besucher erfasst, und *nicht* die Klicks bei individuellen Artikeln. Daher schließen sich beide Plugins gegenseitig nicht aus, sondern können ergänzend eingesetzt werden.

  Ob Sie diese Option auf **Ja, am unteren Ende der Seite** oder **Ja, oben auf der Seite** stellen, ist nur für die Darstellung der Statistik erheblich und gibt an, wo die Besucherstatistiken eingebunden werden sollen.

  Wenn die Benutzerzählung aktiviert ist, sollten Sie von Zeit zu Zeit manuell die Datenbanktabellen `serendipity_visitors` und `serendipity_visitors_count`

[2] `http://awstats.sourceforge.net/`
[3] `http://www.mrunix.net/webalizer/`
[4] `http://modlogan.org/`
[5] `http://www.google.com/analytics/`

prüfen, da diese sehr groß werden können. Schlagen Sie auf Seite 443 nach, um zu erfahren, wie Sie derartige Prüfungen gezielt durchführen.

Alles zeigen
Über die Option **Alles zeigen** können Sie festlegen, ob das Statistik-Plugin sämtliche Daten anzeigt (Einstellung **Ja, alle Statistiken anzeigen**) oder ob nur die erweiterten Besucherstatistiken (falls aktiviert) eingebunden werden sollen.

Robot-Zählung verhindern
Nur wenn Sie diese Option aktivieren, kann Serendipity bekannte Suchmaschinen von der Besucherzählung ausnehmen. Meist macht es für Sie keinen Sinn, Suchmaschinen in dieser Zählung aufzuführen, daher ist es selten zu empfehlen, die Option auf **Nein, Robots bitte mitzählen** zu setzen.

Die Darstellung des Statistik-Plugins umfasst folgende Daten:

- Anzahl der Besucher (monatsbezogen, tagesbezogen)
- Letzte Besucher
- Top-Referrer[6]
- Datum des ersten und letzten Blog-Artikels
- Anzahl insgesamt verfasster Artikel (veröffentlicht, Entwürfe)
- Anzahl der Artikel vereilt auf einzelne Redakteure
- Anzahl vorhandener Kategorien
- Verteilung der Artikel auf die vorhandenen Kategorien
- Anzahl der Dateien in der Mediendatenbank, Aufteilung nach Dateityp
- Anzahl der Kommentare, Verteilung der Kommentare zu den populärsten Artikeln, Namen der häufigsten Kommentatoren
- Anzahl der Abonennten von Blog-Artikeln, Verteilung der Abonnenten auf populäre Artikel. Als Abonnent wird ein Kommentator bezeichnet, der beim Kommentieren die Option **Bei Aktualisierung dieser Kommentare benachrichtigen** aktiviert hat.
- Anzahl und Verteilung der Trackbacks auf populäre Artikel, Namen der am häufigsten Trackbacks sendenden Blogs
- Durchschnittliche Kommentare, Trackbacks pro Artikel
- Durchschnittliche Artikel pro Tag, Woche und Monat

[6] Referrer sind Webseiten, von denen aus Besucher zu Ihrem Blog gelangt sind.

- Menge der insgesamt geschriebenen Zeichen, durchschnittliche Zeichenzahl pro Artikel, Nennung der längsten Blog-Artikel
- Top-Referrer und Top Exits (siehe zugehörige Plugins auf den Seiten 258 und 168)

Weiterhin können etwaige weitere Plugins auf dieser Seite ihre eigenen Statistiken einbinden, wie beispielsweise das Karma-Plugin.

Datenbanktabellen

Die Tabelle `serendipity_visitors` enthält für jeden Besucher der Seite einen Eintrag:

`counter_id`
: enthält eine fortlaufende ID.

`sessID`
: enthält die Session-ID des Besuchers, damit nur der erste seiner Aufrufe gezählt werden muss.

`day`
: enthält das Datum des Besuchs im Textformat.

`time`
: enthält das Datum des Besuchs im UNIX-Zeitstempel-Format.

`ref`
: enthält den HTTP-Referrer (Verweisende Seite).

`browser`
: enthält den Browser-Typ des Besuchers.

`ip`
: enthält die IP des Besuchers.

Die Datenbanktabelle `serendipity_visitors_count` enthält eine Zusammenfassung der Besucher an einem einzelnen Tag:

`year`
: enthält das Datum (Jahr) der Zugriffe.

`month`
: enthält das Datum (Monat) der Zugriffe.

`day`
: enthält das Datum (Tag) der Zugriffe.

`visits`
: enthält die Anzahl der Besucher an diesem Tag.

`hits`
: enthält die Gesamtzahl an aufgerufenen Seiten an diesem Tag.

Die Referrer werden in der Tabelle `serendipity_refs` gespeichert. Neben dem fortlaufenden Primärschlüssel `id` enthält die Spalte `refs` die URL der Webseite und `count` die Anzahl der Besucher von dieser Seite.

Diese Tabellen können auf großen Blogs sehr umfangreich werden. Warten Sie diese daher regelmäßig.

7.2 Auswahl externer Plugins

Abgesehen von den mitgelieferten gekoppelten Plugins finden Sie auch eine große Zahl an Plugins über `http://spartacus.s9y.org/`. Eine Auswahl an häufig gekoppelten Ereignis-Plugins finden Sie auf den folgenden Seiten.

7.2.1 Geotag Google Map, Geotag serendipity_plugin_geotag, serendipity_event_geotag

Als *Geotagging* bezeichnet man den Vorgang, zu einem Artikel (oder auch einem Bild) einen geographischen Bezug (*Meta-Informationen*) zuzuordnen.

Diese Meta-Informationen sind oft für Besucher gar nicht offensichtlich, da sie im XHTML-Code versteckt sind und nur von Schnittstellen oder Browser-Plugins ausgewertet werden. Meta-Informationen (oder auch *Microformats*) sind die Grundidee der nächsten Evolutionsstufe des Internets, genannt *Semantic Web*. Anhand klar strukturierter, maschinenlesbarer Informationen können Suchmaschinen Details auswerten und miteinander verknüpfen.

Konkret kann dies bedeuten, dass man mit einer Suchmaschine alle Blog-Artikel suchen kann, die im Starbucks-Cafe am Kölner Hauptbahnhof verfasst wurden; oder alle Blog-Einträge, die sich auf die Semperoper beziehen. Der sinnvolle Umgang mit den Metadaten eröffnet zahlreiche Möglichkeiten und Verkettungen von Daten, die ein eigenes Buch zu diesem Thema rechtfertigen würden.

Alle diese Such- und Verkettungsmöglichkeiten werden aber erst dann nutzbar, wenn die Informationen überhaupt von Autoren zur Verfügung gestellt werden. Diese redaktionelle Leistung kann Ihnen derzeit noch keine Maschine abnehmen.

Aber die Eingabe kann Ihnen erleichtert werden, und aus diesem Grund wurde das Serendipity-Plugin `Geotag` erfunden. Wenn Sie dieses Plugin installieren, können Sie in den *Erweiterten Optionen* jeden Blog-Artikels mittels einer Google-Map[7] einem Punkt auf der Welt zuord-

[7] `http://maps.google.com/`

nen. Die geographischen Daten bindet das Plugin daraufhin in Ihrem RSS-Feed und auf der Webseite ein, von wo sie von Suchmaschinen indiziert werden können.

Ein zugehöriges Seitenleisten-Plugin kann in einer Google-Map außerdem die aktuellsten Einträge geographisch zugeordnet darstellen.

Für beide Einsatzzwecke (Seitenleisten- und Ereignis-Plugin) benötigen Sie einen Google API-Schlüssel. Diesen erhalten Sie bei `http://www.google.com/apis/maps/signup.html`, nachdem Sie den Lizenzbedingungen zugestimmt und die URL Ihres Blogs mitgeteilt haben. Ohne den API-Schlüssel kann das Plugin keine geographische Karte darstellen, und Sie müssten die Längen- und Breitengrade zu einem Eintrag per Hand festlegen.

Das Ereignis-Plugin bindet lediglich diese eingegebenen Längen- und Breitengrade in die Metadaten des RSS-Feeds ein (`<geo:long>, <geo:lat>`). Um sichtbare Informationen für Ihre Besucher zu verketten, bietet das Ereignis-Plugin auch die Möglichkeit an, eine Karte zu dem jeweiligen verbundenen Ort anzuzeigen. Hier wird standardmäßig Google Maps eingebunden, aber Sie können auch einen anderen Kartendienst benutzen. In dem Eingabefeld **Karten URL** der Konfiguration des Ereignis-Plugins können Sie die Platzhalter `%GEO_LAT%` und `%GEO_LONG%` für die Längen- und Breitengrade innerhalb der URL einsetzen, und `%TITLE%` wird mit dem Titel des zugehörigen Blog-Eintrages mit den Geodaten ersetzt.

Anschließend erfolgt die Darstellung dieses Links unterhalb jedes Blog-Eintrags im Fuß der Seite, innerhalb eines HTML-Containers mit der CSS-Klasse `div.serendipity_geotag`.

In der Konfiguration des Seitenleisten-Plugins können Sie einige Darstellungsoptionen der Google Map festlegen (Breite, Höhe, Zoomlevel). Das Google Map JavaScript kann die Darstellung einer Karte mit geographischen Punkten anreichern. Als Datenbasis dazu dient der RSS-Feed Ihres (oder auch eines anderen!) Blogs. Diese RSS-URL müssen Sie in der Konfiguration des Plugins festlegen, standardmäßig zeigt die Einstellung bereits auf Ihren RSS-Feed. Als weitere Alternative kann das Plugin auch direkt auf Ihre Datenbank zugreifen und spart so den Umweg über einen RSS-Feed.

7.2.2 Sprachauswahl, Multilinguale Einträge serendipity_plugin_multilingual, serendipity_event_multilingual

Wenn Sie in einem Blog Einträge in unterschiedlichen Sprachen verfassen wollen, können Sie dafür beispielsweise eigenständige Blog-Kategorien einrichten und die Einträge dort entsprechend einordnen. Dabei entstünde jedoch pro Übersetzung eines Artikels ein neuer, eigenständiger Artikel.

Zwar hat dies auch möglicherweise Vorteile, da die Einträge so unabhängig voneinander sind (z. B. für Kommentare), aber schöner wäre es doch, wenn ein Artikel in mehreren Sprachversionen verwaltet werden könnte.

Dies ermöglicht das Ereignis-Plugin *Multilinguale Einträge*. Für mehrsprachige Einträge

empfiehlt es sich, dass Sie Ihr Blog mit UTF-8-Zeichensatz konfiguriert haben. Mit nationalen Zeichensätzen könnte es später Darstellungsprobleme bei Sonderzeichen unterschiedlicher Sprachen geben. Beim UTF-8-Zeichensatz können jedoch sowohl chinesische Sprachzeichen als auch deutsche Umlaute parallel auf derselben Seite angezeigt werden.

Sobald Sie das Ereignis-Plugin installiert haben, bindet es sich in die bekannte Oberfläche zur Erstellung eines Artikels ein. Im Bereich *Erweiterte Optionen* im Abschnitt *Multilinguale Einträge* erscheint bei einem neuen Eintrag ein Hinweistext. Dieser erklärt Ihnen, dass Sie einen Beitrag erst einmal abspeichern müssen, bevor Sie den Artikel in weitere Sprachen übersetzen können.

Erstellen Sie also beispielhaft einen Artikel mit Titel und Inhaltstext. Diesen speichern Sie als *Entwurf* ab (damit er nicht direkt im Frontend erscheint). Sobald Sie diesen Artikel erstmalig gespeichert haben, finden Sie im Abschnitt *Multilinguale Einträge* ein Ausklappfeld und den Button **Sprache wechseln**. Im Ausklappfeld ist anfangs *Standard* ausgewählt.

Intern verhält es sich so, dass Serendipity einen normalen Datenbankeintrag mit den Stammdaten des Artikels sichert. Zu den Stammdaten gehören die Felder `Titel`, `Eintrag`, `Erweiterter Eintrag` sowie weitere Daten wie Artikelzeit und Autor. Diese Stammdaten werden niemals durch das Plugin *Multilinguale Einträge* verändert. Wenn Sie später einen Artikel bearbeiten, wird Ihnen standardmäßig immer der Stammartikel angezeigt.

Als *Standardsprache* gilt die Sprache, die der jeweilige Redakteur in seinen *Eigenen Einstellungen* (also *nicht* die in der globalen Konfiguration eingestellte Blog-Sprache!) festgelegt hat. Dies führt oft zur Verwirrung bei Benutzern des Plugins, daher sollten Sie sich diese Besonderheit gut einprägen. Wenn Sie als Redakteur *Deutsch* als Sprache gewählt haben, so entspricht dies der Standardsprache. Sie können daher im Ausklappfeld die Sprache *Deutsch* nicht auswählen.

Um nun eine Übersetzung des Artikels einzupflegen, müssen Sie erst die gewünschte Zielsprache im Abschnitt *Multilinguale Einträge* eines bestehenden Artikels auswählen und auf den Button *Sprache wechseln* klicken.

Daraufhin speichert das Plugin die aktuelle Version des Artikels und tauscht im Hintergrund die Eingabemaske des Artikels aus. In der Datenbank wird in einer eigenständigen Tabelle (`serendipity_entryproperties`) nun der Datensatz für die Übersetzung des Artikels angelegt.

Damit Sie als Redakteur wissen, was für einen Text Sie übersetzen müssen, wird der Artikeltext des Ursprungsartikels in der Artikeloberfläche angezeigt. Nun können Sie nach Belieben den Titel, den Eintrag und den erweiterten Eintrag überarbeiten. Alle anderen Felder und erweiterten Eigenschaften beziehen sich nach wie vor auf den Stammartikel – Sie können also einen übersetzten Artikel nicht einer anderen Kategorie zuordnen. Hierfür müssten Sie einen eigenständigen Artikel erstellen.

Sobald Sie nach der Übersetzung einen Eintrag speichern, schließt das Plugin die Ergänzung des Stammdatensatzes ab. Sie können daraufhin weitere Sprachübersetzungen ausfüllen, indem Sie weitere Sprachversionen im Ausklappfeld anwählen.

Wenn Sie später einen übersetzten Artikel bearbeiten wollen, sehen Sie wie erwähnt anfangs nur den Stammartikel. Sie müssen daher zur Bearbeitung einer Sprachversion stets mittels des Ausklappfeldes die gewünschte Bearbeitungssprache wählen und auf den Button **Sprache wechseln** klicken. Eine Übersetzung können Sie löschen, indem Sie den Titel, den Eintrag und den erweiterten Eintrag leeren und den Artikel speichern.

Beachten Sie, dass das Plugin nur dann optimal funktionieren kann, wenn Ihre Redakteure alle dieselbe Standardsprache verwenden. Sollten andere Redakteure andere Spracheinstellungen verwenden, kann dies die Verkettung der *Standardsprache* mit einem Artikel durcheinander bringen.

Nachdem Sie nun erfolgreich einen mehrsprachigen Artikel angelegt haben, sollten Sie einen Blick auf die Artikeldarstellung im Frontend werfen. Dort sehen Sie Ihren Artikel in der Standarddarstellung. In der Fußzeile des Artikels ist eine Sprachwahl neu hinzugekommen. Dort können Sie gezielt eine der vorhandenen Sprachen anklicken, damit der Artikel in der gewählten Sprache dargestellt wird. Genauso können Ihre Blog-Besucher später mehrere Sprachversionen des Artikels lesen. Bei der Umstellung auf die Zielsprache wechselt das Frontend zudem seine Sprache, damit der Besucher beispielsweise die Kommentarhinweise in dieser Sprache lesen kann.

Da der Artikel immer über dieselben Stammdaten angesprochen wird, werden auch Kommentare nur diesem Beitrag zugewiesen. Wenn also ein Teil Ihrer Besucher auf Englisch kommentiert und andere auf Deutsch, so werden alle Kommentare später in allen Sprachversionen stets gleichzeitig erscheinen. Wenn Sie dies vermeiden wollen, müssen Sie separate Einträge ohne Verwendung des multilingualen Plugins erstellen.

Ein gekoppeltes Seitenleisten-Plugin ermöglicht es Ihren Besuchern, ihre bevorzugte Sprache auszuwählen. Die Besucher sehen daraufhin alle Einträge in dieser gewählten Sprache, und auch die restlichen Ausgaben Serendipitys erscheinen in dieser Sprache. Sie können in der Konfiguration des Seitenleisten-Plugins gezielt festlegen, welche Sprachen Sie zur Auswahl anbieten wollen.

Die Volltextsuche wird vom Plugin ebenfalls aufgerüstet, so dass bei der Sprachänderung auch jeweils der Artikeltext in der gewählten Sprache anstelle der Standardsprache berücksichtigt wird.

Zwei URL-Variablen bestimmen, wie die Artikelsprachen dargestellt werden. Über die Variable `serendipity[lang_display]=`*`en`* können Sie das Plugin anweisen, in Artikelübersichten ausschließlich Artikel in der übermittelten Sprache (hier `en` = Englisch) darzustellen. Existiert ein Artikel nicht in der gewünschten Übersetzung, wird der Artikel nicht dargestellt. Die URL-Variablen

```
serendipity[lang_selected]=en
serendipity[serendipityLanguage]=en
user_language=en
```

werden synonym verwendet. Es ist daher egal, welche dieser drei URL-Variablen Sie be-

nutzen[8]. Im Unterschied zur Variable `serendipity[lang_display]` legen diese drei lediglich eine Präferenz des Besuchers fest – wenn ein Artikel in der Zielsprache nicht vorhanden ist, sehen Sie die Standardsprache. Auch legt diese letzte Variable die Ausgabe von Sprachvariablen des Serendipity-Frontends fest.

Alle URL-Variablen können Sie auch für RSS-Feeds benutzen, um beispielsweise nur englische Artikel zu erhalten: `http://www.example.com/serendipity/rss.php?serendipity[lang_display]=`

Das Ereignis-Plugin verfügt über zwei Konfigurationsoptionen. **Behalten Sie vorhergehenden Spracheninhalt bei** legt fest, ob beim Wechsel der Sprache zur Eingabe eines Artikels der Stammartikel als Vorlage eingetragen werden soll. Bei deaktivierter Option starten Sie eine Artikelübersetzung mit einem leeren Artikel. Einigen Autoren hilft das, den Durchblick über noch nicht übersetzte Sprachversionen nicht zu verlieren.

Mit der Option **Where to place entry links** legen Sie fest, wo im Frontend die Links für den Besucher angezeigt werden, mit denen er zu Übersetzungen des Artikels wechseln kann. Die Einstellung **Footer of an entry** platziert diese Links im Fußbereich des Artikels. Wenn Sie die Einstellung **multilingual_footer for custom Smarty output** aktivieren, können Sie in Ihrem Smarty-Template `entries.tpl` selbst bestimmen, wo die Links erscheinen sollen. Verwenden Sie dafür die Variable {`$entry.multilingual_footer`}.

7.2.3 Registrierung neuer User serendipity_plugin_adduser, serendipity_event_adduser

Technisch gesehen gibt es keinen Grund, dass Sie Ihr Blog in einsamem Schattendasein mit Inhalten füllen. Gruppen-Blogs mit mehreren Redakteuren könenn den Reiz für Ihre Leser erhöhen, Ihr Blog regelmäßig zu verfolgen. Mehr Redakteure schreiben mehr Artikel, können Themen unterschiedlich beleuchten oder durch einen eigenen Schreibstil verschiedene Gruppen von Lesern anziehen.

Neue Redakteure können Sie relativ leicht über das Serendipity-Backend im Menüpunkt **Administration** → **Benutzerverwaltung** hinzufügen. Dennoch entsteht dadurch ein nicht zu unterschätzender Verwaltungsaufwand: Sie müssen dem Autor Passwort und Benutzernamen zuweisen, ihn einer Gruppe zuordnen und ihm den Link zum Backend mitteilen.

Einfacher geht das alles mithilfe des Plugins **Registrierung neuer User**. Ein Seitenleisten-Plugin bietet ein einfaches Registrierungsformular für Ihren Besucher an, über das er sich selbständig als Autor anmelden kann. Das zugehörige Ereignis-Plugin kümmert sich um die weitere Verarbeitung und sendet eine automatische Registrierungs-E-Mail an den Benutzer. Nach dessen Bestätigung kann der Benutzer sich sofort als neuer Redakteur einloggen und mit den ihm zugeordneten Rechten Artikel erstellen. Und schon ist aus dem einsamen, egozentrischen Blogger ein gemütliches Herdentier geworden.

[8] Aus Kompatibilitätsgründen zu älteren Versionen des Plugins und der Serendipity-Kernversion ist dieselbe Variable mit drei verschiedenen Namen vorhanden.

Ein Besucher, der sich als Redakteur melden will, kann sich in Ihrem Blog auf zwei Arten registrieren. Zum einen können Sie das Seitenleisten-Plugin *Registrierung neuer User* installieren, das in der Seitenleiste ein kleines Eingabeformular einbindet. Dort trägt der Besucher seinen gewünschten Benutzernamen, das Passwort und seine E-Mail-Adresse ein.

Als zweite Möglichkeit bietet das Ereignis-Plugin eine selbständige Seite an, die man unter `http://www.example.com/serendipity/index.php?serendipity[subpage]=adduser` aufrufen kann. Auch hier werden dieselben Eingabefelder angezeigt.

Ob Sie beide oder nur eine Variante der Registrierungsformulare in Ihr Blog einbinden, bleibt Ihren Layout-Vorstellungen überlassen. Sie könnten auch eine eigene statische Seite mit freiem Eingabeformular entwerfen. Dabei muss das HTML-Formularziel lediglich auf die Datei `http://www.example.com/serendipity/index.php` zeigen, und Sie müssen die GET- oder POST-Variable `serendipity[adduser_user]` (Benutzername), `serendipity[adduser_pass]` (Passwort), `serendipity[adduser_email]` (Mail-Adresse) abfragen. Weiterhin müssen die *hidden*-Variablen `serendipity[subpage]=adduser` und `serendipity[adduser_action]=tru` übertragen werden.

Die Konfiguration der Verwaltungsmöglichkeiten verteilt sich auf das Seitenleisten- und das Ereignis-Plugin. Ohne das Ereignis-Plugin kann die Registrierung neuer Redakteure nicht durchgeführt werden. Auch das Seitenleisten-Plugin muss vorhanden sein, da Sie dort zentrale Konfigurationsoptionen festlegen. Jedoch können Sie dieses Seitenleisten-Plugin später im Layout auch durchaus verstecken (siehe Konfigurationsoption **Seitenleisten-Plugin anzeigen** des Seitenleisten-Plugins auf Seite 298 oder über die zentrale Plugin-Verwaltung, siehe Seite 145).

Die Hauptkonfiguration des Seitenleisten-Plugins bietet die Optionen:

Eigene Hinweise
: Im Eingabefeld **Eigene Hinweise** können Sie einen beliebigen Text eintragen, der Ihren Besuchern vor dem Registrierungsformular angezeigt wird. Hier können Sie Bedingungen für die Redakteursaufnahme eintragen oder auch redaktionelle Richtlinien festlegen.

Standard Benutzerlevel, Gruppenzugehörigkeit
: Wenn ein Besucher sich als Redakteur anmeldet, wird für diesen ein eigener Blog-Account erstellt. Seine Zugriffsrechte richten sich nach dem Login danach, welchen Benutzerlevel und welche Benutzergruppe Sie derartigen neuen Redakteuren über die Konfigurationsoptionen **Standard Benutzerlevel** und **Gruppenzugehörigkeit** zuweisen.

  Sie sollten möglichst eine eigene Benutzergruppe für solche *freiwilligen Redakteure* erstellen, damit Sie dieser Gruppe nur die Rechte geben, die sie benötigt (siehe Seite 185).

  Bitte achten Sie darauf, dass Sie von neuen freiwilligen Redakteuren immer *das Beste hoffen und das Schlimmste erwarten* müssen. Nicht jeder Freiwillige ist wohlgesinnt:

Wenn Sie jedem neuen Redakteur erlauben, neue Artikel zu veröffentlichen, könnten Sie ganz schnell Vandalen anlocken, die Ihr Blog verunstalten.

Benutzer deaktivieren/Rechte entziehen

Jeder Benutzeraccount in Serendipity kann über die Option **Benutzer deaktivieren/Rechte entziehen** vorübergehend deaktiviert werden. Wenn bei einem Account dieser Parameter aktiviert ist, kann ein Benutzer sich nur noch in das Backend einloggen und sonst nichts Weiteres mehr tun (siehe Seite 185).

Über die Plugin-Option **Benutzer deaktivieren/Rechte entziehen** können Sie einstellen, ob automatisch hinzugefügte Redakteure standardmäßig *inaktiv* sind.

Diese Option hat zwei Vorteile: Zum einen können Sie so eine manuelle Benutzerkontrolle erreichen. Ein Administrator müsste einen Benutzeraccount dann erst gezielt aktivieren, um ihm redaktionelle Rechte (gemäßig seiner Benutzergruppe) zu geben. Der andere Vorteil ist, dass Sie durch diese Option Benutzeraccounts anlegen können, die nichts anderes dürfen, als sich einzuloggen. Eine solche Benutzergruppe (*registrierter Gast*) kann bei Blogs hilfreich sein, die nur registrierten Autoren Kommentare erlauben oder spezielle Artikel nur registrierten Autoren anzeigen. So können sich Besucher einfach in Ihrem Blog anmelden, und Sie können Spam-Roboter und Suchroboter relativ leicht aus Ihrem Blog verbannen. Einige weitere Optionen des *Registrierung neuer User*-Plugins zielen auf exakt einen solchen Einsatzzweck ab.

Seitenleisten-Plugin anzeigen

Das Seitenleisten-Plugin ist zwar grundsätzlich durch das Registrierungsformular unter `http://www.example.com/serendipity/index.php?serendipity[subpage]=adduser` nicht erforderlich, aber ohne das Seitenleisten-Plugin könnten die wichtigen Konfigurationsoptionen des Registrierungsvorganges nicht festgelegt werden.

Löschen Sie also das Seitenleisten-Plugin nicht, wenn Sie dessen Darstellung nicht sehen wollen, sondern setzen Sie die Option **Seitenleisten-Plugin anzeigen** auf **Nein**.

Straight insert

Standardmäßig erhält ein Redakteur eine E-Mail mit einem Link, mit dem er den Registrierungsvorgang bestätigen muss. Ohne diese Bestätigung, die nur der jeweilige Benutzer anhand eines zufällig erzeugten Schlüssels aufrufen kann, ist sein Account nicht zu benutzen.

Da dieser Vorgang jedoch etwas Zeit und einen Medienbruch erfordert, können Sie auch die Option **Straight insert** aktivieren. Dann wird ein Redakteursaccount sofort aktiv.

Beachten Sie, dass bei aktivierter Option die freie Registrierung wesentlich leichter durch böswillige Benutzer oder Spam-Roboter missbraucht werden kann. Nur in dem Fall, dass Ihr Webserver keine Aktivierungs-E-Mails verschicken kann oder Ihr Blog nur im Intranet läuft, wäre diese Option tatsächlich sinnvoll.

Das gekoppelte Ereignis-Plugin bietet weitere Konfigurationsoptionen:

Eigene Hinweise

Ähnlich wie die Anweisungen für Redakteure im Seitenleisten-Plugin können Sie separate (ausführlichere) Informationen angeben, die das Plugin anzeigt, wenn Sie die URL `http://www.example.com/serendipity/index.php?serendipity[subpage]=adduse` aufrufen.

Nur registrierte Nutzer dürfen Kommentare schicken

Wenn Sie diese Option aktivieren, ist es nur noch eingeloggten Besuchern erlaubt, einen Kommentar zu hinterlassen.

Mittels dieser Plugin-Option können Sie das Spam-Aufkommen für Kommentare deutlich reduzieren. Sie erhöhen jedoch auch deutlich die Barriere, die ein neuer Besucher überwinden muss, um erstmals einen Kommentar zu hinterlassen.

Zusätzlich muss der Kommentator sich einen neuen Benutzernamen und ein eigenes Passwort für Ihr Blog merken. Man kann also mit an Sicherheit grenzender Wahrscheinlichkeit behaupten, dass die Aktivierung dieser Option nicht nur Spam-Kommentare reduziert, sondern auch weniger gültige Kommentare zur Folge hat. Blogs mit einem hohen *Troll-Faktor*[9] wissen es jedoch sicher zu schätzen, nur noch wirklich ernsthafte Kommentare zu erhalten.

Etwas einfacher wird die Registrierung durch den neuen Webservice `OpenID` (`http://openid.net/`). OpenID ist eine technische Schnittstelle, die zentrale Benutzeraccounts verwaltet, auf die andere Webseiten zugreifen können. Ein Benutzer muss nur einmal einen Account für die Schnittstelle erzeugen und kann sich dann auf allen Webseiten authentifizieren, die OpenID unterstützen.

Serendipity kann diese `OpenID`-Schnittstelle unterstützen, wenn Sie das Plugin *OpenID Authentication* installieren. Dieses Plugin ist derzeit noch im Beta-Stadium und kann sich noch stark verändern; es wird daher in diesem Buch nicht weiter vertieft. Zudem ist OpenID als Web-Schnittstelle leider noch nicht sehr weit verbreitet. Sollten Sie sich für dieses Thema interessieren, sei Ihnen das Plugin jedoch wärmstens ans Herz gelegt.

Autoren-Identitäten schützen

Wenn ein Benutzer oder ein Redakteur einen Kommentar zu einem Artikel schreibt, kann er seinen Namen frei ausfüllen. Daher kann grundsätzlich jeder Besucher sich mit einem beliebigen Namen eintragen und unter anderem auch den Namen des Redakteurs *klauen*.

Die einzige Methode, um Benutzernamen einmalig und eindeutig zu verwenden, ist, Kommentare nur von registrierten Autoren zuzulassen und den Namen automatisch anhand des Benutzeraccounts auszufüllen.

Da eine solche Methode aber, wie eingangs erwähnt, auch die Hürde zum Schreiben von Kommentaren höher legen würde, geht das Ereignis-Plugin mit der Option **Autoren-Identitäten schützen** einen Mittelweg.

[9] `http://de.wikipedia.org/wiki/Troll_%28Netzkultur%29`

Wenn Sie diese Option aktivieren, können Namen von bereits vorhandenen Redakteuren nur durch diese Redakteure verwendet werden. Ein anonymer Besucher kann daher nach wie vor kommentieren und sich einen Namen ausdenken – aber er darf keine bestehenden Namen mehr verwenden.

Wenn ein Kommentator einen geschützten Namen verwenden will, teilt das Plugin im Kommentarformular mit, dass er sich vorher mit diesem Namen einloggen muss. Dazu wird ihm ein Link angeboten, mit dem er ein Popup zum Login öffnen kann. Das Popup ist insofern hilfreich, als er dann seinen möglicherweise bereits geschriebenen Kommentar im Browser-Hintergrundfenster offen lassen und nach dem Login erneut abschicken kann.

Das Layout des Login-Fensters können Sie über die Smarty-Template-Datei `plugins/serendipity_plugin_addu` anpassen. Die verfügbaren Template-Variablen in dieser Datei sind:

{`$loginform_add`}
: enthält etwaige Zusatzvariablen für das Login-Formular, z. B. die Ausgaben des Plugins *Passwort vergessen*.

{`$loginform_url`}
: enthält die URL zum Login.

{`$loginform_user`}
: enthält den vom Besucher eingetragenen Benutzernamen (falls eingeloggt).

{`$loginform_mail`}
: enthält die vom Besucher eingetragene E-Mail-Adresse (falls eingeloggt).

{`$close_window`}
: ist auf `true` gesetzt, wenn der Login erfolgreich war und das Popup-Fenster geschlossen werden soll.

{`$is_logged_in`}
: ist auf `true` gesetzt, wenn der Benutzer eingeloggt ist.

{`$is_error`}
: enthält etwaige Fehlermeldungen beim Login.

Das Plugin erstellt eine Datenbanktabelle namens `serendipity_pending_authors`, in der die neu angemeldeten, aber noch nicht freigeschalteten Benutzer zwischengespeichert werden. Die Datenbanktabelle verfügt weitestgehend über dieselben Tabellenfelder wie die zentrale Tabelle `serendipity_authors`. Anstelle einer zentralen, primären ID bietet diese Tabelle jedoch das Datenbankfeld `hash`, mit dem der zukünftige Redakteur eindeutig identifiziert wird. Dieser Hash dient auch der Freischaltung via E-Mail.

7.2.4 Getaggte Artikel, Freie Artikel-Tags serendipity_plugin_freetag, serendipity_event_freetag

Das *Tagging* (*Tag*, englischer Begriff für *Markierung, Auszeichnung*) ist eine sehr verbreitete und komfortable Möglichkeit, Blog-Artikel zu verschlagworten.

Ein Blog besitzt in seinem Kerngedanken keine Struktur, wie es Content-Management-Systeme vorgeben. In einem Blog sind grundsätzlich erst einmal alle Artikel gleichrangig und werden alle auf einer einzigen Übersichtsseite angezeigt. Das macht es sehr einfach, bei einer sich regelmäßig ändernden Webseite (Tagebücher, News, neue Fachthemen) als Leser auf dem Laufenden zu bleiben. Man muss nur eine Seite (die Übersichtsseite) aufrufen und sieht chronologisch sortiert alle neuen Artikel. Da jeder Artikel eine Detailseite besitzt (der *Permalink*), kann man sich solche Artikel auch gut als Browser-Lesezeichen speichern.

Diese Struktur ist solange hervorragend geeignet, bis man als Leser einmal einen speziellen älteren Artikel sucht. Meist hat man nur eine vage Erinnerung, zu welchem Zeitpunkt der Artikel geschrieben wurde – die chronologische Suche über den Kalender bzw. die Archive fällt also aus. Auch eine Suche nach Stichworten kann kompliziert werden, wenn man sich nicht gerade ein besonderes Wort im Artikel gemerkt hat.

Hier kommen die hierarchischen Sortierungsmöglichkeiten von Serendipity ins Spiel: Mit etwas Glück hat der Redakteur seinen Artikel einer Kategorie zugeordnet, und der Leser kann seinen gewünschten Artikel über eine thematische Zuordnung wiederfinden.

Leider haben solche Hierarchien für Redakteure einen Nachteil: Sie sind starr und unflexibel. Man muss sich Gedanken darum machen, über welche Themen man schreiben möchte, und entsprechende Kategorien, möglicherweise mit Unterkategorien, anlegen. Nachdem man sich als Redakteur Nächte um die Ohren geschlagen hat, wie man seine Artikel kategorisiert, findet man nach einigen Monaten des Schreibens heraus, dass eine Kategorie mit einer Fülle an Artikeln überschwemmt wird und andere, liebevoll angelegte Kategorien nur mit einem oder gar keinem Artikel aufwarten können. Nun begint für den Redakteur also das Ausmisten: Kategorien vereinen oder aufteilen und danach alle Artikel neu kategorisieren.

Verständlicherweise kann dies Redakteure zum Verzweifeln bringen. Nicht jedes Blog ist nunmal klar zu klassifizieren.

Dieses Problems hat sich das Konzept des *Tagging* angenommen. Ein Artikel wird nicht starr kategorisiert, sondern ihm werden einfach beliebige Schlagworte zugeordnet. Wenn ein Redakteur also einen Artikel über die *Geschichte der Weblogs unter besonderer Berücksichtigung der Katzenbilder* geschrieben hat, muss er sich nicht lange den Kopf über eine sinnvolle Kategorisierung zerbrechen, sondern weist einfach die Schlagwörter *Geschichte, Weblogs, Katzenbilder* zu.

Diese Schlagwörter sieht der Besucher unterhalb des Artikels. Ein Klick auf ein Schlagwort zeigt ihm dann alle weiteren Artikel, die das Schlagwort ebenfalls enthalten. Nachträglich kann der Redakteur jederzeit ein neues Schlagwort zu bestehenden Artikeln hinzufügen oder bestehende Schlagwörter löschen. Zusätzlich ist natürlich auch eine beliebig feine Zuordnung in feste Kategorien möglich.

Zusammenfassend lässt sich sagen, dass *Tags* unheimlich komfortabel und selbsterklärend sind und eine leichte Verkettung (die Grundidee des Internets!) ermöglichen. Dennoch sollte man erwähnen, dass auch die sinnvolle Verwendung von Tags den Redakteuren obliegt. Da der Redakteur sich Stichwörter jedesmal neu ausdenken muss, kann es passieren, dass er für dieselbe Sache einmal *TV* und einmal *Fernsehen* benutzt. Gerade in Blogs mit mehreren

Redakteuren ist eine gemeinsame Harmonisierung der Schlüsselwörter daher unabdingbar.

Sogenannte *Tag-Wolken* (*Tag Clouds*) können zudem Ihre Blog-Artikel grafisch repräsentieren: In einer Auflistung sehen Sie alle Schlagwörter, die Sie benutzt haben. Diese sind abhängig von ihrer Einsatzhäufigkeit unterschiedlich stark hervorgehoben – das meistverwendete Tag ist groß geschrieben, selten verwendete Tags erscheinen klein am Rande.

Berühmt gemacht hat diese Tag-Wolke der Bilderdienst Flickr, der mit diesem Tagging-Konzept ein wahres Novum in die komplexe Bildersuche im Internet gebracht hat. Zu kaum einen Stichwort findet man heutzutage auf `http://flickr.com` nicht bereits mehrere Bilder, die zudem mit *verwandten Tags* kreuz-referenziert sind.

Das Ereignis-Plugin *Freie Artikel-Tags* kümmert sich um die Verwaltung und Eingabe von Tags. Es bindet eine Tagging-Maske in die Artikeloberfläche ein, und eine Verwaltungsoberfläche im Menüpunkt **Einträge** → **Tags verwalten**.

Bei der Erstellung eines Artikels sehen Sie im Bereich **Erweiterte Optionen** ein einfaches Eingabefeld namens **Freie Artikel-Tags**. In dieses Eingabefeld können Sie kommasepariert beliebige Tags eingeben. Oberhalb des Eingabefelds sehen Sie eine Liste von Tags, die bereits früher eingegeben wurden. Diese können Sie automatisch in das Eingabefeld übernehmen, indem Sie auf den jeweiligen Tag klicken. Die Tags, die Sie einem Artikel zugewiesen haben, zeigt das Plugin darauf im Frontend im Fußbereich des Artikels an. Durch einen Klick auf das entsprechende Tag sehen Sie eine Übersichtsseite mit weiteren Artikeln, denen das Tag zugeordnet wurde.

Die Verwaltungsoberfläche, die Sie über den Menüpunkt **Einträge** → **Tags verwalten** erreichen, bietet folgende Möglichkeiten:

Alle Tags verwalten

Ein Klick auf den Menüpunkt **Alle Tags verwalten** ruft eine Übersicht auf, in der Sie alle vorhandenen Tags einsehen können. Diese werden tabellarisch mit alphabetischer Sortierung dargestellt. Der Zahlenwert in der Spalte **Häufigkeit** gibt an, wie oft Sie das entsprechende Tag bereits zugeordnet haben.

Jedes Tag können Sie **Löschen**. Dies löscht nur die Zuordnung des Tags, ein Artikel wird also nicht gelöscht. Ein bestehendes Tag können Sie zudem **Umbenennen**. Dies benennt das Tag auch in den Artikeln um, denen Sie es bereits zugeordnet haben.

Wenn Sie ein pauschales Tag wie *Medien* vergeben haben, können Sie es über den Button **Aufteilen** auch in mehrere Tags zerkleinern. Dabei tragen Sie eine kommaseparierte Liste in das erscheinende Eingabefeld ein, beispielsweise *Kino,Fernsehen,Radio*. Nach der Aufteilung werden nun jedem Artikel, dem bisher das Tag *Medien* zugewiesen war, die neuen Tags *Kino, Fernsehen* und *Radio* zugeordnet. In der Zukunft können Sie dann die feineren Tags gezielt verteilen und bei bestehenden Einträgen auch ggf. einzelne der feineren Tags entfernen.

Verwaiste Tags verwalten, Einträge mit verwaisten Tags anzeigen

Verwaiste Tags sind Schlüsselwörter, die Sie bisher nur einmal einem Artikel zugewiesen haben. Solche verwaisten Tags sind meist weder für Besucher noch für Redakteure besonders hilfreich. Daher sollten Sie darüber nachdenken, solche Tags möglicherweise zu löschen oder umzubenennen.

Über den Menüpunkt **Einträge mit verwaisten Tags anzeigen** können Sie eine Liste von Blog-Artikeln einsehen, denen solche verwaisten Tags zugeordnet sind. Von dort aus können Sie komfortabel diese Einträge überarbeiten.

Einträge ohne Tags anzeigen

Analog zu Artikeln mit verwaisten Tags können Sie auch eine Liste einsehen, in der alle Artikel aufgeführt sind, die noch keine Tags erhalten haben.

Automatische Schlüsselwörter

Die Zuordnung von Schlüsselwörtern in Einträgen kann oft zu einer Fleißarbeit werden. Daher können Sie mit dem Plugin eine Wortliste einpflegen, die beim Auftreten von bestimmten Zeichenketten im Text Ihres Artikels automatisch ein zugeordnetes Tag zuweist.

Ein einfaches Beispiel kann dies am besten erläutern. Gegeben ist folgender Artikeltext:

```
Das beste Blog der Welt ist in einer neuen Version erschienen:
Serendipity 19.0 ist ab sofort verfügbar. Bugfixes sind bereits
seit Version 3.0 nicht mehr nötig, und da sämtliche Features schon
mit Version 4.0 das Ende der erdenkbaren Fahnenstange erreicht
haben, ist die einzige Neuerung diesmal die neue Versionsnummer.
An einigen Stellen wurden zudem Leerzeichen miteinander vertauscht,
um das Shakra der Bits und Bytes mal etwas in Schwingung zu
versetzen.
```

Folgende Tags ließen sich dem Artikel sinnvollerweise zuordnen: *Serendipity, Blog, Release, Spiritualität, Security*. Diese Zuordnung könnte man relativ einfach automatisieren: Die Tags *Serendipity* und *Blog* können immer dann eingebunden werden, wenn genau diese Wörter im Artikeltext vorkommen. Das Tag *Release* kann vergeben werden, wenn die Zeichenkette *neue Version* oder *neuen Version* vorkommt. Und das Tag *Spiritualität* könnte man aufgrund des Artikeltextes *Shakra* vergeben.

Genau diese Zuordnung können Sie über den Verwaltungspunkt **Einträge** → **Tags verwalten** → **Automatische Schlüsselwörter** festlegen. Dort werden alle vorhandenen Tags aufgelistet. Klicken Sie auf den zugehörigen Link **Bearbeiten**, so öffnet sich eine Eingabebox. Dort tragen Sie die Zeichenketten ein, bei deren Auftreten im Artikeltext das jeweilige Tag automatisch zugewiesen wird. Für das Tag **Release** müssten Sie also hier *neue Version,neuen Version* eintragen. Beachten Sie dabei, dass Sie automatische Schlüsselwörter nur für Tags eintragen können, die Sie bereits einmal verwendet haben. Eventuell müssen Sie also erst einen Eintrag mit dem gewünschten Tag erstellen, bevor Sie automatisch Schlüsselwörter dafür vergeben können. Durch die Trennung mehrerer möglicher Zeichenketten mit einem Komma können beliebig viele Zeichenketten eine Tag-Zuweisung automatisch auslösen – diese Zeichenketten können auch Leer- und Sonderzeichen enthalten, sie werden später unabhängig von Groß- und Kleinschreibung durchsucht.

Die Tags weist das Plugin immer beim Speichern eines Artikels automatisch zu. Wenn in Ihrem Blog-Artikel ein automatisches Schlüsselwort gefunden wurde, gibt das Plugin einen Hinweistext über die durchgeführte Zuordnung aus:

```
Schlüsselwort neue Version gefunden, Tag Release
automatisch zugewiesen.
```

Automatische Schlüsselwörter neu parsen

Nachdem Sie die Liste automatischer Schlüsselwörter eingepflegt haben, sollten alle bereits bestehenden Artikel daraufhin geprüft werden, so dass eventuell fehlende Tags hinzugefügt werden können. Dabei können keine doppelten Tag-Zuordnungen entstehen.

Der Vorgang ruft intern automatisch jeden einzelnen Artikel zur Bearbeitung auf, analysiert den Artikeltext und weist die neuen Tags anhand der Schlüsselwörter zu. Da hierbei die zentrale Serendipity-Funktion zum Speichern eines Artikels aufgerufen wird, kann dies bei einem größeren Artikelbestand zu Performanceproblemen führen. Daher werden jeweils nur 25 Artikel pro Aufruf analysiert und neu gespeichert, und danach ruft das Plugin die nächste Seite mit den nächsten 25 Artikeln auf.

Durch die Verwendung der zentralen Speicherungsfunktion werden zudem alle mittlerweile neu installierten Plugins neu angewendet. Je nachdem, wie alt Ihre Artikel sind und welche Plugins Sie seither installiert haben, könnte dies zu problematischen Änderungen Ihrer Artikel führen. Wenn Sie beispielsweise das Plugin zum Ersetzen von Zeichenketten erst später installiert haben, würden nun auch in älteren Artikeln Wortersetzungen durchgeführt, die Sie bisher nur bei neuen Artikeln erhielten.

Sicherer ist es daher, wenn Sie vor der Ausführung dieser Funktion ein Backup erstellen (siehe Seite 450).

Alle zugewiesenen Kategorien bestehender Artikel zu Tags konvertieren

Wenn Sie Kategorien und Tags parallel nutzen, kann es unter Umständen sinnvoll sein, sämtliche Kategorien (wie **TV**) auch automatisch als identisch benanntes Tag einem Artikel zuzuweisen. Diese Doppelung kann Besuchern später helfen, verwandte Themen zu finden.

Auch wenn Sie das Tagging-Plugin erst installiert haben, nachdem Sie bereits eine Reihe von Artikeln in Kategorien zugeordnet haben, kann es Sinn machen, sämtliche bestehenden Kategorien als Tags zuzuordnen. Um diesen Vorgang einmalig auszuführen, können Sie in der Tag-Verwaltung den Menüpunkt **Alle zugewiesenen Kategorien bestehender Artikel zu Tags konvertieren** benutzen. Daraufhin holt das Plugin eine Liste aller vorhandenen Artikel, geht deren Zuordnung zu Kategorien durch und weist diese Kategorien jeweils den neu erstellten, gleichnamigen Tags zu.

Bei einem großen Artikelbestand kann dieser Vorgang eine Weile dauern. Während der Bearbeitung gibt Ihnen das Plugin eine Meldung aus, welche Kategorien es umgewandelt hat.

Nach dem Vorgang können Sie entweder die bestehenden Kategorien löschen (dabei werden die bestehenden Artikel beibehalten) oder Sie auch nach wie vor behalten, um Ihren Besuchern beide Darstellungsweisen zu ermöglichen.

Die von Ihnen zugewiesenen Schlüsselwörter eines Artikels bindet das Plugin in der Übersichtsseite ein. Dort kann der Besucher auf ein Schlüsselwort klicken, um weitere Einträge zu sehen, die diesem Tag entsprechen. Die URL für derartige Ansichten entspricht dem Format `http://www.example.com/serendipity/plugin/tag/`*`Schluesselwort`*. Alternativ kann auch jede Serendipity-URL mit einem Parameter wie `http://www.example.com/serendipity/rss.php?serendipity[tag]=`*`Schluesselwort`* aufgerufen werden und so beispielsweise einen RSS-Feed ausliefern, der nur Artikel mit dem gewünschten Tag anzeigt.

Wenn Sie eine solche Tag-Übersichtsseite aufrufen, stellt das Plugin oberhalb der Einträge die erwähnte Tag-Wolke dar, anhand der ein Besucher weitere Artikel aufrufen kann, die mit ähnlichen Tags versehen wurden.

Die Darstellung der Tag-Listen und Tag-Wolken können Sie über die CSS-Klassen `.serendipity_freeTag` (unterhalb des Artikels) und `.serendipity_freetag_taglist` (für die Tag-Wolke) anpassen.

In den Smarty-Templates (`entries.tpl`) bindet das Plugin folgende Variablen ein:

{`$entry.properties.freetag_tagList`}
: enthält eine kommaseparierte Liste aller Tags eines Artikels.

{`$entry.properties.freetag_tags`}
: enthält ein nummeriertes Array mit den Tags eines Artikels.

{`$CONST.PLUGIN_VARS_TAG`}
: enthalten den Namen eines Tags, wenn die Eintragsübersicht durch den Besucher auf alle Artikel mit einem speziellen Tag eingeschränkt wurde. Da dieser Wert als Konstante vom Plugin gesetzt wird (wie auch das Array `$serendipity['plugin_vars']['tag']`), können Sie auch innerhalb anderer Plugins (mit PHP-Code) auf die Ansicht des Tagging-Plugins zugreifen.

Das Tagging-Plugin bindet die zugewiesenen Schlüsselwörter selbständig in Ihre RSS-Feeds ein. Dort werden sie über die Elemente `<category>` (RSS 2.0, 0.91) bzw. `<dc:subject>` (RSS 1.0, Atom) integriert, die übliche Feedreader problemlos auslesen können. Wenn Sie diese Tags im Feed gerne speziell (ähnlich wie der FeedBurner-Service, siehe Seite 207) formatieren möchten, können Sie dazu die `feed_*.tpl`-Template-Dateien bearbeiten und die oben aufgeführten Variablen auslesen (siehe Seite 490).

Wenn Sie das XML-RPC Plugin nutzen (siehe Seite 351), kann das Tagging-Plugin etwaige Tags aus Ihrem XML-RPC Client interpretieren, wenn diese als *mt_keywords* gesendet werden.

Das Ereignis-Plugin bietet Ihnen folgende Konfigurationsoptionen:

Erstelle Tags für zugewiesene Kategorien
: Wenn Sie diese Option aktivieren, kann das Plugin beim Speichern eines Artikels, der

einer Kategorie zugeordnet wird, den Namen dieser Kategorie zusätzlich als Tag zuweisen. Dies kann sehr nützlich sein, wenn Sie sich Tipparbeit beim Erstellen von Tags sparen und eine Kategorisierung von Artikeln abseits der Schlüsselwörter weiterhin nutzen wollen. Weitere Erklärungen hierzu finden Sie auf Seite 404.

Taglink

Wenn Ihre Besucher auf den Link eines Tags klicken, führt dieser standardmäßig zu einer URL wie `http://www.example.com/serendipity/plugin/tag/`*`tagname`*. Dieser virtuelle Verzeichnisname wird vom Tagging-Plugin ausgewertet. Dank der globalen URL-Umformung können alle Pfadverweise zu `/plugin/` zu einem beliebigen Plugin führen.

Wenn Sie gerne stattdessen eine eigene URL verwenden wollen, müssen Sie eigenständig für eine korrekte `mod_rewrite`-Regel sorgen, die Zugriffe beispielsweise von `http://www.example.com/serendipity/schluesselwoerter/`*`tagname`*`/` auf `http://www.example.com/serendipity/index` umleitet. Dies könnte beispielsweise über folgenden `.htaccess`-Eintrag erreicht werden:

```
RewriteRule ^schluesselwoerter/(.*)/$ index.php?serendipity[tag]=$1
[L,QSA]
```

Damit das Tagging-Plugin danach Ihre konfigurierte URL verwenden kann, müssen Sie den Pfad im Feld **Taglink** einfügen.

Bitte ändern Sie den Wert dieses Feldes nur dann, wenn Sie manuelle Vorkehrungen für die korrekte Umleitung getroffen haben! Wenn Sie hier einfach einen Fantasiewert eintragen, kann das Plugin die Tag-Übersichtsseiten nicht mehr übernehmen.

Zeige die Tags in der Fußzeile an

Das Plugin zeigt die einem Artikel zugewiesenen Schlüsselwörter in der Fußzeile jedes Eintrages an. Dies hat den Vorteil, dass es losgelöst vom Artikeltext formatiert werden kann und unabhängig vom Artikeltext ist.

Wenn Sie die Option **Zeige die Tags in der Fußzeile an** deaktivieren, kann das Plugin die Tag-Listen direkt in den Artikeltext einfügen. Dann werden Tag-Zuweisungen auch direkt im Artikeltext sichtbar, wenn dieser beispielsweise in einem RSS-Reader betrachtet wird. Sie verlieren dadurch jedoch Flexibilität in der Positionierung der Tag-Liste.

Zeige Wolke mit verwandten Tags an

Diese standardmäßig aktivierte Option sorgt dafür, dass Ihre Besucher beim Aufrufen einer Tag-Übersichtsseite eine Information darüber erhalten, welche weiteren Tags den Artikeln zugewiesen wurden, die dem aktuell gewählten Tag entsprechen.

Minimale/Maximale Schriftgröße eines Tags in der Wolke

Das Plugin kann die Größe der Tags innerhalb der Tag-Wolke abhängig von deren Verwendungshäufigkeit skalieren. Oft verwendete Tags erscheinen größer, selten benutzte

Tags etwas kleiner. Das Plugin kann dies mittels relativer Schriftgrößenänderungen umsetzen. Sie können über entsprechende Prozentwerte kontrollieren, wie stark die Varianz der Schriftgröße sein darf.

Anzahl der Stichwörter, die in die Meta-Angaben des HTML-Codes

eingesetzt werden sollen Das Tagging-Plugin kann beim Aufrufen einer Artikeldetailseite die zugewiesenen Tags innerhalb eines HTML `<meta>`-Tags einbinden. Solche `<meta>`-Tags können von Suchmaschinen ausgewertet werden, um die Schlüsselwörter automatisiert auszuwerten.

Aufgrund des hohen Missbrauchs durch Spammer werden diese Keywords heutzutage von Suchmaschinen jedoch sehr, sehr gering gewichtet. Dennoch macht es aus semantischer Sicht durchaus Sinn, dass Sie auf Ihren Artikel-Einzelseiten die von Ihnen eingetragenen Schlüsselwörter in dieser Meta-Angabe einbinden.

In dem Optionsfeld können Sie eine beliebige Zahl eintragen, mit der Sie bestimmen, wie viele Ihrer eingegebenen Schlüsselwörter als Meta-Tag eingebunden werden sollen. Suchmaschinen können meist nur zwischen 10 und 15 Meta-Keywords auslesen. Wenn Sie Ihren Artikeln also sehr viele Tags zuweisen, ist es sinnvoll, die Liste der maschinenlesbaren Keywords an dieser Stelle zu begrenzen. Im täglichen Einsatz dürfte es jedoch recht selten sein, dass Sie einmal zu viele Tags benutzen – stellen Sie diese Option daher ruhig auf einen Wert wie 15 oder 20.

Die Zahl `0` deaktiviert die Verwendung von Meta-Tags. Beachten Sie, dass das automatische Ausfüllen von Meta-Keyword-Tags des Plugins mit dem speziellen Plugin namens *Meta-Keywords* kollidieren kann. Mithilfe dieses Plugins können Sie die Meta-Felder der Detailseiten Ihrer Artikel gezielt befüllen und auch Kurzbeschreibungen eintragen.

Zeige Artikel mit ähnlichen Themen an, Wieviele Artikel mit ähnlichen

Themen sollen angezeigt werden Unterhalb der Liste der zugewiesenen Tags eines Artikels kann das Plugin eine weitere Artikelliste einblenden. In dieser werden Links zu den aktuellsten Artikeln (konfigurierbare Anzahl) ausgegeben, die ähnliche Tags verwenden wie der aktuelle Artikel.

Dies kann für Besucher besonders hilfreich sein, wenn sie Ihren Artikel über eine Suchmaschine gefunden haben. Mit weiterführenden Artikeln in Ihrem Blog können Sie so den Besucher an Ihre Seite binden und ihn dort mit weiteren Informationen oder Texten dazu verlocken, sich noch etwas umzusehen.

Tags in Kleinbuchstaben umwandeln

Üblicherweise werden Tags allesamt klein geschrieben, um bei mehreren Redakteuren zu vermeiden, dass dieselben Tags unterschiedlich geschrieben werden.

Wenn Sie jedoch redaktionell sicherstellen, dass identische Tags nicht in unterschiedlicher Klein- oder Großschreibung auftreten, dann können Sie das Plugin anweisen, keine automatische Umwandlung vorzunehmen.

Sende X-FreeTag-HTTP-Header
Wenn das Plugin eine Übersichtsseite von Artikeln zu einem gewissen Tag ausliefert, kann das verwendete Tag als HTTP-Kopfzeile (*Header*) ausgegeben werden. Diese Information wird im Hintergrund an den Browser übermittelt und dient nur dem Zweck, dass Entwickler leichter feststellen können, welches Tag vom Plugin verwendet wird. Da das Plugin mehrere technische Möglichkeiten zur Bestimmung des Tags bietet, kann es hier theoretisch zu nicht nachvollziehbaren Problemen kommen, die sich mithilfe der HTTP-Kopfzeile besser analysieren lassen.

Diese Option kann gefahrlos deaktiviert werden, jedoch hat deren Aktivierung auch keine spürbaren Auswirkungen. Paranoiker werden diese Option ggf. deaktivieren wollen, da man anhand dieser Ausgabe feststellen kann, ob ein Blog mit Serendipity betrieben wird.

Klickbare Liste aller schon vorhandenen Tags beim Schreiben eines Eintrags

anzeigen, Zeige Tag-Vorschläge bei der Eingabe Beim Verfassen eines Artikels kann das Plugin alle bereits vergebenen Tags in einer Liste anzeigen. Dort können Sie jedes vorhandene Tag anklicken, um es komfortabel dem Eintrag zuzuweisen.

Bei sehr vielen vorhandenen Tags kann diese Liste sehr umfangreich werden. Daher können Sie diese klickbare Liste gezielt deaktivieren, um den Bildschirm übersichtlicher zu machen.

Weiterhin kann das Plugin beim Aktivieren der Option **Zeige Tag-Vorschläge bei der Eingabe** mit einem dynamischen JavaScript dafür sorgen, dass, sobald Sie einen Buchstaben im Eingabefeld der Tags eintippen, eine Liste der mit diesem Buchstaben beginnenden vorhandenen Tags eingeblendet wird. So könnten Sie also bereits bei Eingabe von `Se` die Auswahl `Serendipity, Sepsis, Sekundärnavigation` sehen und aus dieser Liste das gewünschte Tag auswählen. Dies spart sowohl Tipparbeit als auch die umfangreiche Liste aller vorhandenen Tags, benötigt jedoch einen JavaScript-fähigen Browser.

Technorati Tag Link, Technorati Tag image
Falls das Plugin Links zum Technorati-Webserver einbinden soll, können Sie dies über die Option **Technorati Tag Link** aktivieren. Standardmäßig wird ein Bild von den Technorati-Servern als Symbol dargestellt, Sie können jedoch auch eine eigene URL für die Grafik eintragen.

Weiterhin steht ein gekoppeltes Seitenleisten-Plugin zur Verfügung. Darüber können Sie eine

Übersicht aller verwendeten Tags in der Seitenleiste[10] einbinden. Die Konfigurationsoptionen dieses Seitenleisten-Plugins sind:

XML-Icons anzeigen, XML-Icon URL
: Zu jedem Tag kann in der Seitenleiste ein Link zu dem jeweiligen RSS-Feed eingebunden werden.

Zeilenumbruch nach jedem Tag
: Standarmäßig stellt das Plugin alle Tags hintereinander dar, so dass die volle Breite der Seitenleiste für die Tag-Auflistung benutzt werden kann. Bei unterschiedlichen Schriftgrößen kann dies je nach Menge Ihrer Tags möglicherweise zu unübersichtlich aussehen, daher können Sie die Option **Zeilenumbruch nach jedem Tag** aktivieren, um pro Zeile nur ein einzelnes Tag auszugeben.

Taglink
: Analog zu der gleichnamigen Konfigurationsoption des Ereignis-Plugins können Sie hier festlegen, zu welcher URL der Klick auf ein Schlüsselwort führt (siehe Seite 406).

Schriftgröße des Font-Tags je nach Popularität vergrößern
: Das Seitenleisten-Plugin kann die Schriftgröße der Tags je nach Einsatzhäufigkeit verändern. Häufig verwendete Tags erscheinen groß, selten eingesetzte kleiner. Dies erleichtert es dem Besucher, einen thematischen Überblick zu Ihrem Blog zu erhalten.

Minimale/Maximale Schriftgröße eines Tags
: Legt fest, wie klein oder groß ein Tag in der Darstellung (in absoluten Prozentwerten) sein darf. Das Plugin skaliert die Größen daraufhin automatisch in dem festgelegten Von/Bis-Bereich.

Wieviele Tags sollen angezeigt werden
: Mit dieser Option legen Sie fest, auf wie viele Schlüsselwörter die Ausgabeliste begrenzt werden soll.

Wie oft muss ein Tag vorkommen, damit er angezeigt wird
: Wenn Sie sehr viele Tags in Ihrem Blog einsetzen, macht es wenig Sinn, jedes einzelne in der Seitenleiste aufzuführen. Daher können Sie diese Ausgabe auf populäre Tags beschränken und mithilfe der Option **Wie oft muss ein Tag vorkommen...** eine Mindestanzahl festlegen.

Sortierung
: Hier können Sie festlegen, ob die Tags in der Seitenleiste alphabetisch nach Name oder nach Einsatzhäufigkeit sortiert werden sollen.

  Wenn Sie die Anzahl der Tags einschränken, bestimmt die Sortierungsreihenfolge auch, welche Tags am Ende der Liste entfernt werden. Wenn Sie die Liste alphabetisch sortieren und auf 20 Tags beschränken, aber Ihre populärsten Tags mit dem Buchstaben

[10] Die Ausgabe eines Seitenleisten-Plugins können Sie auch an jeder anderen beliebigen Stelle in einem Template einbinden, schlagen Sie dazu in Kapitel 9.6.3 auf Seite 564 nach.

Z anfangen, könnte es sein, dass diese Schlüsselwörter nicht ausgegeben werden, da bereits 20 Tags dargestellt wurden, die in der alphabetischen Sortierung weiter vorn liegen.

Datenbanktabelle

Die Datenbanktabelle `serendipity_event_entrytags` enthält jeweils ein zugeordnetes Tag (`tag`) zu einer Artikel-ID (`entryid`).

Die freien Schlüsselwörter, die für ein Tag zugeordnet werden können, werden in der Tabelle `serendipity_event_tagkeywords` gespeichert. Dort wird ebenfalls jedem Tag (`tag`) eine Liste an kommaseparierten Schlüsselwörtern (`keywords`) zugeordnet.

In älteren Versionen des Plugins wurden Tags in der Tabelle `serendipity_entryproperties` gespeichert. Dies hat sich performancetechnisch nicht günstig ausgewirkt, und die Tags liegen nun in einer eigenen Tabelle.

7.2.5 Liste der statischen Seiten, Statische Seiten serendipity_plugin_staticpage, serendipity_event_staticpage

Das Plugin **Statische Seiten** ist das wohl wichtigste Plugin, wenn Sie Serendipity als *Content-Management-System* (CMS) einsetzen wollen.

Die Grenzen zwischen einem Blog und einem CMS sind relativ fließend. Letztlich ist ein Blog nur eine Unterform eines CMS, denn ein CMS dient (per definitionem) lediglich der Erfassung und Darstellung redaktioneller Inhalte.

Die allgemeine Auffassung zum Unterschied zwischen Blog und CMS betrifft die spezielle chronologische Aufteilung von Blog-Artikeln. Ein Blog stellt üblicherweise nur zeitlich sortierte Archive dar und arbeitet immer mit chronologisch sortierten Artikelübersichten. Ein CMS hingegen kann Inhalte unterschiedlich gewichten, in Hierarchien einordnen und Artikel meist separiert darstellen.

Glücklicherweise kann man solche isolierten Ansichten mithilfe des Serendipity-Plugins *Statische Seiten* recht gut lösen. Sie können eigenständige *Seiten* erstellen, die sich parallel (und unabhängig) von Blog-Artikeln einbinden lassen.

Klassischer Anwendungsfall einer statischen Seite ist das (in Deutschland obligatorische) Impressum, eine *Über mich*-Seite oder auch beliebige andere Informationsseiten, die Sie nicht mit dem chronologischen Fluss der Blog-Artikel vermischen möchten.

Jede statische Seite lässt sich über eine eigene URL aufrufen, den *Permalink*. Dieser kann beliebig formatiert werden und muss nicht dem üblichen Blog-Artikel-Link wie `http://www.example.com/serendpity/archives/1-Mein-Impressum.htm` entsprechen, sondern könnte auch `http://www.example.com/serendipity/seiten/Impressum.htm`

lauten.

Ein Vorteil einer statischen Seite ist, dass Sie dort beliebigen HTML-Code einbinden können und dabei die Seitenleisten-Plugins und das zentrale Layout Ihres Blogs beibehalten. So müssen Sie also Zusatzseiten nicht kompliziert über FTP-Uploads realisieren, sondern können diese menü- und datenbankgestützt einpflegen.

Die statischen Seiten werden stets aus der Datenbank ausgelesen und dargestellt, das Plugin erstellt *keine* physikalischen Dateien auf dem Server. Sollte bei Ihrem Blog Performance eine große Rolle spielen, kann es daher möglicherweise sinnvoller sein, zusätzliche Seiten manuell über HTML-Dateien anzulegen, anstatt dieses Plugin zu benutzen. In Zukunft könnte das Plugin jedoch möglicherweise erweitert werden, um statische Seiten vollständig *statisch* auf dem Server abzuspeichern.

Statische Seiten können auch über individuelle Templates eingepflegt werden, so dass Sie sämtliche Möglichkeiten der Smarty-Programmierung für individuelle Gestaltung anwenden können.

Ein gekoppeltes Seitenleisten-Plugin kümmert sich um die Darstellung der vorhandenen statischen Seiten über ein Menü in der Seitenleiste des Frontends und ist auch in der Lage, die von Ihnen angelegten Hierarchien darzustellen.

Nachdem Sie das Plugin *Statische Seiten* installiert haben, können Sie Inhalte über den Menüpunkt **Einträge** → **Statische Seiten** einpflegen. Diese Oberfläche des Plugins stellt sozusagen einen separaten CMS-Bereich dar. Dort können Sie neue Seiten anlegen und bearbeiten, die Reihenfolge von Seiten verändern sowie Seitentypen (Vorlagen) verwalten.

Seiten bearbeiten

Nachdem Sie diese Oberfläche aufrufen, sehen Sie ein Ausklappfeld, in dem alle vorhandenen statischen Seiten aufgeführt werden. Um eine neue Seite anzulegen, wählen Sie in diesem Ausklappfeld den Punkt **Neuer Eintrag** und klicken auf *Los!*.

Bestehende Seiten können Sie in dem Ausklappfeld auswählen und über einen Klick auf *Los!* bearbeiten oder über einen Klick auf **Löschen** aus der Datenbank (unwiderruflich) entfernen.

In der Liste der statischen Seiten sieht der jeweilige Redakteur nur die Seiten, zu denen er als Eigentümer eingetragen ist. Weiterhin sieht auch jeder Redakteur alle Seiten von Benutzern, die einen niedrigeren Benutzerrang haben. Administratoren sehen daher alle Seiten, Chefredakteure sehen Seiten von normalen Redakteuren. Eine weitergehende Kontrolle der Zugriffsrechte bietet das Plugin in seiner jetzigen Version noch nicht, für die Zukunft denkt der fleißige Entwickler des Plugins (Falk Döring) jedoch daran, eine Rechteverwaltung mit individuellen Zugriffsrechten für Benutzergruppen (anstelle von Benutzerrängen) einzubinden. Wenn bestimmte Benutzergruppen keinen Zugriff auf statische Seiten haben sollen, so können Sie das Plugin global für diese Gruppe(n) deaktivieren (siehe Seite 191).

Die Maske zum Erstellen oder Bearbeiten eines Artikels ist identisch und wird beim Bearbeiten lediglich mit den vorhandenen Daten vorausgefüllt. Lassen Sie sich von der Fülle der

Eingabeoptionen der Maske nicht abschrecken! Alle Felder werden in diesem Kapitel näher besprochen, und Sie werden feststellen, dass Sie meist nur einen kleinen Teil dieser Felder ausfüllen müssen.

Kopfzeile
: Die *Kopfzeile* bestimmt bei einer statischen Seite die Überschrift. Tragen Sie also hier eine eindeutige Bezeichnung der statischen Seite an, wie die Besucher sie später sehen sollen. Sie können hier Leerzeichen und Sonderzeichen eintragen, achten Sie jedoch darauf, dass zu lange Kopfzeilen möglicherweise das Layout sprengen könnten.

Permalink
: Der Permalink gibt an, unter welcher URL die statische Seite aufgerufen werden kann (weitere Details siehe Seite 238). Ein Permalink muss für jede statische Seite eindeutig sein.

URL-Titel der Seite
: Der URL-Titel einer statischen Seite ist sehr wichtig für interne Vorgänge. Er legt eine Alternativ-Syntax fest, die eine statische Seite zusätzlich zum Permalink eindeutig beschreibt. Weitere Informationen zu dieser Option finden Sie auf Seite 238.

Artikeltyp/Seitentyp
: Die Formatierung einer statischen Seite geschieht anhand einer Vorlage, die über *Seitentypen* verwaltet wird. Seitentypen können Sie selbständig über die Oberfläche **Seitentypen bearbeiten** einpflegen (siehe Seite 416).

  Jeder statischen Seite muss ein Seitentyp zugeordnet werden. An einigen Stellen des Plugins wird der Begriff *Artikeltyp* verwendet, dieser ist jedoch gleichbedeutend mit *Seitentyp*.

Artikelstatus
: Statische Seiten können ähnlich wie Blog-Artikel entweder veröffentlicht oder noch als Entwurf gespeichert werden. Nur veröffentlichte Artikel sind für Besucher im Frontend sichtbar.

Sprache
: Wenn Sie Ihr Blog mithilfe des Plugins *Multilinguale Einträge* mit mehrsprachigen Inhalten führen, können Sie auch die statischen Seiten jeweils in unterschiedlichen Sprachen anlegen.

  Multilinguale Blog-Artikel fügen die in andere Sprachen übersetzten Texte demselben Datenbank-Eintrag hinzu, statische Seiten jedoch müssen pro Übersetzung eine eigenständige Seite erhalten. Einer statischen Seite kann daher immer nur genau eine Sprache zugeordnet werden.

  Sobald Sie einer statischen Seite eine Sprache zuordnen, kann der Besucher einer Webseite diese nur aufrufen, wenn er das Blog mit der übereinstimmenden Spracheinstellung besucht. Jemand, der also das Blog in englischer Sprache aufruft, kann eine deutsche statische Seite nicht lesen.

Inhalt

Der Inhalt ist glücklicherweise ein selbsterklärender Eingabebereich: Hier tragen Sie den HTML-Text ein, der später auf der statischen Seite erscheinen soll.

Wenn Sie den WYSIWYG-Editor (siehe Seite 104) aktiviert haben, können Sie an dieser Stelle dessen Fähigkeiten einsetzen. Sämtliche Buttons oberhalb dieses Eingabefelds entsprechen denen bei der Erstellung eines Blog-Artikels.

Textformatierung(en) durchführen

Wenn Sie die Option aktivieren, wird der Inhalt einer statischen Seite an alle installierten Textformatierungs-Plugins weitergereicht. Diese Plugins können dann die eingestellten Formatierungen (beispielsweise Wandlung von Text-Smileys zu Grafiken oder Umwandlung von BBCode in HTML) durchführen.

Wenn Sie eigenen HTML-Code in den **Inhalt** einer statischen Seite eingefügt haben, könnten Textformatierungs-Plugins Ihren Code möglicherweise ungewollt umformatieren, daher können Sie diese Funktion gezielt deaktivieren.

Weitere Hinweise zu Textformatierungs-Plugins finden Sie auf Seite 237.

Als Artikel formatieren

Standardmäßig wird eine statische Seite in das Layout Ihres Blogs eingefügt. Die Stellen, an denen Serendipity üblicherweise Datumsformatierungen und Überschriften ausgibt, werden dann mit den Daten der statischen Seite gefüllt.

Wenn Sie jedoch eigenen HTML-Code in den **Inhalt** der statischen Seite eingefügt haben, kann eine solche Formatierung für Sie zu eng gefasst sein. Daher können Sie die Option **Als Artikel formatieren** deaktivieren, damit Ihre statische Seite sozusagen in das rohe Layout der Seite eingefügt wird.

Weitere Details dieser Option finden Sie auf Seite 239.

Seitentitel für „Als Artikel formatieren"-Ansicht

Wenn Sie die Option **Als Artikel formatieren** gewählt haben, muss eine Seitenüberschrift im Inhaltsbereich ausgefüllt werden, in der Serendipity üblicherweise das Datum eines Blog-Artikels darstellt. Da eine statische Seite jedoch keine chronologische Relevanz hat, können Sie hier beispielsweise die Überschrift der statischen Seite einbetten.

Name des Autors

Der Name des Autors der statischen Seite legt zugleich fest, wer der Eigentümer dieser Seite ist. Abhängig davon können später nur der Eigentümer und alle höherrangigen Redakteure diese statische Seite bearbeiten. Zudem wird der Eigentümer einer statischen Seite standardmäßig im Layout der Seite angezeigt.

Elternseite

Statische Seiten können in beliebige Hierarchien eingeordnet werden. Die Hierarchien richten sich dabei nach bereits angelegten statischen Seiten.

Standardmäßig wird jede statische Seite als **Ist Elternseite** eingetragen, also auf der ersten Ebene der Seiten. Über das Ausklappfeld **Elternseite** können Sie aber auch jede bereits angelegte statische Seite auswählen, um festzulegen, dass die aktuelle statische Seite dieser Seite untergeordnet ist.

Die Hierarchie der statischen Seiten kann später von Besuchern über das Seitenleisten-Plugin wie auch über die Quernavigation der statischen Seiten eingesehen werden.

Zugeordnete Kategorie

Grundsätzlich besteht zwischen statischen Seiten und Blog-Kategorien keine Zuordnung. Das Serendipity-System kann daher statische Seiten nicht einfach innerhalb einer Kategorie-Übersicht ausgeben.

Jedoch bietet das Plugin eine Möglichkeit, um einer statischen Seite eine Kategorie zuzuweisen. Ruft der Besucher später eine so zugewiesene statische Seite auf, kann er direkt zu der hinterlegten Kategorie wechseln. Auch ist es möglich, innerhalb einer solchen statischen Seite die Einträge der hinterlegten Kategorie anzuzeigen. Umgekehrt besteht die Möglichkeit, dass der Besucher bei der Ansicht einer Kategorie-Übersichtsseite auf die zugeordnete statische Seite springt.

Dies funktioniert nur, wenn Sie den **Artikeltyp** der jeweiligen statischen Seite auf den Eintrag **Staticpage with related category** gesetzt haben. Nur diese Seitentypen besitzen ein Template, bei dem die Kategoriezuordnung berücksichtigt wird, alle anderen Seitentypen ignorieren dies. Sie können auch eigene Seitentypen anlegen, die die Kategoriezuordnung individuell regeln, dies ist auf Seite 427 näher beschrieben.

Sie können nur eine statische Seite pro Kategorie zuordnen.

Kinderseiten anzeigen

Wenn Sie eine statische Seite anlegen oder bearbeiten, der andere statische Seiten über das Ausklappfeld **Elternseite** zugeordnet sind, können Sie bei der Ansicht der Elternseite eine Navigation einbinden, die sämtliche zugeordneten Kinderseiten darstellt.

Stellen Sie sich vor, Sie haben eine statische Seite namens *Meine Familie* angelegt. Bei dieser Seite stellen Sie die **Elternseite** auf **Ist Elternseite** und **Kinderseiten anzeigen** auf den Wert **Ja**. Zusätzlich legen Sie drei weitere statische Seiten namens *Wolfram*, *Sabine* und *Horst* an. Bei diesen drei Seiten legen Sie als **Elternseite** jeweils die statische Seite **Meine Familie** fest. Wenn Sie nun die statische Seite *Meine Familie* ansehen, werden Sie Navigationsmöglichkeiten zu den Unterseiten *Wolfram, Horst* und *Sabine* sehen.

Einleitung

Die **Einleitung** gilt für statische Elternseiten. Beim obigen Beispiel der Familienseite könnten Sie in der statischen Seite *Meine Familie* einen beschreibenden Text einfügen.

Dieser Text wird daraufhin in einer Übersichtsseite vor der Auflistung der vorhandenen Kinderseiten angezeigt, was es Ihnen erspart, gleichlautende Texte in alle Unterseiten einzubinden.

Standardmäßig wird die Einleitung oberhalb des **Inhalts** einer statischen Seite eingebunden. Dies können Sie jedoch über die Smarty-Templates auch beliebig verändern.

Passwort
Statische Seiten können mit einem Passwortschutz versehen werden. Ein Besucher kann den Inhalt einer statischen Seite dann erst einsehen, wenn er in einem Formular vorher das korrekte Passwort hinterlegt hat.

Diese Seite als Startseite definieren
Eine statische Seite kann die Blog-Startseite (die Artikelübersicht der letzten Einträge aller Kategorien) ersetzen. So können Sie dem Besucher auf der ersten Seite des Blogs einen Hinweistext oder Disclaimer darstellen. Die ursprüngliche Startseite des Blogs können Sie danach via `http://www.example.com/serendipity/index.php?/frontpage` aufrufen.

Damit diese Funktion korrekt arbeiten kann, müssen Sie sicherstellen, dass Sie (pro Sprache) nur eine einzige statische Startseite angelegt haben. Zudem können andere Plugins, die eigene Inhalte darstellen, mit der statischen Startseite kollidieren. Dazu zählt beispielsweise das Kontaktformular oder das Gästebuch. Damit eine statische Startseite korrekt angezeigt werden kann, muss sie in der Reihenfolge der Ereignis-Plugins (siehe Seite 145) *vor* den kollidierenden Ereignis-Plugins erscheinen. Generell ist es empfehlenswert, das Plugin *Statische Seiten* als eines der ersten Ereignis-Plugins einzusortieren.

Navigation anzeigen
Die **Navigation** einer statischen Seite dient dazu, von einer zur nächsten statischen Seite (oder Unterseite) zu wechseln. In manchen statischen Seiten ist so eine Navigation nicht erwünscht, daher können Sie diese pro statischer Seite gezielt deaktivieren.

In der Navigation der Seitenleiste einbinden
Mittels des gekoppelten Seitenleisten-Plugins *Liste der statischen Seiten* können Sie in der Seitenleiste eine Liste aller verfügbaren statischen Seiten einbinden. Dort erscheinen jedoch nur diejenigen statischen Seiten, bei denen Sie die Option **In der Navigation der Seitenleiste einbinden** aktiviert haben.

Versteckte oder unwichtige statische Seiten können Sie aus dieser Liste daher durch Deaktivieren der Option ausschließen.

Seitenreihenfolge

Die Seitenreihenfolge der statischen Seiten können Sie über den Menüpunkt **Einträge** → **Statische Seiten** → **Seitenreihenfolge** verwalten. Diese Reihenfolge ist für die Erstellung der Navigation und die Ausgabe des Seitenleisten-Plugins von Interesse.

In der Oberfläche sehen Sie alle statischen Seiten untereinander und hierarchisch eingerückt aufgelistet. Über einen Klick auf den Pfeil nach oben sortieren Sie eine Seite weiter oben ein, der Pfeil nach unten schiebt eine Seite einen Schritt zurück.

Sie können die Reihenfolge nur innerhalb der jeweiligen Hierarchieebene verändern, ein Verschieben eines Untermenüpunkts in eine andere Hauptebene ist nicht möglich. Dies können Sie erreichen, indem Sie die jeweilige zu verschiebende statische Seite bearbeiten und eine andere **Elternseite** auswählen.

Seitentypen bearbeiten

Seitentypen (oder auch *Artikeltypen*) entsprechen einer Vorlage für eine Gruppe von statischen Seiten.

Standardmäßig gibt es zwei Arten von statischen Seiten: eine Übersichtsseite (**Overview**) und eine Artikelseite (**Article**).

Übersichtsseiten können gut als Elternseiten einer statischen Seite eingesetzt werden, da hier gezielt auf die Navigation der Unterseiten eingegangen wird. Der Seitentyp **Article** hingegen konzentriert sich auf die Darstellung von Einzelseiten.

Jedem Seitentyp können Sie eine eigene Smarty-Template-Datei zuweisen, über die Sie die Formatierung gezielt beeinflussen können. Beispielsweise könnten Sie einen Seitentyp erstellen, der den Inhalt grafisch wie einen Zeitungsartikel formatiert, ein anderer Seitentyp könnte Inhalte wie auf einer Postkarte darstellen.

Einen Sonderfall stellt der Seitentyp **Staticpage with related category** dar. In dieser Vorlage bindet das Plugin Inhalte einer zugeordneten Kategorie ein, um eine Verbindung von Blog-Artikeln und statischen Seiten zu ermöglichen.

Die einzelnen Seitentypen können Sie über ein Ausklappfeld bearbeiten oder erstellen, analog zu dem Vorgang bei statischen Seiten. Ein Seitentyp hat lediglich folgende Eingabefelder:

Beschreibung
: Die Beschreibung eines Seitentyps gibt an, wie dieser in dem Ausklappfeld **Artikeltyp** einer statischen Seite eingebunden wird. Sie können ihre Artikelvorlagen so individuell benennen, Sonder- und Leerzeichen sind erlaubt.

Templatename
: Der **Templatename** gibt den Dateinamen des Smarty-Templates an, das für den jeweiligen Seitentypen gültig sein soll.

  Das Plugin wird mit drei Standard-Templates ausgeliefert: `plugin_staticpage.tpl` (**Article**), `plugin_staticpage_aboutpage.tpl` (**Overview**), `plugin_staticpage_related_category.tpl` (**Staticpage with related category**).

  Sie können die Dateien entweder im Plugin-Verzeichnis (`plugins/serendipity_event_staticpag` bearbeiten oder auch in Ihr eigenes Template-Verzeichnis wie `templates/default/` kopieren, um sie dort anzupassen. Neue Dateien können Sie ebenfalls in beiden Verzeichnissen anlegen. Im Eingabefeld **Templatename** tragen Sie keine Verzeichnisnamen, sondern lediglich den Dateinamen der Datei ein (mit Endung `.tpl`).

Bildpfad

Das Standard-Template **plugin_staticpage_aboutpage.tpl** ermöglicht es, in der Navigation der Unterseiten für eine Elternseite Bilder einzubinden. Die Bilder der Unterseiten richten sich dabei nach Ihrer Eingabe im Feld **Bildpfad** des zugehörigen Seitentyps. Dieses Vorgehen ähnelt dem bei Kategoriebildern, die Sie in Blog-Artikeln einbinden können.

Für das Beispiel des Familienstammbaums *Meine Familie* könnten Sie die Seitentypen `Eltern` und `Geschwister` anlegen. Für beide Seitentypen können Sie denselben Template-Namen **plugin_staticpage.tpl** angeben. Für den Seitentyp **Eltern** vergeben Sie ein Bild wie `http://www.example.com/serendipity/uploads/Eltern.gif`, für **Geschwister** z. B. `http://www.example.com/serendipity/uploads/Geschwister.gif`.

Nun müssen Sie die Seite *Wolfram* bearbeiten und das Ausklappfeld **Artikeltyp** auf **Geschwister** stellen, die Seiten *Horst* und *Sabine* auf den **Artikeltyp Eltern**.

Zuletzt stellen Sie den **Artikeltyp** der Stammseite **Meine Familie** auf **Overview**, damit es die Template-Datei `plugin_staticpage_aboutpage.tpl` auslesen kann. Diese Template-Datei wertet die im Seitentyp hinterlegten Bilder aus und wird in der Navigation nun das Bild **Eltern.gif** sowohl für den Link zu *Horst* als auch zu *Sabine* anzeigen. Für die Unterseite *Wolfram* wird die Grafik **Geschwister.gif** eingebunden.

Dieses Vorgehen ist zwar mit mehreren Schritten und Eingabefeldern verbunden, ermöglicht Ihnen aber, beliebige Layoutvorhaben umzusetzen.

Da das Eingabefeld *Bildpfad* ein freies Textfeld darstellt, könnten Sie theoretisch auch andere Eigenschaften als eine Bild-URL einbinden. Wie Sie diese Variable einsetzen, obliegt einzig der Gestaltung Ihres Templates und dem Einsatz der Smarty-Variable {`$staticpage_extchildpages.X.image`}.

Es kann vorkommen, wenn Sie das Plugin **Statische Seiten** mehr als einmal installiert haben, dass Sie einige Seitentypen mehrfach in der Liste sehen. Überflüssige, doppelte Seitentypen können Sie entweder manuell löschen oder einfach beibehalten, da dadurch keine Probleme entstehen.

Diese Doppelung kann entstehen, weil bei jeder Einrichtung des Ereignis-Plugins die Installationsroutine die Standard-Seitentypen erneut in der Datenbank einträgt.

Andere Plugins

Das gekoppelte Seitenleisten Plugin *Liste der Statischen Seiten* kann abseits von den angelegten statischen Seiten auch die Ausgabeseiten einiger Plugins mit einbinden. Zu diesen Plugins zählen der *Downloadmanager*, das *Gästebuch*, *Kontaktformular*, *Forum*, die *FAQs* sowie die *Bildgallerie*.

Links zu derartigen Plugins müssten Sie andernfalls immer manuell über HTML-Klötze oder Template-Änderungen einbinden.

Über den Menüpunkt **Einträge** → **Statische Seiten** → **Andere Plugins** können Sie die Liste der unterstützten Plugins einsehen. Alle Plugins, die unterstützt und installiert sind, werden als **Plugin ist installiert** aufgeführt. Plugins, die Sie heruntergeladen, aber noch nicht über die Plugin-Verwaltung aktiviert haben, sind mit **Plugin ist verfügbar, aber nicht installiert** ausgezeichnet.

Alle unterstützen und installierten Plugins werden auf der Seite mit einer vorangestellten Ankreuzbox dargestellt. Kreuzen Sie die gewünschten Plugins zur Einbindung in der Seitenleiste an und klicken Sie auf **Los!**, damit das Seitenleisten-Plugin sich Ihren Wünschen anpassen kann.

Konfigurationsoptionen

In dem Ereignis-Plugin können Sie über die Konfigurationsoptionen festlegen, wie einige der Felder bei der Erstellung einer neuen statischen Seite belegt sind. Folgende Felder können dabei in ihrer Voreinstellung beim Anlegen neuer Seiten verändert werden:

- Textformatierung(en) durchführen
- Als Artikel formatieren
- Kinderseiten anzeigen
- Navigation anzeigen
- In der Navigation der Seitenleiste einbinden

Zusätzlich gibt es zwei weitere Konfigurationsoptionen:

Überschriften oder Vor/Zurück-Navigation anzeigen
: Die Standard-Templates der statischen Seiten können eine Navigation innerhalb der angelegten Seiten einbinden, so dass Sie zwischen mehreren statischen Seiten vor- und zurückblättern können. Dabei wird jeweils ein Link zur vorigen und zur nächsten Seite eingebunden. Der Titel dieses Links richtet sich nach der Einstellung der Option **Überschriften oder Vor/Zurück-Navigation anzeigen**. Wenn Sie hier die Option **Text: Vor/Zurück** auswählen, erscheint der feste Text *Vor* für die nächste Seite und *Zurück* für die vorhergehende Seite. Alternativ können Sie die Option **Überschrift** auswählen, dann werden die Überschriften der vorhergehenden und folgenden statischen Seiten dargestellt und bieten dem Besucher so möglicherweise eine bessere Übersicht.

Suche
: Die Volltextsuche des Blogs über das Plugin **Suche** durchsucht per Default lediglich Blog-Artikel. Wenn Sie auch statische Seiten durchsuchen möchten, müssen Sie diese Option aktivieren. Details zur Suche finden Sie auf Seite 426.

Das gekoppelte Seitenleisten-Plugin verfügt über diese Konfigurationsoptionen:

Seitenanzahl
: Standardmäßig stellt das Seitenleisten-Plugin alle vorhandenen statischen Seiten dar. Wenn Sie die Anzahl auf eine gewisse Menge beschränken wollen, können Sie dies über die Option **Seitenanzahl** tun. In den meisten Fällen ist es jedoch sinnvoller, eher gezielt statische Seiten von der Ausgabe auszunehmen, indem Sie bei einer statischen Seite die Option **In der Navigation der Seitenleiste einbinden** deaktivieren.

Startseitenlink anzeigen
: Wenn Sie die Option **Startseitenlink anzeigen** aktivieren, bindet das Plugin auch einen Link zur Blog-Startseite ein. So kann ein Besucher leicht zur Startseite zurückfinden.

Nur Eltern-Seiten darstellen
: Bei aktivierter Option **Nur Eltern-Seiten darstellen** werden in der Seitenleiste nur die statischen Seiten angezeigt, die als **Ist Elternseite** festgelegt sind. Etwaige Unterseiten werden dann nicht dargestellt.

Icons bzw. Klartext
: Wenn Sie die Option **Baumstruktur** aktivieren, stellt das Plugin die Hierarchie der statischen Seite mittels eines dynamischen JavaScripts ein. Dies kann, ähnlich wie der Windows Explorer, aufklappbare Verzeichnisstrukturen einblenden.

  Die Option **Klartext** gibt gewöhnliche HTML-Links aus. Nur diese können von Suchrobotern problemlos interpretiert werden, sehen aber nicht so hübsch aus wie die **Baumstruktur** und belegen möglicherweise mehr Sichtfläche.

Grafiken für Baumstruktur aktivieren
: Bei aktivierter **Baumstruktur** können Sie die grafischen Symbole zu jeder statischen Seite optional deaktivieren.

Verzeichnis für Bilder dieses Plugins
: Die grafischen Symbole des JavaScripts zur Darstellung einer **Baumstruktur** bezieht das Plugin über eine URL Ihres Blogs. Sie müssen Siese Option nur verändern, wenn Sie Ihr Plugin-Unterverzeichnis umbenannt oder unterschiedlich verschachtelt haben und Ihnen daher sonst keine Grafiken angezeigt werden könnten.

Template-Integration

Jeder Seitentyp einer statischen Seite kann einer eigenen Smarty-Template-Datei zugeordnet werden. Standardmäßig stehen die Dateien `plugin_staticpage.tpl` und `plugin_staticpage_aboutpage.tpl` zur Verfügung.

Innerhalb dieser Datei kann eine Vielzahl von Smarty-Variablen und -Funktionen eingesetzt werden:

Funktion {`getCategoryLinkByID`}
: Die Smarty-Funktion {`getCategoryLinkByID`} wird innerhalb der Plugin-Datei

`smarty.inc.php` definiert und kann in jeder beliebigen Smarty-Template-Datei (auch in den zentralen Serendipity-Dateien!) dazu benutzt werden, die URL (den *Permalink*) zu einer gewissen Blog-Kategorie zu erhalten.

Als Parameter der Smarty-Funktion kann `cid=`*`ID der Kategorie`* übergeben werden. Als Rückgabewert liefert die Funktion eine URL.

Wenn Sie in einer Template-Datei einen Code wie:

```
<a href="getCategoryLinkByID cid=4">Link</a>
```

eintragen, erhalten Sie eine HTML-Ausgabe wie:

```
<a href="/serendipity/categories/4-Kategoriename.html">Link</a>
```

Diese Funktion ist vor allem dann hilfreich, wenn Sie die Kategoriezuordnung für statische Seiten (siehe Seite 427) verwenden möchten.

Funktion {`staticpage_display`}

Mit der Funktion {`staticpage_display`} können Sie innerhalb einer beliebigen Smarty-Template-Datei die Ausgabe einer statischen Seite auslösen.

Hilfreich ist dies, wenn Sie innerhalb einer Template-Datei einer statischen Seite eine weitere statische Seite einbetten wollen, oder wenn Sie in der Blog-Übersicht den Inhalt einer statischen Seite anzeigen wollen. Wenn Sie das Plugin *Textformatierung: Smarty* (siehe Seite 372) verwenden, können Sie statische Seiten auch einfach in Blog-Artikeln einbinden.

Stellen Sie sich nun folgendes Szenario vor: Sie legen zwei statische Seiten an. Eine hat den Titel `Impressum`, die andere den Titel `Über mich`. Die statische Seite `Impressum` soll den Seitentyp `Article` benutzen, um mittels der Template-Datei `plugin_staticpage.tpl` ausgegeben zu werden.

Als fauler Blogger möchten Sie nun gerne, dass der Inhalt der Seite `Impressum` auf der Seite `Über mich` erscheint, denn Sie wollen den Text ja nicht doppelt ändern müssen, wenn Sie einmal umziehen.

Um dies zu erreichen, müssen Sie einen neuen Seitentyp anlegen. Nennen Sie diesen z. B. `Kombiseite`. Als Vorlage für das Template übernehmen Sie die Datei `plugin_staticpage.tpl` und kopieren sie als neue Datei `kombiseite.tpl` entweder in Ihr Plugin-Verzeichnis oder Ihr eigenes Template-Verzeichnis. Den Namen der kopierten Template-Datei tragen Sie für den Seitentyp ein und weisen nun die Seite `Über mich` diesem Seitentyp zu.

Nun erfolgt der eigentliche Schritt, mit dem Sie den Inhalt der Seite `Impressum` übernehmen können. Sie öffnen die Datei `kombiseite.tpl` und suchen nach der Stelle, wo der Text einer statischen Seite mit der Variable {`$staticpage_content`} ausgegeben wird. Da Sie das Impressum danach ausgeben wollen, ändern Sie die Zeile

```
<div class="staticpage_content">{$staticpage_content}</div>
```

um in:

```
<div class="staticpage_content">
  {$staticpage_content}
  {staticpage_display pagetitle="Impressum"
    template="plugin_staticpage.tpl"}
</div>
```

Nun wird der Inhalt der statischen Seite mit dem Seitentitel `Impressum` vollständig ausgegeben. Als Template-Datei wird die Datei `plugin_staticpage.tpl`[11] verwendet.

Wenn Sie anstelle eines vollständigen Inhalts der statischen Seite nur Teile des Textes (beispielsweise nur {`$staticpage_content`}) ausgeben wollen, können Sie dafür ein eigenes Template anlegen, das nur diese Smarty-Variable verwendet. Diese Template-Datei können Sie dann der Smarty-Funktion {`staticpage_display`} übergeben.

Die Funktion kann mit mehreren Parametern aufgerufen werden:

```
{staticpage_display
  pagevar='...'
  template='...'
  id='...'
  permalink='...'
  pagetitle='...'
  authorid='...'
  query='...'}
```

`pagevar` (optional)
: Mit dieser Option legen Sie ein Variablen-Präfix fest. Standardmäßig verwendet jede Template-Datei der statischen Seiten Variablen mit einem Vorzeichen wie {`$staticpage_pagetitle`}. Wenn Sie ein anderes Präfix verwenden wollen, um Kollisionen in den Variablennamen zu vermeiden, können Sie dieses mit dem Parameter `pagevar` festlegen.

`template` (optional)
: Standardmäßig benutzt die Smarty-Funktion zur Darstellung der statischen Seite das Template, das für die jeweilige Seite im **Artikeltyp** eingetragen wurde.

  Wenn Sie eine individuelle Template-Datei verwenden wollen, können Sie den Dateinamen über diesen Parameter angeben. Die Template-Datei können Sie entweder im Plugin-Verzeichnis `plugins/serendipity_event_staticpage` oder im jeweiligen eigenen Template-Verzeichnis speichern.

[11] Hätten Sie als Template `kombiseite.tpl` eingetragen, würde dies zu einer Endlosschleife führen!

`id, permalink, pagetitle`
: Die Smarty-Funktion {`staticpage_display`} kann nur eine eindeutige statische Seite darstellen. Daher müssen Sie als Parameter unbedingt entweder `id`, `permalink` oder `pagetitle` angeben.

  Am sichersten ist die Übermittlung der ID einer statischen Seite, jedoch ist diese ID auch nur anhand der Datenbank herauszufinden. Leichter ist es, eine statische Seite über den konfigurierten **Permalink** oder den **URL-Titel** der Seite aufzurufen, da dieser auch eindeutig für jede statische Seite sein muss.

`authorid`
: Zusätzlich zur Einschränkung via `id, permalink` und `pagetitle` können Sie eine statische Seite auch auf einen Autor einschränken. Geben Sie dafür die ID des Autors als Wert des Parameters an.

`query`
: Letztlich führen die Parameter `id`, `permalink`, `pagetitle` und `authorid` lediglich dazu, eine Datenbank-Abfrage zusammenzustellen. Jeder der genannten Parameter wird für das Bedingungsfeld der SQL-Abfrage (`WHERE`) ausgewertet und bildet am Ende eine Abfrage wie:

  ```
  SELECT *
    FROM serendipity_staticpages
   WHERE pagetitle = 'Impressum'
   LIMIT 1
  ```

  Wenn Sie eine komplexere Datenbankabfrage benötigen, können Sie über den Parameter `query` der Smarty-Funktion eine eigene SQL-Abfrage übergeben. Diese Abfrage darf maximal einen Datensatz liefern.

  Wenn Sie diesen Parameter verwenden, müssen Sie trotzdem mindestens einen der Parameter `id, permalink, pagetitle` angeben, obwohl diese in Ihrer eigenen SQL-Abfrage möglicherweise nicht ausgewertet werden. Sie können daher auch einen Fantasiewert wie `id=42` übermitteln, wenn Sie eine `query` übergeben.

  Beispielsweise könnten Sie folgenden Aufruf verwenden, um eine als Startseite definierte Seite einzubinden:

  ```
  {staticpage_display id=42 query="SELECT * FROM
  serendipity_staticpages WHERE is_startpage = 1 LIMIT 1"}
  ```

Folgende Smarty-Variablen stehen in allen Template-Dateien der statischen Seiten bzw. Seitentypen zur Verfügung:

{`$staticpage_*`} (Mixed)
: Zu jeder statischen Seite werden mehrere Felder ausgefüllt, die allesamt in eigenen Datenbankspalten gespeichert werden. Diese Felder umfassen den Titel der Seite, den Autor, den Inhalt und alles Weitere.

Jedes dieser Felder enthält eine Template-Variable, die Sie in einer Template-Datei ansprechen können. Der Teil nach `staticpage_...` entspricht dabei dem Namen des Datenbankfeldes.

Diese Variablen beziehen sich jeweils auf die aktuell im Template dargestellte Seite:

{`$staticpage_headline`} (String)
: enthält die **Kopfzeile**.

{`$staticpage_permalink`} (String)
: enthält den **Permalink**.

{`$staticpage_pagetitle`} (String)
: enthält den **URL-Titel der Seite**.

{`$staticpage_articletype`} (String)
: enthält den Namen des zugeordneten Seitentyps.

{`$staticpage_publishstatus`} (Bool)
: ist auf `true` gesetzt, wenn die Seite veröffentlicht wurde.

{`$staticpage_language`} (String)
: enthält das Kürzel der **Sprache**.

{`$staticpage_content`} (String)
: enthält den Inhaltstext.

{`$staticpage_markup`} (Bool)
: ist auf `true` gesetzt, wenn **Textformatierungen** für diese Seite durchgeführt werden sollen.

{`$staticpage_articleformat`} (Bool)
: ist auf `true` gesetzt, wenn eine statische Seite **als Artikel formatiert** wird.

{`$staticpage_articleformattitle`} (String)
: enthält den **Seitentitel für „Als Artikel formatieren“-Ansicht**.

{`$staticpage_authorid`} (Int)
: enthält die ID des Autors.

{`$staticpage_parent_id`} (Int)
: enthält die ID einer übergeordneten statischen Seite. Bei Elternseiten ist dieser Wert auf `0` gesetzt.

{`$staticpage_related_category_id`} (Int)
: enthält die ID einer zugeordneten Kategorie.

{`$staticpage_show_childpages`} (Bool)
: ist auf `true` gesetzt, wenn Unterseiten angezeigt werden sollen.

{`$staticpage_pre_content,`

`$staticpage_precontent`} (String) enthält die **Einleitung**, die bei Elternseiten eingetragen werden kann.

{`$staticpage_pass`} (String)
: enthält das Passwort (als Klartext) zum Aufruf der Seite.

{`$staticpage_is_startpage`} (Bool)
: ist auf `true` gesetzt, wenn die statische Seite die Startseite des Blogs darstellen soll.

{`$staticpage_pageorder`} (Int)
: enthält eine Zahl, die für die Sortierungsreihenfolge einer statischen Seite benutzt wird. Je höher die Zahl, desto weiter am Ende einer Liste (beispielsweie in der Seitenleiste oder Navigation) wird die Seite ausgegeben.

{`$staticpage_shownavi`} (Bool)
: ist auf `true` gesetzt, wenn die Navigation in einer statischen Seite eingebunden werden soll.

{`$staticpage_showonnavi`} (Bool)
: ist auf `true` gesetzt, wenn eine statische Seite in der Ausgabe der Seitenleiste enthalten sein soll.

{`$staticpage_timestamp`} (Int)
: enthält den UNIX-Zeitstempel (vergangene Sekunden seit dem 01.01.1970) für den Zeitpunkt, an dem die statische Seite erzeugt wurde.

{`$staticpage_last_modified`} (Int)
: enthält den UNIX-Zeitstempel für den Zeitpunkt, an dem die statische Seite zuletzt bearbeitet wurde.

{`$serendipityArchiveURL`} (String)
: Diese Variable enthält die URL zu dem Artikelarchiv Ihres Blogs, standardmäßig `/serendipity/archi`

{`$staticpage_form_pass`} (String)
: Enthält das Passwort, das ein Besucher übermittelt hat. Dies kann zum Vergleich mit dem für die statische Seite eingetragenen Passwort herangezogen werden.

{`$staticpage_form_url`} (String)
: Enthält die URL, die als Formularziel für die Passworteingabe verwendet werden muss, um zur gewünschten statischen Seite zu gelangen.

{`$staticpage_childpages`} (Array)
: Dieses verschachtelte Array enthält eine Liste aller Seiten, die der aktuellen statischen Seite untergeordnet sind. Als Array-Schlüssel stehen der URL-Titel der Seite ({`$staticpage_childpages.X.pagetitle`}) sowie der Permalink ({`$staticpage_childpag malink`}) zur Verfügung.

{`$staticpage_extchildpages`} (Array)
: Die Variable {`$staticpage_childpages`} enthält lediglich eine kurze Übersicht der Unterseiten einer statischen Seite, die zu Navigationszwecken benutzt werden.

Die Variable {`$staticpage_extchildpages`} enthält jedoch eine ausführlichere Liste dieser Seiten (*ext* entspricht *extended* = erweitert). Dieses Array enthält folgende Array-Schlüssel:

{`$staticpage_extchildpages.X.image`} (String)
: enthält den Wert, den Sie bei dem zugehörigen Seitentyp der statischen Seite unter **Bildpfad** (siehe Seite 416) eingetragen haben. Dieser Wert kann beispielsweise eine Grafik für die jeweilige Unterseite repräsentieren, die in der Navigation für grafische Zuordnungen dienen kann.

{`$staticpage_extchildpages.X.precontent`} (String)
: enthält eine (optionale) Einleitung, die für weitere Unterseiten einer Unterseite dienen kann.

{`$staticpage_extchildpages.X.permalink`} (String)
: enthält die URL zu der Unterseite.

{`$staticpage_extchildpages.X.pagetitle`} (Int)
: enthält den `URL-Titel der Seite`.

{`$staticpage_extchildpages.X.headline`} `(Int)`
: enthält die Überschrift der Unterseite.

{`$staticpage_adminlink`} (String)
: Enthält eine URL, über die ein Administrator die jeweilige statische Seite im Backend bearbeiten kann.

{`$staticpage_navigation`} (Array)
: Dieses verschachtelte Array enthält die notwendigen Daten zur Einbindung einer Navigation. Folgende Array-Schlüssel stehen zur Verfügung:

  {`$staticpage_navigation.prev.name`}
  : enthält den Linktitel der vorigen statischen Seite (abhängig von der Konfigurationsoption **Überschriften oder Vor/Zurück-Navigation anzeigen** des Ereignis-Plugins).

  {`$staticpage_navigation.prev.link`}
  : enthält die URL der vorigen statischen Seite.

  {`$staticpage_navigation.next.name`}
  : enthält den Linktitel der nächsten statischen Seite.

  {`$staticpage_navigation.next.link`}
  : enthält die URL der nächsten statischen Seite.

  {`$staticpage_navigation.top.name`}
  : enthält den Linktitel der übergeordneten statischen Seite.

  {`$staticpage_navigation.top.link`}
  : enthält die URL der übergeordneten statischen Seite.

{`$staticpage_navigation.crumbs`}
: enthält ein durchnummeriertes Array, das von vorne nach hinten die Auflistung der übergeordneten statischen Seiten enthält. Jedem Array-Schlüssel ist hierbei ebenfalls `.name` und `.link` zugeordnet. Bei einer Unterseite der dritten Ebene enthielte dieses Array also zwei Elemente mit den URLs und Linknamen der Oberseite der ersten und zweiten Ebene.

{`$staticpage_author`} (String)
: Enthält den Autorennamen (im Gegensatz zur ID mittels {`$staticpage_authorid`}) der statischen Seite.

{`$staticpage_created_on`} (Int)
: Enthält den UNIX-Zeitstempel (vergangene Sekunden seit dem 01.01.1970) für den Zeitpunkt, an dem die statische Seite erzeugt wurde. Identisch zu {`$staticpage_timestamp`}.

{`$staticpage_lastchange`} (Int)
: Enthält den UNIX-Zeitstempel für den Zeitpunkt, an dem die statische Seite zuletzt bearbeitet wurde. Identisch zu {`$staticpage_last_modified`}.

{`$staticpage_categorypage`} (Array)
: Das Plugin *Statische Seiten* ist auch aktiv, während Sie keine statische Seite anzeigen. Dadurch kann das Plugin auch Smarty-Variablen zuweisen, die Verweise zu statischen Seiten enthalten, während ein Beusucher die Übersichtsseite der Einträge pro Kategorie anschaut.

  Wenn Sie einer Kategorie eine Kategorieseite zugewiesen haben (siehe Seite 427), dann wird die Smarty-Variable {`$staticpage_categorypage`} mit den Werten für diese statische Seite belegt. Daher können Sie mithilfe dieser Variable die statische Seite in eine zentrale Template-Datei wie `index.tpl` oder `entries.tpl` einbinden.

  Als Array-Schlüssel sind alle Werte der Datenbanktabelle verfügbar, wie auf Seite 422 aufgeführt (`id`, `pagetitle`, `permalink`, `content` etc.).

Volltextsuche

Wenn Sie in der Konfiguration des Ereignis-Plugins die Option **Suche** aktiviert haben, kann ein Besucher nicht nur Blog-Artikel, sondern auch statische Seiten durchsuchen. Die Suchergebnisse der statischen Seiten werden dabei separat ausgegeben, da sie nicht in die Formatierung der Suchergebnisse von Blog-Artikeln eingebunden werden können.

Standardmäßig wird für jedes Suchergebnis ein Link zu der jeweiligen Seite sowie die Überschrift, der Autorenname und die ersten 200 Zeichen des Inhalts ausgegeben. Die Formatierung der Suchergebnisse können Sie über die Smarty-Template-Datei `plugin_staticpage_searchresults` anpassen. Folgende Variablen stehen dabei zur Verfügung:

{`$staticpage_searchresults`} (Int)
: enthält die Anzahl der Suchergebnisse

{`$staticpage_results`} (Array)
: enthält ein verschachteltes Array mit allen Suchergebnissen. Jedem Array-Eintrag sind jeweils die Schlüsselnamen der Datenbank-Spaltennamen zugewiesen, also beispielsweise {`$staticpage_results.X.headline`}. Die Liste aller Spaltennamen finden Sie auf Seite 430.

Kategoriezuordnung

Zwischen Blog-Artikeln und statischen Seiten besteht üblicherweise kein Bezug. Beide werden unabhängig voneinander im System verwaltet. Wenn statische Seiten angezeigt werden, ignoriert das Plugin die Einbindung von Artikeln. Andersherum ignoriert Serendipity statische Seiten beim Darstellen von Artikeln.

Grundsätzlich ist dies gewollt, da man mit statischen Seiten ja exakt eine solche Trennung von redaktionellen Inhalten und *Nebeninhalten* erreichen möchte. Wenn Sie Serendipity benutzen, um eine CMS-Seite zu pflegen, möchten Sie jedoch gerade eine Verflechtung von Artikeln und statischen Seiten erreichen.

Dies können Sie mit Hilfe des Kategoriesystems von Serendipity und des Plugins *Statische Seiten* erreichen. Die Verflechtung besteht dabei aus zwei Komponenten: die Einbindung von Kategorien in eine statische Seite und umgekehrt die Einbindung von statischen Seiten in eine Kategorie.

Mithilfe eines Beispiels sollte dies etwas verständlicher werden: Sie betreuen den Webauftritt einer Kirchengemeinde. Auf der Webseite wollen Sie über mehrere Themenbereiche berichten: *Unternehmungen*, *Kirchenleben* und *Mitarbeiter*. In jedem dieser Bereiche möchten Sie gerne sowohl aktuelle Artikel als auch feste Informationen anbieten.

Dazu richten Sie zuerst einmal die drei Kategorien im Blog ein. So können Sie bereits einfach aktuelle Artikel schreiben. Es fehlt Ihnen jedoch die Möglichkeit, auch eine statische Seite mit grundlegenden Informationstexten einzupflegen. Beispielsweise möchten Sie im Themenbereich *Mitarbeiter* stets eine vollständige Liste aller Mitarbeiter darstellen und Neuigkeiten erst darunter einbinden. Eine solche Seite dient später dem Besucher als optimale Verbindung: Er sieht immer aktuelle und statische Informationen.

Als Einstiegspunkte zu so einer Seite gibt es nun zwei Möglichkeiten: Der Besucher kann entweder die Artikelübersicht der Kategorie aufrufen (`http://www.example.com/serendipity/categories/1-Mit`
oder er kann gezielt zu einer statischen Seite springen, die der Kategorie zugeordnet ist (`http://www.example.com/serendipity/seiten/Mitarbeiter.html`).

Diese beiden Verknüpfungsoptionen sind in zwei unterschiedlichen Menüs untergebracht. Das erste Menü befindet sich beim Bearbeiten einer statischen Seite über **Einträge** → **Statische Seiten** → **Seiten bearbeiten** im Ausklappfeld **Zugeordnete Kategorie**. Wichtig ist, dass Sie

einer statischen Seite, die einer Kategorie zugeordnet ist, als Seitentyp **Staticpage with related category** zuweisen. Nur diese Formatierungsansicht enthält die spezielle Kopplung, die auf einer statischen Seite die Blog-Artikel einer gewählten Kategorie anzeigt.

Als zweite Verknüpfungsmöglichkeit kann ein Redakteur einer Kategorie eine beliebige statische Seite zuordnen, indem er im Menü **Einträge** → **Kategorien** die betreffende Kategorie bearbeitet und im Ausklappfeld **Zugeordnete statische Seite** die jeweilige Seite auswählt. Damit nun eine verbundene statische Seite angezeigt werden kann, wenn ein Besucher die betreffende Kategorie aufruft, müssen Sie Änderungen an Ihrem gewählten Blog-Template vornehmen. Andernfalls ist eine Verknüpfung nur konfiguriert, wird aber nicht ausgewertet.

Beide Verknüpfungsoptionen sind für erfahrenere Template-Bauer gedacht. Die folgenden Detailbeschreibungen sollten Sie daher möglichst erst dann umsetzen, wenn Sie sich mit Templates (siehe Beschreibung ab Seite 480) bereits eingehender beschäftigt haben.

Zuordnung Kategorie zu statischer Seite

Einer statischen Seite können Sie eine Kategorie zuordnen. Beim Ansehen der statischen Seite werden dann z. B. die aktuellsten fünf Einträge einer Kategorie unterhalb des regulären Inhalts eingebunden.

Diese Einbindung erfolgt mittels eines speziellen Seitentyps namens **Staticpage with related category**. Dieser Seitentyp verwendet die Template-Datei `plugin_staticpage_related_cat` in der Smarty-Funktionsaufrufe vorhanden sind, die die zugeordnete Kategorie auswerten und darstellen.

Sie können unterschiedlichen statischen Seiten dieselbe Kategorie zuordnen, d. h., Sie können sowohl der statischen Seite `Impressum` als auch `Mitarbeiter` die Blog-Kategorie `Mitarbeiter` zuweisen. Dann werden die aktuellsten Artikel dieser Blog-Kategorie auf beiden statischen Seiten eingebunden.

Die Anzeige dieser Funktionalität erfolgt, wenn ein Besucher gezielt eine statische Seite aufruft (`http://www.example.com/serendipity/seiten/Mitarbeiter.html`).

In der Datei `plugin_staticpage_related_category.tpl` befindet sich der bereits aus der Datei `plugin_staticpage.tpl` bekannte Code zur gewöhnlichen Darstellung einer statischen Seite. Abweichend von der gewöhnlichen Darstellung enthält die Template-Datei keine Navigationselemente und zusätzlich den besonderen Smarty-Funktionsaufruf zur Einbindung der Blog-Artikel:

```
<div class="staticpage_related_category_entry_list">
{serendipity_fetchPrintEntries
category=$staticpage_related_category_id
template="../../plugins/serendipity_event_staticpage/staticpage-ent
ries-listing.tpl"
limit=5 noSticky="true"} </div>
```

Die Beschreibung der Smarty-Funktion {`serendipity_fetchPrintEntries`} finden Sie auf Seite 568. Im vorliegenden Fall weist die Funktion Serendipity dazu an,

die letzten fünf Artikel darzustellen, die der aktuellen statischen Seite zugeordnet sind. Die Kategorie-ID wird dabei über die Smarty-Variable {`$staticpage_related_category_id`} übermittelt.

Damit die Ausgabe der Artikel nur als Übersicht erfolgt, benutzt der Smarty-Aufruf ein eigenständiges Template namens `staticpage-entries-listing.tpl`, das mit dem Plugin ausgeliefert wird. Sie können die genannte Datei zur Anpassung in Ihr eigenes Template-Verzeichnis kopieren und sich dadurch die Angabe des vollständigen Pfades sparen.

Die Datei `staticpage-entries-listing.tpl` entspricht einem stark reduzierten `entries.tpl`-Template (siehe Seite 519). Es gibt lediglich die Überschrift und den Link zu den jeweiligen Artikeln aus. Sie können jedoch auch alle verfügbaren Smarty-Variablen anwenden, um die Blog-Artikelliste individuell zu formatieren. Dazu können Sie auch eine eigene Template-Datei wie `staticpage-entries-listing-detail.tpl` erstellen. Auf diese müssen Sie dann in der Datei `plugin_staticpage_related_category` verweisen oder alternativ auch einen eigenständigen Seitentyp mit eigener Template-Datei verwenden.

Grundsätzlich können Sie sämtliche Parameter des Smarty-Funktionsaufrufs auch durch zentrale Template-Optionen abstrahieren. Dabei könnten Sie in Ihrem Template beispielsweise eine konfigurierbare Option für die Anzahl der Blog-Artikel in einer statischen Seite hinterlegen. Eine Beschreibung der Template-Optionen finden Sie auf Seite 499.

Zuordnung statische Seite zu Kategorie

Diese Verknüpfung kümmert sich um die Einbindung einer statischen Seite in eine Artikelübersicht einer Kategorie. Diese statische Seite wird einer Kategorie im Bereich **Einträge** → **Kategorien** zugewiesen.

Einer Kategorie können Sie lediglich eine einzelne statische Seite zuordnen, genauso wie Sie einer statischen Seite nur eine einzelne Kategorie zuordnen können. Das bedeutet daher auch, dass Sie mehreren unterschiedlichen Kategorien dieselbe statische Seite zuordnen können, und mehreren unterschiedlichen statischen Seiten dieselbe Kategorie.

Angezeigt wird die zugewiesene statische Seite, wenn ein Besucher die Artikelübersicht über eine URL wie `http://www.example.com/serendipity/categories/1-Mitarbeiter` aufruft.

Serendipity stellt eine Kategorieübersicht über die `entries.tpl` Ihres aktivierten Templates dar. Dort ist eine statische Seite üblicherweise nicht vorgesehen, daher müssen Sie die Einbindung dieser statischen Seite selbständig vornehmen.

Das Grundprinzip ist dabei analog der Zuordnung einer Kategorie zu einer statischen Seite. Sie ändern dabei die `entries.tpl` so ab, dass sie den Aufruf der zugeordneten statischen Seite enthält.

Das Plugin *Statische Seiten* kann selbständig erkennen, ob der Kategorie, die ein Besucher gerade anschaut, eine statische Seite zugeordnet ist. Ist dies der Fall, setzt

das Plugin die Smarty-Variable {`$staticpage_categorypage`}. Diese Variable enthält alle Werte `id`, `pagetitle`, `permalink`, `content` etc. der zugewiesenen statischen Seite (siehe Seite 423).

Um also bei der Betrachtung einer Artikelübersicht zu einer Kategorie die zugewiesene statische Seite zu verlinken, könnten Sie am Anfang Ihrer Datei `entries.tpl` Folgendes einfügen:

```
{if $staticpage_categorypage}
<div class="staticpage_header">
    Lesen Sie ausführliche Informationen zu dieser Kategorie auf
    der Seite <a href="{$staticpage_categorypage.permalink}">{$stat
icpage_categorypage.pagetitle}</a>
</div>
{/if}
```

Dieser Code gibt einen Hinweistext und einen Link zu der zugehörigen statischen Seite aus. Diesen Block Code können Sie selbstverständlich auch in andere Dateien Ihres Templates, wie `index.tpl`, einbauen.

Datenbanktabellen

Das Plugin richtet folgende Datenbanktabellen und -felder ein:

`serendipity_staticpages`
: In dieser Tabelle werden die Inhalte der statischen Seiten gespeichert. Folgende Felder werden benutzt:

  `id`
  : enthält den fortlaufenden Primärschlüssel zur Identifikation einer statischen Seite.

  `parent_id`
  : enthält eine etwaige zugeordnete Oberseite.

  `articleformattitle`
  : enthält den **Seitentitel für „Als Artikel formatieren“-Ansicht**.

  `articleformat`
  : gibt an, ob eine statische Seite in der Darstellung wie ein Artikel formatiert werden soll.

  `markup`
  : gibt an, ob Textformatierungs-Plugins angewendet werden sollen.

  `pagetitle`
  : enthält den **URL-Titel der Seite**.

  `permalink`
  : enthält den **Permalink**.

`is_startpage`
: gibt an, ob die Seite als Startseite des Blogs angezeigt wird.

`show_childpages`
: gibt an, ob etwaige Unterseiten angezeigt werden sollen.

`content`
: enthält den Inhalt der statischen Seite.

`pre_content`
: enthält den einleitenden Inhalt für Kinderseiten der statischen Seite.

`headline`
: enthält die Überschrift des Artikels.

`filename`
: enthält den Dateinamen des Templates, der für die Formatierung der Seite verwendet werden soll.

`pass`
: enthält ein Passwort, falls die Seite geschützt werden soll.

`timestamp`
: enthält die Erstellungszeit des Artikels.

`last_modified`
: enthält die Zeit der letzten Aktualisierung des Artikels.

`authorid`
: enthält die ID des Autors, der die Seite erstellt hat.

`pageorder`
: enthält ein Ordnungskriterium zur Sortierung der statischen Seite.

`articletype`
: gibt an, welchem Seitentyp die Seite entspricht.

`related_category_id`
: enthält eine möglicherweise festgelegte Kategoriezuordnung der statischen Seite.

`shownavi`
: gibt an, ob Navigationsmöglichkeiten im Template der statischen Seite angezeigt werden sollen.

`showonnavi`
: gibt an, ob die statische Seite in der Liste der verfügbaren Seiten eingebunden werden soll.

`publishstatus`
: legt den Status des Artikels fest (veröffentlicht/Entwurf).

`language`
: legt die Sprache des Artikels fest.

`serendipity_staticpages_types`
: Enthält die möglichen Seitentypen der statischen Seiten.

  `id`
  : enthält den fortlaufenden Primärschlüssel zur Identifikation des Seitentyps.

  `description`
  : enthält die Beschreibung des Seitentyps.

  `template`
  : enthält die verwendete Smarty-Template-Datei zur Formatierung von Artikeln dieses Typs.

  `image`
  : enthält eine Bild-URL, die zur Identifizierung von Artikeln dieses Typs im Frontend dargestellt werden kann.

`serendipity_staticpages_categorypage`
: Enthält die Zuordnungen einer Blog-Kategorie zu einer statischen Seite.

  `categoryid`
  : enthält die ID der Blog-Kategorie.

  `staticpage_categorypage`
  : enthält die ID der statischen Seite.

Kapitel 8

Wartung und Betrieb

Grundsätzlich sollten Sie jetzt in der Lage sein, Ihre Artikel in einer gewünschten Hierarchie zu veröffentlichen, kommentieren zu lassen und mit Plugins zu erweitern. Das bisher Erreichte ist vergleichbar mit dem Häuslebau: Jetzt steht die Villa, sieht von außen auch gut aus, ist aber noch unbewohnt, und die Hausfront ist unbewacht vor Graffiti-Vandalen; zudem fehlt noch die Versicherung für Wertsachen im Haus.

Daran wollen wir in den kommenden Kapiteln etwas ändern. Sie erfahren, wie Sie mit Kommentaren umgehen können, Datenbanksicherungen erstellen, Plugins und das Kernsystem aktualisieren und sich so gut wie möglich vor Spam schützen können.

8.1 Einträge und Trackbacks

Die Vernetzung von Blogs untereinander ist ein wichtiger Grund für die hohe Relevanz von Blogs heutzutage. Blogs haben Zeitschriftencharakter, und einzelne interessante Artikel werden häufig von anderen Bloggern aufgegriffen und weitergetragen.

So kann sich eine Neuigkeit rasend schnell verbreiten, und das hat in der Vergangenheit schon oft dazu geführt, dass Blogs einen großen Aktualitätsvorsprung vor klassischen Informationsmedien haben. Populäre Nachrichtensendungen werden erst am Tagesende ausgestrahlt, Radionachrichten erfolgen meist nur stündlich – da kann ein Blog mit relevanten Nachrichten wie Reaktionen auf eine Pressekonferenz oder Vor-Ort-Berichten wie beim Bombenattentat in London weitaus schneller reagieren. Dank RSS-Feeds (siehe Terminologie auf Seite 38) ist es für Besucher zudem leicht nachzuverfolgen, welche Blogs neue Nachrichten geliefert haben.

Private Blogger fühlen sich jedoch selten an einen journalistischen Kodex gebunden – eine Falschmeldung kann so möglicherweise schnell verbreitet werden und nur schwer zu korrigieren sein. Genau diese fehlende Kontrollinstanz und die freie Meinungsäußerung in Blogs

ist großen Unternehmen oft ein Dorn im Auge und der Grund, warum sich Firmen nur distanziert mit Blogs beschäftigen.

Mittlerweile hat man aber auch den positiven Nebeneffekt durch die schnelle Verbreitung bemerkt: Gerade Werbeagenturen nutzen Blogger oft bewusst dazu aus, um Werbekampagnen voranzutreiben. Sogenannte *Virale Kampagnen* zielen darauf ab, dass sie schnell von einem zum nächsten Blog gelangen.

Die technische Basis für schnelle Verbreitung und Vernetzung stellt neben den RSS-Feeds (für Aktualisierungshinweise) eine Technik namens *Trackback* dar.

Ein Trackback ermöglicht es Ihnen, sich auf ein anderes Blog als Quelle zu beziehen. Wenn Sie nun die Quelle in Ihrem Artikel nennen, kümmert sich Serendipity mittels einer automatischen Schnittstelle darum, dass in dem Quellblog ein Verweis auf Ihren neuen Artikel hinterlegt wird. Leser des Quellblogs können nun auf Ihre Seite gelangen und Ihre Sicht der Dinge nachlesen. Möglicherweise beziehen sich danach weitere Redakteure auf Ihren Eintrag, und es entsteht eine Netzstruktur von Verweisen.

Trackbacks sind dabei für beide Seiten interessant. Der Ursprungsautor erfährt so davon, wer seinen Artikel gelesen hat und dazu etwas ergänzen oder diskutieren möchte. Und Sie als Bezug nehmender Autor können zusätzliche Leser gewinnen, die im Ursprungsblog auf Sie aufmerksam werden.

Im Endeffekt ermöglichen Trackbacks, dass Sie Diskussionen dezentral führen können, losgelöst von dem Ursprungsblog. In früheren Zeiten wurden Diskussionen stets zentral an einer Stelle geführt, also in einem Bereich eines Forums oder (noch früher) in einer Usenet- oder Mailboxgruppe. Der Community-Gedanke steht dort im Vordergrund, während Weblogs die *Individualisierung einer Meinung* verfolgen. In Ihrem persönlichen Weblog stehen Sie als Individuum im Zentrum, sämtliche Inhalte sind grundsätzlich auf Ihre Meinung ausgelegt. Bei Diskussionsforen ist man nur Teil eines Ganzen.

Welcher Variante Sie den Vorzug geben, obliegt Ihrem eigenen Geschmack und den Anforderungen Ihres Themas. Ein paar Beispiele:

- In einem Blog lesen Sie einen Artikel zu einer rechtlichen Problematik, der konkret mit einer Frage endet. Wenn Sie die Antwort zu dieser Frage wissen, macht es wenig Sinn, dass Sie dazu einen eigenen Artikel in Ihrem Blog schreiben und ein Trackback zu dem Blog des Fragenden hinterlassen. Dadurch würden Sie eine unnötig hohe Barriere für den Ursprungsautoren aufbauen, zu der Antwort zu gelangen. Hier wäre es sinnvoller, einfach nur einen Kommentar im Blog des Fragenden zu hinterlassen.

- In einem Blog wird heftig über den Unsinn der GEZ-Gebühren für Internet-PCs diskutiert. Es gibt bereits 300 Kommentare zu diesem Artikel. Sie haben jedoch eine grundsätzlich andere Auffassung zu der Thematik, die in den Kommentaren des Eintrags niedergeschmettert und größtenteils ignoriert wird. Weil Ihnen Ihre Meinung jedoch wichtig ist, erstellen Sie einen eigenen Blog-Artikel, in dem Sie Ihre Meinung nochmals ausführlich darlegen. Mittels eines Trackbacks zu dem Ursprungsartikel ermöglichen Sie es den Interessierten, Ihre abweichende Meinung gesondert zu

diskutieren, und behalten den Ursprungsbezug bei.

- In seinem persönlichen Blog fragt ein Film-Fan seine Leser, welches deren Lieblingsfilme sind. Er bittet um ausführliche Erklärung. Da Sie selbst ein Film-Blog führen und schon immer mal einen langen Artikel über Ihren Lieblingsfilm verfassen wollten, schreiben Sie direkt einen Blog-Artikel dazu und hinterlassen auf dem Blog des Film-Fans ein Trackback. Der Film-Fan entdeckt später einen Fehler in Ihrer Erklärung des Films und hinterlässt auf Ihrem Blog einen Kommentar dazu – anstatt auf seinem eigenen Blog dazu Stellung zu beziehen.

8.1.1 Trackbacks und Pingbacks senden

Wenn Sie sich dazu entschieden haben, einen Blog-Artikel zu verfassen, der sich auf einen anderen Beitrag beziehen soll, dann können Sie bei Serendipity sehr leicht automatisch ein Trackback setzen.

Serendipity durchsucht jeden Ihrer Einträge nach Hyperlinks, wenn Sie den Artikel speichern. Dazu sucht es nach allen `<a>`-HTML-Tags Ihres Eintrags und ruft die darin angegebene Internet-Adresse auf. Auf dieser Seite sucht Serendipity nach einem speziellen Code[1], den ein Blog enthalten muss, um die Trackback-Schnittstelle anzubieten.

Findet Serendipity diesen Code, sendet es einen Ausschnitt Ihres Artikeltexts zusammen mit der URL Ihres Beitrags an das betreffende Blog. Dabei stellt Ihr Blog den Sender dar, das fremde Blog ist der Empfänger. Die gesendeten Daten müssen vom Zielblog verarbeitet und gespeichert werden. Die Rückmeldung über den Fortschritt des Vorgangs wird dabei direkt von Serendipity beim Speichern dargestellt.

Ohne den entsprechenden Code versucht Serendipity, Ihre Artikel-URL mit der Ergänzung `/trackback/` aufzurufen. Die meisten WordPress-Blogs verwenden dieses Schema, geben aber den von Serendipity benötigten Code nicht aus. Damit ein Trackback in so einem Fall nicht fehlschlägt, wendet Serendipity daher diese zweite Methode an.

Wenn auch die zweite Methode fehlschlägt, versucht Serendipity ein Pingback an das entsprechende Blog zu senden. Ein Pingback stellt sozusagen ein funktionsreduziertes Trackback dar, denn ein Pingback enthält keinen Artikelauszug oder die URL zu Ihrem Artikel. Es dient daher lediglich dem fremden Blog als Hinweis, dass sich jemand auf dessen Artikel bezieht.

Wenn ein Trackback oder Pingback erfolgreich gesendet wurde, kann es dennoch eine Weile dauern, bis es im fremden Blog angezeigt wird. Häufig müssen Trackbacks vom Autor erst freigeschaltet werden, bevor sie erscheinen.

Beim Speichern eines Artikels kann Serendipity folgende Meldungen ausgeben:

Überprüfe *http://...* auf mögliche Trackbacks ...

[1] Konkret werden die RDF-Metatags `<rdf:Description trackback:ping="...">` und `<rdf:Description dc:identifier="...">` gesucht.

wird für jede im Artikel angegebene URL angezeigt.

Sende Trackback zu URI *http://…* …
wird ausgegeben, wenn eine Trackback-fähige URL gefunden wurde. Wenn nach dieser Ausgabe nichts weiter erscheint, wird Ihr Server durch eine Firewall blockiert und verhindert die Ausführung des Trackbacks. In diesem Fall müssen Sie Trackbacks deaktivieren.

Trackback erfolgreich!
erscheint, wenn ein Trackback erfolgreich ausgeführt wurde.

URI enthielt keine Daten
erscheint, wenn die angegebene URL nicht aufgerufen werden konnte, da sie keine Daten zurücklieferte. Dies kann passieren, wenn ein Webserver nicht erreichbar ist oder die URL ungültig war.

Trackback gescheitert: Keine Trackback-URI gefunden
erscheint, wenn die URL die notwendigen Meta-Daten zum Senden eines Trackbacks nicht enthielt. Diese Meldung erscheint auch, wenn Sie beispielsweise Links zu Amazon in Ihrem Blog-Artikel verwenden, da Amazon keine Trackbacks unterstützt. Daher muss diese Meldung nicht zwangsläufig auf einen Fehler hinweisen, sondern informiert Sie lediglich, dass kein Trackback gesendet wurde.

Trackback gescheitert: Die automatisch erkannte Trackback-URI gleicht

nicht der angegebenen URI weist Sie darauf hin, dass Serendipity zwar die benötigten Meta-Daten auf einem Blog finden konnte, diese aber eine falsche URL ausweisen. Wenn Sie z. B. ein Trackback zu `http://example.com/serendipity/archives/1-Eintrag.` senden wollen, aber das Blog eigentlich unter `http://www.example.com/serendipity/archiv` `trag.htm` läuft (beachten Sie das `www.` in der URL), kann ein solcher Fehler entstehen. Serendipity benötigt immer exakt die URL, die das fremde Blog als *offiziell* angibt.

Kein Trackback: Konnte Verbindung zu … auf Port … nicht herstellen
erscheint, wenn Ihr Webserver aufgrund von Verbindungsproblemen die angegebene URL nicht erreichen konnte. Probieren Sie es eventuell später einmal, oder prüfen Sie, ob Sie die korrekte URL eingetragen haben.

Sending pingback to URI …
erscheint, wenn eine Pingback-fähige URL gefunden wurde und Serendipity nun einen Pingback sendet.

Pingback successful
erscheint bei einem erfolgreichen Pingback.

Pingback failed: No pingback-URI found
erscheint, wenn keine Pingback-fähige URL gefunden wurde.

Damit ein Trackback erfolgreich gesendet werden kann, muss Ihr Webserver ausgehende HTTP-Verbindungen zulassen. Wird dies durch eine Firewall verhindert, können Sie eventuell einen Proxy-Server dazu einsetzen, den Sie über das Plugin *Trackbacks kontrollieren* (siehe Folgeabschnitt) festlegen können.

Da Serendipity jede angegebene URL (bis auf bekannte Dateitypen wie `.exe, .pdf, .avi` etc.) aufruft, kann der Vorgang beim Speichern einige Zeit in Anspruch nehmen.

Serendipity führt Trackbacks nur bei Artikeln aus, die Sie im Status *Veröffentlichen* speichern. Wenn Sie eine neue URL zu einem bestehenden, veröffentlichten Artikel hinzufügen, wird beim Speichern ebenfalls ein Trackback gesendet. Zu bereits enthaltenen URLs wird nach der Veröffentlichung kein weiterer Trackback geschickt. Wenn es Probleme beim ersten Versand gab, können Sie ein neues Trackback senden, indem Sie Ihren Artikel zuerst wieder als *Entwurf* speichern und danach nochmals als *Veröffentlichung*.

8.1.2 Plugin: Trackbacks kontrollieren

Wie beschrieben, sendet Serendipity automatisch immer Trackbacks an alle erkannten URLs im Beitrag. Es kann jedoch auch vorkommen, dass Sie einmal bewusst kein Trackback setzen möchten oder dass Sie ein Trackback zu einer speziellen URL senden wollen, die Sie im Artikeltext selbst aber nicht angeben möchten. Bei beiden Fällen hilft das Ereignis-Plugin *Trackbacks kontrollieren* (siehe Seite 283).

Sobald Sie dieses Plugin installiert haben, können Sie bei einem Eintrag im Abschnitt *Erweiterte Optionen* einstellen, ob Trackbacks gesendet werden sollen und an welche zusätzlichen URLs Sie ein Trackback schicken wollen.

Die zusätzlichen URLs können dabei direkt Trackback-URLs entsprechen, wie sie auf einem fremden Blog beworben/angezeigt werden. Üblicherweise geben Sie in Ihrem Artikel bereits immer einen Link an, mit dem der Besucher zu der normalen Seite gelangt. Wenn dieser Link jedoch keinen Trackback-Code enthält, müssten Sie eine gültige Trackback-URL manuell eintragen.

Dazu ein Beispiel: In einem WordPress-Blog finden Sie unter der URL `http://wp.com/?p=1337` einen Artikel, auf den Sie sich beziehen möchten. Dieses Blog enthält jedoch keine Meta-Daten, und auf der Seite selbst sehen Sie, dass der Trackback-Link `http://wp.com/p1337/trackback` lauten müsste. Unbekümmert davon erstellen Sie einen Blog-Eintrag mit folgendem Inhalt:

```
Mein Bruder schreibt <a href="http://wp.com/?p=1337">seine
Doktorarbeit</a>.
```

Wenn Sie nun den Eintrag speichern, meldet Serendipity, dass es unter der eingegebenen URL keine Trackback-Daten entdecken kann. Daher kann kein Trackback gesendet werden. Sie erinnern sich an die Trackback-URL und ändern Ihren Artikel so ab, dass Sie den Link `http://wp.com/p1337/trackback` einsetzen. Doch auch hier meldet Serendipity, dass es keine Meta-Daten finden kann.

Hier kommt nun das Plugin ins Spiel. In dem Eingabefeld des Abschnitts **Erweiterte Optionen** müssen Sie die Trackback-URL `http://wp.com/p1337/trackback` eintragen, und im Artikel selbst können Sie wieder die normale URL `http://wp.com/?p=1337` verwenden. Nun erkennt Serendipity, dass Sie ein Trackback zu der eingetragenen URL erzwingen wollen, und führt es aus.

Weitere Anwendungsmöglichkeiten schlagen Sie bitte auf der genannten Seite bei der Plugin-Beschreibung nach.

Sie können Trackbacks zentral deaktivieren, indem Sie die Variable `$serendipity['noautodiscovery']` in der Serendipity-Konfigurationsdatei `serendipity_config_local.inc.php` auf `true` setzen (siehe Seite 175).

8.1.3 Plugin: Einträge ankündigen (XML-RPC Pings)

Zusätzlich zu Trackbacks und Pingbacks gibt es ein Angebot an Webservices, die regelmäßig Weblogs indizieren und automatisch miteinander verketten. Der bekannteste Dienst ist Technorati[2], das unter anderem auch die *Wichtigkeit* von Blogs auswerten kann, indem es Blogs hochrangig einstuft, die besonders häufig zitiert werden.

Derartige Webservices durchsuchen üblicherweise (wie Google) regelmäßig Ihr Blog. Damit ein Service aber sofort bei neuen Einträgen von Ihnen tätig werden kann, müssen Sie dafür sorgen, dass der Webservice kontaktiert wird, sobald Sie einen Eintrag speichern.

Das Plugin *Einträge ankündigen* (siehe Seite 275) kann dies für Sie erledigen. Mittels einer XML-RPC-Schnittstelle kann es einen sogenannten Ping senden, der dem Webservice mitteilt: „*Guck mal, hier gibt's was Neues*“.

8.1.4 Trackbacks und Pingbacks empfangen

Selbstverständlich kann Serendipity nicht nur Trackbacks versenden, sondern auch empfangen.

Dazu werden die notwendigen Meta-Daten für die Trackback-URL in jedem Blog-Eintrag automatisch eingebettet. Die Trackback-URL ist die Schnittstelle, die ein anderes Blog aufrufen muss, um dorthin einen Verweis zu einem eigenen Artikel zu übermitteln.

Eine Trackback-URL sieht aus wie `http://www.example.com/serendipity/comment.php?type=` Diese URL ist nicht dazu gedacht, von einem normalen Browser aufgerufen zu werden, daher erhalten Sie in diesem Fall eine Fehlermeldung. Ein fremdes Blog jedoch übermittelt an diese URL alle notwendigen Variablen, die Serendipity benötigt, um Ihrem Artikel ein Trackback hinzuzufügen.

Damit Sie Trackbacks zu einem Artikel empfangen können, müssen Sie bei dem jeweiligen Artikel die Ankreuzbox **Kommentare für diesen Eintrag zulassen** aktiviert haben (dies ist

[2] `http://www.technorati.com/`

standardmäßig der Fall).

Auch das Anti-Spam-Plugin von Serendipity (siehe Seite 239) weist einige Optionen auf, die Sie dafür einsetzen können, Trackbacks zu erlauben, automatisch zu moderieren oder auch gänzlich abzuweisen (siehe Folgekapitel).

Empfangene Trackbacks werden auf der Detailseite des Blog-Eintrags oberhalb der Kommentare angezeigt. Sie können die Darstellung von Trackbacks (und auch Kommentaren) gänzlich verhindern, indem Sie den jeweiligen Bereich in Ihrem Template löschen (siehe Seite 489).

Neuere Templates wie **Bulletproof** ermöglichen es sogar, über Template-Optionen festzulegen, ob Sie einen Kommentar- und Trackback-Bereich anzeigen oder verstecken wollen.

8.1.5 Trackback- und Kommentar-Spam

Weblogs laden Besucher dazu ein, Kommentare zu hinterlassen oder sich mit Trackbacks auf Einträge zu beziehen. Erst diese Vernetzung und Interaktivität ist es, was Weblogs so populär und beliebt macht und von einfachen Newstickern oder Content-Management-Systemen unterscheidet.

Die große Popularität und hohe Einstufung in Suchmaschinen bezahlen Blogger jedoch mit einem ebenso hohen Preis: SPAM.

Spam ist der Sammelbegriff für jegliche Form der unerwünschten Werbung. Kommentarspam bewirbt Webseiten oder andere Produkte und wird von Spammern meist automatisiert verschickt. Spammern ist es dabei egal, ob ihre Werbung auf einem Blog irgendwann gelöscht oder moderiert wird, denn wenn mehrere hunderttausend Blogs mit ihren Botschaften bestückt werden und davon 90% der Kommentare gelöscht werden, bleiben dennoch tausende Kommentare übrig.

Spammer nennen ihre Webseiten in Blog-Kommentaren und erhöhen so ihren Suchmaschinenrang durch den Rang des Blogs, auf dem sie kommentieren. Durch eine Kooperation von Blog-Entwicklern und Suchmaschinenbetreibern wurde ein HTML-Attribut *nofollow* eingeführt. Dieses Attribut wurde automatisch an die Verweise aller Webseiten in Blog-Kommentaren angehängt und führt dazu, dass Suchmaschinen den Rang einer solchen Webseite nicht mit dem Blog verknüpfen.

Leider gilt dies aber nicht nur für Spammer, sondern für alle Kommentatoren. Und so stellte sich heraus, dass diese Maßnahme eher den aufrichtigen Bloggern und dem generellen Vernetzungsvorteil von Blogs schadet. Da nicht jedes Blogsystem die *nofollow*-Attribute einsetzt, bleiben immer noch Blogs online, in denen Kommentarspam suchmaschinenrelevante Wirkung erzielt. Die Spammer senden also nach wie vor automatisierten Spam, denn auch hier zählt die Masse. Wenn von 100.000 Kommentaren nur 100 ohne *nofollow*-Attribut durchkommen, ist das für einen Spammer immer noch eine gute Quote. Denn Kommentarspam kostet einzig Ressourcen durch benötigte Bandbreite und Computer, die den Spam verschicken.

Dabei verschicken Spammer inzwischen ihre Nachrichten (genauso wie bei E-Mail-Spam) nicht mehr über eigene Leitungen und Computer, sondern sie nutzen sogenannte *Bot-Netze*. Ein Bot-Netz ist ein Netzwerk aus viren- und trojanerverseuchten Computern ganz normaler Internetbenutzer. Viren und Trojaner kann sich ein normaler Computerbenutzer mittlerweile sehr einfach über Browser einfangen, und aufgrund der Unkenntnis vieler Benutzer fällt ihnen das gar nicht auf. Bei den Milliarden von Internet-Nutzern ist es leider nach wie vor nicht selbstverständlich, aktuelle Virenscanner einzusetzen –, und so kommen jeden Tag tausende neuer Benutzer in den riesigen Bot-Netzen hinzu.

Diese Bot-Netze können von den Spammern ferngesteuert werden, indem sie fremde Computer dazu anweisen, Kommentar- oder E-Mail-Spam zu versenden. So können die Spammer Bandbreiten bündeln und mit tausenden gleichzeitigen Zugriffen einen Webserver sogar völlig überlasten.

Die Betreiber der angegriffenen Server haben wenig Chancen, dagegen vorzugehen: Sie können die Benutzer nicht blocken, da sie für den Server wie ein ganz normaler Besucher einer Webseite aussehen. Die Bot-Netze verfügen über riesige Kontingente an voneinander unabhängigen PCs mit ganz unterschiedlichen IP-Adressen und können so nicht lokalisiert werden.

Man kann daher das Spam-Problem nur an seiner Wurzel packen, indem man auf Spam-Angebote wie Viagra, Sex-Offerten und Geldanleihen verzichtet. Dies ist jedoch ein soziales, menschliches Problem, das wohl niemals aus der Welt zu schaffen ist. Mittelfristig kann es daher nur helfen, wenn Sie selbst auf die Sicherheit ihres Computers achten, Anti-Viren-Software einsetzen und Ihre Bekannten und Freunde über die Gefahren des Internets aufklären. Auch die Gesetzgebung ist gefragt, mit Spam-Verboten und möglicherweise der Einführung von *Internet-Führerscheinen* dem Missbrauch Einhalt zu gebieten. Immerhin darf man auch ein Auto erst führen, wenn man es bedienen kann – auch mit dem Internet ist heutzutage viel Schaden anzurichten.

Bei immer größer werdender Bandbreite der privaten Internetanschlüsse dank DSL und VSDL werden solche Bot-Netze für Server-Betreiber zu einer immer größeren Gefahr, da sie ein Blog vollständig lahmlegen können. Dem können Sie nur begrenzt vorbeugen (siehe Seite 463).

Nach so viel Pessimismus sei aber auch erwähnt, dass Sie dem Treiben zumindest in gewissem Maße Einhalt gebieten können. Zentrale Anlaufstelle hierfür ist das Serendipity-Ereignis-Plugin *Anti-Spam*. Wenn Sie dieses installieren, können Sie durch zahlreiche Mechanismen gezielt gegen Kommentar- und Trackbackspam vorgehen (siehe Seite 239).

Wenn Spammer Ihr Blog torpedieren, kann es jedoch sein, dass selbst das Anti-Spam-Plugin zu viele Ressourcen bindet. Denn bei jedem Kommentar oder Trackback muss sich der Serendipity-Softwarekern initialisieren, eine Datenbankverbindung aufbauen und die Anti-Spam-Checks durchführen sowie möglicherweise weitere Server (wie Akismet) kontaktieren. All dies benötigt Zeit, die der Webserver aber womöglich nicht hat, wenn er mehrere hundert gleichzeitige Zugriffe regeln muss.

In so einem Fall sind Sie sehr auf die Hilfe Ihres Server-Providers angewiesen. Dieser kann

auf Netzwerk-Ebene und auf Server-Ebene einige Vorkehrungen treffen, damit ein Webserver nicht völlig unerreichbar wird. Er kann die Anzahl der maximalen Zugriffe limitieren und etwaige IP-Adressbereiche bei Überbenutzung sperren. Eine Blockade durch Bot-Netze nennt man *DDoS (Distributed Denial of Service)*. Während viele Provider lediglich mit Abschaltung ihres Servers reagieren, gibt es auch ernsthaft bemühte Provider, die hier gemeinsam mit Ihnen an einer Lösung arbeiten. Provider wie *Manitu*[3], *TiggersWelt*[4] und *All-Inkl.com*[5] sind bekannt dafür, hier individuelle Lösungsmöglichkeiten anzubieten.

Temporär können Sie Kommentare und Trackbacks vollständig unterbinden, indem Sie die Datei `comment.php` von Serendipity löschen oder umbenennen. Über diese Datei werden Kommentare und Trackbacks angenommen.

Meist werden Trackbacks aufgrund ihrer Automatisierbarkeit von Spammern bevorzugt. Zwar können Sie Trackbacks auch über das Anti-Spam-Plugin vollständig deaktivieren, aber dies bindet wie erwähnt Systemressourcen. Daher können Sie alternativ die Serendipity-Datei `comment.php` mit einem Editor bearbeiten und dort in der ersten Zeile Folgendes eingeben:

```
<?php if ($_GET['type'] == 'trackback') die('No Service.'); ?>
```

Diese Zeile weist Serendipity an, bei einem Trackback-Versuch die Ausführung des Scripts sofort zu beenden und Ressourcen zu sparen.

8.2 Wartung der Datenbank und der Dateien

Serendipity speichert bis auf die Dateien der Mediendatenbank alle Informationen in seiner Datenbank. Einträge speichert Serendipity beispielsweise in der Datenbanktabelle `serendipity_entries`, Kategorien in der Tabelle `serendipity_category` (eine vollständige Liste der Tabellen finden Sie ab Seite 623).

Diese Tabellen werden im Laufe der Betriebszeit Ihres Blogs immer voller und größer. Möglicherweise hat Ihr Provider Ihnen jedoch eine bestimmte Größenbeschränkung der Datenbanktabelle auferlegt, und irgendwann könnte es zu Fehlermeldungen kommen, die Sie darauf hinweisen, dass Sie Ihr Limit überschritten haben.

Daher kann es von Zeit zu Zeit sinnvoll sein, dass Sie ihre Datenbank warten und nicht länger benötigte Einträge löschen. Die Datenbank speichert Ihre Dateien meist in (für Sie) versteckten Verzeichnissen, daher können Sie nur mittels spezieller Datenbank-Wartungsprogramme auf die Inhalte direkt zugreifen. Abhängig von der bei Ihnen eingesetzten Datenbank haben Sie die Wahl zwischen verschiedenen Programmen.

Für MySQL ist das Web-basierte Programm `phpMyAdmin`[6] sehr verbreitet. Viele Provider

[3] `http://www.manitu.net/`
[4] `http://www.tiggerswelt.net/`
[5] `http://www.all-inkl.com/`
[6] `http://www.phpmyadmin.net`

bieten eine vorinstallierte Version für Sie an. Die Installation von phpMyAdmin ist zwar mit etwas Konfigurationsaufwand verbunden, aber auf der Homepage gut dokumentiert.

Für PostgreSQL steht `phpPgAdmin`[7] zur Verfügung, das an phpMyAdmin angelehnt ist.

Beim Einsatz von SQLite ist die Wahl etwas schwieriger, da dieser Datenbanktyp weniger stark verbreitet ist. `phpSQLiteAdmin`[8] kann Ihnen möglicherweise gute Dienste leisten.

Abgesehen von den Web-basierten Verwaltungsprogrammen gibt es auch eigenständige Windows- oder Linux-Anwendungen, die Sie selbstverständlich auch einsetzen können.

Die Wahl und Installation der Verwaltungssoftware ist unabhängig von Serendipity und kann daher hier nicht eingehender erklärt werden. Die folgenden Beispiele beziehen sich allesamt auf das verbreitete `phpMyAdmin`.

In Ihrem Verwaltungsprogramm können Sie alle Datenbanktabellen von Serendipity einsehen. Wir gehen im Folgenden davon aus, dass Sie nur Tabellen warten möchten, in denen weniger relevante Daten (wie moderierte Kommentare, Logfiles, Referrer) gespeichert werden. Wenn Sie Blog-Artikel oder Plugins löschen wollen, können Sie dies über die normale Serendipity-Oberfläche tun. Wartbare Daten finden Sie in den im Folgenden erwähnten Tabellen.

8.2.1 serendipity_spamblocklog, serendipity_karmalog, serendipity_cronjoblog

Einige Plugins erstellen Datenbanktabellen, um darin ihre Protokolle (*Logs*) abzulegen. Andere Plugins nutzen dafür auch normale Dateien (im Verzeichnis `templates_c`), jedoch hat die Speicherung in der Datenbank den Vorteil, dass Sie diese mit Programmen wie phpMyAdmin komfortabel ansehen und auch beliebig filtern können.

In der Tabelle `serendipity_spamblocklog` werden beispielsweise alle Vorgänge gespeichert, bei denen das *Anti-Spam*-Plugin Kommentare oder Trackbacks moderiert/abgewiesen hat. Da diese Protokolle in einer Tabelle gespeichert werden, können Sie diese leicht nach Datum oder Typ filtern. Die Beschreibung der jeweiligen Datenbankfeldnamen ist auf Seite 636 dokumentiert.

Das Plugin löscht in der Tabelle selbständig keine Einträge. Wenn Sie Ihr Blog also zwei Jahre lang betreiben, sammeln sich in dieser Tabelle *alle* abgewiesene Kommentare über diesen Zeitraum hinweg. In den seltensten Fällen benötigen Sie diese Datensätze jedoch noch. Daher können Sie regelmäßig die Einträge dieser Protokoll-Tabelle über einen SQL-Dump (siehe Abschnitt 8.3 ab Seite 450) auf Ihrer Festplatte sichern und dann die Datenbanktabelle leeren (SQL-Anweisung: `TRUNCATE TABLE serendipity_spamblocklog`). Sie können aber auch mit einer einfachen SQL-Anweisung alle Einträge löschen, die älter als vier Wochen sind:

[7] `http://phppgadmin.sourceforge.net`
[8] `http://www.phpguru.org/static/phpSQLiteAdmin.html`

```
DELETE FROM serendipity_spamblocklog WHERE timestamp <
(UNIX_TIMESTAMP(NOW()) - (86400*30))
```

Der Zeitstempel (*timestamp*) eines Eintrags wird in Sekunden gespeichert, und die SQL-Abfrage sucht alle Einträge, die älter als der heutige Zeitstempel minus 30 Tage (86400 Sekunden entsprechen einem Tag) sind.

Diese SQL-Abfrage können Sie auch automatisch einmal im Monat über einen Cronjob ausführen, wenn Sie über einen SSH-Zugang zu Ihrem Blog verfügen. Denkbar wäre auch, dass Sie folgende PHP-Datei im Serendipity-Verzeichnis unter dem Namen `log_recycle.php` speichern:

```
<?php
include 'serendipity_config.inc.php';
serendipity_db_query("DELETE FROM serendipity_spamblocklog WHERE
timestamp < (UNIX_TIMESTAMP(NOW()) - (86400*30))");
)
?>
```

Diese Datei können Sie (via cronjob oder manuell) regelmäßig dann unter `http://www.example.com/serendipity/l` aufrufen
und mit weiteren Aufrufen der `serendipity_db_query()`-Funktionen für andere Datenbanktabellen ergänzen.

Dasselbe gilt für die Datenbanktabelle `serendipity_karmalog`. Diese wird vom Plugin *Karma (Abstimmung über die Einträge* – siehe Seite 271) erzeugt und enthält eine Liste aller abgegebenen Abstimmungen pro Eintrag und pro Person. Die Datenbanktabelle `serendipity_cronjoblog` wird durch das Plugin *Cronjob* (siehe Seite 288) angelegt und enthält eine Liste der zuletzt automatisch aufgerufenen Plugins.

Alle genannten Tabellen können über die Zeit sehr groß werden. Vergessen Sie daher nicht, diese in regelmäßigen Abständen zu prüfen. Natürlich können Sie jedes Plugin auch so konfigurieren, dass die Protokolle erst gar nicht gespeichert werden, wenn Sie diese nicht benötigen.

8.2.2 serendipity_spamblocklog_htaccess

Einen Sonderfall der Protokollierung des *Anti-Spam*-Plugins stellt die Tabelle `serendipity_spamblocklog_htaccess` dar. Wenn in diesem Plugin die automatische Aktualisierung der Datei `.htaccess` aktiviert wurde, werden in dieser Tabelle alle zu blockenden IP-Adressen aufgeführt. Diese Datenbasis wird dann für die Erzeugung der `.htaccess`-Datei verwendet.

Dabei benutzt das Plugin jedoch standardmäßig nur die Daten der letzten zwei Tage. Die älteren Daten bleiben weiterhin vorhanden und müssen von Ihnen bei Bedarf gelöscht werden, da das Plugin sonst davon ausgeht, dass Sie die alten Daten zu statistischen (oder anderen externen) Zwecken möglicherweise beibehalten möchten.

8.2.3 serendipity_visitors

Das *Statistik*-Plugin von Serendipity kann jeden Besucher zählen und eine Aufrufstatistik einbinden. Dafür muss jeder Besucher in der Datenbanktabelle `serendipity_visitors` erfasst werden.

Bei größeren Blogs kann diese Datenbank enorme Größen annehmen, da ältere Einträge nicht automatisch gelöscht werden. Da bei aktivierter Besucherstatistik bei jedem Seitenaufruf ein schreibender Zugriff auf diese Tabelle erfolgt, kann bei einer großen Tabelle die Performance des gesamten Frontends spürbar sinken.

Daher sollten Sie besonders diese Tabelle im Auge behalten. Selten ist es von Interesse, die Statistik des letzten Jahres anzusehen – löschen Sie also in diesem Fall alte Datensätze auch regelmäßig (wie eingangs beschrieben).

8.2.4 serendipity_exits

Wenn Sie das Plugin *Textformatierung: Externe Links zählen* aktiviert haben, wird jeder Klick auf einen Link, den Sie in einem Blog-Artikel eingebunden haben, in der Datenbanktabelle `serendipity_exits` gespeichert. Anhand dieser Daten kann später eine Statistik erstellt werden, welche Links Ihre Besucher am häufigsten geklickt haben.

Abweichend von den bisher aufgeführten Datenbanktabellen werden die Daten hier pro Tag gespeichert und nicht mit einem Zeitstempel versehen. Wenn Sie also selektiv alte Datensätze löschen möchten, müssen Sie dies mit folgender SQL-Anweisung tun:

```
DELETE FROM serendipity_exits WHERE UNIX_TIMESTAMP(day) <
(UNIX_TIMESTAMP(NOW()) - (86400*30))
```

8.2.5 serendipity_referrers

Wenn Sie in der Serendipity-Konfiguration das **Referrer-Tracking** aktiviert haben, kann Ihr Blog bei jedem Besucher auswerten, von welcher Seite er zu Ihnen gelangt ist. Diese Angaben können in der Statistik später angezeigt werden, um aufzuschlüsseln, woher die meisten Besucher kommen.

Ähnlich wie die Tabelle `serendipity_exits` enthält die Tabelle `serendipity_referrers` eine Liste von Links, die tagesabhängig sind. Sie können daher folgenden SQL-Code verwenden, um die Referrer, die älter als einen Monat sind, zu löschen:

```
DELETE FROM serendipity_referrers WHERE UNIX_TIMESTAMP(day) <
(UNIX_TIMESTAMP(NOW()) - (86400*30))
```

8.2.6 Dateisystem

Im Dateisystem speichert Serendipity vor allem die Dateien der Mediendatenbank (im Verzeichnis `uploads`) und temporäre Dateien (kompilierte Templates, Caches).

Während Sie die Mediendatenbank über das normale Serendipity-Backend verwalten können (und sollten), müssen Sie sich um temporäre Dateien eigentlich selten kümmern. Diese werden im Verzeichnis `templates_c` gespeichert.

In diesem Verzeichnis finden Sie Dateien nach dem Muster `defaultˆ%%C8ˆC86ˆC86%%entries.tpl.php`. Dies sind die Smarty-Templatedateien, die von Serendipity und Smarty in gültigen PHP-Code umgewandelt werden (siehe Smarty-Beschreibung ab Seite 480). Der erste Teil des Dateinamens steht dabei für den Templatenamen (`default`), dann folgt eine zufällige Folge von Sonderzeichen. Das Ende des Dateinamens enthält den ursprünglichen Template-Dateinamen.

Für jede Smarty-Templatedatei finden Sie eine entsprechende temporäre, kompilierte Datei. Diese Dateien können Sie immer gefahrlos löschen, denn wenn sie nicht vorhanden sind, legt sie Smarty selbständig wieder an.

Wenn Sie in einer Testphase einmal unterschiedliche Templates aktiviert haben, werden Sie für diese ebenfalls kompilierte Dateien auffinden. Diese können Sie dann löschen, um etwas Speicherplatz zu sparen.

Weiterhin können sich im Verzeichnis `templates_c` auch beliebige temporäre Dateien von Plugins befinden. Beispielsweise legt das Spartacus-Plugin (siehe Seite 151) die XML-Dateien mit Paketinformationen hier unter dem Namen `package_event_de.xml` oder `package_sidebar_de.xml` an. Grundsätzlich gilt: Sie können *alle* Dateien in diesem Verzeichnis stets gefahrlos löschen. Jedes Plugin ist so entwickelt, dass temporäre Dateien automatisch neu (und mit frischem Inhalt) erstellt werden, wenn sie nicht mehr vorhanden sind.

Gerade wenn Sie Plugins deinstallieren, bleiben die alten Cache-Dateien in diesem Verzeichnis (je nach Plugin) weiterhin vorhanden. Daher lohnt es sich, von Zeit zu Zeit dieses Verzeichnis zu überprüfen. Sie können in einem FTP-Programm beispielsweise alle Dateien nach Erstellungsdatum sortieren und dann die Dateien löschen, die offensichtlich seit längerer Zeit nicht mehr aktualisiert wurden. Meist sind Cache-Dateien jedoch so klein, dass sich der Wartungsaufwand nur in seltenen Fällen lohnt.

Im Verzeichnis `archives` befinden sich wider Erwarten *keine* Blog-Artikel. Dieses Verzeichnis war in älteren Serendipity-Versionen für statische HTML-Dateien vorgesehen, wurde jedoch nie richtig verwendet. Mittlerweile wird dieses Verzeichnis von einigen Plugins verwendet, die dauerhaft Daten speichern möchten. Beispielsweise speichert das Plugin zur Konvertierung eines Blog-Artikels in eine PDF-Datei seine Dateien dort.

8.3 Datenbankprobleme, Fehlermeldungen

Wenn Serendipity sich nicht mit der Datenbank verbinden kann oder ein Defekt der Datenbank vorliegt, gibt Serendipity die Fehlermeldung direkt über die PHP-Fehlerausgabe aus.

Je nach Konfiguration Ihres Webservers werden derartige Fehler entweder direkt in Ihrem Browser beim Besuchen des Blogs dargestellt oder in einer Log-Datei auf dem Server gespeichert.[9] Fragen Sie Ihren Provider nach dem Speicherort eines solchen Logfiles, wenn Sie darüber nicht Bescheid wissen.

Die häufigsten Datenbankfehlermeldungen (bei Benutzung von MySQL) sind:

`Warning: mysql_connect(): Access denied for user`

`'...'@'...' (using password: YES) serendipity error:`

`unable to connect to database - exiting.` Diese Fehlermeldung erhalten Sie, wenn Serendipity sich nicht erfolgreich mit der Datenbank verbinden konnte. Häufig passiert dies, wenn Sie Ihre Zugangsdaten bzw. Passwörter zum Datenbankserver geändert haben. Überprüfen Sie die Serendipity-Datei `serendipity_config_local.inc.php`, dort müssen die aktuell gültigen Zugangsdaten eingetragen werden.

Das Problem kann jedoch auch dann auftreten, wenn eine Datenbankverbindung über Sockets oder den TCP/IP-Port fehlschlägt. Hier muss der Datenbankadministrator die Zugangsparameter und Client-Bibliotheken auf Korrektheit und Übereinstimmung prüfen.

`Table '...' is crashed and should be repaired`
: Die Fehlermeldung erscheint, wenn der Datenbankserver abgestürzt ist und Tabellen nicht wieder korrekt einlesen konnte. Wenn dies häufiger passiert, lässt es Rückschlüsse auf einen Hardwaredefekt zu, und Sie sollten Ihren Provider kontaktieren. In vielen Fällen können Sie die betreffende Tabelle über den SQL-Befehl `REPAIR TABLE` *`tabellenname`* mittels des Wartungsprogramms reparieren. phpMyAdmin verfügt dazu über eigene Operationen, mit denen Sie menügesteuert Tabellen reparieren können.

  Serendipity kann erst wieder auf die Tabelle zugreifen, wenn sie repariert oder wiederhergestellt wurde.

`Warning: mysql_query(): Unable to save result set in ...`
: Diese Fehlermeldung kann erscheinen, wenn ein Übertragungsproblem vom Datenbankserver zum Webserver vorliegt oder die Tabelle nicht korrekt ausgelesen werden

[9] Die Protokollierung erfolgt mittels der `php.ini`-Einstellung `error_reporting`, `display_errors` und `error_log`, die Sie möglicherweise auch über eine `.htaccess`-Datei anpassen können. Wenn PHP als CGI ausgeführt wird, können zusätzliche Fehlermeldungen auch im `ErrorLog` der Webserver-Konfiguration erscheinen.

konnte. In manchen Fällen hilft es auch hier, alle beteiligten Datenbanktabellen zu reparieren. Zudem kann die Fehlermeldung auch auftreten, wenn die Datenbank-Client-Bibliothek für PHP nicht korrekt (oder mit einer falschen Version kompiliert) wurde. In diesem Fall sollten Sie Ihren Provider kontaktieren.

`Can't open file: 'serendipity_entries.MYI'`

`(errno: 145, 138)` Diese gefürchtete Fehlermeldung tritt auf, wenn die Datei, in der eine Datenbanktabelle gespeichert wurde, nicht mehr gelesen werden kann. Entweder passiert dies, weil die Datei nicht mehr vorhanden oder weil sie defekt ist. In manchen Fällen kann ein `REPAIR TABLE` SQL-Befehl weiterhelfen. Wenn die Datei jedoch physisch nicht mehr vorhanden ist, muss sie vom Provider (oder Ihnen) anhand eines Backups erneut eingespielt werden.

Solche Fehler können nur durch einen Absturz des Datenbankservers oder andere Einwirkung von außen geschehen.

`Lost connection to MySQL server during query Too many`

`connections` MySQL erlaubt lediglich eine bestimmte Anzahl von parallelen Datenbankverbindungen. Wenn Ihr Blog sehr viele gleichzeitige Besucher hat oder gerade eine Spamwelle über Sie hereinbricht, kann dieses Limit schnell überschritten werden.

Die genannte Fehlermeldung sagt Ihnen, dass die Datenbank keine weiteren Zugriffe mehr zulässt. Sie müssen daher ggf. den Datenbankserver für mehr Verbindungen konfigurieren (lassen) oder alternativ mit Ihrem Provider über mögliche Fehlerursachen sprechen.

Weiterhin kann es auch zu PHP-Fehlermeldungen wie diesen kommen[10]:

Keine Schreibrechte für Verzeichnis `templates_c`, INCLUDE_ERROR
: Serendipity benötigt stets volle Schreibrechte zu dem temporären Verzeichnis `templates_c`. Wenn PHP bzw. der Webserver dort keine Dateien mehr speichern kann, schlägt Serendipity Alarm und kann das Blog nicht mehr aufrufen. Sobald Sie die Schreibrechte korrigieren (`chmod 777 templates_c` via SSH oder FTP), sollte das Blog wieder erscheinen.

Warning: `session_start(): ...`
: Serendipity benötigt Zugriff auf das PHP-Session-Modul. Wenn dies nicht (oder fehlerhaft) konfiguriert ist, können Schreibzugriffe auf die Sessiondaten fehlschlagen. Der Administrator des Webservers sollte daher die konfigurierten Verzeichnisse[11] prüfen und nachsehen, ob das PHP `session`-Modul installiert ist.

[10] Weitere Hinweise zur korrekten Einrichtung können Sie im Kapitel 2.2.6 ab Seite 62 nachschlagen.
[11] php.ini: `session.save_path`

Fatal error: `Maximum execution time of 30 seconds exceeded`
: Wenn Serendipity für einen Seitenaufruf mehr Zeit als das konfigurierte Limit[12] benötigt, bricht PHP mit einer Fehlermeldung ab. Bitten Sie entweder den Server-Administrator um ein großzügigeres Limit oder entfernen Sie etwaige Performance-intensive Plugins.

Fatal error: `Allowed memory size of ...bytes exhausted`
: Ähnlich wie das zeitliche Limit für einen Seitenaufruf gibt PHP vor, wie viel RAM ein einzelner Seitenaufruf belegen darf[13]. Meist sind dies 8MB RAM, mit denen Serendipity in der üblichen Konfiguration auskommen sollte. Für einige Operationen, und je nach installierten Plugins, müssen Sie das Speicherlimit jedoch erhöhen lassen.

Weiße/leere Seite: `Internal Server Error`
: Wenn Sie beim Aufruf einer Webseite lediglich den Fehler `Internal Server Error` oder eine leere, weiße Seite erhalten, kann dies zahlreiche Problemursachen haben. Die Fehlermeldung bedeutet hier lediglich: *„Etwas lief schief“*. Die Fehlerursache erfahren Sie nur über die Server-Logdateien. Häufige Probleme hier sind überschrittener Speicherbedarf, fehlende Schreibrechte zu einem Verzeichnis, Datenbank-Verbindungsprobleme.

  Aber auch spezielle Anweisungen in der `.htaccess`-Datei können diese Meldung hervorrufen. Benennen Sie Ihre `.htaccess`-Datei daher vorübergehend um, um herauszufinden, ob ohne diese Datei alles funktioniert. Ist dies der Fall, sollten Sie die Datei bearbeiten und alle enthaltenen Zeilen mit einer Raute (#) deaktivieren. Aktivieren Sie nun jede Zeile für sich und prüfen Sie nach jeder einzelnen aktivierten Zeile, ob Ihr Blog noch aufgerufen werden kann. Meist finden Sie so heraus, dass eine Zeile wie `ErrorDocument 404 index.php` oder `php_value register_globals Off` in der `.htaccess`-Datei bei Ihnen nicht erlaubt ist. Weitere Informationen zur `.htaccess` lesen Sie auf Seite 34.

Fatal error: `Cannot instantiate non-existent class: smarty`
: Diese PHP-Fehlermeldung weist darauf hin, dass Serendipity eine benötigte Datei nicht finden oder lesen konnte. Prüfen Sie, ob Sie tatsächlich alle Dateien des Serendipity-Pakets vollständig hochgeladen haben und ob das Verzeichnis `templates_c` lesbar ist.

Fatal error: `Cannot use string offset as an array in ...`
: Diese Fehlermeldung erscheint meist, wenn eine vorhergehende Datenbankabfrage fehlgeschlagen ist oder eine Datei des Serendipity-Pakets defekt ist.

Warning: `Invalid argument supplied for foreach() in ...`
: Diese Fehlermeldung ist nur ein Hinweis und führt nicht zwangsläufig zu einem Abbruch. Meist entsteht eine Meldung wie diese, wenn ein Plugin unsauber programmiert

[12] php.ini: `max_execution_time`
[13] php.ini: `memory_limit`

wurde und einen Fall nicht berücksichtigt, in dem keine Daten gefunden werden. Klassisches Beispiel hierfür ist die Auswahl einer zugehörigen Kategorie für einen Blog-Artikel. Serendipity ging in älteren Versionen immer davon aus, dass Sie mindestens eine Kategorie angelegt haben, und stellte diese Fehlermeldung dar, wenn die Liste aller Kategorien (die ja leer wäre) durchgegangen werden muss.

Fatal error: `Call to a member function on a non-object in ...`
Diese Fehlermeldung gibt an, dass Serendipity eine Funktion ausführen wollte, die nicht geladen werden konnte. Dies kann passieren, wenn Sie Plugin-Dateien manuell gelöscht haben oder diese Dateien nicht mehr lesbar sind.

Fatal error: `Call to undefined function serendipity_db_time`
Wenn Sie nicht alle Dateien des Serendipity-Pakets hochgeladen haben, kann es passieren, dass Serendipity eine Fehlermeldung darstellt, wenn eine fehlende Funktionalität aufgerufen werden soll. Stellen Sie daher sicher, dass alle Dateien des Pakets vorhanden sind.

Fatal error: `Call to undefined method`

`serendipity_smarty_emulator::...` Serendipity liefert zwei besondere Templates aus: `default-php` und `default-xml` (siehe Seite 692). Diese sind nur für Entwickler gedacht. Wenn Sie als Unkundiger diese Templates in Ihrem Blog aktivieren, kann dies zu der genannten Fehlermeldung führen. Sie sollten daher ein anderes Template (wie das Serendipity 3.0 Standard-Template) auswählen.

Wenn Sie als Entwickler dieses Template bewusst gewählt haben, sagt Ihnen diese Fehlermeldung, dass Sie eine Template-Datei aufrufen wollten, die noch nicht in das notwendige XML- oder PHP-Format überführt wurde.

`open_basedir restriction in effect SAFE MODE Restriction`

`in effect` Wenn Serendipity in einem PHP-Umfeld mit `open_basedir` und `Safe Mode`-Einschränkungen läuft, kann es nicht frei auf alle Dateien und Verzeichnisse zugreifen. Bei einer falschen Konfiguration dieser PHP-Variablen (siehe Seite 32) führt dies dazu, dass Serendipity zentrale Dateien nicht einbinden kann und daher der Aufruf fehlschlägt.

Lassen Sie in diesem Fall vom Server-Provider die PHP-Konfiguration prüfen, so dass die in der Fehlermeldung angegebene Datei gelesen/geschrieben werden kann.

Fatal error: `Cannot redeclare class pear in .../PEAR.php`
Einige Serendipity-Funktionen setzen die zentrale PEAR-Bibliothek ein. Diese benötigten Bibliotheken liefert Serendipity mit. Manchmal sind die PEAR-Bibliotheken jedoch auf dem Webserver bereits zentral installiert.

Wenn nun ein Serendipity-Plugin auf einen Fehler stößt, versucht die PEAR-Bibliothek eine Fehlermeldung auszugeben. Dafür wird die zentrale Datei `PEAR.php` eingebunden, die Serendipity aber üblicherweise bereits eingebunden hat. Daher kommt es zu einer Fehlermeldung, die angibt, dass eine Klasse doppelt geladen werden müsste.

Grundsätzlich sollte diese Fehlermeldung bei keinem offiziellen Plugin mehr vorkommen. Wenn Sie darauf stoßen, muss man dafür sorgen, dass die jeweilige PEAR-Bibliothek so eingebunden wird, dass sie die `PEAR.php` nicht zentral einbindet, sondern erst prüft, ob die Klasse bereits definiert ist. Bitte melden Sie eine solche Fehlermeldung im Serendipity-Forum mit einer detaillierten Angabe, wann und wo diese Fehlermeldung auftritt.

Weitere Fehlermeldungen sind in der Dokumentation Ihres Datenbanksystems[14] aufgeführt. Im Zweifelsfall sollten Sie bei hier nicht aufgeführten Fehlermeldungen entweder Ihren Provider oder das Serendipity-Forum kontaktieren.

8.4 Backups erstellen

Computer können immer einmal abstürzen, sei es durch Stromausfälle, Hardwaredefekte oder einfache Softwarefehler. Daher ist es wichtig, Datenverlusten vorzubeugen.

Sie sollten regelmäßig sogenannte *Backups* (Sicherheitskopien) Ihres Blogs erstellen. Dieser Hinweis gilt natürlich für sämtliche Ihrer persönlichen Daten. Auch private Fotos, Dokumente und andere Dateien, die Sie auf Ihrem Computer sichern, sollten von Zeit zu Zeit auf dauerhafte Medien wie DVDs oder CDs übertragen werden. Denn auch eine fein säuberlich auf Ihrer Festplatte abgelegte Datei könnte einmal durch Defekte beschädigt werden.

Da Serendipity nicht wie eine Bild- oder Dokumentdatei auf Ihrem Computer liegt, können Sie es nicht ohne Weiteres sichern, sondern benötigen Zusatzsoftware.

Serendipity besteht aus zwei Komponenten: der Datenbank und den Dateien im Dateisystem. Die Dateien können Sie wie gewohnt per FTP oder Ähnlichem auf Ihren Computer herunterladen. Besonders wichtig sind dabei folgende Dateien und Verzeichnisse:

`.htaccess`, `serendipity_config_local.inc.php`
: In diesen beiden Dateien speichert Serendipity zentrale Konfigurationswerte. Ohne diese beiden Dateien kann später beim Wiederherstellen eines Backups keine Verbindung zur Datenbank hergestellt werden!

  Serendipity schützt diese beiden Dateien über unbefugte Zugriffe. Dies kann jedoch abhängig von der Einrichtung Ihres Servers bedeuten, dass Sie auch selbst per FTP keine Zugriffsrechte besitzen. In diesem Fall können Sie sich mittels eines kleinen PHP-Scripts namens `fixperm.php` (oder auch mit der Hilfe Ihres Providers) jedoch

[14] Für MySQL z. B. `http://dev.mysql.com/doc/refman/5.0/en/error-messages-server.html`

leicht die fehlenden Rechte zuschieben. Dieses Script ist auf Seite 64 näher beschrieben.

Verzeichnis `uploads`
: In diesem Verzeichnis speichert Serendipity alle von Ihnen hochgeladenen Mediendaten. Wenn Sie diese nicht sowieso separat auf Ihrem Computer sichern, sollten Sie die Dateien dringend für ein Backup herunterladen.

Verzeichnis `templates`, `plugins`
: Wenn Sie eigene Plugins und Templates installiert oder angepasst haben, sollten Sie die beiden Verzeichnisse `templates` und `plugins` ebenfalls lokal sichern.

Alle anderen Dateien und Verzeichnisse können Sie zwar trotzdem regelmäßig sichern, aber diese sind bei einem Datenverlust nicht kritisch, da Sie problemlos stattdessen einfach die aktuellste Serendipity-Version herunterladen können. In diesem Dateiarchiv sind sämtliche benötigten Dateien enthalten.

Wenn Sie abseits von Serendipity noch weitere Dateien angepasst oder unabhängige Dateien hochgeladen haben, müssen Sie diese natürlich eigenverantwortlich in Ihre Datensicherungsstrategie einschließen.

Als zweite (und wichtigere) Komponente der Datenhaltung verwendet Serendipity die Datenbank. Diese müssen Sie mit einem separaten Programm wie `phpMyAdmin`[15] (siehe auch Seite 441) durchführen.

Exemplarisch folgt eine Beschreibung, wie man mittels phpMyAdmin ein Backup der Datenbank erstellen kann:

phpMyAdmin aufrufen
: Öffnen Sie phpMyAdmin, indem Sie in Ihrem Browser die entsprechende URL aufrufen, unter der Sie das Programm installiert haben. Je nach Konfiguration von phpMyAdmin müssen Sie nun Ihre Zugangsdaten eingeben, mit denen Sie auf die Serendipity-Datenbank zugreifen können.

Datenbank auswählen
: Wählen Sie auf der linken Seite die Datenbank, in der die Serendipity-Tabellen installiert wurden.

Reiter Exportieren auswählen
: Auf der rechten Seite sehen Sie nun eine Liste aller verfügbaren Datenbanktabellen. Klicken Sie im oberen Bereich auf den Reiter **Exportieren**.

Tabellen auswählen
: Auf der linken Seite des nun geöffneten Bereichs sehen Sie ein Mehrfach-Auswahlfeld mit allen Tabellen der Datenbank. Klicken Sie auf `Alle auswählen` oder sorgen Sie durch manuelle Auswahl dafür, dass alle Serendipity-Tabellen (erkennbar anhand des Präfixes `serendipity_`) ausgewählt sind.

[15] `http://www.phpmyadmin.net`

Export-Format festlegen
phpMyAdmin ermöglicht den Export in zahlreiche Formate. Wählen Sie die Option **SQL** am Ende der Liste aus.

Optionen setzen
Auf der rechten Seite des Export-Dialogs sehen Sie zahlreiche Optionen. Diese sind auch abhängig davon, welche Version von phpMyAdmin Sie einsetzen. Wenn Sie eine der folgenden Optionen nicht finden, ignorieren Sie diese einfach.

Individuelle Kommentare für den Kopfbereich
kann leer gelassen werden.

Export in einer Transaktion zusammenfassen
Eine Transaktion bedeutet, dass mehrere SQL-Befehle miteinander gruppiert werden. Wenn Sie eine derartige Transaktion später importieren und dabei ein Import-Fehler auftritt, werden alle Vorgänge wieder rückgängig gemacht. Damit Sie also später einmal bei denkbaren Defekten der SQL-Datei trotzdem die funktionstüchtigen Datensätze importieren können, sollten Sie diese Option besser *nicht* aktivieren.

Fremdschlüsselüberprüfung deaktivieren
Diese Option hat für Serendipity keine Bedeutung.

SQL-Kompatibilitätsmodus
In diesem Ausklappfeld können Sie festlegen, in welchem *SQL-Format* das Backup geschrieben wird. Abhängig von der eingesetzten Version Ihres MySQL-Servers kann die Struktur der Backup-Datei unterschiedlich ausfallen. Wichtig ist dabei vor allem, dass Sie ein Format wählen, welches Sie später wieder problemlos importieren können. Wenn Sie MySQL ab mindestens Version 4.0 einsetzen, sollten Sie hier die Option **MYSQL40** wählen, andernfalls sollten Sie **MYSQL323** auswählen.

Struktur
Diese Checkbox muss aktiviert sein, damit phpMyAdmin die strukturellen Informationen der Datenbanktabellen exportiert.

Füge DROP TABLE/DROP VIEW hinzu
Deaktivieren Sie diese Option, damit die SQL-Befehle nicht möglicherweise existierende Tabellen unbeabsichtigt überschreiben.

Füge IF NOT EXISTS hinzu
Aktivieren Sie diese Option, damit die SQL-Befehle nur Tabellen erstellen, die noch nicht bestehen.

AUTO_INCREMENT-Wert hinzufügen
Damit Sie beim Einfügen neuer Datensätze in eine wiederhergestellte Tabelle später keine abweichenden, automatisch vergebenen ID-Werte produzieren, sollten Sie diese Option aktivieren.

Tabellen- und Feldnamen in einfachen Anführungszeichen
Aktivieren Sie diese Option, damit die genannten Namen innerhalb von SQL-Befehlen speziell mittels Anführungszeichen abgegrenzt werden können. Dies erhöht die Kompatibilität mit anderen SQL-Serversystemen, ist aber nicht zwingend erforderlich.

Füge CREATE PROCEDURE/FUNCTION hinzu
Deaktivieren Sie diese Option, da Serendipity diese Funktionen nicht verwendet.

In Kommentarbereich einbeziehen
Deaktivieren Sie alle drei Unteroptionen (**Erzeugungszeiten . . . , Tabellenverknüpfungen, MIME-Typ**), da sie von Serendipity nicht benötigt werden.

Daten
Aktivieren Sie diese Option unbedingt, damit die Inhalte Ihrer Datenbanktabellen gesichert werden können.

Vollständige INSERTs
Aktivieren Sie diese Option für bestmögliche SQL-Kompatibilität.

Erweiterte INSERTs
Aktivieren Sie diese Option, um eine möglichst kleine und optimierte Backup-Datei zu erhalten. Wenn Sie jedoch höhere Kompatibilität mit anderen SQL-Systemen benötigen und lieber auf *Nummer sicher* gehen wollen (indem Sie redundante Daten speichern und eine größere Datei erhalten), dann können Sie diese Option auch deaktivieren.

Maximale Länge der erstellten Abfrage
Der Zahlenwert beschränkt die Anzahl der Tabellenzeilen. Der Standardwert von 50.000 ist meistens sehr großzügig bemessen. Um sicher zu gehen, sollten Sie den Wert jedoch auf etwas wie `999999999` setzen, damit keine Datensätze ignoriert werden.

Verzögerten INSERT-Befehl verwenden
Deaktivieren Sie diese Option, um später Import- und Kompatibilitätsprobleme zu vermeiden.

Fehlerübergehenden INSERT-Befehl verwenden
Deaktivieren Sie diese Option ebenfalls, um etwaige Probleme beim Import zu vermeiden.

Use hexadecimal for BLOB
Aktivieren Sie diese Option, um etwaige in Serendipity-Tabellen enthaltene Binärdaten korrekt in das Backup einzubeziehen.

Exporttyp
Setzen Sie dieses Auswahlfeld unbedingt auf **INSERT**, da ansonsten die Datensätze für den SQL-Export später nicht korrekt einfügbar wären.

Senden
Aktivieren Sie diese Checkbox, damit die Backup-Datei von Ihrem Browser her-

untergeladen werden kann. Alternativ können Sie phpMyAdmin auch dazu anweisen, diese Datei direkt auf Ihrem Webserver zu speichern. In den Optionen darunter können Sie einen beliebigen Dateinamen eingeben, die Zeichensatzkodierung des Backups sollten Sie üblicherweise auf **utf-8** einstellen. Wenn Ihr Blog bzw. Ihre Datenbank einen anderen Zeichensatz verwendet, müssen Sie diesen stattdessen auswählen. Die Backup-Datei sollten Sie sicherheitshalber *nicht* komprimieren, um Problemen beim Import vorzubeugen.

SQL-Dump speichern

Als *Dump* bezeichnet man den Vorgang, alle Inhalte einer Datenbanktabelle in SQL-Befehle umzuwandeln. Diese SQL-Befehle enthalten die nötigen Anweisungen, um Ihre Rohdaten später wieder in eine neue, leere Datenbanktabelle einzufügen.

Stellen Sie sich vor, Sie erstellen einen *Dump* Ihrer Blog-Einträge. Diese sind in der Tabelle `serendipity_entries` gespeichert und enthalten solche Datenspalten wie *Titel*, *Text* und *Uhrzeit*. Sie besitzen zwei Einträge, einen mit dem Titel *Henne* und einen anderen mit dem Titel *Ei*. Wenn Sie diese Einträge nun im SQL-Dump-Format speichern, erhalten Sie eine Datei mit folgenden SQL-Befehlen:

```
INSERT INTO ‘serendpity_entries‘ (title, body, timestamp)
     VALUES (’Henne’, ’...’, ’2007-11-11 11:11’);

INSERT INTO ‘serendpity_entries‘ (title, body, timestamp)
     VALUES (’Ei’, ’...’, ’2007-11-11 11:11’);
```

Diese SQL-Befehle enthalten ausreichend Informationen, um den Inhalt einer Datenbanktabelle später auf einem beliebigen SQL-fähigen Datenbankserver wiederherzustellen.

Klicken Sie in phpMyAdmin nun auf den Button **OK**, um die Erstellung der Backup-Datei (also des SQL-Dump) auszuführen. Ihr Browser sollte Ihnen daraufhin anbieten, diese Datei auf Ihrer Festplatte zu speichern.

Wenn Sie diesen Vorgang einige Male ausgeführt haben, werden Sie feststellen, dass Sie die Erstellung eines vollständigen Backups nicht länger als fünf Minuten beschäftigt.

Legen Sie sich einen Zeitplan zurecht, mit dem Sie regelmäßig ein Backup erstellen, oder sprechen Sie alternativ mit Ihrem Web-Provider darüber, ob und wie er automatische Backups für Sie erstellen kann. Zwar gibt es ein Serendipity-Backup-Plugin, dies funktioniert jedoch nur mit relativ kleinen Datenbanken.

8.5 Backups einspielen

Ein Backup ist natürlich nur dann etwas wert, wenn Sie es im Ernstfall auch verwenden können. Nachdem Sie Ihr erstes Backup erstellt haben, versuchen Sie also ruhig einmal den Ernstfall zu simulieren.

Laden Sie dazu ihr Serendipity-Datei-Backup wie gewohnt per FTP auf Ihren Webserver in ein *eigenständiges* Verzeichnis hoch, da Sie ja die bestehende Installation möglichst nicht zerstören wollen. In diesem Beispiel verwenden wir dazu ein Verzeichnis `/var/www/example.com/serendipity-backup/`, das Sie später unter `http://www.example.com/serendipity-backup/` aufrufen.

Nach dem FTP-Upload müssen Sie sicherstellen, dass alle ursprünglichen Datei-Zugriffsrechte wie vorher gesetzt sind. Das heißt, das Verzeichnis `templates_c` muss existieren und schreibbar sein, und die Datei `serendipity_config_local.inc.php` muss für den Webserver sowohl schreib- als auch lesbar sein. Details zu den Leserechten finden Sie auf Seite 53.

Nachdem Sie die Dateien hochgeladen haben, müssen Sie Ihr Datenbank-Backup (den *Dump*) wiederherstellen. Grundbedingung dazu ist, dass Sie wieder über einen Zugang zu einem Datenbanktool wie phpMyAdmin verfügen und eine Datenbank zur Verfügung gestellt bekommen haben, in der Sie die gesicherten Daten wiederherstellen können.

Unseren gesicherten SQL-Dump können wir nicht einfach ohne Weiteres in die Datenbank importieren, da derzeit ja auch noch Ihre echten Blog-Tabellen darin bestehen.

Der einfachste Weg, um herauszufinden, ob Ihr gesicherter SQL-Dump fehlerfrei ist, bestünde darin, eine unabhängige zweite Datenbank zu erstellen. Nur wenige Provider erlauben Ihnen aber die Erstellung von mehr als einer Datenbank.

Daher ist die zweiteinfachste Möglichkeit, den Tabellenpräfix Ihrer Tabellen innerhalb des SQL-Dumps zu verändern. Standardmäßig ist dieser Präfix auf `serendipity_` eingestellt und sorgt dafür, dass Ihre Datenbanktabellen benannt sind wie `serendipity_entries`, `serendipity_authors` und so weiter. Benutzen Sie also nun einen Texteditor, um damit Ihre SQL-Dump-Datei zu öffnen. Diese Datei enthält einfache Textdaten, daher können Sie mit jedem beliebigen Editor arbeiten.

Verwenden Sie nun die **Suchen und Ersetzen**-Funktion Ihres Editors, und ersetzen Sie alle Vorkommen von `CREATE TABLE IF NOT EXISTS serendipity_` in `CREATE TABLE IF NOT EXISTS serendipity2_`. Damit sorgen Sie dafür, dass die SQL-Befehle der Dump-Datei später eine eigene Tabelle anlegen. Beachten Sie, dass Sie diesen Vorgang nur deshalb ausführen, weil Ihre Wiederherstellung des Backups keinen Ernstfall darstellt. Wenn Ihre Datenbanktabellen wirklich verloren wären, müssten Sie diesen Aufwand nicht treiben, da nichts überschrieben werden könnte. Als Zweites müssen Sie auch alle `INSERT INTO serendipity_`-Statements ändern in `INSERT INTO serendipity2_`.

Nun können Sie die veränderte SQL-Dump-Datei mit einer Oberfläche wie phpMyAdmin exemplarisch wie folgt hochladen:

phpMyAdmin aufrufen
: Öffnen Sie phpMyAdmin, indem Sie in Ihrem Browser die entsprechende URL aufrufen, unter der Sie das Programm installiert haben.

Datenbank auswählen
: Wählen Sie auf der linken Seite die Datenbank, in der Sie die Serendipity-Tabellen

wiederherstellen möchten.

Reiter Importieren auswählen
Auf der rechten Seite sehen Sie nun eine Oberfläche, in der Sie beliebige SQL-Dumps einlesen können.

SQL-Dump auswählen
Klicken Sie auf den Menüpunkt **Durchsuchen...** (oder **Browse...**), um ein Auswahlmenü zu öffnen. Wählen Sie nun die SQL-Dumpdatei von Ihrer Festplatte aus.

Optionen setzen
Im Ausklappfeld **Zeichencodierung der Datei** müssen Sie den Zeichensatz des SQL-Dumps auswählen, im obigen Beispiel haben wir UTF-8 verwendet. Die Option **Abbruch wenn die maximale Scriptlaufzeit erreicht wird** können Sie aktivieren, um bei besonders großen SQL-Dump-Dateien zu erreichen, dass phpMyAdmin diese so weit wie möglich einlesen kann und nicht aufgrund von Ressourcenlimits abbricht. Sollte Ihre Datei zu groß sein, müssen Sie den Import in mehreren Schritten vornehmen und jeweils die **Anzahl der am Anfang zu überspringenden Einträge (Abfragen)** in das Eingabefeld eintragen. Beispiel: Wenn Sie 100 Datensätze importieren wollen, aber nur jeweils 25 importieren können, müssen Sie beim ersten Mal 0 Datensätze überspringen, beim nächsten Mal 25 usw.

Als Dateiformat der zu importierenden Datei wählen Sie **SQL** aus. Als **SQL-Kompatibilitätsmodus** müssen Sie die Option auswählen, mit der Sie die Backup-Datei erstellt haben (üblicherweise **MYSQL40**). Wenn Sie unsicher sind, wählen Sie **NONE** aus.

SQL-Dump einlesen
Klicken Sie nun auf den **OK**-Knopf, damit phpMyAdmin Ihre Datenbanktabellen anhand der Backup-Datei erstellen kann. Der Import-Vorgang sollte nun alle SQL-Befehle abarbeiten, was einige Zeit beanspruchen kann. Danach sollten alle Tabellen erstellt worden sein.

Um Serendipity nun in Ihrem Backup-Verzeichnis aufrufen zu können, müssen Sie noch eine Kleinigkeit anpassen. Da Sie beim Test-Import den Tabellen-Präfix von `serendipity_` in `serendipity2_` verändert haben, müssen Sie zusätzlich in einem solchen Fall auch die Datei `serendipity_config_local.inc.php` anpassen und dort die Variable `$serendipity['dbPrefix']` auf den Wert `'serendipity2_'` ändern.

Wenn dies erfolgt ist, müssten Sie in der Lage sein, Ihr Blog-Backup über `http://www.example.com/seren` `php` aufzurufen. Da Ihr Blog-Backup im Pfad `/` lief, müssten Sie nun eigentlich überall in der Administrationsoberfläche Pfadanpassungen vornehmen (siehe auch Abschnitt 8.8 ab Seite 462), damit das Blog später perfekt läuft. Für unsere Zwecke zum Testen eines Backups ist dies jedoch nicht notwendig, da Sie nun wissen, wie Sie ein Backup im Ernstfall wiederherstellen.

8.6 Statistiken

Als Blog-Besitzer wird Ihnen im Laufe der Zeit womöglich eine Sache sehr am Herzen liegen: Wer Ihre Besucher sind, was und wie viel sie lesen und wie regelmäßig sie vorbeischauen.

Für diese Zwecke gibt es eine große Auswahl an Statistik-Möglichkeiten. Einige bietet Serendipity (mittels Plugins oder internen Funktionen) an, andere werden durch externe Anbieter ergänzt.

Sie können auf folgende Möglichkeiten der Statistik zurückgreifen:

Externe Links
: Über das Ereignis-Plugin *Externe Links verfolgen* (siehe Seite 258) können Sie dafür sorgen, dass alle Links, die Sie in Artikeln einbinden, speziell markiert werden. Dadurch können Sie als Autor feststellen, wie oft Ihre Besucher auf die von Ihnen vorgestellten Webseiten klicken. Diese Links nennt man *Exits*, da sie auf fremde Webseiten verweisen und Ihre Besucher von Ihrer eigenen Webseite wegleiten.

  Die so gesammelten statistischen Daten können Sie entweder über das Seitenleisten-Plugin *Top Exits* anzeigen oder über das Ereignis-Plugin *Statistiken* im Backend einsehen.

Verweisende Seiten, Referrer, Google-Verweise
: Wenn ein Besucher über eine Suchmaschine wie Google oder durch Links fremder Seiten auf Ihr Blog gelangt, nennt man diese verweisende Seite einen *Referrer*. Die Browser Ihrer Besucher übermitteln diese Seite standardmäßig beim ersten Zugriff.

  Aktivieren Sie das Serendipity-Referrer-Tracking (siehe Seite 168), können Sie die am häufigsten verweisenden Seiten über das Plugin *Top Referrer* in der Seitenleiste anzeigen oder über das *Statistik-Plugin* einsehen.

  Leider können Referrer von Spammern leicht gefälscht werden, die somit in Ihre Top-Listen gelangen und die Statistiken verfälschen können.

  Zusätzlich übermitteln Suchmaschinen wie Google ein wichtiges Detail in ihrer Referrer-URL: den Suchbegriff, mit dem ein Besucher zu Ihnen gefunden hat. Das Ereignis-Plugin *Hebe Suchwörter hervor* kann dafür sorgen, dass die so gewonnenen Suchwörter automatisch hervorgehoben werden, wenn sie in Ihren Einträgen vorkommen. Dies hilft Ihren Besuchern, sich auf Ihrer Seite zu orientieren, da ihr Suchwort direkt ins Auge fällt.

  Die so herausgefilterten Suchwörter können Sie zusätzlich auch über das Seitenleisten-Plugin *Letzte Google-Suche* darstellen.

Zugriffs-Logfiles
: Ein sehr hilfreiches Mittel bei der Analyse Ihrer Zugriffs- und Besucherzahlen stellen die Logfiles Ihres Webservers dar. Meistens stellt Ihr Provider Ihnen ein Verzeichnis zur Verfügung, in dem Sie die täglichen oder monatlichen Logfiles einsehen können. Dort

wird jeder Besucher Ihrer Seite mit seiner IP-Adresse und der von ihm aufgerufenen URL aufgeführt.

Es gibt viele Logfile-Analyseprogramme, mit denen Sie komfortabel auch grafische Auswertungen durchführen können. Diese Programme teilen sich in zwei Sparten: Serverseitige Tools wie `awStats`[16], `modlogan`[17] oder `webalizer`[18] sowie Anwendungsprogramme wie `WebSuxess`[19], `WebTrends` und weitere.

Externe Dienstleister

Als sehr hilfreich haben sich auch externe Dienstleister zur Statistik-Verfolgung erwiesen, allen voran `Google Analytics`[20] und freie Dienste wie `blogcounter.de`, `blogstats.de` oder weitere.

Alle arbeiten nach einem einfachen Grundprinzip: Der Blog-Betreiber bindet auf seiner Seite einen JavaScript-Aufruf in seinen Quellcode ein. Bei Serendipity erfolgt das einfach über ein HTML-Klotz-Plugin (siehe Seite 216).

Dieses JavaScript wird vom Browser des Besuchers aufgerufen und übergibt die Daten seines Besuchs (aufgerufene URL, Datum, Referrer, Bildschirmauflösung, verwendeter Browser, IP-Adresse) an den Dienstleister weiter.

Google Analytics hebt sich in den detaillierten Verknüpfungs- und Analyze-Optionen sehr hervor und bietet alles, was das Statistiker-Herz begehrt. Dennoch müssen Sie berücksichtigen, dass eine JavaScript-basierte Statistik nicht alle Aufrufe zählen kann: Wenn ein Besucher JavaScript deaktiviert hat oder Ihr JavaScript wie Werbung filtert, gelangen seine Daten nicht an den Dienstleister.

Individuelle Tracking-Software

Wer Statistiken nicht bei externen Dienstleistern führen will, kann auch spezielle PHP-Tracking-Software auf seinem eigenen Server installieren. Dazu stehen Tools wie `Mint`[21] oder `phpOpenTracker`[22] zur Verfügung.

Derartige Tools lassen sich üblicherweise problemlos in HTML-Kopf-Klötze (siehe Plugin-Beschreibung auf Seite 286) einbinden. Sollte zur Einbindung eigener PHP-Code erforderlich sein, können Sie diesen leicht in die Smarty-Templatedateien einbinden.

Der Vorteil von lokaler Analysesoftware ist, dass sie zuverlässiger alle Besucher verfolgen und in Echtzeit verarbeiten kann. Auch datenschutzrechtlich ist es besser, Ihre Besucher-Bestandsdaten nicht an externe Dienstleister weiterzureichen. Als Nachteil

[16] `http://awstats.sourceforge.net/`
[17] `http://modlogan.org/`
[18] `http://www.mrunix.net/webalizer/`
[19] `http://www.exody.net/ger/products/websuxess/websuxess.cfm?kat2=p1`
[20] `http://www.google.com/analytics/`
[21] `http://haveamint.com/`
[22] `http://www.phpopentracker.de/`

ist jedoch anzusehen, dass die Einbindung komplexer ist als die für externe Dienstleister mittels einfachem JavaScript.

8.7 Plugins und Serendipity aktualisieren

Da die Entwicklung rund um Serendipity natürlich nicht stillsteht, werden häufiger neue Versionen von Plugins oder des Kernsystems herausgegeben. Über neue Versionen können Sie über das offizielle Serendipity-Blog unter `http://blog.s9y.org/` auf dem Laufenden bleiben.

Plugins lassen sich sehr leicht aktualisieren, indem Sie einfach die neue Plugin-Version herunterladen und die alten Dateien auf dem Server überschreiben. Wenn Sie einmal selbständig Änderungen an Plugin-Dateien vorgenommen haben, müssen Sie natürlich darauf achten, dass Sie Ihre eigenen Änderungen danach auch wieder in das neue Plugin übernehmen.

Wenn Sie das Spartacus-Plugin benutzen, können Sie neue Plugin-Versionen auch einfach über das Internet aktualisieren. Dazu bietet die Plugin-Verwaltung im Backend Ihres Blogs einen einfachen Button namens **Neue Versionen von Seitenleisten/Ereignis-Plugins** an. Wenn Sie auf diesen Button klicken, wird Ihnen eine Auflistung aller installierten Plugins gezeigt, für die eine neue Version im Internet vorliegt. Ein Klick auf den danebenstehenden Button eines Plugins kann das Update dann automatisch ausführen. Sollten keine neuen Versionen vorliegen, erhalten Sie eine leere Liste.

Das Serendipity-Kernsystem lässt sich grundsätzlich ähnlich einfach aktualisieren. Dazu müssen Sie folgende Schritte durchführen:

Backup
: Um sicher zu gehen, dass bei der Aktualisierung nichts schiefläuft, sollten Sie vorher ein Backup der Dateien und der Datenbank durchführen (siehe Seite 450).

Blog sperren
: Wenn Sie Ihr Blog aktualisieren wollen, werden Sie in den folgenden Schritten einige Dateien ersetzen und administrative Aktionen ausführen. Grundsätzlich sollte dieser Vorgang sehr schnell gehen, und Ihre Besucher sollten davon nichts mitkriegen. Dennoch besteht die Gefahr, dass, wenn Sie Ihr Blog aktualisieren, während gerade Besucher darin unterwegs sind, diese mit Fehlermeldungen konfrontiert werden könnten.

  Daher ist es sicherer, wenn Sie Ihr Blog temporär deaktivieren. Wenn Ihnen dies zu kompliziert erscheint, überspringen Sie diesen Schritt einfach. Ansonsten erreichen Sie einen Schutz des Blogs am einfachsten, indem Sie die Datei `.htaccess` in Ihrem Serendipity-Verzeichnis am Anfang um folgende Zeilen ergänzen:

```
AuthType Basic
AuthName ''Serendipity Upgrade''
AuthUserFile /var/www/example.com/serendipity/.htpasswd
require valid-user
```

  Diese Zeilen werden vom Apache-Webserver ausgewertet und sorgen dafür, dass Ihr Blog nur noch dann aufgerufen werden kann, wenn Sie ein spezielles Passwort eingeben. Dieses Passwort müssen Sie noch in der Datei `.htpasswd` eintragen. Diese Datei speichert Passwörter in einem speziell verschlüsselten Format, daher müssen Sie Ihr gewünschtes Schutz-Passwort vorher mit einem Tool wie auf `http://www.advancehost.com/htpa` verschlüsseln. Andere derartige Verschlüsselungshilfen finden Sie, wenn Sie in einer Suchmaschine nach `.htpasswd` suchen.

  Wenn Sie beispielsweise einen Benutzernamen „s9y“ mit dem Passwort „s9y“ erstellen wollen, müsste die `.htpasswd`-Datei wie folgt aussehen:

```
s9y:s9QXoc9dcFOT2
```

  Wenn Sie nun Ihr Blog über `http://www.example.com` aufrufen, sollte Ihr Browser Sie nach Ihrem Passwort fragen.

  Anstelle eines Passwortschutzes können Sie Ihr Blog auch über die `.htaccess`-Regeln wie `Deny/Allow`[23] sperren.

Datenbank- und Schreibrechte prüfen
: Damit Serendipity sich menügesteuert aktualisieren kann, müssen Schreibrechte zu der Konfigurationsdatei `serendipity_config_local.inc.php` vergeben werden (siehe Seite 586).

  Zur Aktualisierung der Datenbank muss Ihr Datenbank-Account über Rechte verfügen, um `ALTER`, `INDEX` und `CREATE` SQL-Abfragen ausführen zu können. Diese Rechte

[23] Siehe `http://de.selfhtml.org/servercgi/server/htaccess.htm`

werden auch zur Installation von Serendipity benötigt, daher sollten sie gewöhnlich kein Problem darstellen.

Neue Version hochladen

Laden Sie die neue Serendipity-Version von der Webseite herunter,[24] ähnlich wie Sie das für die Installation bereits getan haben. Entpacken Sie das Archiv nun und laden Sie *alle* Dateien wie üblich auf Ihren Webserver hoch (via FTP oder SSH).

Stellen Sie unbedingt sicher, dass alle Dateien korrekt hochgeladen worden sind, Ihr FTP-Programm darf keine Fehlermeldungen für den Upload darstellen. Da ein Serendipity-Paket aus vielen kleinen Dateien besteht, müssen Sie etwas Geduld beim Dateitransfer haben oder alternativ (falls vorhanden) die Threading-Fähigkeit (parallele Uploads) Ihres FTP-Programms aktivieren.

Achten Sie auch darauf, dass Sie die Standard-Templates von Serendipity einer neuen Version vollständig hochladen. Auch wenn Sie ein eigenes Template benutzen, heißt das nicht, dass Serendipity auf die Dateien im Verzeichnis `/templates/default` verzichten kann.

Nachdem alle Dateien überschrieben worden sind, können Sie nun etwaige eigene Anpassungen an den Serendipity-Dateien, die Sie früher einmal durchgeführt haben, erneut anwenden. Da üblicherweise aber alle Anpassungen über eigene Template-Dateien, Plugins oder Konfigurationsoptionen erreicht werden können, ist eine solche Anpassung von Kerndateien des Serendipity-Systems meist nicht nötig.

Administrationsoberfläche aufrufen

Öffnen Sie nun Ihr Blog, und Sie werden eine Oberfläche sehen, in der Serendipity Ihnen erklärt, dass eine neue Version installiert wurde.

Es folgt eine Auflistung von Operationen, die das Backend automatisch ausführen möchte, sowie eine Liste von notwendigen Datenbank-Anpassungen. Diese Änderungen sollten Sie bestätigen.

Wenn auf der Folgeseite Fehlermeldungen aufgeführt werden, liegt dies üblicherweise daran, dass Ihr Datenbank-Account nicht über die notwendigen Rechte verfügt. Sprechen Sie in diesem Fall mit Ihrem Provider oder wenden Sie sich an das Serendipity-Forum.

Wenn Sie beim Aufrufen Ihres Blogs eine Aktualisierungsoberfläche *nicht* angezeigt bekommen, wurden möglicherweise nicht alle Dateien korrekt hochgeladen, oder das Update wurde bereits ausgeführt. Wurde das Update bereits ausgeführt, sehen Sie dies daran, dass die Versionsnummer Serendipitys in Ihrem Backend bereits die neue Version anzeigt.

Nach der Aktualisierung sollte Ihr Blog wie gewohnt funktionieren. Sämtliche Plugins, Plugin-Einstellungen und Beiträge werden weiterhin wie gewünscht eingestellt, daher müssen Sie sich um nichts weiter kümmern.

[24] `http://www.s9y.org`

Grundsätzlich ist es empfehlenswert, nach der Aktualisierung in Ihrem Browser sämtliche Caches (Zwischenspeicher) und Cookies für Ihr Blog zu löschen, damit Serendipity diese problemlos neu erstellen kann. In einigen Fällen ist andernfalls der Login ins Backend nicht möglich, da Cookies älterer Serendipity-Versionen (besonders beim Umstieg von Version 1.1 auf Version 1.2) sich in ihrem Format geändert haben.

Sollten Sie Ihr Blog über die oben angesprochene `.htaccess`-Methode blockiert haben, müssen Sie diese Blockade auch wieder aufheben.

Im Falle eines Falles...

Sollte dennoch einmal etwas schiefgelaufen sein, können Sie ein Update von Serendipity auch erneut forcieren. Wenn Ihr Datenbank-Account beim Update beispielsweise nicht ausreichend Zugriffsrechte hatte, Sie eine Fehlermeldung angezeigt bekamen und daraufhin den Datenbank-Account korrigiert haben, werden Sie das Update erneut ausführen wollen.

Dazu müssen Sie lediglich die Serendipity-Datei `serendipity_config_local.inc.php` bearbeiten. Wenn Sie keine Schreibrechte zu dieser Datei besitzen, können Sie sich diese über das kleine Script `fixperm.php` (siehe Seite 64) einrichten lassen. Wenn Sie die Konfigurationsdatei `serendipity_config_local.inc.php` in einem Editor öffnen, werden Sie mehrere Variablen sehen. Beim Update ist vor allem der Inhalt der Variable `$serendipity['versionInstalled']` von Interesse. Hierin speichert Serendipity die aktuell konfigurierte Version und kann diese mit der auf dem Server installierten Versionsnummer vergleichen. Weichen die beiden Nummern voneinander ab, wird beim Aufruf des Blogs automatisch die Aktualisierungsoberfläche angezeigt, die Sie zur Durchführung eines Updates auffordert.

Daher können Sie diese Variable einfach nach einem gescheiterten Update zurück auf die Versionsnummer stellen, die Sie vorher installiert hatten.

8.8 Serendipity verschieben

Wenn Sie Ihren Webserver oder Provider wechseln, müssen Sie Ihr Blog *umziehen*. Da Serendipity zahlreiche Konfigurationsoptionen besitzt, die auf den aktuellen Server abgestimmt sind, ist ein Umzug mit etwas Zusatzaufwand verbunden. Auch wenn Sie Serendipity innerhalb desselben Servers in ein anderes Verzeichnis verschieben wollen, können Sie auf Teile dieser Anleitung zurückgreifen!

Der erste Schritt für einen Umzug ist die Erstellung eines Backups aller Dateien und der Datenbank (siehe Seite 450).

Diese Dateien laden Sie auf den neuen Server hoch und achten darauf, dass danach die notwendigen Schreib- und Leserechte wie auf dem alten Server vergeben sind.

Bevor Sie nun den Datenbank-Dump auf den neuen Server hochladen, müssen Sie noch eine Änderung vornehmen. Öffnen Sie diese Datenbank-Dump-Datei mit einem Editor. Su-

chen Sie nun in dieser Datei nach dem alten Server-Verzeichnispfad, in dem Serendipity installiert war (`/var/www/example.com/serendipity`). Dieser Pfadname dürfte an einigen Stellen innerhalb der Datei vorkommen, und Sie müssen nun anstelle des alten Pfades den neuen eingeben. Wenn Sie Serendipity auf dem neuen Server in das Verzeichnis `/var/customers/blog1/serendipity` kopiert haben, müssen Sie genau diesen Pfad in der SQL-Datendatei einsetzen. Andernfalls könnte es später beim Benutzen des Blogs zu fehlenden Verweisen kommen.

Wenn Sie Serendipity auf dem alten Server im Pfad `/serendipity` installiert hatten und diesen Pfad auch auf dem neuen Server einsetzen, müssen Sie an dieser Stelle nichts weiter ändern. Wenn Sie jedoch Serendipity z. B. auf dem neuen Server über `http://www.example.com/neuesblog/` aufrufen wollen, müssen Sie nun einige weitere Änderungen vornehmen. Durchsuchen Sie die Datenbank-Dump-Datei nach der Zeichenkette `/serendipity` und ersetzen Sie diese überall durch `/neuesblog`. So sorgen Sie dafür, dass Links innerhalb Ihres Blogs und Ihrer Blog-Einträge später korrekt aufgerufen werden können.

Laden Sie nun den derart überarbeiteten Datenbank-Dump wie im Backup-Kapitel beschrieben mittels phpMyAdmin oder Ähnlichem auf Ihren neuen Datenbank-Server hoch.

Als Nächstes müssen Sie nun auf dem neuen Server die Datei `serendipity_config_local.inc.php` bearbeiten. Tragen Sie in dieser Datei die Zugangsdaten zu Ihrem neuen Datenbankserver ein und speichern Sie die Datei. Prüfen Sie auch Ihre Datei `.htaccess` und stellen Sie sicher, dass dort eingetragene Pfade (z.B. `/serendipity/index.php`) nach wie vor auf die Verzeichnisstruktur Ihres neuen Servers passen.

Nun sollten Sie in der Lage sein, über Ihre URL auf dem neuen Server die Serendipity-Verwaltungsoberfläche aufzurufen und sich wie gewohnt einzuloggen. Sie sollten nun einmal den Menüpunkt **Konfiguration** aufrufen und dann prüfen, ob alle dort aufgeführten Pfadeinstellungen korrekt sind. Auch kann es empfehlenswert sein, die Konfiguration der einzelnen Plugins zu prüfen, ob dort auch überall die neuen, korrekten Pfade eingetragen sind.

Wenn Sie den Datenbank-Dump bereits eingestellt haben und einige Pfade falsch gesetzt sind, können Sie diese auch nachträglich mit einem Programm wie phpMyAdmin einsehen und korrigieren. Die wichtigsten Serendipity-Tabellen, die Pfade enthalten, heißen `serendipity_config` und `serendipity_plugins`.

8.9 Performance steigern

Ähnlich wie beim Auto-Tuning gibt es bei serverseitiger Software eine schier unendliche Menge an Schrauben, mit denen Sie Ihr Blog *schneller legen* können.

Folgende Liste hilft vielleicht, Ihr Blog in seiner Ausführungsgeschwindigkeit zu beschleunigen:

Spammer ausschließen
: Der häufigste Grund für ein langsames Blog sind meistens Spammer. Wenn Spammer

Ihr Blog gerade besonders stark unter Beschuss nehmen, kann dies sogar dazu führen, dass Serendipity so langsam wird, dass die normalen Besucher keine Inhalte mehr sehen können. Möglicherweise deaktiviert Ihr Provider sogar Ihr Blog vollständig, da es durch die hohe Zugriffsrate der Spammer den Server vollständig auslastet.

Dagegen vorzugehen ist schwierig. Ein guter Web-Provider kann Ihnen dabei stark unter die Arme greifen, da er die Möglichkeit hat, über Server-Tools wie iptables oder Hardware-Firewalls häufige Zugriffe (sogenannte DOS[25] oder DDOS[26]) bereits auf unterster Ebene zu blockieren. Denn wenn ein Spammer erst einmal bis zum Aufruf des Serendipity-Blogs vorgestoßen ist, dann hat er die Ressourcen bereits beansprucht. Daher sind sämtliche Maßnahmen über Serendipity-Plugins (wie das Anti-Spam-Plugin) leider nur eine Möglichkeit zur Eindämmung der Auswirkungen von Spammern, aber können nicht die dadurch verursachten Performance-Probleme einschränken.

Glücklicherweise gibt es auch die Möglichkeit, über `.htaccess`-Dateien einige Zugriffe zu blockieren, sofern Ihr Webserver diese Dateien zulässt. Über die `Deny`-Anweisungen können gezielt bestimmte IP-Adressen vom Blog aus abgewiesen werden. Das Anti-Spam-Plugin kann solche IP-Adressen automatisch in Ihre `.htaccess`-Datei übernehmen, wenn Sie dies in den Optionen des Plugins aktivieren.

Weiterhin können Sie über die Anweisung `BrowserMatch` in einer `.htaccess`-Datei gewisse Roboter von Ihrem Blog ausschließen:

```
# Sperren, wenn Browser-Name mit ‘‘Trackback" beginnt
BrowserMatch ^TrackBack is_spammer
# Sperren, wenn leerer Browser-Name
BrowserMatch ^$ is_spammer

Order Allow,Deny
Allow from all
Deny from env=is_spammer
```

Auch derartige Methoden helfen vermutlich nicht für die Ewigkeit, da sich auch Browser-Namen problemlos manipulieren lassen.

Webserver optimieren

Auf die Austattung und Konfiguration des Webservers haben Sie als Kunde meist wenig Einfluss, daher müssen Sie hinnehmen, was Ihnen geboten wird.

Wenn Sie jedoch Zugriff auf den Server haben, können Sie einige Dinge durchführen, die die Geschwindigkeit aller PHP-Anwendungen optimieren:

- Op-Code-Cache (APC, Zend, ionCube) installieren.
- `register_globals` in der `php.ini` deaktivieren.
- `magic_quotes_gpc` in der `php.ini` deaktivieren.

[25] Denial of Service
[26] Distributed Denial of Service

- PHP als FastCGI kompilieren und einsetzen, möglicherweise den schlanken Lighttpd als Webserver einsetzen.
- In gute Server-Hardware investieren, vor allem schnelle Festplatten und RAM. Merksatz: Besser als RAM ist nur mehr RAM.
- Den Webserver so einrichten, dass er eine sinnvolle Anzahl an Connections/Threads handhaben kann und Speicher- und Ressourcenlimits korrekt setzt.

Datenbank-Server optimieren

Im Endeffekt wichtiger als ein *gut geölter* Webserver ist ein ordentlich konfigurierter Webserver. Über korrekte Tuning-Maßnahmen hierfür sind bereits zahlreiche eigenständige Bücher gefüllt worden.[27]

Gerade MySQL bietet eine fast unüberschaubare Anzahl an Tuning-Schrauben, mit denen man den SQL *QueryCache* einrichtet, Speicherzuweisungen und -annahmen angibt und für sinnvolle Ressourcenlimits und Connection Pools sorgt. Hierüber kann man meist keine pauschale Aussage treffen, da eine gute Konfiguration stets abhängig von dem Einsatzzweck des Servers und der darauf laufenden Applikationen ist.

Serendipity sinnvoll konfigurieren

Als Blog-Betreiber fangen Ihre Tuning-Möglichkeiten meist erst mit Serendipity selbst an.

Grundsätzlich gilt: Je mehr Funktionalität von Serendipity Sie benutzen, desto stärker belastet dies Ihre Ressourcen. Je genauer Sie also gewisse Optionen einschränken, desto schneller wird Serendipity die übrig gebliebenen Features ausführen können. Je weniger Datenbankabfragen pro Seitenaufruf ausgeführt werden müssen, desto schneller die Gesamtgeschwindigkeit.

Die Anzahl der Einträge, die Sie auf den Seiten darstellen, ist ein maßgeblicher Faktor. Standardmäßig werden 15 Artikel pro Seite dargestellt, und jeder einzelne Artikel wird innerhalb einer Schleife an Plugins weitergereicht und später in einer Schleife ausgegeben. Sie sparen daher sowohl HTML-Code als auch Verarbeitungszeit, wenn Sie diese Anzahl geringer halten. Gerade wenn Sie längere Artikel schreiben, kann es zu spürbaren Geschwindigkeitssprüngen führen, wenn Sie das Limit der Artikel beispielsweise auf 7 stellen.

Die Auswertung der Zugriffsrechte für Artikel und Kategorien drückt ebenfalls stark auf die Datenbank-Performance. Wenn Sie gruppenabhängige Leserechte im Frontend des Blogs nicht benötigen, sollten Sie die Option **Leserechte auf Kategorien anwenden** in der Konfiguration auf **Nein** stellen.

Die Permalinks von Serendipity werden standardmäßig mit der Variable `%id%` ausgestattet. Dies erhöht die Geschwindigkeit beim Zugriff auf einzelne Artikel enorm, daher sollten Sie diese Variable nur dann entfernen, wenn es für Ihre Seite oder Ihr Seelenwohl wirklich erforderlich ist.

[27] Empfehlenswert: *MySQL High Performance* von Jeremy D. Zawodny (ISBN 978-0-596003067).

Ähnlich wie bei den Leserechten von Artikeln können Sie die Ausführungsgeschwindigkeit von Plugins stark beschleunigen, wenn Sie die benutzerbasierte Plugin-Ausführung deaktivieren. Dies erreichen Sie durch die Option **Sollen persönliche Plugin-Rechte für Benutzergruppen aktiviert werden?** in der Serendipity-Konfiguration (siehe Seite 166).

Der regelmäßige Abruf von RSS-Feeds durch Besucher kann Ihren Server ebenfalls stark belasten. Hier können Sie auf externe Dienstleister wie FeedBurner zurückgreifen, um diese Last weiterzugeben (siehe Plugin **Blog abonnieren** auf Seite 207).

Plugins sinnvoll einsetzen

Der wohl am stärksten die Performance belastende Vorgang bei Serendipity ist der Einsatz von Plugins. Grundsätzlich sind Plugins zwar die größte Stärke des Systems, aber je mehr Plugins Sie aktivieren, desto langsamer wird selbstverständlich das Blog.

Sie sollten sich daher Gedanken darüber machen, was jedes einzelne Plugin technisch ungefähr erledigen muss, und überlegen, ob dies für Sie wirklich wichtig ist. Natürlich ist es verführerisch, alle möglichen Features in Ihrem Blog zu aktivieren – aber nachdem die erste Spielphase abgeklungen ist, sollten Sie die eingesetzten Plugins Ihres Blogs *konsolidieren*.

Entfernen Sie die Plugins, die geringen Wert haben – denn auch für Ihre Besucher ist weniger oft mehr.

Vermeiden Sie möglichst Seitenleisten-Plugins, die datenbankintensive Aktivitäten durchführen müssen. Das Seitenleisten-Plugin **Kategorien** erlaubt beispielsweise, die Anzahl der Einträge pro Kategorie einzubinden. Das klingt erstmal hilfreich. Aber performancetechnisch analysiert ist es ein wahrer Zeitkiller, denn dies bedeutet, dass bei jeder Erstellung der Kategorienliste alle Einträge nach deren Zuordnung durchsucht werden.

Wenn jede Minute (oder Omas Performance-Pfennige) zählt, können Sie auch darüber nachdenken, gewisse Inhalte von Seitenleisten-Plugins zu *statifizieren*. Wenn sich beispielsweise die Liste Ihrer Kategorien fast nie ändert, gibt es eigentlich keinen Grund, diese bei jedem Aufruf dynamisch aus der Datenbank neu zu erzeugen. Stattdessen könnten Sie einfach einen HTML-Klotz in der Seitenleiste platzieren, in der Sie manuell die Liste der Kategorien eintragen.

Auch die Ereignis-Plugins können starken Einfluss auf die Verarbeitungsgeschwindigkeit von Einträgen haben. Versuchen Sie überflüssige Textformatierungs-Plugins zu vermeiden. Das Plugin **Erweiterte Eigenschaften von Artikeln** verfügt zudem über eine Möglichkeit, Artikelinhalte zu cachen (siehe Seite 262).

Caching, Load Balancing

Das Serendipity-Plugin **Einfache Cached/Pregenerated Seiten** versucht, auf eine simple Art und Weise die Notwendigkeit von dynamischer Inhaltserzeugung zu reduzieren. Es nimmt jede durch Serendipity erzeugte Seite, speichert sie als HTML-Code ab und liefert diesen anstelle der dynamischen Daten aus (siehe Seite 377).

Da dieses Plugin aber weiterhin das Serendipity-Framework aufruft, kann es die Datenbanklast nicht völlig reduzieren.

Ein technisch probateres Mittel ist daher die Nutzung sogenannter Reverse Proxys. Diese Methode kann auch nur durch Ihren Provider veranlasst werden. Das Prinzip ist folgendes: Serendipity läuft nur auf einem internen, versteckten Server. Die Server, auf die ein Benutzer zugreift, dienen lediglich als Vermittler zu dem versteckten Server. Wenn ein Besucher eine Seite aufruft, versucht erst der Vermittler eine vorgehaltene Version des Ergebnisses auszuliefern. Nur wenn diese Version zu alt ist, fragt der Vermittler beim Blog-Server den aktuellen Inhalt ab. Die Einrichtung einer derartigen Serverumgebung würde den Rahmen dieses Buches leider spürbar sprengen. Da ein Reverse Proxy aber (meist) unabhängig von der eingesetzten Anwendung ist, können Sie mit Kenntnissen in dem Bereich Serendipity leicht wie gewünscht konfigurieren.

Auch die Benutzung von Load Balancing-Servern mit Serendipity ist möglich, solange alle verteilten Instanzen des Blogs auf dieselbe geteilte Datenbank zugreifen. Die lokal von Serendipity gespeicherten Daten sind für die Auslieferung der Seite irrelevant, da sämtliche Inhalte aus der Datenbank bezogen werden können.

8.10 Suchmaschinenoptimierung

Um Ihr Blog populär zu machen, gibt es unterschiedlichste Maßnahmen. Die wohl dauerhaft beste ist zugleich auch die schwerste: Betreiben Sie ein spannendes, unterhaltsames und einzigartiges Blog. Dann werden die Besucher mittelfristig von alleine kommen.

Um aber auch mit gewöhnlichen Blogs Besucher anzulocken, lohnt es sich, Ihre Seite bei Suchmaschinen anzumelden. Nutzen Sie Plugins wie das *Sitemap*-Plugin, damit sich Suchmaschinen bei Ihnen wohlfühlen. Setzen Sie auch Plugins wie **Meta-Beschreibungen** dazu ein, um Ihre Blog-Artikel optimal zu verschlagworten. Sprechende URLs durch URL-Rewriting unterstützen eine höhere Einordnung Ihrer Seite bei den eingesetzten Schlagwörtern.

Auch die Benutzung von Linkservices wie Mr. Wong, Digg und Technorati (siehe Plugin *Show links to services like Digg, Technorati etc.* auf Seite 303) hilft, Ihren Artikeln eine einfache Verbreitung zu ermöglichen.

Abseits davon gibt es viele Marketing-Möglichkeiten, um Ihr Blog bekannter zu machen. Veranstalten Sie Aktionen (Quizzes, Umfragen, Anleitungen), nehmen Sie an Bloggertreffen teil und kommentieren Sie auf anderen Blogs.

Analysieren Sie zudem Ihre Leserschaft anhand der Statistiken Ihres Blogs. Schauen Sie, wie die Besucher zu Ihnen gelangen, nach welchen Begriffen sie suchen – und verstärken Sie dieses Angebot.

Zu guter Letzt gilt aber: Spaß haben. Ein Blog ist wie kein anderes Web-Medium dazu da, um den Internet-Aufenthalt unterhaltsam zu machen. Nicht nur für Besucher, sondern auch für Sie als Schreiber. Ganz gleich, ob Sie kommerzielle Interessen verfolgen oder einfach nur

ein Tagebuch für sich schreiben wollen.

Kapitel 9

Anpassungen

Früher oder später wird jeder Benutzer von Serendipity zu dem Punkt kommen, an dem individuelle Anpassungen des Blogs gewünscht sind. Übliche Anpassungen reichen dabei von der Wahl der Grundfarbe und des Grundlayouts, eigenen Kopfgrafiken bis hin zu eigenständigen Unterseiten und Funktionalitäten.

Glücklicherweise bietet Serendipity für solche Anpassungen oft mehrere Möglichkeiten an, und Sie können über unterschiedliche technische Wege Einfluss nehmen. In den folgenden Abschnitten werden die einzelnen Möglichkeiten dargestellt, die meist aufeinander aufbauen und zum Ende hin komplexer werden.

9.1 Eingebaute Anpassungsmöglichkeiten

In den vorhergehenden Kapiteln haben Sie bereits gerlernt, wie Sie das Aussehen Ihres Blogs durch die Wahl eines Templates verändern können.

Seit Serendipity Version 1.1 besitzen Templates die Möglichkeit, Template-abhängige Optionen festzulegen. Diese Optionen können Sie als Administrator bzw. Redakteur des Blogs mittels der Template-Verwaltung (siehe Kapitel 4.6 ab Seite 142) beliebig anpassen.

Ab Serendipity 1.2 wird standardmäßig ein neues Template namens *Bulletproof* mitgeliefert. Dieses Template unterscheidet sich in seiner Zielsetzung wesentlich von normalen Serendipity-Templates. Üblicherweise ist ein Template auf ein spezielles Design ausgelegt. Dabei legt der Designer des Templates für Sie fest, wie Artikel dargestellt werden, welche Farben das Template benutzt und vor allem, wie das Grundlayout ausgelegt ist. Wenn Sie etwas an den Farben ändern möchten, müssen Sie meist selbst Hand an der Template-Programmierung anlegen, was ohne Detailkenntnisse vielen Benutzern schwerfällt.

An genau diese Benutzer richtet sich *Bulletproof*. Es stellt ein Template-Grundgerüst zur Verfügung, das sich an bestmöglicher Browser-Kompatibilität orientiert. Dieses Grundgerüst

können Sie mittels der Template-Optionen nun vielfältig gestalten: Sie können Farbschemas auswählen, bestimmen, wie viele Spalten das Layout besitzt, welche Navigationsleisten Sie einbinden wollen und auch wie die Blog-Artikel selbst gestaltet sind.

Über ein derartiges Baukasten-Prinzip (*Framework*) können Sie als Blog-Eigentümer z. B. leicht andere Kopfgrafiken einsetzen und sich vor allem darauf verlassen, dass die Basis des Frameworks browser-übergreifend annähernd gleich aussieht.

In zukünftigen Serendipity-Versionen wird dieses Template das Standarddesign ersetzen und dem Benutzer direkt von Anfang an diesen gestalterischen Freiraum genehmigen. Auch das Bulletproof-Template können Sie später über eigene Programmierung/Anpassungen verändern, wie Abschnitt 9.5 ab Seite 499 zeigen wird.

Abgesehen von den Template-Optionen bietet auch die Serendipity-Grundkonfiguration (siehe Seite 155) einige Möglichkeiten zur Anpassung der Darstellung Ihres Blogs (beispielsweise dem Blog-Titel und ob Popup-Fenster verwendet werden sollen).

Über die verfügbaren Plugins zu Serendipity können Sie bereits zahlreiche Funktionalitäten nachrüsten. Universale Plugins, wie die Einbindung beliebiger HTML-Texte, JavaScript oder PHP-Dateien (ab Seite 201), ermöglichen es Ihnen, leicht Änderungen vorzunehmen, ohne Dateien bearbeiten zu müssen.

9.2 Cascading Style Sheets

Alle Anpassungen des Layouts nimmt man heutzutage mittels der Formatierungssprache *CSS* (*Cascading Stylesheets*) vor.

Ein Stylesheet ist eine Datei, in der eine oder mehrere Anweisungen regeln können, wie ein HTML-Element dargestellt werden soll. Ein Stylesheet ist sozusagen eine Ergänzung der HTML-Datei: Die HTML-Datei legt die Struktur eines Dokumentes fest, und das Stylesheet regelt die Darstellung der Struktur.

Eine derartige Trennung von Inhalt und Aussehen hat zahlreiche Vorteile. Zum einen entschlackt dies die HTML-Datei, da in dieser nur noch der eigentliche Inhalt gespeichert wird. Dank dieser Trennung kann dasselbe Dokument abhängig von der Umgebung, in der es dargestellt wird, unterschiedlich formatiert sein. Klassisches Beispiel ist die Darstellung des Dokumentes auf dem Bildschirm und die Ausgabe auf dem Drucker. In der Druckversion kann z. B. ein Kommentarformular versteckt werden, da man es ja ausgedruckt sowieso nicht ausfüllen kann. Auch kann die Textbreite beim Ausdruck besser auf das Papierformat abgestimmt werden.

Gerade für sehbehinderte Menschen hat diese Trennung des Inhalts und der Präsentation enorme Vorteile: Sie können selbständig die Schriftgröße beeinflussen oder sich Texte von einem Lesegerät vorlesen lassen, ohne dass lästige Navigationselemente den Vorlesevorgang unterbrechen.

Vor allem machen es Stylesheets aber sehr einfach, schnell grafische Änderungen auszupro-

bieren. Um die Schriftart der ganzen Seite zu verändern, reicht es, eine einzige Anweisung einzufügen. In frühen Zeiten des Internets mussten dafür zahlreiche `<font>`-Tags einer Datei einzeln überarbeitet werden. Somit sind Stylesheets also gerade für Designer ein wahrer Segen.

Eine HTML-Datei besteht aus einer Menge von beliebig verschachtelten Elementen, den sogenannten Tags. Jedes Tag kann eine Reihe von Attributen und einen Inhalt besitzen. Ein Textabsatz wird beispielsweise so beschrieben:

```
<p>Dies ist ein Absatz.</p>
<p>Der zweite Absatz folgt <em>meistens</em> auf den ersten Absatz.</p>
<p><img src="bild.jpg" /></p>
```

Das Tag für einen Absatz lautet `p`, und der Absatztext selbst muss von einem öffnenden `<p>`-Tag und einem schließenden `</p>`-Tag umgeben sein. Im zweiten Absatz wurde ein weiteres Tag eingesetzt, das eine Zeichenkette *hervorhebt*. Auch hier ist jeweils ein öffnendes und ein schließendes Tag vonnöten.

Es gibt jedoch auch HTML-Tags, die keinen öffnenden/schließenden Teil benötigen, sondern alleine stehen können. Ein Beispiel für ein solches Tag ist die Einbindung eines Bildes mittels `img`. Die XHTML-Syntax erfordert für solche alleinstehenden Tags, dass diese mit einem `/>` enden. Anhand des `<img/>`-Tags lässt sich auch der Einsatz von HTML-Attributen erläutern. Das Attribut `src` enthält den Namen der Bilddatei, die angezeigt werden soll. Jedes Attribut muss immer durch ein Gleichheitszeichen einen Wert zugewiesen bekommen, der in einfachen oder doppelten Anführungszeichen zu stehen hat.

Nach diesen Grundprinzipien richten sich alle HTML-Dokumente, wobei es jedoch wesentlich mehr HTML-Tags als die hier aufgeführten gibt. Mittels einer Stylesheet-Datei kann man nun das Aussehen der Tags beeinflussen:

```
p {
  font-size: 1.2em;
}

p em {
   font-weight: bold;
}
```

Diese Syntax unterscheidet sich augenscheinlich von der Art und Weise, wie HTML programmiert wird. Dennoch haben beide Dateien einen gemeinsamen Nenner, nämlich die Namen der HTML-Tags sowie deren Verschachtelung.

Obiges Stylesheet würde dafür sorgen, dass unser HTML-Beispiel so ausgegeben wird, dass die Schriftgröße beider Absätze auf 120% der vom Benutzer eingestellten Schriftgröße gestellt wird. Das Wort *meistens* innerhalb unseres HTML-Beispiels wird dabei fett ausgezeichnet.

Grundsätzlich enthält ein Stylesheet also eine beliebige Anzahl von Formatierungsanweisungen. Jede Anweisung besteht aus einem Block: Erst folgt die Angabe, auf welche Elemente eine Formatierung angewandt werden soll (beispielsweise das `p`-Tag). Es folgt eine geschweifte Klammer ({) und danach eine Auflistung von Formatierungsanweisungen. Der Übersichtlichkeit halber sollte man jede Formatierungsanweisung in eine eigene Zeile schreiben. Als Erstes kommt der Name der Formatierungsanweisung (`font-size`), danach ein Doppelpunkt und danach ein Wert (`1.2em`). Das Ende der Zeile wird durch ein Semikolon (`;`) markiert. Nach einer beliebigen Anzahl von Formatierungsanweisungen wird der Block durch eine weitere geschweifte Klammer (}) wieder geschlossen, und es kann ein weiterer Block folgen.

Der Begriff `Cascading` bezeichnet eine wichtige Fähigkeit der Stylesheets: Die jeweiligen Formatierungsausdrücke können *vererbt* werden. Dazu muss man wissen, dass die Struktur einer HTML-Seite einer Baumstruktur ähnelt: Die Verschachtelung einzelner Elemente könnte man grafisch ähnlich darstellen, wie ein Dateimanager die Verzeichnisstruktur anzeigt.

Es kann also sein, dass ein Absatz im HTML-Dokument sowohl auf oberster Ebene vorkommt als auch innerhalb eines Unterabschnitts oder eines bestimmten Containers:

```
<div id="Seite">
  <div class="abschnitt">
    <p>Mein erster Abschnitt.</p>
  </div>

  <div class="abschnitt">
    <p>Mein zweiter Abschnitt.</p>
  </div>

  <div class="fusszeile">
    <p>Seitenende.</p>
  </div>
</div>
```

Anhand dieser Struktur sieht man, dass die Absätze (`p`-Tags) zwar in derselben Tiefe (immer als dritte Ebene) verwendet werden, aber jeweils einen unterschiedlichen Kontext besitzen: Einmal ist der Absatz Teil eines Abschnitts, aber am Ende ist ein Absatz innerhalb einer Fußzeile gesetzt.

Erst die grafische Umsetzung der HTML-Struktur über ein Stylesheet kann diesen strukturellen Unterschied sichtbar machen. Denn jeder Formatierungsblock, der das zu formatierende Element vor der geschweiften Klammer angibt, kann die hierarchische Einordnung des HTML-Elements wiedergeben:

```
#Seite {
  background-color: white;
}

.abschnitt {
  border: 1px solid black;
}

p {
  font-size: 0.8em;
}

.fusszeile p {
  font-weight: bold;
}
```

Wenn man sich jeden Block dieser Formatierungen einzeln ansieht, kann man die gewünschte Formatierung ablesen:

Der erste Block sorgt dafür, dass der `<div id="Seite">...</div>`-Container der HTML-Struktur mit einer weißen Hintergrundfarbe gestaltet werden soll. Jeder Stylesheet-Formatierungsblock kann über das Symbol # eine Formatierung direkt auf einen beliebigen Container anwenden, der in der HTML-Datei über das Attribut `id="..."` benannt wurde, in unserem Fall also `Seite`. Der zweite Block zeigt eine Formatierung, die auf einen

Container angewendet werden soll, der über das Attribut `class="abschnitt"` benannt wurde.

Eine CSS-Datei kann eine Formatierung auf ein HTML-Element also auf dreierlei Weise festlegen:

- `p { ...}` formatiert ein HTML-Tag anhand des Tag-Namens.
- `.klasse {...}` formatiert ein HTML-Tag anhand des `class`-Attributes (`<div class="klasse">`).
- `#bezeichner { ...}` formatiert ein HTML-Tag anhand des `id`-Attributes (`<div id="klasse">`).

Der Unterschied zwischen `id` und `class` liegt lediglich darin, dass eine ID nur einmalig vergeben werden darf, eine Klasse aber beliebig häufig Verwendung finden darf.

Das `id`- und `class`-Attribut lassen sich innerhalb der HTML-Struktur nicht nur einem `<div>`-Container zuweisen, sondern auch direkt einem `<p>` oder auch `<img>`-Element. Man kann es also immer dann verwenden, wenn man ein beliebiges HTML-Element eindeutig auszeichnen möchte.

Diese eindeutige Auszeichnung ist innerhalb des Stylesheets wichtig, damit man Formatierungen gezielt auf ein Element zuweisen kann. Dabei ist es nicht unbedingt erforderlich, dass man jedes zu formatierende Element mit einer speziellen Klasse benennt, da die Stylesheets wie angesprochen die *Vererbung* bzw. *Verschachtelung* einer Struktur berücksichtigen können. Dies verdeutlicht das obige Beispiel im vierten Formatierungsblock:

```
.fusszeile p {
  font-weight: bold;
}
```

Diese Formatierung sagt dem Browser: „Mache die Schriftart eines Absatzes fett, wenn der Absatz innerhalb einer Fußzeile eingesetzt wird“.

Jeder Formatierungsblock kann einzelne HTML-Elemente (entweder über deren Tag-Namen, die Klasse oder die ID) mit einem Leerzeichen voneinander getrennt angeben und entspricht dabei der Verschachtelung, wie sie später im Dokument vorzufinden ist.

Wenn also folgende HTML-Struktur gegeben ist:

```
<div class="seite">
  <div class="abschnitt">
     <p class="absatz">
       Mein <em>Einsatz</em>.
     </p>
  </div>
</div>
```

dann könnte man gezielt das Wort *Einsatz* über folgende CSS-Anweisung formatieren:

```
.seite .abschnitt .absatz em {
  font-weight: bold;
}
```

Anhand dieses Beispiels kann man auch gut verdeutlichen, dass eine derart komplexe Formatierungskette gar nicht unbedingt notwendig wäre. Denn die CSS-Anweisung:

```
em {
  font-weight: bold;
}
```

würde das obige Beispiel exakt gleich formatieren. Achten Sie also später beim Einsatz von CSS immer darauf, es nicht unnötig kompliziert zu machen. Überlegen Sie sich, anhand welcher Struktur Sie das zu formatierende Element bereits gezielt *herauspicken* können.

Mittels CSS-Anweisungen können Sie Elemente auch beliebig oft formatieren, da der Browser ein Element anhand aller zutreffenden Formatierungsanweisungen darstellt. Gegeben sei folgender HTML-Code:

```
<div class="Seite">
  <p>Mein Absatz.</p>
</div>
```

sowie folgende CSS-Anweisungen:

```
.Seite {
  color: red;
}

p {
  font-size: 1.2em;
}
```

Obwohl Sie die Farbe Rot nur dem Container `.Seite` zugewiesen haben, wird diese Farbe auch auf den Absatz angewendet, da dieser Teil des Seiten-Containers ist. Sie müssen also die Farbe nicht extra der Formatierungsanweisung für den Absatz zuweisen.

Abgesehen von diesen grundsätzlichen Strukturen bietet CSS noch einige spezielle Möglichkeiten an, die den Rahmen dieses Buches deutlich sprengen würden.[1]

[1] Wenn Sie diese Thematik interessiert, sollten Sie sich mit Büchern wie *Webdesign mit CSS* von Vladimir Simovic und Jan Heinicke oder Online-Werken wie SelfHTML (`http://de.selfhtml.org/css/`) vertraut machen.

9.2.1 CSS-Anweisungen in Serendipity-Templates

CSS-Formatierungen sind auch bei Serendipity ein integraler Bestandteil von Templates. In der Datei `style.css` eines jeden Template-Verzeichnisses liegen alle notwendigen CSS-Anweisungen, die auf die HTML-Ausgabe des Blogs angewendet werden. Diese CSS-Datei können Sie natürlich bearbeiten, um individuelle Formatierungen zu erreichen.

Die meisten Serendipity-Templates sind von einem Standard-Template abgeleitet, so dass die meisten Klassennamen der HTML-Elemente trotz unterschiedlicher Templates gleich benannt sind. Die meisten dieser Klassennamen sind über das Präfix `serendipity_` eindeutig benannt, so dass Sie über CSS-Anweisungen gezielt die Ausgabe eines Templates oder Plugins beeinflussen können.

Jedes Plugin der Seitenleiste ist üblicherweise in einen selbständig benannten HTML-Container eingebunden, damit Sie jedes Seitenleisten-Plugin individuell formatieren können:

```
<div id="serendipityLeftSideBar">
  <div class="serendipitySideBarItem
   container_serendipity_categories_plugin">
    <h3 class="serendipitySideBarTitel
     serendipity_categories_plugin">Titel</h3>
    <div class="serendipitySideBarContent">...</div>
  </div>
</div>
```

Wenn Sie beispielsweise alle Links der Kategorien-Ausgabe in der Seitenleiste grün einfärben wollen, könnten Sie dies mit der folgenden CSS-Regel erreichen:

```
.container_serendipity_categories_plugin a {
  color: green;
}
```

Eine Auflistung aller CSS-Klassen ist an dieser Stelle leider nicht möglich, da diese Klassen abhängig vom eingesetzten Template und dessen HTML-Ausgabe sind. Wenn Sie also eine Anpassung vornehmen wollen, müssen Sie die entsprechenden eingesetzten HTML-Klassen im Quelltext des jeweiligen Templates nachschlagen.

Dabei hilft z. B. die Extension `firebug`[2] für den Firefox-Browser enorm, da Sie per Cursor die HTML-Elemente einer Seite auswählen und direkt alle CSS-Anweisungen zu diesem Element sehen können.

Dennoch hier eine kleine Übersicht über CSS-Klassen, die in vielen Templates Verwendung finden:

`#mainpane, #content`
: Enthält in den meisten Templates das Grundgerüst der Seite.

[2] `http://www.getfirebug.com`

`#serendipity_banner`
: Enthält den Seitenkopfbereich, meist mit einem Hintergrundbild.

`.homelink1, .homelink2`
: Enthält innerhalb des Seitenkopfbereichs die Links zu der aktuellen Blog-Seite bzw. Blog-Übersicht.

`#serendipityLeftSideBar, #serendipityRightSideBar`
: Bezeichnet den Container, der jeweils die Elemente der Seitenleiste(n) enthält.

`.serendipitySideBarItem`
: Für jedes Element der Seitenleiste kapselt die Klasse den jeweiligen Inhalt des Plugins mitsamt dem Titel.

`.serendipitySideBarTitle`
: Enthält den Titel des jeweiligen Seitenleisten-Plugins.

`.serendipitySideBarContent`
: Enthält den Inhalt des jeweiligen Seitenleisten-Plugins.

`.serendipity_Entry_Date`
: Alle Blog-Einträge sind nach Veröffentlichungsdatum gruppiert, so dass mehrere Einträge eines Tages unter derselben Überschrift zusammengefasst werden und nicht jedesmal das Datum wiederholt wird. Der Container `.serendipity_Entry_Date` umfasst dabei jeweils einen Veröffentlichungstag.

`.serendipity_date`
: Das jeweilige Datum der Gruppe `.serendipity_Entry_Date` wird in der HTML-Klasse `serendipity_date` ausgegeben (meist ein `h3`-HTML-Tag).

`.serendipity_title, .serendipity_entry`
: Jeder einzelne Blog-Artikel der Klasse `.serendipity_Entry_Date` wird innerhalb dieses Containers noch weiter separiert. Der Titel wird jeweils über die Klasse `.serendipity_title` ausgegeben, der Inhalt des jeweiligen Artikels (mitsamt seiner Meta-Informationen wie Autorname, Kategoriezuordnung und Plugin-Ausgabe) wird innerhalb der Klasse `.serendipity_entry` ausgegeben.

`serendipity_entry_body, .serendipity_entry_extended`
: Enthält den Text der jeweiligen Blog-Einträge.

`.serendipity_imageComment_left,`

`.serendipity_imageComment_center,`

`.serendipity_imageComment_right, .serendipity_image_link` Falls Bilder mit Bildunterschriften in einen Blog-Artikel eingebunden worden sind, werden diese Klassen eingesetzt.

`.serendipity_entryFooter`
: Enthält die jeweiligen Meta-Informationen eines Blog-Artikels.

`.serendipity_entry_author_self`
: Diese Klasse wird bei allen Blog-Einträgen vergeben, die vom aktuell eingeloggten Besucher geschrieben wurden.

`.serendipity_entryIcon`
: Etwaige Kategorie- oder Autorenbilder werden innerhalb dieser Klasse ausgegeben.

`.serendipity_comments`
: Enthält den Container für Kommentare und Trackbacks zu einem Blog-Artikel.

`.serendipity_comment_author_self`
: Falls vom Autor des Blog-Artikels ein Kommentar geschrieben wurde, wird diese CSS-Klasse eingebunden.

`.serendipity_commentsTitle`
: Enthält den Titel für den jeweiligen Kommentarabschnitt („Kommentare“ oder „Trackbacks“).

`.serendipity_comment`
: Kapselt jeden einzelnen Kommentar zu einem Blog-Artikel.

`.serendipity_commentForm`
: Enthält das Kommentarformular.

`.serendipity_commentsLabel`
: Enthält die Überschriften zu den jeweiligen Eingabebereichen des Kommentarformulars.

`.serendipity_commentsValue`
: Enthält die Eingabeboxen der jeweiligen Eingabebereiche des Kommentarformulars.

`.serendipity_pageFooter`
: Enthält die Blätter-Anzeige für die Archivseiten der Blog-Übersichten.

`.serendipity_center`
: Etwaige Bildschirmmeldungen von Serendipity (z. B. bei Kommentarübermittlung oder Fehlern) werden mit dieser Klasse ausgezeichnet.

`.serendipity_msg_important, .serendipity_msg_notice,`

`.serendipity_content_message` Die jeweiligen Bildschirmmeldungen werden entsprechend ihrer Gewichtung (`notice` = Hinweis, `important` = Wichtige Information) mit den entsprechenden Klassen ausgezeichnet.

`.serendipity_search`
: Wenn der Besucher eine Volltextsuche ausgeführt hat, werden die Suchergebnisse innerhalb dieser Klasse ausgegeben.

9.2.2 CSS-Anweisungen von Serendipity-Plugins

Einige Plugins (wie *Freie Artikel-Tags*) beinhalten eigenständige CSS-Anweisungen. Diese Anweisungen werden nicht in die zentrale `style.css`-Datei Ihres Templates aufgenommen, sondern durch das Plugin dynamisch ausgegeben.

Dies hat den Vorteil, dass Sie Ihre CSS-Datei nicht manuell anpassen müssen, wenn Sie ein beliebiges Serendipity-Plugin installieren. Auch wird so die Notwendigkeit umgangen, dass alle Templates für die Benutzung mit Plugins speziell erweitert werden müssen.

Alle Serendipity-Plugins sind so programmiert, dass Sie die CSS-Anweisungen der Plugins *überschreiben* können. Dazu müssen Sie die Plugin-Datei nicht anfassen, sondern nur die Stylesheet-Datei Ihres Templates mit der CSS-Regel erweitern, die die Vorgaben des Plugins überschreibt.

Konkret heißt das, wenn das Plugin *Freie Artikel-Tags* eine CSS-Regel wie

```
.serendipity_freeTag {
  ...
}
```

ausgibt, müssen Sie diesen CSS-Code einfach in die eigene `style.css`-Datei überführen und dort beliebig anpassen. Das Plugin findet daraufhin den Code in Ihrer eigenen Datei und weiß, dass es selbst keine CSS-Anweisungen mehr ausgeben soll.

Sollte dies einmal nicht verlässlich klappen (wenn z. B. die HTML-Elementnamen nicht übereinstimmen), können Sie in Ihrer `style.css`-Datei die CSS-Regeln eines Plugins immer noch überschreiben, indem Sie eine Regel wie:

```
.serendipity_freeTag {
  color: red !important;
}
```

in Ihr Stylesheet übernehmen. Die Anweisung `!important` sorgt dafür, dass diese Zeile immer Vorrang vor anderen Anweisungen haben wird.

9.2.3 CSS-Hilfen

Beim Bearbeiten von CSS-Dateien brauchen Sie in der Regel immer zwei Dinge: Zum einen müssen Sie das zu formatierende HTML-Element innerhalb des Quelltextes ausfindig machen. Als Zweites müssen Sie dann den benötigten Element-Pfad bzw. Klassennamen mit einer entsprechenden Formatierungsanweisung in Ihre Stylesheet-Datei einfügen.

Dieser Prozess kann ziemlich müßig werden. Daher haben findige Köpfe für den Firefox-Browser ein Modul namens `Firebug`[3] erfunden. Dieses Programm ermöglicht es, mit einem einfachen Mausklick das benötigte Element ausfindig zu machen und direkt zu sehen, welche CSS-Anweisungen für dieses Element bereits belegt sind.

9.3 Smarty-Templates

Nachdem Sie nun gelernt haben, wie Sie CSS-Anweisungen überarbeiten können, wundern Sie sich möglicherweise bereits, wo Sie denn Einfluss auf die HTML-Ausgabe Serendipitys nehmen können.

Da Serendipity ein dynamisches System ist, das auf verschiedenen Seiten unterschiedliche Inhalte, Plugins und Artikellisten darstellen kann, muss auch die Ausgabe des HTML-Codes dynamisch erfolgen.

Dazu bedient sich Serendipity eines sogenannten Template-Frameworks. Die Hauptaufgabe eines Template-Frameworks ist es, Variablen zu interpretieren, abzufragen und auszugeben.

Einfach gesprochen, wollen Sie Serendipity gerne sagen, wo es einen Blog-Titel ausgeben soll, wo der Artikeltext hinkommt und wo die Seitenleisten platziert werden. Alle diese Inhalte werden durch Platzhalter (*Variablen*) markiert, über diese Platzhalter im Template eingebunden und später bei der Ausgabe automatisch durch die echten Inhalte ersetzt.

Außerdem soll ein Serendipity-Template erkennen können, ob der Besucher gerade die Detailansicht eines Artikels betrachtet oder die Liste aller Blog-Artikel zu einem gewissen Zeitpunkt. Abhängig von diesen Variablen muss Ihr Template unterschiedliche Dinge tun, dies nennt man *Darstellungslogik*.

Beides könnte man mit HTML alleine nicht erreichen, da HTML eine *statische* Beschreibungssprache ist, keine Programmiersprache. Mit PHP-Programmieranweisungen könnte man eine derartige Darstellungslogik ausdrücken, jedoch ist dies für viele Anwender viel zu kompliziert. Daher vertraut Serendipity auf eine Software-Bibliothek namens Smarty[4].

Smarty ist das Bindeglied zwischen Serendipity und Ihnen als Template-Bauer. Eine Smarty-Datei ist eine einfache Ansammlung von HTML-Elementen, Variablen und Darstellungslogik. Dabei greift Smarty auf übliche Programmiermuster zurück, eine Variablenabfrage erfolgt beispielsweise mit:

```
<div class="Absatz">
{if $is_single_entry}
 ...
{/if}
</div>
```

[3] `http://www.getfirebug.com/`, ähnliche Tools für andere Browser bzw. JavaScript-basierte Tools sind ebenfalls verfügbar.

[4] `http://smarty.php.net/`

Smarty-spezifischer Code wird dabei immer in geschweiften Klammern eingebunden. Folgt in einer geschweiften Klammer ein Dollar-Zeichen, gefolgt von einem Namen, so bezieht sich dies auf eine Variable. In obigem Beispiel wird geprüft, ob die Variable `$is_single_entry` gesetzt ist. Serendipity setzt diese Variable, und innerhalb eines Smarty-Templates können Sie diese Variable entweder ausgeben oder abfragen.

Das ist bis auf einige Sonderfälle bereits die ganze Magie: Smarty versteckt die Komplexität von PHP hinter einer eigenen Syntax, die von HTML-kundigen Benutzern leichter eingesetzt werden kann. PHP-Fans können für Serendipity dennoch PHP-Templates basteln, was ab Seite 692 eingehender beschrieben wird. Im Hintergrund arbeitet Smarty eine Template-Datei ab und wandelt sie in maschinenlesbaren PHP-Code um. Diesen Vorgang nennt man `kompilieren`, das Ergebnis des Kompilierens landet als temporäre Datei im Verzeichnis `templates_c`.

Smarty verfügt über eine große Anzahl an eigenen Funktionen, die in der offiziellen Dokumentation auf `http://smarty.php.net/` ausführlich beschrieben sind. Der folgende kurze Einblick in die Smarty-Syntax soll daher nur als Einstiegspunkt in die Materie dienen. Tiefergehende Möglichkeiten durch Erweiterung von Smarty finden Sie ab Seite 509.

9.3.1 Variablen und Modifiers

Eine Variable kann in einem Smarty-Template einfach über {`$variable`} ausgegeben werden. Der Inhalt der Variable erscheint dann später bei der HTML-Ausgabe anstelle des Platzhalters.

Häufig möchte man den Inhalt einer Variablen jedoch weiterverarbeiten. Wenn beispielsweise der Inhalt eines Blog-Artikels in der Variable {`$entry`} gespeichert wird, möchte man davon nur die ersten 20 Zeichen ausgeben.

Für diesen Zweck kennt Smarty das Konzept der `Variable Modifiers`. Ein Modifier ist eine Funktion, die auf eine Variable angewendet wird und deren Inhalt modifiziert zurückliefert. In obigem Beispiel würde man die Kürzung auf 20 Zeichen wie folgt erreichen:

```
{$entry|truncate:20}
```

Einen Modifier wendet man an, indem man das Pipe-Symbol (`|`) hinter die Variable schreibt, und danach den Namen des Variable Modifiers, in diesem Fall `truncate` zum Beschneiden. Ein Variable Modifier kann Parameter entgegennehmen, in diesem Fall die Zahl `20` zur Angabe der Menge von Zeichen, die ausgegeben werden sollen. Mehrere Parameter können mit `:` voneinander getrennt werden. Die vollständige Dokumentation der verfügbaren Parameter einer Funktion können Sie in der Smarty-Dokumentation unter `http://smarty.php.net/` nachschlagen. `truncate` verfügt z. B. über eine zweite Option, mit der man die Kürzung eines Textes durch das Anhängen einer Zeichenkette wie `...` verdeutlichen kann:

```
{$entry|truncate:20:'...'}
```

Des Weiteren sind Variable Modifiers schachtelbar, man kann also mehrere Modifier nacheinander (in der gewünschten Reihenfolge) ausführen. Wenn Sie beispielsweise den gekürzten Text noch so umwandeln wollen, dass jeder erste Buchstabe eines Wortes großgeschrieben wird, erreichen Sie dies mittels:

```
{$entry|truncate:20:'...'|capitalize}
```

Anstelle eines Parameters wie `:20` kann man an dieser Stelle auch immer auf im Template gesetzte Variablen zugreifen.

9.3.2 Arrays

Arrays sind eine besondere Form von Variablen, die in beinahe jeder Programmiersprache existieren. Während eine Variable wie {`$entry`} nur einen einzelnen Artikeltext enthalten kann, ist es manchmal wichtig, eine Variable zu besitzen, die mehrere Inhalte speichern kann.

Serendipity weist beispielsweise die Liste aller Blog-Artikel einer Smarty-Variable namens {`$entries`} zu. Diese Variable stellt eine Gruppierung von einzelnen Artikeln dar, wobei jeder einzelne Artikel Attribute wie `title`, `body`, `author` und Weiteres enthält. Solche Gruppierungen nennt man *Array*, sie können beliebig verschachtelt sein.

Ein Array ist so aufgebaut, dass es immer einen Schlüssel (*Key*) und einen Wert (*Value*) geben muss. Um den abstrakten Begriff eines Arrays besser zu verstehen, stellen Sie sich einen großen Schrank vor. Der Name des Schranks entspricht dem Namen des Arrays. In unserem Fall stellen wir uns die Beschriftung {`$entries`} auf dem Schrank vor. Dieser soll unsere Einträge beinhalten. In jede einzelne Schublade haben wir einen Eintrag hineingelegt. Jede Schublade besitzt eine eindeutige Beschriftung, wie *Artikel1*, *Artikel2* und so weiter. Einen Tick komplexer wird es, wenn wir nun in jeder einzelnen Schublade nochmal eine Fächerunterteilung haben. In einem Fach liegt die Beschreibung des Artikels, in einem anderen Fach der Name des Autors und in noch einem weiteren Fach der Titel des Artikels. Jedes einzelne Fach beschreibt eindeutig, was darin liegt: *author*, *title* und *body*.

Anhand dieser Beschriftungen kann nun der Inhalt jeder Schublade präzise adressiert werden. Wenn wir den Titel des zweiten Artikels herausholen wollen, würden wir sagen: Ich öffne den Schrank {`$entries`} , ziehe die Schublade `Artikel2` heraus und gucke dort in das Fach `title`. Diesen Vorgang kann man mittels Smarty-Syntax wie folgt ausdrücken:

```
{$entries.Artikel2.title}
```

Diese Syntax trennt mittels Punkten die unterschiedlichen Ebenen des Arrays untereinander auf. Von links nach rechts gelesen steht zuerst der *gröbste* Name des Arrays, dann geht es mit immer detaillierteren Fächerbezeichnungen in die Tiefe.

Das Beispiel mit dem Schrank mag zunächst trivial klingen, gewinnt aber an Komplexität, wenn man sich vorstellt, dass jede Schublade entweder einen endgültigen Wert besitzen kann

oder sich darin ein weiteres Fach (ein weiterer Schrank, im Endeffekt) befindet. So kann man unendliche Verschachtelungen erreichen, die man später anhand der Schlüssel-Bezeichnungen (*keys*) adressieren kann.

Während bei JavaScript solche Arrays ebenfalls der *Punkt-Notation* folgen und so angesprochen werden können, formatiert man den Zugriff auf ein Array innerhalb von PHP-Code leicht unterschiedlich. Dort schreibt man anstelle von {`$entries.Artikel2.title`} die Syntax {`$entries['Artikel2']['title']`}, setzt also jeden einzelnen Array-Schlüssel in eckige Klammern. Bei Smarty können Sie diese Notation meist nicht einsetzen, lediglich in einigen Sonderfällen (IF-Abfragen) wird dies von Smarty akzeptiert.

9.3.3 Schleifen, IF-Abfragen

Natürlich möchte man später eine Liste von Artikeln dynamisch ausgeben können. Die Abarbeitung eines Arrays, in der man dynamisch alle Inhalte ausgibt, nennt man *Schleife*. Schleifen werden in Smarty mittels der Funktionen {`foreach ...`} und {`/foreach`} erzeugt. Alles zwischen diesen beiden Aufrufen stellt den eigentlichen Inhalt der Schleife dar und bestimmt, was mit jedem Element eines Arrays passiert.

Greifen wir dazu erneut auf obiges Beispiel des Artikel-Schranks zurück. Um die Inhalte dynamisch auszulesen, müssten wir dies einem Roboter wie folgt erklären:

Öffne den Schrank mit dem Titel `entries`. *Gehe nacheinander alle Schubladen durch, und pro geöffneter Schublade machst Du Folgendes:* ...

In Smarty-Syntax formuliert sieht dies wie folgt aus:

```
{foreach from=$entries item="schublade"}
  <p>In dieser Schublade liegt der Artikel mit Titel
<strong>{$schublade.title}</strong>.</p>
{/foreach}
```

Wenn dieses Smarty-Template nun später kompiliert und ausgeführt wird, würde man folgende HTML-Ausgabe erhalten:

```
<p>In dieser Schublade liegt der Artikel mit Titel <strong>Artikel 1
  Titel</strong>.</p>
<p>In dieser Schublade liegt der Artikel mit Titel <strong>Artikel 2
  Titel</strong>.</p>
<p>In dieser Schublade liegt der Artikel mit Titel <strong>Artikel 3
  Titel</strong>.</p>
```

Für jeden im Schrank enthaltenen Artikel wurde die Schleife also einmal durchwandert, und die Inhalte wurden der Variable zugewiesen.

Ein besonders wichtiges Merkmal von Schleifen ist, dass bei jedem Durchgang (*Iteration*) einer Schleife durch ein Array das jeweilige Element immer einem Schlüssel (`key`) und einem Wert (`value`) zugewiesen wird. Ein Schlüssel ist dabei *immer* entweder ein numerischer Index oder eine Zeichenkette, er kann niemals ein weiteres Array enthalten. Jedoch kann der *Wert* entweder ein Array oder eine einfache Zeichenkette enthalten.

Beim Durchwandern eines Arrays muss Smarty (bzw. jede Programmiersprache) eine temporäre Variable benennen, mit der man auf das „aktuelle Element“ zugreifen kann. In obigem Beispiel ist dies die Variable `schublade`. Diese Bezeichnung haben wir nirgendwo auf dem Schrank angebracht, sie ist also ein komplett frei wählbares Konstrukt innerhalb der Programmierung. Sie könnten der Foreach-Schleife auch den Elementnamen `pusemuckel` zuweisen.

Wenn später der Zugriff auf {`$schublade.title`} erfolgt, ist der Unter-Array-Schlüssel (`title`) nicht frei wählbar. Denn dieser entspricht der Bezeichnung, die wir vorher auf dem Schrank fest angebracht haben. Der Begriff ist essentiell wichtig, damit auf den richtigen Wert zugegriffen werden kann.

Obige Foreach-Schleife ist leicht vereinfacht angewendet. Wie Sie vielleicht festgestellt haben, wird nur der `Wert` eines Array-Elementes weiterverwendet (als {`$schublade`}). Der Array-Schlüssel scheint irrelevant und wird nicht weiter benutzt. Manchmal wird für Sie aber der Schlüssel von Interesse sein, und in diesem Fall müssen Sie den Inhalt ebenfalls einer Übergangsvariable zuweisen:

```
{foreach from=$entries key="schluessel" item="schublade"}
  <p>In dieser Schublade liegt der Artikel mit Titel
<strong>{$schublade.title}</strong>.</p>
{/foreach}
```

Neu hinzugekommen ist das Attribt `key='...'`, das den Titel der Variable enthält. Beachten Sie hier, niemals Sonderzeichen wie Umlaute zu benutzen. Innerhalb der Foreach-Schleife können Sie nun {`$schluessel`} an beliebiger Stelle ausgeben. In unserem Beispiel würde dann *Artikel1* und *Artikel2* als Inhalt der Variable ausgegeben.

Nicht zu verwechseln mit einer Schleife ist eine `IF-Abfrage`. Mit dieser prüft man, ob eine Variable einen bestimmten Wert hat – nur wenn diese Abfrage nicht scheitert, wird ein dahinterliegender Code-Teil ausgeführt:

```
<div class="Absatz">
{if $is_single_entry}
 ...
{/if}
</div>
```

Nur wenn man sich beim Betrachten eines Artikels in der Detailansicht befindet, würde man die Ausgabe

```
<div class="Absatz">
 ...
</div>
```

erhalten, andernfalls würde man nur

```
<div class="Absatz">
</div>
```

sehen.

9.3.4 Funktionsaufrufe

Während Variablen durch Serendipity fest zugewiesen und an Ihre Smarty-Templates weitergereicht werden, können Funktionen einen beliebigen Algorithmus durchführen und dessen Rückgabewert darstellen.

Ein Beispiel für einen Funktionsaufruf wäre die Addition zweier Variablen. Wenn Sie die Anzahl von Kommentaren und Trackbacks für einen Blog-Artikel addieren wollen, können Sie nicht einfach Folgendes schreiben:

```
Gesamt: {$entry.comments}+{$entry.trackbacks}
```

denn dadurch würden Sie lediglich eine Ausgabe wie

```
Gesamt: 2+3
```

erhalten. Stattdessen müssen Sie folgenden Smarty-Funktionsaufruf bemühen:

```
Gesamt: {math equation="x+y" x=$entry.comments y=$entry.trackbacks}
```

Man muss also erst eine mathematische Formel mit Platzhaltern aufbauen und dann die substituierenden Werte als Attribute der Funktion übermitteln. Weitere gängige Smarty-Funktionen sind z. B. der Einsatz eines Zählers ({`counter`}) oder das beliebige Zuweisen von Variablen mittels {`assign`}.

Darüber hinaus werden von Serendipity zahlreiche Smarty-Funktionen zur Verfügung gestellt, diese sind ab Seite 563 erläutert.

Smarty-Templates werden bei Serendipity grundsätzlich im **Sicherheits-Modus** ausgeführt. Dieser verhindert, dass man innerhalb einer Smarty-Datei beliebige PHP-Funktionen aufrufen kann. Dies macht Templates weitaus sicherer im Einsatz, und wenn dies nicht gewollt ist, kann man es über die Template-Datei `config.inc.php` auch gezielt übergehen (siehe Seite 493).

9.3.5 Kommentare, Escaping, JavaScript

Grundsätzlich können Sie innerhalb einer Smarty-Template-Datei beliebigen HTML-Code und auch beliebiges JavaScript einfügen.

Da Smarty jedoch auf die Zeichen { und } speziell reagiert, können Sie diese Sonderzeichen nicht einfach innerhalb eines JavaScripts einsetzen, wo diese Zeichen häufig vorkommen. Stattdessen müssen Sie diese Zeichen als {`ldelim`} und {`rdelim`} platzieren. Alternativ können Sie einen Block mit JavaScript aber auch in einen {`literal`}`...`{`/literal`}-Funktionsblock setzen.

Kommentare, die später in der HTML-Ausgabe nicht sichtbar sein sollen, können Sie mittels {`* ...Kommentar ...*`} in der Smarty-Template-Datei einfügen.

9.3.6 Einbindung von Dateien

Smarty ermöglicht es mittels der Funktionen `include`, `include_php`, `insert` und `capture`, externe HTML-Dateien oder PHP-Dateien einzubinden. Dies ist jedoch bereits durch andere Serendipity-Plugins abgedeckt und auch aufgrund der aktivierten Smarty-Sicherheitsfunktionen standardmäßig nicht nutzbar. Ab Seite 679 wird genauer beschrieben, wie man externe Inhalte mit Serendipity am besten einbindet.

9.4 Template-Dateien

Ein Serendipity-Template besteht aus zahlreichen Dateien: Stylesheet, Info-Datei, Grafiken und Smarty-Templates. Damit Sie diese bearbeiten können, müssen Sie zunächst Ihr eigenes Template-Verzeichnis ausfindig machen.

Den Namen Ihres Template-Verzeichnisses finden Sie, indem Sie mit einem FTP-Programm (oder Ähnlichem) einen Blick in das Verzeichnis `templates` werfen. Dort sehen Sie Unterverzeichnisse, deren Namen meist einen Rückschluss auf das jeweilige Template zulassen. Der Verzeichnisname muss jedoch nicht zwingend dem entsprechen, den Sie im Backend im Menüpunkt `Styles verwalten` sehen. Daher müssen Sie ggf. die Datei `info.txt` im jeweiligen Template-Verzeichnis öffnen, darin ist der angezeigte Name des Templates enthalten.

Ein Serendipity-Template besteht also mindestens aus einem Verzeichnis, der angesprochenen `info.txt`-Datei und einer beliebigen Anzahl weiterer Dateien. Findet Serendipity eine Datei nicht in Ihrem Template-Verzeichnis, benutzt es automatisch die Standarddatei.

Um ein eigenes Template zu erstellen, benötigen Sie als Erstes ein neues Verzeichnis für Ihre Dateien. Nehmen wir an, Sie wollen Ihr Template `nirvana` nennen. Erstellen Sie dazu ein neues Verzeichnis `templates/nirvana/`. In diesem Verzeichnis müssen Sie nun eine `info.txt`-Datei erstellen, beispielsweise mit folgendem Inhalt:

```
Name: Nirvana
Author: Ich
Date: 2007
```

Die jeweiligen Bezeichner `Name, Author, Date` müssen Sie exakt so übernehmen. Alles was hinter dem Doppelpunkt steht, bleibt Ihrer Kreativität überlassen.

Nachdem diese Datei erstellt wurde, können Sie Ihr Template im Backend über den Menüpunkt `Styles verwalten` bereits auswählen. Da ansonsten noch keine Dateien vorhanden sind, bezieht Serendipity diese automatisch aus dem Standard-Template, `/templates/default` und `/templates/carl_contest/`.

Das `carl_contest`-Verzeichnis wurde mit Veröffentlichung von Serendipity 1.0 zum Standardverzeichnis, während `default` noch einige ältere Standarddateien enthält. Somit ergänzen sich beide Verzeichnisse, was für Template-Anpassungen manchmal verwirrend scheinen mag.

Wenn Serendipity in Ihrem Template-Verzeichnis eine angeforderte Datei wie `index.tpl` nicht findet, sucht es erst im `carl_contest`-Verzeichnis und danach im `default`-Verzeichnis. Wenn Sie eine Datei wie `index.tpl`, `style.css` oder `entries.tpl` anpassen wollen, müssen Sie diese Datei erst in Ihr eigenes Template-Verzeichnis kopieren und dort anpassen.

Niemals sollten Sie Dateien außerhalb eines eigenen Template-Verzeichnisses bearbeiten. Obwohl es verlockend scheinen mag, einfach die Datei `templates/default/style.css` direkt zu ändern, sollten Sie darauf unbedingt verzichten. Andernfalls würden Ihre Änderun-

gen an solchen Dateien bei einem Serendipity-Update unweigerlich überschrieben werden, und Ihre Änderungen wären verloren.

Damit Sie später bei Serendipity-Aktualisierungen etwaige Verbesserungen der Standard-Templates automatisch auch in Ihrem Template sehen können, ist es sehr empfehlenswert, wirklich nur die Dateien in Ihr Template-Verzeichnis zu kopieren, die Sie anpassen wollen. Wenn Sie `index.tpl` niemals verändern, sollten Sie sie der besseren Übersichtlichkeit und Kompatibilität wegen aus Ihrem eigenen Template-Verzeichnis löschen.

Grundsätzlich können Sie in einem Template-Verzeichnis beliebige Dateien speichern, also auch Bilddateien, JavaScripts, Flash-Videos oder eigene PHP-Codeschnipsel. Einige Serendipity-Plugins können ebenfalls über angepasste Template-Dateien ausgegeben werden, diese Template-Dateien (z. B. `plugin_staticpage.tpl`) können Sie auch in Ihr eigenes Template-Verzeichnis übernehmen – dieses Verzeichnis hat immer Vorrang vor dem Standard-Plugin-Verzeichnis.

Es folgt eine Liste aller Dateien, die Serendipity in einem Template-Verzeichnis ansprechen kann.

9.4.1 CSS-Dateien

Die aufgeführten Cascading-Stylesheet-Dateien enthalten Anweisungen zur Formatierung. Innerhalb der Datei `style.css` kann der Platzhalter {`$TEMPLATE_PATH`} eingesetzt werden, der später durch den vollständigen Pfad zum gewählten Template ersetzt wird. Da das Stylesheet vom Stammordner und nicht vom jeweiligen Template-Ordner aus vom Browser aufgerufen wird, müssen die Pfadverweise jeweils mit absoluter Angabe erfolgen. Eine relative Angabe müsste sich stets auf den Stammpfad beziehen. Eine Anweisung wie

```
.banner {
  background-image: url(pic.jpg);
}
```

würde nach der Datei `http://www.example.com/serendipity/pic.jpg` suchen, da das Stylesheet über `http://www.example.com/serendipity/serendipity.css` virtuell aufgerufen wird. Um aber auf die Template-Datei `http://www.example.com/serendipity/tem` zuzugreifen, wäre folgende CSS-Anweisung korrekt:

```
.banner {
  background-image: url({TEMPLATE_PATH}pic.jpg);
}
```

`style.css`
: Enthält die zentralen CSS-Regeln für den gesamten Frontend-Bereich eines Templates.

`admin/style.css`
: Enthält die CSS-Regeln für den Backend-Bereich eines Templates.

`admin/imgedit.css`
: Enthält die CSS-Regeln für den (derzeit noch nicht verwendeten) Bild-Editor der Mediendatenbank.

`admin/pluginmanager.css`
: Enthält CSS-Regeln für die Darstellung der Plugin-Übersicht im Backend. Diese CSS-Regeln werden zusätzlich zum zentralen Admin-Stylesheet eingelesen.

`atom.css`
: Enthält CSS-Regeln, die bei der Darstellung eines Atom-Feeds angewandt werden können. Üblicherweise wird ein Atom-Feed nur innerhalb eines Feedreaders eingebunden, das Stylesheet wird nur dann angewendet, wenn ein Besucher den Atom-Feed innerhalb eines normalen Browsers anzeigt.

`htmlarea.css`
: Enthält CSS-Regeln für Serendipitys integrierten WYSIWYG-Editor.

9.4.2 TPL-Dateien (Frontend)

Die Dateien mit der Endung `.tpl` stellen Smarty-Templates dar. Sie enthalten die jeweiligen Anweisungen, um die von Serendipity gelieferten Variablen korrekt in eine HTML-Ausgabe einzubinden.

Innerhalb jeder Datei sind spezielle Smarty-Variablen verfügbar (siehe Seite 510). Für einen vollständigen Seitenaufruf können mehrere einzelne Template-Dateien involviert sein, die am Ende von Serendipity vor der Ausgabe automatisch miteinander verbunden werden.

`index.tpl`
: Enthält das Grundgerüst (HTML-Kopf, -Fuß und Seitenleisten-Einbindung) jeder Serendipity-Seite. Hier können Sie sämtliche statischen Inhalte einbinden, wie zum Beispiel eigene Navigationsleisten, JavaScripts oder Banner und Footer.

`content.tpl`
: Enthält den grundsätzlichen Inhaltsbereich, der in der `index.tpl`-Datei über die Variable {`$CONTENT`} eingebunden wird. Der Inhaltsbereich entscheidet, ob etwaige Meldungen an den Besucher weitergereicht werden und ob z. B. eine Volltextsuche ausgeführt wurde.

`entries.tpl`
: Stellt einen oder mehrere Blog-Artikel dar. Innerhalb dieser Datei können die Blog-Artikel ({`$entries`}) in einer Schleife abgearbeitet werden. Abhängig von weiteren Variablen wird entschieden, ob eine Übersicht oder eine Einzel-Artikel-Darstellung gewünscht ist.

`sidebar.tpl`
: Enthält die Ausgaben für die Elemente einer Seitenleiste. Diese Datei wird über den Smarty-Funktionsaufruf `serendipity_printSidebar` innerhalb des `index.tpl`-Templates ausgegeben und jeweils einzeln für die linke, rechte und etwaige weitere Seitenleisten durchlaufen.

`commentform.tpl`
: Enthält das Kommentarformular, mit dem Besucher Kommentare zu einem Blog-Artikel hinterlassen können.

`commentpopup.tpl`
: Falls Kommentare in einem Popup-Fenster dargestellt werden sollen, wird diese Template-Datei verwendet, um das HTML-Grundgerüst für dieses Popup-Fenster auszugeben.

`comments.tpl, trackbacks.tpl`
: Stellt eine Liste von Kommentaren oder Trackbacks dar.

`comments_by_author.tpl`
: Stellt eine Liste aller Kommentare eines gewünschten Blog-Kommentators im Inhaltsbereich dar.

`entries_archives.tpl`
: Stellt eine Archiv-Übersicht der Blog-Artikel (im Inhaltsbereich) dar.

`entries_summary.tpl`
: Stellt eine zusammengefasste Archiv-Übersicht der Blog-Artikel im Inhaltsbereich dar.

`feed_0.91.tpl`, `feed_1.0.tpl`, `feed_2.0.tpl`, `feed_atom0.3.tpl`,

`feed_atom1.0.tpl`, `feed_opml1.0.tpl` Für jeden RSS-Feed bestimmt eine nach dem jeweiligen Feed-Typen benannte Template-Datei die Darstellung. So können gewünschte Änderungen am Layout und den eingesetzten Elementen der RSS-Feeds template-basiert durchgeführt werden.

: Eigene RSS-Typen können ebenfalls hinzugefügt werden. Diese werden nach dem Schema `http://www.example.com/serendipity/rss.php?version=nirvana` aufgerufen und sprechen die Template-Datei nach dem Schema `feed_nirvana.tpl` an.

`plugin_calendar.tpl`
: Die Ausgabe des Kalenders in der Seitenleiste wird über die Datei `plugin_calendar.tpl` kontrolliert. Diese Template-Datei enthält mehrere Schleifen zur Darstellung des jeweiligen Monats bzw. Jahres.

`plugin_remoterss.tpl`
: Das Seitenleisten-Plugin *Fremder RSS/OPML-Blogroll Feed* (siehe Seite 210) macht

seine Ausgaben via Smarty-Template-Datei `plugin_remoterss.tpl`, wenn es entsprechend konfiguriert wurde. Andernfalls nutzt das Plugin die fest einprogrammierte HTML-Ausgabe, um die Performance zu steigern.

`plugin_categories.tpl`
: Das Seitenleisten-Plugin *Kategorien* kann so konfiguriert werden, dass es die Smarty-Template-Datei `plugin_categories.tpl` zur Ausgabe anspricht. Andernfalls nutzt das Plugin die fest einprogrammierte HTML-Ausgabe, um die Performance zu steigern.

  Innerhalb dieser Template-Datei können Dinge wie die Darstellung der RSS-Symbole pro Kategorie wie auch die Verschachtelung von Unterkategorien individualisiert werden.

`preview_iframe.tpl`
: Wird im Backend die Vorschau für einen Blog-Artikel ausgeführt, stellt der Browser diese Vorschau in einem eigenen *iframe* dar. Dieser *iframe* simuliert die Darstellung des Artikels im Frontend, indem die dafür benötigten Stylesheets eingebunden werden.

  Die Art und Weise der Darstellung des iframe wird über die Template-Datei `preview_iframe.tpl` kontrolliert. In dieser Datei sind JavaScript-Anweisungen vorhanden, die dynamisch die korrekte Höhe der Darstellung des Blog-Artikels ermitteln. Falls das gewählte Template den zentralen Blog-Inhaltsbereich individuell formatiert, kann es notwendig sein, diese `preview_iframe.tpl`-Datei an die HTML-Struktur des Frontends anzugleichen.

9.4.3 TPL-Dateien (Backend)

Der HTML-Code des Serendipity-Backends ist größtenteils fest im Quellcode von Serendipity verdrahtet. Dies geschieht, damit neue Serendipity-Versionen problemlos neue Features einbauen können, für die andernfalls jedes Mal eine Anpassung Ihres Templates notwendig wäre. Zudem ist die Darstellung mit Verzicht auf Smarty etwas schneller, und da der Backend-Bereich selten von Benutzern angepasst wird, ist dieser Kompromiss meist problemlos.

Damit Sie jedoch dennoch Teile des Backends zu eigenen Zwecken und für eigene Templates anpassen können, sind das grundlegende Seitenlayout, die Oberfläche der Mediendatenbank sowie der Artikel-Editor von Serendipity über Template-Dateien zugänglich. Zusammen mit der Stylesheet-Datei des Backends können so beinahe alle Aspekte des Backends verändert werden.

`admin/index.tpl`
: Enthält die Grundstruktur des Serendipity-Backends, mit Positionierung des Inhaltsbereichs, der Überschrift und der Menüpunkte.

`admin/entries.tpl`
: Enthält die Oberfläche zum Bearbeiten bzw. Erstellen eines Blog-Artikels. In dieser

Datei können die Eingabefelder für den Titel des Beitrags, das Datum, die Kategorieauswahl, die erweiterten Optionen und alle weiteren Felder beliebig platziert werden.

`admin/media_choose.tpl`
: Enthält das Grundgerüst der Mediendatenbank-Oberfläche, die über ein Popup-Fenster ausgegeben wird. Innerhalb dieser Datei werden mehrere Framesets verwaltet und die Inhalte des Verzeichnis-Frames ausgegeben. Somit dient die Datei als eine Art zentrale Anlaufstelle für alle Aktionen, die in der Mediendatenbank ausgeführt werden können (Verzeichnisverwaltung, Rechteverwaltung, Darstellung der Übersicht, Darstellung einer einzelnen Datei, Auswahl einer Datei).

`admin/media_imgedit.tpl, admin/media_imgedit_done.tpl`
: Enthält einen rudimentären Bildeditor der Mediendatenbank, der von Serendipity derzeit noch nicht genutzt wird.

`admin/media_pane.tpl`
: Dieses Template gibt das Gerüst für die Ansicht der Mediendatenbank aus und stellt den Kopf- und Fußbereich für die Verzeichnisinhalte dar (Filter, Blättern, Suche ...).

  Der Inhaltsbereich wird über die Variable {`$MEDIA_ITEMS`} aus der Datei `media_items.tpl` bezogen.

`admin/media_items.tpl`
: Diese Datei regelt die Darstellung der Liste von Dateien in der Mediendatenbank. Sowohl die Übersicht als auch die Darstellung einer einzelnen Datei wird innerhalb dieses Templates ausgegeben.

  Die Liste wird mittels der Variable {`$MEDIA_ITEMS`} in der Datei `media_pane.tpl` oder `media_properties.tpl` eingebunden und ausgegeben.

`admin/media_properties.tpl`
: Stellt die Eigenschaften einer gewählten Datei der Mediendatenbank dar. Die Datei `media_properties.tpl` regelt dabei den umgebenden Rahmen, die eigentlichen Detailinformationen werden über die Variable {`$MEDIA_ITEMS`} aus der Datei `media_items.tpl` bezogen.

`admin/media_upload.tpl`
: Dieses Template enthält das Formular zum Hochladen einer neuen Datei in die Mediendatenbank.

`admin/media_showitem.tpl`
: Falls die Mediendatenbank benutzt wird, um im Frontend eine einzelne Datei darzustellen, dient das Template `media_showitem.tpl` dazu, das Grundgerüst zu den Informationen der Datei zu formatieren. Diese Datei kann somit unabhängig von der Backend-Darstellung der Mediendatenbank angepasst werden.

9.4.4 PHP-Dateien

`config.inc.php`

Jedes Template kann eine `config.inc.php`-Datei benutzen, um darin beliebige PHP-Befehle einzubinden.

Diese Datei wird stets geladen, wenn Serendipity eine Seite im Frontend einbindet. Kurz nach Initialisierung des Smarty-Frameworks wird die `config.inc.php` aufgerufen, so dass Sie innerhalb dieser Datei bereits auf das Objekt `$serendipity['smarty']` zugreifen können, beispielsweise um eigene Modifiers oder Functions zu registrieren.

Auch die Einbindung von PHP-Zugriffszählern oder Ähnlichem bietet sich in dieser Datei an. Die `config.inc.php` kann dabei jedoch keine Ereignis-Plugins ersetzen, sondern dient ergänzend dazu.

Ein häufiger Einsatzzweck für diese Datei ist es, die Smarty-Sicherheitsfunktion zu deaktivieren. Dazu tragen Sie in der `config.inc.php`-Datei folgenden Code ein:

```
<?php
$serendipity['smarty']->security = false;
?>
```

Dies bewirkt, dass Sie nun auch auf die potenziell gefährlichen Smarty-Befehle wie `include` und `include_php` zugreifen können. Sie sollten diese Sicherheitsfunktion nur dann deaktivieren, wenn nur vertrauenswürdige Personen FTP- oder Dateizugriff zu Ihrem Server haben.

Abgesehen von beliebigem PHP-Code dient `config.inc.php` auch zur Festlegung der Optionen eines Templates, die im Backend im Menüpunkt **Styles verwalten** angezeigt werden können. So macht beispielsweise das Template *Bulletproof* intensiven Gebrauch von solchen Optionen, um dem Benutzer die Wahl des Header-Bildes zu lassen, eine Farbwahl zu treffen oder Navigationslinks einzubinden.

Die Syntax zur Festlegung und Benutzung von Template-Optionen finden Sie ab Seite 499.

`lang_*.php, UTF-8/lang_*.php`

Falls das Template mittels der `config.inc.php`-Datei eigene Template-Optionen einbindet, sind die Beschreibungen dieser Optionen für die Darstellung im Backend meist in eigenständige Sprachdateien ausgelagert. Diese entsprechen dem Schema `lang_de.inc.php` (deutsche Sprache) und können anhand des Sprachkürzels später von Serendipity abhängig von der vom Benutzer gewählten Sprache geladen werden.

Diese Dateien befinden sich nochmals im Unterverzeichnis `UTF-8`, dort liegen sie im UTF-8-Zeichensatz gespeichert. Im Hauptverzeichnis hingegen liegen die Dateien immer im nativen Zeichensatz, meist `ISO-8859-1`.

9.4.5 Bilder, Metadaten

Jedes Template liefert meist auch eine Fülle an Grafikdateien. Sie können innerhalb Ihres Templates beliebige Grafikdateien ansprechen, aber zusätzlich greift Serendipity zur Darstellung einiger Objekte auf feste Dateinamen zurück.

Wenn Ihr Template eine Datei gleichen Namens besitzt, wird diese von Serendipity anstelle der Standarddatei ausgegeben. Zur Gestaltung dieser Dateien können Sie sich nach Vorlagen im Verzeichnis `templates/default/` richten.

`preview.png`
: Diese Datei enthält eine Vorschaugrafik des Aussehens des jeweiligen Templates. Den Screenshot müssen Sie manuell erstellen, damit man Ihr Template anhand der Vorschaugrafik identifizieren kann.

  Die Vorschaugrafik wird in der Oberfläche `Styles verwalten` angezeigt. Die Grafik sollte genau 100 Pixel breit sein, die Höhe ist beliebig.

`preview_fullsize.jpg`
: Die große Variante der Vorschaugrafik kann unter dem Namen `preview_fullsize.jpg` gespeichert werden und einen Screenshot Ihres Templates in voller Pracht enthalten. Die Größe der Grafik kann beliebig gewählt werden, mindestens 800 x 600 Pixel werden empfohlen.

`treeview/*.gif, treeview/*.css`
: Wenn ein Redakteur ein Bild in einen Blog-Artikel einfügen möchte, stellt die Verzeichnisliste der Mediendatenbank, die in einem Popup-Fenster angezeigt wird, einige Grafiken dar, damit die Verschachtelung der Verzeichnisstruktur gut erkennbar ist. Dazu greift Serendipity auf das YahooUI-Element `treeview` zurück, dessen Grafikdateien im Ordner `templates/default/treeview` liegen. Sie können diese Dateien durch eigene Grafiken ersetzen und in Ihrem jeweiligen Template abspeichern. Die Stylesheet-Datei im gleichen Verzeichnis kontrolliert die weitere grafische Gestaltung der Verzeichnisliste.

`img/emoticons/`
: Die Smilies des gleichnamigen Textformatierungs-Plugins (siehe Seite 253) werden im Verzeichnis `img/emoticons/` gespeichert und können in Ihrem eigenen Template-Verzeichnis beliebig ersetzt werden. Sie können beispielsweise die Smilie-Grafiken Ihres Lieblingsforums in dieses Verzeichnis einstellen, die Dateinamen der jeweiligen Smilies sind selbsterklärend.

`img/back.png`, `img/forward.png`, `img/down.png`, `img/up.png`
: Diese Grafiken stellen einen Pfeil dar und werden zum Vor- und Zurückblättern des Kalenders in der Seitenleiste eingebunden.

`img/delete.png`, `img/rename.png`, `img/scale.png`, `img/zoom.png`
: Enthält Symbole für Elemente der Mediendatenbank (Bild löschen, Bild umbenennen, Beschneiden, Vergrößern).

`img/img_align_left.png`, `img/img_align_right.png`,

`img/img_align_top.png` Wenn ein Redakteur in einem Blog-Artikel ein Bild aus der Mediendatenbank einbindet, kann er in einem Auswahldialog bestimmen, wie diese Bilder ausgerichtet sein sollen. Dazu werden kleine Vorschaugrafiken zur Ausrichtung angezeigt, die der jeweiligen Grafikdatei wie `img_align_left.png` entsprechen. Wenn Sie das Aussehen dieser Ausrichtungsoption stärker an das tatsächliche Design Ihres Blogs anpassen wollen, können Sie diese Vorschaugrafiken austauschen.

`img/minus.png, img/plus.png`
Enthält Grafiken zum Ein- und Ausklappen von Elementen, wie beispielsweise in der Oberfläche zur Konfiguration von Serendipity.

`img/s9y_banner_small.png`
Zeigt ein kleines Werbebanner für Serendipity, das vom Plugin *Powered by* (siehe Seite 210) in der Seitenleiste dargestellt werden kann.

`img/xml.gif`
An mehreren Stellen des Frontends stellt Serendipity eine kleine Grafik für RSS-Feeds dar, die typischerweise einem XML-Button ähnelt. Wenn Sie diese Grafik austauschen wollen, können Sie dies über die Datei `xml.gif` erreichen.

9.4.6 Bilder im Backend

Besonders im Backend setzt Serendipity einige Bilder und Symbole ein, die abhängig vom gewählten Template sein können. Diese Grafiken enthalten oft spezielle Symbole, die Sie aber problemlos durch andere bevorzugte Symbole austauschen können.

`admin/img/accept.png, admin/img/admin_msg_success.png`
Stellt ein grünes Häkchen dar, wenn eine Aktion erfolgreich war oder bestätigt wurde.

`admin/img/admin_msg_error.png`
Stellt ein Warndreieck dar, wenn eine Serendipity-Aktion nicht erfolgreich war.

`admin/img/admin_msg_note.png`
Stellt eine Informationsgrafik dar, wenn eine Serendipity-Aktion einen Hinweis ausgibt.

`admin/img/background.jpg`
Wird im Standard-Stylesheet als Hintergrundgrafik des Backends eingesetzt.

`admin/img/banner_background.png`
Wird im Standard-Stylesheet als Hintergrundgrafik der Überschrift im Backend eingesetzt.

`admin/img/infobar_background.png`
: Wird im Standard-Stylesheet als Hintergrundgrafik für den Login-Bereich des Backends eingesetzt.

`admin/img/big_delete.png, admin/img/big_rename.png,`

`admin/img/big_resize.png, admin/img/big_rotate_ccw.png,`

`admin/img/big_rotate_cw.png, admin/img/big_zoom.png,`

`admin/img/rotate.png, admin/img/zoom.png` Mehrere Symbole der Mediendatenbank zum Löschen, Umbenennen, Beschneiden, Rotieren und Vergrößern.

`admin/img/button_background.png`
Hintergrundgrafik für Aktions-Buttons im Serendipity-Backend.

`admin/img/clock.png`
Symbol für die Uhrzeit beim Erstellen/Bearbeiten eines Blog-Artikels.

`admin/img/clock_future.png`
Eingefärbtes Symbol für die Uhrzeit, das bei einem Blog-Artikel in der Backend-Übersicht erscheint, wenn der Artikel noch nicht veröffentlicht wurde.

`admin/img/configure.png, admin/img/unconfigure.png`
Symbol zum Konfigurieren eines Elements (aktiv/inaktiv).

`admin/img/delete.png`
Symbol zum Löschen eines Elements (Beiträge, Kategorien, Verzeichnisse . . .).

`admin/img/downarrow.png, admin/img/uparrow.png`
Symbole zum Verschieben (höher/tiefer) eines Elements.

`admin/img/edit.png`
Symbol zum Bearbeiten eines Elements.

`admin/img/folder.png`
Symbol zur Darstellung eines Verzeichnisses.

`admin/img/grablet.gif, admin/img/grablet_over.gif`
Symbole zur Darstellung in der Plugin-Verwaltung, um den Maus-Anfassbereich eines Plugins zu markieren.

`admin/img/imgedit_.gif`
Mehrere Bilder für den rudimentären Bildeditor von Serendipity, der noch nicht verwendet wird.

`admin/img/install.png`
Symbol zum Aktivieren eines Plugins oder Templates.

`admin/img/install_now.png`
Symbol zum Installieren eines Plugins oder Templates.

`admin/img/install_now_spartacus.png`
: Symbol zum Download eines Plugins oder Templates (mittels Spartacus-Plugin).

`admin/img/install_template.png`
: Symbol zum Aktivieren eines Templates.

`admin/img/upgrade_now.png`
: Symbol zum Aktualisieren eines Plugins/Templates.

`admin/img/menu_background.png`,

`admin/img/menuheader_background.png`,

`admin/img/menuitem.png` Grafiken zur Darstellung des Menüs im Backend-Bereich.

`admin/img/mime_unknown.png`
: Symbol für unbekannte Dateitypen der Mediendatenbank.

`admin/img/next.png`, `admin/img/previous.png`
: Symbole zum Vor- und Zurückblättern einer Seite.

`admin/img/thumbnail.png`
: Symbol für die Voransicht einer Grafikdatei.

`admin/img/user_admin.png`, `admin/img/user_chief.png`,

`admin/img/user_editor.png` Symbole für Benutzergruppen der Redakteure.

9.4.7 JavaScripts

Jedes Template kann zudem eine kleine Menge von JavaScripts liefern. Diese werden von Templates jedoch meist selten angepasst und greifen üblicherweise auf das Standard-Template zurück.

`dragdrop.js`
: JavaScript für die Drag&Drop-Funktionalitäten in Serendipity, beispielsweise bei der Plugin-Verwaltung.

`imgedit.js`
: JavaScript für Aktionen des rudimentären Bildeditors, der noch nicht verwendet wird.

`YahooUI/treeview/*.js`
: YahooUI JavaScripts zur Darstellung von Verzeichnishierarchien der Mediendatenbank.

`admin/category_selector.js`
: JavaScript zum Erweitern der Kategorie-Auswahl beim Bearbeiten eines Blog-Artikels.

9.5 Template-Optionen und Bulletproof

Seit Serendipity 1.2 ist ein Template namens *Bulletproof* Bestandteil des Serendipity-Pakets. Dieses ist auf Seite 469 detaillierter beschrieben und zielt darauf ab, nach Art eines Baukasten-Prinzips dem Blog-Administrator die Anpassung seines Layouts zu ermöglichen.

Dieses Baukasten-Prinzip wird mittels Template-Optionen umgesetzt. Sobald das *Bulletproof*-Template aktiviert ist, können Sie im Backend im Menüpunkt **Aussehen** → **Styles verwalten** die einzelnen Optionen einstellen. Diese werden ähnlich wie Konfigurationsoptionen eines Plugins dargestellt, pro Zeile stellen Sie eine Option ein.

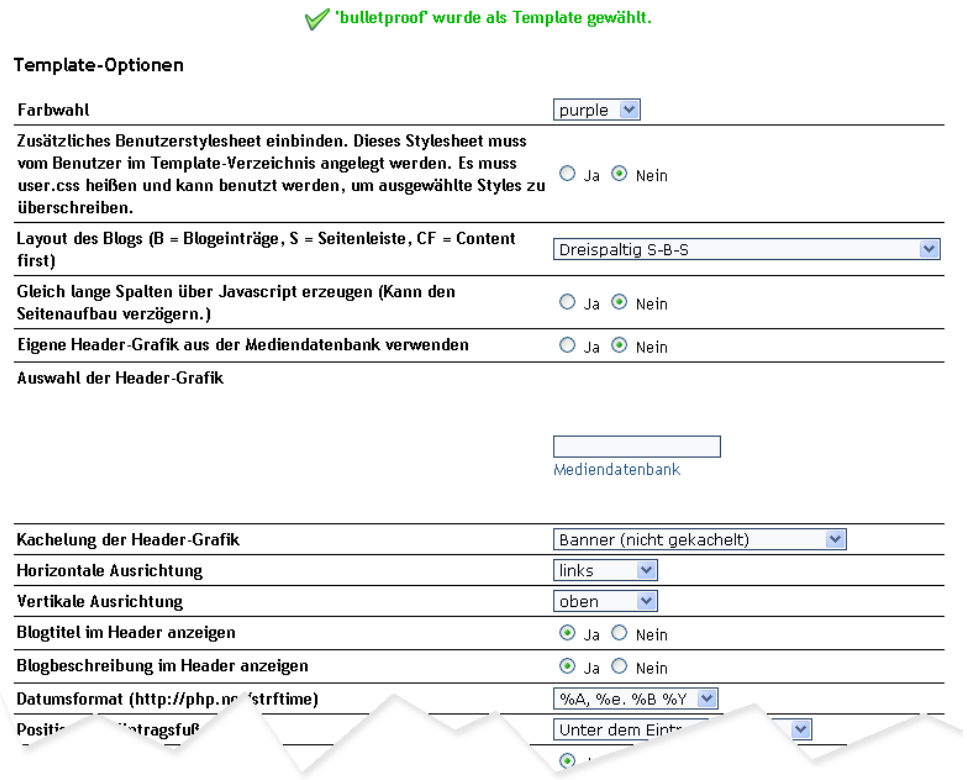

Abbildung 9.1: Ausschnitt der Template-Optionen des Bulletproof-Themes

Wie auf dem Screenshot (Abbildung 9.1) zu erkennen, ist beinahe jeder Aspekt des Templates einstellbar: Wie viele Spalten soll der Inhaltsbereich anbieten, welche Header-Grafik wird eingebunden, wie werden die Datumsangaben formatiert, wo erscheinen Artikelfußzeilen und weitere Angaben.

9.5.1 Farbwahl

Eine Besonderheit stellt das Ausklappfeld für die Option **Farbwahl** dar. Im Hintergrund verwaltet das Bulletproof-Template alle globalen Layout-Optionen über eine zentrale Stylesheet-Datei, die `style.css`. Mittels eines zweiten Stylesheets können separat die Farben und Gestaltungen einiger Elemente detaillierter bestimmt werden. Das `base.css`-Stylesheet wird eingesetzt, um die grundlegende Formatierung des *kugelsicheren* Layouts jeweils kompatibel zu allen Browsern festzulegen. Für einige Problembrowser werden individuelle Stylesheets wie `ie5.css` und `ie6.css` mitgeliefert, die die technischen Besonderheiten für Layout-Umsetzungen abdecken.

Dies hat den großen Vorteil, dass, wenn Sie dem Bulletproof-Framework Ihr eigenes Design aufsetzen wollen, Sie sich nur um die Erstellung eines eigenen Farb-Stylesheets kümmern müssen. In dieser Datei legen Sie Ihre eigenen Farben (und Dinge wie Rahmenstärke und Grafiken) fest, und da Ihre Stylesheet-Datei unabhängig vom Rest des Bulletproof-Frameworks nachgeladen werden kann, können Sie später das komplette Bulletproof-Template aktualisieren, ohne Ihre Anpassungen zu verlieren.

Ihre eigenen Stylesheet-Dateien speichern Sie einfach unter dem Namen *`stylename`*`_style.css` im Verzeichnis `templates/bulletproof/`. Der Teil des Dateinamens `stylename` wird daraufhin automatisch als neue Auswahloption in dem Ausklappfeld **Farbwahl** der Template-Optionen angeboten.

9.5.2 Template-Optionen mittels config.inc.php

Welche Optionen des Weiteren angeboten werden, wird durch die Template-Datei `config.inc.php` festgelegt. In dieser Datei wird eine PHP-Variable `$template_config` definiert. Diese Variable stellt ein mehrdimensionales, verschachteltes Array[5] dar. Für jede Konfigurationsoption wird ein eigenes Unter-Array in der zweiten Ebene definiert, das einige fest vorgeschriebene Array-Schlüssel mit Werten füllt.

Ein Beispiel:

```
<?php
$template_config = array(
  array(
    'var'     => 'name',
    'name'    => 'Ihr Name',
    'type'    => 'string',
    'default' => 'Milla Jovovich'
  ),

  array(
    'var'     => 'ort',
```

[5] Siehe Seite 482.

```
    'name'    => 'Ihr Wohnort',
    'type'    => 'string',
    'default' => 'Remagen'
  )
);
```

Wenn Sie in einem eigenen Template die Konfigurationsoptionen mittels dieser PHP-Variable festlegen, müssen Sie dringend darauf achten, dass die Datei gültigen (*validen, syntaxfehlerfreien*) PHP-Code enthält. Wenn Sie an einer Stelle andere Anführungszeichen verwenden oder ein Komma vergessen, kann dies dazu führen, dass Ihr gesamtes Template nicht angezeigt wird. Lesen Sie daher, falls nötig, die PHP-Syntax-Anweisungen unter `http://de2.php.net/types.array` und `http://de2.php.net/manual/en/language.basic-syntax.php`.

Das Array `$template_config` kann beliebig viele Konfigurationsoptionen deklarieren. Auf diese können Sie später innerhalb jeder Smarty-Template-Datei wie der `index.tpl` über die Smarty-Array-Variable {`$$template_option.`*`optionsname`*} zugreifen. Der `optionsname` entspricht dabei dem Namen, den Sie der jeweiligen Option im Array-Schlüssel `var` gegeben haben (z. B. {`$template_option.name`} oder {`$template_option.ort`}).

Bei der `config.inc.php`-Datei des Bulletproof-Frameworks werden für die Beschreibungen der Konfigurationsoptionen jeweils Sprachkonstanten eingesetzt. Diese werden in den Dateien wie `lang_de.inc.php` festgelegt. Durch deren Verwendung kann die Template-Konfiguration später in beliebigen Übersetzungen eingebunden werden, daher ist der Einsatz von Sprachkonstanten sehr zu empfehlen. Es ist aber auch grundsätzlich möglich, die Beschriftungen direkt in den jeweiligen Array-Wert einzufügen, wie es in den Beispielen dieses Buches zur Vereinfachung auch gemacht wurde.

Bei der Verwendung von Sprachkonstanten sollte man darauf achten, nur notwendige Begriffe zu abstrahieren. Häufig gibt es bereits in der zentralen Serendipity-Sprachdatei (z. B. `lang/serendipity_lang_de.inc.php`) Konstanten mit ähnlichem Inhalt, auf die Sie auch innerhalb eines Templates zugreifen können.

9.5.3 Aufbau des Arrays $template_config

Jedes Unter-Array muss folgende Array-Schlüssel definieren[6]:

[6] Diese Schlüssel entsprechen demselben Schema wie beim Einsatz von Konfigurationsoptionen eines Plugins mittels der `introspect_config_item()`-Methode der Plugin-API.

`var`
: Legt den Namen einer Konfigurationsoption fest. Mit diesem Namen wird der Konfigurationswert in der Datenbank gespeichert, und zugleich wird hier der Name der Option festgelegt, wenn in einem Smarty-Template mittels {`$template_option.`*`var`*} darauf zugegriffen wird.

`name`
: Enthält die Beschriftung einer Template-Option, die in der Konfigurationsoberfläche angezeigt wird.

`type`
: Definiert den Typ einer Konfigurationsoption. Mögliche Typen sind:

  `radio`
  : Stellt mehrere Radio-Buttons dar. Die möglichen Optionen werden über den Array-Schlüssel `radio` angegeben. Die Anzahl der Radio-Buttons pro Zeile wird über den Array-Schlüssel `radio_per_row` angegeben. Beispiel:

```
array(
  'var'           => 'count',
  'name'          => 'Anzahl der Grafiken',
  'type'          => 'radio',
  'radio'         => array('value' => array('5', '10'),
                           'desc'  => array('Fünf', 'Zehn')),
  'radio_per_row' => 2
);
```

  `boolean`
  : Ein Sonderfall der Radio-Button-Konfiguration. Hierbei werden zwei Standardoptionen (`Ja=true` und `Nein=false`) dargestellt. Beispiel:

```
array(
  'var'           => 'showcomments',
  'name'          => 'Kommentare anzeigen',
  'type'          => 'boolean',
  'default'       => 'true',
);
```

  `tristate`
  : Ein Sonderfall der Radio-Button-Konfiguration. Hierbei werden drei fest definierte Felder ausgegeben: `Ja`, `Nein` und `Standard verwenden`. Beispiel:

```
array(
  'var'           => 'popups',
  'name'          => 'Popups verwenden',
  'type'          => 'tristate',
  'default'       => 'default',
);
```

`select`
: Stellt ein Auswahlfeld dar. Die möglichen Optionen werden über den Array-Schlüssel `select_values` angegeben. Beispiel:

```
array(
  'var'           => 'color',
  'name'          => 'Farbauswahl',
  'type'          => 'select',
  'default'       => 'ff0000',
  'select_values' => array(
                        'ff0000' => 'Rot',
                        '00ff00' => 'Grün',
                        '0000ff' => 'Blau'
                     )
);
```

`multiselect`
: Sonderfall des `select`-Typs. Hier wird statt eines Ausklappfelds ein Mehrfach-Ausklappfeld ausgegeben. Alle gewählten Menüfelder werden in der Datenbank als eine Zeichenkette gespeichert, wobei die jeweiligen Werte mit dem Sonderzeichen ^ voneinander getrennt sind. Wenn diese Variable später ausgegeben wird, muss der Inhalt eventuell vorher separiert werden, beispielsweise mittels PHP `explode()`-Funktion. Da dies sehr komplex werden kann, sollte anstelle einer Mehrfach-Auswahlbox eher der Einsatz anderer Konfigurationstypen in Betracht gezogen werden.

  Die Größe des Mehrfach-Ausklappfeldes wird über den Array-Schlüssel `select_size` bestimmt. Beispiel:

```
array(
  'var'            => 'color',
  'name'           => 'Farbauswahl',
  'type'           => 'select',
  'default'        => 'ff0000',
  'select_values'  => array(
                        'ff0000' => 'Rot',
                        '00ff00' => 'Grün',
                        '0000ff' => 'Blau'
                      ),
  'select_size'    => '2'
);
```

`string`
: Stellt ein einfaches, einzeiliges Text-Eingabefeld dar, in das der Benutzer beliebige Inhalte eintragen kann. Beispiel:

```
array(
  'var'          => 'name',
  'name'         => 'Ihre große Liebe heißt...',
  'type'         => 'string',
```

```
  'default'      => 'Emba',
);
```

`text`

Stellt eine große, mehrzeilige Text-Eingabebox dar, in die der Benutzer beliebige Inhalte eintragen kann. Beispiel:

```
array(
  'var'          => 'biographie',
  'name'         => 'Tragen Sie hier Ihren Lebenslauf ein',
  'type'         => 'string',
  'default'      => "Lorem ipsum, und davon 'ne große Menge.",
);
```

`html`

Stellt eine große, mehrzeilige Text-Eingabebox dar, in die der Benutzer beliebige Inhalte eintragen kann. Im Gegensatz zum `text`-Typen wird für dieses Eingabefeld der WYSIWYG-Editor (falls vom Benutzer aktiviert) eingebunden. Beispiel:

```
array(
  'var'       => 'lovesong',
  'name'      => 'Songtext ihres Lieblingsliedes
                  (HTML erlaubt)',
  'type'      => 'string',
  'default'   => '<strong>Wir sind gekommen, um zu bleiben,
                  und bloggen hier nichts mehr weg</strong>'
);
```

`content`

Eine Option des Typs `content` stellt keine direkte Eingabemöglichkeit dar, sondern ermöglicht es, in der Konfigurationsmaske des Serendipity-Backends beliebigen HTML-Code auszugeben. Dies kann helfen, um beispielsweise die Darstellung der Template-Optionen mit eigener HTML-Formatierung anzureichern und optisch ansprechend darzustellen. Da diese Option später keinen vom Benutzer eintragbaren Inhalt enthält, entfällt die Angabe der meisten erforderlichen Array-Schlüssel. Beispiel:

```
array(
  'type'     => 'content',
  'default'  => '<table><tr><th colspan="2">Hinweis</th></tr>
                   <tr><td><img src="/pic.jpg" /></td><td>
                   Isotopp-Sturm voraus.</td></tr></table>'
);
```

`custom`

Falls das Template eine eigene Darstellung der HTML-Formularelemente erreichen will, können Sie den Typ `custom` verwenden und das notwendige HTML-Konstrukt manuell über den Array-Schlüssel `custom` ausgeben. Serendipity trägt dabei automatisch den gespeicherten Inhalt der Konfigurationsoption (anhand

`var`-Schlüssel benannt) in ein `hidden`-Formularfeld ein. Auf dieses Formularfeld kann z. B. mittels JavaScript-Code in Ihrer individuellen Ausgabe über `document.getElementById('config_template_`*var*`')` zugegriffen werden. Beispiel:

```
array(
  'type'     => 'custom',
  'var'      => 'mycustom',
  'custom'   => '<script type="text/javascript">
                $ajax = new AjaxForm();
                $ajax.onUpdateField =
                document.getElementById("config_template_myc
ustom").value;
                $ajax.generate();
                       </script>'
);
```

`hidden`

Stellt ein verstecktes Formularfeld dar. Dieses kann der Benutzer nicht selbständig ändern, daher kann diese Variante gewählt werden, wenn man eine Variable mit fest definiertem Inhalt über Smarty aufrufen möchte. Der Wert wird dabei über den Array-Schlüssel `value` übergeben, im Gegensatz zum sonst üblichen `default`. Beispiel:

```
array(
  'type'            => 'hidden',
  'var'             => 'fixedVar',
  'value'           => $_SERVER['HTTP_HOST']
);
```

`media`

Stellt ein Eingabefeld dar, das der Benutzer mit der URL einer Datei aus der Serendipity-Mediendatenbank ausfüllen kann. Dies ist z. B. sehr hilfreich, wenn für ein Template eine Header-Grafik aus dem Pool der Mediendatenbank gewählt werden soll, ohne dass der Benutzer eine URL eingeben muss. Ein Klick auf einen zusätzlichen Button öffnet dabei die Mediendatenbank.

Zusätzlich zu dem Button wird ein Vorschaubild eingebunden, das das ausgewählte Bild in einem kleinen Bereich darstellt. Die beiden Array-Schlüssel `preview_width` und `preview_height` geben dabei an, wie groß dieses Vorschaubild maximal sein soll (standardmäßig 400 Pixel breit und 100 Pixel hoch). Beispiel:

```
array(
  'type'           => 'media',
  'var'            => 'headerbild',
  'preview_width'  => '200px',
  'preview_height' => '50px',
  'default'      => serendipity_getTemplateFile('header.jpg')
);
```

Hinweis: Als Standardwert wird eine Serendipity-API-Funktion aufgerufen, die den vollständigen Pfad zu der Datei `header.jpg` im aktuellen Template-Verzeichnis zurückliefert.

`seperator`
: Gibt lediglich einen Trennbereich ohne Inhalt aus.

`default`
: Enthält die Standardbelegung einer Variable, wenn der Benutzer noch keinen eigenen Wert eingegeben hat.

`select_values`
: Enthält ein Array mit möglichen Werten zur Befüllung einer Konfigurationsoption vom Typ `select` oder `multiselect`. Der Schlüssel dieses Arrays enthält den Wert, der für das `<option value="...">`-HTML-Feld ausgegeben wird. Der Wert des Arrays enthält die Beschreibung des `<option>...</option>`-Feldes.

`select_size`
: Bestimmt die Größe eines Mehrfach-Auswahlfeldes, falls `type=multiselect`. Standardwert: 1

`radio`
: Enthält ein Array mit möglichen Werten zur Befüllung einer Konfigurationsoption vom Typ `radio`. Abweichend von der Angabe der `select_values` werden hier zwei Array-Schlüssel namens `value` und `desc` vorgegeben, die jeweils ein Array als Wert besitzen müssen. Das Array für den Schlüssel `value` enthält die Werte eines Radio-Buttons, das Array `desc` enthält die Beschriftungen der Radio-Buttons.

`radio_per_row`
: Legt die Anzahl der Radio-Buttons pro Zeile fest, wenn die jeweilige Konfigurationsoption vom Typ `radio` ist. Standardwert: 2.

`preview_width, preview_height`
: Enthält die Vorschaugröße bei der Ausgabe einer Konfigurationsoption vom Typ `media`. Die Angabe erfolgt in einem CSS-kompatiblen Wert wie `400px` oder `1.2em`. Standardwert: `400px` Breite und `100px` Höhe.

Wenn für den Array-Schlüssel `type` keiner der oben aufgeführten Werte eingetragen wurde, wird eine derartige Konfigurationsoption an die Plugin-Schnittstelle weitergegeben. So können eigene Plugins über einen eigenen Event-Hook (`backend_pluginconfig_`*`typwert`* zugreifen[7] und die Ausgabe des notwendigen HTML-Codes für die Konfigurationsoption übernehmen.

[7] Als Werte für `$eventData` wird ein Array mit allen Schlüsseln des jeweiligen Konfigurations-Unter-Arrays übergeben.

9.5.4 Dynamische Template-Optionen

Am Ende der `config.inc.php` benutzt das Bulletproof-Template einen kleinen Trick, um abhängig von den vorher getätigten Konfigurationsoptionen weitere Optionen einzufügen. Konkret werden dabei abhängig von der Option **Anzahl der Links in der Navigationsleiste** jeweils Eingabeboxen für die Beschreibung und URL des jeweiligen Links als Template-Option dargestellt.

Der Trick besteht darin, dass mittels PHP-Code die bisherige Konfiguration des Benutzers eingelesen wird:

```
$template_loaded_config = serendipity_loadThemeOptions($template_config,
$serendipity['smarty_vars']['template_option']);
```

Diese Zeile liest die aktuellen Konfigurationswerte in das PHP-Array `$template_loaded_config` ein. Kurz darauf werden mittels einer `for`-Schleife abhängig von der Anzahl der Navigationslinks pro Link zwei weitere Konfigurationsoptionen in das Array `template_config` eingefügt:

```
$navlinks = array();
for ($i = 0; $i < $template_loaded_config['amount']; $i++) {
    $navlinks[] = array(
        'title' => $template_loaded_config['navlink' . $i . 'text'],
        'href'  => $template_loaded_config['navlink' . $i . 'url']
    );
    $template_config[] = array(
        'var'           => 'navlink' . $i . 'text',
        'name'          => NAV_LINK_TEXT . ' #' . $i,
        'type'          => 'string',
        'default'       => 'Link #' . $i,
        );
    $template_config[] = array(
        'var'           => 'navlink' . $i . 'url',
        'name'          => NAV_LINK_URL . ' #' . $i,
        'type'          => 'string',
        'default'       => '#',
    );
}

$serendipity['smarty']->assign_by_ref('navlinks', $navlinks);
```

Die Variable `$navlinks` liest die bereits konfigurierten Navigationslinks aus und wird am Ende direkt als Smarty-Variable durchgereicht, damit sie innerhalb der `index.tpl`-Datei leicht durchlaufen und ausgegeben werden kann. Der Platzhalter `$i` wird innerhalb der Schleife als fortlaufender Index (beginnend mit der 0) behandelt.

Die Liste aller Bulletproof-Template-Optionen wird innerhalb der Konfigurationsoberfläche ausführlich beschrieben und ist größtenteils bereits selbsterklärend.

9.5.5 Frei definierbare Seitenleisten

In der Plugin-Verwaltung Serendipitys sehen Sie standardmäßig Bereiche für eine linke, eine rechte und eine *versteckte* Seitenleiste. Die meisten Serendipity-Templates unterstützen diese drei Seitenleisten, einige Templates stellen die linke und rechte Seitenleiste jedoch direkt untereinander dar.

Innerhalb der `config.inc.php`-Datei eines Templates kann festgelegt werden, welche und wie viele Seitenleisten-Bereiche das Design anbietet. Danach richtet sich auch die Darstellung der Plugin-Verwaltungsoberfläche. Wenn ein Template vier Seitenleisten (links, rechts, oben, unten) unterstützt, würden in der Oberfläche zum Installieren und Verschieben der Seitenleisten-Plugins auch diese vier Bereiche angezeigt.

Wenn Sie einmal Plugins auf zusätzliche Seitenleisten wie `oben` und `unten` verteilt haben und Ihr Blog später auf ein Template umstellen, das nur die üblichen Seitenleisten unterstützt, wird Serendipity die nicht mehr zugeordneten Seitenleisten-Plugins in einem eigenständigen Container anzeigen. Von dort müssen Sie die Plugins dann wieder in die vorhandenen Seitenleisten einordnen.

Das Template muss die Namen der jeweiligen Seitenleisten über die Variable `$sidebars` in der `config.inc.php` angeben:

```
<?php
$sidebars = 'left,right,hide,oben,unten';
?>
```

An welcher Stelle die jeweiligen Plugins eines Seitenleistenbereichs ausgegeben werden, kontrolliert die Smarty-Template-Datei `index.tpl` wie folgt:

```
<div id="serendipityLeftSideBar">
  {serendipity_printSidebar side="left"}
</div>
<div id="serendipityObenSideBar">
  {serendipity_printSidebar side="oben"}
</div>
```

Die Dokumentation der Smarty-Funktion {`$serendipity_printSidebar`} finden Sie auf Seite 563.

Der Name einer Seitenleiste darf keine Leer- oder Sonderzeichen enthalten und darf maximal 6 Zeichen lang sein.

Falls die Liste der Seitenleisten-Elemente über eine Template-Konfigurationsoption bestimmt werden soll, kann dies anstatt mit der `$sidebars`-Variable auch wie folgt erreicht werden:

```
array(
  'var'            => 'sidebars',
```

```
  'name'           => 'Seitenleisten',
  'type'           => 'select',
  'select_values' => array(
                       'left,right,hide' => 'Standard',
                       'oben,unten' => 'Nur oben, unten',
                       'left,right,hide,oben,unten' => 'Vollständig'
                     )
);
```

Innerhalb der Templates kann {`$template_option.sidebars`} abgefragt werden:

```
{if $template_option.sidebars == 'left,right,hide'}
...
{/if}

{if $template_option.sidebars == 'left,right,hide,oben,unten'}
...
{/if}
```

9.6 Template-Variablen

Innerhalb der Smarty-Template-Dateien können Sie auf zahlreiche Variablen zugreifen, die im jeweiligen Kontext von Serendipity belegt werden. Diese Variablen sind unterteilt in globale Variablen, die Sie innerhalb jeder Smarty-Template-Datei verwenden können, und in Variablen, die nur innerhalb einer speziellen Datei wie z. B. der `commentform.tpl` verfügbar sind.

Abgesehen von den Variablen stehen Ihnen in allen Smarty-Template-Dateien auch besondere, von Serendipity eingebundene Funktionen und Modifier zur Verfügung.

Im Folgenden finden Sie eine detaillierte Übersicht dieser Objekte. Jeder Variable ist zugeordnet, an welcher Stelle im PHP-Code sie zugewiesen wird, von welchem Typ die Variable ist (Zeichenkette, Zahl, eindimensionales Array, mehrdimensionales Array, Objekt oder Boolean[8]) und ob die Variable nur in einer bestimmten Datei abgerufen werden kann.

Wenn die Schlüsselpaare eines mehrdimensionalen Arrays dokumentiert werden, wird die erste Dimension eines Arrays meist beispielhaft erwähnt, z. B. mittels {`$entries.X.timestamp`} . Üblicherweise werden solche verschachtelten Arrays niemals mit einem festen Index (`X`) aufgerufen, sondern immer innerhalb einer Schleife, die dynamisch den Inhalt des Schlüssels `X` einer Variable wie `$index` zuweist. Das heißt, dass man auf eine derartige Variable eher wie folgt zugreifen würde:

[8] Boolesche Werte stehen für `true` (*Ja*) oder `false` (*Nein*).

```
{foreach from=$entries item=$entry key=$index}
  Datum: {$entry.timestamp}
{/foreach}
```

Somit wird das durchnummerierte Unter-Array der Variable {`$entry`} zugewiesen und der laufende Index X würde durch die Variable `$index` abgedeckt. Auf den eigentlichen Unterschlüssel `timestamp` wird so nur über das zugewiesene Array mittels {`$entry.timestamp`} zugegriffen.

9.6.1 Globale Variablen

Die Liste der folgenden Variablen gilt für alle Template-Dateien. Die Ausgabe der Templates erfolgt ausgehend von der `index.tpl`. Weiterführende Inhalte werden mittels der Variable {`$CONTENT`} über die Datei `content.tpl` eingebunden. Innerhalb des `content.tpl`-Templates werden Artikellisten über die Variable {`$ENTRIES`} mittels der Datei `entries.tpl` geparst.

{`$admin_view`}
: *Variablentyp:* Zeichenkette
Verfügbar in: `admin/entries.tpl, admin/index.tpl`
Zugewiesen in: `include/functions_entries_admin.inc.php`
Für die Template-Dateien des Backends ist diese Variable auf einen Wert gesetzt, der den Bereich angibt, innerhalb dessen das Template ausgegeben wird. Derzeit wird nur der Wert `entryform` (für die Ausgabe des Artikel-Editors) unterstützt, zukünftige können folgen, sobald weitere Bereiche des Backends auf Smarty-Templates umgesetzt werden.

{`$admin_vars`}
: *Variablentyp:* Eindimensionales Array
Verfügbar in: `admin/entries.tpl, admin/index.tpl`
Zugewiesen in: `serendipity_admin.php`
Enthält einige Variablen mit Werten, die für das Backend benötigt werden. Folgende Array-Schlüssel sind verfügbar:

  {`$admin_vars.out`}
  : *Variablentyp:* Eindimensionales Array
  Kann zusätzliche HTML-Ausgaben von Plugins enthalten, die bei der Darstellung des Login-Fensters zum Backend-Bereich eingebunden werden können.

  {`$admin_vars.no_create`}
  : *Variablentyp:* Boolean
  Bei eingeloggten Benutzern, die keine Rechte haben, Einträge zu erstellen (siehe Seite 185), enthält diese Variable den Wert `true`. In diesem Fall werden viele der Menüpunkte des Backends ausgeblendet.

{$admin_vars.version_info}
: *Variablentyp:* Zeichenkette
Enthält die HTML-Ausgabe der aktuellen Serendipity- und PHP-Version für den Footer des Backends.

{$admin_vars.css_file}
: *Variablentyp:* Zeichenkette
Enthält die URL zu der CSS-Datei (`style.css`), die das Stylesheet für das Backend einbindet.

{$admin_vars.admin_css_file}
: *Variablentyp:* Zeichenkette
Enthält die URL zu der CSS-Datei (`pluginmanager.css`), die ein Stylesheet für die Plugin-Verwaltung des Backends einbindet.

{$admin_vars.main_content}
: *Variablentyp:* Zeichenkette
Enthält den HTML-Code für den Inhaltsbereich des Backends. Der Inhalt aller Funktionen bis auf die Erstellung/Bearbeitung eines Artikels erfolgt nicht mittels Smarty-Templates, sondern mit fest eingebautem PHP-Code, und kann daher nur als zentrale Variable in das Grundgerüst der `admin/index.tpl`-Datei eingebunden werden.

{$admin_vars.no_banner}
: *Variablentyp:* Boolean
Gibt an, ob das Kopfbild im Backend-Bereich eingebunden werden soll. Mittels der URL-Variable `serendipity_admin.php?serendipity[noBanner]=true` können Plugins oder andere Teile des Backends bestimmen, ob dieser Bereich aktiviert oder deaktiviert werden soll. Wenn {$admin_vars.no_banner} den Wert `true` enthält, wird das Kopfbild nicht eingebunden.

{$admin_vars.no_sidebar}
: *Variablentyp:* Boolean
Gibt an, ob das Menü der Seitenleiste im Backend-Bereich eingebunden werden soll. Mittels der URL-Variable `serendipity_admin.php?serendipity[noBanner]=true` können Plugins oder andere Bestandteile des Backends bestimmen, ob dieses Menü aktiviert oder deaktiviert werden soll. Enthält {$admin_vars.no_sidebar} den Wert `true`, werden alle Menüpunkte versteckt.

{$admin_vars.post_action}
: *Variablentyp:* Zeichenkette
Enthält die URL-Variable `$serendipity['POST']['action']`, um dessen Inhalt innerhalb des Templates prüfen zu können. Diese Variable enthält Anweisungen, welche Aktion im Backend gerade ausgeführt wird.

{$admin_vars.is_logged_in}

Variablentyp: Boolean
Enthält den Wert `true`, wenn der aktuelle Benutzer bereits eingeloggt ist.

{`$admin_vars.admin_installed`} ,

{`$admin_vars.use_installer`} *Variablentyp:* Boolean
Enthält den Wert `true`, wenn Serendipity bereits installiert wurde. Ist dieser Wert nicht gesetzt, wird die Installationsroutine Serendipitys aufgerufen.

{`$admin_vars.self_info`}
Variablentyp: Zeichenkette
Enthält eine HTML-Ausgabe, die angibt, welchen Benutzerrang und Namen der aktuell eingeloggte Benutzer hat.

{`$is_preview`}
Variablentyp: Boolean
Verfügbar in: `*.tpl (Frontend)`
Zugewiesen in: `include/functions_config.inc.php, serendipity_iframe()`
Enthält den Wert `true`, falls die Vorschau eines einzelnen Artikels im Design des Frontends angezeigt werden soll. Die Vorschau wird im Backend eingebettet.

{`$CONST`}
Variablentyp: Eindimensionales Array
Verfügbar in: `*.tpl`
Zugewiesen in: `lang/(UTF-8/)serendipity_lang_XX.inc.php`
Das Smarty-Array {`$CONST`} ist nur eine Umleitung, um den Zugriff auf Sprachkonstanten zu ermöglichen. Diese kapselt Smarty sonst üblicherweise in einem {`$smarty.const`} -Array. Um Tipparbeit zu vermeiden (und aus Gründen der Rückwärtskompatibilität) nutzt Serendipity stattdessen nur {`$CONST`} . Die Array-Schlüssel entsprechen dabei dem jeweiligen Namen der Sprachkonstante, der in einer Sprachdatei wie `lang/serendipity_lang_de` festgelegt wird. Dargestellte Meldungen sollten in einer Smarty-Template-Datei niemals direkt eingebunden werden, sondern immer über das Einfügen von Sprachkonstanten (notfalls mittels eigener `lang_de.inc.php`-Datei, wie auf Seite 644 und 493 beschrieben).

{`$startpage`}
Variablentyp: Boolean
Verfügbar in: `*.tpl (Frontend)`
Zugewiesen in: `include/genpage.inc.php`
Enthält den Wert `true`, wenn Serendipity die Startseite des Blogs darstellt.

{`$uriargs`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl (Frontend)`
Zugewiesen in: `include/genpage.inc.php`
Enthält eine Liste der URL-Pfadkomponenten der aktuellen Seite.

{$head_link_stylesheet}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält die URL zur zentralen Stylesheet-Datei.

{$head_charset}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den aktuell konfigurierten Zeichensatz der HTML-Ausgabe (meist UTF-8 oder ISO-8859-1).

{$head_version} , {$serendipityVersion}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält die Serendipity-Versionsnummer.

{$head_title}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den Text, der im Titelbereich des Blogs angezeigt wird. Dieser kann vom konfigurierten Titel des Blogs abweichen, wenn beispielsweise die Detailansicht eines Artikels angezeigt wird.

{$head_subtitle}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den Text, der im Untertitelbereich des Blogs angezeigt wird. Dieser kann vom konfigurierten Untertitel des Blogs abweichen, wenn beispielsweise die Detailansicht eines Artikels angezeigt wird. In diesem Fall wird der Titel des Blogs standardmäßig als Untertitel eingebunden.

{$blogTitle}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den Titel des Blogs, wie er in der Konfiguration angegeben wurde.

{$blogSubTitle}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl

Zugewiesen in: `include/functions_smarty.inc.php`
Enthält den Untertitel des Blogs, wie er in der Konfiguration angegeben wurde.

{`$blogDescription`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Enthält die Beschreibung des Blogs, wie sie in der Konfiguration angegeben wurde.

{`$is_xhtml`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn Serendipity XHTML-Ausgaben produzieren soll. In älteren Serendipity-Versionen war diese Variable konfigurierbar, mittlerweile ist sie immer fest auf `true` gesetzt.

{`$use_popups`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn in der Serendipity-Konfiguration die Benutzung von Popups erlaubt wurde.

{`$is_embedded`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn in der Serendipity-Konfiguration die eingebettete Nutzung aktiviert wurde. In so einem Fall geben Templates keinen HTML-Kopf- und -Fußbereich aus.

{`$is_raw_mode`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn ein altes Serendipity-Template (mit `layout.php`-Datei) verwendet wird und keine Smarty-Dateien ausgewertet werden müssen.

{`$is_logged_in`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn ein eingeloggter Redakteur das Frontend aufruft.

{`$entry_id`}
Variablentyp: Zahl
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Kann die ID eines Blog-Artikels enthalten, wenn gerade die Detailansicht eines Artikels aufgerufen wird.

{`$is_single_entry`}
Variablentyp: Boolean
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Ist auf `true` gesetzt, wenn die Detailansicht eines Artikels aufgerufen wird.

{`$serendipityHTTPPath`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Enthält den HTTP-Pfad zum Serendipity-Blog.

{`$serendipityBaseURL`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Enthält die vollständige URL zum Serendipity-Blog.

{`$serendipityRewritePrefix`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Falls im Blog keine URL-Umformung (siehe Seite 168) verfügbar ist, enthält diese Variable ein Präfix (`index.php?/`), das vor jede URL gesetzt wird, um aussagekräftige URLs zu imitieren.

{`$serendipityIndexFile`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Enthält den Namen der Index-Datei zur Darstellung des Blogs, wie in der Serendipity-Konfiguration festgelegt. Beim Einsatz der eingebetteten Nutzung (siehe Seite 679) zeigt diese Variable meist auf eine eigene Wrapper-Datei.

{`$lang`}
Variablentyp: Zeichenkette
Verfügbar in: `*.tpl`
Zugewiesen in: `include/functions_smarty.inc.php`
Enthält das Kürzel der aktuell gewählten Sprachdatei (z. B. `de`, `en` usw.).

{$category}
Variablentyp: Zahl
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Falls eine Kategorie-Übersicht dargestellt wird, enthält diese Variable die ID dieser Kategorie.

{$category_info}
Variablentyp: Eindimensionales Array
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Falls eine Kategorie-Übersicht dargestellt wird, enthält diese Variable ein Array mit den Metadaten der Kategorie. Die möglichen Arrayschlüssel sind in der Liste auf Seite 530 aufgeführt.

{$template}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den Namen des Template-Verzeichnisses, das zur Darstellung verwendet wird.

{$dateRange}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Enthält den Wert der $serendipity['range']-Variable. Diese legt den Zeitraum fest, der für die Auswahl der dargestellten Artikel ausgewertet wird.

{$template_option}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl
Zugewiesen in: include/functions_smarty.inc.php
Falls das aktuell gewählte Template über Konfigurationsoptionen verfügt, sind die gespeicherten Werte über die Variable {$template_option} zugänglich. Die jeweils vorhandenen Arrayschlüssel müssen der config.inc.php-Datei des Templates entnommen werden; siehe auch Seite 499.

{$view}
Variablentyp: Zeichenkette
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: include/genpage.inc.php
Gibt an, welchen Seiteninhalt der Benutzer angefordert hat. Abhängig davon werden unterschiedliche Template-Dateien eingelesen. Die Template-Dateien können diese Variable prüfen, um festzustellen, welche Art Ausgaben angefordert werden.

Mögliche Werte sind:

`404`
: für die Ausgabe einer Übersicht, wenn die aktuelle URL ungültig ist.

`plugin`
: wenn ein Ereignis-Plugin (wie z. B. die *statischen Seiten*) die Ausgabe übernimmt.

`feed`
: wenn die RSS-Feeds dargestellt werden.

`archives`
: wenn Übersichtsseiten zu bestimmten Monaten, Tagen, Jahren dargestellt werden.

`archive`
: wenn die kurzen Übersichtsseiten für Monate oder Jahre dargestellt werden.

`entry`
: wenn eine Artikel-Detailseite dargestellt wird.

`admin`
: wenn das Backend dargestellt wird.

`categories`
: wenn Übersichtsseiten zu einer Kategorie angezeigt werden.

`authors`
: wenn Übersichtsseiten zu einem speziellen Autor angezeigt werden.

`search`
: wenn Ergebnisse der Volltextsuche angezeigt werden.

`css`
: wenn das Stylesheet ausgegeben wird.

`comments`
: wenn Übersichtsseiten von Kommentaren dargestellt werden.

`start`
: wenn die Startseite dargestellt wird.

{`$content_message`}
: *Variablentyp:* Zeichenkette
Verfügbar in: `*.tpl (Frontend)`
Zugewiesen in: `include/genpage.inc.php`
Enthält etwaige Meldungen (inklusive HTML-Code), die auf der Übersichtsseite dargestellt werden, wenn z. B. die Volltextsuche ausgeführt wurde.

{`$searchresult_error`}
: *Variablentyp:* Boolean
Verfügbar in: `*.tpl (Frontend)`
Zugewiesen in: `include/genpage.inc.php`
Wenn die Volltextsuche nicht ausgeführt werden konnte, enthält diese Variable den Wert `true`.

{$searchresult_noEntries}
: *Variablentyp:* Boolean
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: include/genpage.inc.php
Wenn die Volltextsuche ausgeführt wurde, aber keine Ergebnisse gefunden wurden, enthält diese Variable den Wert true.

{$searchresult_results}
: *Variablentyp:* Boolean
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: include/genpage.inc.php
Wenn die Volltextsuche ausgeführt wurde und mindestens ein Ergebnis vorliegt, enthält diese Variable den Wert true.

{$raw_data}
: *Variablentyp:* Zeichenkette
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: index.php
Standardmäßig werden alle Serendipity-Inhalte mittels Smarty-Funktionen und Smarty-Templates zusammen gebündelt und dann als vollständige HTML-Seite ausgegeben. Dennoch kann es passieren, dass einige Ereignis-Plugins rohe HTML-Ausgaben innerhalb ihrer PHP-Struktur ausgeben. Diese sind zur Smarty-Ausgabe inkompatibel; sie werden daher mittels PHP *Output-Buffering*-Funktionalität zusammen gebündelt und letztendlich innerhalb der Smarty-Variable {$raw_data} gespeichert und ausgegeben.

{$leftSideBarElements}, {$rightSideBarElements}
: *Variablentyp:* Zahl
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: include/genpage.inc.php
Enthält die Anzahl an Seitenleisten-Plugins für die linke und rechte Seitenleiste.

{$CONTENT}
: *Variablentyp:* Zeichenkette
Verfügbar in: *.tpl (Frontend)
Zugewiesen in: include/genpage.inc.php
Enthält den geparsten HTML-Code des content.tpl-Templates.

9.6.2 Dateispezifische Variablen

In dieser Liste werden nur die vom Serendipity-Kern verwendeten Variablen ausgeführt. Zusätzlich können in den einzelnen Template-Dateien auch abhängig von den eingesetzten Ereignis-Plugins weitere Variablen zur Verfügung stehen. Falls ein Plugin weitere Variablen einbindet, wird darauf separat in der Beschreibung der einzelnen Ereignis-Plugins hingewiesen, wie beispielsweise beim *Freie Artikel-Tags*-Plugin auf Seite 399.

entries.tpl, entries_summary.tpl

Die Template-Datei `entries.tpl` ist die zentralste Datei für die Darstellung von Blog-Artikeln im Frontend. Alle für diese Datei aufgeführten Variablen sind zwar grundsätzlich auch innerhalb anderer Template-Dateien (`index.tpl, content.tpl`) verfügbar, machen aber üblicherweise nur im Kontext der Datei `entries.tpl` Sinn.

{`$footer_prev_page`}
: *Variablentyp:* Zeichenkette
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Enthält die URL zur vorigen Seite innerhalb einer Artikelübersicht. Wird innerhalb der Fußzeile dargestellt.

{`$footer_next_page`}
: *Variablentyp:* Zeichenkette
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Enthält die URL zur nächsten Seite innerhalb einer Artikelübersicht. Wird innerhalb der Fußzeile dargestellt.

{`$footer_totalEntries`}
: *Variablentyp:* Zahl
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Bei Archivübersichten, die sich über mehr als eine Seite erstrecken, enthält diese Variable die Anzahl der insgesamt blätterbaren Einträge.

{`$footer_totalPages`}
: *Variablentyp:* Zahl
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Bei Archivübersichten, die sich über mehr als eine Seite erstrecken, enthält diese Variable die Anzahl der insgesamt verfügbaren Seiten.

{`$footer_currentPage`}
: *Variablentyp:* Zahl
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Bei Archivübersichten, die sich über mehr als eine Seite erstrecken, enthält diese Variable die aktuelle Seitennummer.

{`$footer_pageLink`}
: *Variablentyp:* Zeichenkette
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFooter()`
Enthält eine generische URL, die verwendet werden kann, um Links zu beliebigen Folgeseiten zu formatieren. Die URL enthält einen Platzhalter `%s`, der durch die gewünschte Zielseitennummer ersetzt werden kann.

{`$footer_info`}
: *Variablentyp:* Zeichenkette

Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntryFoote`
Bei Archivübersichten, die sich über mehr als eine Seite erstrecken, enthält diese Variable einen Satz, der darüber informiert, wie viele Ergebnisse und Seiten es insgesamt gibt, z. B.: **Seite 1 von 10, insgesamt 20 Einträge**.

{`$plugin_clean_page`}
Variablentyp: Boolean
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntries()`
Ereignis-Plugins können für die Ausgabe das Layout der Artikelübersicht übernehmen und darin ihre eigenen Inhalte einbinden. Die Variable {`$plugin_clean_page`} enthält den Wert `true`, wenn ein Plugin sozusagen eine *saubere Seite* ohne Artikel ausgeben möchte.

{`$comments_messagestack`}
Variablentyp: Eindimensionales Array
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntries()`
Enthält ein Array mit etwaigen Fehlermeldungen, die bei der Erstellung eines Kommentars auftreten können (fehlende Angaben, Spam-Erkennung). Pro Array-Schlüssel ist eine Fehlermeldung aufgeführt, die HTML-Code enthalten kann.

{`$is_comment_added`}
Variablentyp: Boolean
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntries()`
Enthält den Wert `true`, wenn ein Kommentar innerhalb des Seitenaufrufs erfolgreich hinzugefügt wurde.

{`$is_comment_moderate`}
Variablentyp: Boolean
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntries()`
Enthält den Wert `true`, wenn ein Kommentar innerhalb des Seitenaufrufs erfolgreich hinzugefügt wurde, aber aufgrund der aktivierten Kommentarmoderation nicht direkt freigeschaltet wird.

{`$entries`}
Variablentyp: Mehrdimensionales Array
Zugewiesen in: `include/functions_entries.inc.php, serendipity_printEntries()`
Enthält das Array mit den Metadaten und Eigenschaften eines Artikels. Die erste Dimension des Arrays enthält das gruppierende Datum, so dass im Wert des Arrays für einen Tag mehrere Artikel enthalten sein können. Dies ermöglicht später bei der Ausgabe der Variable {`$entries`} , dass dasselbe Datum nur einmal ausgegeben wird und darunter mehrere Artikel erscheinen können.

Der Array-Schlüssel enthält das Datum dabei in Form eines Unix-Timestamps (bzw. den Wert `sticky` für dauerhafte Einträge). Die Unter-Arrays enthalten die im Folgenden aufgeführten Schlüssel.
Standardmäßig werden diese Unter-Arrays über die Template-Datei `entries.tpl`

mittels einer {`foreach`}-Schleife der Smarty-Variable {`$entry`} zugewiesen, so dass Sie darauf nicht mittels {`$entries.X.entries.title`} zugreifen, sondern mit {`$entry.title`} .

{`$entries.X.date`}
: *Variablentyp:* Zahl
Enthält das Datum des ältesten Artikels für eine Datumsgruppe (einen Tag).

{`$entries.X.is_sticky`}
: *Variablentyp:* Boolean
Falls ein Eintrag *dauerhaft* (siehe Plugin-Beschreibung Seite 263) ist, wird er im Array {`$entries.sticky`} abgelegt; für alle Einträge dieser Gruppe ist {`$entries.sticky.is_sticky`} auf `true` gesetzt. Die Artikel anderer Datumsgruppen besitzen für diese Variable den Wert `false`.

{`$entries.X.entries`}
: *Variablentyp:* Mehrdimensionales Array
Dieses Array enthält die Daten der eigentlichen Blog-Artikel. Der Schlüssel dieses Arrays ist durchnummeriert, jeder Wert enthält ein weiteres Unter-Array mit den Inhalten der Datenbank-Spalten und folgenden Schlüsseln.

{`$entries.X.entries.Y.id`}
: *Variablentyp:* Zahl
Enthält die eindeutige ID eines Artikels.

{`$entries.X.entries.Y.title`}
: *Variablentyp:* Zeichenkette
Enthält den Titel des Artikels. Sonderzeichen wie Umlaute und spitze Klammern sind in ihre jeweiligen HTML-Sonderzeichen konvertiert.

{`$entries.X.entries.Y.html_title`}
: *Variablentyp:* Zeichenkette
Enthält den Titel des Artikels. Etwaige Sonderzeichen werden im Gegensatz zu {`$entries.X.entries.Y.title`} nicht konvertiert.

{`$entries.X.entries.Y.body`}
: *Variablentyp:* Zeichenkette
Enthält den Artikeltext.

{`$entries.X.entries.Y.extended`}
: *Variablentyp:* Zeichenkette
Enthält den erweiterten Artikeltext.

{`$entries.X.entries.Y.timestamp`}
: *Variablentyp:* Zahl
Enthält das Veröffentlichungsdatum des Artikels (in Unix-Sekunden).

{`$entries.X.entries.Y.last_modified`}
: *Variablentyp:* Zahl
Enthält das Datum der letzten Bearbeitung des Artikels (in Unix-Sekunden).

{$entries.X.entries.Y.isdraft}
Variablentyp: Boolean
Enthält den Wert `true`, falls der Artikel ein Entwurf ist. Hinweis: Entwürfe können nur als Vorschau angesehen werden, da sie niemals im Frontend eingebunden werden.

{$entries.X.entries.Y.is_cached}
Variablentyp: Boolean
Enthält den Wert `true`, wenn der Inhalt eines Artikels aus dem Cache geholt wird. Dies ist nur dann der Fall, wenn das Plugin **Erweiterte Eigenschaften von Artikeln** verwendet wird und die Caching-Option aktiviert wurde (siehe Seite 262).

{$entries.X.entries.Y.is_entry_owner}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Autor des Artikels dem aktuell eingeloggten Besucher des Frontends entspricht. Ist dies der Fall, können Links zum direkten Bearbeiten eines Artikels im Frontend eingeblendet werden.

{$entries.X.entries.Y.plugin_display_dat}
Variablentyp: Zeichenkette
Enthält etwaigen HTML-Code, der durch Ereignis-Plugins dem Eintrag hinzugefügt werden soll. Mehrere Plugins können ihre individuellen Inhalte nacheinander in diese Variable speichern. Die Reihenfolge der Ausgabe entspricht dabei der Reihenfolge der Ereignis-Plugins.

{$entries.X.entries.Y.has_extended} ,

{$entries.X.entries.Y.exflag} *Variablentyp:* Boolean
Ist auf `true` gesetzt, wenn der Artikel über einen erweiterten Eintrag verfügt.

{$entries.X.entries.Y.is_extended}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Besucher die Detailansicht eines Artikels aufgerufen hat.

{$entries.X.entries.Y.viewmode}
Variablentyp: Zeichenkette
Die Art der Darstellung von Kommentaren zu einem Artikel ist abhängig davon, ob der Besucher auf den Link zur **Linearen** oder **Verschachtelten** Ansicht geklickt hat. Standardmäßig wird eine verschachtelte Ansicht dargestellt, bei der Kommentare, die sich auf einen vorhergehenden beziehen, direkt untereinander verschachtelt dargestellt werden. Die lineare Ansicht zeigt die Kommentare in ihrer Eingangsreihenfolge.

Die Variable enthält entweder den Wert der Konstante `VIEWMODE_LINEAR` oder `VIEWMODE_THREADED`, je nach Entscheidung des Besuchers.

{`$entries.X.entries.Y.comments`}
Variablentyp: Zahl
Enthält die Anzahl von Kommentaren zu einem Artikel.

{`$entries.X.entries.Y.trackbacks`}
Variablentyp: Zahl
Enthält die Anzahl von Trackbacks zu einem Artikel.

{`$entries.X.entries.Y.allow_comments`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Autor eines Artikels die Option aktiviert hat, dass Kommentare hinterlassen werden können.

{`$entries.X.entries.Y.moderate_comments`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Autor eines Artikels die Option aktiviert hat, dass Kommentare vor deren Darstellung freigeschaltet werden müssen.

{`$entries.X.entries.Y.has_comments`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn ein Artikel mindestens einen Kommentar besitzt.

{`$entries.X.entries.Y.has_trackbacks`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn ein Artikel mindestens einen Trackback besitzt.

{`$entries.X.entries.Y.label_comments`}
Variablentyp: Zeichenkette
Enthält jeweils die zugehörige Bezeichnung, die neben der Anzahl von Kommentaren dargestellt wird. Wenn ein Artikel nur einen Kommentar besitzt, ist der Wert der Variable `Kommentar`, andernfalls immer `Kommentare`. Der tatsächliche Wert dieser Variable richtet sich nach der eingerichteten Sprache des Blogs und verwendet die Sprachkonstante `COMMENT` bzw. `COMMENTS`.

{`$entries.X.entries.Y.label_trackbacks`}
Variablentyp: Zeichenkette
Enthält jeweils die zugehörige Bezeichnung, die neben der Anzahl von Trackbacks dargestellt wird. Wenn ein Artikel nur einen Trackback besitzt, ist der Wert dieser Variable `Trackback`, andernfalls immer `Trackbacks`. Der tatsächliche Wert dieser Variable richtet sich nach der Sprache des Blogs und verwendet die Sprachkonstante `TRACKBACK` bzw. `TRACKBACKS`.

{`$entries.X.entries.Y.author`}
Variablentyp: Zeichenkette
Enthält den Namen des Autors eines Artikels.

{`$entries.X.entries.Y.loginname`}
Variablentyp: Zeichenkette
Enthält den Login-Namen des Autors eines Artikels. Aus Sicherheitsgründen sollte dieser nicht öffentlich dargestellt werden.

{$entries.X.entries.Y.email}
Variablentyp: Zeichenkette
Enthält die E-Mail-Adresse des Autors eines Artikels.

{$entries.X.entries.Y.authorid}
Variablentyp: Zahl
Enthält die ID des Autors eines Artikels.

{$entries.X.entries.Y.title_rdf}
Variablentyp: Zeichenkette
Enthält den Titel des Artikels (mit konvertierten Sonderzeichen), bei dem jedoch etwaige doppelte Bindestriche durch einen einzelnen Bindestrich ersetzt werden, damit der Inhalt dieser Variable für XML-RDF-Metadaten verwendet werden kann.

{$entries.X.entries.Y.rdf_ident}
Variablentyp: Zeichenkette
Enthält einen eindeutigen Permalink des Artikels, dessen URL vollständig mit HTTP-Hostname angegeben wird.

{$entries.X.entries.Y.link}
Variablentyp: Zeichenkette
Enthält die relative URL zu der Detailseite eines Artikels (z. B. /serendipity/archives/1-Ti

{$entries.X.entries.Y.commURL}
Variablentyp: Zeichenkette
Enthält die vollständige URL zu der Detailseite eines Artikels, inklusive HTTP-Hostname (z. B. http://www.example.com/serendipity/archives/1-Titel.html).

{$entries.X.entries.Y.link_rdf}
Variablentyp: Zeichenkette
Enthält einen eindeutigen, gekürzten Permalink des Artikels für die Identifikation innerhalb eines XML-RDF-Metadatenblocks.

{$entries.X.entries.Y.link_deny_comments}
Variablentyp: Zeichenkette
Enthält die URL, damit der Autor eines Artikels auf der Detailseite das Hinzufügen von Kommentaren zukünftig verbieten kann.

{$entries.X.entries.Y.link_allow_comments}
Variablentyp: Zeichenkette
Enthält die URL, damit der Autor eines Artikels auf der Detailseite das Hinzufügen von Kommentaren wieder erlauben kann.

{$entries.X.entries.Y.link_popup_comments}
Variablentyp: Zeichenkette
Enthält eine URL, unter der die Kommentare eines Artikels separat (z. B. als Popup) angezeigt werden können.

{$entries.X.entries.Y.link_popup_trackbacks}
Variablentyp: Zeichenkette

Enthält eine URL, unter der die Trackbacks eines Artikels separat (z. B. als Pop-up) angezeigt werden können.

{`$entries.X.entries.Y.link_edit`}
Variablentyp: Zeichenkette
Enthält eine URL, mit der der Autor eines Artikels direkt ins Backend gelangen kann, um den jeweiligen Artikel dort zu bearbeiten.

{`$entries.X.entries.Y.link_trackback`}
Variablentyp: Zeichenkette
Enthält die Trackback-URL zu einem Artikel. An diese Ziel-URL müssen fremde Blogsysteme ihr Trackback senden. Diese URL ist nicht zum üblichen Aufruf innerhalb des Browsers konzipiert.

{`$entries.X.entries.Y.link_viewmode_threaded`}
Variablentyp: Zeichenkette
Enthält den Link zur Detailseite eines Artikels, um die Kommentare in verschachtelter Ansicht darzustellen.

{`$entries.X.entries.Y.link_viewmode_linear`}
Variablentyp: Zeichenkette
Enthält den Link zur Detailseite eines Artikels, um die Kommentare in linearer (chronologischer) Ansicht darzustellen.

{`$entries.X.entries.Y.link_author`}
Variablentyp: Zeichenkette
Enthält eine URL zur Übersichtsseite aller Beiträge des Autors, der den aktuellen Artikel verfasst hat.

{`$entries.X.entries.Y.categories`}
Variablentyp: Mehrdimensionales Array
Enthält ein Array mit allen dem Artikel zugewiesenen Kategorien. Die Schlüssel der durchnummerierten Unter-Arrays entsprechen denen, die auf Seite 530 aufgeführt sind. Hinzu kommt der Array-Schlüssel {`$entries.X.entries.Y.categories.Z.category_link`} , der eine URL zu einer Übersichtsseite enthält, auf der alle Artikel der jeweiligen Kategorie aufgeführt sind.

{`$entries.X.entries.Y.properties`}
Variablentyp: Mehrdimensionales Array
Das Array {`$entries.X.entries.Y.properties`} enthält alle **Erweiterten Eigenschaften**, die mittels diverser Plugins einem Artikel zugeschrieben werden können. Die zusätzlich verfügbaren Schlüssel können Sie in der Dokumentation der jeweiligen Plugins nachschlagen, beispielsweise für das Plugin **Erweiterte Eigenschaften für Artikel** auf Seite 262. Unter anderem können auch beliebige selbstdefinierte Array-Schlüssel in diesem Unter-Array aufgeführt werden, wobei der jeweilige Schlüssel wie {`$entries.X.entries.Y.properties.ep_meinSchluessel`} (bzw. {`$entry.properties.ep_meinSchluessel`}) eingebunden wird.

feed_*.tpl

Die Darstellung der RSS-Feeds erfolgt über die Smarty-Template-Dateien wie `feed_2.0.tpl`. Folgende Variablen sind dort verfügbar:

{`$metadata`}
: *Variablentyp:* Eindimensionales Array
Zugewiesen in: `rss.php`
Enthält eine Liste von Metadaten, die den Typ des RSS-Feeds sowie globale Blog-Daten angibt. Folgende Array-Schlüssel sind verfügbar:

  {`$metadata.title`}
  : *Variablentyp:* Zeichenkette
  Enthält den Titel des RSS-Feeds (setzt sich aus gewählter Kategorie, Autoren und Name des Blogs zusammen).

  {`$metadata.description`}
  : *Variablentyp:* Zeichenkette
  Enthält eine Beschreibung des RSS-Feeds (abhängig von gewählter Kategorie, Autoren und Beschreibung des Blogs).

  {`$metadata.language`}
  : *Variablentyp:* Zeichenkette
  Enthält die Sprache des Blogs, z. B. `de` für ein deutschsprachiges Blog.

  {`$metadata.link`}
  : *Variablentyp:* Zeichenkette
  Enthält die URL zum Blog.

  {`$metadata.email`}
  : *Variablentyp:* Zeichenkette
  Enthält die konfigurierte E-Mail-Adresse des Blogs.

  {`$metadata.fullFeed`}
  : *Variablentyp:* Boolean
  Ist auf `true` gesetzt, wenn der RSS-Feed Artikel vollständig einbinden soll. Wird beeinflusst durch das Plugin *Blog abonnieren* (siehe Seite 207).

  {`$metadata.showMail`}
  : *Variablentyp:* Boolean
  Ist auf `true` gesetzt, wenn E-Mail-Adressen gültig eingebunden werden sollen. Steht der Wert auf `false`, werden E-Mail-Adressen verschleiert. Diese Variable wird beeinflusst durch die Konfiguration des Anti-Spam-Plugins (siehe Seite 239).

  {`$metadata.version`}
  : *Variablentyp:* Zeichenkette
  Enthält die Versionsnummer des angeforderten Typs (z. B. `2.0` für RSS 2.0, oder `atom1.0` für Atom 1.0).

{`$metadata.template_file`}
Variablentyp: Zeichenkette
Enthält den Namen der Smarty-Template-Datei, die für den jeweiligen Feed verwendet wurde.

{`$metadata.additional_fields`}
Variablentyp: Eindimensionales Array
Enthält etwaige Zusatzfelder, die durch Plugins hinzugefügt werden. Das Plugin *Blog abonnieren* bindet Werte für folgende Array-Schlüssel ein:
{`$metadata.additional_fields.image`} ,
{`$metadata.additional_fields.image_atom1.0`} ,
{`$metadata.additional_fields.image_rss1.0_channel`} ,
{`$metadata.additional_fields.image_rss1.0_rdf`} (Feed-Bilder für die unterschiedlichen Versionen der Feed-Typen) und
{`$metadata.additional_fields.channel`} (Liste zusätzlicher XML-Elemente wie `webMaster`, `ttl`, `pubDate`).

{`$entries`}
Variablentyp: Mehrdimensionales Array
Zugewiesen in: `rss.php; include/functions_rss.inc.php, serendipity_printEntries_rss()`
Enthält Informationen für die darzustellenden Daten des Feeds. Dies können entweder Kommentare oder Blog-Artikel sein, abhängig vom gewählten Feed.

Für beide Feed-Arten gelten folgende Schlüssel:

{`$entries.X.feed_id`}
Variablentyp: Zahl
Enthält die ID des Blog-Artikels, auf die sich ein Feed-Objekt (Kommentar oder Blog-Eintrag) bezieht.

{`$entries.X.feed_guid`}
Variablentyp: Zeichenkette
Enthält eine eindeutige ID für ein Feed-Objekt. Diese ID entspricht einer URL (verkürzter Permalink), unter der sich das Feed-Objekt jederzeit aufrufen lässt.

{`$entries.X.feed_entryLink`}
Variablentyp: Zeichenkette
Enthält den Permalink des Blog-Eintrags, auf den sich ein Feed-Objekt bezieht.

{`$entries.X.feed_title`}
Variablentyp: Zeichenkette
Enthält den Titel eines Feed-Objekts. Sonderzeichen werden in UTF-8 umgewandelt, etwaiger HTML-Code wird maskiert.

{`$entries.X.feed_blogTitle`}
Variablentyp: Zeichenkette
Enthält den Titel des Blogs. Sonderzeichen werden in UTF-8 umgewandelt, etwaiger HTML-Code wird maskiert.

{$entries.X.feed_author}
Variablentyp: Zeichenkette
Enthält den Autorennamen eines Feed-Objekts. Sonderzeichen werden in UTF-8 umgewandelt, etwaiger HTML-Code wird maskiert.

{$entries.X.feed_email}
Variablentyp: Zeichenkette
Enthält die E-Mail-Adresse des Autors eines Feed-Objekts. Sonderzeichen werden in UTF-8 umgewandelt, HTML-Code wird ggf. maskiert.

{$entries.X.feed_timestamp}
Variablentyp: Zeichenkette
Enthält das Veröffentlichungsdatum eines Feed-Objekts im Format *Jahr-Monat-Tag Stunde:Minute:Sekunde Zeitzone*.

{$entries.X.feed_last_modified}
Variablentyp: Zeichenkette
Enthält das Datum der letzten Aktualisierung eines Feed-Objekts im Format *Jahr-Monat-Tag Stunde:Minute:Sekunde Zeitzone*.

{$entries.X.feed_timestamp_r}
Variablentyp: Zeichenkette
Enthält das Veröffentlichungsdatum eines Feed-Objekts im RFC-2822-Format[9].

{$entries.X.feed_body}
Variablentyp: Zeichenkette
Enthält den Inhalt des Feed-Objekts. Sonderzeichen werden in den UTF-8-Zeichensatz umgewandelt, HTML-Code wird ggf. maskiert.

{$entries.X.feed_ext}
Variablentyp: Zeichenkette
Falls ein Blog-Artikel nur ausschnittweise im Feed enthalten ist, wird in der Variable {$entries.X.feed_ext} ein HTML-Code für einen Link zur detaillierten Ansicht des Artikels gespeichert. Sonderzeichen werden in den UTF-8-Zeichensatz umgewandelt, etwaiger HTML-Code wird maskiert.

{$entries.X.per_entry_display_dat}
Variablentyp: Zeichenkette
Enthält etwaige zusätzliche HTML/XML-Ausgaben von Ereignis-Plugins.

Für Kommentar-Feeds ist das Array mit denselben Schlüsseln ausgestattet wie das Array für die Darstellung der Kommentare (siehe Seite 548). Abweichend sind einzig die beiden Schlüssel {$entries.X.author} (dieser wird automatisch bei der Darstellung eines Trackbacks um den Titel des Trackbacks erweitert) und {$entries.X.title} (dieser enthält zusätzlich den Autorennamen vorangestellt, um innerhalb des RSS-Feeds eine eindeutigere Überschrift zu erhalten).

[9] http://www.faqs.org/rfcs/rfc2822

Normale Artikel-Feeds enthalten ein Array mit den Schlüsseln, die auf Seite 521 aufgeführt sind.

{`$namespace_display_dat`}
Variablentyp: Zeichenkette
Zugewiesen in: `rss.php`
Enthält etwaige XML-Ausgaben von zusätzlichen Ereignis-Plugins, die sich in die Ausgabe des Feeds (im `namespace`-Bereich) einklinken können.

{`$once_display_dat`}
Variablentyp: Zeichenkette
Zugewiesen in: `rss.php`
Enthält etwaige XML-Ausgaben von zusätzlichen Ereignis-Plugins, die sich in die Ausgabe des Feeds (Inhaltsbereich) einklinken können.

{`$is_comments`}
Variablentyp: Boolean
Zugewiesen in: `rss.php`
Enthält den Wert `true`, falls der RSS-Feed Kommentare zu Artikeln anstelle der Artikel selbst darstellt.

{`$last_modified`}
Variablentyp: Zahl
Zugewiesen in: `rss.php`
Enthält das Datum (in Unix-Sekunden) des aktuellsten Artikels. Dieses Datum wird von RSS-Clients benutzt, um Bandbreite sparend herauszufinden, ob neue Artikel vorliegen (siehe Seite 87).

{`$self_url`}
Variablentyp: Zeichenkette
Zugewiesen in: `rss.php`
Enthält die vollständige URL, die zum Aufruf des RSS-Feeds benutzt wurde.

admin/entries.tpl

Die Template-Datei `admin/entries.tpl` ist verantwortlich für die Darstellung der Oberfläche zum Erstellen eines Blog-Artikels.

{`$entry_vars`}
Variablentyp: Eindimensionales Array
Verfügbar in: `admin/entries.tpl`
Zugewiesen in: `include/functions_entries_admin.inc.php, serendipity_printEntryForm()`
Enthält alle für den Eintragseditor notwendigen Variablen sowie die Daten des zu bearbeitenden Blog-Artikels.

{`$entry_vars.draft_mode`}
Variablentyp: Zeichenkette
Gibt an, ob anhand der Voreinstellung des Redakteurs ein neuer Blog-Artikel als *Veröffentlichung* (Wert `publish`) oder *Entwurf* (Wert `draft`) erstellt werden soll.

{`$entry_vars.moderate_comments`}
Variablentyp: Boolean
Gibt an, ob anhand der Voreinstellung des Redakteurs neue Kommentare zu dem erstellenden Blog-Artikel moderiert werden sollen (Wert `true`) oder nicht (Wert `false`).

{`$entry_vars.allow_comments`}
Variablentyp: Boolean
Gibt an, ob anhand der Voreinstellung des Redakteurs Kommentare zu dem erstellenden Blog-Artikel erstellt werden dürfen (Wert `true`) oder nicht (Wert `false`).

{`$entry_vars.category_options`}
Variablentyp: Mehrdimensionales Array
Enthält eine Liste aller dem Blog-Artikel zuweisbaren Kategorien. Jedes Unter-Array enthält alle Eigenschaften der Kategorie-Datenbanktabelle sowie einige zusätzliche Werte:

{`$entry_vars.category_options.X.categoryid`}
Variablentyp: Zahl
ID der aktuellen Kategorie.

{`$entry_vars.category_options.X.category_name`}
Variablentyp: Zeichenkette
Name der aktuellen Kategorie.

{`$entry_vars.category_options.X.category_icon`}
Variablentyp: Zeichenkette
URL für ein repräsentatives Bild der Kategorie.

{`$entry_vars.category_options.X.category_description`} *Variablentyp:* Zeichenkette
Beschreibung der aktuellen Kategorie.

{`$entry_vars.category_options.X.category_left`} , {`$entry_vars.category_options.X.category_left`} *Variablentyp:* Zahl
Verweist auf die jeweils davor und dahinter liegende Kategorie-ID und wird verwendet, um eine Baumstruktur für Kategorien herzustellen.

{`$entry_vars.category_options.X.parentid`}
Variablentyp: Zahl

Verweist auf die in der Hierarchie übergeordnete Kategorie der aktuellen Kategorie.

{`$entry_vars.category_options.X.authorid`}
Variablentyp: Zahl
ID des Autors, der die Kategorie erstellt hat.

{`$entry_vars.category_options.X.username`} ,

{`$entry_vars.category_options.X.realname`} *Variablentyp:* Zeichenkette
Name des Autors, der die Kategorie erstellt hat.

{`$entry_vars.category_options.X.loginname`}
Variablentyp: Zeichenkette
Login-Name des Autors, der die Kategorie erstellt hat.

{`$entry_vars.category_options.X.is_selected`}
Variablentyp: Boolean
Falls die aktuelle Kategorie dem neu zu erstellenden bzw. zu bearbeitenden Blog-Artikel zugewiesen ist, ist der Wert dieser Variable `true`, andernfalls `false`.

{`$entry_vars.category_options.X.depth_pad`}
Variablentyp: Zeichenkette
Enthält den benötigten HTML-Code, um die Einrückung der Kategorie-Hierarchie darzustellen. Für jede tiefer liegende Ebene wird ein geschütztes Leerzeichen (` `)
dargestellt.

{`$entry_vars.allowDateManipulation`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn in der Blog-Konfiguration die Vor-/Rückdatierung von Einträgen erlaubt wird (`$serendipity['allowDateManipulation']`, siehe Seite 179).

{`$entry_vars.show_wysiwyg`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Redakteur die Benutzung des WYSIWYG-Editors aktiviert hat (siehe Seite 104).

{`$entry_vars.wysiwyg_advanced`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Redakteur einen Browser einsetzt, der einen WYSIWYG-Editor darstellen kann (Microsoft Internet Explorer, Mozilla Firefox, Opera).

{`$entry_vars.timestamp`}
Variablentyp: Zahl
Enthält das aktuelle Datum oder, falls ein Blog-Artikel bearbeitet wird, das Datum des Artikels (in Unix-Sekunden).

{$entry_vars.reset_timestamp}
Variablentyp: Zahl
Enthält immer das aktuelle Datum (in Unix-Sekunden).

{$entry_vars.hidden}
Variablentyp: Zeichenkette
Enthält HTML-Code mit versteckten HTML-Formularfeldern, die einige interne Variablen zuweisen.

{$entry_vars.errMsg}
Variablentyp: Zeichenkette
Falls beim Speichern oder Bearbeiten eines Artikels Fehlermeldungen auftreten, werden sie in dieser Variable gespeichert.

{$entry_vars.targetURL}
Variablentyp: Zeichenkette
Enthält das Verweisziel (URL) des HTML-Formulars. Da das Formular zum Bearbeiten von Artikeln von unterschiedlichen Stellen des Serendipity-Kerns auch z. B. durch Plugins aufgerufen werden kann, kann die Ziel-URL unterschiedlich lauten und wird daher über eine Variable abstrahiert.

{$entry_vars.cat_count}
Variablentyp: Zahl
Enthält die Anzahl der zuweisbaren Blog-Kategorien.

{$entry_vars.cat_state}
Variablentyp: Zeichenkette
Wenn einem Blog-Artikel mehr als eine Kategorie zugewiesen werden soll, kann ein Mehrfach-Auswahlfeld für die Kategorie-Auswahl aktiviert werden (siehe Seite 103). Sobald einem Artikel mehr als eine Kategorie zugewiesen wurde, erscheint beim Bearbeiten dieses Artikels standardmäßig das Mehrfach-Auswahlfeld, andernfalls nur ein einfaches Ausklappfeld. Die Variable `entry_vars.cat_state` bestimmt, wie die Kategorie-Auswahl dargestellt wird. Der Wert `on` steht für *Mehrfach-Auswahlfeld*, `off` für *einfaches Ausklappfeld*.

{$entry_vars.serendipityRightPublish}
Variablentyp: Boolean
Enthält den Wert `true`, wenn ein Redakteur die Erlaubnis hat, Artikel zu veröffentlichen (siehe Seite 185).

{$entry_vars.wysiwyg_blocks}
Variablentyp: Eindimensionales Array
Enthält eine Liste von Blöcken, die durch den WYSIWYG-Editor gefüllt werden können (`body`, `extended`).

{$entry_vars.entry_template}
Variablentyp: Zeichenkette
Enthält den Pfad zur darzustellenden Template-Datei `admin/entries.tpl`. Plugins können ein Backend-Template frei modifizieren, um unterschiedliche Artikel-Editor-Oberflächen darzustellen.

{`$entry_vars.entry`}
Variablentyp: Eindimensionales Array
Enthält die Meta-Daten des Blog-Artikels, sofern dieser gespeichert wurde oder sich in der Vorschau befindet. Folgende Array-Schlüssel sind verfügbar:

{`$entry_vars.entry.isdraft`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn der Artikel als Entwurf gespeichert wurde.

{`$entry_vars.entry.moderate_comments`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn die Kommentare zu dem Blog-Artikel moderiert werden sollen.

{`$entry_vars.entry.allow_comments`}
Variablentyp: Boolean
Ist auf `true` gesetzt, wenn Kommentare zu dem Blog-Artikel erlaubt sind.

{`$entry_vars.entry.title`}
Variablentyp: Zeichenkette
Enthält den Titel des Blog-Artikels.

{`$entry_vars.entry.body`}
Variablentyp: Zeichenkette
Enthält den Artikeltext.

{`$entry_vars.entry.extended`}
Variablentyp: Zeichenkette
Enthält den erweiterten Artikeltext.

{`$entry_vars.entry.id`}
Variablentyp: Zahl
Enthält die ID des Artikels.

{`$entry_vars.entry.timestamp`}
Variablentyp: Zahl
Enthält das Erstellungsdatum des Artikels (in Unix-Sekunden).

{`$entry_vars.entry.entry_form`}
Variablentyp: Zeichenkette
Enthält etwaigen HTML-Code, der als Zusatz im `<form>`-HTML-Tag ausgegeben werden kann, um z. B. individuelles JavaScript auszugeben. Plugins können diese Variable benutzen, um erweiterte Funktionen einzubinden.

sidebar.tpl

Die Template-Datei `sidebar.tpl` ist zuständig für die Darstellung der Seitenleisten-Plugins. Sie wird jeweils pro konfigurierter Seitenleiste (links, rechts ...) mittels des Funktionsaufrufs {`$serendipity_printSidebar`} (siehe Seite 563) eingelesen und meist in der `index.tpl` aufgerufen.

{$plugindata}
Variablentyp: Mehrdimensionales Array
Verfügbar in: sidebar.tpl
Zugewiesen in: include/plugin_api.inc.php, serendipity_plugin_api::generate_plugins()
Das Array {$plugindata} enthält die Ausgaben (und Metadaten) aller dargestellten Seitenleisten-Plugins. Das mehrdimensionale Array enthält für jedes durchnummerierte Unter-Array folgende Schlüssel:

{$plugindata.X.side}
Variablentyp: Zeichenkette
Enthält den Namen der zugehörigen Seite des jeweiligen Seitenleisten-Plugins, z. B. left für ein Element der linken Seitenleiste.

{$plugindata.X.class}
Variablentyp: Zeichenkette
Enthält den PHP-Klassennamen des Seitenleisten-Plugins, z. B. serendipity_plugin_comments
Der PHP-Klassenname entspricht dabei auch üblicherweise dem Datei- und Verzeichnisnamen des Plugins, somit wäre der genannte Plugin-Code in der Datei plugins/serendipity_plugin_comments/serendipity_plugin_comments.php enthalten.

{$plugindata.X.title}
Variablentyp: Zeichenkette
Enthält den Titel eines Seitenleisten-Plugins, wie er in der Seitenleiste angezeigt werden soll. Dieser Titel ist bei vielen Plugins vom Benutzer frei wählbar.

{$plugindata.X.content}
Variablentyp: Zeichenkette
Enthält die Ausgabe eines Seitenleisten-Plugins als HTML-Code.

{$plugindata.X.id}
Variablentyp: Zeichenkette
Enthält eine eindeutige Identifikationsnummer des jeweiligen Seitenleisten-Plugins. Die ID wird bei Installation des Plugins vergeben, ist also auch bei mehrfach installierten gleichartigen Seitenleisten-Plugins für jede einzelne Instanz unterschiedlich.

Interne Plugins (die im Serendipity-Kern enthalten sind) enthalten in der Identifikationsnummer ein vorangestelltes @-Symbol, darauf folgt der Klassenname des Plugins und abschließend ein Doppelpunkt mit einer zufälligen, 32 Zeichen langen Kombination aus Zahlen und Buchstaben.

{$pluginside}
Variablentyp: Zeichenkette
Verfügbar in: sidebar.tpl
Zugewiesen in: include/plugin_api.inc.php,

`serendipity_plugin_api::generate_plugins()`
Da die Template-Datei `sidebar.tpl` pro Seitenleiste (links, rechts ...) mehrfach aufgerufen wird, bestimmt die zentrale Variable jeweils die entsprechende Seitenleiste. Die Werte der Variable entsprechen dem konfigurierten Seitenleistennamen wie `Left`, `Right` (siehe auch Seite 508). Zu beachten ist, dass der erste Buchstabe des Namens großgeschrieben wird.

Diese Variable wird üblicherweise dazu benutzt, um eindeutige CSS-Klassenselektoren (wie `serendipityLeftSidebar`) automatisch zu benennen und zuzuweisen.

admin/media_*.tpl

Die Template-Dateien für die Darstellung der Mediendatenbank[10] werden innerhalb des Backends eingebunden. Für welchen Bereich die jeweilige Datei zuständig ist, wird auf Seite 492 beschrieben.

{`$media`} (gültig in `media_pane.tpl`, `media_properties.tpl`)
Variablentyp: Eindimensionales Array
Verfügbar in: `media_upload.tpl, media_properties.tpl`
Zugewiesen in: `include/functions_images.inc.php, serendipity_showMedia()`
Enthält die Liste ein oder mehrerer Medienobjekte bei Darstellung in der Mediendatenbank-Übersicht. Die Ergebnisse der Template-Dateien `media_upload.tpl` oder `media_properties.tpl` werden über die Smarty-Variable {`$MEDIA_ITEMS`} in den jeweiligen Kontext anderer Smarty-Templates eingebunden.

{`$media.manage`}
Variablentyp: Boolean
Gibt an, ob Symbole zum Bearbeiten einer Datei der Mediendatenbank (Bearbeiten, Löschen, Drehen ...) angezeigt werden sollen. Enthält den Wert `true`, falls aktiviert.

{`$media.perPage`}
Variablentyp: Zahl
Enthält die Anzahl der maximal pro Seite darzustellenden Dateien der Mediendatenbank.

{`$media.page`}
Variablentyp: Zahl
Enthält die Seitennummer der aktuell dargestellen Übersichtsseite der Mediendatenbank.

[10] `admin/media_choose.tpl, media_imgedit.tpl, media_imgedit_done.tpl, media_pane.tpl, media_items.tpl, media_properties.tpl, media_upload.tpl, media_showitem.tpl`

{`$media.pages`}
Variablentyp: Zahl
Enthält die Seitenanzahl der dargestellten Übersicht der Mediendatenbank.

{`$media.linkNext`} , {`$media.linkPrevious`}
Variablentyp: Zeichenkette
Enthält die URLs zum Zurück- und Weiterblättern der Mediendatenbank-Übersicht.

{`$media.extraParems`}
Variablentyp: Zeichenkette
Enthält etwaige URL-GET-Variablen, die an die URL zum Blättern angehängt werden müssen, beispielsweise Filtereinschränkungen und Sortierungsoptionen.

{`$media.show_upload`}
Variablentyp: Boolean
Falls auf `true` gesetzt, soll die Einbindung der Oberfläche zum Hochladen einer Datei in die Mediendatenbank mit angezeigt werden.

{`$media.lineBreak`}
Variablentyp: Zahl
Gibt an, nach wie vielen Dateien der Mediendatenbank ein Zeilenumbruch erfolgen soll.

{`$media.lineBreakP`}
Variablentyp: Zahl
Enthält die relative Breite in Prozent für jede Datei der Mediendatenbank bei die Darstellung innerhalb der Übersicht. Bei z. B. 3 Bildern pro Zeile kann jedes Bild 33% der Breite einnehmen. Diese Zahl errechnet sich anhand der Variable {`$media.lineBreak`} .

{`$media.url`}
Variablentyp: Zeichenkette
Enthält die URL zum Backend, mittels derer die Bearbeitungsaktionen ausgeführt werden können.

{`$media.enclose`}
Variablentyp: Boolean
Falls auf `true` gesetzt, werden die Dateien der Mediendatenbank mittels des Templates `admin/media_pane.tpl` in einer Übersicht dargestellt. Falls auf `false` gesetzt, wird das Template `admin/media_properties.tpl` verwendet, das die Detaileigenschaften einer einzelnen Datei der Mediendatenbank darstellen kann.

{`$media.zoomIMG`} , {`$media.renameIMG`} ,

{`$media.resizeIMG`} , {`$media.rotatecwIMG`} ,

{`$media.rotateccwIMG`} , {`$media.configureIMG`} ,

{$media.deleteIMG} , {$media.prevIMG} , {$media.nextIMG} *Variablentyp:* Zeichenkette
Enthält den Pfad zur jeweiligen Datei mit den Symbolen zum Bearbeiten der Dateien in der Mediendatenbank (Löschen, Umbenennen . . .).

{$media.token}
Variablentyp: Zeichenkette
Enthält eine zufällige Zeichenkette, die ein Formular vor dem Fernsteuern durch unberechtigte Zugriffe schützt (siehe XSRF auf Seite 23). Diese Zeichenkette muss als verstecktes Element des Formulars zum Hochladen der Datei eingebunden werden.

{$media.form_hidden}
Variablentyp: Zeichenkette
Enthält eine Liste von benötigten, versteckten HTML-Formularfeldern. Diese werden vom Serendipity-Kern geliefert und müssen in das Formular des Templates eingebunden werden. Die Variable enthält volllständig formatierten HTML-Code.

{$media.blimit_path}
Variablentyp: Zeichenkette
Falls die Ausgabe der Dateien der Mediendatenbank auf ein Verzeichnis eingeschränkt wird, enthält diese Variable den letzten übergeordneten Verzeichnisnamen.

{$media.only_path} , {$media.limit_path}
Variablentyp: Zeichenkette
Falls die Ausgabe der Dateien der Mediendatenbank auf ein Verzeichnis eingeschränkt wird, enthält diese Variable den vollständigen Verzeichnisnamen.

{$media.only_filename}
Variablentyp: Zeichenkette
Falls die Ausgabe der Dateien der Mediendatenbank auf einen speziellen Dateinamen eingeschränkt wird, enthält diese Variable den begrenzenden Dateinamen.

{$media.sortorder}
Variablentyp: Eindimensionales Array
Falls die Ausgabe der Dateien der Mediendatenbank mit einer vom Benutzer bestimmten Sortierungsreihenfolge erfolgt, enthält diese Variable die Details zur Sortierung. Dazu werden die Unter-Array-Schlüssel order (Datenbank-Spaltenname), ordermode (aufsteigende oder absteigende Sortierung, ASC/DESC), perpage (Anzahl der Elemente pro Seite) eingebunden.

{$media.keywords_selected}
Variablentyp: Zeichenkette
Falls die Ausgabe der Dateien der Mediendatenbank auf Dateien mit speziellem Schlüsselwort eingeschränkt wird, enthält diese Variable die Liste der Schlüsselwörter.

{$media.keywords}
Variablentyp: Eindimensionales Arrays
Enthält ein Array aller verfügbaren Schlüsselwörter (siehe Seite 174).

{$media.filter}
Variablentyp: Mehrdimensionales Array
Falls die Ausgabe der Dateien der Mediendatenbank mit einer vom Benutzer bestimmten Filterung erfolgt, enthält diese Variable die jeweiligen Filtermechanismen. Jeder Unter-Array-Schlüssel entspricht dem Namen eines Datenbankfeldes, das jeweils weitere Array-Schlüssel enthalten kann. Jeder Filter besitzt einen `from`- und `to`-Schlüssel, der einen Wert *von/bis* beschränkt. Beispiel:

```
$media['filter'] = array(
  'title'     => 'Intern',
  'timestamp' => array(
                  'from' => 1192455852,
                  'to'   => 1182455852
                 ),
  'authorid'  => '12'
);
```

Die Liste der verfügbaren Filter richtet sich nach der Variable {`$media.sort_order`}.

{$media.sort_order}
Variablentyp: Mehrdimensionales Array
Enthält die Liste aller filterbaren Datenbankfelder der Mediendatenbank. Diese Variable wird durch die Serendipity-Funktion `serendipity_getImageFields()` (Datei `include/functions_images.inc.php`) gefüllt und gibt Metadaten wie Typ und Beschreibung jedes Datenfeldes aus. Der Haupt-Array-Schlüssel gibt den Datenbankspaltennamen (wie `i.date`) an, die Unterschlüssel `desc` und `type` enthalten die jeweiligen Metadaten.

Die Liste dieser filterbaren Datenbankspalten wird dynamisch in den Template-Dateien ausgewertet, um die entsprechenden Eingabeboxen für das Suchformular zu erstellen.

{$media.authors}
Variablentyp: Mehrdimensionales Array
Enthält ein Array aller verfügbaren Autoren des Blogs, so dass die Auswahl der Dateien der Mediendatenbank auf einen gewünschten Eigentümer eingeschränkt werden kann. Die Liste der Autoren wird mittels der Serendipity-Funktion `serendipity_fetchUsers()` (Datei `include/functions.inc.php`) bezogen. Die verfügbaren Array-Schlüssel entsprechen den Datenbankspaltennamen der `serendipity_authors`-Tabelle.

{$media.sort_row_interval}
Variablentyp: Eindimensionales Array
Enthält ein Array mit den Werten, wie viele Dateien pro Seite dargestellt werden können. Standardmäßig sind das die Werte `8, 16, 50, 100`.

{`$media.nr_files`}
Variablentyp: Zahl
Enthält die Anzahl der auf der Seite dargestellten Dateien der Mediendatenbank.

{`$media.files`}
Variablentyp: Mehrdimensionales Array
Enthält ein Array aller darzustellenden Dateien der Mediendatenbank.

{`$media.paths`}
Variablentyp: Mehrdimensionales Array
Enthält ein Array aller eingerichteten Verzeichnisse der Mediendatenbank. Die verfügbaren Array-Schlüssel sind identisch mit den auf Seite 540 aufgeführten.

{`$media.is_edit`}
Variablentyp: Boolean
Verfügbar in: `media_properties.tpl`
Ist `true` gesetzt, wenn das Formular der Template-Datei `media_properties.tpl` zum Bearbeiten der Eigenschaften einer Datei der Mediendatenbank angezeigt werden soll.

{`$media.editform_hidden`}
Variablentyp: Zeichenkette
Verfügbar in: `media_properties.tpl`
Enthält HTML-Code für versteckte Formularwerte, die zum Bearbeiten einer Datei der Mediendatenbank benötigt werden.

{`$media.keywordsPerBlock`}
Variablentyp: Zahl
Verfügbar in: `media_properties.tpl`
Gibt an, wie viele Schlüsselwörter beim Zuweisen zu einer Datei der Mediendatenbank nebeneinander in einer Zeile dargestellt werden können (standardmäßig `3`).

{`$media.dprops`}
Variablentyp: Eindimensionales Array
Verfügbar in: `media_properties.tpl`
Enthält ein Array mit den verfügbaren Eigenschaftsfeldern für eine Datei der Mediendatenbank, wie im Blog konfiguriert (siehe Seite 262).

{`$media`} (gültig in `media_upload.tpl`)
Variablentyp: Eindimensionales Array
Verfügbar in: `media_upload.tpl`
Zugewiesen in: `include/admin/images.inc.php, case 'addSelect'`
Enthält globale Eigenschaften der Mediendatenbank und deren vom Benutzer angelegte Verzeichnisse, die benötigt werden, um das Formular zum Hochladen einer Datei darzustellen.

{$media.token}
Variablentyp: Zeichenkette
Enthält eine zufällige Zeichenkette, die ein Formular vor dem Fernsteuern durch unberechtigte Zugriffe schützt (siehe XSRF auf Seite 23). Diese Zeichenkette muss als verstecktes Element des Formulars zum Hochladen der Datei eingebunden werden.

{$media.form_hidden}
Variablentyp: Zeichenkette
Enthält eine Liste von benötigten, versteckten HTML-Formularfeldern. Diese werden vom Serendipity-Kern geliefert und müssen in das Formular des Templates eingebunden werden. Die Variable enthält vollständig formatierten HTML-Code.

{$media.folders}
Variablentyp: Mehrdimensionales Array
Enthält eine Liste aller Verzeichnisse und Unterverzeichnisse der Mediendatenbank. Folgende Array-Schlüssel sind verfügbar:

{$media.folders.X.name}
Variablentyp: Zeichenkette
Enthält den Namen des jeweiligen Verzeichnisses.

{$media.folders.X.depth}
Variablentyp: Zeichenkette
Gibt die Tiefe eines Verzeichnisses an. Für jedes Unterverzeichnis wird dieser Wert um eins erhöht. Für /bilder/aktuell/pfadfinder/ würde die Variable den Wert 3 enthalten.

{$media.folders.X.relpath}
Variablentyp: Zeichenkette
Enthält den vollständigen Pfad des jeweiligen Verzeichnisses (z. B. bilder/aktuell/pfadf
Das Verzeichnis wird ohne führenden, aber mit abschließendem Schrägstrich ausgegeben.

{$media.folders.X.directory}
Variablentyp: Boolean
Enthält den Wert true, wenn das Element ein Verzeichnis darstellt. Ein Array derselben Struktur kann von einer internen Serendipity-Funktion auch erstellt werden, wenn eine Liste aller verfügbaren Dateien der Mediendatenbank ausgegeben werden soll. Nur in einem solchen Fall kann der Wert auch false annehmen. Im Kontext der Template-Datei media_upload.tpl ist der Wert der Variable {$media.folders.X.directory} jedoch erwartungsgemäß stets true.

{$media.only_path}
Variablentyp: Zeichenkette
Falls die Auswahl der Verzeichnisse für die hochzuladende Datei auf einen Teil des Verzeichnisbaums beschränkt werden soll, enthält die Variable media.only_path

den Pfad des Verzeichnisses, ab dem Unterverzeichnisse eingebunden werden sollen.

{`$media.max_file_size`}
Variablentyp: Zahl
Enthält die maximale Größe (in Bytes) einer hochzuladenden Datei. Diese kann in der Serendipity-Konfiguration festgelegt werden (siehe Seite 172).

{`$media.maxImgHeight, media.maxImgWidth`}
Variablentyp: Zahl
Enthält die maximale Bildbreite und -höhe einer hochzuladenden Datei. Diese kann in der Serendipity-Konfiguration festgelegt werden (siehe Seite 172).

{`$media`} (gültig in `media_choose.tpl`, `media_showitem.tpl`)
Variablentyp: Eindimensionales Array
Verfügbar in: `media_choose.tpl, media_showitem.tpl`
Zugewiesen in: `serendipity_admin_image_selector.php`

{`$media.GET_STRING`}
Variablentyp: Zeichenkette
Enthält eine Zeichenkette mit den benötigten URL-Variablen, die innerhalb des Popups zur Auswahl einer Mediendatenbank-Datei an die Folgeseiten weitergereicht werden müssen.

{`$media.frameset`}
Variablentyp: Boolean
Enthält den Wert `true`, wenn die erste Seite des Mediendatenbank-Popups eingebunden wird. In diesem Fall stellt die Template-Datei `media_choose.tpl` keine Inhalte dar, sondern nur das globale Frameset (links Verzeichnisnavigation, rechts Inhalt).

{`$media.body_id`}
Variablentyp: Zeichenkette
Enthält eine eindeutige ID, die dem `<body>`-Element des Templates zugewiesen wird, um via CSS unterscheiden zu können, ob innerhalb des Framesets die Verzeichnis- (Wert `serendipityAdminBodyImageSelectorTree`) oder die Dateiauswahl (`serendipityAdminBodyImageSelector`) dargestellt wird.

{`$media.css`}
Variablentyp: Zeichenkette
Enthält die URL zur `admin/style.css`-Datei für die Darstellung des Serendipity-Backends. In diesem Stylesheet wird auch das Aussehen der Mediendatenbank definiert.

{`$media.css_tree`}
Variablentyp: Zeichenkette

Enthält die URL zur `treeview/tree.css`-Datei für die Darstellung der Verzeichnisauswahl der Mediendatenbank.

{`$media.css_front`}
Variablentyp: Zeichenkette
Enthält die URL zur `style.css`-Datei für die Darstellung des Frontends.

{`$media.token_url`}
Variablentyp: Zeichenkette
Enthält eine zufällige Zeichenkette, die ein Formular vor dem Fernsteuern durch unberechtigte Zugriffe schützt (siehe `XSRF` auf Seite 23). Diese Zeichenkette muss als Teil einer URL (GET-Variablen) eingebunden werden.

{`$media.imgID`}
Variablentyp: Zahl
Falls nur eine einzelne Datei in der Oberfläche der Mediendatenbank angezeigt werden soll, enthält diese Variable die ID für dieses Objekt. Diese Variable entspricht der URL-GET-Variable `$serendipity['GET']['image']`. Wurde eine Datei gerade hochgeladen, enthält diese Variable die neue ID dieser Datei in der Mediendatenbank.

{`$media.from`}
Variablentyp: Zeichenkette
Enthält den Wert der URL-GET-Variable `$serendipity['GET']['from']`, der die Verweisquelle für ein Bild ({`$media.imgID`}) angibt, von der aus ein Bild eingebunden wurde.

{`$media.case`}
Variablentyp: Zeichenkette
An einigen Stellen bindet das Popup-Fenster der Mediendatenbank Inhalte des Backends ein, die sonst über andere Menüpunkte ereichbar sind. Diese werden extern eingebunden.

Wenn ein Bild über das Mediendatenbank-Popup-Fenster eingebunden wird, bestimmt der Benutzer über mehrere aufeinanderfolgende Schritte die Auswahl und Platzierung des Bildes. Die Ausgabe erfolgt jeweils innerhalb desselben Templates (`media_choose.tpl`). Um die Schritte voneinander unterscheiden zu können, enthält diese Template-Variable einen eindeutigen Identifikator. Dieser kann auch dann abgefragt werden, wenn das Popup-Fenster administrative Aktionen einbindet, die sonst eigentlich über andere Menüpunkte erreichbar sind (z. B. **Verzeichnisse verwalten**). Die Variable kann folgende Werte aufweisen:

`external` bedeutet, dass Teile des Backends in der Mediendatenbank eingebunden werden. Diese HTML-Ausgaben werden in der Variable {`$media.external`} gespeichert.

`choose` bedeutet, dass der Benutzer eine Datei ausgewählt hat und nun deren Platzierung bestimmen möchte.

`tree` bedeutet, dass die Verzeichnisstruktur (bei der Darstellung des linken Frames) ausgegeben wird.

`showitem` bedeutet, dass die Eigenschaften eines speziellen Bildes dargestellt werden.

`start` bedeutet, dass die Frameset-Übersicht angezeigt wird.

`default` bedeutet, dass die Übersicht der Dateien der Mediendatenbank angezeigt wird.

{`$media.external`}
Variablentyp: Zeichenkette
Enthält etwaige HTML-Ausgaben von Teilen des Backends (Formular zum Datei-Upload, Verzeichnisse verwalten ...) zur Einbindung in das Popup der Mediendatenbank.

{`$media.is_uploaded`}
Variablentyp: Boolean
Ist auf den Wert `true` gesetzt, wenn gerade eine Datei in die Mediendatenbank hochgeladen wurde. Mittels dieser Variable können dann entsprechende Meldungen über den Erfolg oder das Fehlschlagen des Vorgangs ausgegeben werden.

{`$media.is_created`}
Variablentyp: Boolean
Ist auf den Wert `true` gesetzt, wenn gerade ein Verzeichnis in der Mediendatenbank erstellt wurde. Mittels dieser Variable können dann entsprechende Meldungen über den Erfolg oder das Fehlschlagen des Vorgangs ausgegeben werden.

{`$media.is_deleted`}
Variablentyp: Boolean
Ist auf den Wert `true` gesetzt, wenn gerade ein Verzeichnis oder eine Datei in der Mediendatenbank gelöscht wurde. Mittels dieser Variable können dann entsprechende Meldungen über den Erfolg oder das Fehlschlagen des Vorgangs ausgegeben werden.

{`$media.new_dir`}
Variablentyp: Zeichenkette
Falls ein neues Verzeichnis erstellt wurde, enthält diese Variable den Pfad zu diesem neuen Verzeichnis (siehe auch {`$media.is_created`}).

{`$media.file`}
Variablentyp: Mehrdimensionales Array
Enthält die Metadaten für eine gewählte Datei der Mediendatenbank. Die Array-Schlüssel sind:

{`$media.file.id`}
Variablentyp: Zahl
Enthält die ID des Mediendatenbank-Objekts.

{`$media.file.name`}
Variablentyp: Zeichenkette

Enthält den Dateinamen des Mediendatenbank-Objekts (ohne Datei-Endung).

{`$media.file.extension`}
Enthält die Datei-Endung des Mediendatenbank-Objekts. *Variablentyp:* Zeichenkette

{`$media.file.mime`}
Variablentyp: Zeichenkette
Enthält den MIME[11]-Typ eines Mediendatenbank-Objekts.

{`$media.file.size`}
Variablentyp: Zahl
Enthält die Dateigröße (in Bytes) des Mediendatenbank-Objekts.

{`$media.file.dimensions_width`} ,

{`$media.file.dimensions_height`} *Variablentyp:* Zahl
Falls das Mediendatenbank-Objekt eine Grafik ist, enthalten diese Variablen die Breite und Höhe (in Pixeln). Bei anderen Dateitypen (Dokumente, Videos) enthalten diese Variablen den Wert 0.

{`$media.file.thumbWidth`} ,

{`$media.file.thumbHeight`} *Variablentyp:* Zahl
Enthält die Breite und Höhe der Vorschaugrafik.

{`$media.file.date`}
Variablentyp: Zahl
Enthält Hochladedatum eines Mediendatenbank-Objekts (in Unix-Sekunden).

{`$media.file.thumbnail_name`}
Variablentyp: Zeichenkette
Wurde ein Vorschaubild (*Thumbnail*) für ein Mediendatenbank-Objekt erstellt, enthält diese Variable das Suffix für diese Dateien, standardmäßig `serendipityThumb`. Dieses Suffix wird nach dem Dateinamen, aber vor der Datei-Endung platziert, so dass eine Vorschaudatei dem Schema `Bild.serendipityThumb.jpg` entspricht. Das Suffix kann vom Administrator des Blogs frei vergeben werden (siehe Seite 170).

{`$media.file.authorid`}
Variablentyp: Zahl
Enthält die ID des Autors, der die Datei hochgeladen hat. Wenn eine Datei für alle Redakteure zugänglich sein soll, enthält diese ID den Wert `0`.

[11] *Multipurpose Internet Mail Extensions* sind eine standardisierte Spezifikation von Datentypen und geben an, ob ein Objekt ein Bild, eine Videodatei oder anderes darstellt. Ein MIME-Type besteht aus einem Basistypen, gefolgt von einem Schrägstrich und dann einem Detailwert. `image/jpeg` wird beispielsweise für JPEG-Grafiken verwendet.

{`$media.file.path`}
Variablentyp: Zeichenkette
Enthält den Namen des Verzeichnisses (relativ zum Stammverzeichnis der Mediendatenbank), in dem das Objekt gespeichert wurde.

{`$media.file.realfile`}
Variablentyp: Zeichenkette
Enthält den vollständigen Pfad einer Datei der Mediendatenbank, wo sie auf dem Server abgelegt wurde.

{`$media.file.hotlink`}
Variablentyp: Zeichenkette
Wenn ein Objekt der Mediendatenbank nicht lokal gespeichert wurde, sondern von einem fremden Server geladen werden soll (`hotlinking`), enthält diese Variable die URL, unter der die Datei verfügbar ist.

{`$media.file.nice_hotlink`}
Variablentyp: Zeichenkette
Falls das Objekt der Mediendatenbank von einem fremden Server geladen wird (`hotlinking`), enthält die Variable die Ziel-URL, die jeweils nach 45 Zeichen automatisch umbrochen wird, so dass man sie problemlos unterhalb eines Bildes anzeigen kann.

{`$media.file.realname`}
Variablentyp: Zeichenkette
Enthält den vollständigen Dateinamen, wie die Datei ursprünglich hochgeladen wurde. Da im Falle doppelter Dateinamen der Name auf dem Server automatisch durchnummeriert wird, kann mittels {`$media.file.realname`} dennoch der ursprüngliche Dateiname ausgelesen werden.

{`$media.file.full_thumb`}
Variablentyp: Zeichenkette
Enthält den vollständigen Pfad im Dateisystem zum Vorschaubild eines Mediendatenbank-Objekts.

{`$media.file.full_thumbHTTP`} ,

{`$media.file.show_thumb`} *Variablentyp:* Zeichenkette
Enthält den vollständigen Pfad zum Vorschaubild eines Mediendatenbank-Objekts, über den das Bild mittels HTTP-URL aufgerufen werden kann.

{`$media.file.imgsrc`}
Variablentyp: Zeichenkette
Enthält den relativen Pfad (ab Serendipity-Stammverzeichnis) zum Vorschaubild eines Mediendatenbank-Objekts, über den das Bild mittels HTTP-URL aufgerufen werden kann.

{`$media.file.full_file`}
Variablentyp: Zeichenkette
Enthält den vollständigen HTTP-Pfad zum Objekt der Mediendatenbank.

{$media.file.diskname}
Variablentyp: Zeichenkette
Enthält den vollen Dateinamen eines Objekts der Mediendatenbank, zusammengesetzt aus {$media.file.name} und {$media.file.extension} .

{$media.file.links}
Variablentyp: Eindimensionales Array
Enthält ein Array mit URLs, von denen auf das Objekt der Mediendatenbank verwiesen wurde. Diese Linkliste kann nur dann automatisch ausgewertet werden, wenn die Objekte der Mediendatenbank mittels spezieller Methode ausgeliefert werden (siehe Seite 687).

{$media.file.dim}
Variablentyp: Eindimensionales Array
Enthält ein Array mit Metadaten des Vorschaubildes. Das Array wird mittels der PHP-Funktion `getimagesize`[12] erstellt.

{$media.file.is_image}
Variablentyp: Boolean
Enthält den Wert `true`, wenn ein Objekt der Mediendatenbank eine Bilddatei ist. Bei Dokumenten und anderen Dateitypen enthält diese Variable den Wert `false`.

{$media.file.mediatype}
Variablentyp: Zeichenkette
Enthält den genauen Typ eines Objektes der Mediendatenbank: `image`, `video`, `audio`, `document`, `archive` (gepackte Dateiarchive) oder `binary` (alle anderen Dateien).

{$media.file.is_editable}
Variablentyp: Boolean
Falls der aktuelle Redakteur das Objekt der Mediendatenbank bearbeiten oder löschen darf (aufgrund seiner Benutzerrechte), enthält diese Variable den Wert `true`.

{$media.file.preview_url}
Variablentyp: Zeichenkette
Enthält die URL, mit der das Bild mittels der Funktionen der Mediendatenbank dargestellt werden kann.

{$media.file.preview}
Variablentyp: Zeichenkette
Enthält ein HTML <img>-Tag, das die Vorschaugrafik darstellt und ggf. einen Link (<a href) zur Originalversion der Datei (als Popup-Fenster) einbindet.

{$media.file.popupWidth} ,

[12] http://www.php.net/getimagesize

{`$media.file.popupHeight`} *Variablentyp:* Zahl
Enthält die Breite und Höhe für ein Popup-Fenster, in dem die Originalversion der Datei per Klick auf das Vorschaubild geöffnet wird.

{`$media.file.nice_size`}
Variablentyp: Zeichenkette
Enthält die Dateigröße als auf zwei Nachkommastellen gerundeten Kilobyte-Wert.

{`$media.file.finishJSFunction`}
Variablentyp: Zeichenkette
Wenn ein Objekt der Mediendatenbank vom Benutzer zum Einfügen in einen Artikel ausgewählt wird, können externe Plugins die Darstellung des Dialogs zum Einfügen verändern. Am Ende wird der zusammengestellte HTML-Code für die Einbindung in den Artikel zurückgeliefert und über die JS-Funktion `serendipity_imageSelector_done()` ausgewertet. Wenn ein Plugin stattdessen ein eigenes JavaScript verwenden will, muss der Name dieser JavaScript-Funktion in der Variable {`$media.file.finishJSFunction`} hinterlegt werden und wird dort später über die Smarty-Template-Datei `media_choose.tpl` aufgerufen.

{`$media.file.origfinishJSFunction`}
Variablentyp: Zeichenkette
Diese Variable enthält den ursprünglichen JavaScript-Funktionsaufruf für die Einbindung eines Objekts der Mediendatenbank (siehe vorige Variable).

{`$media.file.fast_select`}
Variablentyp: Boolean
Falls ein Redakteur ein Objekt der Mediendatenbank, das keine Grafik darstellt, zum Einfügen in einen Artikel auswählt, kann ein vollständiger Link zu dieser Datei ohne weitere Positionierungsdialoge zurückgeliefert werden. Die
Template-Datei `media_choose.tpl` kann in einem solchen Fall einige Funktionsaufrufe übergehen. Dies geschieht über die Variable {`$media.file.fast_select`}, die den Wert `true` enthält, wenn keine Grafikdatei gewählt wurde.

{`$media.file.props`}
Variablentyp: Mehrdimensionales Array
Dieses Array enthält die Meta-Eigenschaften einer Datei, wie sie vom Benutzer frei vergeben werden können (Schlüsselwörter, Beschreibungen, EXIF-Daten ...). Das Array ist in vier Hauptschlüssel untergliedert:

{$media.file.props.base_property}
Variablentyp: Eindimensionales Array
Enthält die vom Benutzer zugewiesenen freien Eigenschaften für Objekte (siehe Seite 262). Der jeweilige Schlüssel des Unter-Arrays entspricht dem Namen der Eigenschaft (DPI, COPYRIGHT ...), der Wert enthält die Eingabe des Benutzers.
{$media.file.props.base_hidden}
Variablentyp: Eindimensionales Array
Enthält interne Wertezuweisungen. {$media.file.props.base_hidden.author} enthält den Namen des Autors für das jeweilige Objekt der Mediendatenbank und {$media.file.props.base_hidden.authorid} die ID dieses Autors.
{$media.file.props.base_keyword}
Variablentyp: Eindimensionales Array
Enthält die vom Benutzer zugewiesenen Schlüsselwörter zu einem Objekt der Mediendatenbank (siehe Seite 174). Der Array-Schlüssel entspricht dem gesetzten Schlüsselwort, als Wert wird immer 1 vergeben.
{$media.file.props.base_metadata}
Variablentyp: Eindimensionales Array
Enthält etwaige EXIF- oder MP3-Metadaten. Der Name des Array-Schlüssels entspricht dem Metadaten-Schlüssel und der Wert dem jeweiligen Inhalt.

{$media.perm_denied}
Variablentyp: Boolean
Ist auf den Wert true gesetzt, wenn ein Benutzer für eine gewählte Aktion keine ausreichenden Zugriffsrechte hat.

{$media.paths}
Variablentyp: Mehrdimensionales Array
Enthält ein Array aller eingerichteten Verzeichnisse der Mediendatenbank. Die verfügbaren Array-Schlüssel sind identisch mit den auf Seite 540 aufgeführten.

{$MEDIA_LIST}
Variablentyp: Zeichenkette
Verfügbar in: media_choose.tpl, media_showitem.tpl
Enthält die Dateiübersicht als HTML-Ausgabe, die anhand der Template-Datei media_items.tpl geparst wurde.

commentpopup.tpl, comments.tpl, trackbacks.tpl

Die Template-Datei commentpopup.tpl wird zur Ausgabe von Kommentaren zu einem Blog-Artikel verwendet. Die Kommentare und Trackbacks selbst werden über die Template-Dateien comments.tpl und trackbacks.tpl eingebunden, die auch für die Einbin-

dung von Kommentaren innerhalb der Detailansicht eines Blog-Artikels herangezogen werden.

{`$entry_id`}
: *Variablentyp:* Zahl
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält die ID des Blog-Artikels, zu dem die Kommentare eingebunden werden.

{`$comment_url`}
: *Variablentyp:* Zeichenkette
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält die URL der aktuellen Seite, die für das Ziel von Formularen oder Links eingesetzt werden kann.

{`$comment_string`}
: *Variablentyp:* Eindimensionales Array
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält ein Array mit Wortstücken. Dieses enthält Platzhalter, so dass die später notwendigen Variablen dazwischen eingebunden werden können. Die Wortstücke enthalten je nach Aktion des Besuchers entweder einen Hinweis zum erfolgreichen Kommentieren oder Fehlermeldungen.

{`$comment_entryurl`}
: *Variablentyp:* Zeichenkette
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält den Permalink zum Blog-Artikel, dessen Kommentare/Trackbacks angesehen werden.

{`$is_comment_added`}
: *Variablentyp:* Boolean
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält den Wert `true`, wenn ein Kommentar hinzugefügt wurde.

{`$is_showtrackbacks`}
: *Variablentyp:* Boolean
Verfügbar in: `commentpopup.tpl`
Zugewiesen in: `comment.php`
Enthält den Wert `true`, wenn innerhalb des Popup-Fensters Trackbacks angezeigt werden.

{$is_showcomments}
Variablentyp: Boolean
Verfügbar in: commentpopup.tpl
Zugewiesen in: comment.php
Enthält den Wert true, wenn innerhalb des Popup-Fensters Kommentare angezeigt werden.

{$is_comment_allowed}
Variablentyp: Boolean
Verfügbar in: commentpopup.tpl
Zugewiesen in: comment.php
Enthält den Wert true, wenn Kommentare zum jeweiligen Blog-Artikel zugelassen werden.

{$is_comment_notadded}
Variablentyp: Boolean
Verfügbar in: commentpopup.tpl
Zugewiesen in: comment.php
Enthält den Wert true, wenn ein Kommentar nicht hinzugefügt werden konnte.

{$is_comment_empty}
Variablentyp: Boolean
Verfügbar in: commentpopup.tpl
Zugewiesen in: comment.php
Enthält den Wert true, wenn ein Kommentar keinen Inhalt enthielt und daher nicht hinzugefügt werden konnte.

{$COMMENTFORM}
Variablentyp: Zeichenkette
Verfügbar in: commentpopup.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den geparsten HTML-Inhalt der Datei commentform.tpl zur Darstellung eines Kommentarformulars.

{$comments}
Variablentyp: Mehrdimensionales Array
Verfügbar in: comments.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_printComments()
Das Array {$comments} enthält die Liste der Kommentare zu einem Blog-Artikel und wird mittels der {${serendipity_printComments} -Smarty-Funktion innerhalb der entries.tpl-Template-Datei eingebunden.

{$comments.X.comment}
Variablentyp: Zeichenkette

Enthält den Kommentartext. HTML-Tags werden entfernt und etwaige Sonderzeichen zur Sicherheit maskiert.

{`$comments.X.body`}
Variablentyp: Zeichenkette
Enthält den ursprünglichen, unveränderten Kommentartext, wie er eingegeben wurde. HTML-Tags und etwaige Sonderzeichen sind nach wie vor enthalten.

{`$comments.X.url`}
Variablentyp: Zeichenkette
Enthält die Homepage des Kommentators.

{`$comments.X.link_delete`}
Variablentyp: Zeichenkette
Enthält eine URL, mit der Blog-Redakteure (die entsprechenden Rechte vorausgesetzt) einen Kommentar löschen können.

{`$comments.X.no_email`}
Variablentyp: Boolean
Enthält den Wert `true`, wenn die E-Mail-Adresse der Kommentatoren entfernt werden soll.

{`$comments.X.email`}
Variablentyp: Zeichenkette
Enthält die E-Mail-Adresse des Kommentators. Das @-Zeichen wird durch die Zeichenkette `[at]` ersetzt.

{`$comments.X.clear_email`}
Variablentyp: Zeichenkette
Enthält die ursprüngliche, unveränderte E-Mail-Adresse des Kommentators.

{`$comments.X.pos`}
Variablentyp: Zahl
Enthält eine fortlaufende Zahl, die die aktuelle Nummer eines Kommentars darstellt.

{`$comments.X.parent_id`}
Variablentyp: Zahl
Wenn sich ein Kommentator auf einen vorhergehenden Kommentar bezieht, enthält der Wert der Variable {`$comments.X.parent_id`} die ID dieses Kommentars.

{`$comments.X.trace`}
Variablentyp: Zeichenkette
Bei aufeinander bezogenen Kommentaren wird die Verschachtelungstiefe durch die Variable {`$comments.X.trace`} angegeben. Für den zweiten Kommentar unterhalb des dritten Kommentars würde diese Variable den Wert `3.2` enthalten.

{$comments.X.depth}
Variablentyp: Zahl
Enthält die aktuelle Verschachtelungstiefe für aufeinander bezogene Kommentare.

{$comments.X.author}
Variablentyp: Zeichenkette
Enthält den Namen des Kommentators.

{$comments.X.id} , {$comments.X.commentid}
Variablentyp: Zahl
Enthält eine eindeutige ID des Kommentars.

{$comments.X.authorid}
Variablentyp: Zahl
Enthält die ID des Autors, der den Blog-Artikel verfasst hat, auf welchen sich der Kommentar bezieht.

{$comments.X.entry_id} , {$comments.X.entryid}
Variablentyp: Zahl
Enthält die eindeutige ID des Blog-Artikels, auf den sich der Kommentar bezieht.

{$comments.X.title}
Variablentyp: Zeichenkette
Enthält den Titel des Blog-Artikels, auf den sich der Kommentar bezieht.

{$comments.X.timestamp}
Variablentyp: Zahl
Enthält die Uhrzeit (in Unix-Sekunden), zu der der Kommentar verfasst wurde.

{$comments.X.ctitle}
Variablentyp: Zeichenkette
Enthält eine etwaige Überschrift des Kommentars. Dieses Eingabefeld wird standardmäßig in Serendipity-Templates nur für Trackbacks verwendet.

{$comments.X.ip}
Variablentyp: Zeichenkette
Enthält die IP-Adresse des Kommentators.

{$comments.X.type}
Variablentyp: Zeichenkette
Gibt den Typ des Kommentars an. Dies kann entweder NORMAL (Kommentare), TRACKBACK oder PINGBACK sein.

{$comments.X.subscribed}
Variablentyp: Boolean
Ist auf 1 gesetzt, wenn der Verfasser eines Kommentars die Option **Bei Aktualisierung dieser Kommentare benachrichtigen** aktiviert hat.

{$trackbacks}
Variablentyp: Mehrdimensionales Array

Verfügbar in: `trackbacks.tpl`
Zugewiesen in: `include/functions_trackbacks.inc.php`, `serendipity_printTrackbacks()`
Das Array {`$trackbacks`} enthält die Liste der Trackbacks zu einem Blog-Artikel und wird mittels der Smarty-Funktion {`$serendipity_printTrackbacks`} innerhalb der `entries.tpl`-Template-Datei eingebunden.

Das Trackbacks-Array verfügt über ähnliche Array-Schlüssel wie das ab Seite 550 aufgeführte {`$comments`} -Array. Dabei sind folgende Schlüssel identisch: `id`, `entry_id`, `parent_id`, `timestamp`, `title`, `author`, `email`, `url`, `ip`, `body`, `type`, `subscribed`.

Hinzu kommt folgender Schlüssel:

{`$trackbacks.X.status`}
Variablentyp: Zeichenkette
Gibt an, ob ein Trackback bereits veröffentlicht (`approved`) ist oder sich noch in Freischaltung (`pending`) befindet.

commentform.tpl

Die Template-Datei `commentform.tpl` ist ein Formular zum Hinzufügen von Kommentaren. Dieses wird über die Smarty-Variable {`$COMMENTFORM`} sowohl innerhalb der Template-Datei `entries.tpl` als auch in `commentpopup.tpl` eingebunden.

{`$commentform_action`}
Variablentyp: Zeichenkette
Verfügbar in: `commentform.tpl`
Zugewiesen in: `include/functions_comments.inc.php`, `serendipity_displayCommentForm()`
Enthält die Ziel-URL, an die die Daten des Kommentarformulars gesendet werden.

{`$commentform_id`}
Variablentyp: Zahl
Verfügbar in: `commentform.tpl`
Zugewiesen in: `include/functions_comments.inc.php`, `serendipity_displayCommentForm()`
Enthält die ID des Artikels, für den das Kommentarformular eingebunden wird.

{`$commentform_name`}
Variablentyp: Zeichenkette
Verfügbar in: `commentform.tpl`
Zugewiesen in: `include/functions_comments.inc.php`, `serendipity_displayCommentForm()`
Enthält die Eingabe `Name` des Benutzers im Kommentarformular.

{$commentform_email}
Variablentyp: Zeichenkette
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält die Eingabe E-Mail des Benutzers im Kommentarformular.

{$commentform_url}
Variablentyp: Zeichenkette
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält die Eingabe Homepage des Benutzers im Kommentarformular.

{$commentform_remember}
Variablentyp: Boolean
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Wert true, wenn der Benutzer die Option **Daten merken?** im Kommentarformular aktiviert hat, um seine Stammdaten in einem Cookie zu speichern und bei Folgekommentaren nicht erneut eintragen zu müssen.

{$commentform_subscribe}
Variablentyp: Zeichenkette
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Wert true, wenn der Benutzer die Option **Bei Aktualisierung dieser Kommentare benachrichtigen** im Kommentarformular aktiviert hat.

{$commentform_replyTo}
Variablentyp: Zeichenkette
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den HTML-Code zur Darstellung eines Auswahlfeldes mit allen bereits bestehenden Kommentaren eines Artikels. Der kommentierende Benutzer kann darin auswählen, auf welchen Kommentar er sich beziehen möchte.

{$commentform_data}
Variablentyp: Zeichenkette
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Kommentartext des Benutzers.

{$is_commentform_showToolbar}
Variablentyp: Boolean
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Wert true, wenn die beiden Sonderoptionen **Daten merken** und **Bei Aktualisierung dieser Kommentare benachrichtigen** angezeigt werden sollen.

{$is_allow_Subscriptions}
Variablentyp: Boolean
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Wert true, wenn **Bei Aktualisierung dieser Kommentare benachrichtigen** angezeigt werden soll. Diese Einstellung kann über die globale Serendipity-Konfiguration (siehe Seite 164) vorgenommen werden.

{$is_moderate_comments}
Variablentyp: Boolean
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält den Wert true, wenn Kommentare des aktuellen Artikels moderiert werden.

{$commentform_entry}
Variablentyp: Mehrdimensionales Array
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_displayCommentForm()
Enthält die Metadaten des Blog-Artikels (siehe Seite 521).

comments_by_author.tpl

Die Template-Datei comments_by_author.tpl wird ausgegeben, wenn eine Übersichtsseite der Kommentare unabhängig von Ihren Blog-Artikeln dargestellt wird (siehe Seite 82). Die Template-Datei wird über die Variable {$ENTRIES} in die Template-Datei content.tpl eingebunden.

{$comments_by_authors}
Variablentyp: Mehrdimensionales Array
Verfügbar in: commentform.tpl
Zugewiesen in: include/functions_comments.inc.php, serendipity_printCommentsByAuthor()

Enthält die Daten der Kommentare, die in der Übersicht dargestellt werden sollen. Der Schlüssel der ersten Dimension des Arrays enthält die ID des Blog-Artikels, auf die sich ein Kommentar bezieht. Als Wert enthält die zweite Dimension ein Unter-Array mit den Daten des ersten Kommentars, der zu der ID des Blog-Artikels gehört. Die Array-Schlüssel sind dabei identisch zu dem ab Seite 550 aufgeführten {`$comments`} -Array. Dabei sind folgende Schlüssel identisch: `id, entry_id, parent_id, timestamp, title, author, email, url, ip, body, type, subscribed`.

Zusätzlich verfügt das Array über den Schlüssel {`$comments_by_authors.X.link`} , das den Permalink zum Blog-Artikel enthält. Ein weiterer Schlüssel {`$comments_by_authors.comme` enthält ein Array mit allen Kommentaren zu diesem Blog-Artikel. Dabei ist der erste Kommentar, der sich bereits in den Werten der zweiten Dimension wiederfindet, nochmals enthalten. Das Array der zweiten Dimension dient grundsätzlich nur der Speicherung der Metadaten zu einem Artikel, da diese mit den Stammdaten eines Kommentars vermischt sind.

Zuletzt enthält der Schlüssel {`$comments_by_authors.X.tpl_comments`} jeweils den vollständigen HTML-Code des zu einem Blog-Artikel geparsten `comments.tpl`-Templates. Da durch diese Einbindung des `comments.tpl`-Templates Redundanzen vermieden werden können, enthält die Template-Datei `comments_by_authors.tpl` nur verhältnismäßig wenig HTML/Smarty-Code.

plugin_calendar.tpl

Das Seitenleisten-Plugin **Kalender** gibt seinen HTML-Code über das Smarty-Template `plugin_calendar.tp` aus. Folgende Variablen sind dort verfügbar:

{`$plugin_calendar_weeks`}
Variablentyp: Mehrdimensionales Array
Verfügbar in: `plugin_calendar.tpl`
Zugewiesen in: `include/plugin_internal.inc.php`, `serendipity_calendar_plugin::generate_content()`
Enthält ein mehrdimensionales Array mit den Daten des jeweiligen dargestellten Monats.

Das Array gibt sozusagen eine Tages-Matrix wieder, besteht also aus Zeilen und Spalten. Die Array-Schlüssel der ersten Dimension entsprechen dabei der Zeile, die jeweils eine ganze Kalenderwoche enthält. Das heißt, {`$plugin_calendar_weeks.0`} enthält die ersten 7 Tage, {`$plugin_calendar_weeks.1`} die zweiten 7 Tage und so fort bis zum Ende des jeweiligen Monats.

Jedes dieser Arrays, die einer Kalenderzeile entsprechen, besitzt einen Array-Schlüssel {`$plugin_calendar_weeks.0.days`} , in dem ein durchnummeriertes Unter-Array steckt. Dies verfügt jeweils über die Array-Schlüssel `name` (Tag), `properties` (spezielle Attribute) und

`classes` (zugewiesene CSS-Klasse). Folgende Attribute (`properties`) können für jeden Tag zugewiesen werden:

{`$plugin_calendar_weeks.0.days.0.properties.FirstRow`}
ist auf `1` gesetzt, wenn der aktuelle Tag in der ersten Kalenderzeile steht.

{`$plugin_calendar_weeks.0.days.0.properties.FirstInRow`}
ist auf `1` gesetzt, wenn der aktuelle Tag in der ersten Kalenderzeile und an erster Stelle der Zeile steht.

{`$plugin_calendar_weeks.0.days.0.properties.LastRow`}
ist auf `1` gesetzt, wenn der aktuelle Tag in der letzten Kalenderzeile steht.

{`$plugin_calendar_weeks.0.days.0.properties.LastInRow`}
ist auf `1` gesetzt, wenn der aktuelle Tag in der letzten Kalenderzeile und an letzter Stelle der Zeile steht.

{`$plugin_calendar_weeks.0.days.0.properties.Today`}
ist auf `1` gesetzt, wenn der jeweilige Tag dem heutigen Datum entspricht.

{`$plugin_calendar_weeks.0.days.0.properties.Title`}
enthält ggf. einen Termin, wenn das via Spartacus erhältliche Ereignis-Plugin **Mein Kalender** seine Termine einbettet.

{`$plugin_calendar_weeks.0.days.0.properties.Active`}
ist gesetzt, wenn der aktuelle Tag des Kalenders ausgewählt wurde.

{`$plugin_calendar_weeks.0.days.0.properties.Link`}
enthält die URL zu der Artikelübersicht des jeweiligen Tages.

Die Liste der zugewiesenen CSS-Klassen richtet sich nach den `properties` des Tages; diese können daher später in der Template-Datei direkt innerhalb eines `class='...'`-Attributs eingesetzt werden.

Aufgrund der hohen Verschachtelungstiefe des Kalenders werden die Variablen in mehrfach verschachtelten {`${foreach}`} -Schleifen abgearbeitet.

{`$plugin_calendar_dow`}
Variablentyp: Mehrdimensionales Array
Verfügbar in: `plugin_calendar.tpl`
Zugewiesen in: `include/plugin_internal.inc.php`, `serendipity_calendar_plugin::generate_content()`
Da abhängig von regionalen Gepflogenheiten die Woche entweder an einem Sonntag oder Montag anfängt, muss der Kalender dies berücksichtigen. Dafür steht das Array {`$plugin_calendar_dow`} zur Verfügung. Es enthält das jeweilige Datum zum zugehörigen Wochentag.

Die Array-Schlüssel der ersten Dimension sind aufeinanderfolgend durchnummeriert (von 1 bis 7). Der Array-Schlüssel `date` der zweiten Dimension enthält als Wert das jeweilige Datum (in Unix-Sekunden).

{$plugin_calendar_head}
Variablentyp: Mehrdimensionales Array
Verfügbar in: plugin_calendar.tpl
Zugewiesen in: include/plugin_internal.inc.php, serendipity_calendar_plugin::generate_content()
Enthält einige Stammdaten des Kalenders:

{$plugin_calendar_head.month_date}
Variablentyp: Zahl
Enthält das Datum (in Unix-Sekunden) des dargestellten Monats.

{$plugin_calendar_head.minScroll} ,

{$plugin_calendar_head.maxScroll} *Variablentyp:* Zahl
Enthält das Datum (in Unix-Sekunden) des ersten und letzten Blog-Artikels, so dass die Anzeige des Kalenders den Zeitraum vorhandener Artikel nicht über- oder unterschreiten kann.

{$plugin_calendar_head.uri_previous} ,

{$plugin_calendar_head.uri_next} *Variablentyp:* Zeichenkette
Enthält die URLs zum Zurück- und Weiterblättern des aktuellen Kalenderblatts.

{$plugin_calendar_head.uri_month}
Enthält die URL für den aktuellen Monat. *Variablentyp:* Zeichenkette

plugin_categories.tpl

Die Template-Datei plugin_categories.tpl wird zur Ausgabe des Seitenleisten-Plugins **Kategorien** verwendet, falls die Option **Smarty benutzen** dort aktiviert ist. Bei deaktivierter Smarty-Option gibt das Plugin direkt unveränderbaren HTML-Code aus (der jedoch Performance spart).

{$is_form}
Variablentyp: Boolean
Verfügbar in: plugin_categories.tpl
Zugewiesen in: include/plugin_internal.inc.php, serendipity_categories_plugin::generate_content()
Ist auf true gesetzt, wenn das Seitenleisten-Plugin so konfiguriert wurde, dass der Besucher mehrere Kategorien gleichzeitig (mittels Ankreuzfeldern) auswählen darf.

{$category_image}
Variablentyp: Zeichenkette

Verfügbar in: `plugin_categories.tpl`
Zugewiesen in: `include/plugin_internal.inc.php`, `serendipity_categories_plugin::generate_content()`
Enthält die URL zum Bild für die kleinen Feed-Symbole, die optional neben einer Kategorie eingeblendet werden können.

{`$form_url`}
Variablentyp: Zeichenkette
Verfügbar in: `plugin_categories.tpl`
Zugewiesen in: `include/plugin_internal.inc.php`, `serendipity_categories_plugin::generate_content()`
Falls der Benutzer mehrere Kategorien gleichzeitig auswählen kann, enthält die Variable {`$form_url`} das Ziel des HTML-Formulars zur Darstellung der Artikel der gewählten Kategorien.

{`$categories`}
Variablentyp: Mehrdimensionales Array
Verfügbar in: `plugin_categories.tpl`
Zugewiesen in: `include/plugin_internal.inc.php`, `serendipity_categories_plugin::generate_content()`
Enthält die Liste der Kategorien des Blogs. Die Array-Schlüssel der zweiten Dimension entsprechen den auf Seite 530 aufgeführten. Hinzu kommen folgende Schlüssel:

{`$categories.X.feedCategoryURL`}
Variablentyp: Zeichenkette
Enthält die URL zum RSS-Feed einer Kategorie.

{`$categories.X.categoryURL`}
Variablentyp: Zeichenkette
Enthält die URL für die Übersichtsseite aller Blog-Artikel einer Kategorie.

{`$categories.X.paddingPx`}
Variablentyp: Zahl
Enthält abhängig von der Verschachtelungstiefe einer Kategorie eine Zahl, die der notwendigen Einrückung entspricht. Pro Verschachtelungsebene werden dafür 6 Einheiten (Pixel) vorgesehen.

{`$categories.X.category_name`}
Variablentyp: Zeichenkette
Falls im Seitenleisten-Plugin die Option zur Zählung der Artikel pro Kategorie aktiviert wurde, enthält diese Variable den Namen sowie in Klammern gesetzt die Anzahl der Artikel.

{`$categories.X.true_category_name`}
Variablentyp: Zeichenkette
Falls im Seitenleisten-Plugin die Option zur Zählung der Artikel pro Kategorie aktiviert wurde, enthält diese Variable den ursprünglichen Kategorienamen, ohne Nennung der Kategorieanzahl in Klammern.

plugin_remoterss.tpl

Die Template-Datei `plugin_remoterss.tpl` wird zur Ausgabe des Seitenleisten-Plugins **Fremder RSS/OPML-Blogroll Feed** (siehe Seite 210) verwendet, falls die Option **Smarty benutzen** dort aktiviert ist. Bei deaktivierter Smarty-Option gibt das Plugin direkt unveränderbaren HTML-Code aus (der jedoch Performance spart).

{`$remoterss_items`}
Variablentyp: Verschachteltes Array
Verfügbar in: `plugin_remoterss.tpl`
Zugewiesen in: `plugins/serendipity_plugin_remoterss/serendipity_plugin_remoter`
`serendipity_plugin_remoterss::generate_content()`
Enthält ein mehrfach verschachteltes Array mit allen Ausgaben und Optionen des Plugins. Das Array besitzt folgende Schlüssel:

{`$remoterss_items.items`}
Variablentyp: Verschachteltes Array
Enthält ein Array mit dem Inhalt des fremden RSS/OPML-Feeds. Pro durchnummeriertem Array-Schlüssel liegt ein RSS-Artikel vor, der wiederum nach den XML-Tags des Feeds benannte Unterschlüssel besitzt (`link`, `title`, `comments`, `author`, `description`, `pubDate`, `guid` ...) sowie folgende vom Plugin vergebene Schlüssel:

{`$remoterss_items.items.X.timestamp`} (Zahl)
enthält den Unix-Zeitstempel, wann der RSS-Artikel veröffentlicht wurde.

{`$remoterss_items.items.X.css_class`} (Zeichenkette)
enthält einen CSS-Klassennamen des RSS-Elements, das in der Plugin-Konfiguration als **RSS Zielelement** festgelegt wurde.

{`$remoterss_items.items.X.decoded_...`} (Zeichenkette)
ist für jedes XML-Element des RSS-Feeds gesetzt und enthält den in den korrekten Zeichensatz konvertierten Wert. {`$remoterss_items.items.X.decoded_title`} enthält beispielsweise das XML-Element `title` im Zeichensatz des Blogs. Taucht im XML-Elementnamen ein Doppelpunkt auf (wie bei `<content:encoded>`), wird dieser durch einen Unterstrich ersetzt ({`$remoterss_items.items.X.decoded_conten`
).

{`$remoterss_items.items.X.display_elements`} (Array)
enthält eine Liste der RSS-XML-Elemente, die der Benutzer als **RSS Zielelemente** eingetragen hat. In der Smarty-Datei wird dieses Array dafür verwendet, die darzustellenden RSS-Elemente einzubinden. Sie können jedoch auch auf den Einsatz dieses Arrays verzichten, wenn Sie die einzubindenden RSS-Zielelemente direkt über das Smarty-Template hartkodiert auslesen (z. B. über {`$remoterss_items.items.X.`
).

{`$remoterss_items.use_rsslink`}
Variablentyp: Boolean

Enthält den Wert `true`, wenn jeder RSS-Artikel zum Originallink führen soll. Wird durch die Konfigurationsoptionen des Plugins bestimmt.

{`$remoterss_items.bulletimg`}
Variablentyp: Zeichenkette
Enthält die URL einer Symbolgrafik, wenn diese zu jedem RSS-Artikel dargestellt werden soll. Wird durch die Konfigurationsoptionen des Plugins bestimmt.

{`$remoterss_items.escape_rss`}
Variablentyp: Boolean
Enthält den Wert `true`, wenn die Sonderzeichen der RSS-Artikel in HTML-Sonderzeichen umgewandelt werden sollen, so dass kein HTML oder JavaScript ausgeführt werden kann. Wird durch die Konfigurationsoptionen des Plugins bestimmt.

{`$remoterss_items.displaydate`}
Variablentyp: Boolean
Enthält den Wert `true`, wenn das Datum eines RSS-Artikels angezeigt werden soll. Wird durch die Konfigurationsoptionen des Plugins bestimmt.

{`$remoterss_items.dateformat`}
Variablentyp: Zeichenkette
Enthält das Format, in dem das Datum eines RSS-Artikels dargestellt werden soll. Wird durch die Konfigurationsoptionen des Plugins bestimmt.

{`$remoterss_items.target`}
Variablentyp: Zeichenkette
Enthält den **Link-Target** (siehe Konfigurationsoptionen des Plugins, Seite 213) des RSS-Feeds.

preview_iframe.tpl

Die Template-Datei `preview_iframe.tpl` wird verwendet, um innerhalb des Backends beim Bearbeiten/Erstellen eines Blog-Artikels eine Vorschau anzuzeigen, die der Darstellung im Frontend gleicht.

Dabei wird die `entries.tpl`-Template-Datei wie gewohnt geparst und deren Ausgabe innerhalb der `preview_iframe.tpl` eingebunden.

{`$preview`}
Variablentyp: Zeichenkette
Zugewiesen in: `include/functions_config.inc.php`, `serendipity_iframe()`
Enthält den HTML-Code für die Vorschau des Eintrags.

content.tpl

Über die Template-Datei `content.tpl` wird das zentrale Serendipity-Frontend zusammengebaut. Abhängig von der darzustellenden Seite enthält es einen einzelnen Artikel, Artikelübersichten, Suchergebnisse oder anderes. Grundsätzlich dient die Datei dabei nur zur Einbindung weiterer geparster Template-Dateien, die über folgende Variablen verfügbar sind:

{`$ENTRIES`}
: *Variablentyp:* Zeichenkette
Zugewiesen in: `include/functions_entries.inc.php`, `serendipity_printArchives()`
Enthält den HTML-Code für die Artikelübersicht oder einzelne Artikel. Die Inhalte dieser Variable stammen aus der geparsten Template-Datei `entries.tpl`.

{`$ARCHIVES`}
: *Variablentyp:* Zeichenkette
Zugewiesen in: `include/functions_entries.inc.php`, `serendipity_printArchives()`
Enthält den HTML-Code für die chronologische Artikelübersicht. Die Inhalte dieser Variable stammen aus der geparsten Template-Datei `entries_archives.tpl`.

entries_archives.tpl

Mittels der Template-Datei `entries_archives.tpl` werden chronologische Artikelübersichten dargestellt. Im Gegensatz zur vollständigen Artikelübersicht enthält diese Übersicht nur Titel und Datum der Einträge.

{`$archives`}
: *Variablentyp:* Mehrdimensionales Array
Zugewiesen in: `include/functions_entries.inc.php`, `serendipity_printArchives()`
Enthält das Array mit den in der Übersicht darzustellenden Einträgen. Dieses mehrfach verschachtelte Array enthält in der ersten Dimension als Array-Schlüssel das Jahr, unter dem die jeweiligen Unter-Arrays gegliedert sind.

  In der zweiten Dimension stehen die Schlüssel `year` und `months` zur Verfügung, die jeweils das Jahr und den Monat für die Artikelliste enthalten. Innerhalb des `months`-Unter-Arrays befindet sich ein durchnummeriertes Array mit folgenden Schlüsseln:

  {`$archives.X.months.Y.entry_count`}
  : *Variablentyp:* Zahl
  Enthält die Anzahl der Einträge für den jeweiligen Monat eines Jahres.

{$archives.X.months.Y.link}
Variablentyp: Zeichenkette
Enthält die URL, die die Übersichtsseite der Artikel des gewählten Monats anzeigt.

{$archives.X.months.Y.link_summary}
Variablentyp: Zeichenkette
Enthält die URL, die eine zusammenfassende Übersichtsseite der Artikel des gewählten Monats anzeigt. In der Zusammenfassung werden nur Titel und Datum der Artikel aufgeführt.

{$archives.X.months.Y.date}
Variablentyp: Zeichenkette
Enthält das Datum (in Unix-Sekunden, gerundet auf den vollen Tag) für die Liste der Artikel des zugehörigen Monats.

{$max_entries}
Variablentyp: Zahl
Zugewiesen in: include/functions_entries.inc.php, serendipity_printArchives()
Enthält die höchste Zahl von Artikeln, die in der Übersichtsseite für ein Monatssegment auftaucht. Diese Zahl wird verwendet, um eine relative Grafik einzubinden, anhand derer man die Häufigkeitsverteilung der Artikel ablesen kann.

9.6.3 Smarty-Funktionen

Auf den folgenden Seiten sind alle Smarty-Funktionen aufgeführt, die Serendipity innerhalb der *.tpl-Template-Dateien zur Verfügung stellt.

Der Aufruf einer Funktion kann mit einigen Parametern ausgestattet werden, die im Funktionsaufruf wie {serendipity_funktion parameter1=wert1 parameter2=wert2 ...} eingetragen werden.

{$serendipity_printSidebar}
Zugewiesen in: include/functions_smarty.inc.php
Stellt die Ausgaben aller Seitenleisten-Plugins für die gewünschte Seitenleiste im Frontend zusammen.

Parameter:

side (Pflicht)
Enthält den Namen der auszugebenden Seitenleiste (left, right, hidden ...).

Rückgabewert: HTML-Zeichenkette (sidebars.tpl)

{$serendipity_hookPlugin}
Zugewiesen in: include/functions_smarty.inc.php

Über die Funktion {$serendipity_hookPlugin} kann ein internes Ereignis (Hook) ausgeführt werden. Dies führt dazu, dass Serendipity alle installierten Ereignis-Plugins durchgeht und prüft, ob das jeweilige Plugin für den gewünschten Hook eine Funktionalität ausführen möchte. Die Rückgaben dieser Funktionalität (meist HTML-Ausgaben) werden daraufhin direkt ausgegeben.

Diese Funktion dient beispielsweise dazu, um die Ausgaben von zusätzlichen Ereignis-Plugins in den HTML-Kopfbereich (bspw. Meta-Tags) umzusetzen, aber auch für zahlreiche andere interne Serendipity-Funktionen, wo sich Plugins einklinken können.

Intern ruft diese Funktion die Serendipity-API-Funktion serendipity_plugin_api::hook_event(auf.

hook (Pflicht)
: Enthält den Namen des auszuführenden Ereignisses (Hook). Standardmäßig sind hier nur die Hooks frontend_header, entries_header, entries_footer, frontend_comment oder frontend_footer zulässig.

hookAll (optional)
: Wenn dieser Parameter auf den Wert true gesetzt wird, können auch beliebige andere Hooks als die oben aufgeführten ausgeführt werden. So können Sie selbständig Hooks zu Serendipity-Templates hinzufügen und mit eigenen oder angepassten Ereignis-Plugins ausführen.

data (optional)
: Enthält ein beliebiges Array mit Daten, die an das Ereignis-Plugin weitergeleitet werden. Diese Daten sind für jeden Hook unterschiedlich. Die übergebenen Daten werden von einem Plugin zum nächsten als referenzierte Variable weitergegeben und können daher von Plugins beliebig modifiziert werden. Die Variable verliert nach dem Durchlauf der zugehörigen Plugins ihren Wert. Falls die Variable jedoch später nochmal verarbeitet werden soll, wird der Einsatz des Smarty-Modifiers {serendipity_refhookPlugin} (siehe Seite 575) empfohlen.

addData (optional)
: Enthält ein beliebiges Array mit Daten, die zusätzlich an das Ereignis-Plugin weitergeleitet werden. Im Gegensatz zu den Daten aus data können Plugins auf addData nur lesend zugreifen.

Rückgabewert: HTML-Zeichenkette (falls vorhanden)

{$serendipity_showPlugin}
Zugewiesen in: include/functions_smarty.inc.php

Gibt die Inhalte eines installierten Seitenleisten-Plugins aus. Dies kann unabhängig von der gewohnten Darstellung innerhalb der Seitenleiste erfolgen, so dass ein Seitenleisten-Plugin an beliebiger Stelle in einem Template platziert werden kann.

Diese Smarty-Funktion kann entweder ein ganz spezielles Plugin ausgeben oder auch mehrere Plugins derselben Klasse (z. B. alle HTML-Klötze).

Parameter:

`class` (Pflicht)
: Enthält den PHP-Klassennamen des auszugebenden Plugins, z. B. `serendipity_quicksearch_plugin` oder `serendipity_plugin_amazon`. Die Ausgabe beinhaltet alle zutreffenden Plugin-Instanzen dieser Klasse.

`id` (Pflicht)
: Wenn der PHP-Klassenname nicht über `class` angegeben wird, muss die ID des auszugebenden Plugins mittels dieses Parameters gesetzt werden. Als ID wird die vollständige Zeichenkette des Plugins, wie sie innerhalb der `serendipity_plugins`-Datenbanktabelle erscheint, eingefügt. Beispiel: `id="@serendipity_quicksearch_plugin:3embaf334l rvo3693f`

`side` (optional)
: Seitenleisten-Plugins sind üblicherweise einer direkten Seitenleiste (links, rechts, versteckt ...) zugewiesen. Falls Sie nur Plugins einer bestimmten Leiste ausgeben wollen, können Sie den Namen dieser Leiste (`left, right, hidden`) mit dem Parameter `side` festlegen. Standardmäßig ist dieser Wert auf `*` gesetzt und liest alle Seitenleisten aus.

`negate` (optional)
: Wenn dieser Wert auf `true` gesetzt wird, kehrt dies die Auswahl der auszugebenden Seitenleisten-Plugins um. Anstatt nur die Plugins einer bestimmten Klasse auszugeben, werden stattdessen alle Plugins *außer* der festgelegten Klasse/ID ausgegeben.

Rückgabewert: HTML-Zeichenkette (`sidebars.tpl`)

{`$serendipity_getFile`}
: *Zugewiesen in:* `include/functions_smarty.inc.php`
Liefert den vollständigen Pfad zum Template-Verzeichnis einer beliebigen Datei. Mithilfe dieser Funktion können die URLs für Grafiken und Weiteres automatisch und template-unabhängig zusammengebaut werden, so dass Sie keine absoluten URLs in einer Smarty-Template-Datei einbinden müssen.

Liegt `header.gif` im Verzeichnis `templates/bulletproof/img` und wollen Sie diese in der Template-Datei `index.tpl` einbauen, müssen Sie nicht `<img src="/blog/templates/bulletproof/img/header.gif" />` eintragen, sondern `<img src="`{`serendipity_getFile file="img/header.gif"`}`"`

/>. Dadurch kann das Template später auch mit anderen Verzeichnisstrukturen eingesetzt werden.

Parameter:

`file` (Pflicht)
: Hiermit geben Sie an, welchen Dateinamen (und etwaige Unterverzeichnisse ab dem Stammverzeichnis des Templates) Sie im Template einbinden wollen.

Rückgabewert: Zeichenkette (vollständiger Pfad zum gewünschten Bild)

{`$serendipity_printComments`}
Zugewiesen in: `include/functions_smarty.inc.php`
Stellt die Liste aller Kommentare für einen Eintrag dar.

Parameter:

`entry` (Pflicht)
: Enthält die ID des Blog-Artikels, für den die Kommentare dargestellt werden sollen.

`mode` (optional)
: Legt fest, ob die Artikel hierarchisch (`$CONST.VIEWMODE_THREADED`) oder chronologisch ((`$CONST.VIEWMODE_LINEAR`) dargestellt werden sollen.

`order` (optional)
: Legt die Reihenfolge fest, mit der die Kommentare ausgegeben werden (`ASC` für aufsteigende Sortierung, `DESC` für absteigende).

`limit` (optional)
: Begrenzt die Anzahl der dargestellten Kommentare auf den gewünschten Wert (standardmäßig alle Kommentare).

Rückgabewert: HTML-Zeichenkette (`comments.tpl`)

{`$serendipity_printTrackbacks`}
Zugewiesen in: `include/functions_smarty.inc.php`
Stellt die Liste aller Trackbacks für einen Eintrag dar.

Parameter:

`entry` (Pflicht)
: Enthält die ID des Blog-Artikels, für den die Trackbacks dargestellt werden sollen.

Rückgabewert: HTML-Zeichenkette (`trackbacks.tpl`)

{`$serendipity_rss_getguid`}
Zugewiesen in: `include/functions_smarty.inc.php`

Wird von Feed-Template-Dateien (`feed.tpl`) verwendet, um einen eindeutigen Permalink zu einem einzelnen Blog-Artikel zu erzeugen und zurückzuliefern.

Parameter:

`entry` (Pflicht)
: Enthält die ID des Blog-Artikels, zu dem der Permalink zurückgegeben werden soll.

`is_comments` (Pflicht)
: Bei der Darstellung von Kommentar-Feeds muss dieser Parameter auf `true` gesetzt werden, damit der korrekte Permalink zum zugehörigen Blog-Artikel bezogen werden kann.

Rückgabewert: Zeichenkette (Link zum Blog-Artikel)

{`$serendipity_getTotalCount`}
: *Zugewiesen in:* `include/functions_smarty.inc.php`
Kann die gesamte Anzahl an Blog-Artikeln, Kommentaren oder Trackbacks (abhängig vom Parameter `what`) darstellen.

Parameter:

`what` (Pflicht)
: Kann entweder `entries` (Blog-Artikel) , `trackbacks` oder `comments` als Wert enthalten und bestimmt, welche Zahl zurückgeliefert wird.

Rückgabewert: Zahl

{`$serendipity_pickKey`}
: *Zugewiesen in:* `include/functions_smarty.inc.php`
Da Smarty-Arrays von der Notation der PHP-Arrays abweichen, kann man innerhalb der Smarty-Syntax keine Array-Schlüssel angeben, die in einer weiteren Variable benannt sind. Folgendes wäre daher nicht möglich:

```
Eintrag: {$entries.$latest_entry}
```

Stattdessen wurde die Serendipity-Smarty-Funktion `serendipity_pickKey` eingeführt. Mit dieser kann ein gewünschter Array-Wert zurückgeliefert werden:

```
Eintrag: {serendipity_pickKey array=$entries key=$latest_entry}
```

Parameter:

`array` (Pflicht)
: Enthält das Array, aus dem ein Wert zurückgeliefert werden soll.

`key` (Pflicht)
: Enthält den Namen des Array-Schlüssels, zu dem der Wert zurückgeliefert werden soll.

`default` (optional)
: Falls kein Array-Schlüssel gefunden wird, kann anstelle eines leeren Wertes ein anderer Standard-Array-Schlüssel zurückgeliefert werden.

Rückgabewert: Array-Wert

{$serendipity_showCommentForm}

Zugewiesen in: include/functions_smarty.inc.php

Stellt das Kommentarformular für einen Beitrag dar.

Parameter:

id oder entry (Pflicht)
: Enthält die ID des Blog-Artikels, für den das Kommentarformular dargestellt werden soll.

url (optional)
: Enthält die URL, an die die Inhalte des Kommentarformulars übermittelt werden sollen. Zeigt standardmäßig auf die Detailseite des Blog-Artikels.

comments (optional)
: Kann ein Array mit bereits vorhandenen Kommentaren zum Blog-Artikel enthalten. Falls nicht angegeben, wird dies automatisch aus der Datenbank bezogen.

data (optional)
: Enthält die Eingaben des Benutzers im Kommentarformular (POST-Daten).

showToolbar (optional)
: Wenn auf true gesetzt, werden zusätzliche Felder im Kommentarformular angezeigt, mit dem ein Benutzer bestimmt, ob zusätzliche Felder eingebunden werden sollen. Wird als Smarty-Variable {$is_commentform_showToolbar} weitergereicht (siehe Seite 555).

moderate_comments (optional)
: Gibt an, ob Kommentare zu einem Blog-Artikel moderiert werden. Standardmäßig wird diese Variable entsprechend der Einstellung des jeweiligen Blog-Artikels gesetzt.

Rückgabewert: HTML-Zeichenkette (commentform.tpl)

{$serendipity_fetchPrintEntries}

Zugewiesen in: include/functions_smarty.inc.php

Mittels der Smarty-Funktion kann eine Liste von Blog-Einträgen innerhalb beliebiger Smarty-Template-Dateien erzeugt werden. So können auch mehrere Eintragsauflistungen pro Seite eingefügt werden.

Intern ist die Funktion recht komplex gebaut. Anhand der Funktionsparameter werden die gewünschten Blog-Einträge ausgelesen, ausgewertet und dann über eine beliebige Template-Datei (standardmäßig entries.tpl) geparst und zurückgeliefert.

Parameter:

category (optional)
: Legt die Kategorie-ID fest, aus der die Blog-Artikel gelesen werden sollen. Ohne eine Angabe werden alle Kategorien ausgelesen. Mehrere Kategorien können mittels Semikolon voneinander getrennt werden.

`viewAuthor` (optional)
: Legt die Redakteurs-ID fest, vom der die Blog-Artikel gelesen werden sollen. Ohne eine Angabe werden die Blog-Artikel aller Redakteure berücksichtigt. Mehrere IDs können mittels Semikolon voneinander getrennt werden.

`page` (optional)
: Wenn die Liste der Blog-Artikel mehr als eine Seite ausfüllt[13], bestimmt der Parameter `page` die Seitennummer, die aktuell angezeigt wird. Sollten insgesamt 20 Artikel gefunden und 10 Artikel pro Seite dargestellt werden, dann würde der Wert `2` im `page`-Parameter dafür sorgen, dass die Artikel 11-20 angezeigt werden.

`id` (optional)
: Falls ein einzelner Artikel angezeigt werden soll, enthält dieser Parameter die ID des Blog-Eintrags.

`range` (optional)
: Kann die Liste der Artikel auf einen gewissen Zeitraum einschränken. Der Parameter kann entweder einen Wert wie JJJJMMTT (Jahr, Monat, Tag) enthalten, um nur die Artikel eines Tages darzustellen (z. B. `20071224` für den 24.12.2007). Ein ganzer Monat wird über einen Wert wie `20071200` angezeigt.
  Alternativ kann der Parameter ein Array enthalten, dessen Array-Schlüssel 0 und 1 jeweils einen Unix-Zeitstempel für den Start- und Endzeitpunkt bestimmen.
  Wird die Variable leer gelassen oder auf `null` gesetzt, werden die aktuellsten Einträge angezeigt.

`modified_since` (optional)
: Falls der Parameter `range` leer gelassen wird oder auf `null` gesetzt ist, kann der Parameter einen Unix-Zeitstempel enthalten, der das Datum angibt, ab dem alle neuen Artikel ausgelesen werden sollen.

`full` (optional)
: Wenn dieser Parameter auf `true` gesetzt wird, kann auch der erweiterte Eintrag der Blog-Artikel ausgelesen werden. Dies ist besonders wichtig in Kombination mit dem `id`-Parameter.

`limit` (optional)
: Enthält eine Zeichenkette, die angibt, wie viele Blog-Artikel ausgelesen werden sollen. Diese Zeichenkette kann entweder eine einzelne Zahl oder eine Angabe `X, Y` enthalten, wobei `X` für den Index des Artikels steht, ab dem Artikel angezeigt werden, und `Y` für die Anzahl der zu lesenden Beiträge. `limit='5, 10'` würde 10 Artikel ausgeben und die ersten 5 Artikel überspringen.
  Wird dieser Parameter nicht angegeben, werden so viele Artikel ausgelesen, wie in der Konfiguration des Blogs standardmäßig festgelegt.

[13] Die Anzahl der Artikel pro Seite wird bestimmt durch den `limit`-Parameter.

`fetchDrafts` (optional)
: Wenn dieser Parameter auf `true` gesetzt wird, können auch Blog-Entwürfe ausgelesen werden. Standardmäßig werden nur veröffentlichte Artikel berücksichtigt.

`short_archives` (optional)
: Wenn auf `true` gesetzt, erfolgt die Ausgabe der Blog-Artikel nicht wie auf den normalen Übersichtsseiten, sondern in einer zusammengefassten Ansicht wie bei den chronologischen Archivseiten.

`orderby` (optional)
: Legt die Sortierung der Artikel fest. Dieser Parameter legt den Namen des Datenbankschlüssels der involvierten Tabellen fest und entspricht dem SQL `ORDER BY`-Statement. Standardmäßig wird nach dem Veröffentlichungsdatum eines Artikels sortiert (`e.timestamp`).

`filter_sql` (optional)
: Um die Eingrenzung der gewünschten Blog-Artikel feinzusteuern, kann beliebiger SQL-Code eingefügt werden, der in der Datenbankabfrage in den `WHERE`-Teil integriert wird.

`noCache` (optional)
: Wenn auf `true` gesetzt, werden etwaige Caching-Plugins für das Auslesen der Blog-Artikel deaktiviert. Dies kann in Problemfällen dabei helfen, die Artikel korrekt auszulesen.

`noSticky` (optional)
: Wenn auf `true` gesetzt, werden sogenannte *Dauerhafte Einträge* (siehe Seite 263) nicht mit ausgelesen. Wenn aktuelle Einträge jedoch als *Dauerhaft* markiert werden, sind sie trotzdem in der Liste vorhanden – lediglich ältere dauerhafte Einträge werden nicht wie sonst üblich mit aufgenommen.

`template` (optional)
: Legt den Namen der Smarty-Template-Datei fest, der für die Darstellung der Einträge verwendet werden soll. Standardmäßig ist dies `entries.tpl`.

`block` (optional)
: Legt den Namen der Smarty-Variable fest, die den HTML-Code des geparsten Templates enthält. Standardmäßig ist dies `ENTRIES`, gemäß der zentralen Smarty-Variable {`$ENTRIES`} , die über die `content.tpl`-Template-Datei eingebunden wird. Diese Variable sollte unbedingt eindeutig sein und wird daher standardmäßig auf {`$smarty_entries_X`} gesetzt, wobei `X` einer fortlaufenden Zahl entspricht, die pro neuem `serendipity_fetchPrintEntries`-Funktionsaufruf erhöht wird.

`preview` (optional)
: Wenn auf `true` gesetzt, wird der Artikel so behandelt, als würde er im Backend als Vorschau angezeigt. Dabei werden einige Plugins deaktiviert, die sich andernfalls an den Anfang und das Ende der Artikelübersicht einklinken würden.

`groupmode` (optional)
: Diese Variable legt fest, ob die Liste der Artikel nach dem Tag der Veröffentlichung gruppiert wird (Wert `date`, Standard) oder ob die Artikel nach Kategorie sortiert dargestellt werden sollen (Wert `category`).

`use_hooks` (optional)
: Wenn auf `true` gesetzt, werden Ereignis-Plugins für die Darstellung der Einträge wie gewohnt ausgeführt. Enthält der Parameter den Wert `false`, werden diese Plugins temporär übergangen. Dies gilt nur für den Aufruf von Textformatierungs-Ereignis-Plugins. Alle weiteren Ereignis-Plugins werden weiterhin aufgerufen und können durch den Parameter `skip_smarty_hooks` beeinflusst werden.

`use_footer` (optional)
: Wenn auf `true` gesetzt, wird der Footer unterhalb der Einträge mit Informationen zur Menge der gefundenen Artikel und mit Blättermöglichkeit eingebunden.

`skip_smarty_hooks` (optional)
: Um den Aufruf weiterer Ereignis-Plugins bei der Darstellung der Artikel zu unterbinden, kann dieser Parameter auf `true` gesetzt werden. Dieser Parameter beeinflusst die globale Variable `$serendipity['skip_smarty_hooks']` (siehe Seite 181) temporär und wird nach dem Aufruf der Smarty-Funktion `serendipity_fetchPrintEntries` wieder auf den ursprünglichen Wert zurückgesetzt.

`skip_smarty_hook` (optional)
: Im Gegensatz zur pauschalen Blockierung aller Ereignis-Plugins kann diese Variable ein Array enthalten, das eine Liste aller zu blockierenden Ereignisse (*Hooks*) festlegt. So können gezielt bestimmte Ereignis-Plugins unterdrückt werden.

`select_key` (optional)
: Kann eine kommaseparierte Liste von SQL-Tabellenfeldnamen enthalten, die in der Ausgabe gesetzt werden sollen. Standardmäßig selektiert Serendipity alle benötigten Felder selbständig.

`group_by` (optional)
: Enthält den Tabellenfeldnamen, nach dem die Artikelliste via SQL gruppiert werden soll. Standardmäßig wird nach Artikel-ID gruppiert, so dass ein Eintrag niemals mehr als einmal in der Liste auftreten kann.

`returncode` (optional)
: Legt fest, ob die Artikelausgabe als ein Datensatz (`single`) oder mehrfach verschachtelt (`array`) ausgegeben wird. Im Kontext der Smarty-Funktion ist grundsätzlich nur der Returncode `array` sinnvoll.

`joinauthors` (optional)
: Enthält den Wert `true`, wenn die Metadaten für den Redakteur eines Artikels mit ausgelesen werden sollen.

`joincategories` (optional)
: Enthält den Wert `true`, wenn die Metadaten für die zugewiesenen Kategorien eines Artikels mit ausgelesen werden sollen.

`joinown` (optional)
: Kann SQL-Anweisungen enthalten, die den `JOIN`-Teil der Abfrage nach eigenen Wünschen beeinflussen und weitere Tabellen in Betracht ziehen können.

`entryprops` (optional)
: Mit diesem Parameter können Sie eine Liste von zusätzlichen `freien Eigenschaften` (siehe Seite 262) angeben. Daraufhin werden nur die Artikel dargestellt, die die gewünschten Eigenschaften enthalten. So können Sie dafür sorgen, dass beispielsweise nur Artikel angezeigt werden, die keine dauerhaften Einträge darstellen:

```
{serendipity_fetchPrintEntries entryprops="ep_is_sticky
 != 'false'"}
```

Anderes Beispiel: Wenn Sie eine freie Eigenschaft wie *IstArchiviert* festgelegt haben und dort nur Einträge mit dem Wert `1` darstellen wollen:

```
{serendipity_fetchPrintEntries
  entryprops="ep_IstArchiviert = '1'"}
```

Mehrere Eigenschaften können mit Komma voneinander getrennt werden:

```
{serendipity_fetchPrintEntries
  entryprops="ep_IstArchiviert = '1',ep_is_sticky !=
  'false'"}
```

Die Smarty-Funktion setzt die notwendigen Bedingungen um und schreibt sie in den Parameter `joinown`, so dass Sie keine eigene SQL-JOIN-Bedingung entwickeln müssen.

`prevent_reset` (optional)

Standardmäßig enthält die Variable den Wert `false` und sorgt dafür, dass die zentrale Smarty-Variable {`$entries`} nicht überschrieben wird. Wenn Sie die Funktion `serendipity_fetchPrintEntries` jedoch dazu verwenden, die Standardausgabe von Serendipity zu unterdrücken, müssen Sie den Parameter `prevent_reset` auf `true` setzen, da Sie andernfalls eine doppelte Artikelausgabe erhalten würden.

Rückgabewert: HTML-Zeichenkette (`entries.tpl`)

9.6.4 Smarty-Modifier

Auf den folgenden Seiten sind alle Smarty-Modifier aufgeführt, die Serendipity innerhalb der `*.tpl`-Template-Dateien zur Verfügung stellt.

Ein Modifier wendet eine spezielle Funktion auf eine Variable an und liefert den Rückgabewert direkt zurück. Im Gegensatz zu einer Smarty-Funktion kann ein Modifier nicht ohne zugeordnete Variable aufgerufen werden. Modifier können überall dort eingesetzt werden, wo man Variablen platziert:

```
Aktuelles Datum: {$smarty.now|@formatTime:'%d.%m.%Y'}
```

Der Modifier wird mittels `|`-Symbol hinter einer Variable (hier als *Stammvariable* bezeichnet) aufgeführt. Modifier können beliebig oft hintereinander verkettet werden, um sie nacheinander anzuwenden. Einige Modifier können Parameter festlegen, die mittels Doppelpunkt vom Namen des Modifiers getrennt werden.

Wenn die Smarty-Sicherheit ausgeschaltet ist (`$serendipity['smarty']->security`, siehe Seite 493), können auch alle PHP-Funktionen als Modifier eingesetzt werden. Der erste Parameter einer derart aufgerufenen PHP-Funktion entspricht dabei immer dem Wert, der links vom Modifier angegeben wurde. Mit aktivierter Smarty-Sicherheit sind nur die PHP-Funktionen `sprintf, sizeof, count, rand, print_r` und `str_repeat` aufrufbar.

Das Sonderzeichen @ vor dem Aufruf eines Modifiers heißt, dass die davor aufgeführte Variable einen einzelnen Wert und kein Array enthält. So kann Smarty den Zugriff auf die Variable geringfügig beschleunigen. Serendipity-Templates machen von diesem Sonderzeichen regen Gebrauch.

{$emit_htmlarea_code}
: *Zugewiesen in:* include/functions_entries_admin.inc.php, serendipity_printEntryForm()
Gibt den nötigen JavaScript-Code aus, der vom Artikeleditor benötigt wird, um den WYSIWYG-Editor darzustellen. Dieser Modifier spricht die Serendipity-Funktion serendipity_emit_h an (include/functions_entries_admin.inc.php) und nimmt den Namen des jeweils zu konvertierenden Formularelements als Stammvariable an.

{$makeFilename}
: *Zugewiesen in:* include/functions_smarty.inc.php
Konvertiert eine Stammvariable (Zeichenkette) so, dass sie innerhalb einer URL eingesetzt werden kann. Etwaige Sonderzeichen werden dabei automatisch konvertiert. Wenn der erste Parameter den Wert true enthält, werden Punkte in der Zeichenkette entfernt.

{$xhtml_target}
: *Zugewiesen in:* include/functions_smarty.inc.php
Gibt XHTML-gültigen Code zum Öffnen eines Popup-Fensters aus. Die Rückgabe enthält ein onclick="..."-Attribut, das innerhalb eines HTML-Tags wie <a> eingebunden werden kann.

{$emptyPrefix}
: *Zugewiesen in:* include/functions_smarty.inc.php
Prüft, ob die Stammvariable einen Wert enthält. Wenn dies der Fall ist, liefert der Modifier als Ergebnis eine vorangestellte Zeichenkette und erst dann den Wert (bei dem sämtliche HTML-Sonderzeichen escaped werden).

  Wenn {$blogTitle} beispielsweise den Wert Dollhouse enthält, erreicht man mit dem Smarty-Befehl {$blogTitle|emptyPrefix:'Blogtitel - '} die Ausgabe Blogtitel - Dollhouse. Wäre der Wert der Variable leer, würde auch das Präfix Blogtitel - nicht angezeigt.

{$formatTime}
: *Zugewiesen in:* include/functions_smarty.inc.php
Formatiert eine Stammvariable mit Unix-Sekunden-Wert in eine lesbare Zeit. Der erste Parameter enthält dabei die Platzhalter für das Ausgabeformat (siehe http://www.php.net/date). Der Platzhalter kann dabei auch PHP-Konstanten enthalten, die ein Datum festlegen (bspw. $CONST.DATE_FORMAT_ENTRY). Der zweite Parameter enthält standardmäßig

den Wert `true`, damit Serendipity die Zeitzone des Servers berücksichtigt. Die weiteren Parameter werden für interne Sonderfälle benötigt, um die Behandlung ungültiger Zeitstempel und die Umgehung der Kalender-Umrechnung festzulegen.

{`$serendipity_utf8_encode`}
Zugewiesen in: `include/functions_smarty.inc.php`
Wandelt eine Zeichenkette in den UTF-8-Zeichensatz.

{`$ifRemember`}
Zugewiesen in: `include/functions_smarty.inc.php`
Prüft, ob der Benutzer einen benötigten Wert in einem Cookie gespeichert hat. Die Stammvariable legt dabei den Namen des Cookies fest, das geprüft werden soll. Der erste Parameter enthält den Inhalt, auf den ein Cookie geprüft wird. Der zweite Parameter legt fest, ob, wenn ein Cookie keine Daten enthält, der übermittelte Prüfwert zurückgeliefert werden soll. Der dritte Parameter gibt den Namen des HTML-Attributs aus, der zurückgeliefert wird.

Diese Funktion wird vor allem für die Darstellung der Mediendatenbank-Übersicht verwendet, um die Standardzustände von Auswahlfeldern und Ankreuzboxen festzulegen.

{`$checkPermission`}
Zugewiesen in: `include/functions_smarty.inc.php`
Prüft, ob ein Benutzer aufgrund seiner Mitgliedschaft in den Autor-Benutzergruppen ein bestimmtes Recht hat. Als Stammvariable dient der Name des zu prüfenden Rechts (z. B. `adminUsersMaintainOthers`). Als erster Parameter gilt die ID des Redakteurs, für den das Recht geprüft werden soll (standardmäßig auf die ID des aktuell eingeloggten Redakteurs gesetzt). Der zweite Parameter legt fest, ob der Modifier anstelle von `true/false` für die Prüfung eine Liste aller zugehörigen Gruppen des Redakteurs zurückliefern soll.

{`$serendipity_refhookPlugin`}
Zugewiesen in: `include/functions_smarty.inc.php`
Der Aufruf der Smarty-Funktion {`serendipity_hookPlugin`} (Seite 563) kann die von den Plugins durchgereichten Daten (`$eventData`) nicht als referenzierte Variable weitergeben. Nur Modifier können mit referenzierten Variablen arbeiten, daher dient dieser Modifier nur dem Zweck, ein Ereignis-Plugin aufzurufen, das mit Smarty-Variablen arbeiten kann. Die Template-Dateien des Backends machen von diesem Modifier Gebrauch, um zum Beispiel das Login-Formular mit einigen Zusatzausgaben anzureichern.

Die Stammvariable entspricht dabei dem, was als `$eventData` an die Plugins weitergereicht wird. Der erste Parameter des Modifiers gibt den Namen des auszuführenden Ereignisses (`Hook`) an, der zweite Parameter die etwaigen Zusatzdaten `$addData`.

9.7 index.tpl, content.tpl und entries.tpl im Detail

Anhand der Variablenliste sollte es Ihnen nun möglich sein, den Hauptaufbau einer Serendipity-Seite mittels der drei zentralen Template-Dateien `index.tpl`, `content.tpl` und `entries.tpl` nachzuvollziehen.

Um die Logik der einfachen Smarty-Templates im Schnelldurchlauf kennen zu lernen, folgt eine kleine Beschreibung mittels der Dateien des Standard-Templates (`templates/default/`).

Wir werden diese Zeile für Zeile durchgehen, daher können Sie in einem Text-Editor Ihrer Wahl die entsprechende Template-Datei öffnen, um die Schritte nachzuvollziehen.

9.7.1 index.tpl

Über die Datei `index.tpl` wird das Grundgerüst der HTML-Seite mitsamt Kopf- und Fußbereich ausgegeben.

Dazu prüft eine Smarty-Abfrage {`if $is_embedded != true`} in der ersten Zeile, ob die Serendipity-Konfigurationsoption zur eingebetteten Nutzung aktiviert wurde (siehe Seite 679). Ist dies der Fall, werden alle HTML-Kopfbereiche unterdrückt, und das Template schreitet erst ab Zeile 29 fort.

Bei standardmäßig deaktivierter eingebetteter Nutzung wird im Folgenden die Variable {`$is_xhtml`} geprüft und abhängig davon der entsprechende HTML `<!DOCTYPE>`-Tag ausgegeben (Zeile 2 bis 8).

Im nächsten Block begint der `<html>`- und `<head>`-Bereich der Seite, der einige Metadaten wie Seitentitel, Sprache und Links zu den Stylesheets enthält.

In Zeile 18 wird geprüft, ob ein spezieller Trackback-Link eingebettet wird. Die Variable `entry_id` ist nur dann gesetzt, wenn im Blog ein einzelner Artikel angezeigt wird. Andernfalls ist diese Variable leer, und der entsprechende `<link>`-HTML-Tag wird nicht ausgegeben.

Kurz vor dem Ende des `<head>`-Bereichs wird in Zeile 22 ein Aufruf der Plugin-Schnittstelle mittels {`serendipity_hookPlugin hook="frontend_header"`} durchgeführt. An dieser Stelle werden später etwaige Zusatzausgaben von Plugins eingebunden. Dieser Plugin-Aufruf findet sich auch in Zeile 27 wieder, damit Plugins auch dann ausgeführt werden, wenn der Modus zur eingebetteten Nutzung aktiviert wurde.

Eine weitere Abfrage, die den generellen Seitenaufbau betrifft, ist in Zeile 30 zu finden: {`if $is_raw_mode != true`} prüft, ob das Template im Folgenden seinen eigenen Kopf-Bereich mit Plugin-Seitenleisten ausgeben soll oder nicht. Diese spezielle Variable wird intern belegt und ist dafür gedacht, um mittels Plugin-Schnittstelle fremde Seiten in das Serendipity-Layout einzubinden. Es wird auch verwendet, wenn ein Template aus einer Serendipity-Version vor 0.7 aktiviert wurde. Dort gab es noch keine Smarty-Templates, und die Ausgabe erfolgte mittels einer eigenständigen `layout.php`-Datei. Daher wird im üblichen Fall die Variable {`$$is_raw_mode`} meist nicht aktiviert werden, so dass Ihr

`index.tpl`-Template in fast allen Fällen den Bereich zwischen Zeile 30 und 47 auswerten wird. Im aktivierten *Roh/Raw-Modus* wird der einzubindende Inhalt über die Variable {`$raw_data`} (Zeile 49) ausgeliefert.

In Zeile 31 bis 34 wird der Kopfbereich der Seite mitsamt Blog-Titel und -Beschreibung eingebunden, darauf folgt eine Layout-Tabelle in Zeile 36, die den Bereich in linke Seitenleiste, rechte Seitenleiste und mittleren Inhaltsbereich aufteilt.

In der linken und rechten Seitenleiste wird in Zeile 38 und 42 vorerst über die Variablen {`$leftSidebarElements`} und {`$rightSidebarElements`} geprüft, ob die jeweilige Seitenleiste überhaupt Elemente einbindet. Die Seitenleisten-Zuordnung wird über das Blog-Backend vorgenommen, daher müssen Templates immer selbständig überprüfen, welche Seitenleisten an welcher Stelle ausgegeben werden. Dies erfolgt letztlich über den Aufruf {`serendipity_printSidebar side="..."`}.

Der eigentliche Inhalt des Blogs wird über die Variable {`$CONTENT`} in Zeile 41 eingebunden. Dieser wird über die folgende Template-Datei `content.tpl` bestimmt.

Am Ende von `index.tpl` in Zeile 50 wird der Fußbereich durch Plugin-Ausgaben mit {`serendipity_hookPlugin hook="frontend_footer"`} ausgegeben, gefolgt von den schließenden `</body>`- und `</html>`-Tags (nur bei deaktivierter eingebetteter Nutzung).

9.7.2 content.tpl

Die Template-Datei `content.tpl` ist letztlich nur ein Verteiler, der entweder die Artikeldarstellung, Suchergebnisse oder eine Archivübersicht aufruft und mittels {`$ENTRIES`} und {`$ARCHIVES`} einbindet.

Zusätzlich kann diese Datei für zentrale Ausgaben wie z. B. für Suchergebnisse verwendet werden. Diese Ausgaben werden über Smarty-Variablen innerhalb der Datei von Zeile 3 bis 17 abgefragt.

9.7.3 entries.tpl

Die `entries.tpl`-Template-Datei ist die wohl komplexeste und umfangreichste Template-Datei. Im Gegensatz zu anderen Blog-Systemen verfolgt Serendipity den Ansatz der kleinstmöglichen Redundanz: Eine Template-Datei wird für alle Fälle der Artikeldarstellung verwendet, sowohl Einzelbeiträge als auch Eintragsübersichten werden (mittels IF-Abfragen) über diese Template-Datei gesteuert.

Das Template arbeitet dazu hauptsächlich mit einem großen Array namens {`$entries`} . Dieses mehrdimensionale Array wird mittels Schleifen durchlaufen und ausgegeben, daher macht es für das Template keinen Unterschied, ob in dem Array ein Eintrag oder 200 Einträge enthalten sind.

In Zeile 2 wird als mit {`serendipity_hookPlugin hook="entries_header" addData="$entry_` dafür gesorgt, dass etwaige Ereignis-Plugins ihre Ausgaben vor der Artikeldarstellung ausliefern können.

Von Zeile 4 bis Zeile 181 wird die erste Schleife über das mehrdimensionale {`$entries`} -Array ausgegeben. Die erste Dimension enthält die Einträge gruppiert nach deren Veröffentlichungsdatum. So können Einträge, die zum selben Tag gehören, auch gemeinsam ausgegeben werden, und pro Datumsgruppe wird nur eine Datumsüberschrifft dargestellt. Dies muss nicht zwangsläufig so sein, Templates können die Schleifen auch so anordnen, dass jeder Eintrag eine eigene Überschrift enthält. Für jeden Schleifen-Durchlauf wird die aktuelle Datumsgruppe der Variablen {`$dategroup`} zugewiesen.

Die Zeilen 5 bis 11 regeln diese Ausgabe der Datumsüberschrift in einem eigenen `<div>`-Container. Die Variable {`$dategroup.is_sticky`} wird abgefragt, damit für dauerhafte Einträge kein Datum, sondern nur eine Textüberschrift angezeigt wird. Im `else`-Fall wird das Datum der jeweiligen Artikelgruppe dargestellt. Dieser Wert liegt in Unix-Sekunden vor und muss daher über den Smarty-Modifier `formatTime` erst in ein menschenlesbares Format überführt werden.

Ab Zeile 12 beginnt nun die innere Schleife, die die einzelnen Artikel des {`$entries`} -Array ausliest. Die Artikel enstammen dabei der Variable {`$dategroup.entries`} . Pro Durchlauf der `foreach`-Schleife wird der aktuell durchlaufene Artikel der Variable {`$entry`} zugewiesen, so dass auf dessen Daten mit einfacher, kurzer Notation zugegriffen werden kann.

Wird die Detailansicht eines Artikels aufgerufen, enthält das Array nur einen Datensatz, und zusätzlich ist die Variable {`$is_single_entry`} gesetzt. Diese Variable wird an mehreren Stellen abgefragt, um die Ansicht der Detailseite und der Artikelübersicht unterschiedlich zu gestalten.

In Zeile 13 wird der Artikeltitel ausgegeben, gefolgt vom Autorennamen in Zeile 15. Falls der Artikel einer oder mehreren Kategorien zugeordnet ist, werden die Kategoriebilder in Zeile 18 bis 22 in einer Schleife ausgegeben.

Der Artikeltext wird in Zeile 34 bis 36 ausgegeben. Falls der Artikel einen erweiterten Eintragstext ({`$entry.is_extended`}) besitzt, wird ein HTML-Link zur Detailseite dieses Artikels ausgegeben.

Damit der erweiterte Eintrag angezeigt wird, wenn die Detailansicht aufgerufen wurde, wird die Variable {`$entry.extended`} in Zeile 26 bis 28 dargestellt.

Die Zeilen 38 bis 71 geben weitere Metadaten eines Artikels in einem Fußbereich aus: Zugeordnete Kategorien, Anzahl der Einträge, Anzahl der Kommentare und Anzahl der Trackbacks. Dazu werden jeweils einfache IF-Abfragen auf die entsprechenden Variablen eingebunden. Fallweise kann die Darstellung der Kommentare als Popup erfolgen, daher wird die zentrale Konfigurationsvariable `use_pops` in einige IF-Abfragen miteinbezogen. Falls der aktuelle Besucher ein eingeloggter Redakteur ist, wird ihm zusätzlich in Zeile 67 ein Link zum Bearbeiten des Artikels angeboten. Die Zusatzvariable {`$entry.add_footer`} enthält zusätzliche HTML-Ausgaben, die von Ereignis-Plugins stammen.

RDF-Metadaten zur Verwendung für Trackbacks und Suchmaschinen werden in den Zeilen 73 bis 83 dargestellt. Nach diesem Abschnitt dient die Zusatzvariable {`$entry.plugin_display_dat`} erneut dazu, weitere Ausgaben von Ereignis-Plugins einzugliedern.

Ab Zeile 86 bis 117 wird bei der Detailansicht eines Artikels die Liste von Trackbacks angezeigt. Innerhalb dieses Blocks werden von Zeile 87 bis 105 etwaige Nachrichten eingebunden, falls ein Besucher gerade ein Trackback an die Artikelseite gesendet hat. Die Liste der Trackbacks erfolgt in Zeile 115 mittels Smarty-Funktion {`serendipity_printTrackbacks entry=$entry.id`}.

Zeile 119 bis 172 regelt ähnlich wie im vorausgehenden Block die Darstellung der Kommentare. Hier wird in Zeile 124 bis 130 ein kleiner Auswahlbereich für den Benutzer angezeigt, damit die Kommentare entweder verschachtelt oder chronologisch sortiert dargestellt werden können. Abhängig von dieser Auswahl erfolgt eine Einbindung der Kommentarliste mittels Smarty-Funktion {`serendipity_printComments`}.

Zeile 134 bis 140 ermöglichen einem eingeloggten Redakteur, den Eintrag für weitere Kommentare zu sperren oder wieder zu öffnen.

Ab Zeile 143 bis 162 werden Hinweise und Nachrichten für einen Besucher angezeigt, der gerade einen Kommentar übermittelt hat. Entweder informieren diese Hinweise über den Erfolg ober die Abweisung des Kommentars, jeweils abhängig davon, ob Kommentare des Artikels moderiert werden, gesperrt sind oder Anti-Spam-Maßnahmen zutrafen.

Das Kommentarformular selbst wird über die Variable {`$COMMENTFORM`} in Zeile 165 bis 169 eingebunden.

Sollte das Array {`$entries`} einmal keine Artikel enthalten, wird eine Information zur leeren Artikelliste in Zeile 177 bis 181 ausgegeben. Dieser `else`-Fall der `foreach`-Schleife tritt beispielweise dann ein, wenn eine Volltextsuche keine Ergebnisse ausliefert oder wenn

man die Übersichtsseite eines Monats aufruft, in dem kein Artikel verfasst wurde.

Im Anschluss an die beiden Schleifen zur Darstellung von Artikeln erfolgt in den Zeilen 183 bis 197 die Einbindung eines Fußbereichs zum Vor- und Zurückblättern etwaiger Archivseiten. Auch hier können sich in Zeile 196 etwaige Ereignis-Plugins einbinden.

9.8 Freie Eigenschaften von Artikeln

Grundsätzlich bietet Serendipity beim Erstellen eines Artikels nur die Möglichkeit, die Textbereiche in *Eintrag* und *Erweiterten Eintrag* aufzuteilen. Zusätzlich gibt es Plugins wie *Textformatierung: Eintragsdaten einfügen* (siehe Seite 373), mit denen man andere Artikel referenzieren oder beliebige Blöcke (beispielsweise mit Werbung) anhängen kann.

Eine generische und flexible Lösung für eigene Erweiterungen bietet das Ereignis-Plugin *Erweiterte Eigenschaften von Artikeln* (siehe Seite 262).

Mithilfe dieses Plugins können Sie beliebig viele zusätzliche Eingabefelder zu einem Artikel hinzufügen und später darstellen. Die Anwendungsmöglichkeiten sind vielfältig: Fügen Sie einen Block mit Werbung hinzu, hängen Sie bestimmten Artikeln Videos und Bilder an oder erfassen Sie die aktuelle Phase Ihres persönlichen Mondkalenders.

Zwar könnten Sie solche Zusatzinformationen auch direkt in den normalen Eintrag einfügen, würden aber dadurch möglicherweise (versehentlich) eine von Artikel zu Artikel unterschiedliche Formatierung oder Darstellungsweise erhalten; oder Sie müssten ständig eine große Menge an Vorlage-Code in den Eintrag übernehmen, damit die benötigten HTML-Formatierungsanweisungen immer identisch sind.

Die *Freien Eigenschaften* (engl. *entryproperties*) bieten den Vorteil, dass Sie nur den rohen Text erfassen müssen und die Darstellung später anhand Ihres Templates immer an exakt derselben Stelle erfolgt.

Als Beispiel setzen wir uns folgendes Ziel: Zu jedem unserer Artikel wollen wir neben dem Text auch eine Youtube-Videodatei verlinken können und unsere aktuelle Stimmung dokumentieren.

9.8.1 Plugin installieren

Die folgenden Schritte setzen voraus, dass Sie das Ereignis-Plugin *Erweiterte Eigenschaften von Artikeln* (siehe Seite 262) installiert haben. Wenn dies nicht der Fall ist, richten Sie es im Backend über **Aussehen** → **Plugins verwalten** → **Hier klicken, um Ereignis-Plugin zu installieren** ein.

9.8.2 Freie Eigenschaften anlegen

Als Erstes müssen Sie die beiden zusätzlichen Eigenschaftsfelder anlegen. Gehen Sie dazu auf den Menüpunkt **Aussehen** → **Plugins verwalten** und suchen Sie in der Auflistung der Ereignis-Plugins das Plugin *Erweiterte Eigenschaften von Artikeln*; klicken Sie auf das Schraubstock-Symbol, um zur Konfiguration zu gelagen.

Dort finden Sie die Option **Freie Felder**. Hier kann man eine Liste von Feldnamen eintragen, die zusätzlich angezeigt werden sollen. Dazu benötigt das Plugin einen eindeutigen Namen pro Feld, diesen können Sie frei vergeben. Er darf nur aus einem einzelnen Wort bestehen und keine Sonderzeichen oder Umlaute enthalten. Für unser Beispiel tragen wir dort für zwei Felder **Youtube,Stimmung** ein.

9.8.3 Artikel erstellen

Um die Einbindung der freien Felder zu testen, müssen Sie einen neuen Artikel erstellen. Klicken Sie im Backend dazu auf **Einträge** → **Neuer Eintrag**. Dort finden Sie nach dem Eingabefeld für den *Erweiterten Eintrag* einen Bereich mit dem Titel *Erweiterte Optionen*. In diesem Bereich finden Sie nun einen Abschnitt *Erweiterte Eigenschaften von Artikeln*. Dort bindet das Plugin neben einigen weiteren Optionen nun zwei große Eingabefelder ein: eines mit dem Titel **Youtube** und eines mit dem Titel **Stimmung**.

Dort können Sie nun für jeden Artikel einen beliebigen Code eintragen. Was Sie in die Felder eintragen, ist Ihnen überlassen, jedoch sollten Sie darauf achten, dort immer nur die *Rohdaten* einzutragen. Alles, was sich bei jeder Eingabe wiederholen würde (z. B. umgebende HTML-Formatierung wie eine Überschrift oder eine Trennlinie), gehört nicht an diese Stelle, sondern ins Template.

Da die Darstellung des Youtube-Videos später einiger HTML-Anweisungen bedarf, trägt man in das Eingabefeld möglichst nur die Youtube-Video-ID (wie z. B. `2BpLGASxt0Q`) ein. Diese Video-ID kann man immer der URL einer Youtube-Seite (`http://www.youtube.com/watch?v=2BpLGASxt0Q`) entnehmen.

Zusätzlich werden Sie sicher bemerken, dass neben dem Eingabefeld ein Link **Mediendatenbank** eingebunden wird. Wenn Sie auf diesen Link klicken, öffnet sich das bekannte Popup zur Auswahl einer Datei in Ihrer Mediendatenbank (siehe Seite 107). Dort können Sie eine Datei auswählen, die Sie einbinden möchten. Daraufhin erscheint der Dateiname der ausgewählten Datei innerhalb des Eingabefeldes und kann somit später im Template z. B. als Downloadlink oder Bild ausgegeben werden, ganz wie Sie es benötigen.

Speichern Sie einen Artikel mit ausgefülltem **Youtube**- und **Stimmung**-Feld ab.

9.8.4 Template anpassen

Nachdem die Daten nun erfasst wurden und mit dem Artikel zusammen abgespeichert sind, sollen diese natürlich auch in Ihrem Blog dargestellt werden.

Dafür müssen Sie nun die `entries.tpl`-Template-Datei Ihres Blogs in einem Editor öffnen. Diese Datei enthält alle Anweisungen, die zur Darstellung eines Eintrags verwendet werden.

Innerhalb dieser Datei (genaue Beschreibung siehe Seite 519) können Sie auf die freien Felder flexibel mittels Smarty-Syntax zugreifen. Um Ihre aktuelle Stimmung direkt unterhalb des Artikeltexts darzustellen, suchen Sie in der Datei nach einer Stelle, die ungefähr so aussieht:

```
{if $entry.is_extended}
    <div class="serendipity_entry_extended">
     <a id="extended"></a>
     {$entry.extended}
    </div>
{/if}
```

Diese Stelle sorgt üblicherweise einfach nur für die Ausgabe des erweiterten Eintrags. Wenn Sie Ihre aktuelle Stimmung nur dann anzeigen wollen, wenn ein Besucher die Detailansicht des Artikels aufruft, müssen Sie nun ein Stück Smarty-Code *innerhalb* der IF-Abfrage einfügen. Wenn Sie die Stimmung auch bereits in der Artikelübersicht darstellen wollen, fügen Sie den folgenden Smarty-Code *nach* dem abschließenden {`/if`} ein:

```
<h5>Meine Stimmung:</h5>
<div class="mood">{$entry.properties.ep_Stimmung}</div>
```

Die Ausgabe liefert also eine einfache Überschrift und gibt den Text, den Sie im Artikel hinterlegt haben, innerhalb eines `<div>`-Feldes mit eigener CSS-Klasse aus.

Der Feldname `ep_Stimmung` richtet sich dabei nach dem Namen, den Sie in der Konfiguration des Plugins anfangs eingetragen haben. Er muss exakt mit der gleichen Groß- und Kleinschreibung wie dort und mit einem vorangestellten `ep_` aufgeführt werden.

Da die Inhalte dieser Daten immer im {`$entry.properties`} -Array landen, müssen sie innerhalb dieses Arrays mit {`$entry.properties.ep_`*`Feldname`*} innerhalb der Template-Datei ausgegeben werden.

Als Zweites fügen Sie nun unterhalb der Darstellung Ihrer Stimmung den Youtube-Codeschnipsel ein:

```
<h5>Youtube-Video</h5>
<div class="youtube">
<object width="425" height="355">
 <param
   name='movie'
   value='http://www.youtube.com/v/{$entry.properties.ep_Youtube}'>
 </param>
 <param
```

```
   name="wmode"
   value="transparent"></param>
 <embed
   src='http://www.youtube.com/v/{$entry.properties.ep_Youtube}'
   type='application/x-shockwave-flash'
   wmode='transparent'
   width='425'
   height='355'></embed>
</object>
</div>
```

Dieser Code ist direkt von der Youtube-Seite entnommen, lediglich die ID des Videos wird ersetzt durch die Variable {`$entry.properties.ep_Youtube`} .

Speichern Sie das geänderte Template nun ab und rufen Sie Ihr Blog erneut auf. Sie werden nun bei dem vorhin erfassten Artikel die Zusatzausgaben sehen.

9.8.5 Weitere Prüfungen

Da Ihre Eingaben nur an das Smarty-Array {`$entry.properties`} durchgereicht werden, sind Sie in der Lage, damit weitaus mehr anzustellen, als nur rohen Text auszugeben.

Sie können beispielsweise auch ein freies Feld wie `ArtikelAnzahl` anlegen und dann mittels Smarty-Syntax eine Liste von aktuellen Artikeln des gleichen Redakteurs holen und darstellen. Dadurch wird ein freies Feld soszusagen als flexibler Parameter für beliebige Smarty-Funktionsaufrufe verwendet:

```
{$entry.body}

<div>Weitere Artikel dieses Autoren:</div>
<div>
{serendipity_fetchPrintEntries viewAuthor=$entry.authorid limit=$entry.p
roperties.ep_ArtikelAnzahl}
</div>
```

Im vorigen Beispiel wurde die aktuelle Stimmung und das Youtube-Video immer angezeigt, selbst wenn in Ihrem Artikel derartige Angaben gar nicht gemacht wurden. Daher ist es sinnvoll, eine kleine zusätzliche Abfrage einzufügen. Diese prüft, ob eine entsprechende Variable für den aktuellen Artikel überhaupt gesetzt ist:

```
{if $entry.properties.ep_Stimmung}
  <h5>Meine Stimmung:</h5>
  <div class="mood">{$entry.properties.ep_Stimmung}</div>
{/if}
```

9.9 Eigene Modifier, Funktionen oder Dateien einbinden

Smarty ermöglicht es Ihnen nicht nur, auf vorgeschriebene Smarty-Funktionen und -Modifier zuzugreifen (siehe auch `http://smarty.php.net/`), sondern auch eigene beliebig zu verwenden.

Standardmäßig ist für Serendipity-Templates die *Template-Security* aktiviert. Dies heißt, dass Sie keine beliebigen PHP-Funktionen und keine PHP-Include-Anweisungen in Ihre Templates einbauen dürfen. Dies dient als kleine zusätzliche Barriere, damit beim Einsatz von Smarty-Markup Ihre Redakteure kein Unheil auf dem Webserver anrichten können.

Für PHP-Entwickler ist aber genau dies gewünscht. Daher können Sie in der `config.inc.php`-Konfigurationsdatei Ihres Templates sehr einfach mittels der Zeile

```
<?php
$serendipity['smarty']->security = false;
?>
```

diese Sicherheitabfrage umgehen.

Weiterhin können Sie ganz eigene Smarty-Funktionen oder Smarty-Modifier einfügen, wenn Sie diese benötigen.

Auch dazu müssen Sie die `config.inc.php`-Datei Ihres Templates anpassen (oder erstellen, wenn sie noch nicht existiert). Dort können Sie eine Funktion wie folgt deklarieren:

```
<?php
function zeigeFooter($params, &$smarty) {
  if ($params['lang']) {
    return "Langer Footer";
  } else {
    return "Kurzer Footer";
  }
}

$serendipity['smarty']->register_function('zeigeFooter', 'zeigeFooter');
```

Eine Smarty-Funktion muss demnach mittels `register_function` einer existierenden PHP-Funktion zugewiesen werden.[14] Diese PHP-Funktion muss immer zwei Funktionsparameter haben: `$params` und `$smarty`. Das Array `$params` enthält alle Funktionsparameter, wie sie in der Template-Datei via {`zeigeFooter parameter=...lang=t ...`} aufgerufen wurden. Dabei entspricht der Array-Schlüssel von `$params` immer dem Parameternamen beim Smarty-Aufruf. Das `$smarty`-Objekt kann innerhalb der PHP-Funktion selbstverständlich auch für erweiterte Zugriffe auf das Smarty-Framework verwendet werden. Der Rückga-

[14] Siehe auch `http://smarty.php.net/register_function`

bewert der Funktion enthält das, was später im Smarty-Template zurückgeliefert wird (üblicherweise HTML-Code).

Die beispielhaft definierte Funktion `zeigeFooter` kann so verwendet werden, um einen fest definierten Footer in Ihrer Template-Datei einzubinden. Selbstverständlich könnte diese Funktion auch Datenbankabfragen ausführen oder andere HTML-Dateien einbinden.

Eine ähnliche Syntax gilt für die Festlegung eigener Smarty-Modifier:

```
<?php
function meinDatum($timestamp, $kurz = false) {
  if ($kurz) {
    return date('d.m.Y', $timestamp);
  } else {
    return date('d.m.Y um H:i Uhr', $timestamp);
  }
}

$serendipity['smarty']->register_modifier('meinDatum', 'meinDatum');
```

Ein Modifier wie oben würde im Smarty-Template einen Aufruf wie

```
Artikeldatum: {$entry.timestamp|meinDatum:true}
```

ermöglichen. Der Unix-Zeitwert wird an die eigene Funktion `meinDatum` weitergereicht. Der erste Parameter der PHP-Funktion entspricht dabei immer dem Wert, der vor dem Modifier steht. Alle weiteren PHP-Funktionsparameter entsprechen der Liste von Parametern, die, mit Doppelpunkten getrennt, nach dem Smarty-Modifier-Aufruf folgen.

Die PHP-Funktion `meinDatum` wertet {`$entry.timestamp`} aus und liefert abhängig vom Parameter `kurz` entweder ein ausführlich oder kurz formatiertes deutsches Datum zurück.

Beliebige PHP-Dateien können mittels Smarty-Funktion `include_php` eingebunden werden. Einen derartigen Aufruf können Sie z. B. in Ihrer Datei `index.tpl` verwenden, um ein eigenes Counter-Script aufzurufen:

```
 <html>
 ...
 {include_php file="/var/www/example/counter.php"}
```

Alternativ können Sie einen Counter auch direkt über die `config.inc.php`-Datei aufrufen. Dies ist in jedem Fall dann zu bevorzugen, wenn Sie das Counter-Script auch dann ausführen wollen, wenn Serendipitity eine andere Seite als die Übersicht darstellt. Dazu können Sie üblichen PHP-Code verwenden:

```
<?php
include '/var/www/example/counter.php';
?>
```

Achten Sie jedoch darauf, dass innerhalb der `config.inc.php` niemals HTML-Ausgaben erfolgen dürfen. Sollte dies vorkommen, müssen Sie mittels PHP Output Buffering[15] die HTML-Ausgaben in einer Variable abfangen, an die Smarty-Template-Dateien weiterreichen und von dort aus einbinden.

`config.inc.php`:

```
<?php
ob_start();
include '/var/www/example/counter.php';
$tmp = ob_get_contents();
ob_end_clean();
$serendipity['smarty']->assign('counter_code', $tmp);
?>
```

`index.tpl`:

```
<html>
 ...
 {$counter_code}
```

9.10 Konfigurationsoptionen, Eigene Einstellungen

Die Konfigurationsoptionen von Serendipity (**Administration** → **Konfiguration** und **Eigene Einstellungen**) werden über einfache PHP-Dateien gesteuert. Darüber können Sie leicht eigene neue Konfigurationsoptionen hinzufügen, die automatisch dargestellt und in der Datenbank gespeichert werden können.

Dies kann dazu verwendet werden, um Serendipity auf eigene Optionen anzupassen, falls man diese Optionswerte in eigenen PHP-Dateien oder Plugins abfragen möchte. Dabei müssen Sie jedoch beachten, dass die für die Konfiguration zuständigen PHP-Dateien bei jedem Serendipity-Update überschrieben werden. Sie sollten etwaige Anpassungen an dieser Datei also strikt protokollieren und nach einem Update erneut einfügen.

Die Datei `include/tpl/config_personal.inc.php` stellt die Datenbasis für die **Eigenen Einstellungen** dar. In dieser Datei wird eine Variable deklariert, die ein mehrdimensionales Array enthält. Die Datei wird vom Serendipity-Kern ausgelesen, und die Darstellungsoptionen werden anhand des Inhalts dieser Variable eingebunden.

Eine identische Struktur wird von der Datei `include/tpl/config_local.inc.php` eingesetzt und dient der Darstellung für die **Konfiguration**.

Der Schlüsselname des Arrays der ersten Dimension gibt den Abschnittsnamen an. Bei den **Eigenen Einstellungen** gibt es zwei solche Abschnitte: **Persönliche Einstellungen** und **Voreinstellungen für neue Einträge**. Jeder Abschnitt kann für sich ein- und ausgeklappt werden.

[15] Siehe `http://www.php.net/ob_start`

Weitere individuelle Abschnitte können Sie daher über einen beliebigen Array-Schlüsselnamen einbinden.

Der Titel und die Beschreibung jedes Abschnitts wird in den Array-Schlüsseln `title` und `description` der zweiten Dimension festgelegt.

Der Array-Schlüssel `items` der zweiten Dimension legt ein weiteres Array an, in dem alle Konfigurationsfelder enthalten sind.

Pro Konfigurationsfeld wird ein Array mit folgenden Array-Schlüsseln festgelegt:

`var`
: Legt den Variablennamen fest. Mit diesem Namen ist die Konfigurationsoption später über `$serendipity['Variablenname']` zugänglich und wird mit demselben Namen in der Datenbanktabelle `serendipity_config` abgespeichert. Der Variablenname darf keine Sonderzeichen oder Umlaute enthalten.

`title`
: Legt den dargestellten Titel der Konfigurationsoption fest (kann auf eine eigene Sprachkonstante verweisen).

`description`
: Legt die Beschreibung der Konfigurationsoption fest (kann auf eine eigene Sprachkonstante verweisen).

`type`
: Legt den Typ der Konfigurationsoption fest. Gültige Werte sind:

  `bool`
  : Feld mit *Ja/Nein*-Auswahlbox.

  `protected`
  : Geschütztes Eingabefeld, dessen Eingaben mit einem `*` maskiert werden (z. B. für Passwörter).

  `multilist`
  : Ein Mehrfach-Auswahlfeld.

  `list`
  : Ein Auswahlfeld (Dropdown).

  `file`
  : Ein Dateiauswahlfeld (Upload).

  `textarea`
  : Ein mehrzeiliges Texteingabefeld.

  `string (Standard)`
  : Ein einzeiliges Texteingabefeld.

`default`
: Legt den Standardwert der Option fest, wenn der Benutzer noch keine eigene Auswahl getroffen hat. Bei Arrays, die ein Auswahlfeld anbieten, kann dieser Wert ein Array mit sämtlichen möglichen Optionswerten enthalten. Dabei entspricht der Array-Schlüssel jeweils dem Optionswert und der Array-Inhalt der Beschreibung des Wertes.

`permission`
: Gibt den Namen eines Serendipity-Rechtes an, das ein Redakteur über die Blog-Benutzergruppenzugehörigkeit besitzen muss, um die entsprechende Konfigurationsoption ausfüllen zu dürfen. Üblicherweise prüft dieses Feld z. B., ob ein Benutzer der Admin-Gruppe zugehörig ist, um erweiterte Optionen ausfüllen zu dürfen.

  Dieses Feld kann entweder einen einzelnen Rechtenamen enthalten oder ein Array mit mehreren Rechtenamen. Bei der Angabe mehrerer Rechtenamen muss der Benutzer alle Rechte besitzen, um eine Option aufrufen zu können. Dies kann über den Array-Schlüssel `perm_mode` gesteuert werden – enthält dieser den Wert `or`, werden die Rechte mit einem logischen *oder* abgefragt, so dass der Benutzer nur eines der angegebenen Rechte besitzen muss.

`view`
: Wenn dieser Array-Schlüssel als Wert `dangerous` enthält, wird die Konfigurationsoption nicht im Menübereich **Eigene Einstellungen** angezeigt, sondern nur, wenn das Redakteursprofil über den Bereich **Administration** → **Benutzerverwaltung** verändert wird. Dies dient dazu, damit sich Benutzer nicht selbständig und unabsichtlich notwendige Rechte entziehen.

`flags`
: Kann ein Array mit einigen Zusatzoptionen enthalten, die bestimmen, wie eine Konfigurationsoption gehandhabt wird. Folgende Flags werden unterstützt:

  `installOnly`
  : Eine Konfigurationsoption wird nur bei der Installation angezeigt.

  `hideValue`
  : Der Wert der Konfigurationsoption wird nicht in der HTML-Ausgabe mitgeliefert, so dass z. B. Passwörter nicht im Klartext (oder MD5-Hash) erscheinen.

  `config`
  : Ist dieses Flag gesetzt, wird das Rechtefeld in der Serendipity-Konfigurationstabelle gespeichert.

  `local`
  : Ist dieses Flag gesetzt, wird das Rechtefeld in der Tabelle `serendipity_authors` gespeichert. Dabei wird vorausgesetzt, dass der entsprechende Feldname als Spaltenname dieser Tabelle vorhanden ist.

  `parseDescription`
  : Falls gesetzt, werden spezielle Zeichenketten in der Beschreibung des Konfigura-

tionsfeldes durch einen Wert ersetzt. Derzeit wird nur die Zeichenkette `%clock%` innerhalb einer Beschreibung durch die aktuelle Uhrzeit ersetzt.

`probeDefault`
: Anstatt einen festen Standard vorzugeben, prüft Serendipity einige festgelegte Felder (`dbType, rewrite`) auf gültige Standardwerte.

`ifEmpty`
: Falls gesetzt, wird der Standardwert für einen Konfigurationswert von Serendipity speziell geprüft, falls der Benutzer noch keine eigene Option festgelegt hat.

`nosave`
: Ist dieses Feld gesetzt, wird der Wert einer Konfigurationsoption nicht in der Datenbank gespeichert. Dies ist üblicherweise der Fall bei Datenbank-Zugangsvariablen, die ausschließlich in der Datei `serendipity_config_local.inc.php` gespeichert werden.

`simpleInstall`
: Wenn dieses Flag gesetzt ist, wird eine Konfigurationsoption bei der Installation nur dann angezeigt, wenn der Benutzer die **Einfache Installation** gewählt hat.

Kapitel 10

Der Serendipity-Kern

Serendipity ist im Inneren ein Framework, das entsprechend der angeforderten URL die gewünschte Aktion ausführt. Daher initialisiert sich bei jedem Aufruf einer URL von Serendipity das gesamte Kernkonstrukt auf die immer gleiche Art und Weise.

10.1 Frontend

Zentrale Anlaufstelle fast aller Aktionen im Frontend ist die `index.php`-Datei. Egal ob ein einzelner Blog-Artikel, eine Übersicht oder der Inhalt eines Plugins ausgegeben wird – alle Ausgaben werden durch diese Datei *geroutet*. Auch bei der Verwendung von *URL-Umformung* mittels `mod_rewrite` oder Ähnlichem werden alle Aufrufe über die `.htaccess`-Datei im Hintergrund zur `index.php` weitergeleitet.

Aufgabe der `index.php`-Datei ist es nun, die Serendipity-Komponenten *zusammenzubauen*. Zuerst sendet diese Datei einige benötigte HTTP-Kopfzeilen und bindet die Datei `serendipity_config.inc.php` ein. Diese Datei ist das Herz des Frameworks, da sie sich darum kümmert, alle benötigten Komponenten und Sourcecode-Dateien einzubinden und erforderliche Konfigurationswerte einzulesen.

Dort werden einige PHP-Konstanten gesetzt, eine PHP-Session gestartet und die wichtigen Serendipity-Stammpfade festgelegt. Daraufhin erfolgt die Einbindung der Datei `include/compat.inc.php`. Hier werden die URL-Variablen korrekt zugewiesen und abhängig von der Konfiguration des PHP-Servers bearbeitet. Zusätzlich wird hier die zentrale `$serendipity`-Variable erstmals instanziert. Auch werden einige zentrale PHP-Funktionen definiert, die Serendipity im Folgenden immer wieder benötigen wird.

Nachdem diese Datei abgearbeitet wurde, folgt in der Datei `serendipity_config.inc.php` die weitere Definition des `$serendipity`-Arrays. Hier wird beispielsweise die Versionsnummer der Software festgehalten, die Art der Fehlerbehandlung bestimmt, und Standard-

werte für einige der Serendipity-Optionen werden vorbelegt.

Nachdem diese Variablen feststehen, ist es an der Zeit, den Sprachkern nachzuladen. Dieser richtet sich nach der vom Besucher festgelegten Sprache. Falls der Besucher ein angemeldeter Redakteur ist, wird dessen Sprachvariable aus den persönlichen Einstellungen bezogen. Auch die konfigurierte Browser-Sprache kann einen Einfluss darauf haben, in welcher Sprache Serendipity mit dem Benutzer spricht. Die einmal festgelegte Sprache wird daraufhin in der PHP-Session für den nächsten Aufruf gespeichert. Alle sprachbezogenen Aktionen werden mittels der `include/lang.inc.php`-Datei durchgeführt.

Nachdem nun die gewünschte Sprachdatei eingebunden wurde, kann Serendipity mit dem Besucher interagieren. An dieser Stelle wird geprüft, ob Serendipity überhaupt bereits installiert wurde (falls nicht, startet der Installationsprozess).

Als Nächstes setzt Serendipity die PHP-Include-Pfade so, dass alle zu Serendipity gehörenden Dateien eingebunden werden können. Dies ist besonders dann wichtig, wenn Serendipity in einer *Shared Installation* (siehe Seite 673) betrieben wird.

Nun kann Serendipity die zentrale Konfigurationsdatei `serendipity_config_local.inc.php` einlesen. Hierin sind die Zugangsparameter zur Datenbank enthalten. Wenn diese Zugangsparameter korrekt gelesen wurden, kann eine Verbindung zur Datenbank hergestellt werden.

Die Einbindung der `include/functions.inc.php` sorgt dafür, dass sämtliche PHP-Kernfunktionen Serendipitys geladen werden. Dazu werden weitere Dateien wie `include/functions_confi` eingebunden, die diese Funktionen jeweils themenspezifisch bündeln (siehe auch Seite 595).

Nun wird das `$serendipity`-Array mit allen Konfigurationswerten der Datenbank befüllt. Der Redakteur kann so mit der Benutzerdatenbank abgeglichen werden, um einen gültigen Login zu erkennen und ggf. die persönliche Konfiguration des Besuchers zu laden.

Als Letztes erfolgt in der `serendipity_config.inc.php` der Aufruf der Plugin-API (siehe Seite 642). Beim ersten Aufruf werden alle installierten Plugins in den Speicher geladen und etwaige Aktionen ausgeführt.

Nachdem nun das Serendipity-Framework geladen wurde, kann der Ablauf der `index.php` fortgesetzt werden. Hier werden weitere HTTP-Kopfzeilen gesendet.

Da nun der vollständige Zugriff auf die Datenbank möglich ist, kann Serendipity abhängig von der Konfiguration der Permalinks prüfen, welche Seite der Besucher angefordert hat. Dazu dienen zahlreiche `IF-Abfragen`, die den Großteil dieser Datei ausmachen.

Jeder mögliche Ausgabefall (Archiv-Übersicht, Kategorie-Übersicht, Einzelartikel, Volltextsuche ...) wird hier mit der aktuellen URL (`$uri`) verglichen. Falls eine der IF-Bedingungen zutrifft, ruft Serendipity die für die Unterseite notwendigen Befehle ab.

Meist bestehen diese Befehle darin, benötigte Variablen vorzubelegen und Komponenten der URL (Autor-ID, Kategorie-ID, Zeitraum der Archive ...) auszulesen. Abschließend wird in beinahe jeder der IF-Bedingungen die Datei `include/genpage.inc.php` aufgerufen, die für die Ausgabe einer Template-Datei verantwortlich zeichnet.

In dieser Datei wird hauptsächlich die *Smarty*-Bibliothek geladen, die für die Auswertung der

Template-Dateien zuständig ist. Serendipity reicht einige Parameter an die dafür zuständige `serendipity_smarty_init()`-Funktion weiter. Diese Funktion wird in der Datei `include/functions_smarty.inc.php` deklariert. Sie setzt alle gewünschten Variablen und wertet anschließend etwaige `config.inc.php`-Dateien Ihres Templates aus. Darüber können zusätzliche Template-Optionen eingestellt werden. Sämtliche zusätzlichen Smarty-Modifier und eigene Smarty-Funktionen werden an dieser Stelle des Ablaufs gesetzt, und Standard-Serendipity-Variablen werden an die Templates weitergereicht.

Daraufhin kann Serendipity anhand der URL-Parameter den eigentlichen Inhalt der Seite zusammenstellen. Die Funktion `serendipity_fetchEntries()` sorgt dafür, dass die angeforderten Artikel aus der Datenbank eingelesen werden. Daraufhin werden sie über die Funktion `serendipity_printEntries` bearbeitet und letztlich an die erforderliche Template-Datei zugewiesen.

Nachdem die Datei `genpage.inc.php` den eigentlichen Seiteninhalt aufbereitet hat (Smarty-Variable {`$CONTENT`}), kann die `index.php`-Datei als letzte Maßnahme für die Ausgabe des Inhalts sorgen, indem sie den Darstellungsprozess von *Smarty* aufruft.

10.2 Backend

Der Ablauf im Backend ist beinahe identisch. Anstelle der `index.php` wird hier jedoch immer die `serendipity_admin.php` aufgerufen. Diese prüft den Aufruf zusätzlich auf einen gültigen Login und stellt das Menü zusammen. Anschließend wird abhängig von der gewählten Aktion im Backend die entsprechende Include-Datei aus dem Verzeichnis `include/admin/` eingebunden. Diese Datei führt die gewünschte Aktion aus (z. B. Verwalten von Einträgen) und liefert den HTML-Teil zurück, der im rechten Bereich des Backends angezeigt werden soll.

Dieser HTML-Teil wird über die Template-Datei `admin/index.tpl` entsprechend eingebunden und anschließend zusammen mit dem Menü und dem Kopfbereich ausgegeben.

Eine Übersicht über die Inhalte aller zu Serendipity gehörenden Dateien finden Sie ab Seite 595.

10.3 Sonderfälle

Obwohl die meisten Serendipity-Ansichten entweder über das Backend (also `serendipity_admin.php`) oder Frontend (`index.php`) geroutet werden, gibt es einige Sonderfälle. Diesen Sonderfällen ist jedoch gemein, dass sie alle das Serendipity-Framework einbinden, um auf identische Funktionen und Plugins zugreifen zu können.

10.3.1 Kommentare, Trackbacks

Eingehende Trackbacks (und Pingbacks) werden nicht über das zentrale Framework abgearbeitet, sondern über eine eigenständige Datei `comment.php`. Diese bindet das Serendipity-Framework zwar ein, dient aber ansonsten aus Gründen der Übersichtlichkeit ausschließlich der Abarbeitung von Trackbacks. Da diese einen ressourcenintensiven Prozess darstellen, werden sie nicht über die zentrale Datei bearbeitet. Dies erleichtert Logging und evtl. auch den Ausschluss des Trackback-Mechanismus' bei Ressourcenproblemen (siehe Seite 441).

Falls Ihr Serendipity-Blog so konfiguriert ist, dass Kommentare in einem Popup angezeigt werden, dient die `comment.php` auch zur Darstellung und Entgegennahme von Kommentaren. Ohne Popup-Darstellung übernimmt dies wie gewohnt die `index.php`-Datei, daher befindet sich hier stellenweise duplizierter Code.

Einen Sonderfall von Kommentaren stellen die sogenannten `wfwComments` dar. Sie sind Teil eines Standards, der speziell für RSS-Reader ins Leben gerufen wurde. Er erlaubt, dass man direkt vom RSS-Reader aus Kommentare zu einem Blog-Eintrag verfassen kann, ohne dazu erst die URL des Blogs und dort ein Kommentarfeld aufrufen zu müssen. Serendipity nimmt derartige Kommentare über die Datei `wfwcomment.php` entgegen.

10.3.2 Exit-Nachverfolgung

Serendipitys Plugin zum `Externe Links zählen` (siehe Seite 258) dient zur statistischen Erfassung, auf welche Links Ihre Besucher klicken. Dabei leitet Serendipity den Besucher auf die eigentlich gewünschte URL weiter. Damit dies Ressourcen sparend geschehen kann, ist diese Funktionalität in der Datei `exit.php` gekapselt.

10.3.3 RSS-Feeds

Da die Darstellung von RSS-Feeds unabhängig vom Frontend geschieht, ist diese Aufgabe über die separate Datei `rss.php` geregelt.

10.3.4 Mediendatenbank-Popup

Die Mediendatenbank kann an einigen Stellen des Front- und Backends aufgerufen werden und erfolgt meist als Popup. Um dies komfortabler und isolierter zu verwalten, befindet sich der dafür notwendige Code in der Datei `serendipity_admin_image_selector.php`.

10.3.5 XML-RPC

Serendipity unterstützt über ein Plugin die Möglichkeit, Blog-Artikel mittels fremder Anwendungen zu erstellen, ohne dafür das Serendipity-Backend benutzen zu müssen.

Hierfür wurde die Datei `serendipity_xmlrpc.php` geschaffen. In früheren Serendipity-Versionen war hier der vollständige Code enthalten, um die gewünschte API zum Erstellen von Artikeln abzudecken. Nachdem dies als Plugin ausgelagert wurde, dient die Datei nur noch als eine Art Grundgerüst, um das Plugin aufzurufen.

10.4 Serendipity-Dateien

Serendipity besteht aus einer größeren Menge an Dateien und Systembibliotheken:

Stammverzeichnis
: Das Stammverzeichnis enthält folgende Dateien:

`.htaccess`
: Steueranweisungen für den Webserver (*URL-Umformung* etc.)

`comment.php`
: stellt Kommentare/Trackbacks in einem Popup dar.

`exit.php`
: leitet verfolgte URLs an das gewünschte Ziel weiter (bei Verwendung des Plugins *URL-Exits verfolgen*).

`index.php`
: stellt den zentralen Anlaufpunkt für alle Frontend-Ausgaben dar.

`rss.php`
: gibt RSS-Feeds aus.

`serendipity.css.php`
: gibt die Stylesheets von Serendipity aus.

`serendipity_admin.php`
: stellt den zentralen Anlaufpunkt für alle Backend-Ausgaben dar.

`serendipity_admin_image_selector.php`
: stellt das Mediendatenbank-Popup dar.

`serendipity_config.inc.php`
: lädt das Serendipity Framework.

`serendipity_config_local.inc.php`
: enthält zentrale Konfigurationswerte Serendipitys.

`serendipity_define.js.php`
: konvertiert einige PHP-Variablen, damit sie in JavaScripts zur Verfügung stehen.

`serendipity_editor.js`
: stellt JavaScript-Funktionen der Mediendatenbank bereit.

`serendipity_xmlrpc.php`
: bindet die Anlaufstelle für die XML-RPC API ein.

`wfwcomment.php`
: bindet eine Anlaufstelle für wfwComment-API-Aufrufe ein.

Verzeichnis `archives`
: Dient nur als Platzhalter für das symbolische Verzeichnis `archives`, das Serendipity innerhalb der URL verwendet. Hier werden jedoch keine Dateien gespeichert, sondern diese werden aus der Datenbank gelesen. Einige Plugins speichern in diesem Verzeichnis temporäre Dateien.

Verzeichnis `bundled-libs`
: Enthält einige Fremd-Bibliotheken zur Erweiterung der Funktionen Serendipitys (z. B. `PEAR`-Klassen und `Smarty`).

Verzeichnis `deployment`
: Enthält funktionsreduzierte Rumpf-Dateien für die *Shared Installation* (siehe Seite 673) von Serendipity. Diese entsprechen den Dateinamen des Stammverzeichnisses und binden die entsprechende Datei später ein.

Verzeichnis `docs`
: Enthält einige Dateien zur Dokumentation Serendipitys. Die Datei `NEWS` enthält ein Protokoll von Änderungen zwischen den verschiedenen Versionen Serendipitys.

Verzeichnis `htmlarea`
: Enthält den Sourcecode des `HTMLArea`-WYSIWYG-Editors zur Darstellung einer Microsoft Word-ähnlichen Oberfläche beim Bearbeiten von Artikeln (siehe Seite 104).

Verzeichnis `include`
: Enthält die zentralen Funktionskomponenten Serendipitys:

  `compat.inc.php`
  : enthält Bearbeitungsroutinen, die zur Kompatibilitätswahrung und Vereinheitlichung verschiedener PHP-Versionen nötig sind.

  `functions.inc.php`
  : enthält zentrale Funktionen für Serendipity und lädt die Unter-Funktionsdateien.

  `functions_calendars.inc.php`
  : enthält Funktionen zur Bearbeitung von Datumsangaben und Zeitwerten.

  `functions_comments.inc.php`
  : enthält Funktionen zur Behandlung von Blog-Kommentaren.

  `functions_entries.inc.php`
  : enthält Funktionen zur Behandlung von Blog-Artikeln.

  `functions_entries_admin.inc.php`
  : enthält Funktionen zur Bearbeitung von Blog-Artikeln im Backend.

  `functions_images.inc.php`
  : enthält Funktionen zur Behandlung von Objekten der Mediendatenbank.

`functions_images_crop.inc.php`
: enthält Funktionen zum Beschneiden von Bildern (derzeit noch experimentell).

`functions_installer.inc.php`
: enthält Funktionen zur Installation Serendipitys.

`functions_permalinks.inc.php`
: enthält Funktionen zur Bearbeitung von Permalinks/URL-Werten.

`functions_plugins_admin.inc.php`
: enthält Funktionen zur Konfiguration von Plugins im Backend.

`functions_rss.inc.php`
: enthält Funktionen zur Ausgabe von RSS-Feeds.

`functions_smarty.inc.php`
: enthält Funktionen zur Interaktion mit der Smarty-Bibliothek.

`functions_trackbacks.inc.php`
: enthält Funktionen zur Annahme und zum Versand von Trackbacks.

`functions_upgrader.inc.php`
: enthält Funktionen zum Aktualisierungsvorgang für Serendipity.

`genpage.inc.php`
: enthält Anweisungen zur Erstellung und zur Ausgabe von Seiten im Frontend.

`lang.inc.php`
: enthält Anweisungen zum Laden der notwendigen Sprachdateien.

`plugin_api.inc.php`
: enthält die Klassen der Plugin-Schnittstelle.

`plugin_api_extension.inc.php`
: enthält erweiterte Funktionen, auf die Plugins zurückgreifen können.

`plugin_internal.inc.php`
: enthält eine Liste eingebauter Plugins, die für Serendipity zur Verfügung stehen.

`template_api.inc.php`
: enthält die Klasse der Template-Schnittstelle, wenn eine andere Schnittstelle als Smarty zum Einsatz kommen soll (siehe Seite 692).

Verzeichnis `include/admin`
: Enthält die Unterseiten des Backends:

`category.inc.php`
: enthält die Maske zur Bearbeitung von Kategorien.

`comments.inc.php`
: enthält die Maske zur Bearbeitung von Kommentaren.

`configuration.inc.php`
: enthält die Maske zur Konfiguration Serendipitys.

`entries.inc.php`
: enthält die Maske zum Bearbeiten/Erstellen von Blog-Artikeln.

`entries_overview.inc.php`
: enthält die Maske für die Übersicht der Blog-Artikel.

`export.inc.php`
: enthält die Maske zum Exportieren von Blog-Artikeln.

`groups.inc.php`
: enthält die Maske zur Bearbeitung von Benutzergruppen.

`images.inc.php`
: enthält die Maske zum Verwalten der Mediendatenbank.

`import.inc.php`
: enthält die Maske zum Importieren von Blog-Artikeln.

`installer.inc.php`
: enthält die Maske zur Installation Serendipitys.

`overview.inc.php`
: enthält die Maske zur Startseite des Backends.

`personal.inc.php`
: enthält die Maske zu den persönlichen Einstellungen.

`plugins.inc.php`
: enthält die Maske zur Verwaltung von Plugins.

`templates.inc.php`
: enthält die Maske zum Verwalten von Templates.

`upgrader.inc.php`
: enthält die Maske zum Aktualisieren von Serendipity.

`users.inc.php`
: enthält die Maske zur Verwaltung von Redakteuren.

Verzeichnis `include/admin/importers`
: Enthält die verschiedenen Import-Module für die Schnittstelle zu anderen Blog-Systemen. Der Dateiname steht jeweils für das Blog-System, die Datei `generic.inc.php` enthält den RSS-Feed-Import.

Verzeichnis `include/db`
: Enthält die Funktionsbibliotheken zur Verwendung unterschiedlicher Datenbanksysteme. Die Datei `db.inc.php` enthält die systemübergreifenden Funktionen, die anderen Dateien enthalten den Funktionssatz des jeweils gleichnamigen Datenbanksystems.

Verzeichnis `include/tpl`
: Enthält einige Vorlagedateien zur Darstellung von Konfigurationsoptionen des Blogs.

Die `htacces_....tpl`-Dateien enthalten Vorlagen für die automatisch erstellte `.htaccess`-Datei, in Abhängigkeit von der gewählten Form der URL-Umformung und der eingesetzten PHP-Version (mit/ohne CGI-Einbindung). Die Datei `config_local.inc.php` enthält die Anweisungen zur zentralen Blog-Konfiguration (siehe Seite 586) und `config_personal.inc.php` die persönlichen Einstellungen.

Verzeichnis `lang`
: Enthält die Sprachdateien zur Übersetzung der Serendipity-Meldungen. Das Kürzel am Ende jeder Datei steht für die entsprechende Sprache (`de` = Deutsch, `en` = Englisch usw.). Die Dateien `addlang.*`, `plugin_lang.php` und `append.sh` sind nur für Entwickler gedacht, damit einfach neue Sprachkonstanten in alle Dateien hinzugefügt werden können.

  Innerhalb dieser Datei sorgt der PHP-Befehl `define()` dafür, einer Konstante (erster Parameter) einen beliebigen Wert (zweiter Parameter) zuzuordnen. Wenn einfache oder doppelte Anführungszeichen im Inhalt einer Konstante vorkommen, müssen Sie darauf achten, diese mit einem Backslash (\) zu escapen:

```
define('KONSTANTENNAME1', 'Ich bin\'s, der "Waldi"!');
define('KONSTANTENNAME2', ''Ich bin's, der
\"Waldi\"!');
```

  Serendipity setzt das @-Zeichen vor jeden `define()`-Aufruf, damit etwaige doppelte Deklarationen einer Konstante keine Fehlermeldung provozieren.

  Das Unterverzeichnis `UTF-8` enthält dieselben Dateien ein weiteres Mal, jedoch entgegen zur nationalen Zeichensatzkonvertierung (meist ISO) im UTF-8-Zeichensatz.

Verzeichnis `plugins`
: Enthält die installierbaren Plugins. Der Name des jeweiligen Unterverzeichnisses entspricht dem Namen der Plugin-Dateien. Hier abgelegte Plugins müssen zuerst über das Backend installiert werden, bevor sie aktiv sind.

Verzeichnis `sql`
: Enthält Dateien mit SQL-Anweisungen. Die Datei `db.sql` enthält alle Anweisungen, die bei der Installation Serendipitys ausgeführt werden. Diese Datei enthält einige Variablen (eingefasst in geschweiften Klammern), die bei der Installation automatisch entsprechend des Datenbank-Zielsystems in den korrekten SQL-Dialekt übersetzt werden.

  Die Dateien `db_update_X_Y_Z.sql` enthalten die SQL-Anweisungen, die beim Aktualisieren von Serendipity automatisch ausgeführt werden sollen, um neue Tabellen zu erstellen bzw. bestehende Tabellen anzupassen. Dabei steht `X` für die vorher installierte Serendipity-Version und `Y` für die neue Version. Bei einem Update werden alle Versionszwischenstände ebenfalls ausgeführt. `Z` steht dabei für das verwendete Datenbank-Zielsystem (MySQL, PostgreSQL oder SQLite).

Verzeichnis `templates`
: Enthält die verfügbaren Templates. Das Verzeichnis `carl_contest` enthält seit Serendipity 1.0 alle Standarddateien, die verwendet werden, wenn ein Template-Verzeichnis nicht über eine entsprechende Datei verfügt. Das Verzeichnis `default` enthält darüber hinaus weitere Standarddateien, falls diese nicht im Verzeichnis `carl_contest` liegen. Diese beiden Sonderverzeichnisse sollten daher bei jeder Serendipity-Installation beibehalten und stets aktualisiert werden. Änderungen an diesen Dateien sollten Sie niemals dort vornehmen, sondern stattdessen in einer lokalen Kopie eines eigenen Template-Verzeichnisses.

Verzeichnis `templates_c`
: Enthält die von Smarty erstellten Temporärdateien mit in PHP-Code übersetzten Templates.

Verzeichnis `tests`
: Enthält (rudimentäre) Unit-Test-Dateien für Serendipity.

Verzeichnis `uploads`
: In diesem Verzeichnis werden die Dateien der Mediendatenbank gespeichert.

10.5 Serendipity-Funktionen

Auf eine ausführliche Beschreibung aller Serendipity-Funktionen muss aus Platzgründen in diesem Buch verzichtet werden. Alle verfügbaren Parameter sowie Rückgabewerte sind jedoch als PHPDoc-Kommentare oberhalb der Funktionsdeklaration jeder Serendipity-Kerndatei enthalten und können dort leicht nachgeschlagen werden. Der Name der Datei, in der sich die jeweilige Funktion befindet, ist in der folgenden Liste aufgeführt.

Im Folgenden finden Sie eine Liste von gebräuchlichen Serendipity-Funktionen, die Sie bei der Entwicklung von Plugins und Ähnlichem häufig einsetzen werden.

10.5.1 Zentrale Funktionen

Zentrale Funktionen können an mehreren Stellen Serendipitys eingesetzt werden.

`serendipity_die()` (`include/compat.inc.php`)
: Gibt eine (beliebige) Fehlermeldung aus, falls die reguläre Ausführung von Serendipity unmöglich ist.

`serendipity_mb()` (`include/lang.inc.php`)
: Dient als Wrapper-Funktion, um Multi-Byte-Zeichensatzoperationen durchzuführen. PHP bietet für derartige Multi-Byte-Sprachen (asiatische) spezielle Funktionen an, was Serendipity für den Programmierer einfacher ansprechbar werden lässt.

`serendipity_serverOffsetHour()` (`include/functions.inc.php`)
: Liefert einen UNIX-Zeitstempel zurück, der abhängig von der Konfiguration des Servers in die richtige Zeitzone umgerechnet wurde.

`serendipity_strftime(), serendipity_formatTime()`

(`include/functions.inc.php`) Konvertiert einen UNIX-Zeitstempel in ein lesbares Format und wendet etwaige Zeitzonen-Umrechnungen an.

`serendipity_walkRecursive()` (`include/functions.inc.php`)
: Dient als Hilfsfunktion, um eine hierarchische Struktur innerhalb eines eindimensionalen Arrays zu durchwandern und aufzubereiten. Dies wird vor allem für verschachtelte Kategorien und Kommentare verwendet, die eine Verschachtelung anhand eines `parent_id`-Array-Schlüssels angeben.

  Üblicherweise enthält z. B. die Liste von Kategorien keine hierarchische Struktur, sondern jede Kategorie ist in eine eindimensionale Liste alphabetisch einsortiert. Der Sinn der Funktion `serendipity_walkRecursive` ist es nun, das Array später korrekt eingerückt darzustellen, so dass Unterkategorien sequentiell direkt auf die Oberkategorie folgen und mit Leerzeichen (oder anderen Zeichen) eingerückt werden.

`serendipity_sendMail()` (`include/functions.inc.php`)
: Sendet eine E-Mail mit Serendipity-Kopf- und -Fußzeilen.

`serendipity_utf8_encoded()` (`include/functions.inc.php`)
: Kodiert eine Zeichenkette im Zeichensatz des Blogs in das UTF-8-Format. Falls die Zeichenkette bereits im UTF-8-Format vorliegt, wird keine weitere Umkodierung vorgenommen.

`serendipity_parseFileName()` (`include/functions.inc.php`)
: Teilt eine Datei in ihre Bestandteile auf (Grundname, Datei-Endung).

`serendipity_trackReferrer()` (`include/functions.inc.php`)
: Liest den HTTP-Referrer aus und speichert ihn für statistische Zwecke in der Datenbank.

`serendipity_request_start(), serendipity_request_end()`

(`include/functions.inc.php`) Führt bestimmte Operationen aus, bevor/nachdem Serendipity eine fremde URL aufruft. Dies dient der Verhinderung von Schreibproblemen mit PHP-Sessions.

`serendipity_build_query()` (`include/functions.inc.php`)
: Erstellt eine Zeichenkette anhand eines übergebenen Arrays, damit dessen Werte z. B. in einer HTTP-GET-URL übertragen werden können.

`serendipity_pickKey()` (`include/functions.inc.php`)
Dient als Smarty-Modifier, um die Elemente eines Unter-Arrays auszuliefern.

`serendipity_getTemplateFile()`

(`include/functions_config.inc.php`) Liefert den vollen Pfad zu einer gewünschten Datei des aktuellen Templates.

`serendipity_load_configuration()`

(`include/functions_config.inc.php`) Liest alle Konfigurationswerte Serendipity in das zentrale `$serendipity`-Array.

`serendipity_logout()` (`include/functions_config.inc.php`)
Führt den Logout eines Benutzers durch.

`serendipity_session_destroy()`

(`include/functions_config.inc.php`) Löscht alle PHP-Sessiondaten des aktuellen Benutzers, um ihn vollständig auszuloggen.

`serendipity_login(), serendipity_authenticate_author`

(`include/functions_config.inc.php`) Wertet den Login eines Redakteurs aus und initalisiert dessen Konfigurationswerte.

`serendipity_issueAutologin(), serendipity_setAuthorToken()`

(`include/functions_config.inc.php`) Setzt ein Login-Cookie, damit ein Redakteur automatisch bei den folgenden Seitenaufrufen eingeloggt werden kann.

`serendipity_checkAutologin()`

(`include/functions_config.inc.php`) Wertet ein automatisches Login-Cookie eines Redakteurs aus.

`serendipity_userLoggedIn()`

(`include/functions_config.inc.php`) Liefert zurück, ob ein Redakteur momentan eingeloggt ist.

`serendipity_JSsetcookie()`

(include/functions_config.inc.php) Gibt JavaScript-Code aus, um ein Cookie zu setzen.

serendipity_setCookie()

(include/functions_config.inc.php) Setzt ein Cookie.

serendipity_deleteCookie()

(include/functions_config.inc.php) Löscht ein Cookie.

serendipity_is_iframe()

(include/functions_config.inc.php) Prüft, ob ein iframe zur Vorschau eines Artikels im Backend eingebunden werden muss.

serendipity_iframe(), serendipity_iframe_create()

(include/functions_config.inc.php) Erstellt einen iframe zur Artikelvorschau im Backend.

serendipity_probeInstallation()

(include/functions_config.inc.php) Prüft, welche Voraussetzungen zur Installation der Server unterstützt (Datenbanken, URL-Umformung).

serendipity_header() (include/functions_config.inc.php)
Sendet einen HTTP-Header.

serendipity_getSessionLanguage()

(include/functions_config.inc.php) Liefert die aktuelle Sprache des Besuchers.

is_utf8() (include/functions_trackbacks.inc.php)
Prüft, ob eine Zeichenkette im UTF-8-Zeichensatz vorliegt.

_serendipity_send() (include/functions_trackbacks.inc.php)
Öffnet eine Socket-Verbindung zu einem fremden Server und sendet Daten (meist Trackbacks/Pingbacks).

10.5.2 Redakteure, Rechte

Funktionen zur Verwaltung von Redakteuren und deren Rechten.

`serendipity_fetchUsers()` (`include/functions.inc.php`)
Holt ein Array mit der Liste aller verfügbaren Serendipity-Redakteure.

`serendipity_fetchUser()` (`include/functions.inc.php`)
Holt ein Array mit den Eigenschaften eines speziellen Serendipity-Redakteurs.

`serendipity_addAuthor()` (`include/functions_config.inc.php`)
Fügt einen neuen Redakteur hinzu.

`serendipity_deleteAuthor()`

(`include/functions_config.inc.php`) Löscht einen Redakteur.

`serendipity_remove_config_var()`

(`include/functions_config.inc.php`) Entfernt eine Konfigurationsoption eines Redakteurs.

`serendipity_set_config_var(), serendipity_set_user_var()`

(`include/functions_config.inc.php`) Setzt einen Konfigurationswert eines Redakteurs.

`serendipity_get_config_var(), serendipity_get_user_var()`

(`include/functions_config.inc.php`) Liest den Konfigurationswert eines Redakteurs aus.

`serendipity_getPermissions()`

(`include/functions_config.inc.php`) Liefert die Rechte eines Redakteurs zurück.

`serendipity_getPermissionNames()`

(`include/functions_config.inc.php`) Liefert alle verfügbaren internen Rechte sowie deren Einteilung in die Benutzerränge von älteren Serendipity-Versionen zurück.

`serendipity_getDBPermissionNames()`

(`include/functions_config.inc.php`) Liefert alle in der Datenbank hinterlegten Rechte zurück. Dort können auch etwaige von Plugins vergebene Rechte miteinbezogen werden.

`serendipity_getAllPermissionNames()`

(`include/functions_config.inc.php`) Liefert alle verfügbaren Rechte (interne und aus der Datenbank) zurück.

`serendipity_checkPermission()`

(`include/functions_config.inc.php`) Prüft, ob ein Redakteur ein bestimmtes Recht besitzt.

`serendipity_updateGroups()`

(`include/functions_config.inc.php`) Aktualisiert die Gruppenmitgliedschaften eines Redakteurs.

`serendipity_getAllGroups()`

(`include/functions_config.inc.php`) Liefert alle eingerichteten Benutzergruppen zurück.

`serendipity_fetchGroup()` (`include/functions_config.inc.php`)
Liefert die einer Benutzergruppe zugeordneten Rechte zurück.

`serendipity_getGroups()` (`include/functions_config.inc.php`)
Liefert alle Gruppenmitgliedschaften eines Redakteurs zurück.

`serendipity_getGroupUsers()`

(`include/functions_config.inc.php`) Liefert alle Mitglieder einer gewünschten Benutzergruppe zurück.

`serendipity_deleteGroup()`

(`include/functions_config.inc.php`) Löscht eine Benutzergruppe.

`serendipity_addGroup()` (`include/functions_config.inc.php`)
Fügt eine Benutzergruppe hinzu.

`serendipity_addDefaultGroup()`

include/functions_config.inc.php) Konvertiert einen Benutzerrang einer alten Serendipity-Installation in eine neue Benutzergruppe gleichen Namens.

serendipity_intersectGroup()

(include/functions_config.inc.php) Prüft, ob ein gewünschter Benutzer in derselben Gruppe ist wie der derzeit eingeloggte Redakteur.

serendipity_updateGroupConfig()

(include/functions_config.inc.php) Aktualisiert die einer Benutzergruppe zugewiesenen Rechte.

serendipity_ACLGrant() (include/functions_config.inc.php)
Gewährt einer Benutzergruppe Zugriff auf einen speziellen Objekttyp (hauptsächlich für Objekte der Mediendatenbank).

serendipity_ACLGet() (include/functions_config.inc.php)
Prüft, ob ein Benutzer anhand seiner Gruppenmitgliedschaften Zugriff auf einen speziellen Objekttyp besitzt.

serendipity_ACL_SQL() (include/functions_config.inc.php)
Liefert Teile einer SQL-Abfrage, die benötigt werden, um Zugriffe auf bestimmte Objekttypen (Mediendatenbank, Kategorien ...) in Abhängigkeit von den Gruppenmitgliedschaften auszuwerten.

serendipity_hasPluginPermissions()

(include/functions_config.inc.php) Prüft, ob es einem Redakteur erlaubt ist, die Funktionalitäten eines auszuführenden Plugins aufzurufen (siehe Seite 191).

serendipity_checkXSRF(), serendipity_reportXSRF()

(include/functions_config.inc.php) Prüft bei einem Seitenaufruf, ob der Redakteur diese Aktion tatsächlich willentlich ausgelöst hat oder ob er Opfer einer XSRF-Attacke (siehe Seite 23) war.

serendipity_checkFormToken(), serendipity_setFormToken()

(include/functions_config.inc.php) Setzt bzw. überprüft einen eindeutigen Formularwert (*Token*) auf gültigen Inhalt. Dies wird für die XSRF-Prüfung benötigt.

`show_plugins()` (`include/functions_plugins_admin.inc.php`)
: Erstellt die Plugin-Verwaltungsansicht im Backend für eine gewünschte Plugin-Sektion (Seitenleisten-Plugins, Ereignis-Plugins). Links zum Konfigurieren und Verschieben der Plugins werden mit ausgeliefert. Über weitere Hilfsfunktionen `ownership()` und `placement_box()` werden Teilelemente des Layouts (Plugin-Eigentümer, Platzierung) ausgeliefert.

`serendipity_plugin_config()`

(`include/functions_plugins_admin.inc.php`) Stellt die Konfigurationsmaske für Plugins dar. Dabei werden die Konfigurationswerte der Plugin API entnommen.

10.5.3 Artikel, Kategorien, Kommentare, Trackbacks

Funktionen zur Verwaltung von Artikeln, Kommentaren/Trackbacks und Kategorien.

`serendipity_addCategory()` (`include/functions.inc.php`)
: Fügt dem Blog eine neue Kategorie hinzu.

`serendipity_updateCategory()` (`include/functions.inc.php`)
: Ändert eine bestehende Kategorie des Blogs.

`serendipity_rememberComment()`,

`serendipity_rememberCommentDetails()`,

`serendipity_forgetCommentDetails()`

(`include/functions_comments.inc.php`) Speichert/löscht die Eingaben eines Kommentarformulars in Cookie-Werten.

`serendipity_displayCommentForm()`

(`include/functions_comments.inc.php`) Stellt das Kommentarformular dar und weist die notwendigen Smarty-Variablen zu.

`serendipity_fetchComments()`

(`include/functions_comments.inc.php`) Holt eine Liste gewünschter Kommentare.

`serendipity_generateCommentList()`

(include/functions_comments.inc.php) Erstellt ein HTML SELECT-Auswahlfeld mit einer Liste aller gewünschten Kommentare.

serendipity_printComments()

(include/functions_comments.inc.php) Stellt eine Liste gewünschter Kommentare mittels Smarty-Templates dar.

serendipity_printCommentsByAuthor()

(include/functions_comments.inc.php) Stellt eine Liste gewünschter Kommentare anhand der Übergabe eines Kommentatornamens mittels Smarty-Templates dar.

serendipity_deleteComment()

(include/functions_comments.inc.php) Löscht einen Blog-Kommentar.

serendipity_allowCommentsToggle()

(include/functions_comments.inc.php) Schaltet einen Blog-Artikel für die Kommentierung frei oder hebt diese Möglichkeit auf.

serendipity_approveComment()

(include/functions_comments.inc.php) Schaltet einen moderierten Kommentar frei.

serendipity_saveComment()

(include/functions_comments.inc.php) Speichert einen neuen Kommentar zu einem Blog-Artikel und führt etwaige Anti-Spam-Prüfungen durch.

serendipity_mailSubscribers()

(include/functions_comments.inc.php) Schickt eine E-Mail an alle Abonnenten eines Blog-Artikels.

serendipity_cancelSubscription()

(include/functions_comments.inc.php) Hebt ein Abonnement eines Blog-Artikels auf.

`serendipity_sendComment()`

(`include/functions_comments.inc.php`) Verschickt beim Eintreffen eines neuen Blog-Kommentars eine Information per E-Mail.

`serendipity_deleteCategory()`

`include/functions_entries.inc.php`) Löscht eine Kategorie.

`serendipity_fetchCategoryRange()`

(`include/functions_entries.inc.php`) Liefert eine Liste von Blog-Kategorien einer gewünschten Oberkategorie.

`serendipity_getMultiCategoriesSQL()`

(`include/functions_entries.inc.php`) Liefert einen SQL-Codeteil, damit eine Artikelabfrage auf bestimmte Kategoriezuordnungen eingeschränkt werden kann. Dabei werden etwaige Ober- und Unterkategorien korrekt berücksichtigt.

`serendipity_fetchCategoryInfo()`

(`include/functions_entries.inc.php`) Liefert die Eigenschaften einer gewünschten Kategorie.

`serendipity_fetchEntries()`

(`include/functions_entries.inc.php`) Zentrale Funktion, um eine Liste von Blog-Artikeln zu erhalten, die bestimmten Kriterien entsprechen.

`serendipity_fetchEntryData()`,

`serendipity_fetchEntryProperties()`,

(`include/functions_entries.inc.php`) Weist die erweiterten Eigenschaften von Artikeln und Kategoriezuordnungen einer Liste von Blog-Artikeln zu, die mittels `serendipity_fetchEntries()` bezogen wurden.

`serendipity_fetchEntry()`

(`include/functions_entries.inc.php`) Liefert einen speziellen Artikeldatensatz mit allen erweiterten Eigenschaften und Kategoriezuordnungen zurück.

`serendipity_fetchCategories()`

(`include/functions_entries.inc.php`) Liefert eine Liste von Blog-Kategorien, auf die ein gewünschter Redakteur Zugriff hat. Kann auch unabhängig vom Redakteur eine Liste aller Kategorien des Blogs liefern.

`serendipity_rebuildCategoryTree()`

(`include/functions_entries.inc.php`) Führt die notwendigen Datenbankoperationen aus, um bei neu hinzugefügten Kategorien die IDs der Nested-Set-Architektur (siehe z. B. `http://www.klempert.de/nested_sets/`) korrekt zu setzen.

`serendipity_searchEntries()`

(`include/functions_entries.inc.php`) Liefert eine Liste von Blog-Artikeln zurück, die ein gewünschtes Suchwort enthalten.

`serendipity_getTotalEntries()`

(`include/functions_entries.inc.php`) Liefert die Anzahl von aktuell im Frontend dargestellten Artikeln zurück.

`serendipity_printEntries()`

(`include/functions_entries.inc.php`) Zentrale Funktion, um Blog-Einträge für die Ausgabe im Smarty-Template aufzubereiten. Benötigte Variablen und Arrays werden gesetzt, etwaige Plugins ausgeführt und an das Smarty-Template `entries.tpl` weitergegeben.

`serendipity_printEntries_rss()`

(`include/functions_entries.inc.php`) Stellt Einträge innerhalb eines RSS-Feeds dar.

`serendipity_purgeEntry(), serendipity_deleteEntry()`

(`include/functions_entries.inc.php`) Löscht einen Artikel.

`serendipity_updertEntry()`

(`include/functions_entries.inc.php`) Aktualisiert einen Artikel oder fügt einen neuen ein (*updert* = *Upd*ate + Ins*ert*).

`serendipity_generateCategoryList()`

(`include/functions_entries.inc.php`) Erzeugt ein HTML-Ausklappfeld mit allen verfügbaren Kategorien.

`serendipity_generateCategoryList()`

(`include/functions_entries.inc.php`) Aktualisiert die Kategoriezuordnungen eines Artikels.

`serendipity_printArchives()`

(`include/functions_entries.inc.php`) Erzeugt eine Archivübersicht mit Artikeln eines gewünschten Zeitraums und gibt diese Artikelliste an das Smarty-Template `entries_archives.tpl` weiter.

`serendipity_getTotalCount()`

(`include/functions_entries.inc.php`) Liefert die Anzahl aller im Blog vorhandenen Kommentare, Trackbacks oder Artikel zurück.

`serendipity_printEntryForm()`

(`include/functions_entries_admin.inc.php`) Stellt das HTML-Formular zum Bearbeiten eines Blog-Artikels im Backend dar.

`serendipity_emit_htmlarea_code()`

(`include/functions_entries_admin.inc.php`) Gibt den benötigten JavaScript-Code zurück, damit ein WYSIWYG-Editor angezeigt werden kann.

`serendipity_handle_references()`

(`include/functions_trackbacks.inc.php`) Überprüft alle in einem Blog-Artikel enthaltenen Links und speichert sie in den entsprechenden Datenbanktabellen.

`serendipity_trackback_is_success()`,

`serendipity_pingback_is_success()`

(`include/functions_trackbacks.inc.php`) Prüft, ob ein Trackback/Pingback gültig versendet wurde.

`serendipity_pingback_autodiscover()`,

`serendipity_trackback_autodiscover()`

(`include/functions_trackbacks.inc.php`) Überprüft einen Blog-Artikel auf Trackback/Pingback-Links und führt diese aus.

`serendipity_reference_autodiscover()`

(`include/functions_trackbacks.inc.php`) Überprüft einen Blog-Artikel auf enthaltene Hyperlinks und speichert diese in der Datenbank.

`add_trackback(), add_pingback()`

(`include/functions_trackbacks.inc.php`) Speichert ein eingehendes Trackback/Pingback.

`report_trackback_success()`, `report_trackback_failure()`,

`report_pingback_success()`, `report_pingback_failure()`

(`include/functions_trackbacks.inc.php`) Liefert XML-Code für eine Erfolgs-/Fehlermeldung bei einem Trackback/Pingback zurück.

10.5.4 Permalinks

Funktionen zur Behandlung von URLs und Permalinks.

`serendipity_makeFilename()`

(`include/functions_permalinks.inc.php`) Konvertiert eine Zeichenkette in ein Format, das innerhalb einer URL ohne Sonderzeichen verwendet werden kann.

`serendipity_initPermalinks()`

(`include/functions_permalinks.inc.php`) Setzt die Serendipity-Variablen, die die Konfiguration der Permalinks enthalten.

`serendipity_permalinkPatterns()`

(`include/functions_permalinks.inc.php`) Erzeugt eine Liste von regulären Ausdrücken, die verwendet werden, um zu prüfen, ob die aktuelle URL einem festgelegten Permalink entspricht.

`serendipity_searchPermalink()`

(`include/functions_permalinks.inc.php`) Durchsucht die Datenbank nach einem Permalink für die aktuelle URL.

`serendipity_getPermalink(), serendipity_rewriteURL(),`

`serendipity_archiveURL(), serendipity_authorURL(),`

`serendipity_categoryURL(),`

`serendipity_feedCategoryURL(),`

`serendipity_feedAutorURL(),`

`serendipity_archiveDateUrl()`

(`include/functions_permalinks.inc.php`) Gibt einen Permalink zu einem Objekt (Blog-Artikel, Redakteur, Kategorie) aus.

`serendipity_updatePermalink()`

(`include/functions_permalinks.inc.php`) Aktualisiert einen in der Datenbank gespeicherten Permalink.

`serendipity_insertPermalink()`

(`include/functions_permalinks.inc.php`) Fügt einen Permalink in die Datenbank ein, so dass die Ansicht des gewünschten Objekts (Artikel, Redakteur, Kategorie) später über diese URL wieder aufgerufen werden kann.

`serendipity_buildPermalinks()`

(`include/functions_permalinks.inc.php`) Erzeugt Permalinks in der Datenbank für alle Artikel, Redakteure und Kategorien des Blogs. Dabei wird die momentan konfigurierte Permalink-Struktur verwendet.

`serendipity_makePermalinks()`,

`serendipity_makePermalinkRegex()`

(`include/functions_permalinks.inc.php`) Erzeugt einen Permalink, indem die Platzhalter der Permalink-Konfiguration durch die Werte des aufrufenden Objekts ersetzt werden.

`serendipity_currentURL()`

(`include/functions_permalinks.inc.php`) Liefert die URL des im Frontend dargestellten Permalinks.

`serendipity_getUriArguments()`

(`include/functions_permalinks.inc.php`) Liefert die URL-Bestandteile des im Frontend dargestellten Permalinks.

10.5.5 Installation, Upgrades

Funktionen zur Installation und Aktualisierung Serendipitys.

`serendipity_ini_bool()`, `serendipity_ini_bytesize()`

(`include/functions_installer.inc.php`) Liest PHP-Konfigurationswerte ein und wandelt sie in ein von Serendipity interpretierbares Format um.

`serendipity_updateLocalConfig()`

(`include/functions_installer.inc.php`) Aktualisiert die zentrale Konfigurationsdatei `serendipity_config_local.inc.php` mit den aktuellen Konfigurationswerten.

`serendipity_installDatabase()`,

`serendipity_parse_sql_tables()`,

`serendipity_parse_sql_inserts()`

(`include/functions_installer.inc.php`) Richtet die Datenbank ein und erstellt die Serendipity-Tabellen.

`serendipity_query_default()`

(`include/functions_installer.inc.php`) Prüft einen Konfigurationswert auf jeweils sinnvolle Voreinstellungen bei der Installation.

`serendipity_check_rewrite()`

(`include/functions_installer.inc.php`) Prüft, ob der Server dynamische URL-Umformung unterstützt.

`serendipity_parseTemplate(),`

`serendipity_printConfigTemplate()`

(`include/functions_installer.inc.php`) Stellt die Konfigurationsoptionen dar, die durch die Dateien `include/tpl/config_local.inc.php` oder `include/tpl/config_personal.inc.php` bestimmt werden.

`serendipity_checkConfigItemFlags()`

(`include/functions_installer.inc.php`) Prüft die Werte der einzelnen Konfigurationsoptionen, um deren Ausgabeort zu bestimmen.

`serendipity_guessInput()`

(`include/functions_installer.inc.php`) Gibt den HTML-Code für eine Konfigurationsoption zurück (einzeilige Texteingabefelder, Ja/Nein-Schalter, Auswahlfelder etc.)

`serendipity_checkInstallation()`

(`include/functions_installer.inc.php`) Prüft, ob die Serendipity-Installation problemlos durchgeführt werden konnte.

`serendipity_installFiles()`

(`include/functions_installer.inc.php`) Erzeugt die für Serendipity notwendigen Dateien bei der Installation (`.htaccess` und `serendipity_config_local.inc.php`).

`serendipity_updateConfiguration()`

(include/functions_installer.inc.php) Aktualisiert die zentralen Serendipity-Dateien und die Datenbank, wenn sich wichtige Konfigurationswerte (Permalinks) ändern.

`serendipity_httpCoreDir()`

(include/functions_installer.inc.php) Liefert den aktuellen Systempfad zum Serendipity-Verzeichnis.

10.5.6 Bilder

Funktionen zur Mediendatenbank.

`serendipity_isActiveFile()`

(include/functions_images.inc.php) Prüft, ob eine hochzuladende Datei eine potenziell gefährliche Dateiendung besitzt.

`serendipity_fetchImagesFromDatabase()`

(include/functions_images.inc.php) Holt eine Liste von gewünschten Objekten der Mediendatenbank.

`serendipity_fetchImageFromDatabase()`

(include/functions_images.inc.php) Holt eine einzelnes Objekt aus der Mediendatenbank.

`serendipity_updateImageInDatabase()`

(include/functions_images.inc.php) Aktualisiert ein Objekt der Mediendatenbank.

`serendipity_deleteImage()`

(include/functions_images.inc.php) Löscht ein Objekt der Mediendatenbank.

`serendipity_fetchImages()`

(include/functions_images.inc.php) Holt eine Liste von Mediendateien direkt aus der Verzeichnisstruktur anstelle der Mediendatenbank.

`serendipity_insertHotlinkedImageInDatabase()`

(`include/functions_images.inc.php`) Fügt einen Verweis zu einer Mediendatei eines fremden Servers in die eigene Mediendatenbank ein.

`serendipity_insertImageInDatabase()`

(`include/functions_images.inc.php`) Fügt der Mediendatenbank eine Datei hinzu.

`serendipity_makeThumbnail()`

(`include/functions_images.inc.php`) Erzeugt ein Vorschaubild einer Datei.

`serendipity_scaleImg(),`

`serendipity_resize_image_gd()`

(`include/functions_images.inc.php`) Ändert die Größe eines Bildes.

`serendipity_rotateImg(), serendipity_rotate_image_gd()`

(`include/functions_images.inc.php`) Rotiert ein Bild.

`serendipity_generateThumbs(), serendipity_syncThumbs()`

(`include/functions_images.inc.php`) Erzeugt Vorschaubilder aller Dateien der Mediendatenbank.

`serendipity_guessMime()`

(`include/functions_images.inc.php`) Erkennt den MIME-Typ einer Datei anhand des Dateinamens.

`serendipity_functions_gd()`

(`include/functions_images.inc.php`) Kapselt PHP-Grafikfunktionen und liefert den richtigen PHP-Funktionsnamen in Abhängigkeit vom gewählten Dateiformat zurück.

`serendipity_calculate_aspect_size()`

(include/functions_images.inc.php) Berechnet die notwendigen Bildmaße, wenn ein Bild proportional verkleinert oder vergrößert werden soll.

`serendipity_displayImageList()`

(include/functions_images.inc.php) Stellt eine Übersicht der Objekte der Mediendatenbank dar. Die darzustellenden Objekte werden anhand der Benutzereingaben ausgelesen und mittels Smarty-Variablen und Templates dargestellt.

`serendipity_isImage()`

(include/functions_images.inc.php) Prüft, ob ein Objekt der Mediendatenbank ein Bild ist.

`serendipity_killPath(), serendipity_deletePath()`

(include/functions_images.inc.php) Löscht ein Verzeichnis mitsamt Unterverzeichnissen und darin enthaltenen Dateien.

`serendipity_traversePath()`

(include/functions_images.inc.php) Durchsucht ein Verzeichnis mitsamt seiner Unterverzeichnisse nach Dateien und liefert eine Verzeichnis- und Dateiliste zurück.

`serendipity_sortPath()`

(include/functions_images.inc.php) Sortiert eine Verzeichnisliste.

`serendipity_uploadSecure()`

(include/functions_images.inc.php) Prüft einen Dateinamen, der von einem Redakteur angegeben wurde, und entfernt potenziell gefährliche Sonderzeichen.

`serendipity_getimagesize()`

(include/functions_images.inc.php) Liefert die Datei- und Bildgröße sowie den MIME-Typ einer Datei zurück.

`serendipity_getImageFields()`

(include/functions_images.inc.php) Liefert ein Array mit möglichen Datenbank-Feldnamen der Objekte einer Mediendatenbank.

`serendipity_directoryACL()`

(`include/functions_images.inc.php`) Prüft die Zugriffsrechte eines Redakteurs auf die Verzeichnisse der Mediendatenbank.

`serendipity_getImageData()`

(`include/functions_images.inc.php`) Liest die Metadaten eines Objekts aus der Mediendatenbank.

`serendipity_showPropertyForm()`

(`include/functions_images.inc.php`) Stellt das Formular zum Bearbeiten der Metadaten einer Datei in der Mediendatenbank dar.

`serendipity_parseMediaProperties()`

(`include/functions_images.inc.php`) Liest die Metadaten eines Objekts der Mediendatenbank und bereitet sie für die Darstellung auf.

`serendipity_mediaTypeCast()`,

`serendipity_metaFieldConvert()`,

`serendipity_getMediaRaw()`

(`include/functions_images.inc.php`) Versucht, Metadaten einer Datei (EXIF-Daten, ID3-Daten) in ein für die Datenbank verwendbares Format zu bringen. Konvertiert Datums- und Zahlenwerte in das benötigte Format.

`serendipity_insertMediaProperty()`

(`include/functions_images.inc.php`) Speichert Metadaten einer Datei in der Mediendatenbank.

`serendipity_parsePropertyForm()`

(`include/functions_images.inc.php`) Bereitet die vom Benutzer übermittelten Metadaten einer Datei auf und speichert sie entsprechend in der Mediendatenbank.

`serendipity_fetchMediaProperties()`,

`serendipity_getMetaData()`

(`include/functions_images.inc.php`) Liest Metadaten eines Objekts aus der Mediendatenbank.

`serendipity_checkPropertyAccess()`

(`include/functions_images.inc.php`) Prüft, ob ein Redakteur auf bestimmte Objekte der Mediendatenbank zugreifen darf, und entfernt alle nicht zulässigen Metadaten aus einer Liste.

`serendipity_prepareMedia()`

(`include/functions_images.inc.php`) Bereitet die Metadaten eines Objekts der Mediendatenbank auf und setzt benötigte Zusatzvariablen für die Darstellung eines Objekts.

`serendipity_showMedia()`

(`include/functions_images.inc.php`) Stellt ein Objekt der Mediendatenbank mittels Smarty-Templates (`media_pane.tpl` oder `media_properties.tpl`) dar.

`serendipity_imageAppend()`

(`include/functions_images.inc.php`) Prüft, ob eine Datei in der Mediendatenbank bereits existiert. Ist das der Fall, wird ein numerischer Index an die Datei angehängt (aus `bild.jpg` wird `bild2.jpg`).

`serendipity_checkMediaSize()`

(`include/functions_images.inc.php`) Prüft, ob eine hochgeladene Datei bestimmte Ressourcenlimits überschreitet.

`serendipity_moveMediaDirectory()`

(`include/functions_images.inc.php`) Verschiebt ein Verzeichnis innerhalb der Mediendatenbank und im Verzeichnissystem.

`serendipity_getMediaPaths()`

(`include/functions_images.inc.php`) Holt eine Liste aller Verzeichnisse der Mediendatenbank, auf die ein Redakteur Zugriff hat.

`serendipity_checkDirUpload()`

(`include/functions_images.inc.php`) Prüft, ob ein Redakteur Schreibzugriff auf ein bestimmte Verzeichnis der Mediendatenbank besitzt.

10.5.7 Smarty

Funktionen zur Verwaltung von Smarty. Die meisten von den in `include/functions_smarty.inc.php` festgelegten Funktionen dienen als Smarty-Modifier oder Smarty-Functions. Diese sind ab Seite 563 und Seite 573 dokumentiert.

`serendipity_smarty_init()`

(`include/functions_smarty.inc.php`) Startet das Smarty-Framework und initialisiert alle notwendigen Variablen und Objekte.

`serendipity_smarty_purge()`

(`include/functions_smarty.inc.php`) Löscht die temporären Dateien des Smarty-Frameworks (kompilierte Templates im `templates_c`-Verzeichnis).

`serendipity_smarty_shutdown()`

(`include/functions_smarty.inc.php`) Beendet die Seitenausführung und gibt die aktuelle Smarty-Template-Seite aus.

`serendipity_printEntryFooter()`

(`include/functions_entries.inc.php`) Setzt abhängig von den auf der aktuellen Seite dargestellten Blog-Artikeln die notwendigen Footer-Elemente (Anzahl der Artikel, Anzahl der Seiten, aktuelle Seite) zusammen.

`serendipity_loadthemeOptions()`

(`include/functions_config.inc.php`) Lädt die Template-Konfigurationsoptionen, die dem aktuellen Template zugeordnet wurden.

10.5.8 Datenbank

Funktionen zur Abfrage und Verwaltung der Datenbank.

`serendipity_db_update()` (`include/db.inc.php`)
: Führt eine Datenbankabfrage aus, die einen Datensatz aktualisiert (`UPDATE table SET ...`). Dabei wird die Liste der zu aktualisierenden Feldnamen und Werte der Funktion übergeben und automatisch in das benötigte SQL-Format übersetzt.

`serendipity_db_insert()` (`include/db.inc.php`)
: Führt eine Datenbankabfrage aus, die einen Datensatz einfügt (`INSERT INTO table (...) VALUES (...)`). Dabei wird die Liste der einzufügenden Feldnamen und Werte der Funktion übergeben und automatisch in das benötigte SQL-Format übersetzt.

`serendipity_db_bool()` (`include/db.inc.php`)
: Prüft, ob ein der Funktion übergebener Wert einem *Boolean*-Wert (`true/false`) entspricht.

`serendipity_db_get_interval()` (`include/db.inc.php`)
: Liefert abhängig vom eingesetzten Datenbanksystem ein SQL-Codefragment zurück, mit dem anhand eines Datenbankfeldes eine Einschränkung des Zeitraums der Datensätze vorgenommen wird.

`serendipity_db_implode()` (`include/db.inc.php`)
: Liefert ein SQL-Codefragment, damit ein PHP-Array mit Werten sicher innerhalb einer Datenbankabfrage verwendet werden kann.

`serendipity_db_begin_transaction()`,

`serendipity_db_end_transaction()`

(z. B. `include/mysql.inc.php`) Führt eine Datenbankabfrage zum Anfang/Ende einer Transaktion aus.

`serendipity_db_query()` (z. B. `include/mysql.inc.php`)
: Führt eine Datenbankabfrage mittels einer SQL-Abfrage aus und liefert die Ergebnisse zurück.

`serendipity_db_insert_id()` (z. B. `include/mysql.inc.php`)
: Liefert den Primärschlüssel der zuletzt mittels `INSERT INTO...` eingefügten Datenbankzeile.

`serendipity_db_affected_rows()`,

`serendipity_db_matched_rows()`

(z. B. `include/mysql.inc.php`) Liefert die Anzahl der von der letzten SQL-Abfrage betroffenen Ergebniszeilen (`SELECT`, `DELETE` ...) zurück.

`serendipity_db_updated_rows()` (z. B. `include/mysql.inc.php`)
Liefert die Anzahl der von der letzten SQL-Abfrage aktualisierten Datensätze (`UPDATE`) zurück.

`serendipity_db_escape_string()` (z. B. `include/mysql.inc.php`)
Liefert den Wert einer übergebenen Variable in einer Formatierung zurück, die eine gefahrlose Verwendung innerhalb einer SQL-Abfrage ermöglicht. Etwaige Sonderzeichen werden dabei `escaped` (umformatiert), so dass kein Missbrauch stattfinden kann. Diese Funktion muss immer auf alle Variablen angewendet werden, die der Benutzer in Formularen oder per URL-Wert einbringt.

`serendipity_db_limit()` (z. B. `include/mysql.inc.php`)
Liefert den für das jeweilige Datenbanksystem benötigten SQL-Code zurück, um die Anzahl der Ergebnisse einer `SELECT`-Abfrage einzuschränken. Dieser SQL-Code weicht in den verschiedenen SQL-Dialekten fundamental voneinander ab.

`serendipity_db_concat()` (z. B. `include/mysql.inc.php`)
Liefert den für das jeweilige Datenbanksystem benötigten SQL-Code zurück, um mehrere Datenbankspalten miteinander in einer Ausgabe zu verbinden (`CONCAT`).

`serendipity_db_connect(), serendipity_db_reconnect()`

(z. B. `include/mysql.inc.php`) Stellt eine Verbindung zur Datenbank her.

`serendipity_db_probe()` (z. B. `include/mysql.inc.php`)
Testet die Verbindung zur Datenbank.

`serendipity_db_schema_import()` (z. B. `include/mysql.inc.php`)
Importiert eine SQL-Anweisung, die aus einem Plugin oder einer Datei des `sql`-Verzeichnisses von Serendipity stammt. Dabei werden etwaige Platzhalter korrekt in das für das jeweilige Datenbanksystem benötigte SQL-Format gewandelt.

10.6 Datenbank

Serendipity speichert seine Daten in verschiedenen Datenbanktabellen. Viele Tabellen sind miteinander verknüpft, meist anhand von 1:n-Primärschlüsseln oder weiteren unabhängigen n:m-Verbindungen.

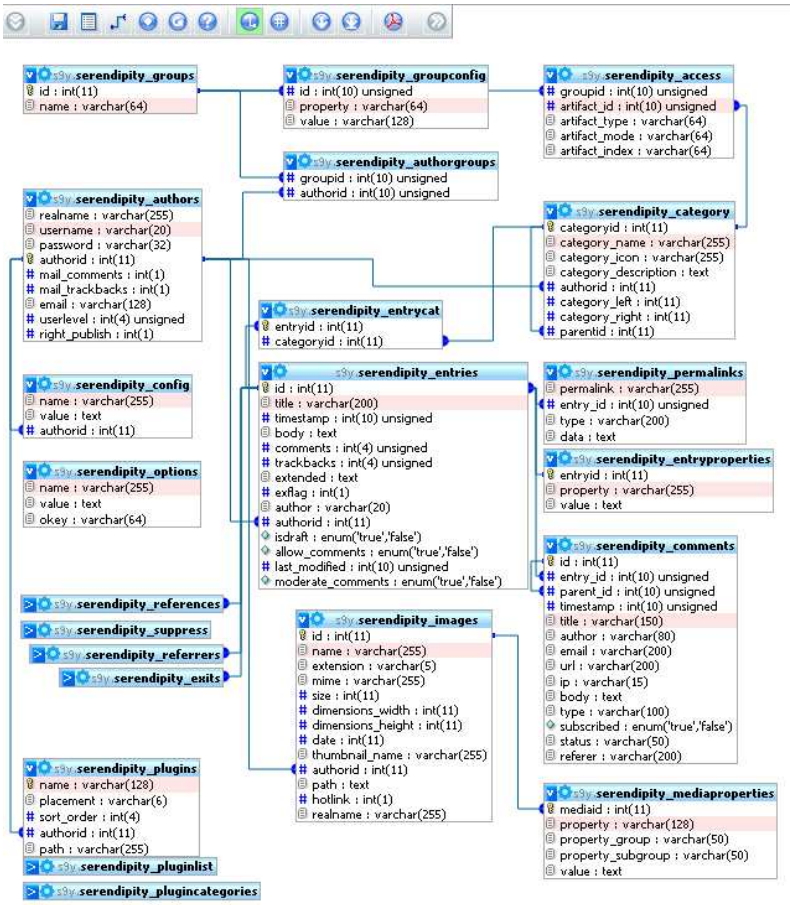

Abbildung 10.1: Datenbank ER-Modell

Dabei benutzt Serendipity keine *Foreign Key*-Assoziierung, die PostgreSQL oder MySQL zu bieten haben. Der Grund dafür ist, dass die Serendipity-Tabellen möglichst ohne viel Portierungsaufwand auch auf anderen Datenbanksystemen wie SQLite einsetzbar sein sollen. Jeglicher Einsatz von Fremdschlüsseln wird durch den Serendipity-Code verwaltet und nicht die Datenbank. Abbildung 10.1 zeigt einen Screenshot des Datenbank-Relationsschemas.

10.6.1 Benutzer- und Rechtemanagement

Die folgenden Tabellen dienen der Verwaltung von Benutzern, Gruppen und deren Rechten. Sie werden in den SQL-Datenbankabfragen mit `JOIN`s einbezogen.

serendipity_authors

Enthält die Redakteure des Blogs.

`authorid` (Primärschlüssel, int(11))
: enthält die fortlaufende ID eines Redakteurs.

`realname` (varchar(255))
: enthält den dargestellten Namen des Redakteurs.

`username` (varchar(32))
: enthält den Loginnamen des Redakteurs.

`password` (varchar(32))
: enthält das Passwort des Redakteurs als MD5-Prüfsumme.

`email` (varchar(128))
: enthält die E-Mail-Adresse des Redakteurs.

`mail_comments` (int(1))
: gibt an, ob der Redakteur bei Eingang neuer Kommentare eine E-Mail erhält.

`mail_trackbacks` (int(1))
: gibt an, ob der Redakteur bei Eingang neuer Trackbacks eine E-Mail erhält.

`userlevel` (int(4))
: enthält den Benutzerrang eines Redakteurs. Der numerische Wert 255 steht für Administratoren, 1 für Chefredakteure und 0 für normale Redakteure. Der Benutzerrang ist seit Einführung der Benutzergruppen nur noch für ältere Plugins von Bedeutung.

`right_publish` (int(1))
: gibt an, ob ein Redakteur Einträge veröffentlichen darf.

serendipity_groups

Enthält die Benutzergruppen des Blogs.

`id` (Primärschlüssel, int(11))
: enthält die fortlaufende ID einer Gruppe.

`name` (varchar(64))
: enthält den Namen der Redakteursgruppe.

serendipity_authorgroups

Enthält eine n:m-Zuordnungstabelle, mit der Autoren den Benutzergruppen zugeordnet werden.

`groupid` (Fremdschlüssel, int(10))
: enthält die ID der Benutzergruppe.

`authorid` (Fremdschlüssel, int(10))
: enthält die ID des Redakteurs.

serendipity_groupconfig

Enthält eine Zuordnung von Rechten zu Benutzergruppen. Eine Benutzergruppe kann beliebig viele und beliebig benannte Rechte haben.

`id` (Fremdschlüssel, int(10))
: enthält die ID der Benutzergruppe.

`property` (varchar(128))
: enthält den Namen des Rechts.

`value` (varchar(64))
: enthält den Wert des Rechts, d. h. ob dieses gesetzt (`true`) oder nicht gesetzt (`false`) ist. Auch andere Werte als `true` oder `false` können von Plugins gesetzt werden.

serendipity_access

Enthält die ACL (*Access Control List*), anhand derer bestimmten Benutzergruppen Zugriff auf Teile der Serendipity-Inhalte zugewiesen werden kann. Die Tabelle ist so abstrahiert, dass beliebige Rechte eingerichtet werden können. Für jedes Tupel `groupid, artifact_id` können mehrere unterschiedliche `artifact_mode`-Werte hinterlegt werden.

`groupid` (Fremdschlüssel, int(10))
: enthält die ID einer Benutzergruppe.

`artifact_id` (Fremdschlüssel, int(10))
: enthält die ID des Fremdobjekts, für das ein Recht vergeben wird.

`artifact_type` (varchar(64))
: legt fest, für welchen Objekttyp ein Recht vergeben wird. Mögliche Werte sind: `category` (für Kategorien), `directory` (für Verzeichnisse der Mediendatenbank).

`artifact_mode` (varchar(64))
: legt fest, welche Eigenschaft ein Recht hat. Mögliche Werte sind: `read` (Leserecht), `write` (Schreibrecht).

`artifact_index` (varchar(64))
: legt ein zusätzliches Attribut für ein Recht fest, falls ein Fremdobjekt nicht direkt mittels `artifact_id` zugeordnet werden kann. Beispielsweise enthält `artifact_index` den Namen des Verzeichnisses der Mediendatenbank, für das ein Recht vergeben werden soll, da die vorhandenen Verzeichnisse nicht in der Datenbank verfolgt werden.

10.6.2 Mediendatenbank

Die Objekte der Mediendatenbank dienen lediglich als Container für Metadaten. Die eigentlichen Dateien liegen weiterhin im Dateisystem.

serendipity_images

`id` (Primärschlüssel, int(11))
: enthält die fortlaufende ID eines Medienobjekts.

`name` (varchar(255))
: enthält den Dateinamen (ohne Endung).

`extension` (varchar(5))
: enthält die Dateiendung.

`mime` (varchar(255))
: enthält den MIME-Typen für das Medienobjekt.

`size` (int(11))
: enthält die Dateigröße (in Bytes).

`dimensions_width` (int(11))
: enthält die Bildbreite.

`dimensions_height` (int(11))
: enthält die Bildhöhe.

`date` (int(11))
: enthält das Hochladedatum (in UNIX-Sekunden).

`thumbnail_name` (varchar(255))
: enthält das Suffix für die Vorschaubilder.

`authorid` (Fremdschlüssel, int(11))
: verweist auf den Eigentümer der Datei.

`path` (text)
: enthält den Namen des Verzeichnisses, in dem das Bild gespeichert ist.

`hotlink` (int(1))
: gibt an, ob das Bild von einem fremden Server geladen wird.

`realname` (varchar(255))
: enthält den ursprünglichen Namen einer Datei, wenn diese zur Vermeidung doppelter Dateinamen automatisch umbenannt wurde.

serendipity_mediaproperties

Zu jedem Objekt der Mediendatenbank können beliebig viele Metadaten gespeichert werden. Diese Metadaten können in verschiedene Unterkategorien aufgeteilt werden, um zwischen frei vergebenen Metadaten und solchen, die in der Datei festgelegt wurden, zu unterscheiden.

`mediaid` (Fremdschlüssel, int(11))
: verweist auf die ID des Medienobjekts.

`property` (int(11))
: enthält den Bezeichner einer Medieneigenschaft.

`property_group` (int(11))
: enthält die Zuordnung in eine Metadaten-Gruppe. Mögliche Werte: `base_keyword` (freie Schlüsselwörter), `base_property` (freie Metafelder), `base_metadata` (Metadaten der Datei).

`property_subgroup` (int(11))
: kann eine Untergruppierung der Metadaten enthalten, vor allem im Falle von `base_metadata`-Gruppen. Werte wie `EXIF`, `XMP` oder `ID3` geben die Quelle der Eigenschaft an.

`value` (int(11))
: enthält den zugeordneten Wert einer Medieneigenschaft.

10.6.3 Artikel, Kategorien, Kommentare

Die folgenden sechs Datenbanktabellen enthalten den Kern der redaktionellen Inhalte Ihres Blogs: Artikel, Kommentare und Kategorien.

serendipity_entries

Enthält die Blog-Artikel.

`id` (Primärschlüssel, int(11))
: enthält die ID des Artikels.

`title` (varchar(200))
: enthält den Titel des Artikels.

`timestamp` (int(10))
: enthält die Erstellungszeit des Artikels.

`last_modified` (int(10))
: enthält das Datum der letzten Aktualisierung.

`body` (text)
: enthält den Artikeltext.

`extended` (text)
: enthält den erweiterten Artikeltext.

`comments` (int(4))
: enthält die Anzahl an Kommentaren zu diesem Artikel.

`trackbacks` (int(4))
: enthält die Anzahl an Trackbacks zu diesem Artikel.

`exflag` (int(1))
: gibt an, ob der Artikel einen erweiterten Artikeltext besitzt.

`author` (varchar(20))
: enthält den Namen des erstellenden Redakteurs.

`authorid` (int(11))
: enthält die ID des erstellenden Redakteurs.

`isdraft` (bool)
: gibt an, ob der Eintrag ein Entwurf ist.

`allow_comments` (bool)
: gibt an, ob Kommentare zu diesem Artikel erlaubt sind.

`moderate_comments` (bool)
: gibt an, ob Kommentare zu diesem Artikel moderiert werden.

serendipity_entryproperties

Enthält eine Reihe an zusätzlichen Eigenschaften zu einem Artikel (n:m-Zuordnung).

`entryid` (Fremdschlüssel, int(11))
: enthält die zugeordnete Artikel-ID

`property` (varchar(255))
: enthält den Namen der zusätzlichen Eigenschaft.

`value` (text)
: enthält den Wert der Eigenschaft, z. B. HTML-Text oder Dateinamen der Mediendatenbank.

serendipity_category

Enthält die im Blog erstellten Kategorien. Die Kategorien sind unendlich ineinander verschachtelbar, da sie der Nested-Set-Datenbankstruktur folgen.

`categoryid` (Primärschlüssel, int(11))
: enthält die fortlaufende ID einer Kategorie.

`category_name` (varchar(255))
: enthält den Namen einer Kategorie.

`category_icon` (varchar(255))
: enthält ein optionales Symbolbild, das Einträgen dieser Kategorie zugewiesen werden kann.

`category_description` (text)
: enthält die Beschreibung einer Kategorie.

`authorid` (Fremdschlüssel, int(11))
: enthält die ID des Redakteurs, der diese Kategorie erstellt hat.

`category_left` (Fremdschlüssel, int(11))
: enthält einen Verweis auf die *linke* Kategorie in der Nested-Set-Hierarchie.

`category_right` (Fremdschlüssel, int(11))
: enthält einen Verweis auf die *rechte* Kategorie in der Nested-Set-Hierarchie.

`parentid` (Fremdschlüssel, int(11))
: enthält einen Verweis auf die übergeordnete Kategorie.

serendipity_entrycat

Weist Einträge bestehenden Kategorien mittels n:m-Verknüpfung hinzu. Ein Eintrag kann somit mehreren Kategorien zugeordnet werden.

`entryid` (Fremdschlüssel, int(11))
: enthält die ID des Artikels.

`categoryid` (Fremdschlüssel, int(11))
: enthält die ID einer Kategorie.

serendipity_comments

Enthält die Kommentare und Trackbacks zu Artikeln.

`id` (Primärschlüssel, int(11))
: enthält die ID eines Kommentars.

`entry_id` (Fremdschlüssel, int(10))
: enthält die ID des Artikels, zu dem der Kommentar gehört.

`parent_id` (Fremdschlüssel, int(10))
: enthält die ID des übergeordneten Kommentars, falls sich der aktuelle Kommentar auf einen anderen beziehen soll.

`timestamp` (int(10))
: enthält das Datum, an dem der Kommentar gespeichert wurde.

`title` (varchar(150))
: enthält den Titel des Kommentars.

`author` (varchar(80))
: enthält den Namen des Kommentators.

`email` (varchar(200))
: enthält die E-Mail-Adresse des Kommentators.

`url` (varchar(200))
: enthält die Homepage-Adresse des Kommentators.

`ip` (varchar(15))
: enthält die IP-Adresse des Kommentators.

`body` (text)
: enthält den Kommentartext.

`type` (varchar(100))
: gibt an, ob der Eintrag ein Kommentar (`NORMAL`), Trackback (`TRACKBACK`) oder Pingback (`PINGBACK`) ist.

`subscribed` (bool)
: gibt an, ob der Kommentator über weitere Kommentare zum Artikel per E-Mail benachrichtigt werden soll.

`status` (varchar(50))
: gibt den Status eines Kommentars an: `approved` (veröffentlicht) oder `pending` (in Moderation).

`referer` (referer(200))
: enthält die URL, die der Kommentator vor dem Aufruf Ihres Blogs besucht hat.

serendipity_references

Alle URLs, die ein Redakteur in einem Artikel einbindet, werden in dieser Datenbanktabelle gespeichert. Solche Referenzen können durch Plugins dargestellt oder ausgewertet werden. Des Weiteren enthält diese Tabelle auch alle Referenzen von aufgerufenen Bildern Ihrer Mediendatenbank (siehe Seite 687).

`id` (Primärschlüssel, int(11))
: enthält die fortlaufende ID einer referenzierten URL.

`entry_id` (Fremdschlüssel, int(10))
: enthält die ID des Artikels, zu dem die referenzierte URL gehört.

`link` (text)
: enthält die vollständige referenzierte URL.

`name` (text)
: enthält den beschreibenden Text zu der referenzierten URL.

`type` (varchar(128))
: legt den Typ der Referenz fest. `media` steht für verwiesene Mediendatenbank-Objekte, ein leerer Typ steht für gewöhnlich referenzierte URLs.

10.6.4 Zentraltabellen

Weiterhin greift Serendipity auf eine Menge weiterer Tabellen zu, die nicht den vorigen Gruppierungen zuzuordnen sind.

serendipity_config

Enthält sämtliche Konfigurationsoptionen des Blogs, von Plugins und persönlichen Einstellungen der Redakteure.

`name` (varchar(255))
: enthält den Namen einer Konfigurationsoption. Plugin-Optionen werden mit dem Namen des Plugins als Präfix gespeichert.

`value` (text)
: enthält den Wert einer Konfigurationsoption.

`authorid` (Fremdschlüssel, int(11))
: enthält die ID eines Redakteurs, falls die Konfigurationsoption einer persönlichen Einstellung entspricht.

serendipity_permalinks

Enthält eine Look-Up-Tabelle von Permalinks. Serendipity kann die aufgerufene URL mit einem der in dieser Tabelle hinterlegten Permalinks vergleichen, um dann die entsprechenden Inhalte darzustellen. Dabei unterstützt Serendipity Permalinks für Einträge, Kategorien, Autoren und RSS-Feeds.

`permalink` (varchar(255))
: enthält die URL des Permalinks.

`entry_id` (varchar(255))
: enthält die ID des Zielinhalts (Kategorie-ID, Autor-ID, Artikel-ID).

`type` (varchar(255))
: legt den Typ des Permalinks (`category`, `author`, `entry`) fest.

`data` (varchar(255))
: kann zusätzliche Werte enthalten, derzeit noch nicht genutzt.

serendipity_plugincategories

Enthält eine Zuordnung der Plugins zu ihren jeweiligen Gruppen. Diese Tabelle wird automatisch gefüllt, sie kann gefahrlos geleert werden, um eine Neu-Initialisierung zu erzwingen.

`class_name` (varchar(250))
: enthält den Namen eines Plugins.

`category` (varchar(250))
: enthält den Namen einer Kategorie, die dem Plugin zugeordnet ist.

serendipity_pluginlist

Enthält Metadaten zu den verfügbaren, installierbaren Plugins. Diese Tabelle wird automatisch gefüllt, sie kann gefahrlos geleert werden, um eine Neu-Initialisierung zu erzwingen. Der Inhalt dieser Tabelle gilt als Zwischenspeicher, damit die performance-intensive Plugin-Abfrage nur bei Bedarf ausgeführt werden muss.

`plugin_file` (varchar(255))
: enthält den Basisnamen eines Plugins.

`class_name` (varchar(255))
: enthält den PHP-Klassennamen eines Plugins.

`plugin_class` (varchar(255))
: enthält den Serendipity-internen Klassennamen eines Plugins, bei dem interne Plugins mit einem @-Zeichen gekennzeichnet werden.

`pluginPath` (varchar(255))
: enthält den Pfad zum Plugin.

`name` (varchar(255))
: enthält den Namen des Plugins.

`description` (text)
: enthält die Beschreibung eines Plugins.

`version` (varchar(12))
: enthält die Versionsnummer eines Plugins.

`upgrade_version` (varchar(12))
: enthält die aktuell online verfügbare Version des Plugins.

`plugintype` (varchar(255))
: legt den Typ eines Plugins fest (`sidebar` für Seitenleisten-Plugins, `event` für Ereignis-Plugins).

`pluginlocation` (varchar(255))
: gibt an, wo das Plugin heruntergeladen werden kann (`local` oder `Spartacus`).

`stackable` (int(1))
: gibt an, ob ein Plugin mehrfach installiert werden kann.

`author` (varchar(255))
: gibt den Namen des Plugin-Autors an.

`requirements` (text)
: gibt etwaige Rahmenbedingungen für das Plugin an.

`website` (varchar(255))
: gibt eine Homepage des Plugin-Autors an.

`last_modified` (int(11))
: gibt an, ob und wann das Plugin zuletzt auf Ihrem Server aktualisiert wurde.

serendipity_plugins

Enthält eine Liste der installierten und aktivierten Plugins im Blog.

`name` (Primärschlüssel, varchar(128))
: enthält den Namen des Plugins inklusive der zufällig zugeteilten ID.

`placement` (varchar(6))
: legt fest, wo das Plugin platziert ist. Mögliche Werte: `event` (Ereignis-Plugin), `eventh` (deaktiviertes Ereignis-Plugin), `left` (Plugin in linker Seitenleiste), `right` (Plugin in rechter Seitenleiste), `hidden` (verstecktes Seitenleisten-Plugin) sowie die Namen aller selbständig festgelegten Seitenleisten.

`sort_order` (int(4))
: gibt einen numerischen Wert zur Sortierungsreihenfolge der Plugins untereinander an.

`authorid` (int(11))
: enthält die ID des Redakteurs, der das Plugin installiert hat.

`path` (varchar(255))
: enthält den Verzeichnispfad zum jeweiligen Plugin.

serendipity_exits, serendipity_referrers

`serendipity_exits` enthält bei aktiviertem Exit-Tracking (siehe Seite 258) Statistiken über Links in Ihren Blog-Artikeln, auf die Besucher geklickt haben. `serendipity_referrers` enthält bei aktiviertem Referrer-Tracking (siehe Seite 168) die Statistiken über Webseiten, von denen aus Besucher auf Ihr Blog gelangen. Beide Tabellen werden über den Umweg der Tabelle `serendipity_suppress` gefüllt.

`entry_id` (Primärschlüssel, int(11))
: enthält die ID des zugehörigen Blog-Artikels, von dem aus ein Exit-Link geklickt wurde bzw. auf dessen URL von einem fremden Blog verwiesen wurde.

`day` (date)
: enthält das Datum, an dem der Tabelleneintrag erstellt wurde.

`count` (int(11))
: enthält die Anzahl für den jeweiligen statistischen Wert.

serendipity_suppress

Diese Tabelle dient der Befüllung der Tabellen `serendipity_exits` und `serendipity_referrers`. Eine verweisende URL oder die in einem Artikel eingebundene URL wird beim Aufruf durch einen Besucher erst in dieser Tabelle zwischengespeichert. Erst sobald der jeweilige Link mehr als einmal aufgerufen/übermittelt wurde, wird der Besuch auch in der zugehörigen Tabelle `serendipity_exits` oder `serendipity_referrers` gespeichert. Dies kann in geringem Maß Spam erschweren, aber leider mittlerweile auch nicht mehr verhindern.

`ip` (varchar(15))
: IP des Besuchers.

`scheme` (varchar(5))
: enthält den Teil der URL des Tabelleneintrags, der das Protokoll angibt (z. B. `http://`).

`host` (int(128))
: enthält den Servernamen einer URL.

`port` (int(5))
: enthält den Port einer URL.

`path` (int(255))
: enthält den Verzeichnisteil einer URL.

`query` (int(255))
: enthält zusätzliche GET-Parameter einer URL.

`last` (timestamp)
: enthält das Datum des letzten Besuchs der URL.

serendipity_spamblocklog

Diese Tabelle enthält Protokollmeldungen des Antispam-Plugins (siehe Seite 239), beispielsweise Hinweise zu abgewiesenen oder moderierten Kommentaren und Fehlermeldungen. Mit der Zeit kann diese Tabelle recht groß werden, da Serendipity sie nicht selbständig leert. Sie können alle Einträge gefahrlos löschen, wenn Sie diese nicht weiter benötigen (siehe Seite 442). Selbstverständlich können Sie in den Einstellungen des Anti-Spam-Plugins die Protokollierung auch vollständig deaktivieren.

Ein Protokolleintrag wird vom Anti-Spam-Plugin nur dann erstellt, wenn ein neuer Kommentar zu einem Blog-Artikel verfasst wurde. Daher ist ein Protokolleintrag stets mit einem Blog-Artikel und einem Kommentar verbunden.

`timestamp` (int(10))
: enthält das Datum, an dem der Protokolleintrag erstellt wurde.

`type` (varchar(255))
: enthält eine Klassifizierung des Protokolleintrags.

`reason` (text)
: enthält den ausführlichen Protokolltext.

`entry_id` (int(10))
: enthält die ID des Artikels, auf den sich der Protokolleintrag bezieht.

`author` (varchar(80))
: enthält den Namen des Kommentarautors.

`email` (varchar(200))
: enthält die E-Mail-Adresse des Kommentarautors.

`url` (varchar(200))
: enthält die Homepage des Kommentarautors.

`useragent` (varchar(255))
: enthält den verwendeten Browser des Kommentarautors.

`ip` (varchar(15))
: enthält die IP-Adresse des Kommentarautors.

`referer` (varchar(255))
: enthält die URL der vorher vom Kommentarautor besuchten Webseite.

`body` (text)
: enthält den Kommentartext.

10.7 Sourcecode-Verwaltung

Serendipity ist letztlich nur eine Ansammlung von PHP-Dateien. In allen OpenSource-Projekten ist es üblich, den Quellcode für Interessierte vollständig offen und transparent zu verwalten.

Dazu hat sich seit langer Zeit der Service von SourceForge[1] als Standard etabliert, der OpenSource-Projekten kostenlos Verwaltungsdienste und Speicherplatz anbietet.

Die Projektseite von Serendipity befindet sich unter `http://www.sourceforge.net/projects/php-blog`. Dort gibt es einige Mailinglisten, Bug-Tracker und vor allem den Download der offiziellen Serendipity-Releases.

Bis vor einigen Jahren wurde auch der Quellcode von Serendipity dort mittels einer Software namens CVS[2] verwaltet. CVS ist eine Software, die auf einem Server läuft, um dort mehreren Entwicklern den direkten Zugriff auf den Quellcode zu ermöglichen. Die Entwickler können dort alle Dateien auf ihren lokalen Entwicklungsrechner herunterladen, ihre Änderungen vornehmen und die Dateien dann wieder auf den Server (in das sogenannte *Repository*, das Archiv) hochladen.

Gewissermaßen kann man CVS mit FTP vergleichen, nur dass CVS weitaus komplexere Upload- und Downloadmethoden unterstützt. Bei jeder Änderung am Quellcode (*Commit*, eine Übertragung) protokolliert der CVS-Server dies und speichert es als neue *Revision*. So können jederzeit einfach Änderungen rückgängig gemacht oder Vergleiche zwischen verschiedenen Versionsständen angestellt werden. Auch ist durch dieses Vorgehen sichergestellt, dass zwei Entwickler nicht versehentlich die Änderungen einer anderen Person unwiderruflich löschen – CVS dient also gleichzeitig als Backup.

[1] `http://www.sourceforge.net/`
[2] `http://de.wikipedia.org/wiki/Concurrent_Versions_System`

Entwickler können zu einem gewünschten Zeitpunkt sogenannte *Branches* des Quellcodes erstellen. Dabei wird eine Kopie des aktuellen Quellcodes angelegt, neue Änderungen am Code werden in dieser Kopie vorgenommen. Dadurch kann eine Software bereits neue, experimentelle Features in Version 2.0 einbauen, während die stabile Versionsnummer 1.0 parallel weiterentwickelt wird. CVS erlaubt es auch, zwischen diesen verschiedenen *Verästelungen* Codevergleiche durchzuführen. Auch Serendipity nutzt dieses Prinzip, bei dem lediglich Bugfixes in einer stabilen Versionsnummer eingefügt werden. Nur die jeweils neueste Beta-Versionslinie (*trunk*, engl. Baumstamm) enthält möglicherweise noch nicht vollständig getestete Features.

Um CVS nutzen zu können, benötigt man eine Software wie `TortoiseCVS` oder den Kommandozeilen-Client `cvs`. Erst über diese Programme kann man die Fähigkeiten von CVS nutzen, ähnlich wie bei Verwendung eines FTP-Programms.

Vor einigen Jahren wurde der Nachfolger von CVS verfügbar: *Subversion*, kurz SVN. SVN ähnelt CVS in sehr vielen Bereichen, ist aber leistungsfähiger, schneller und erlaubt einige sehr sinnvolle Verwaltungsoperationen wie z. B. vollständig symbolisch erzeugte *Branches*.

Damals bot SourceForge noch keinen SVN-Zugriff an, daher migrierten die Entwickler von Serendipity auf einen anderen Dienstleister namens *BerliOS*[3]. Bis heute wird dort das Serendipity-Sourcecode-Repository mittels SVN gepflegt.

Dennoch ist das alte CVS-Repository bei SourceForge noch vorhanden. Dort werden separat die erweiterten Plugins (*Spartacus*) gepflegt. Die Aufteilung in SVN und CVS hat somit noch den Vorteil, dass das verbreitetere (und einfachere) CVS so für freiwillige Plugin-Entwickler leichter verfügbar ist.

10.7.1 Freier Zugriff

Die SVN- und CVS-Quellen sind für Besucher mit vollem Lesezugriff frei zugänglich. Um darauf zuzugreifen, benötigen Sie eine CVS- oder SVN-Software. Weitere Anweisungen, wie Sie auf ein *Repository* zugreifen können, sind auf den Projektseiten `http://sourceforge.net/cvs/?group_id=75065` (CVS) und `http://developer.berlios.de/svn/?group_id=2573` (SVN) ausführlich dokumentiert.

Schreibzugriff können Sie erlangen, indem Sie im Serendipity-Entwicklerforum danach fragen und einige Beispiele dafür geben, was Sie beisteuern wollen. Wie dieser Prozess vonstatten geht, ist auf Seite 695 näher erläutert.

Wenn Sie über kein CVS-Programm verfügen, können Sie auch über webbasierte CVS-Clients Einblick in das *Repository* erhalten. Den Quellcode von Serendipity finden Sie unter `http://svn.berlios.de/viewcvs/serendipity/trunk/` und alle Spartacus-Plugins unter `http://php-blog.cvs.sourceforge.net/php-blog/additional_plugins/`. Von dort aus können Sie sogar einzelne Dateien direkt herunterladen oder neue Dateiversionen miteinander vergleichen.

[3] `http://developer.berlios.de`

10.7.2 Aktualisierung über CVS/SVN

CVS ist nicht nur für Entwickler von Interesse. Wenn Sie SSH-Zugriff zu Ihrem Server besitzen, kann CVS sehr hilfreich dabei sein, Serendipity auf dem aktuellsten Stand zu halten.

Anstatt erst ein ZIP-Archiv mit der neuen Serendipity-Version herunterzuladen, könnten Sie Ihren Server auch direkt über einen *CVS/SVN checkout* laufen lassen. Sobald Sie Serendipity z. B. mittels

```
$ cd /home/www/example.com/
$ svn checkout svn://svn.berlios.de/serendipity/trunk
```

auf Ihren Server geladen haben, reicht später ein einfaches

```
$ cd /home/www/example.com/
$ svn update
```

um die aktuellste Version der Serendipity-Dateien zu erhalten. Dabei hat CVS/SVN einen zentralen Vorteil: Falls Sie jemals manuell eine Datei editiert haben, kann der CVS/SVN-Client dafür sorgen, dass Ihre Änderungen nicht überschrieben werden. Bis zu gewissen Grenzen kann CVS/SVN automatisch dafür sorgen, dass Ihre Änderungen mit den offiziellen Änderungen an einer Datei verbunden werden. Nur wenn sich eine Datei grundlegend ändert, wird CVS/SVN auf einen Konflikt hinweisen.

Im gleichen Zug ermöglicht es Ihnen CVS/SVN, Ihre eigenen Änderungen an Serendipity mittels

```
$ cd /home/www/example.com/
$ svn diff
```

in einem maschinenlesbaren Format auszuwerten. Wenn Sie beispielsweise sinnvollen Code zur Community beisteuern wollen, hilft dieses *DIFF-Format*, Ihre Änderungen allen zur Verfügung zu stellen.

Auch die Spartacus-Plugins können Sie komfortabel über einen *CVS checkout* verwalten. Dabei kommt Ihnen zugute, dass Serendipity beliebig verschachtelte Verzeichnisstrukturen des `plugins`-Verzeichnisses unterstützt. Einen Checkout des CVS-Moduls `additional_plugins` können Sie wie folgt erreichen:

```
$ cd /home/www/example.com/plugins
$ cvs -d:pserver:anonymous@php-blog.cvs.sf.net:/cvsroot/php-blog login
$ cvs -d:pserver:anonymous@php-blog.cvs.sf.net:/cvsroot/php-blog
 checkout additional_plugins
```

Daraufhin enthält das Verzeichnis `/plugins/additional_plugins` alle Zusatz-Plugins von Serendipity. Danach können Sie jedes verfügbare Plugin direkt aktivieren, ohne es herunterzuladen. Regelmäßige Updates an Plugins können dann ebenfalls per `cvs update` eingespielt werden.

Dasselbe Vorgehen gilt neben `plugins` übrigens auch für `templates`:

```
$ cd /home/www/example.com/templates
$ cvs -d:pserver:anonymous@php-blog.cvs.sf.net:/cvsroot/php-blog login
$ cvs -d:pserver:anonymous@php-blog.cvs.sf.net:/cvsroot/php-blog
 checkout additional_themes
```

10.7.3 Spartacus

Spartacus ist Serendipitys zentrales Plugin-Archiv. Es besteht aus zwei Komponenten: einem Server und einem Client. Der Server (SourceForge) bietet die Plugins an, der Client (das Spartacus-Plugin Ihres Blogs) lädt ein Plugin herunter.

Im Hintergrund passiert Folgendes: Auf dem SourceForge-Server wird das CVS-Repository `additional_plugins` verwaltet. Dort werden regelmäßig einmal am Tag durch einen automatisierten Vorgang[4] zahlreiche XML-Dateien erstellt, die alle notwendigen Informationen der Plugins enthalten.

Daraufhin werden anhand dieser XML-Informationen automatisch HTML-Dateien erzeugt, die auf `http://spartacus.s9y.org` dargestellt werden. Jedes Plugin wird zudem als ZIP-Archiv komprimiert und ebenfalls auf dem Webserver zur Verfügung gestellt.

Die Sammlung an XML-Dateien sowie der Quellcode der Plugins werden zusätzlich automatisch an den Mirror `netmirror.org` ausgeliefert. Sollte einmal ein Server nicht erreichbar sein, gibt es somit immer noch einen weiteren Notfallserver.

Nun ist der serverseitige Vorgang abgeschlossen, und der Client (Ihr Blog) muss lediglich darauf zugreifen. Beim Aufruf der Plugin-Verwaltungsoberfläche Ihres Blogs lädt das Plugin automatisch vom konfigurierten Server die XML-Datei herunter. Die Liste von Ereignis-Plugins befindet sich in der Datei `package_event_de.xml`, die Seitenleisten-Plugins in der Datei `package_sidebar_de.xml` und Templates in `package_template.xml`. Plugin-Beschreibungen sind sprachabhängig, das Kürzel am Ende des Dateinamens legt dabei die Sprache fest.

Diese XML-Dateien werden vom Plugin in Ihr Verzeichnis `templates_c` heruntergeladen und von dort aus gecached. Erst wenn Ihre temporären Dateien älter als einen Tag sind, werden sie vom Plugin erneut geladen. Dies verringert den anfallenden Traffic deutlich.

Die XML-Dateien werden vom Plugin ausgewertet und in der Oberfläche dargestellt. Wenn Sie nun eines dieser Spartacus-Plugins installieren wollen, lädt das Plugin die Rohdateien direkt vom konfigurierten Mirror-Server herunter und speichert sie lokal auf Ihrem Server.

Sobald die Dateien lokal vorhanden sind, werden sie von Serendipity wie übliche Plugins behandelt. Einmal via Spartacus heruntergeladene Dateien können dann in Zukunft jeweils mit der aktuellsten Version des Plugins verglichen und gegebenenfalls neu heruntergeladen werden.

[4] Für Interessierte: Das automatische Script zur Erstellung ist in diesem Repository als Datei `emerge_spartacus.php` hinterlegt.

10.8 Plugin-API

Die Serendipity Plugin-API stellt das Herz von Serendipity dar. Als API bezeichnet man eine Sammlung an Funktionen oder Objekten/Klassen, die standardisierten Zugriff auf ein System zulassen.

Serendipity abstrahiert mit seiner Plugin-API beinahe alle Vorgänge im Kernsystem. Plugins haben an vielen (und leicht erweiterbaren) Stellen im Arbeitsablauf des Frontends und Backends die Möglichkeit, beliebig einzugreifen und PHP-Code auszuführen.

Auf PHP-Seite ist diese Plugin-API mittels objektorientierter Techniken (*OO*) umgesetzt. Objektorientierte Programmierung hat den Vorteil, dass sie relativ leicht dokumentiert werden kann und gerade für Rohgerüste durchschaubarer ist als eine einfache Ansammlung von Funktionen. Objektvererbung und Kapselung erhöhen zudem die Code-Qualität und führen zu weniger Redundanz. Für Programmierer ist es wichtig, zwischen einem Objekt und einer Klasse zu unterscheiden: Klassen legen lediglich eine Struktur fest (ähnlich wie ein Bauplan). Objekte sind *umgesetzte* Baupläne, also z. B. ein fertiges Haus. Demzufolge bietet die Serendipity Plugin-API eine Reihe von Bauplänen an, nach denen Sie Ihre Plugins entwickeln können. Teile der Plugin-API stehen als *fertige Häuser* direkt zur Verfügung und können aktiv aufgerufen werden, um z. B. eine Liste aller verfügbaren Plugins zu erhalten.

Die Plugin-API unterscheidet sich dabei leicht, je nachdem, ob sie bei einem Ereignis-Plugin oder einem Seitenleisten-Plugin zum Einsatz kommt.

Ein Plugin besteht dabei aus einer PHP-Datei wie `serendipity_event_testplugin.php`. In dieser PHP-Datei wird eine Sammlung von Plugin-API-Aufrufen innerhalb einer sogenannten *Klasse* gekapselt. Sämtliche Aktionen und Meta-Informationen des Plugins befinden sich in dieser Datei und müssen in einem gleichnamigen Verzeichnis wie z. B. `plugins/serendipity_event_testplugin` abgelegt werden.

Der Serendipity-Kern kümmert sich automatisch darum, alle installierten Plugin-Dateien zu laden (*instanzieren*) und an den gewünschten Stellen auszuführen.

Dazu verwendet Serendipity ein ereignisgesteuertes Prinzip. Ein PHP-Script läuft immer linear von *oben nach unten* ab und führt alle enthaltenen Kommandos aus. Genauso läuft auch (wie auf Seite 591 dokumentiert) der Kern-Workflow von Serendipity ab. An zahlreichen Stellen können sogenannte *Ereignisse* (Events) stattfinden, die die Ausführung des Kerns kurzzeitig *pausieren* und stattdessen alle Plugins ausführen, die sich für das jeweilige Ereignis registriert haben.

Das Plugin erhält dabei eine Liste an Übergabeparametern und -variablen (`Event-Daten`), mit denen es beliebige Aktionen durchführen kann. Nachdem dies geschehen ist, wird das nächste Plugin für dasselbe Ereignis ausgeführt – solange, bis keine weiteren auszuführenden Plugins mehr vorhanden sind. Dann kehrt der Kern-Workflow zurück zu der Stelle vor dem Plugin-Aufruf und fährt fort mit den Daten, die möglicherweise von Plugins verändert wurden.

Ein Beispiel: Wenn ein Artikel gespeichert wird, bereitet Serendipity die Daten in einem um-

fangreichen PHP-Array vor. Bevor diese Daten des Artikels nun in der Datenbank gespeichert werden, wird das Ereignis namens `backend_save` ausgeführt. Wenn ein Plugin auf dieses Ereignis *lauscht*, kann es die Artikeldaten verändern. Beispielsweise kann das Plugin *Google Sitemap* dafür sorgen, dass Google über den gerade erstellten Artikel informiert (*gepingt*) wird.

Ein Plugin kann nur entweder als Ereignis- oder als Seitenleisten-Plugin vorliegen. Seitenleisten-Plugins können lediglich Ausgaben in der Seitenleiste einbinden und auf keine Ereignisse reagieren. Ereignis-Plugins wiederum können ohne Weiteres keine Ausgaben in der Seitenleiste vornehmen. Sie können jedoch Abhängigkeiten von Plugins untereinander definieren und sowohl ein Seitenleisten-Plugin als auch ein Ereignis-Plugin im selben Verzeichnis unterbringen.

Bei der Erstellung eines neuen Plugins ist es am einfachsten, sich ein bestehendes Plugin vorzunehmen, das so ähnlich funktioniert wie das, was Sie programmieren möchten. Kopieren Sie dieses Plugin dann in ein neues Verzeichnis mit neuem Namen, benennen Sie die `.php`-Datei entsprechend um und ändern Sie den Namen der PHP-Klasse in dieselbe Bezeichnung. Nun können Sie nach Belieben diesen Plugin-Klon verändern und Ihren Anforderungen entsprechend anpassen.

Es gibt bereits zahlreiche Plugins für Serendipity, so dass Sie für jedes Vorhaben zumindest eine grobe Vorlage finden sollten.

10.8.1 Seitenleisten-API

Die API eines Seitenleisten-Plugins ist die einfachste Form, um sich mit einem Serendipity-Plugin vertraut zu machen. Seitenleisten-Plugins müssen mit dem Präfix `serendipity_plugin_...` benannt werden.

Ein einfaches Plugin, das lediglich die Ausgabe `Hallo Welt!` in der Seitenleiste liefert, können Sie als `plugins/serendipity_plugin_helloworld/serendipity_plugin_helloworld.php` speichern:

```
<?php
if (IN\_serendipity !== true) \{
    die (’Don’t hack!’);
\}

class serendipity\_plugin\_helloworld extends serendipity\_plugin \{
    var \$title = ’Beispiel-Plugin: Hello world!’;

    function introspect(\&\$propbag) \{
        global \$serendipity;

        \$this->title = \$this->get\_config(’title’, \$this->title);

        \$propbag->add(’name’,          ’Beispiel-Plugin: Hello world!’);
```

```
        \$propbag->add('description',   'Beschreibung des Plugins');
        \$propbag->add('stackable',     true);
        \$propbag->add('author',        'Garvin Hicking');
        \$propbag->add('version',       '47.11');
        \$propbag->add('requirements',  array(
            'serendipity' => '0.8',
            'smarty'      => '2.6.7',
            'php'         => '4.1.0'
        ));
        \$propbag->add('groups', array('FRONTEND\_VIEWS'));
        \$propbag->add('configuration', array('title', 'intro'));
    \}

    function introspect\_config\_item(\$name, \&\$propbag) \{
        switch(\$name) \{
            case 'title':
                \$propbag->add('type', 'string');
                \$propbag->add('name', TITLE);
                \$propbag->add('description', '');
                \$propbag->add('default', \$this->title);
                break;

            case 'intro':
                \$propbag->add('type', 'string');
                \$propbag->add('name', 'Ihr Text');
                \$propbag->add('description', '');
                \$propbag->add('default', 'Hallo welt!');
                break;

            default:
                return false;
        \}
        return true;
    \}

    function generate\_content(\&\$title) \{
        global \$serendipity;
        \$title       = \$this->get\_config('title', \$this->title);
        \$intro       = \$this->get\_config('intro');

        echo \$intro . '<br />\n';
    \}
\}
```

Dieses Plugin ist stark vereinfacht, es enthält beispielsweise keine internationalisierte Sprachverwaltung. Dies wird in Plugins meist per *Include* einer Datei wie `lang_de.inc.php` gelöst, die die entsprechenden Sprachkonstanten festlegt. Eingebunden wird eine solche Sprach-

datei dann mit folgendem Code am Anfang der PHP-Datei:

```
$probelang = dirname(__FILE__) . '/' . $serendipity['charset'] .
'lang_' . $serendipity['lang'] . '.inc.php';
if (file_exists($probelang)) {
    include $probelang;
}

include dirname(__FILE__) . '/lang_en.inc.php';
```

Die Variable `$probelang` enthält den Dateinamen der lokalen Sprachdatei, die eingebunden wird. Zusätzlich bindet das Plugin zur Sicherheit die englische Sprachdatei ein, da diese Datei manchmal einige noch nicht übersetzte Konstanten enthalten kann.

Gehen wir nun den vollständigen Code des Plugins einmal durch. Die ersten drei Zeilen beinhalten eine Sicherheitsabfrage, ob das Plugin auch wirklich im Serendipity-Kontext ausgeführt wird. Die Konstante `IN_serendipity` ist nur dann gesetzt, wenn das Serendipity-Framework geladen wurde.

Danach wird die PHP-Klasse `serendipity_plugin_helloworld` definiert, die sich von der Basisklasse `serendipity_plugin` ableitet und deren Methoden einsetzt.

Die Methode `introspect()` dient dem Plugin dazu, seine Metadaten festzulegen. Dort wird ein Hilfsobjekt `$propbag` verwendet, das ein beliebiges Array mit Daten wie dem Namen des Plugins, der Beschreibung, etwaigen Copyrights, Voraussetzungen und Liste von Konfigurationsoptionen enthalten kann.

Die Methode `introspect_config_item()` wird verwendet, damit das Plugin beliebige Konfigurationsoptionen im Plugin-Manager darstellen und auswerten kann.

Zuletzt wird die Methode `generate_content()` deklariert, die später die eigentliche Ausgabe des Plugins mit beliebigem PHP-Code enthält.

Dabei liest das Plugin die Konfigurationsoption `intro` aus, mit der Sie in der Konfiguration des Plugins einen beliebigen HTML-Text eintragen konnten. Diese Variable wird ganz einfach ausgegeben.

Sobald ein derartiges Plugin auf dem Server gespeichert wurde, können Sie es wie gewohnt über die Plugin-Oberfläche aktivieren. Achten Sie darauf, etwaige PHP-Fehler zu vermeiden. Ansonsten werden diese im Backend oder auch im Frontend ausgegeben. Fatale PHP-Fehlermeldungen (Syntaxfehler) können zudem dazu führen, dass das gesamte Serendipity-Frontend nicht mehr aufgerufen werden kann. Wenn Sie den Fehler nicht direkt finden, können Sie einfach die PHP-Datei des Plugins löschen, um Ihr Blog wieder funktionabel zu machen.

10.8.2 Ereignis-API

Ganz ähnlich wie ein Seitenleisten-Plugin sieht die Struktur eines Ereignis-Plugins aus.

Auch hier wollen wir als Beispiel ein einfaches Plugin herstellen, das lediglich ein Stück HTML-Code im Frontend ausgibt.

Dieses speichern wir als `plugins/serendipity_event_helloworld/serendipity_event_hellow`

```
<?php
if (IN\_serendipity !== true) \{
    die (’Don’t hack!’);
\}

class serendipity\_event\_helloworld extends serendipity\_event \{
    function introspect(\&\$propbag) \{
        global \$serendipity;
        \$propbag->add(’name’,        ’Ereignis-Plugin: Hello world!’);
        \$propbag->add(’description’, ’Ereignis-Plugin Beschreibung’);
        \$propbag->add(’event\_hooks’, array(’entries\_header’ => true,
                                        ’entry\_display’  => true));
        \$propbag->add(’configuration’, array(’headline’, ’pagetitle’));
        \$propbag->add(’author’, ’Garvin Hicking’);
        \$propbag->add(’version’, ’4.8.15.16.23.42’);
        \$propbag->add(’requirements’,  array(
            ’serendipity’ => ’0.8’,
            ’smarty’      => ’2.6.7’,
            ’php’         => ’4.1.0’
        ));
        \$propbag->add(’groups’, array(’FRONTEND\_EXTERNAL\_SERVICES’));
        \$propbag->add(’stackable’, true);
    \}

    function introspect\_config\_item(\$name, \&\$propbag) \{
        global \$serendipity;

        switch(\$name) \{
            case ’headline’:
                \$propbag->add(’type’,        ’string’);
                \$propbag->add(’name’,        ’Seitentitel’);
                \$propbag->add(’description’, ’’);
                \$propbag->add(’default’,     ’Beispiel’);
                break;

            case ’pagetitle’:
                \$propbag->add(’type’,        ’string’);
                \$propbag->add(’name’,        ’URL-Variable’);
                \$propbag->add(’description’, ’’);
                \$propbag->add(’default’,     ’beispiel’);
                break;

            default:
```

```
                return false;
        \}
        return true;
    \}

    function show() \{
        global \$serendipity;

        if (\$this->selected()) \{
            if (!headers\_sent()) \{
                header('HTTP/1.0 200');
            \}

            \$serendipity['smarty']->assign('staticpage\_pagetitle',
              preg\_replace('@[^a-z0-9]@i', '\_',
                \$this->get\_config('pagetitle')));
            echo '<h4 class='serendipity\_title'><a href='#'>' .
              \$this->get\_config('headline') . '</a></h4>';

            echo '<div>Bitte gib mir nur ein: Oh!</div>';
        \}
    \}

    function selected() \{
        global \$serendipity;
        if (\$serendipity['GET']['subpage'] == \$this->get\_config(
'pagetitle') ||
            preg\_match('@^' .
             preg\_quote(\$this->get\_config('permalink'))
               . '@i', \$serendipity['GET']['subpage'])) \{
            return true;
        \}

        return false;
    \}

    function event\_hook(\$event, \&\$bag, \&\$eventData, \$addData =
     null) \{
        global \$serendipity;

        \$hooks = \&\$bag->get('event\_hooks');

        if (isset(\$hooks[\$event])) \{
            switch(\$event) \{
                case 'entry\_display':
                    if (\$this->selected()) \{
                        if (is\_array(\$eventData)) \{
```

```
                            \$eventData['clean\_page'] = true;
                        \} else \{
                            \$eventData = array('clean\_page' => true);
                        \}
                    \}

                    return true;
                    break;

                case 'entries\_header':
                    \$this->show();

                    return true;
                    break;

                default:
                    return false;
                    break;
            \}
        \} else \{
            return false;
        \}
    \}
\}
```

Hier bestehen nur wenige Unterschiede:

- Das Plugin verwendet das Präfix `serendipity_event_` anstelle von `serendipity_plugin_`.
- In der `introspect(`-Methode gibt das Plugin an, auf welche Ereignisse es reagieren will (`event_hooks`).
- Es deklariert zwei Hilfsmethoden, `show()` und `selected()`. Mit diesen beiden Methoden kapselt das Plugin Zugriffe auf die Darstellung seines Inhalts und die Überprüfung, ob es aufgerufen wird.
- Die zentrale Methode stellt `event_hook()` anstelle von `generate_content()` bei Seitenleisten-Plugins dar.

Wichtigste Stelle aller Ereignis-Plugins ist die `event_hook()`-Methode. Sie ist eine Art *Dispatcher* (Verteiler) und wird für jedes Ereignis aufgerufen, so dass das Plugin anhand der Parameter entscheidet, ob es zu diesem Ereignis eine Hilfsmethode oder direkten PHP-Code ausführt.

Diese Prüfung findet in einer großen `switch`-Anweisung statt, die alle möglichen und *gelauschten* Ereignisse mit dem aktuellen Ereignis abgleicht und den entsprechenden Codeteil ausführt.

Sobald Sie obiges Beispiel-Plugin installiert haben, können Sie es über die URL `http://www.example.com/serendipi subpage]=beispiel` aufrufen. Die URL-Variable `subpage` richtet sich dabei nach dem Wert, den Sie im Plugin für `pagetitle` festgelegt haben.

Ereignis-Plugins sind meist komplexer als Seitenleisten-Plugins, weil sie mehr Aktionen ausführen müssen und mehr Abhängigkeiten berücksichtigen. Das hier vorgestellte Plugin ist daher sogar schon etwas komplexer, da es einen ganz eigenen Inhalt im Frontend einbindet. Einfachere Ereignis-Plugins könnten auch mit weniger Code schon simple Dinge ausführen, wie Leerzeilen durch den HTML-Code `<br />` (PHP-Funktion `nl2br()`) zu ersetzen:

```
<?php
if (IN\_serendipity !== true) \{
    die (’Don’t hack!’);
\}

class serendipity\_event\_nl2br extends serendipity\_event \{
    function introspect(\&\$propbag) \{
        global \$serendipity;

     \$propbag->add(’name’,          ’nl2br’);
     \$propbag->add(’event\_hooks’, array(’frontend\_display’ => true));
    \}

    function event\_hook(\$event, \&\$bag, \&\$eventData) \{
        global \$serendipity;

        \$hooks = \&\$bag->get(’event\_hooks’);
        if (isset(\$hooks[\$event])) \{
            switch(\$event) \{
              case ’frontend\_display’:
                \$eventData[’body’] = nl2br(\$eventData[’body’]);
                return true;
                break;

              default:
                return false;
            \}
        \} else \{
            return false;
        \}
    \}
\}
```

Dieses Plugin entspricht einer entschlackten Version des `serendipity_event_nl2br`-Plugins von Serendipity, aus dem alle unnötigen Attribute entfernt wurden.

10.8.3 Methoden der Plugin-API

Die Serendipity Plugin-API besteht im Wesentlichen aus fünf Klassen:

- `serendipity_plugin_api` für generelle API-Aufrufe,
- `serendipity_plugin` als Basis für abgeleitete Plugin-Klassen,
- `serendipity_event` als Basis speziell für abgeleitete Ereignis-Plugins und
- `serendipity_plugin_api_extension` für weiterführende API-Aufrufe.
- Konfigurationseigenschaften und Meta-Werte von Plugins werden über die Klasse `serendipity_proper` abgedeckt.

Alle Klassen sind in `include/plugin_api.inc.php` und `include/plugin_api_extension.inc.p` deklariert. Die Funktionsparameter sind in dieser Datei ausführlich mittels `phpDoc`-Kommentaren beschrieben. Schlagen Sie die Parameter-Dokumentation bitte in dieser Datei nach.

Klasse serendipity_plugin_api

Die Kern-Aufrufe der Plugin-API können über PHP meist statisch aufgerufen werden, d. h. mittels `serendipity_plugin_api::methodenname()`. Der Serendipity-PHP-Kern macht hiervon regen Gebrauch, um nicht unnötigerweise eine Objektinstanz weiterreichen zu müssen.

Nur wenige dieser Methoden sollten in Ihren Plugins angesprochen werden, da die API größtenteils selbstverwaltend ist. Wichtiger für selbstentwickelte Plugins sind vielmehr die Objektmethoden der abgeleiteten Klassen `serendipity_plugin` und `serendipity_event`.

Folgende Methoden sind verfügbar:

`register_default_plugins()`
: Wird bei der Installation des Blogs aufgerufen, um eine Liste von Standard-Plugins einzufügen.

  Zusätzliche Plugins können über die Datei `plugins/preload.txt` gesteuert werden. Diese Datei kann ein Plugin pro Zeile (getrennt durch Zeilenumbruch) festlegen und getrennt durch einen Doppelpunkt angeben, auf welcher Seite (`left`, `right`, `event`) das Plugin installiert werden soll:

```
serendipity_event_nl2br:event
serendipity_plugin_history:left
...
```

`create_plugin_instance()`
: Aktiviert ein Serendipity-Plugin. Die Parameter der Methode bestimmen den PHP-Klassennamen sowie die Positionierung des Plugins. Wenn diese Methode eine Plugin-Datei nicht finden/einbinden kann, liefert sie eine Fehlermeldung wie:

  ```
  Fehler: serendipity_event_nl2br:dsfsdf323424334 ()
  ```

  Dabei gibt die API die eindeutige ID des nicht ladbaren Plugins sowie den Pfad zu der Plugin-Datei aus. Falls keine Pfadangabe vorhanden ist, liegt das Plugin in einem gleichnamigen Verzeichnis der `plugins`-Struktur.

`remove_plugin_instance()`
: Entfernt ein aktiviertes Plugin. Die Plugin-Dateien werden jedoch beibehalten, so dass das Plugin jederzeit erneut aktiviert werden kann.

`remove_plugin_value()`
: Löscht alle Konfigurationswerte eines gewünschten Plugins.

`enum_plugin_classes()`
: Durchsucht die gesamte Serendipity-Verzeichnisstruktur nach aktivierbaren Plugins.

`traverse_plugin_dir()`
: Durchsucht ein spezielles Verzeichnis rekursiv nach vorhandenen Serendipity-Plugins.

`get_installed_plugins(), enum_plugins()`
: Liefert eine Liste von installierten Plugins.

`count_plugins()`
: Liefert die Anzahl von installierten Plugins.

`getClassByInstanceID()`
: Liefert den PHP-Klassennamen eines Plugins anhand seiner ID.

`is_event_plugin()`
: Prüft, ob ein bestimmtes Plugin ein Ereignis-Plugin darstellt.

`exists()`
: Prüft, ob bereits ein Plugin desselben Typs installiert ist.

`autodetect_instance()`
: Prüft, ob bereits ein Plugin desselben Typs installiert ist. Wenn nicht, wird das entsprechende Plugin automatisch aktiviert.

`load_plugin()`
: Erzeugt ein PHP-Objekt einer gewünschten Plugin-Klasse. Ruft alle notwendigen Hilfsfunktionen auf, um die Plugin-Pfade und Metadaten zu belegen.

`includePlugin()`
: Hilfsfunktion, um die PHP-Klassendefinition eines Plugins einzubinden.

`probePlugin()`
: Hilfsfunktion, um die Metadaten eines Plugins einzulesen und gegebenfalls die notwendigen PHP-Dateien des Plugins einzubinden.

`getPluginInfo()`
: Liest die in der Datenbank zwischengespeicherten Meta-Informationen zu einem Plugin ein.

`setPluginInfo()`
: Speichert Meta-Informationen zu einem Plugin in der Datenbank.

`update_plugin_placement()`
: Aktualisiert die Position und Ausführungsreihenfolge (links, rechts ...) eines Plugins.

`update_plugin_owner()`
: Aktualisiert den Eigentümer (Blog-Redakteur) eines Plugins.

`get_event_plugins()`
: Hilfsfunktion, um die Liste aller aktivierten Ereignis-Plugins einzulesen und gegebenenfalls in den Speicher zu laden.

`generate_plugins()`
: Führt alle Plugins einer gewünschten Seitenleiste aus und liefert deren Inhalte zurück. Wird von der Smarty-Template-API aufgerufen (siehe Seite 480).

`get_plugin_title()`
: Liest den Titel eines Plugins aus.

`hook_event()`
: Die `hook_event()`-Methode wird aufgerufen, um ausgehend vom Abarbeitungsprozess des Serendipity-Frameworks ein Ereignis einzubinden. Diese Methode geht die Liste aller aktivierten Ereignis-Plugins durch und prüft, welche Plugins für das gerade aufgerufene Ereignis ausgeführt werden sollen. Die Methode leitet auch die benötigten Parameter und Variablen an das jeweilige Plugin weiter.

Klasse serendipity_property_bag

Ein `serendipity_property_bag` dient als Kapselung für beliebige Variablen einer Serendipity-Plugin-Klasse. Dieser Container kann von der Plugin-API leicht ausgelesen und weitergereicht werden.

Vor allem dient dieser Container der Übergabe an die Plugin-Methode `introspect()`, wo er mit individuellen Informationen eines Plugins gefüllt werden kann. Dieser Vorgang findet jedesmal beim Instanzieren eines Plugins statt.

Intern speichert dieses Objekt seine Werte in einem Klassenarray `$this->properties` ab. Die folgenden Methoden kapseln den Zugriff auf dieses private Array:

`add()`
: Fügt eine neue Eigenschaft (Schlüssel und Wert) hinzu.

`get()`
: Liest den Wert einer Eigenschaft aus.

`is_set()`
: Prüft, ob eine bestimmte Eigenschaft gesetzt ist.

Klasse serendipity_plugin_api_extension

Diese Plugin-API-Erweiterungsklasse dient dazu, weitere (seltener benötigte) Hilfsmethoden zu definieren.

`prepareReorder()`
: Operiert auf einem übermittelten Array und überführt es in eine eindimensionale Struktur, die entsprechend des Arrays sortiert wird. Dies dient z. B. als Hilfsmethode für die Sortierung einer Reihenfolge von Plugins, statischer Seiten oder anderem.

`doReorder()`
: Liefert den notwendigen SQL-Code zum Umsortieren von Datenbankeinträgen anhand des Arrays, das mithilfe der Methode `prepareReorder` erzeugt wurde.

`prepareDelete()`
: Durchwandert ein Array und prüft, ob bestimmte Werte darin aufgrund einer Eltern/Kind-Beziehung gelöscht werden sollen.

`isEmail()`
: Prüft, ob ein übermittelter Wert eine gültige E-Mail-Adresse nach RFC-Norm darstellt.

Klasse serendipity_plugin

Die Klasse `serendipity_plugin` dient als Basis sowohl für Ereignis- als auch Seitenleisten-Plugins. Viele der hier aufgeführten Methoden entsprechen lediglich einer *Rohform*, bei objektorientierten Entwurfsmustern spricht man hier von der *Template-Methode*.

Es obliegt dem jeweiligen Plugin, die Inhalte für diese Methoden zu gestalten. Die API selbst greift später nur auf die vom Plugin ausgestalteten Methoden standardisiert zu.

Jedes Plugin verfügt über folgende Klassenvariablen, die automatisch beim Laden eines Plugins belegt werden:

`$instance`
: Enthält die aktuelle ID des Plugins.

`$protected`
: Gibt an, ob ein Plugin nur durch dessen Eigentümer konfiguriert werden darf.

`$title`
: Enthält den Titel eines Plugins.

`$pluginPath`
: Enthält den Verzeichnispfad zu der Plugin-Datei.

`$pluginFile`
: Enthält den vollständigen Pfad zu einem Plugin.

`$serendipity_owner`
: Enthält die ID des Redakteurs, der das Plugin installiert hat.

`$dependencies`
: Kann ein Array mit Plugin-Namen enthalten, von denen ein Plugin abhängig ist. Die Klassenvariable kann innerhalb der `introspect()`-Methode gesetzt werden:

```
function introspect(&$propbag) {
 ...
 $this->dependencies = array('serendipity_event_creativecommons' =>
 'remove'); ...
}
```

  Der Schlüssel des Arrays enthält dabei den Klassennamen der Abhängigkeit, während der Wert des Arrays entweder `remove` (das zugehörige Plugin wird entfernt, wenn das abhängige Plugin gelöscht wird) oder `keep` (das zugehörige Plugin bleibt auch nach Deinstallation des abhängigen Plugins aktiviert) enthält.

Folgende Methoden werden von der API mit sinnvollem Inhalt vorbelegt und können eingesetzt werden:

Konstruktor `serendipity_plugin()`
: Konstruktor eines Plugins, sorgt dafür, dass die erforderlichen Klassenvariablen korrekt belegt werden.

`validate()`
: Prüft einen Plugin-Konfigurationswert auf Gültigkeit. Dabei werden die mittels `introspect_config_it` Methode festgelegten Prüfklassen (*Property Bag*-Eigenschaft `validate`) eingesetzt.

`get_config()`
: Liefert den Wert einer Konfigurationsoption des Plugins zurück.

`set_config()`
: Setzt einen Konfigurationswert des Plugins.

`register_dependencies()`
: Installiert die abhängigen Plugins, die im Array `$this->dependencies` aufgeführt wurden.

`parseTemplate()`
: Kann von Plugins eingesetzt werden, um Smarty-`.tpl`-Dateien zu parsen. Der in dieser Methode enthaltene Code befindet sich bereits in vielen Plugins an Stellen innerhalb des normalen Codeflusses. Um diese für zukünftige Plugins zu vereinheitlichen, wurde die zentrale Methode eingeführt. Sie sorgt automatisch dafür, dass Template-Dateien auch im lokalen, vorrangigen Template-Verzeichnis eines Blogs anstelle des Plugin-Verzeichnisses liegen können.

Folgende Methoden müssen von jeweiligen Plugin ausgestaltet werden:

`install()`
: Wird aufgerufen, wenn ein Plugin installiert wurde. Hier kann ein Plugin z. B. notwendige Datenbanktabellen erstellen.

`uninstall()`
: Wird beim Deinstallieren eines Plugins aufgerufen, um z. B. angelegte Datenbanktabellen zu löschen.

`performConfig()`
: Wird immer dann aufgerufen, wenn ein Plugin durch einen Redakteur konfiguriert wird. Diese Methode kann vor allem dann hilfreich sein, wenn ein Plugin Operationen durchführen soll, die nur während der Konfigurationsphase wichtig sind und nicht bei jeder Ausführung des Plugins.

`example()`
: Wird ebenfalls aufgerufen, wenn die Konfigurationsoberfläche eines Plugins angezeigt wird. Über diese Methode kann ein Plugin einen erweiterten Hilfe/Info-Text anzeigen, der die Bedienung des Plugins erklärt.

`cleanup()`
: Wird jedesmal dann aufgerufen, wenn ein Redakteur Änderungen an der Konfiguration des Plugins durchgeführt hat. Kann benutzt werden, um leere oder ungültige Konfigurationswerte zu verändern.

`introspect()`
: Diese zentrale Methode legt die Meta-Eigenschaften eines Plugins anhand eines `serendipity_property_bag`-Objekts fest. Folgende Schlüssel für dieses Objekt sind dabei geläufig:

  `name`
  : enthält den Namen des Plugins.

  `description`
  : enthält die Beschreibung eines Plugins.

`configuration`
: enthält ein eindimensionales Array mit einer Liste von Konfigurationsoptionen, die das Plugin anbietet.

`stackable`
: legt fest, ob ein Plugin mehrfach installiert werden darf (`true/false`).

`author`
: enthält den Namen des Autors eines Plugins.

`version`
: enthält die Versionsnummer des Plugins.

`copyright`
: enthält den Lizenztypen des Plugins (BSD, GPL ...).

`groups`
: enthält ein eindimensionales Array mit einer Liste von Plugin-Gruppen, denen dieses Plugin zugewiesen wird. Diese Gruppen werden in der Plugin-Oberfläche ausgewertet, um Plugins sinnvoll zu gliedern.

`requirements`
: kann ein verschachteltes PHP-Array enthalten (Unterschlüssel `serendipity`, `smarty` und `php`), das die jeweiligen Versionsabhängigkeiten des Plugins festlegen kann.

`event_hooks`
: enthält ein eindimensionales Array mit einer Zuordnung der Ereignisse, für die sich ein Ereignis-Plugin registriert hat. Diese Eigenschaft gilt nicht für Seitenleisten-Plugins.

`cachable_events`
: enthält ein Array analog zu `event_hooks`, jedoch in diesem Fall mit einer Liste von Ereignissen, die durch das vorliegende Plugin gecached werden können (siehe Seite 658).

`generate_content()`
: Zentrale Methode eines Seitenleisten-Plugins, um Inhalte in der Seitenleiste darzustellen.

`introspect_config_item()`
: Bei der Konfiguration eines Plugins wird über `introspect()` im Property-Bag-Attribut `configuration` festgelegt, über welche Konfigurationsoptionen ein Plugin verfügt.

  Die Methode `introspect_config_item()` wird von der Plugin-API für jedes dieser Elemente aufgerufen und muss daraufhin festlegen, wie die jeweilige Konfigurationsoption ausgegeben wird. Dazu empfiehlt sich, in der Methode eine große `switch`-Anweisung zu hinterlegen:

```
function introspect_config_item($name, &$propbag) {
    switch ($name) {
    case 'id':
        $propbag->add('type', 'string');
        $propbag->add('name', 'Affiliate ID');
        $propbag->add('description', '');
        $propbag->add('default', '47110815');
        break;

    case 'tax':
        $propbag->add('type', 'string');
        $propbag->add('name', 'Steuern');
        $propbag->add('description', '(in Prozent)');
        $propbag->add('default', '19');
        break;
    }
}
```

Erneut wird ein *Property Bag*-Objekt mit den Eigenschaften jedes Konfigurationswerts gefüllt. Dort gelten folgende Array-Schlüssel:

`type`
: legt den Typ einer Konfigurationsoption fest (`string`, `boolean`, `text` …). Die verfügbaren Typen entsprechen denen bei der Festlegung von Template-Optionsfeldern und sind auf Seite 501 aufgeführt.

`name`
: legt den dargestellten Namen des Konfigurationsfelds fest.

`description`
: legt die Beschreibung des Konfigurationsfelds fest.

`default`
: legt den Standardwert eines Konfigurationsfelds fest.

`validate`
: legt fest, wie ein Konfigurationsfeld geprüft werden kann. Mögliche Werte sind `string` (nur Buchstaben und Ziffern), `words` (nur Zeichenketten ohne Sonderzeichen), `number` (nur Zahlen), `url` (nur URLs), `mail` (nur E-Mail-Adressen) oder `path` (nur gültige Verzeichnisnamen). Wenn keiner dieser festgelegten Typen verwendet wird, können Sie einen beliebigen regulären Ausdruck eintragen, z. B. `/[A-Z]/`. Die *Property Bag*-Eigenschaft `validate_error` enthält eine Fehlermeldung, die bei ungültigen Eingaben angezeigt werden kann.

Die Werte der entsprechenden Konfigurationsoption können im Plugin mittels `$this->get_config('konfigwert` ausgelesen werden.

Klasse serendipity_event

Ereignis-Plugins leiten sich von der Klasse `serendipity_event` ab und erben dadurch sämtliche aufgeführten Klassenvariablen und Methoden der Klasse `serendipity_plugin`. Hinzu kommen lediglich folgende Methoden:

Konstruktor `serendipity_event()`
: Setzt analog zum Konstruktor `serendipity_plugin` die benötigten Klassenvariablen.

`getFieldReference()`
: Hilfsmethode, um bei gecachten Plugins auf ein spezielles Feld (`body` oder `extended`) zuzugreifen und es zu verändern. Würde das Plugin direkt `$eventData['body']` modifizieren, könnte dies in Interaktion mit dem Caching-Plugin zu Problemen führen.

`event_hook()`
: Zentrale *Dispatcher*-Methode, damit die jeweils registrierten Ereignisse durch das Plugin abgefragt und ausgeführt werden können.

10.8.4 Cachable Events

Ein Ereignis-Plugin kann in seiner `introspect`-Methode eine Liste von Ereignissen in der Eigenschaft `event_hooks` des `$propbag`-Objekts registrieren. Diese Liste legt fest, bei welchen Ereignissen das Plugin aufgerufen wird.

Dieser Vorgang wird etwas komplexer, wenn die Caching-Konfigurationsoption des Plugins *Erweiterte Eigenschaften von Artikeln* (siehe Seite 262) aktiviert ist.

Das Caching ist ausschließlich für Textformatierungs-Plugins (siehe Seite 237) gedacht. Normalerweise ruft Serendipity diese für jeden einzelnen Artikel wieder und wieder auf. Bei vielen Textformatierungen macht das performancemäßig jedoch wenig Sinn: Aus einer BBCode-Textformatierung wie `[b]fett[/b]` wird stets ein `<strong>fett</strong>` entstehen; dies müsste Serendipity also eigentlich nur einmal *parsen* und könnte dann für spätere Aufrufe immer nur das vorige Ergebnis (den *Cache*) einlesen und verwenden.

Sobald viele Textformatierungs-Plugins zum Einsatz kommen, kann sich dieses Caching durchaus als performanceschonend auswirken. Vor dem Speichern eines Blog-Artikels führt der Serendipity-Kern das Ereignis `backend_save` aus. Das Caching-Plugin `serendipity_event_entryproperties` lauscht auf dieses Ereignis. Aufgrund aktivierter Caching-Option führt dieses Plugin nun wiederum das Ereignis `frontend_display_cache` aus. Alle Textformatierungs-Plugins, die das Caching unterstützen (das sind die meisten), lauschen auf dieses Ereignis und wenden nun ihre reproduzierbaren Textformatierungen auf den Artikeltext an. Das Ergebnis entspricht nach diesem Vorgang der HTML-Formatierung, wie sie später im Blog dargestellt wird.

Diee Formatierung wird nun nicht einfach im Artikeltext gespeichert, denn dies würde dazu führen, dass beim nächsten Bearbeiten des Artikels der Redakteur nicht mehr sein vorher ein-

gegebenes `[b]fett[/b]` sähe, sondern den entsprechenden HTML-Code. Damit also die Redakteurseingaben immer konsistent bleiben können, wird das erzeugte Cache-Ergebnis separat in der Datenbanktabelle `serendipity_entryproperties` gespeichert. Die beiden Werte `ep_cache_body` und `ep_cache_extended` enthalten die nach Aufruf aller Textformatierungs-Plugins erzeugte HTML-Formatierung des Artikeltexts und des erweiterten Artikels.

Nachdem der einzusetzende Cache gespeichert wurde, prüft die Serendipity-Funktion `serendipity_printEntries` bei der Darstellung der Artikel, ob ein Wert für die Eigenschaften `ep_cache_body` und `ep_cache_extended` vorliegt. Ist das der Fall, tauscht die Funktion einfach den *echten* Inhalt des Artikels mit der gecachten Version aus. Wenn nun später das Ereignis `frontend_display` ausgeführt wird, enthalten die Zusatzarrays `$addData['no_scramble']` und `$eventData['is_cached']` den Wert `true`. Alle cachbaren Textformatierungs-Plugins erkennen diesen Wert und wissen, dass sie keine erneute Formatierung anwenden müssen, da dies in der gecachten Version bereits beim Speichern des Artikels erledigt wurde.

Jedes Plugin, das sich mittels des `$propbag`-Objekts (Eigenschaft `event_hooks`) für das Ereignis `frontend_display` registriert hat und cachbar sein soll, muss dieses Ereignis auch in der Eigenschaft `cachable_events` festlegen. Wurde dies gesetzt, kümmert sich die Plugin-API beim Aufruf des Textformatierungsereignisses `frontend_display` automatisch darum, dass es im Falle eines gecachten Artikels *nicht* mehr erneut ausgeführt wird. Der Aufruf des Ereignisses wird somit komplett verhindert.

Zusätzlich kann das Caching-Plugin auch einmalig alle alten Artikel auf Anfrage cachen, dies wird über das Ereignis `backend_cache_entries` ausgeführt. Das Ereignis `backend_cache_purge` löscht darüber hinaus einen Cache. Viele Textformatierungs-Plugins machen von diesen Ereignissen innerhalb der `install()`- und `uninstall()`-Methode Gebrauch, so dass bei der Installation eines neuen Textformatierungs-Plugins automatisch der bestehende Cache neu erstellt wird und die neu hinzugekommenen Formatierungen beinhaltet.

Wenn Sie ein Textformatierungs-Plugin entwickeln wollen, das diese Möglichkeiten des Cachings anwendet, suchen Sie einfach im Quellcode der Plugins nach der Zeichenkette `cachable_events`. Mithilfe dieser Beispiele können Sie den oben beschriebenen Mechanismus leichter nachvollziehen.

10.8.5 Ereignis-Hooks

In vielen Dateien von Serendipity werden Ereignisse mittels der Plugin-API ausgeführt. Diese Aufrufe sehen dabei wie folgt aus:

```
serendipity_plugin_api::hook_event('frontend_generate_plugins',
$eventData, $addData);
```

Die zentrale Methode `hook_event()` der Plugin-API geht dabei die Liste aller aktivierten Ereignis-Plugins durch. Dabei wird die Ausführungsreihenfolge durch die Reihenfolge der Plugins bestimmt. Jedem einzelnen Plugin werden die beiden Variablen `$eventData` und

`$addData` weitergereicht, falls sich das Plugin für das definierte Ereignis (hier: `frontend_generate_plugins`) registriert hat. Bei jedem Ereignis wird `event_hook()` des jeweiligen Plugins ausgeführt, an die die Variablen weitergereicht werden, wo sie verarbeitet werden können. Die Variable `$eventData` wird dabei als referenzierte Variable weitergegeben und kann vom Plugin verändert werden. `$addData` hingegen dient nur dem Lesezugriff. Beide Variablen können auch beliebig verschachtelte Arrays enthalten.

Falls ein Plugin die Inhalte der Variablen `$eventData` verändert, können diese auch durch später folgende Plugins noch weiter abgeändert werden.

Ereignisse lassen sich bei Serendipity leicht und flexibel erweitern. Dazu genügt es, im PHP-Code von Serendipity eine Zeile wie oben mit einem frei erfundenen Ereignisnamen einzubinden und ein entsprechendes Plugin zu entwickeln, das dieses Ereignis registriert. Zusätzlich können beliebige Ereignis-Hooks auch mittels Smarty-Funktion `serendipity_hookPlugin` aufgerufen werden (siehe Seite 563).

Um das konkrete Umfeld sowie die Parametrisierung eines `hook_event()`-Aufrufs zu prüfen und für ein eigenes Plugin zu nutzen, ist es am einfachsten, im Serendipity-PHP-Code nach dem Auftreten dieser Zeile zu suchen. Unter Linux geht dies leicht mit folgendem Befehl:

```
find -name \*.php \
  -exec grep -i -l -d skip \
  "serendipity_plugin_api::hook_event('" \
  {} \;
```

Es folgt eine Liste von bisher vorhandenen Serendipity-Ereignissen, auf die auch selbstgeschriebene Plugins zugreifen können:

Frontend-Ereignisse

`css`
: für die Darstellung des CSS Stylesheets. `eventData` (Array): CSS-Daten des Template-Stylesheets, kann durch Plugins verändert/erweitert werden.

`frontend_configure`
: wird aufgerufen, nachdem das Serendipity-Framework instanziert wurde. Plugins können an dieser Stelle zentrale Eigenschaften/Werte von Serendipity verändern oder prüfen. `eventData`: Serendipity-Konfigurationswerte.

`entry_display`
: wird aufgerufen, wenn eine Liste von Blog-Artikeln dargestellt wird. Dieses Ereignis wird einmalig ausgeführt. `eventData`: ein vollständiges Array mit allen darzustellenden Blog-Artikeln. `addData`: zusätzliche Parameter mit Eigenschaften des Aufrufs dieses Ereignisses.

`entries_header`
: (via Smarty `entries.tpl`) wird im Template aufgerufen, bevor die Blog-Artikel dargestellt werden. `addData`: enthält die ID eines Artikels, falls eine Einzeldarstellung erfolgt.

`entries_footer`
: (via Smarty `entries.tpl`) wird im Template aufgerufen, nachdem die Blog-Artikel dargestellt wurden.

`frontend_header`
: (via Smarty `index.tpl`) wird im Kopfbereich des Templates aufgerufen.

`frontend_footer`
: (via Smarty `index.tpl`) wird im Fußbereich des Templates aufgerufen.

`entry_groupdata`
: wird aufgerufen, wenn die Liste der darzustellenden Artikel nach speziellen Kriterien sortiert werden soll. `eventData`: vollständiges Array der zu sortierenden Blog-Artikel.

`comments_by_author_footer`
: (via Smarty `comments_by_authors.tpl`) wird im Template innerhalb der Fußzeile aufgerufen, nachdem die Kommentaransicht nach Autoren dargestellt wurde.

`frontend_fetchentries`
: wird aufgerufen, wenn Serendipity eine Datenbankabfrage ausführt, um mehrere Blog-Artikel einzulesen. Über dieses Ereignis kann die SQL-Abfrage beeinflusst werden. `eventData`: Array mit Teilen der SQL-Abfrage (`WHERE`-Bedingungen, `JOINS` etc.) `addData`: Array mit zusätzlichen Parametern beim Aufruf dieses Ereignisses.

`frontend_fetchentry`
: wird aufgerufen, wenn Serendipity eine Datenbankabfrage ausführt, um einen einzelnen Blog-Artikel einzulesen. Über das Ereignis kann die SQL-Abfrage beeinflusst werden. `eventData`: Array mit Teilen der SQL-Abfrage (`WHERE`-Bedingungen, `JOINS` etc.) `addData`: Array mit zusätzlichen Parametern beim Aufruf dieses Ereignisses.

`frontend_entryproperties`
: wird aufgerufen, wenn zusätzliche Eigenschaften zu den ausgelesenen Blog-Artikeln aus weiteren Datenquellen eingefügt werden sollen. `eventData`: verschachteltes Array mit den Daten der Blog-Artikel. `addData`: Array mit allen IDs der in `addData` enthaltenen Blog-Artikel.

`frontend_entryproperties_query`
: wird aufgerufen, wenn Serendipity die Datenbanktabelle `serendipity_entryproperties` ausliest, über die freie Eigenschaften zu Blog-Artikeln zugewiesen werden können. Über dieses Ereignis können innerhalb einer SQL-Abfrage mehrere Tabellen einbezogen werden. `eventData`: enthält ein Array mit Teilen der SQL-Abfrage, über die

erweiterte Eigenschaften der Artikel eingelesen werden können, die später in das zentrale Blog-Artikel-Array übernommen werden können.

`fetchcomments`
: wird aufgerufen, wenn Kommentare zu einem Blog-Artikel ausgelesen werden. `eventData`: Array mit eingelesenen Kommentardaten.

`frontend_display`
: wird aufgerufen, wenn der Text eines Blog-Artikels oder Kommentars formatiert bzw. dargestellt werden soll. Textformatierungs-Plugins nutzen vor allem dieses Ereignis. `eventData`: Daten eines einzelnen Artikels bzw. Kommentars. `addData`: Array mit zusätzlichen Parametern beim Aufruf dieses Ereignisses.

`frontend_display_cache`
: wird aufgerufen, wenn ein Blog-Artikel in der Datenbank gespeichert wird. Ereignis-Plugins, die Caching (siehe Seite 658) unterstützen, führen ihre Textformatierungen bei diesem Ereignis aus. `eventData`: dieselben Daten wie vom Ereignis `backend_publish/backend_sa` (Artikeldaten). `addData`: Array mit zusätzlichen Parametern beim Aufruf dieses Ereignisses.

`frontend_comment`
: (via Smarty `commentform.tpl`) wird aufgerufen, wenn das Eingabeformular für Kommentare im Frontend dargestellt wird.

`frontend_display:opml-1.0:namespace`,

`frontend_display:rss-0.91:namespace`,

`frontend_display:rss-1.0:namespace`,

`frontend_display:rss-2.0:namespace`,

`frontend_display:atom-1.0:namespace`,

`frontend_display:atom-0.3:namespace` wird bei der Darstellung von RSS-Feeds innerhalb des Namespace-XML-Bereichs aufgerufen. `eventData`: Array mit allen Artikeln des Feeds.

`frontend_display:html:per_entry`
: wird vor Darstellung eines Artikels im Frontend aufgerufen. `eventData`: Array mit Blog-Artikeln.

`frontend_display:opml-1.0:per_entry`,

`frontend_display:rss-0.91:per_entry`,

`frontend_display:rss-1.0:per_entry`,

`frontend_display:rss-2.0:per_entry`,

`frontend_display:atom-1.0:per_entry`,

`frontend_display:atom-0.3:per_entry` wird bei der Darstellung eines Blog-Artikels in einem Feed ausgeführt. `eventData`: Array mit einzelnem Artikel des Feeds.

`frontend_rss`
: wird ausgeführt, bevor die Daten eines Feeds an das Smarty-Framework weitergereicht werden.

`frontend_entries_rss`
: wird vor Darstellung eines Feed ausgeführt. `eventData`: Array mit allen Artikeln des Feeds. `addData`: Array mit weiteren Optionen zur Darstellung des Feeds. Array-Schlüssel `version` (Version des Feeds), `comments` (für Kommentar-Feeds), `fullFeed` (für Feeds mit vollständigen Artikeln statt eines Teasers), `showMail` (gesetzt, wenn E-Mail-Adresse eingebunden werden soll).

`frontend_saveComment`
: wird aufgerufen, wenn ein Kommentar zu einem Artikel gespeichert wird. Die Plugins können `$eventData['allow_comments']` auf `false` setzen, wenn ein Kommentar nicht gespeichert werden soll, z. B. aus Gründen des Spamschutzes. `eventData`: Array mit den Daten des Artikels, der kommentiert wird. `addData`: Array mit den Daten des Kommentars, der gespeichert werden soll.

`frontend_generate_plugins`
: wird aufgerufen, wenn die Seitenleisten-Plugins für eine Seite ausgewertet werden. Ereignis-Plugins können so spezielle Seitenleisten-Plugins ausblenden, wenn gewünscht. `eventData` enthält ein Array mit den Plugins einer Seitenleiste, die dargestellt werden sollen. `addData` enthält ein Array der Methodenparameter, die beim Aufruf der Plugin-API-Funktion `serendipity_plugin_api::generate_plugins()` übergeben wurden.

`quicksearch_plugin`
: wird aufgerufen, wenn das Forumular des **Suche**-Seitenleisten-Plugins dargestellt wird. Hierüber kann zusätzlicher HTML- oder JavaScript-Code ausgegeben werden, um das Suchformular z. B. mit Ajax-Fähigkeiten aufzurüsten.

`frontend_image_selector`
: (via Smarty `media_*.tpl`) sowie weitere `frontend_image_...`-Ereignisse werden von dem Mediendatenbank-Popup ausgegeben, wenn die Formatierungsoptionen für eine ausgewählte Datei angezeigt werden. Für jede der Formatierungsmöglichkeiten (Vorschau, Link, Platzierung, Kommentar ...) existiert ein eigenes Ereignis.

`external_plugin`
: wird aufgerufen, wenn ein Plugin die vollständige HTML-Ausgabe des Frontends übernehmen kann. Dies geschieht vor allem beim Aufruf einer URL wie `http://www.example.com/sere` `captcha_4711`. Alle über den virtuellen Pfad `/plugin/` aufgerufenen URLs geben die URL-Parameter an dieses Ereignis weiter, so dass jedes Plugin überprüfen kann, ob es die Ausgabe des Frontends übernehmen soll. `eventData` enthält den Teil der URL, der nach `.../plugin/` folgt.

`genpage`
: wird aufgerufen, bevor Serendipity das Smarty-Framework initiiert. `eventData` enthält die URL der aktuellen Serendipity-Seite. Über die globale Variable `$serendipity['plugindata'][` kann ein Plugin weitere beliebige Variablen an das Smarty-Framework weitergeben, `addData` enthält ein Array mit Zusatzinfos zur Seite sowie die Schlüssel `startpage` (`true`, wenn Serendipity die erste Seite des Blogs darstellt), `uriargs` (zusätzliche URL-Parameter) und `view` (gibt an, welche Template-Ansicht angefordert wurde, siehe Seite 516).

`frontend_calendar`
: wird aufgerufen, wenn das Kalender-Seitenleisten-Plugin seine Inhalte darstellt. Über dieses Ereignis können Ereignis-Plugins interne Termine einbinden. `eventData`: Array mit externen Terminen, der Array-Schlüssel muss dem aktuellen Tag des dargestellten Monats entsprechen, der Array-Wert enthält ein Unterarray mit den Schlüsseln `Class` (CSS-Klasse), `Title` (Titel des Termins) und `Extended` (weitere Textinfos zu dem Termin). `addData` enthält ein Array mit Informationen zum aktuell dargestellten Kalendermonat mit den Array-Schlüsseln `Month` (dargestellter Monat), `Year` (dargestelltes Jahr), `TS` (Unix-Zeitstempel des ersten Tages), `EndTS` (Unix-Zeitstempel des
letzten Tages).

`frontend_xmlrpc`
: wird aufgerufen, wenn die XML-RPC API (siehe Seite 351) des Blogs initiiert werden soll. `eventData`: Array, das mit Metadaten des XML-RPC API-Aufrufs gefüllt werden kann.

`xmlrpc_deleteEntry`
: wird aufgerufen, wenn mittels XML-RPC API ein Artikel gelöscht wurde.

`xmlrpc_fetchEntry`
: wird aufgerufen, wenn mittels XML-RPC API die Daten eines Artikels eingelesen werden.

`xmlrpc_updertEntry`
: wird aufgerufen, wenn mittels XML-RPC API ein Artikel aktualisiert oder gespeichert wird.

Backend-Ereignisse: Kommentare

`backend_comments_top`
: wird in der Kopfzeile der Tabellenansicht von im Backend vorhandenen Kommentaren aufgerufen. `eventData`: Array mit den auf dieser Seite dargestellten Kommentaren.

`backend_view_comment`
: wird aufgerufen, wenn Kommentare im Bereich **Kommentare** des Backends angezeigt werden. `eventData`: Array mit Daten des Kommentars. `addData`: URL-Variablen, die bei der Kommentaransicht die aktuelle Seite identifizieren.

`backend_approvecomment`
: wird aufgerufen, wenn ein moderierter Kommentar von einem Redakteur freigeschaltet wurde. `eventData` enthält ein Array mit den Daten des Kommentars.

`backend_deletecomment`
: wird aufgerufen, wenn ein Kommentar gelöscht wird. `eventData` enthält ein Array mit Daten des Kommentars. `addData` enthält ein Array mit dem Schlüssel `cid` (ID des Kommentars) und `entry_id` (ID des kommentierten Blog-Artikels).

`backend_updatecomment`
: wird aufgerufen, wenn ein Redakteur einen Kommentar bearbeitet hat. `eventData` enthält die HTTP POST-Formulardaten (mit den neuen Daten des Kommentars). `addData` enthält die ID des zu ändernden Kommentars.

Backend-Ereignisse: Artikel

`backend_delete_entry`
: wird aufgerufen, wenn ein Blog-Artikel gelöscht wird. `eventData`: ID des gelöschten Artikels.

`backend_preview`
: wird aufgerufen, wenn die Artikelvorschau im Backend angefordert wurde. `eventData`: enthält das Array mit Daten des Artikels.

`backend_publish`
: wird aufgerufen, wenn ein Artikel veröffentlicht (und somit auch gespeichert) wird. `eventData`: enthält das Array mit Daten des Artikels. `addData`: enthält `true`, wenn der Artikel vorher noch nie gespeichert wurde.

`backend_save`
: wird aufgerufen, wenn ein Artikel als Entwurf gespeichert wird. `eventData`: enthält das Array mit Daten des Artikels. `addData`: enthält `true`, wenn der Artikel vorher noch nie gespeichert wurde.

`backend_display`
: wird aufgerufen, wenn das Formular zum Erstellen und Ändern eines Blog-Artikels im Backend dargestellt wird. Plugins können bei diesem Ereignis eigene Eingabefelder einbinden, die im Bereich **Erweiterte Optionen** angezeigt werden. `eventData`: Array mit Daten des Artikels, der bearbeitet wird.

`backend_entryform`
: wird aufgerufen, wenn das Formular zum Erstellen und Ändern eines Blog-Artikels im Backend dargestellt wird. Im Gegensatz zum Ereignis `backend_display` kann ein Plugin hier auf die Daten eines Artikels zugreifen und Ausgaben am *Anfang* des Eingabeformulars einbinden.`eventData`: Array mit Daten des Artikels, der bearbeitet wird.

`backend_entryform_smarty`
: wird aufgerufen, bevor das Smarty-Template `admin/entries.tpl` zum Bearbeiten eines Artikels dargestellt wird. `eventData` enthält die Variablen des Smarty-Templates, die von der Artikel-Oberfläche zugewiesen wurden.

`backend_entry_checkSave`
: prüft gültige Eingabewerte vor dem Speichern eines neuen Artikels (mittels JavaScript) auf Gültigkeit.

`backend_entry_updertEntry`
: prüft gültige Eingabewerte vor dem Speichern eines Artikels auf etwaige Fehler. Tritt ein Fehler auf, wird der Artikel nicht gespeichert. `eventData`: Array, das mit Fehlermeldungen gefüllt werden kann. `addData`: Array mit Daten des Artikels, der geprüft wird.

`backend_entry_presave`
: wird aufgerufen, kurz bevor ein Artikel in der Datenbank gespeichert wird. `eventData`: Array mit Daten des Artikels, der gespeichert wird.

`backend_cache_purge`
: wird aufgerufen, wenn Caches von Artikeltexten geleert werden müssen (siehe Seite 658).

`backend_cache_entries`
: wird aufgerufen, wenn Caches der aktuellsten bestehenden Artikel neu erstellt werden sollen.

`backend_import_entry`
: kann während des Blog-Imports von einzelnen Importern aufgerufen werden, so dass Plugins zusätzliche Informationen für jeden importierten Artikel setzen können. `eventData`: Array des neu eingefügten Artikels. `addData`: Array mit bereits bestehenden Eigenschaften des Artikels.

`backend_entry_toolbar_body` , `backend_entry_toolbar_extended`
: wird an der Stelle des Artikelformulars im Backend ausgeführt, bei der die Symbolleiste für Formatierungen und die Einbindung von Dateien in der Mediendatenbank ausgegeben wird. Plugins können dort eigene Symbolleisten einbinden. `backend_entry_toolbar_body` gilt für die Symbolleiste des **Eintrags** und `backend_entry_toolbar_extended` für die Symbolleiste des **Erweiterten Eintrags**. `eventData`: enthält ein Array mit den Daten des Artikels, der im Backend bearbeitet wird.

`backend_entry_iframe`
: wird aufgerufen, wenn ein Blog-Artikel als Vorschau im Backend eingebunden wird. Dies erfolgt über einen iframe. `eventData` enthält standardmäßig den Wert `true`. Wenn ein Plugin dies auf `false` ändert, kann es selbständig die Erstellung eines iframe übernehmen. Der Serendipity-Kern überspringt in diesem Fall die vorgesehenen Routinen zur Erstellung.

Backend-Ereignisse: Kategorien

`backend_category_addNew`
: wird ausgeführt, wenn im Blog eine neue Kategorie angelegt wird. `eventData`: ID der neu angelegten Kategorie.

`backend_category_delete`
: wird ausgeführt, wenn ein oder mehrere Kategorien gelöscht werden. `eventData`: ID der zu löschenden Kategorie. Kann einen Von/Bis-Wert enthalten, der in einer MySQL-Abfage wie `DELETE FROM ... WHERE id IN (1, 2, 3, 4 ...)` eingesetzt werden kann.

`backend_category_showForm`
: stellt beim Bearbeiten einer Kategorie das HTML-Formular mit den möglichen Eigenschaften dar. `eventData`: ID der Kategorie, deren Eigenschaften verändert werden sollen. `addData`: enthält ein Array mit den derzeitigen Eigenschaften der Kategorie.

`backend_category_update`
: wird aufgerufen, wenn die Eigenschaften einer Kategorie verändert wurden. `eventData`: ID der Kategorie.

Backend-Ereignisse: Login- und Rechtemanagement

`backend_auth`
: wird beim Login eines Redakteurs ausgeführt, damit dessen Benutzername/Passwort auch mit externen Quellen validiert werden kann. `eventData`: enthält den Wert `true`, wenn das Passwort als MD5-Hash durchgereicht wird. `addData`: Array mit Schlüsseln `username` und `passwort`.

`backend_login`
: wird beim Login eines Redakteurs ausgeführt und muss nicht zwangsläufig ein Benutzername/Passwort-Paar beinhalten. Plugins können so Redakteure z. B. anhand von Cookie-Werten oder fester Daten wie IPs authentifizieren. `eventData`: enthält den Wert `true`, wenn der Redakteur bereits von Serendipity identifiziert werden konnte.

`backend_sidebar_entries_event_display_profiles`
: wird aufgerufen, wenn die persönlichen Einstellungen eines Redakteurs angezeigt werden. `eventData` enthält ein Array mit den Daten des Redakteurs.

`backend_users_add`
: wird aufgerufen, wenn ein neuer Redakteurszugang erstellt wurde. `eventData`: Array mit Eigenschaften des neuen Redakteurs.

`backend_users_delete`
: wird aufgerufen, wenn ein Redakteurszugang gelöscht wurde. `eventData`: Array mit Eigenschaften des zu löschenden Redakteurs.

`backend_users_edit`
: wird aufgerufen, wenn die Eigenschaften eines Redakteurszugangs verändert wurden. `eventData`: Array mit Eigenschaften des zu ändernden Redakteurs.

`backend_sidebar_entries`
: wird bei der Darstellung der Menüpunkte des Backends ausgeführt. Über dieses Ereignis können zusätzliche Menüpunkte im Bereich **Einträge** hinzugefügt werden.

`backend_sidebar_entries_images`
: wird bei der Darstellung der Menüpunkte des Backends ausgeführt. Über dieses Ereignis können zusätzliche Menüpunkte im Bereich **Mediendatenbank** hinzugefügt werden.

`backend_sidebar_admin_appearance`
: wird bei der Darstellung der Menüpunkte des Backends ausgeführt. Über dieses Ereignis können zusätzliche Menüpunkte im Bereich **Aussehen** hinzugefügt werden.

`backend_sidebar_admin`
: wird bei der Darstellung der Menüpunkte des Backends ausgeführt. Über dieses Ereignis können zusätzliche Menüpunkte im Bereich **Administration** hinzugefügt werden.

`backend_frontpage_display`
: wird bei der Darstellung der Startseite des Backends nach dem Login ausgeführt. Dort können Plugins etwaige Informationen oder weiterführende Links ausgeben. `eventData` enthält ein Array, das für jeden einzelnen Schlüssel ein spezielles HTML-Fragment enthalten kann: `welcome` für den Begrüßungstext, `links_title` für die Überschrift der **Weiterführende Links**-Box, `links` als Array mit allen dargestellten Links dieser Box, `links_css` für die CSS-Klasse dieser Box, `more` für alle weiteren HTML-Ausgaben auf der Startseite. `show_links` enthält den Wert `false`, wenn die Box

Weiterführende Links nicht angezeigt werden soll. `bookmarklet` enthält die URL zu einem JavaScript-Code, mit dem Sie in Ihrem Browser ein Lesezeichen speichern können, um sofort einen neuen Blog-Artikel zu verfassen.

`backend_login_page`
: wird aufgerufen, wenn das Login-Formular zum Backend dargestellt wird. `eventData` enthält ein Array mit mehreren HTML-Ausgabefragmenten im Login-Formular und verwendet folgende Schlüssel: `header` für die Überschrift des Login-Formulars, `table` für weitere Tabellenzeilen des Login-Formulars, `footer` für eine abschließende Fußzeile des Formulars.

Backend-Ereignisse: Mediendatenbank

`backend_image_add`
: wird aufgerufen, wenn eine neue Datei in die Mediendatenbank eingefügt wurde. `eventData` enthält den Pfad zu der eingestellten Datei. `addData` enthält den Pfad zur automatisch erzeugten Voransicht der Datei.

`backend_image_addHotlink`
: wird aufgerufen, wenn ein neuer *Hotlink* (siehe Seite 126) in die Mediendatenbank eingefügt wurde. `eventData` enthält den Pfad zu der temporären lokalen Kopie der zu verknüpfenden Datei.

`backend_image_addform`
: (via Smarty `media_upload.tpl`) wird aufgerufen, wenn das Eingabeformular zum Hochladen neuer Dateien für die Mediendatenbank dargestellt wird.

`backend_directory_create`
: wird aufgerufen, wenn ein neues Verzeichnis in der Mediendatenbank erstellt wurde. `eventData`: vollständiger Pfad des neu angelegten Verzeichnisses.

`backend_media_delete`
: wird aufgerufen, wenn eine Datei der Mediendatenbank gelöscht wird. `eventData` enthält ein Array mit zu löschenden zusätzlichen Dateien, z. B. der automatisch erzeugten Voransicht.

`backend_media_check`
: wird aufgerufen, wenn eine Datei geprüft werden soll, die gerade in die Mediendatenbank hochgeladen wird. Wenn `eventData` von einem Plugin auf den Wert `true` gesetzt wird, verbietet Serendipity das Einstellen dieser Datei. Dies kann z. B. dazu verwendet werden, um das Hochladen von JavaScript-Dateien oder unerwünschten Dateinamen zu verhindern. `addData` enthält den hochgeladenen Dateinamen.

`backend_media_makethumb`
: wird aufgerufen, wenn die Voransicht einer Grafik erstellt werden soll. `eventData`:

enthält ein verschachteltes Array, dessen Unterarrays jeweils die Schlüssel `thumbSize` (Bildgröße der Voransicht) und `thumb` (Datei-Suffix für die Voransicht) besitzen müssen. Für jedes Unterarray, das durch Plugins angehängt werden kann, wird eine weitere Voransichts-Datei erstellt.

`backend_media_path_exclude_directories`
: wird bei der automatischen Synchronisation der Mediendatenbank (siehe Seite 172) aufgerufen. Plugins können eine übergebene Liste von Datei- und Verzeichnisnamen modifizieren, um so bestimmte Dateien von der Synchronisation auszuschließen. `eventData` enthält ein Array, dessen Schlüssel die Zeichenkette des auszuschließenden Datei-/Verzeichnisnamens festlegt.

`backend_media_rename`
: wird ausgeführt, wenn ein Objekt der Mediendatenbank umbenannt wird. `eventData` enthält ein verschachteltes Array, bei dem jedes Unterarray eine umzubenennende Datei enthält. So können auch die Namen der Voransichten automatisch umbenannt und durch Plugins verwaltet werden.

`backend_thumbnail_filename_select`
: wird aufgerufen, wenn Dateinamen und Pfade eines Objekts der Mediendatenbank zusammengestellt werden. `eventData` enthält die Metadaten einer Datei der Mediendatenbank.

Backend-Ereignisse: Framework

`css_backend`
: für die Darstellung des CSS Stylesheets im Backend. `eventData` (Array): CSS-Daten des Template-Stylesheets, kann durch Plugins verändert/erweitert werden.

`backend_wysiwyg`
: wird aufgerufen, wenn im Backend ein Texteingabebereich für einen WYSIWYG-Editor erscheinen soll. `eventData`: Array mit Schlüsseln `init` (enthält `true`, wenn das WYSIWYG-Javascript bereits instanziert wurde), `item` (Name des HTML-Formularelements mit der WYSIWYG-Eingabe), `jsname` (ID des HTML-Formularelements) und `skip` (enthält `true`, wenn ein Eingabeelement nicht ausgegeben werden soll).

`backend_wysiwyg_finish`
: wird aufgerufen (entweder über PHP oder auch über Smarty-Templates), wenn das WYSIWYG-Javascript instanziert werden soll.

`backend_wysiwyg_nuggets`
: wird aufgerufen, wenn ein WYSIWYG-Editorfeld für erweiterte Texteingabefelder (z. B. HTML-Klötze, siehe Seite 216) eingebunden werden soll. `eventData`: enthält ein Array mit den Schlüsseln `nuggets` (Array mit allen vorhandenen HTML-WYSIWYG-Elementen) und

`skip_nuggets` (enthält `true`, wenn keine HTML-Klötze zu WYSIWYG-Eingabefeldern konvertiert werden sollen).

`backend_configure`
: wird aufgerufen, wenn das Serendipity-Backend initialisiert wurde (analog zum Ereignis `frontend_configure`).

`backend_header`
: wird aufgerufen, wenn der HTML-Bereich `<head>` des Serendipity-Backends dargestellt wird (analog zum Ereignis `frontend_header`).

`backend_trackbacks`
: wird aufgerufen, wenn Trackbacks eines Artikels versendet werden. `eventData` enthält ein Array mit allen auf Trackbacks zu prüfenden URLs.

`backend_trackback_check`
: wird für jede einzelne URL aufgerufen, bevor an diese ein Trackback verschickt wird. `eventData`: automatisch entdeckte Trackback-URL. `addData`: URL, bei der geprüft werden soll, ob sie Trackback-fähig ist.

`event_additional_statistics`
: wird vom Statistik-Plugin aufgerufen, damit weitere Plugins ihre eigenen Statistiken darstellen können.

`backend_sidebar_entries_event_display_`*X*
: wird mit einer vom Plugin am Ende des Ereignisnamens festgelegten Zeichenkette aufgerufen, um beliebige Ausgaben im Hauptbereich des Backends darzustellen. Über solche Ereignisse kann jedes Plugin eigene Backend-Bereiche realisieren. Diese werden über eine URL wie `http://www.example.com/serendipity/serendipity_admin.php?serendipity[adminModule]=event_display&serendipity[adminAction]=`*X* aufgerufen. Solche Links kann das Plugin über ein Ereignis wie `backend_sidebar_entries` (siehe Seite 668 leicht direkt im Backend-Menü einbinden.

`backend_http_request`
: wird von Serendipity immer dann aufgerufen, wenn vom Server eine HTTP-Verbindung zu einem fremden Server erstellt wird, z. B. bei Trackbacks, fremden RSS-Feeds, Importvorgängen und Weiterem. Der Server wird dabei mittels der PEAR-Klasse `HTTP_Request` angesprochen, die einige Optionen unterstützt. Diese Optionen werden als `eventData` an die Plugins weitergereicht, so dass sie verändert werden können, um z. B. die Verbindung über den Umweg eines Proxies aufzubauen. Der Wert der Variablen `addData` gibt dabei die Quelle der HTTP-Verbindung an: `image` (Herunterladen von Mediendaten), `spartacus` (Operationen des Spartacus-Plugins), `trackback_detect` (Trackbacks erkennen), `trackback_send` (Trackbacks senden).

`backend_sendmail`
: wird aufgerufen, wenn Serendipity eine E-Mail verschickt. `eventData` enthält ein

Array mit den Daten der E-Mail und folgenden Schlüsseln: `to` (Mailempfänger), `subject` (Betreff), `fromName` (Name des Absenders), `fromMail` (E-Mail-Adresse des Absenders), `blogMail` (E-Mail-Adresse des Blogs), `version` (Version von Serendipity), `headers` (E-Mail-Header), `message` (Text). Der Schlüssel `legacy` enthält standardmäßig den Wert `true`, damit Serendipity die eigenen Routinen zum Mailversand benutzt. Wenn ein Plugin diese Variable auf `false` ändert, kann es den Mailversand selbständig übernehmen. `addData` enthält den Zeichensatz des Blogs.

Backend-Ereignisse: Plugins und Templates

`backend_plugins_fetchlist`
: kann die Liste der verfügbaren Plugins der Plugin-Verwaltung erweitern, wird hauptsächlich vom *Spartacus*-Plugin (siehe Seite 265) benutzt. `eventData`: Array mit Daten der verfügbaren Plugins.

`backend_plugins_fetchplugin`
: wird aufgerufen, wenn ein bisher noch nicht heruntergeladenes Plugin installiert werden soll. Dies wird hauptsächlich von Spartacus benutzt, um an dieser Stelle das Plugin und dessen Metadaten einzulesen. `eventData`: Array mit Daten des zu installierenden Plugins.

`backend_plugins_new_instance`
: wird beim Installieren eines neuen Plugins aufgerufen. `eventData`: Name des Plugins. `addData`: Positionierung des Plugins (`event`, `left`, `right` ...).

`backend_pluginlisting_header`
: wird oberhalb der Plugin-Verwaltung aufgerufen.

`backend_plugins_event_header`
: wird oberhalb der Auflistung aller verfügbaren Ereignis-Plugins im Backend aufgerufen.

`backend_plugins_sidebar_header`
: wird oberhalb der Auflistung aller verfügbaren Seitenleisten-Plugins im Backend aufgerufen.

`backend_pluginlisting_header_upgrade`
: wird oberhalb der Plugin-Verwaltung aufgerufen, wenn ein Redakteur die Liste aller aktualisierbaren Plugins einsieht.

`backend_pluginconfig_X`
: wird ausgeführt, wenn eine Konfigurationsoption eines Plugins keinem intern angebotenen Datentypen (wie z. B. `string`, `radio` ...) entspricht. Das Plugin kann so eigenständige Konfigurationstypen intern verwalten. `eventData` enthält ein Array mit

den Schlüsseln `config_item` (Name der Konfigurationsoption), `cbag` (*Property-Bag*-Objekt der `introspect_config_item()`-Methode des Plugins), `plugin` (enthält das Plugin-Objekt, dem die Konfigurationsoption entspringt), `value` (enthält den aktuellen Wert der Konfigurationsoption), `bag` (*Property-Bag*-Objekt des Plugins) und `postKey` (Zeichenkette für die Benennung der URL-Formularvariablen).

`backend_template_fetchlist`
: kann die Liste der verfügbaren Templates im Backend erweitern, wird hauptsächlich vom *Spartacus*-Plugin benutzt. `eventData`: Array mit Daten der verfügbaren Templates.

`backend_template_fetchtemplate`
: wird aufgerufen, wenn ein bisher noch nicht heruntergeladenes Template installiert werden soll. Dies wird hauptsächlich von Spartacus benutzt, um an dieser Stelle das Template und dessen Metadaten einzulesen.

`backend_templates_configuration_top`
: wird am Anfang der Konfigurationsoptionen für ein Template ausgeführt, falls ein Template die Variable `$template_config` belegt hat (siehe Seite 499). `eventData` enthält ein Array mit den Konfigurationsoptionen des Templates.

`backend_templates_configuration_bottom`
: wird am Ende der Konfigurationsoptionen für ein Template ausgeführt, falls ein Template die Variable `$template_config` belegt hat (siehe Seite 499). `eventData` enthält ein Array mit den Konfigurationsoptionen des Templates.

`backend_templates_configuration_none`
: wird ausgeführt, wenn das aktuell gesetzte Template keine eigenständigen Template-Optionen anbietet.

10.9 Shared Installation

Serendipity unterstützt die Möglichkeit, die Serendipity-PHP-Dateien nur einmalig auf einem Webserver hochzuladen und anhand eines zentralen Verzeichnisses mehrere, vollständig unabhängige Blogs zu betreiben.

Dabei dient das Zentralverzeichnis ähnlich wie bei einer *PEAR*-Installation lediglich als Funktionsarchiv. Die Blog-Installationen greifen auf diese Dateien zurück, um das Framework und notwendige Funktionen zu laden, verwalten sich aber mittels eigener Datenbanktabellen vollständig selbst.

Diese Methode dient sozusagen als Notbehelf dafür, dass Serendipity über seine Verwaltungsoberfläche jeweils nur ein einzelnes Blog konfigurieren kann. Zwar lässt sich über den Umweg von individualisierten Unter-Kategorien des Blogs (bei Verwendung des Plugins *Erweiterte Eigenschaften von Kategorien*, siehe Seite 262) etwas Ähnliches erreichen, doch bei

dieser Methode existiert stets nur eine physikalische Installation Serendipitys, die nur auf eine einzelne Datenbank zugreift.

Eine Serendipity-Installation benötigt für die Einrichtung nur folgende individuelle Dateien und Verzeichnisse:

- Datei `serendipity_config_local.inc.php` mit den Zugangsdaten zur Datenbank,
- Datei `.htaccess` mit der Einrichtung der Verzeichnisnamen,
- Verzeichnis `archives` und `templates_c` zum Schreiben von Temporärdateien,
- Verzeichnis `uploads` zum Speichern von Medien.

Alle weiteren Dateien und Verzeichnisse können aus dem zentralen Verzeichnis eingebunden werden. Dies hat den großen Vorteil, dass Sie die Serendipity-Dateien zentral warten können. Bei einem Update müssen Sie nur an einer Stelle neue Dateien hochladen und aktualisieren damit automatisch alle weiteren Blogs.

10.9.1 Beispiel

Am einfachsten lässt sich die Einrichtung mehrerer Blogs auf Basis der *Shared Installation* mit einem Beispiel verdeutlichen.

Unser Ziel soll sein, drei Blogs einzurichten. Zwei dieser Blogs sollen auf einer Domain laufen und über `http://blogs.seniorgamer.de/shooter/` und `http://blogs.seniorgamer.de/` aufgerufen werden. Das dritte Blog befindet sich unter `http://www.percanat-blog.de/`.

Jede Domain besitzt einen sogenannten `Document Root`. In diesem Stammverzeichnis befinden sich die Dateien der jeweiligen Domain. In unserem Beispiel haben wir FTP-Zugriff auf folgende Verzeichnisse:

`/home/www/seniorgamer.de/htdocs/`
: für Blog 1 und 2

`/home/www/percanat-blog.de/htdocs/`
: für Blog 3

In diese Verzeichnisse kopieren wir später das Grundgerüst von Serendipity, die sogenannten *Deployments*.

1. Einrichtung des Core

Der erste Schritt ist die Einrichtung des Serendipity-Zentralverzeichnisses, des sogenannten *Core*. Am sinnvollsten ist es, dieses Verzeichnis in einem zentralen Server-Verzeichnis, ähnlich wie *PEAR*, einzurichten. Oft haben Sie auf einem gehosteten Server aber keinen Zugriff auf ein Verzeichnis wie `/usr/local/lib/php`, daher können Sie das Server-Verzeichnis auch anderweitig einrichten.

Wichtig ist lediglich, dass alle *Deployments* später Lesezugriffsrechte zu diesem Ordner besitzen. Wenn Ihr Server also den Zugriff auf außerhalb liegende Verzeichnisse mittels `open_basedir`-PHP-Anweisungen oder Ähnlichem beschränkt, ist die Methode der *Shared Installation* für Sie nicht anwendbar.

In unserem Beispiel wollen wir einen Mittelweg gehen. Wir ernennen das Verzeichnis `/home/www/seniorgamer.de/htd` als Ziel unseres *Core*, da wir dort sowieso Unterverzeichnisse für zwei der drei Blogs anlegen werden.

Wir entpacken also nun das übliche Serendipity-Installationsverzeichnis unter `/home/www/seniorgamer.de/htdocs/S` Sobald Sie die Dateien hochgeladen haben, müssen Sie keine weitere Installation vornehmen. Nur *Deployments* müssen installiert werden – der *Core* selbst enthält keine Zugangsdaten zu einer Datenbank oder Weiteres.

2. Einrichtung der Deployments

Nun müssen wir die *Deployments* einrichten. Serendipity verfügt über eine kleine Sammlung an Basisdateien, die wir benötigen.

Dazu erstellen wir vorerst ein leeres Blog-Verzeichnis in jedem der Zielverzeichnisse:

```
/home/www/seniorgamer.de/htdocs/shooter/
/home/www/seniorgamer.de/htdocs/mmorpg/
/home/www/percanat-blog/htdocs/
```

In jedes dieser drei leeren Verzeichnisse kopieren wir nun den vollständigen Inhalt des Core-Verzeichnisses `/home/www/seniorgamer.de/htdocs/Serendipity/deployment`. Danach sollten Sie in jedem der drei Verzeichnisse eine Liste wie folgende besitzen:

```
/home/www/seniorgamer.de/htdocs/shooter/comment.php
/home/www/seniorgamer.de/htdocs/shooter/exit.php
/home/www/seniorgamer.de/htdocs/shooter/index.php
/home/www/seniorgamer.de/htdocs/shooter/rss.php
/home/www/seniorgamer.de/htdocs/shooter/serendipity_admin_image_selector.
php
/home/www/seniorgamer.de/htdocs/shooter/serendipity_admin.php
/home/www/seniorgamer.de/htdocs/shooter/serendipity_config.inc.php
/home/www/seniorgamer.de/htdocs/shooter/serendipity.css.php
/home/www/seniorgamer.de/htdocs/shooter/serendipity_define.js.php
```

```
/home/www/seniorgamer.de/htdocs/shooter/serendipity_editor.js
/home/www/seniorgamer.de/htdocs/shooter/serendipity_xmlrpc.php
/home/www/seniorgamer.de/htdocs/shooter/wfwcomment.php
/home/www/seniorgamer.de/htdocs/shooter/comment.php
```

Die Bedeutung der einzelnen Dateien ist auf Seite 595 ausführlich beschrieben. Wenn Sie sich diese Dateien einmal in einem Editor ansehen, werden Sie feststellen, dass die Dateien lediglich inhaltsleere Verweise sind, die die eigentlichen Serendipity-Core-Dateien einbinden:

```
<?php # $Id:$
# Copyright (c) 2003-2005, Jannis Hermanns (on behalf the Serendipity
# Developer Team)
# All rights reserved.  See LICENSE file for licensing details
# Serendipity is provided in managed mode here

require_once 's9y/index.php';
/* vim: set sts=4 ts=4 expandtab : */
?>
```

3. Für Einbindung des Core sorgen

Wie man anhand des Code-Grundgerüsts sehen kann, gehen die standardmäßig gelieferten *Deployments* davon aus, dass Sie den Serendipity-Core in einem Verzeichnis namens `s9y` abgelegt haben. Wenn in einem PHP-`include/require`-Befehl kein voller Verzeichnisfad zu einer Datei angegeben wird, bezieht PHP diesen Pfad automatisch anhand der Konfigurationseinstellung `include_path`. Meist enthalten Web-Server für diesen Wert lediglich das Verzeichnis `/usr/local/lib/php`, wo auch *PEAR* abgelegt wird.

Wenn wir auf unserem Server Zugriff zu einem im `include_path` enthaltenen Verzeichnis hätten und dort Serendipity in ein `s9y`-Unterverzeichnis legen würden, wäre die Einbindung des Core in unseren Deployments problemlos möglich.

In unserem konstruierten Beispiel fehlt uns der Zugriff auf ein derartiges Verzeichnis, daher müssen wir die Dateien der *Deployments* leicht abändern. Da wir Serendipity in den Pfad `/home/www/seniorgamer.de/htdocs/Serendipity/` kopiert haben, müssen wir nun diesen Pfad in allen PHP-Scripts des obigen Verzeichnisses eintragen.

In allen Dateien der *Deployments* ändern Sie daher die Zeilen wie

```
require_once 's9y/index.php';
```

in

```
require_once '/home/www/seniorgamer.de/htdocs/Serendipity/index.php';
```

ab. Wenn Sie die Möglichkeit haben, wäre eine Alternative zu diesem Schritt selbstverständlich einfach das korrekte Setzen des PHP-`include_path` auf dieses Verzeichnis. Achten Sie in diesem Fall aber darauf, dass Ihr zentrales Verzeichnis dann nicht `Serendipity` lauten darf, sondern wie durch die Dateien des Deployments festgelegt `s9y`.

4. Weitere Verzeichnisse anlegen

Jedes Verzeichnis mit den *Deployments* benötigt nun noch einige Kleinigkeiten.

Zum einen müssen die drei (leeren) Verzeichnisse `upload`, `templates_c` und `archives` existieren und Schreibrechte für PHP (z. B. 0777) besitzen. Für eine fehlerfreie Installation muss das jeweilige Stammverzeichnis ebenfalls Schreibrechte besitzen, da andernfalls die beiden Dateien `.htaccess` und `serendipity_config_local.inc.php` nicht erstellt werden könnten.

Um Plugins zu installieren, kann jedes *Deployment* standardmäßig auf die Plugins des *Core* zurückgreifen. Wenn ein Blog jedoch nur auf bestimmte Plugins zugreifen können soll, ist es empfehlenswert, das Verzeichnis `/home/www/seniorgamer.de/htdocs/Serendipity/plugins` in das Verzeichnis des jeweiligen *Deployments* zu übernehmen. Um Redundanz zu vermeiden, wäre es am besten, dieses Verzeichnis symbolisch zu verlinken. Wenn Ihr Server/Provider dies nicht unterstützt, müssen Sie das komplette `plugins`-Verzeichnis einfach kopieren.

Dasselbe gilt für das Verzeichnis `templates`. Dieses müssen Sie auch entweder symbolisch vom *Core*-Verzeichnis in das jeweilige *Deployment* verknüpfen oder kopieren.

Wenn Sie den internen WYSIWYG-Editor von Serendpity verwenden wollen, müssen Sie als Letztes auch das Verzeichnis `htmlarea` übernehmen.

5. Installation der Deployments ausführen

Nun sind alle Deployments entsprechend vorbereitet und können installiert werden. Rufen Sie dazu die URL wie `http://blogs.seniorgamer.de/shooter/serendipity_admin.php` auf und folgen Sie dem Serendipity-Installationsprozess. Dabei wird eine eigenständige Konfigurationsdatei im *Deployment*-Verzeichnis und eine eigenständige Datenbank erstellt. Falls Sie alle Deployments in eine einzelne Datenbank installieren, sollten Sie die *Forgeschrittene Installation* wählen und unterschiedliche Tabellenpräfixe nutzen.

Wenn Sie zusätzliche Demo-Inhalte in ein Deployment einstellen wollen, können Sie die Dateien `sql/preload.sql` mit entsprechenden SQL-Anweisungen füllen. Wenn Ihr Deployment über die Standardsettings von Serendipity hinaus bestimmte Plugins vorinstallieren soll, können Sie eine Datei `plugins/preload.txt` mit den Namen der Plugin-Klassen (z. B. `serendipity_event_nl2br`) füllen. Trennen Sie mehrere Plugins mit einem Zeilenumbruch voneinander. Nach jedem Plugin müssen Sie die Platzierung des Plugins eintragen (`left`, `right` oder `event`):

```
serendipity_event_nl2br:event
```

```
serendipity_plugin_history:left
...
```

6. Betrieb der Deployments

Alle Deployments sind nun aufgrund ihrer eigenen Datenbanktabellen und Konfigurationsdateien unabhängig voneinander. Über die jeweilige `serendipity_admin.php`-URL können diese Deployments verwaltet werden.

Übrigens können Sie mehrere bestehende *normale* Serendipity-Blogs auf einem Server auch mit geringfügigem Aufwand in *Shared Installations* umwandeln.

Dazu müssen Sie wie oben beschrieben ein Zentralverzeichnis einrichten. Ersetzen Sie die bestehenden PHP-Dateien der bereits bestehenden Blog-Installation durch die Dateien eines *Deployments* und passen Sie wie beschrieben die Pfade in den `.php`-Dateien an. Daraufhin können Sie in den bestehenden Blogs die Verzeichnisse wie `include`, `deployment`, `docs`, `bundled-libs`, `lang` und `sql` löschen; diese werden in Zukunft aus dem *Core-Verzeichnis* bezogen. Die Verzeichnisse `templates` und `plugins` sollten Sie beibehalten, ebenso wie die Konfigurationsdateien `serendipity_config_local.inc.php` und `.htaccess`.[5]

10.9.2 Einsatzgebiete

Shared Installations machen besonders dann Sinn, wenn sie automatisiert angelegt werden. Der obige Vorgang ist zwar nicht übermäßig komplex, aber bereits umfangreich genug, um ein Script für diese Sklavenarbeit einzusetzen.

Provider wie `http://supersized.s9y.org/` setzen auf diese Art der Shared Installation, um freie Serendipity-Blogs für Interessierte zu erstellen. So kann Serendipity dann tatsächlich wie ein `blogger.com`-Ersatz verwendet werden.

Leider sind die Rahmenbedingungen der Server zu unterschiedlich, um den Vorgang einer Shared Installation so zu abstrahieren, dass er sich überall einsetzen lässt. Zugriffsrechte, `open_basedir`- und `include_path`-Konfigurationen sind zu unterschiedlich und meist so individuell, dass ein vollständig automatisiertes Script mehr falsch als richtig machen könnte.

Im CVS von Serendipity steht dennoch eine kleine Script-Sammlung zur Verfügung.[6] Diese Sammlung kann Server-Administratoren dazu dienen, ihre eigenen Scripts zur Erstellung von Shared Installations zu entwickeln.

Häufig macht es nämlich direkt Sinn, eine Nutzerverwaltung anzubinden. Damit könnten aggregierte Blog-Übersichtsseiten erzeugt werden, oder ein zentrales SQL-Datenbank-Interface, um SQL-Abfragen für alle *Shared Blogs* auszuführen.

10.10 Embedding/Eingebettete Nutzung

Häufig wünschen sich Blog-Betreiber, ihr Blog in eine bereits bestehende Webseite einzubinden.

[5] Eine ausführlichere Beschreibung dieses Vorgangs anhand einer tatsächlich ausgeführten Umstellung bietet Falk Döring in seinem Blog unter `http://www.fadoe.de/archives/136-Einzel-Blogs-auf-Shared-Blogs-umstellen-Projektbeschreibung.html` an.

[6] `http://php-blog.cvs.sourceforge.net/php-blog/additional_plugins/setup/s9y_conf/` und `http://www.s9y.org/41.html`.

Diese Anforderung klingt eigentlich recht simpel, ist aber technisch relativ komplex. Das Grundproblem dabei ist, dass Serendipity als eigenständige Anwendung agiert. Es sendet vollständige HTML-Seiten, HTTP-Kopfzeilen und benötigt URL-Parameter, um zu bestimmen, welche Inhalte angezeigt werden sollen.

Die einfachste Form der Einbindung von Serendipity in eine bestehende Seite wäre die Erstellung eines *framesets* oder *iframes* innerhalb des HTML-Codes Ihrer Seite. So wird Serendipity für den Besucher im Rahmen Ihrer Seite eingebunden, ist aber technisch weiterhin vollständig losgelöst von Ihrer Webseite.

Dies ist natürlich nur die halbe Miete und bringt zahlreiche Probleme mit sich: Lesezeichen können bei Frames nicht gut verarbeitet werden, Suchmaschinen und sehbehinderte Menschen können weniger gut darauf zugreifen, und auch das Design kann durch den Einsatz von Frames durcheinander geraten.

10.10.1 Die Wrapper-Methode

Bevor Serendipity mit Smarty-Templates umgehen konnte, wurde der *Embed Mode* aus der Taufe gehoben. Diesen Modus können Sie über die Konfiguration Serendipitys aktivieren (Abschnitt **Konfiguration** → **Design und Optionen** → **Eingebettete Nutzung von Serendipity aktivieren**). Er sorgt dafür, dass die Seiten von Serendipity selbst keine HTML-Kopf- und Fußzeilen ausgeben und somit einfacher in eine fremde Seite eingebunden werden können. Der Serendipity-Inhalt dient in so einem Fall also lediglich einem HTML-Konstrukt, das auf einer vollständigen Seite eingebunden werden muss. Wenn man ein Serendipity-Blog mit aktiviertem *Embed Mode* direkt aufruft, sieht man eine unformatierte HTML-Seite.

Serendipity sollte im *Embed Mode* daher nicht mehr direkt aufgerufen werden, sondern über den Umweg eines *Wrappers*. Ein Wrapper ist eine Art Container, der Serendipity und Ihre Webseite verkoppelt. Dafür bietet Serendipity die Option **Konfiguration** → **Pfade** → **Index-Datei** an, mit der Sie festlegen, welche Datei von Serendipity zur Darstellung des Blogs angesprochen wird.

Wenn Sie einen Wrapper einsetzen wollen, müssen Sie an dieser Stelle einen alternativen Dateinamen vergeben. Hierfür empfiehlt sich z. B. ein Dateiname wie `wrapper.php`. Diese Datei muss von Ihnen im folgenden Schritt erstellt werden, und es ist sehr wichtig, dass diese Datei im selben Verzeichnis wie Serendipity liegt. Falls Sie mittels einer Angabe wie `../wrapper.php` tricksen , wird dies die Permalink-Behandlung von Serendipity durcheinanderbringen, und viele Seiten des Frontends werden nicht mehr funktionieren.

Die `wrapper.php`-Datei muss nun dafür sorgen, den Serendipity-Inhalt und den Inhalt Ihrer Webseite zu *verheiraten*. Ein derartiges Script kann wie folgt aussehen:

```
<?php
ob_start();
include 'index.php';
$blog = ob_get_contents();
ob_end_clean();
```

```
include '../homepage.php';
?>
```

Dieses Script sorgt dafür, alle Serendipity-Ausgaben in die Variable `$blog` zu speichern. Danach bindet es eine Datei wie `../homepage.php` ein, die den üblichen Code Ihrer Webseite beinhaltet. Ob dies ein PHP-Script oder einfacher HTML-Code ist, ist eigentlich egal. Sie müssen lediglich an der Stelle Ihrer Homepage, wo Sie die Inhalte des Blogs sehen wollen, die Variable `$blog` ausgeben.

Wie Sie sehen, bindet das Wrapper-Script Dateien aus einem Unterverzeichnis (`../`) ein. Sie sollten bei der Erstellung Ihrer Homepage also möglichst darauf achten, dass alle Ihre Links relativ zum Stammverzeichnis der Homepage angelegt sind, und nicht relativ zum aktuellen Verzeichnis – denn sonst könnten durch den Einsatz des Wrappers die Links Ihrer `homepage.php` nicht mehr zum richtigen Ziel zeigen.

Grundsätzlich können Sie anstelle von `homepage.php` auch jedes andere PHP-Framework einbinden, solange Sie die `$blog`-Variable entsprechend ausgeben. Theoretisch können Sie auch zuerst Ihr PHP-Framework einbinden und erst danach wieder ins Serendipity-Verzeichnis wechseln, die Variable `$blog` zusammenstellen lassen und später ausgeben. Dies hängt allein von dem auf Ihrer Seite benötigten Workflow ab.

Wichtig ist, dass Ihre einbindende Webseite selbständig alle HTML-Kopfzeilen ausgibt und auch alle eventuell benötigten Stylesheets von Serendipity einbindet.

10.10.2 Die Smarty-Methode

Seit Serendipity das *Smarty*-Templating unterstützt, gehört der umständliche Umweg über eine Wrapper-Datei eher zum alten Eisen.

Mittels der `index.tpl`-Template-Datei Ihres Blog-Templates können Sie bereits über die Smarty-Funktion `include_php` oder `include_file` anderen HTML-Code Ihrer Webseite einbinden. Wenn Sie also bereits eigene *Header/Footer*-Dateien haben, können Sie diese ebenfalls ganz einfach im Smarty-Template einbinden.

So müssen Sie sich nur an einer zentralen Stelle um das HTML-Groblayout kümmern und nicht extra Ihre bereits bestehenden PHP-Header/Footer mühsam in ein Serendipity-Template gießen.

Sie können sogar eigene PHP-Klassen und -Funktionen über den Umweg der Smarty-Funktionen gehen. Dazu können Sie die Datei `config.inc.php` Ihres Templates bearbeiten. Sämtlicher dort enthaltener PHP-Code wird vor der Ausgabe des Templates von Serendipity ausgeführt und kann so auf ein Framework Ihrer Webseite zugreifen.

Bedingung für die Fremdeinbindung von PHP-Code ist, dass Sie in der Datei `config.inc.php` (siehe auch Seite 500) die Zeile `$serendipity['smarty']->security = false;` einbinden. Damit wird die Ausführungssicherheit deaktiviert, und Ihnen stehen alle PHP-Funktionen zur Verfügung.

10.10.3 Das Serendipity-Framework nutzen

Die eingangs vorgestellte *Wrapper*-Lösung basierte darauf, dass Serendipity als Erstes aufgerufen wird und dann Ihre Webseite einbindet.

Grundsätzlich können Sie Serendipity auch von einem PHP-Framework aus einbinden. Dazu reicht ein PHP-Code wie dieser:

```
<?php
// Ihr eigener Framework-Code befindet sich hier
...

// Speichern des aktuellen Verzeichnisses
$current = getcwd();

// Zum Serendipity-Verzeichnis wechseln
chdir('/home/www/example.com/serendipity/');

// Serendipity-Framework einbinden
include 'serendipity_config.inc.php';

// Serendipity-Smarty-Framework starten
serendipity_smarty_init();

// Liste der aktuellsten Blog-Artikel holen.
$entries = serendipity_fetchEntries(null, true, 10);

// Einträge formatieren
```

```
serendipity_printEntries($entries);

// Template-Datei zur Darstellung einlesen
$tpl = serendipity_getTemplateFile('entries.tpl', 'serendipityPath');

// Template darstellen
serendipity['smarty']->display($tpl);

// Zurück zu Ihrem Framework wechseln
chdir($current);

// hier Ihr etwaiger weiterer Framework-Code
...
```

Grundsätzlich können Sie nach Einbindung des Serendipity-Frameworks auf alle Funktionen zurückgreifen, die in Kapitel 10.5.1 ab Seite 600 aufgeführt sind.

10.11 Externe Schnittstellen zur Benutzerauthentifikation

Serendipitys Framework ist darauf ausgelegt, auf seine eigenen Datenbanktabellen zugreifen zu können. Daher ist eine Integration einer fremden Benutzerdatenbank oder Rechteverwaltung nicht ohne Weiteres möglich.

Häufig kommt der Wunsch auf, dass Sie eine Webseite z. B. mit einem Forum und einem Blog betreiben. Beide Komponenten liefern eine Benutzerverwaltung, und Sie müssten sich jeweils separat in beiden Anwendungen einloggen, um Zugriff zu erhalten.

Da die Benutzerdatenbanktabellen in vielen Serendipity-SQL-Abfragen zum Tragen kommen, kann man diese leider nicht ohne Weiteres kapseln und mit Fremdanwendungen verkoppeln. Das Gleiche gilt für fremde Forensoftware, die ebenfalls eine eng verzahnte eigene Benutzerdatenbank abbildet.

An dieser Stelle gibt es daher meist nur eine Lösung: Man muss eine der Datenbanktabellen als *Master* deklarieren und alle anderen Datenbanken mit deren Daten befüllen.

10.11.1 LDAP

Als Musterlösung für ein solches Vorgehen bietet Serendipity das Plugin *Externe Benutzer-Authentifizierung (LDAP)* (`serendipity_event_externalauth`) an. Dieses Plugin demonstriert, wie eine fremde LDAP-Benutzerdatenbank eingebunden werden kann.

LDAP steht für *Lightweight Directory Access Protocol* und stellt Mittel einer zentralen Benutzerverwaltung zur Verfügung. Im Gegensatz zu einer SQL-Datenbank besteht ein solches System aus einer beliebigen Baumstruktur mit beliebigen Wertekonstellationen. Daher eignet es sich schlecht für die Verbindung in einem Blog-System, das mit relationalen SQL-

Datenbanken arbeitet.

Um dieses konzeptionelle Problem zu umgehen, geht das Serendipity-Plugin den Umweg über einen sogenannten *Proxy-Mechanismus*. Die zentrale Serendipity-Benutzerdatenbank greift dabei weiterhin auf ihre eigene, gewohnte Datenbanktabelle zu, wenn sich ein Benutzer einloggen will.

Nur wenn der Benutzer in dieser Tabelle nicht gefunden wird, stellt das Plugin eine Verbindung zum *LDAP*-Benutzerverzeichnis her und sucht dort nach dem Benutzer. Wird er dort gefunden, werden alle Login-relevanten Daten ausgelesen und in der Serendipity-Benutzerdatenbank geklont/dupliziert.

Dabei entsteht zwangsläufig Redundanz. Das Plugin versucht diese zumindest in geringem Rahmen zu halten, gleicht die beiden Datenbanktabellen regelmäßig miteinander ab und löscht beispielsweise Benutzeraccounts, die im LDAP-Verzeichnis nicht mehr aufgeführt werden.

Nach einem ähnlichen Schema müssten Sie das Plugin erweitern, um dieselbe Aktion für eine fremde Beutzerdatenbank durchzuführen. Sie brauchen also einen regelmäßigen Synchronisierungsvorgang, der die dedizierte *Master*-Tabelle mit der *Serendipity-Datenbank* abgleicht.

Um zum Beispiel eines fremden Forums zurückzukehren, müssten Sie also die Daten dieser Benutzerdatenbanktabelle regelmäßig mit der von Serendipity abgleichen. Das Plugin könnte dabei auch automatisch entsprechend geteilte PHP-Sessiondaten erstellen.

Dies alles erfordert natürlich eine gewisse Kenntnis der Systemarchitektur der fremden Applikation wie auch der von Serendipity. Daher wird dieser Weg relativ selten beschritten und eher nur von Dienstleistungsagenturen mit guten Systemkenntnissen angeboten. Eine Liste derartiger Dienstleister finden Sie auf der Serendipity-Projektseite unter `http://www.s9y.org/20.html`.

10.11.2 OpenID

Dem Problem der Fremdauthentifikation tritt nicht nur bei Serendipity auf, sondern bei sehr vielen Web-2.0-Anwendungen. Daher hat sich ein Konsortium gegründet, das sich mit der *OpenID*-Initiative[7] das Ziel gesetzt hat, eine einheitliche und dezentrale Benutzerauthentifizierung zur Verfügung zu stellen.

Auch für Serendipity gibt es hierzu bereits ein erstes Plugin, das sich jedoch im täglichen Einsatz noch beweisen muss. Da sich OpenID aufgrund seiner Komplexität generell eher schleppend fortentwickelt, ist jegliche Mitarbeit bei diesem Thema im Serendipity-Forum sehr willkommen.

OpenID kümmert sich derzeit in erster Linie nur um das Problem, Besucher für ihre Kommentare zu authentifizieren. In seiner jetzigen Form unterstützt OpenID kein komplexes Rechtemanagement und kann daher die Verwaltung von Benutzern und Gruppen in Serendipity nicht ersetzen bzw. darauf aufbauen. Für diese Verwaltung werden also nach wie vor zentrale

[7] `http://www.openid.net`

Serendipity-Tabellen benötigt.

Bis auch diese Tabellen durch einen offenen Standard ersetzt werden können, wird noch viel Wasser den Rhein hinunterlaufen – dennoch ist auch hier die Mithilfe bei der Entwicklung nötiger Standards in unser aller Interesse.

10.11.3 MySQL VIEWs

Eine sehr komfortable Lösung zum automatisierten Abgleich einer *Master*-Tabelle mit anderen Fremdtabellen ist der Einsatz von *Views*, die für das *MySQL*-Datenbanksystem zur Verfügung stehen.

Views (*Sichten*) stellen ein Mittel von MySQL (ab Version 5.1) dar, die Ergebnisse einer SQL-SELECT-Abfrage als virtuelle Tabelle zu erstellen. So kann man eine fremde Benutzerdatenbanktabelle so *umbiegen*, dass sie in Wirklichkeit auf eine andere Tabelle zeigt.

Für die folgende Beschreibung gehen wir davon aus, dass Sie mit SQL-Code umgehen und eine SQL-Oberfläche wie `phpMyAdmin` bedienen können.

Gehen wir von einem Forensystem aus, das folgende Benutzertabelle verwendet:

```
CREATE TABLE ‘forum_users` (
  ‘user_id‘ mediumint(8) NOT NULL,
  ‘user_name‘ varchar(30),
  ‘user_password‘ varchar(30),
  PRIMARY KEY (‘user_id‘)
);
```

Im Vergleich dazu die bekannte `serendipity_authors`-Tabelle:

```
CREATE TABLE ‘serendipity_authors‘ (
  ‘authorid‘ int(11) NOT NULL auto_increment,
  ‘realname‘ varchar(255) NOT NULL
  ‘username‘ varchar(20),
  ‘password‘ varchar(32),
  ‘mail_comments‘ int(1) default ’1’,
  ‘mail_trackbacks‘ int(1) default ’1’,
  ‘email‘ varchar(128) NOT NULL,
  ‘userlevel‘ int(4) NOT NULL,
  ‘right_publish‘ int(1) default ’1’,
  PRIMARY KEY  (‘authorid‘)
);
```

Glücklicherweise ist unsere Beispielstruktur hier relativ kompatibel: Das Feld `user_id` kann mit `authorid` verkettet werden, `user_name` mit `realname` und `username`, und `user_password` mit `password`. Wären die Passwörter in einem unterschiedlichen Format gespeichert, müsste man sich mit den MySQL-Funktionen MD5() etc. auseinandersetzen, um ein identisches Format zu erreichen.

Nachdem die Tabellenspalten derart zugeordnet wurden, muss man sich entscheiden, welche Tabelle den *Master* darstellt. Wir wählen Serendipity aus, weil dies über mehr Meta-Daten verfügt und die komplexere Benutzerverwaltung aufweist. Sie sollten stets die Tabelle als *Master* wählen, die die höchste Wichtigkeit in Ihrem Projekt darstellt und die größte Komplexität aufweist.

Die Serendipity-Datenbanktabelle `serendipity_authors` bleibt daher unverändert, und die Tabelle `forum_users` muss nun durch einen *View* virtualisiert werden, damit sie direkt auf die Serendipity-Tabelle zugreift.

Am Ende der Virtualisierung wird das Löschen der Tabelle `forum_users` stehen. Da eine virtuelle Tabelle nicht den Namen einer tatsächlich bestehenden Tabelle besitzen darf, benennen wir als Erstes diese Tabelle um:

```
RENAME TABLE ‘forum_users‘ TO ‘tmp_forum_users‘;
```

Diese Tabelle `tmp_forum_users` dient uns als Sicherheitsbackup. Nun können wir eine SQL-Abfrage ausführen, die uns als Basis der Virtualisierung dient. Ziel dieser Abfrage der `serendipity_authors`-Tabelle ist es, die Ergebnisse in dem Format zu erhalten, wie sie in der `forum_users`-Tabelle stehen würden, damit die Fremdapplikation so weiterarbeiten kann, wie sie es gewohnt ist.

```
SELECT authorid AS user_id,
       username AS user_name,
       password AS user_password
  FROM serendipity_authors
```

Mittels der `X AS Y`-SQL-Syntax können die Spaltennamen der *Master*-Tabelle in die Nomenklatur der Zieltabelle umbenannt werden.

Um nun aus dieser SQL-Abfrage eine *VIEW* zu erstellen, benutzen wir folgende Abfrage:

```
CREATE VIEW forum_users
         AS
     SELECT authorid AS user_id,
            username AS user_name,
            password AS user_password
       FROM serendipity_authors
```

Nach dieser Abfrage steht Ihnen die virtuelle Tabelle `forum_users` zur Verfügung. Dort können Sie auch problemlos weitere `SELECTS` ausführen (wie es die Fremdsoftware gewöhnt ist) und auch neue Datensätze einfügen oder bestehende löschen. Im Hintergrund führt MySQL dann die benötigten Aktionen durch.

Die `SELECT`-Abfrage in unserem Beispiel ist bewusst sehr einfach gehalten. Tatsächlich werden Sie später jedoch sicher komplexere SQL-Statements wie *String Functions* oder *Control Flow Functions*[8] einsetzen. Damit können Sie beispielsweise *IF-Weichen* in einer Abfrage einführen, um verschiedene Datentypen und Tabellenspalten miteinander zu verketten.

Zusätzlich wird der Einsatz von *Stored Procedures* und *Triggers* für Sie von Interesse sein. Wenn bestimmte Aktionen von einer Fremdanwendung an deren Datenbanktabellen ausgeführt werden (Löschen von Gruppen etc.) können Sie mittels eines *Triggers* dafür sorgen, dass eine ähnliche Funktion auch auf die Datenbanktabellen Serendipitys angewendet wird.

10.12 Mediendatenbank

Die Serendipity-Mediendatenbank verfügt abseits von den normal über das Backend zugänglichen Methoden über ein weiteres Feature, das jedoch derzeit nicht von Plugins oder dem Kernsystem unterstützt wird.

Wenn ein Bild über das Mediendatenbank-Popup in einen Blog-Artikel eingebunden wird (siehe Seite 107), sorgt ein JavaScript (aus der Datei `serendipity_editor.js`, Funktion `serendipity_admin_imageselector_done()`) dafür, dass die URL dieses Bildes gemeinsam mit dem entsprechenden HTML-Code in den Beitrag eingebunden wird.

Dies hat den Vorteil, dass beim Auslesen der Datenbank Bildverweise bereits im Artikel enthalten sind, was eine Migration vereinfacht. Plugins können zudem leicht eine Bild-URL aus dem Eintragstext entnehmen und z. B. wie das Lightbox-Plugin (siehe Seite 357) erweitern.

Der Nachteil dabei ist, dass die Bild-URL unveränderbar im Artikel steht. Wenn Sie Ihr Blog einmal auf einen neuen Server laden, sich Ihre Verzeichnisstruktur ändert oder Sie ein Bild umbenennen, wird der HTML-Code auf ein nicht mehr vorhandenes Bild zeigen.

Dies ließe sich nur umgehen, wenn Serendipity anstelle des HTML-Codes nur die ID des Mediendatenbank-Objekts einbinden und dynamisch bei jedem Seitenaufruf die korrekte URL zurückliefern würde. Der erste Schritt hierzu ist für Serendipity 1.1 bereits gemacht

[8] `http://dev.mysql.com/doc/refman/5.1/en/functions.html`

worden, da hier die ID eines Bildes in einem HTML-Kommentar mit ausgeliefert wird. Es existieren jedoch noch keine weiteren Plugins, die diesen Kommentar aufbereiten.

Dennoch bietet die Serendipity-Mediendatenbank intern bereits die Möglichkeit, ein Bild anhand einer ID auszuliefern. Konkret können Sie beispielsweise das Bild mit der ID 42 wie folgt darstellen:

```
http://www.example.com/serendipity/serendipity_admin_image_selector.php?
serendipity[step]=showItem&serendipity[image]=42
```

Dieser Aufruf gibt eine HTML-Seite mit einem Verweis zum Bild und der Darstellung der Metadaten (Titel, Beschreibung ...) des Bildes zurück.

Diese URL können Sie also (bisher nur manuell) auch in Ihre Blog-Einträge einbinden. Dies hat sogar drei weitere beträchtliche Vorteile: Zum einen können beim Aufruf des Bildes Statistiken ausgewertet werden, indem die Verweisquelle (*Referrer*) des Bildes gespeichert wird. Diese Verweise werden in der Darstellung der Eigenschaften eines Objektes mit ausgegeben und dienen Ihnen als Anhaltspunkt, von wo aus auf Ihr Bild verwiesen wird.

Zum anderen bietet diese Art der Einbindung den Vorteil, dass Zugriffsrechte ausgewertet werden können. Zugriffsrechte können auf Verzeichnisebene im Serendipity-Backend vergeben (siehe Seite 138) und entgegen dem direkten Aufruf per URL auch mit dieser Technik ausgewertet werden. So kann dann ein Besucher ohne Kenntnis der direkten URL Ihr Bild nicht aufrufen.

Drittens können Sie anhand weiterer URL-Variablen bei obigem Aufruf steuern, in welcher Größe das Bild zurückgegeben werden soll. Dazu binden Sie das Script direkt als Bildziel in einen HTML-Code wie folgenden ein:

```
<img
src='http://www.example.com/serendipity/serendipity_admin_image_selector
.php?serendipity[step]=showItem&serendipity[image]=42&serendipity[show]=
full&serendipity[resizeWidth]=200' />
```

Dadurch wird das Bild automatisch mit einer Breite von 200 Pixeln skaliert. Alternativ können Sie mittels `serendipity[resizeHeight]` auch die Bildhöhe angeben oder beide Parameter miteinander verbinden. Der URL-Parameter `serendipity[show]` kann folgende Werte haben:

`full`
: zeigt das Originalbild an.

`thumb`
: zeigt das Vorschaubild an.

`redirect`
: führt eine Browser-Weiterleitung zum Originalbild durch.

`redirectThumb`
: führt eine Browser-Weiterleitung zum Vorschaubild durch. Ohne Angabe des Wertes wird eine HTML-Seite mit den Bildinformationen angezeigt.

Diese Art des Aufrufs macht es Serendipity möglich, auch als Bilddatenbank zu einem fremden Content-Management-System genutzt zu werden.

Dabei muss das CMS die Serendipity-Mediendatenbank lediglich per Popup-Fenster z. B. via JavaScript aufrufen. Die URL für dieses Popup lautet `http://www.example.com/serendipity/serendipity_admselector.php` und kann zahlreiche URL-Variablen enthalten, die für eine Übergabe wichtig sind:

`serendipity[only_path]`
: kann einen Pfadnamen enthalten, den das Mediendatenbank-Popup auslesen soll.

`serendipity[htmltarget]`
: enthält die eindeutige ID eines HTML-Elementes, in den der Rückgabewert des Popups geschrieben werden soll. Wenn das CMS z. B. ein Feld wie `<input type='hidden' name='data[foreign_image]' id='id_foreign_image' value='' />` ausgibt, müssen Sie der Mediendatenbank die Variable `serendipity[htmltarget]=id_foreign_image` übergeben. Andernfalls kann das JavaScript keinen Rückgabewert an das CMS liefern.

  Wenn ein Wert für die Variable `serendipity[filename_only]` gesetzt wird, gibt das Popup-Fenster lediglich einen Dateinamen zurück. Ansonsten liefert das Popup den vollständigen HTML-Code zur Darstellung eines Bildes.

Weiterhin muss das fremde CMS dafür sorgen, dass die Datei `serendipity_editor.js` via JavaScript eingebunden wird, damit das Popup auf die öffnende Seite (`parent.window.opener...`) zugreifen kann. Alle zurückgelieferten Werte und angesprochenen JavaScript-Funktionen können Sie nach Belieben über die Template-Dateien `admin/media_choose.tpl` an Ihr CMS anpassen. Suchen Sie in dieser Datei nach dem Einsatz der Funktion `serendipity_imageSelector_done()`.

Es ist Ihrer Art der Einbindung überlassen, ob Serendipity direkt die URL zu einem Bild für die Verwendung in Ihrem CMS-Artikel zurückliefert oder nur eine ID der Datei, die Sie dann später über das CMS automatisiert an die URL zu der oben beschriebenen Bildausgabe weitergeben.

Zwar greift Serendipity auf eine eigene User-Datenbank zurück[9], aber wenn Serendipity auf demselben Server läuft wie das Content-Management-System, können PHP-Sessions bzw. Cookies zur automatischen Anmeldung an Serendipity problemlos durchgereicht werden. Notfalls kann ein fester Login vom CMS auch per URL-Variable an den Aufruf der Serendipity-Mediendatenbank weitergereicht werden: `serendipity_admin_image_selector.php?http_auth_user=`*`benutzername`*`&http_auth_pw=`*`passwort`*.

[9] Diese könnten Sie theoretisch selbst per MySQL5-Views oder Proxy-Tabellen an die User-Datenbank des CMS ankoppeln.

10.13 Importer

Serendipity kann Daten aus zahlreichen Blog-Systemen importieren. Dazu dienen eigene PHP-Klassen, die in Dateien des Verzeichnisses `include/admin/importers` definiert werden.

Die Basisklasse `Serendipity_Import` wird in der Datei `include/admin/import.inc.php` festgelegt. Jede Importerdatei muss eine eigene Klasse von diesem Objekt ableiten. Der jeweilige Klassenname muss dabei `Serendipity_Import_`*`Name`* lauten. Der Dateiname der PHP-Datei mit dieser Klasse kann beliebig gewählt werden, darf aber keine Sonderzeichen enthalten.

Am Ende der PHP-Datei des jeweiligen Importers muss der Name der Klasse mittels

```
return 'Serendipity_Import_Name';
```

zurückgeliefert werden. Wenn Sie einen eigenen Importer für ein neues System erstellen wollen, ist es am einfachsten, als Vorlage eine bestehende Import-Datei heranzuziehen. Die Datei `include/admin/importers/textpattern.inc.php` bietet sich dafür an, da sie am wenigsten unnötige Zusatzabfragen beinhaltet.

Jede Importer-Klasse besitzt drei Klassenvariablen: `$this->info` enthält ein Array mit Informationen zum Importer. Der Array-Schlüssel `software` enthält als Wert den dargestellten Namen der Import-Klasse im Backend. Die Variable `$this->inputFields` enthält ein Array, das alle Konfigurationsoptionen des Plugins bestimmt, die vom Benutzer beim Import angegeben werden müssen. Dabei enthält `$this->data` beim Importvorgang die Eingaben des Benutzers für diese Konfigurationsoptionen.

Jede Import-Klasse muss folgende Methoden implementieren:

`getImportNotes()`
: Diese Methode liefert etwaige Zusatzinfos zum Importer zurück, die im Backend vor dem Import ausgegeben werden sollen. Üblicherweise informiert diese Methode den Benutzer über Einschränkungen oder Besonderheiten des Importers.

`Serendipity_Import_Name()`
: Der Konstruktor der PHP-Klasse muss die Arrays `$this->data` und `$this->inputFields` füllen. Für die Eingabefelder wird dabei ein verschachteltes Array angelegt. Jedes Unter-Array enthält dabei die Definition der Konfigurationsfelder und greift dabei auf folgende Array-Schlüssel zurück:

  `text`
  : enthält die Beschreibung der Konfigurationsoption.

  `type`
  : bestimmt, wie die Konfigurationsoption dargestellt wird. Mögliche Werte: `input` (Eingabefeld), `list` (Ausklappfeld), `bool` (Ja/Nein). Die Typen stimmen mit

denen überein, die auch für die Konfigurationsoptionen des Blogs in der `include/tpl/config_local.inc.php` verfügbar sind (siehe Seite 586).

`name`
: bestimmt den Feldnamen der Konfigurationsoption, um später innerhalb des `$this->data`-Arrays darauf zugreifen zu können.

`default`
: enthält den Standardwert der Option.

`validateData()`
: Überprüft die Benutzereingaben beim Import auf Gültigkeit. Hier können z. B. Datenbankzugänge getestet oder Eingabefelder auf ihren Inhalt geprüft werden.

`getInputFields()`
: Dient als Kapselungsmethode, um das Array `$this->inputFields` zurückzuliefern.

`import()`
: Diese Methode führt den eigentlichen Import durch. In jedem der Importer werden hier die Ursprungsdaten des fremden Systems eingelesen und die Serendipity-Datenbanktabellen entsprechend befüllt.

  Als Erstes werden dabei meist alle Redakteure des fremden Systems ausgelesen und als Serendipity-Autoren hinzugefügt. Danach werden Kategorien, Artikel und Kommentare zugewiesen. Da beim Import die Primärschlüssel des alten Systems für Serendipity neu vergeben werden, wird häufig mit sogenannten `Lookup-Arrays` gearbeitet. Diese speichern die ID eines Objekts (Artikel, Kategorie ...) im alten Blog-System sowie die ID bei Serendipity. So kann später beim Durchlaufen der alten Zuordnungen von Redakteuren zu Einträgen (oder Kategorien zu Einträgen) leicht die neue Zuordnung für Serendipity durchgeführt werden.

Jede Klasse kann weiterhin beliebige eigene PHP-Methoden enthalten, damit Sie für Ihren Importer beliebige Kapselung erreichen können.

Folgende Methoden sind in der `Serendipity_Import`-Klasse definiert und können beliebig eingesetzt werden:

`getCharsets()`
: Hilfsfunktion, um ein Dropdown mit möglichen Zeichensätzen (nativer Zeichensatz, ISO-8859-1, UTF-8) in den Konfigurationsoptionen einzubinden. Die Werte werden dabei anhand der aktuellen Konfiguration des Blogs zusammengestellt und dienen beim Import dazu, Zeichensatzkollisionen zu vermeiden.

`decode(), strtr(), strtrRecursive()`
: Konvertiert eine Zeichenfolge anhand des in der Konfiguration des Importers festgelegten Zeichensatzes in das gewünschte Zielformat des Blogs. Diese Methode greift

auf die Hilfsmethoden `strtr()` und `strtrRecursive()` zu, damit rekursiv auch Arrays mit Zeichenketten konvertiert werden können.

`getTransTable()`
Hilfsmethode, um HTML-Sonderzeichen zurück in native Zeichen umzuwandeln.

`nativeQuery()`
Führt eine Datenbankabfrage in der Datenbank des Zielsystems aus. Die normale `serendipity_db_quer` Funktion operiert weiterhin auf der Serendipity-Datenbank.

10.14 Template Processor/Template API

Serendipity ist grundsätzlich auf den Einsatz mit dem *Smarty*-Template-Framework ausgerichtet. Die Vorteile von Smarty sind eine auch für Nicht-Programmierer erlernbare Syntax, einfache Integration von Zusatzfunktionen und eine hohe Ausführungsgeschwindigkeit. Zudem wird Smarty aktiv weiterentwickelt und verfügt über eine hervorragende Online-Dokumentation und Verbreitung.

Dennoch besteht gerade bei PHP-Entwicklern oft der Wunsch, ohne Umwege eines Template-Frameworks Inhalte ausgeben zu wollen. Daher bietet Serendipity die Möglichkeit an, Templates mit beliebigen eigenen Ausgabestrukturen einzusetzen. An diese Personen richtet sich Serendipitys Template API – Sie sollten PHP gut beherrschen, wenn Sie diese API einsetzen möchten.

Dabei bedient sich Serendipity einer Art *Connector-API*. Dank dem Einsatz von Smarty werden alle Template-Variablen über eine einheitliche API weitergereicht und Template-Dateien mittels weniger Funktionen ausgegeben.

Wenn Sie ein eigenes Template-Framework einsetzen wollen, benötigen Sie lediglich eine Art Emulation der Smarty-Funktionen. Serendipity bietet Ihnen dazu ein Grundgerüst an und liefert beispielhaft zwei Template-Layer mit: `templates/default-php` (PHP-Templates) und `templates/default-xml` (XML/XSLT-Templates).

Eine eigenständige API kann vom Template mit einer `template.inc.php` (ähnlich wie eine `config.inc.php`-Datei) eingebunden werden. Diese PHP-Datei muss die jeweilige Emulator-Klasse laden. `templates/default-php/template.inc.php` tut dies wie folgt:

```
<?php
include_once S9Y_INCLUDE_PATH . 'include/template_api.inc.php';
$GLOBALS['template'] = new serendipity_smarty_emulator_xml();
$GLOBALS['serendipity']['smarty'] =& $GLOBALS['template'];
?>
```

In der ersten Zeile wird die Serendipity-Emulationsklasse geladen, in der zweiten und dritten die Klasse als Ersatz des `$serendipity['smarty']`-Objekts.

Für alle Serendipity-Funktionen bleibt dabei der Zugriff auf das Template-Framework so erhalten, als käme Smarty zum Einsatz. In Wirklichkeit kümmert sich jedoch die Emulationsklasse um die Ausgabe.

Sie können an dieser Stelle jedes beliebige Objekt einsetzen, solange es folgende Methoden implementiert:

`assign()`
: Weist eine Template-Variable zu.

`assign_by_ref()`
: Weist eine Template-Variable als Referenz zu.

`call()`
: Rufen Benutzer in Ihren Templates Smarty-Funktionen auf, dient die Methode `call()` als Zwischenstufe zum Aufruf der tatsächlichen PHP-Funktion.

`display()`
: Gibt den Inhalt einer Template-Datei aus.

`trigger_error()`
: Gibt eine Fehlermeldung aus.

`getdefault()`
: Liefert den Wert einer durch `assign()` gesetzten Template-Variable zurück.

`fetch()`
: Bindet eine Unter-Template-Datei ein und leitet deren Inhalt weiter an eine Template-Variable. Dies wird dazu verwendet, um kleine Template-Dateien wie `comments.tpl` in eine große Datei (`entries.tpl`) einzubinden.

Für PHP und XSLT sind die Beispiel-Emulatorklassen in der Datei `include/template_api.inc.php` enthalten. Diese sollten Sie als Vorlage für etwaige Eigenentwicklungen nutzen.

Das Prinzip der PHP-Emulation ist technisch recht simpel. Sie benötigt anstelle der Smarty-Template-Dateien einfache PHP-Dateien. Während in Smarty-Templates Variablen mittels {`$Variable`} ausgegeben werden, muss dies in PHP mittels `<?php echo $Variable; ?>` erfolgen. Daher müssen für ein PHP-Template alle `.tpl`-Dateien in das entsprechende Format konvertiert werden. Smarty-Modifier und Smarty-Funktionen müssen ebenfalls entsprechend der PHP-Syntax abgeändert werden, wie auch Smarty-Sprachkonstrukte (z. B. {`if...`}`...`{`/if`}).

Beispielhaft wurde das für einige Template-Dateien des Verzeichnisses `templates/default-php/` bereits durchgeführt. Dieses Vorhaben ist jedoch nicht komplettiert, da de facto wenig Nachfrage nach einem PHP-Template-System besteht. Das Smarty-Framework ist bereits so leistungsfähig, dass die Mitglieder der Serendipity-Community dieses System favorisieren und dafür Templates entwickeln.

Die XML-Emulation arbeitet dabei viel simpler. Sie weist einfach alle Template-Variablen einem eigenen XML-Element zu und schachtelt Arrays entsprechend. Die Darstellung derartiger XML-Elemente kann mittels XSLT-Stylesheets oder -Transformationen erfolgen.

Kapitel 11

Die Community

Wenn Sie an an diesem Punkt des Buches angelangt sind, sollten Sie eine Menge über Serendipity gelernt haben.

Sicher wird dieses Buch nicht alle individuellen Fragen klären können. Mit diesen Fragen sollen Sie sich nicht alleingelassen fühlen; Sie sind herzlich eingeladen, der Serendipity-Community einen Besuch abzustatten. Am leichtesten finden Sie uns über das Forum unter `http://board.s9y.org/`.

Dort gibt es einen englisch- und einen deutschsprachigen Teil. Fachlich zielgerichteter geht es meist im englischen Teil zu, während man im deutschen Bereich gut generellere Diskussionen führen kann und mit Anfängerfragen leichter auf Gehör stößt.

Das Grundprinzip einer funktionierenden Community ist, dass Sie nicht nur Fragen stellen, sondern sich auch in andere Diskussionsstränge einbinden. Versuchen Sie, mit Ihrem bereits angeeigneten Wissen anderen zu helfen – denn so, wie Sie nach Hilfe suchen, tun dies auch andere.

Bevor Sie jedoch eine Frage im Forum stellen, sollten Sie Ihre *Hausaufgaben* gemacht haben. Durchsuchen Sie erst das Forum und die Serendipity-Webseite, ob Sie dort nicht schon eine Antwort auf Ihre Frage finden. Ist dies nicht der Fall, sollten Sie Ihr Problem präzise schildern und einen aussagekräftigen Betreff wählen. Versetzen Sie sich in die Lage der Personen, die Ihnen helfen sollen – was würden Sie an Informationen benötigen?

Für die potenziellen Helfer ist meist Folgendes sehr wichtig: Welche Serendipity-Version setzen Sie ein, wie lautet die URL Ihres Blogs, welchen Browser verwenden Sie, was für einen Server (mit PHP- und Datenbankversion) und welche Ereignis-Plugnis setzen Sie ein? Je detaillierter Ihre Auskunft, desto eher können andere Hilfesuchende später Ihren Beitrag nachvollziehen und ihr eigenes Problem lösen.

11.1 Neue Features

Die Entwicklung Serendipitys läuft sehr transparent ab, Beiträge neuer Entwickler werden gerne gesehen. Dabei ist es ganz egal, wobei Sie helfen wollen – Hauptsache, Sie haben Spaß daran und sind motiviert. Sie müssen also kein PHP-Profi sein und können schon mit einfachen Dingen Serendipity verbessern: Sie können anderen über das System erzählen, oder Sie helfen bei der Dokumentation und dem Ausbau des Wiki auf `http://www.s9y.org/`. Auch Template-Entwickler und Designer sind jederzeit sehr willkommen.

Folgende Features stehen auf der Wunschliste der Serendipity-Entwickler – wer bei deren Verwirklichung helfen möchte, ist herzlich eingeladen:

Unit Tests

Mittels PHPUnit[1] können die einzelnen Methoden der Serendipity-Plugin-API und auch einzelne Funktionen des Serendipity-Frameworks durch sogenannte Test-Cases auf Funktionstüchtigkeit geprüft werden.

Test-Cases sollten dabei sämtliche gültigen und ungültigen Wertebereiche jeder einzelnen Funktion beinhalten. Die Tests dienen in erster Linie dazu, dass sich nicht unbemerkt Fehler in zukünftige Serendipity-Versionen einschleichen können.

Leicht kann es passieren, dass ein neues Feature ungewollte Begleiterscheinungen auf andere Stellen des Codes hat. Mithilfe von Unit-Tests kann diese Beeinflussung regelmäßig getestet und notfalls behoben werden. Leider ist die Erstellung dieser Test-Cases eine harte und langwierige Arbeit und erfordert einige Kenntnis der internen Funktionen des Blog-Systems.

Export

Obwohl Serendipity eine recht offene und leicht exportierbare Datenbankstruktur besitzt, ist es wünschenswert, die komplette Datenbasis eines Serendipity-Blogs auf Knopfdruck zu exportieren. Mithilfe dieses Exports sollte man eine Serendipity-Installation leicht auf einen anderen Server und nofalls auf ein anderes System portieren können.

Wünschenswert wäre ein Exportformat, das auch andere Blog-Systeme unterstützen. Der MoveableType-Export beinhaltet gemeinsame Grundfunktionen, geht aber für die erweiterten Features von Serendipity nicht weit genug.

Dokumentation, Übersetzung

Das Buch in Ihren Händen stellt einen großen Fortschritt bei der Dokumentation Serendipitys dar. Jedoch ist dies erst der Anfang, eine detaillierte und aktuelle Dokumentation jedes einzelnen Plugins sollte online zur Verfügung stehen.

[1] `http://www.phpunit.de/`

Darüber hinaus hat auch die Übersetzung der Plugins von und ins Deutsche eine hohe Priorität.

Multi-Blogs

Obwohl Serendipity mehrere Autoren mit unterschiedlichen Rechtestufen unterstützt und mittels erweiterter Plugins auch Unter-Kategorien als eigenständig formatiertes Blog darstellen kann, geht die Multi-Blog-Unterstützung noch nicht weit genug.

Gewünscht ist eine zentrale Oberfläche, mittels derer beliebig viele Unter-Blogs erstellt werden können, die allesamt unterschiedliche Plugins, Templates und Benutzergruppen definieren können.

Dies setzt einige zentrale Datenbankveränderungen voraus sowie eine generelle weitere Abstraktion des Sourcecodes.

Performance-Tuning, Caching

Serendipity achtet bereits (wo es sinnvoll ist) auf bestmögliche Performance. Dennoch ist für die Zukunft ein neuartiges Caching-Konzept vonnöten, ebenso wie die Möglichkeit, komplett statische Seiten erstellen zu können.

PEAR-Integration, Spartacus Mirroring

Das bereits gut funktionierende Spartacus-System zur Verteilung von Plugins und Templates soll noch weiter ausgebaut und möglichst an die PEAR-Channel-Verwaltung angelehnt werden. Dadurch wäre großflächigeres Mirroring möglich, und die Festlegung auf zentrale vertrauenswürdige Server wäre nicht mehr notwendig.

Workflow-Integration

Blog-Artikel folgen in Serendipity derzeit nur einem zweistufigen Konzept: Veröffentlichung und Entwurf. Wünschenswert wäre die Integration eines offenen Workflow-Systems, mit dem ein Artikel diverse selbst definierte Prüfungsstufen durchläuft und Checkpoints mit bestimmten Bedingungen passieren muss.

Statische Seiten, Benutzerrechte

Das Plugin *Statische Seiten* soll neu entwickelt und stärker an die neuen Bedürfnisse der Redakteure angepasst werden. Eine Mehrbenutzer-Fähigkeit mit individuellen Benutzerrechten soll möglich sein, wie auch eine überarbeitete Oberfläche zur einfacheren (lies: hübscheren) Erstellung einer statischen Seite.

Wenn Sie bereits Programmteile von Serendipity angepasst haben, sollten Sie darüber nachdenken, diese der Serendipity-Gemeinschaft zur Verfügung zu stellen. Dies hat mehrere Vorteile: Zum einen sind Ihnen andere Benutzer auf Lebzeiten dankbar, zum anderen stellen Sie so sicher, dass Ihre Änderungen in zukünftigen Versionen von Serendipity enthalten sind und Sie sie nicht bei jedem Update mühsam neu einpflegen müssen. Auch könnte es sein, dass andere Benutzer Feedback zu Ihren Änderungen geben, gemeinsam mit Ihnen Fehler beheben und Ihren Code um neue Funktionen erweitern. So lernen auch Sie dazu und helfen zugleich bei der Verbreitung Serendipitys.

Auch wenn Sie ein Plugin entwickelt haben, würden sich die Entwickler freuen, wenn Sie Ihren Code offenlegen. Je mehr Benutzer Ihr Plugin einsetzen, desto eher werden Sie über Fehler oder denkbare neue Features benachrichtigt. Das gilt auch für Designer von Templates, denn je verbreiteter Ihr Name in einer Open-Source-Community ist, desto eher gelten Sie als Autorität in dem Bereich. Ein hoher Bekanntheitsgrad hat den Vorteil, dass Sie dadurch leichter auf kommerzielle Projekte angesprochen werden und dass man Sie um Entwicklungsarbeiten bitten könnte.

Auch wenn Sie als Privatentwickler tätig sind, macht es sich bei der Bewerbung um eine Programmiererstelle immer sehr gut, wenn man Einsatz in einem Open-Source-Projekt vorweisen kann. Die Erfahrung in der Zusammenarbeit mit einem internationalen Team, gemeinsame Standards und eine öffentlich einsehbare Programmierqualität sind für potenzielle Arbeitgeber von zunehmender Wichtigkeit.

11.2 Tipps für die Programmierung

Grundsätzlich sollten Sie bei der Entwicklung von Code für Serendipity folgende Dinge beachten:

Sourcecode-Verwaltung nutzen

Serendipity wird über öffentlich zugängliche Server gewartet. Der Hauptteil des Quellcodes liegt auf `http://developer.berlios.de/projects/serendipity` und wird mittels Subversion[2] gepflegt. Die zusätzlichen Plugins und Templates werden mittels CVS über den Server `http://www.sf.net/projects/php-blog` gewartet (siehe Seite 637).

Wenn Sie Code-Änderungen veröffentlichen wollen, sollten Sie Ihre Änderungen immer anhand der aktuellsten Sourcecode-Version aus der jeweiligen Quelle durchführen. Anstatt alle geänderten Dateien an die Entwickler zu schicken, sollten Sie nur Ihre geänderten Codezeilen mitschicken; dies erledigt man mittels eines sogenannten `diff`-Formates, das gängige SVN- und CVS-Programme erstellen können.

Wenn Sie erst einmal einige kleine Patches an die Entwickler geschickt haben, können Sie auch nach direktem Dateizugriff zum Serendipity-Code fragen. Um eine gewisse Qualität

[2] `http://subversion.tigris.org/`

des Codes sicherzustellen, wird dies jedoch erst dann genehmigt, wenn man Ihre Vertrauenswürdigkeit garantieren kann. Dass dies etwas strenger kontrolliert wird, kommt Ihnen letztlich nur zugute.

Nur ändern, was man ändern muss

Wenn Sie den Entwicklern einen Patch schicken, mit dem ein Redakteur z. B. neue Kategorien direkt vom Beitragseditor aus erstellen kann, sollte Ihr Patch ausschließlich diese Änderungen beinhalten. Wenn Sie auch noch ein, zwei weitere Änderungen vorgenommen haben, sollten Sie dies nicht in einem einzelnen großen Patch abliefern, sondern lieber zwei oder drei kleinere Patches an die Entwickler weiterleiten. So kann man die einzelnen Dinge gezielter überprüfen und überstrapaziert die Zeit der Tester nicht.

Nur das Notwendigste zu ändern gilt ganz besonders, wenn Sie ein neues Template entwickelt haben. Es wäre nicht gut, wenn in diesem Template Dateien enthalten sind, die im Vergleich zum Standard-Template unverändert sind. Besser wäre es, nur die absolut notwendigen veränderten Dateien zu bündeln.

Coding Style

Alle Dateien sollten stets im Unix-Format (Zeilenumbrüche mit `\n`) gespeichert werden. Einrückungen sollten mit vier Leerzeichen (nicht mit Tabulatoren) vorgenommen werden.

Optionale Einrückungen für IF-Abfragen und Schleifen sollten immer genutzt werden. Grundsätzlich sollten Sie sich bei der Programmierung an die PEAR-Coding-Standards[3] halten.

Versuchen Sie Ihren Code schlank zu halten. Redundante Codeteile sollten Sie durch Auslagerung in eigenständige Funktionen oder Methoden vermeiden. Unnötige Schleifen oder Datenbankabfragen sollten vermieden werden, wenn möglich sollten Sie performance-intensive Dinge über eigenständiges Caching beschleunigen.

Besonderheiten für Templates

Wenn Ihr Template eine `config.inc.php`-Datei beinhaltet, achten Sie darauf, dass diese so wenig wie möglich PHP-Code ausführt. Je größer und komplexer die Datei wird, desto länger dauert der Serendipity-Seitenaufruf.

HTML-Code sollte nach XHTML-1.1-Standard entwickelt werden. Zudem sollte ein Template in allen gängigen Browsern funktionieren, zumindest in Mozilla Firefox und Microsoft Internet Explorer.

Wenn Ihr Template eigene Sprachkonstanten verwendet, lagern Sie diese bitte in `lang_XX.inc.php`-Dateien aus und liefern auch jeweils die korrespondierende Version der Datei im UTF-8-Zeichensatz mit.

[3] `http://www.go-pear.org/manual/en/standards.php`

Besonderheiten für Plugins

Plugins sollten möglichst objektorientiert (wie es die API vorgibt) entwickelt werden. PHP4-Syntax, wo möglich, ist bevorzugt. Wenn gute Gründe dafür sprechen, sind PHP5-Plugins selbstverständlich auch kein Problem.

Plugins sollten nur die Ereignis-Hooks benutzen, die auch absolut notwendig sind. Je weniger Hooks benutzt werden, desto performanter ist ein Plugin.

Wenn ein Plugin externe URLs öffnet, sollten deren Ergebnisse vom Plugin möglichst für einen konfigurierbaren Zeitraum gecached werden. Behalten Sie stets Netzwerkprobleme im Hinterkopf, die nicht dazu führen sollten, dass ein Plugin inoperabel wird.

Setzen Sie referenzierte Variablen ein, wo möglich.

Wenn Sie ein bestehendes Plugin überarbeiten, fügen Sie bitte in dessen Datei `ChangeLog` eine Information über die vorgenommene Änderung hinzu. Dokumentationen für ein Plugin speichern Sie (mit HTML-Syntax) in einer Datei namens `documentation_XX.html`, wobei das `XX` mit dem jeweiligen Sprachkürzel wie `de` zu ersetzen ist.

Falls Ihr Plugin HTML-Code für Buttons und Links im Backend ausgibt, benutzen Sie bitte die `serendipityPrettyButton`-CSS-Klasse. Bei Eingabeboxen nutzen Sie Klassen wie `input_radio, input_checkbox` oder `input_button`.

Falls Ihr Plugin eigene Menüpunkte im Backend ausgibt, geben Sie dies im Ereignis-Hook mit einem Code wie

```
<li class="serendipitySideBarMenuLink">
```

aus. Sollte Ihr Plugin auf fremde Code-Bibliotheken zurückgreifen, prüfen Sie bitte, ob dieser kompatibel mit der gewählten Lizenz Ihres Plugins ist. Standardmäßig werden Serendipity-Plugins unter die BSD-Lizenz gestellt. Wenn Ihr Plugin auf eine externe Bibliothek angewiesen ist, prüfen Sie im Code, ob diese existiert. Das Plugin sollte das Blog nicht unbenutzbar machen, wenn der entsprechende Code fehlt.

Wenn möglich, sollte Ihr Plugin auf viele kleine Dateien verzichten. Kleine Dateien können die Verteilung über das Spartacus-System spürbar verlangsamen.

Plugins sollten nur SQL-Anweisungen benutzen, die von *MySQL*, *SQLite* und *PostgreSQL* interpretiert werden können. Fügen Sie unterschiedliche Varianten der SQL-Abfrage ein, wenn diese auf unterschiedlichen Systemen voneinander abweichen. Wenn das Plugin eine eigene Datenbanktabelle benötigt, sollte diese vom Plugin automatisch mithilfe der Funktion `serendipity_db_schema_import` erstellt werden.

Sicherheit

Achten Sie stets auf die Sicherheit Ihres Codes. Er sollte keine SQL- oder XSS-Injections zulassen oder gar beliebigen PHP-Code von fremden Servern ausführen. Achten Sie darauf,

Werte aus GET/POST-Variablen vor dem Einfügen in die Datenbank mit `serendipity_db_escape_string()` zu behandeln und bei der Ausgabe mit `htmlspecialchars()` zu umgeben.

Wenn ein Plugin administrative Aktionen im Backend ausführt, sollten die HTML-Formulare zum Ausführen der Funktion mittels `serendipityFormToken()` vor XSRF-Angriffen gesichert werden.

Aus anderen Quellen lernen

Wenn Sie Code beisteuern wollen, ist es am einfachsten, sich anzuschauen, was frühere Serendipity-Entwickler geschrieben haben. Schauen Sie sich bestehende Plugins an, nehmen Sie Anleihen an deren Struktur. Und wann immer etwas unklar ist: Fragen Sie ruhig. Niemand beißt die Hand, die uns füttern soll!