Amazon EC2 Container Service

API Reference

API Version 2014-11-13

Amazon EC2 Container Service: API Reference

Table of Contents

Welcome

Amazon EC2 Container Service (Amazon ECS) is a highly scalable, fast, container management service that makes it easy to run, stop, and manage Docker containers on a cluster of EC2 instances. Amazon ECS lets you launch and stop container-enabled applications with simple API calls, allows you to get the state of your cluster from a centralized service, and gives you access to many familiar Amazon EC2 features like security groups, Amazon EBS volumes, and IAM roles.

You can use Amazon ECS to schedule the placement of containers across your cluster based on your resource needs, isolation policies, and availability requirements. Amazon EC2 Container Service eliminates the need for you to operate your own cluster management and configuration management systems or worry about scaling your management infrastructure.

This document was last published on December 9, 2016.

Actions

The following actions are supported:

CreateCluster

Creates a new Amazon ECS cluster. By default, your account receives a `default` cluster when you launch your first container instance. However, you can create your own cluster with a unique name with the `CreateCluster` action.

Request Syntax

```
{
   "clusterName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

clusterName (p. 3)

The name of your cluster. If you do not specify a name for your cluster, you create a cluster named `default`. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

Type: String

Required: No

Response Syntax

```
{
   "cluster": {
      "activeServicesCount": number,
      "clusterArn": "string",
      "clusterName": "string",
      "pendingTasksCount": number,
      "registeredContainerInstancesCount": number,
      "runningTasksCount": number,
      "status": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

cluster (p. 3)

The full description of your new cluster.

Type: Cluster (p. 113) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request creates a cluster called `My-cluster`.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 29
X-Amz-Target: AmazonEC2ContainerServiceV20141113.CreateCluster
X-Amz-Date: 20150429T163840Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "clusterName": "My-cluster"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 16:38:41 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 209
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "cluster": {
    "activeServicesCount": 0,
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/My-cluster",
```

```
    "clusterName": "My-cluster",
    "pendingTasksCount": 0,
    "registeredContainerInstancesCount": 0,
    "runningTasksCount": 0,
    "status": "ACTIVE"
  }
}
```

CreateService

Runs and maintains a desired number of tasks from a specified task definition. If the number of tasks running in a service drops below `desiredCount`, Amazon ECS spawns another copy of the task in the specified cluster. To update an existing service, see UpdateService (p. 105).

In addition to maintaining the desired count of tasks in your service, you can optionally run your service behind a load balancer. The load balancer distributes traffic across the tasks that are associated with the service. For more information, see Service Load Balancing in the *Amazon EC2 Container Service Developer Guide*.

You can optionally specify a deployment configuration for your service. During a deployment (which is triggered by changing the task definition or the desired count of a service with an UpdateService (p. 105) operation), the service scheduler uses the `minimumHealthyPercent` and `maximumPercent` parameters to determine the deployment strategy.

The `minimumHealthyPercent` represents a lower limit on the number of your service's tasks that must remain in the `RUNNING` state during a deployment, as a percentage of the `desiredCount` (rounded up to the nearest integer). This parameter enables you to deploy without using additional cluster capacity. For example, if your service has a `desiredCount` of four tasks and a `minimumHealthyPercent` of 50%, the scheduler may stop two existing tasks to free up cluster capacity before starting two new tasks. Tasks for services that *do not* use a load balancer are considered healthy if they are in the `RUNNING` state; tasks for services that *do* use a load balancer are considered healthy if they are in the `RUNNING` state and the container instance it is hosted on is reported as healthy by the load balancer. The default value for `minimumHealthyPercent` is 50% in the console and 100% for the AWS CLI, the AWS SDKs, and the APIs.

The `maximumPercent` parameter represents an upper limit on the number of your service's tasks that are allowed in the `RUNNING` or `PENDING` state during a deployment, as a percentage of the `desiredCount` (rounded down to the nearest integer). This parameter enables you to define the deployment batch size. For example, if your service has a `desiredCount` of four tasks and a `maximumPercent` value of 200%, the scheduler may start four new tasks before stopping the four older tasks (provided that the cluster resources required to do this are available). The default value for `maximumPercent` is 200%.

When the service scheduler launches new tasks, it attempts to balance them across the Availability Zones in your cluster with the following logic:

- Determine which of the container instances in your cluster can support your service's task definition (for example, they have the required CPU, memory, ports, and container instance attributes).
- Sort the valid container instances by the fewest number of running tasks for this service in the same Availability Zone as the instance. For example, if zone A has one running service task and zones B and C each have zero, valid container instances in either zone B or C are considered optimal for placement.
- Place the new service task on a valid container instance in an optimal Availability Zone (based on the previous steps), favoring container instances with the fewest number of running tasks for this service.

Request Syntax

```
{
   "clientToken": "string",
   "cluster": "string",
   "deploymentConfiguration": {
      "maximumPercent": number,
      "minimumHealthyPercent": number
   },
   "desiredCount": number,
   "loadBalancers": [
```

```
      {
         "containerName": "string",
         "containerPort": number,
         "loadBalancerName": "string",
         "targetGroupArn": "string"
      }
   ],
   "role": "string",
   "serviceName": "string",
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

clientToken (p. 6)

Unique, case-sensitive identifier you provide to ensure the idempotency of the request. Up to 32 ASCII characters are allowed.

Type: String

Required: No

cluster (p. 6)

The short name or full Amazon Resource Name (ARN) of the cluster on which to run your service. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

deploymentConfiguration (p. 6)

Optional deployment parameters that control how many tasks run during the deployment and the ordering of stopping and starting tasks.

Type: DeploymentConfiguration (p. 125) object

Required: No

desiredCount (p. 6)

The number of instantiations of the specified task definition to place and keep running on your cluster.

Type: Integer

Required: Yes

loadBalancers (p. 6)

A load balancer object representing the load balancer to use with your service. Currently, you are limited to one load balancer per service. After you create a service, the load balancer name, container name, and container port specified in the service definition are immutable.

For Elastic Load Balancing Classic load balancers, this object must contain the load balancer name, the container name (as it appears in a container definition), and the container port to access from the load balancer. When a task from this service is placed on a container instance, the container instance is registered with the load balancer specified here.

For Elastic Load Balancing Application load balancers, this object must contain the load balancer target group ARN, the container name (as it appears in a container definition), and the container port to access from the load balancer. When a task from this service is placed on a container instance, the container instance and port combination is registered as a target in the target group specified here.

Type: array of LoadBalancer (p. 130) objects

Required: No

role (p. 6)

The name or full Amazon Resource Name (ARN) of the IAM role that allows Amazon ECS to make calls to your load balancer on your behalf. This parameter is required if you are using a load balancer with your service. If you specify the `role` parameter, you must also specify a load balancer object with the `loadBalancers` parameter.

If your specified role has a path other than `/`, then you must either specify the full role ARN (this is recommended) or prefix the role name with the path. For example, if a role with the name `bar` has a path of `/foo/` then you would specify `/foo/bar` as the role name. For more information, see Friendly Names and Paths in the *IAM User Guide*.

Type: String

Required: No

serviceName (p. 6)

The name of your service. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed. Service names must be unique within a cluster, but you can have similarly named services in multiple clusters within a region or across multiple regions.

Type: String

Required: Yes

taskDefinition (p. 6)

The `family` and `revision` (`family:revision`) or full Amazon Resource Name (ARN) of the task definition to run in your service. If a `revision` is not specified, the latest `ACTIVE` revision is used.

Type: String

Required: Yes

Response Syntax

```
{
   "service": {
      "clusterArn": "string",
      "createdAt": number,
      "deploymentConfiguration": {
         "maximumPercent": number,
         "minimumHealthyPercent": number
      },
      "deployments": [
         {
            "createdAt": number,
            "desiredCount": number,
            "id": "string",
            "pendingCount": number,
            "runningCount": number,
            "status": "string",
            "taskDefinition": "string",
            "updatedAt": number
         }
      ],
      "desiredCount": number,
      "events": [
         {
            "createdAt": number,
            "id": "string",
            "message": "string"
         }
      ],
```

```
      "loadBalancers": [
         {
            "containerName": "string",
            "containerPort": number,
            "loadBalancerName": "string",
            "targetGroupArn": "string"
         }
      ],
      "pendingCount": number,
      "roleArn": "string",
      "runningCount": number,
      "serviceArn": "string",
      "serviceName": "string",
      "status": "string",
      "taskDefinition": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

service (p. 8)

The full description of your service following the create call.

Type: Service (p. 136) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS,

these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example API request creates a service in your default region called `ecs-simple-service`. The service uses the `ecs-demo` task definition and it maintains 10 instantiations of that task.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 87
X-Amz-Target: AmazonEC2ContainerServiceV20141113.CreateService
X-Amz-Date: 20150429T170125Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "serviceName": "ecs-simple-service",
  "taskDefinition": "ecs-demo",
  "desiredCount": 10
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 17:01:27 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 636
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "service": {
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
    "deploymentConfiguration": {
        "maximumPercent": 200,
        "minimumHealthyPercent": 100
    },
    "deployments": [
      {
        "createdAt": 1430326887.362,
        "desiredCount": 10,
        "id": "ecs-svc/9223370606527888445",
        "pendingCount": 0,
        "runningCount": 0,
        "status": "PRIMARY",
        "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-
definition/ecs-demo:1",
        "updatedAt": 1430326887.362
      }
    ],
    "desiredCount": 10,
    "events": [],
    "loadBalancers": [],
```

```
    "pendingCount": 0,
    "runningCount": 0,
    "serviceArn": "arn:aws:ecs:us-east-1:012345678910:service/ecs-simple-
service",
    "serviceName": "ecs-simple-service",
    "status": "ACTIVE",
    "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-definition/
ecs-demo:1"
  }
}
```

DeleteCluster

Deletes the specified cluster. You must deregister all container instances from this cluster before you may delete it. You can list the container instances in a cluster with ListContainerInstances (p. 56) and deregister them with DeregisterContainerInstance (p. 19).

Request Syntax

```
{
   "cluster": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 12)

The short name or full Amazon Resource Name (ARN) of the cluster to delete.

Type: String

Required: Yes

Response Syntax

```
{
   "cluster": {
      "activeServicesCount": number,
      "clusterArn": "string",
      "clusterName": "string",
      "pendingTasksCount": number,
      "registeredContainerInstancesCount": number,
      "runningTasksCount": number,
      "status": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

cluster (p. 12)

The full description of the deleted cluster.

Type: Cluster (p. 113) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterContainsContainerInstancesException
You cannot delete a cluster that has registered container instances. You must first deregister the container instances before you can delete the cluster. For more information, see DeregisterContainerInstance (p. 19).

HTTP Status Code: 400

ClusterContainsServicesException
You cannot delete a cluster that contains services. You must first update the service to reduce its desired task count to 0 and then delete the service. For more information, see UpdateService (p. 105) and DeleteService (p. 15).

HTTP Status Code: 400

ClusterNotFoundException
The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException
The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException
These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request deletes the cluster called `My-cluster`.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 25
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DeleteCluster
X-Amz-Date: 20150429T170952Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "cluster": "My-cluster"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 17:09:54 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 211
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "cluster": {
    "activeServicesCount": 0,
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/My-cluster",
    "clusterName": "My-cluster",
    "pendingTasksCount": 0,
    "registeredContainerInstancesCount": 0,
    "runningTasksCount": 0,
    "status": "INACTIVE"
  }
}
```

DeleteService

Deletes a specified service within a cluster. You can delete a service if you have no running tasks in it and the desired task count is zero. If the service is actively maintaining tasks, you cannot delete it, and you must update the service to a desired task count of zero. For more information, see UpdateService (p. 105).

Note
When you delete a service, if there are still running tasks that require cleanup, the service status moves from `ACTIVE` to `DRAINING`, and the service is no longer visible in the console or in ListServices (p. 59) API operations. After the tasks have stopped, then the service status moves from `DRAINING` to `INACTIVE`. Services in the `DRAINING` or `INACTIVE` status can still be viewed with DescribeServices (p. 38) API operations; however, in the future, `INACTIVE` services may be cleaned up and purged from Amazon ECS record keeping, and DescribeServices (p. 38) API operations on those services will return a `ServiceNotFoundException` error.

Request Syntax

```
{
   "cluster": "string",
   "service": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 15)
The name of the cluster that hosts the service to delete. If you do not specify a cluster, the default cluster is assumed.
Type: String
Required: No

service (p. 15)
The name of the service to delete.
Type: String
Required: Yes

Response Syntax

```
{
   "service": {
      "clusterArn": "string",
      "createdAt": number,
      "deploymentConfiguration": {
         "maximumPercent": number,
         "minimumHealthyPercent": number
      },
      "deployments": [
         {
```

```
            "createdAt": number,
            "desiredCount": number,
            "id": "string",
            "pendingCount": number,
            "runningCount": number,
            "status": "string",
            "taskDefinition": "string",
            "updatedAt": number
         }
      ],
      "desiredCount": number,
      "events": [
         {
            "createdAt": number,
            "id": "string",
            "message": "string"
         }
      ],
      "loadBalancers": [
         {
            "containerName": "string",
            "containerPort": number,
            "loadBalancerName": "string",
            "targetGroupArn": "string"
         }
      ],
      "pendingCount": number,
      "roleArn": "string",
      "runningCount": number,
      "serviceArn": "string",
      "serviceName": "string",
      "status": "string",
      "taskDefinition": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

service (p. 15)

The full description of the deleted service.

Type: Service (p. 136) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

ServiceNotFoundException

The specified service could not be found. You can view your available services with ListServices (p. 59). Amazon ECS services are cluster-specific and region-specific.

HTTP Status Code: 400

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example API request deletes the `test` service from the `default` cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 19
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DeleteService
X-Amz-Date: 20150429T172539Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "service": "test"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 17:25:40 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 13590
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "service": {
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
    "deploymentConfiguration": {
```

```
        "maximumPercent": 200,
        "minimumHealthyPercent": 100
    },
    "deployments": [
      {
        "createdAt": 1430320735.285,
        "desiredCount": 0,
        "id": "ecs-svc/9223370606534040511",
        "pendingCount": 0,
        "runningCount": 0,
        "status": "PRIMARY",
        "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-
definition/sleep360:27",
        "updatedAt": 1430320735.285
      }
    ],
    "desiredCount": 0,
    "events": [],
    "loadBalancers": [],
    "pendingCount": 0,
    "runningCount": 0,
    "serviceArn": "arn:aws:ecs:us-east-1:012345678910:service/test",
    "serviceName": "test",
    "status": "DRAINING",
    "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-definition/
sleep360:27"
  }
}
```

DeregisterContainerInstance

Deregisters an Amazon ECS container instance from the specified cluster. This instance is no longer available to run tasks.

If you intend to use the container instance for some other purpose after deregistration, you should stop all of the tasks running on the container instance before deregistration to avoid any orphaned tasks from consuming resources.

Deregistering a container instance removes the instance from a cluster, but it does not terminate the EC2 instance; if you are finished using the instance, be sure to terminate it in the Amazon EC2 console to stop billing.

> **Note**
> If you terminate a running container instance, Amazon ECS automatically deregisters the instance from your cluster (stopped container instances or instances with disconnected agents are not automatically deregistered when terminated).

Request Syntax

```
{
   "cluster": "string",
   "containerInstance": "string",
   "force": boolean
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 19)
The short name or full Amazon Resource Name (ARN) of the cluster that hosts the container instance to deregister. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstance (p. 19)
The container instance ID or full Amazon Resource Name (ARN) of the container instance to deregister. The ARN contains the `arn:aws:ecs` namespace, followed by the region of the container instance, the AWS account ID of the container instance owner, the `container-instance` namespace, and then the container instance ID. For example, `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:container-instance/`*`container_instance_ID`*.

Type: String

Required: Yes

force (p. 19)
Forces the deregistration of the container instance. If you have tasks running on the container instance when you deregister it with the `force` option, these tasks remain running until you terminate the instance or the tasks stop through some other means, but they are orphaned (no longer monitored or accounted for by Amazon ECS). If an orphaned task on your container instance is part of an Amazon ECS service, then the service scheduler starts another copy of that task, on a different container instance if possible.

Any containers in orphaned service tasks that are registered with a Classic load balancer or an Application load balancer target group are deregistered, and they will begin connection draining according to the settings on the load balancer or target group.

Type: Boolean

Required: No

Response Syntax

```
{
   "containerInstance": {
      "agentConnected": boolean,
      "agentUpdateStatus": "string",
      "attributes": [
         {
            "name": "string",
            "value": "string"
         }
      ],
      "containerInstanceArn": "string",
      "ec2InstanceId": "string",
      "pendingTasksCount": number,
      "registeredResources": [
         {
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "remainingResources": [
         {
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "runningTasksCount": number,
      "status": "string",
      "version": number,
      "versionInfo": {
         "agentHash": "string",
         "agentVersion": "string",
         "dockerVersion": "string"
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

containerInstance (p. 20)

The container instance that was deregistered.

Type: ContainerInstance (p. 121) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request deregisters a container instance with the ID `f4292606-fbed-4b53-833b-92cad7c687c2` in the `default` cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 61
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DeregisterContainerInstance
X-Amz-Date: 20151001T191224Z
User-Agent: aws-cli/1.8.7 Python/2.7.9 Darwin/14.5.0
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "containerInstance": "c9c9a6f2-8766-464b-8805-9c57b9368fb0"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Thu, 01 Oct 2015 19:12:25 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1613
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "containerInstance": {
    "agentConnected": true,
    "attributes": [
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      }
    ],
    "containerInstanceArn": "arn:aws:ecs:us-west-2:012345678910:container-
instance/c9c9a6f2-8766-464b-8805-9c57b9368fb0",
    "ec2InstanceId": "i-0c3826c9",
    "pendingTasksCount": 0,
    "registeredResources": [
      {
        "doubleValue": 0,
        "integerValue": 1024,
        "longValue": 0,
        "name": "CPU",
        "type": "INTEGER"
      },
      {
        "doubleValue": 0,
        "integerValue": 995,
        "longValue": 0,
        "name": "MEMORY",
        "type": "INTEGER"
      },
      {
        "doubleValue": 0,
        "integerValue": 0,
        "longValue": 0,
        "name": "PORTS",
        "stringSetValue": [
```

```
          "22",
          "2376",
          "2375",
          "51678"
        ],
        "type": "STRINGSET"
      },
      {
        "doubleValue": 0,
        "integerValue": 0,
        "longValue": 0,
        "name": "PORTS_UDP",
        "stringSetValue": [],
        "type": "STRINGSET"
      }
    ],
    "remainingResources": [
      {
        "doubleValue": 0,
        "integerValue": 1024,
        "longValue": 0,
        "name": "CPU",
        "type": "INTEGER"
      },
      {
        "doubleValue": 0,
        "integerValue": 995,
        "longValue": 0,
        "name": "MEMORY",
        "type": "INTEGER"
      },
      {
        "doubleValue": 0,
        "integerValue": 0,
        "longValue": 0,
        "name": "PORTS",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678"
        ],
        "type": "STRINGSET"
      },
      {
        "doubleValue": 0,
        "integerValue": 0,
        "longValue": 0,
        "name": "PORTS_UDP",
        "stringSetValue": [],
        "type": "STRINGSET"
      }
    ],
    "runningTasksCount": 0,
    "status": "INACTIVE",
    "versionInfo": {
      "agentHash": "b197edd",
      "agentVersion": "1.5.0",
      "dockerVersion": "DockerVersion: 1.7.1"
```

```
    }
  }
}
```

DeregisterTaskDefinition

Deregisters the specified task definition by family and revision. Upon deregistration, the task definition is marked as `INACTIVE`. Existing tasks and services that reference an `INACTIVE` task definition continue to run without disruption. Existing services that reference an `INACTIVE` task definition can still scale up or down by modifying the service's desired count.

You cannot use an `INACTIVE` task definition to run new tasks or create new services, and you cannot update an existing service to reference an `INACTIVE` task definition (although there may be up to a 10 minute window following deregistration where these restrictions have not yet taken effect).

Request Syntax

```
{
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

taskDefinition (p. 25)

The `family` and `revision` (`family:revision`) or full Amazon Resource Name (ARN) of the task definition to deregister. You must specify a `revision`.

Type: String

Required: Yes

Response Syntax

```
{
   "taskDefinition": {
      "containerDefinitions": [
         {
            "command": [ "string" ],
            "cpu": number,
            "disableNetworking": boolean,
            "dnsSearchDomains": [ "string" ],
            "dnsServers": [ "string" ],
            "dockerLabels": {
               "string" : "string"
            },
            "dockerSecurityOptions": [ "string" ],
            "entryPoint": [ "string" ],
            "environment": [
               {
                  "name": "string",
                  "value": "string"
               }
            ],
            "essential": boolean,
            "extraHosts": [
               {
                  "hostname": "string",
```

```
            "ipAddress": "string"
         }
      ],
      "hostname": "string",
      "image": "string",
      "links": [ "string" ],
      "logConfiguration": {
         "logDriver": "string",
         "options": {
            "string" : "string"
         }
      },
      "memory": number,
      "memoryReservation": number,
      "mountPoints": [
         {
            "containerPath": "string",
            "readOnly": boolean,
            "sourceVolume": "string"
         }
      ],
      "name": "string",
      "portMappings": [
         {
            "containerPort": number,
            "hostPort": number,
            "protocol": "string"
         }
      ],
      "privileged": boolean,
      "readonlyRootFilesystem": boolean,
      "ulimits": [
         {
            "hardLimit": number,
            "name": "string",
            "softLimit": number
         }
      ],
      "user": "string",
      "volumesFrom": [
         {
            "readOnly": boolean,
            "sourceContainer": "string"
         }
      ],
      "workingDirectory": "string"
   }
],
"family": "string",
"networkMode": "string",
"requiresAttributes": [
   {
      "name": "string",
      "value": "string"
   }
],
"revision": number,
"status": "string",
"taskDefinitionArn": "string",
```

```
      "taskRoleArn": "string",
      "volumes": [
         {
            "host": {
               "sourcePath": "string"
            },
            "name": "string"
         }
      ]
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

taskDefinition (p. 25)

The full description of the deregistered task.

Type: TaskDefinition (p. 141) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

The following example request deregisters the first revision of the `cpu-wave` task definition family (`cpu-wave:1`). Note that in the resulting output, the task definition status becomes `INACTIVE`.

Sample Request

```
POST / HTTP/1.1
```

```
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 35
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DeregisterTaskDefinition
X-Amz-Date: 20150429T184806Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "taskDefinition": "cpu-wave:1"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 12 Jun 2015 23:07:39 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 491
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "taskDefinition": {
    "containerDefinitions": [
      {
        "command": [
          "apt-get update; apt-get install stress; while true; do stress --
cpu $(( RANDOM % 4 )) -t $(( RANDOM % 10 )); done"
        ],
        "cpu": 50,
        "entryPoint": [
          "bash",
          "-c"
        ],
        "environment": [],
        "essential": true,
        "image": "ubuntu",
        "memory": 100,
        "mountPoints": [],
        "name": "wave",
        "portMappings": [],
        "volumesFrom": []
      }
    ],
    "family": "cpu-wave",
    "revision": 1,
    "status": "INACTIVE",
    "taskDefinitionArn": "arn:aws:ecs:us-west-2:012345678910:task-definition/
cpu-wave:1",
    "volumes": []
  }
}
```

DescribeClusters

Describes one or more of your clusters.

Request Syntax

```
{
   "clusters": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

clusters (p. 29)

A space-separated list of up to 100 cluster names or full cluster Amazon Resource Name (ARN) entries. If you do not specify a cluster, the default cluster is assumed.

Type: array of Strings

Required: No

Response Syntax

```
{
   "clusters": [
      {
         "activeServicesCount": number,
         "clusterArn": "string",
         "clusterName": "string",
         "pendingTasksCount": number,
         "registeredContainerInstancesCount": number,
         "runningTasksCount": number,
         "status": "string"
      }
   ],
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

clusters (p. 29)

The list of clusters.

Type: array of Cluster (p. 113) objects

failures (p. 29)
Any failures associated with the call.
Type: array of Failure (p. 126) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.
HTTP Status Code: 400

InvalidParameterException
The specified parameter is invalid. Review the available parameters for the API request.
HTTP Status Code: 400

ServerException
These errors are usually caused by a server issue.
HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request provides descriptive information about the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 25
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DescribeClusters
X-Amz-Date: 20150429T185014Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "clusters": [
    "default"
  ]
}
```

Sample Response

```
HTTP/1.1 200 OK
```

```
Server: Server
Date: Wed, 29 Apr 2015 18:50:14 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 220
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "clusters": [
    {
      "activeServicesCount": 1,
      "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
      "clusterName": "default",
      "pendingTasksCount": 0,
      "registeredContainerInstancesCount": 0,
      "runningTasksCount": 0,
      "status": "ACTIVE"
    }
  ],
  "failures": []
}
```

DescribeContainerInstances

Describes Amazon EC2 Container Service container instances. Returns metadata about registered and remaining resources on each container instance requested.

Request Syntax

```
{
   "cluster": "string",
   "containerInstances": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 32)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the container instances to describe. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstances (p. 32)

A space-separated list of container instance IDs or full Amazon Resource Name (ARN) entries.

Type: array of Strings

Required: Yes

Response Syntax

```
{
   "containerInstances": [
      {
         "agentConnected": boolean,
         "agentUpdateStatus": "string",
         "attributes": [
            {
               "name": "string",
               "value": "string"
            }
         ],
         "containerInstanceArn": "string",
         "ec2InstanceId": "string",
         "pendingTasksCount": number,
         "registeredResources": [
            {
               "doubleValue": number,
               "integerValue": number,
               "longValue": number,
               "name": "string",
               "stringSetValue": [ "string" ],
               "type": "string"
            }
```

```
         ],
         "remainingResources": [
            {
               "doubleValue": number,
               "integerValue": number,
               "longValue": number,
               "name": "string",
               "stringSetValue": [ "string" ],
               "type": "string"
            }
         ],
         "runningTasksCount": number,
         "status": "string",
         "version": number,
         "versionInfo": {
            "agentHash": "string",
            "agentVersion": "string",
            "dockerVersion": "string"
         }
      }
   ],
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

containerInstances (p. 32)
The list of container instances.

Type: array of ContainerInstance (p. 121) objects

failures (p. 32)
Any failures associated with the call.

Type: array of Failure (p. 126) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException
The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException
The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request provides descriptive information about a container instance with an ID of `f9cc75bb-0c94-46b9-bf6d-49d320bc1551` in the `default` cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 64
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DescribeContainerInstances
X-Amz-Date: 20160520T171518Z
User-Agent: aws-cli/1.10.30 Python/2.7.11 Darwin/15.4.0 botocore/1.4.17
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
    "containerInstances": [
        "f9cc75bb-0c94-46b9-bf6d-49d320bc1551"
    ]
}
```

Sample Response

```
{
    "containerInstances": [
        {
            "agentConnected": true,
            "attributes": [
                {
                    "name": "com.amazonaws.ecs.capability.privileged-
container"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-
api.1.17"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-
api.1.18"
                },
```

```
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-
api.1.19"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-
api.1.20"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-
api.1.21"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-
driver.json-file"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-
driver.syslog"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-
driver.awslogs"
                },
                {
                    "name": "com.amazonaws.ecs.capability.ecr-auth"
                }
            ],
            "containerInstanceArn": "arn:aws:ecs:us-
west-2:012345678910:container-instance/f9cc75bb-0c94-46b9-bf6d-49d320bc1551",
            "ec2InstanceId": "i-042f39dc",
            "pendingTasksCount": 0,
            "registeredResources": [
                {
                    "doubleValue": 0,
                    "integerValue": 1024,
                    "longValue": 0,
                    "name": "CPU",
                    "type": "INTEGER"
                },
                {
                    "doubleValue": 0,
                    "integerValue": 995,
                    "longValue": 0,
                    "name": "MEMORY",
                    "type": "INTEGER"
                },
                {
                    "doubleValue": 0,
                    "integerValue": 0,
                    "longValue": 0,
                    "name": "PORTS",
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET"
```

```
            },
            {
                "doubleValue": 0,
                "integerValue": 0,
                "longValue": 0,
                "name": "PORTS_UDP",
                "stringSetValue": [],
                "type": "STRINGSET"
            }
        ],
        "remainingResources": [
            {
                "doubleValue": 0,
                "integerValue": 1024,
                "longValue": 0,
                "name": "CPU",
                "type": "INTEGER"
            },
            {
                "doubleValue": 0,
                "integerValue": 995,
                "longValue": 0,
                "name": "MEMORY",
                "type": "INTEGER"
            },
            {
                "doubleValue": 0,
                "integerValue": 0,
                "longValue": 0,
                "name": "PORTS",
                "stringSetValue": [
                    "22",
                    "2376",
                    "2375",
                    "51678"
                ],
                "type": "STRINGSET"
            },
            {
                "doubleValue": 0,
                "integerValue": 0,
                "longValue": 0,
                "name": "PORTS_UDP",
                "stringSetValue": [],
                "type": "STRINGSET"
            }
        ],
        "runningTasksCount": 0,
        "status": "ACTIVE",
        "version": 850,
        "versionInfo": {
            "agentHash": "0931217",
            "agentVersion": "1.9.0",
            "dockerVersion": "DockerVersion: 1.9.1"
        }
    }
],
"failures": []
```

```
}
```

DescribeServices

Describes the specified services running in your cluster.

Request Syntax

```
{
   "cluster": "string",
   "services": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 38)

The name of the cluster that hosts the service to describe. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

services (p. 38)

A list of services to describe. You may specify up to 10 services to describe in a single operation.

Type: array of Strings

Required: Yes

Response Syntax

```
{
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ],
   "services": [
      {
         "clusterArn": "string",
         "createdAt": number,
         "deploymentConfiguration": {
            "maximumPercent": number,
            "minimumHealthyPercent": number
         },
         "deployments": [
            {
               "createdAt": number,
               "desiredCount": number,
               "id": "string",
               "pendingCount": number,
               "runningCount": number,
               "status": "string",
```

```
            "taskDefinition": "string",
            "updatedAt": number
         }
      ],
      "desiredCount": number,
      "events": [
         {
            "createdAt": number,
            "id": "string",
            "message": "string"
         }
      ],
      "loadBalancers": [
         {
            "containerName": "string",
            "containerPort": number,
            "loadBalancerName": "string",
            "targetGroupArn": "string"
         }
      ],
      "pendingCount": number,
      "roleArn": "string",
      "runningCount": number,
      "serviceArn": "string",
      "serviceName": "string",
      "status": "string",
      "taskDefinition": "string"
   }
]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

failures (p. 38)

Any failures associated with the call.

Type: array of Failure (p. 126) objects

services (p. 38)

The list of services described.

Type: array of Service (p. 136) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request provides a full description of the `bunker_buster` service in the `telemetry` cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 55
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DescribeServices
X-Amz-Date: 20150528T163859Z
User-Agent: aws-cli/1.7.30 Python/2.7.9 Darwin/14.3.0
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "services": [
    "bunker-buster"
  ],
  "cluster": "telemetry"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:02:59 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 2449
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "failures": [],
  "services": [
    {
      "clusterArn": "arn:aws:ecs:us-west-2:012345678910:cluster/telemetry",
```

```
      "deploymentConfiguration": {
          "maximumPercent": 200,
          "minimumHealthyPercent": 100
      },
      "deployments": [
        {
          "createdAt": 1432829320.611,
          "desiredCount": 4,
          "id": "ecs-svc/9223370604025455196",
          "pendingCount": 0,
          "runningCount": 4,
          "status": "PRIMARY",
          "taskDefinition": "arn:aws:ecs:us-west-2:012345678910:task-
definition/hpcc-t2-medium:1",
          "updatedAt": 1432829320.611
        }
      ],
      "desiredCount": 4,
      "events": [],
      "loadBalancers": [],
      "pendingCount": 0,
      "runningCount": 4,
      "serviceArn": "arn:aws:ecs:us-west-2:012345678910:service/bunker-
buster",
      "serviceName": "bunker-buster",
      "status": "ACTIVE",
      "taskDefinition": "arn:aws:ecs:us-west-2:012345678910:task-definition/
hpcc-t2-medium:1"
    }
  ]
}
```

DescribeTaskDefinition

Describes a task definition. You can specify a `family` and `revision` to find information about a specific task definition, or you can simply specify the family to find the latest `ACTIVE` revision in that family.

Note
You can only describe `INACTIVE` task definitions while an active task or service references them.

Request Syntax

```
{
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

taskDefinition (p. 42)
The `family` for the latest `ACTIVE` revision, `family` and `revision` (`family:revision`) for a specific revision in the family, or full Amazon Resource Name (ARN) of the task definition to describe.

Type: String

Required: Yes

Response Syntax

```
{
   "taskDefinition": {
      "containerDefinitions": [
         {
            "command": [ "string" ],
            "cpu": number,
            "disableNetworking": boolean,
            "dnsSearchDomains": [ "string" ],
            "dnsServers": [ "string" ],
            "dockerLabels": {
               "string" : "string"
            },
            "dockerSecurityOptions": [ "string" ],
            "entryPoint": [ "string" ],
            "environment": [
               {
                  "name": "string",
                  "value": "string"
               }
            ],
            "essential": boolean,
            "extraHosts": [
               {
                  "hostname": "string",
```

```
                "ipAddress": "string"
             }
          ],
          "hostname": "string",
          "image": "string",
          "links": [ "string" ],
          "logConfiguration": {
             "logDriver": "string",
             "options": {
                "string" : "string"
             }
          },
          "memory": number,
          "memoryReservation": number,
          "mountPoints": [
             {
                "containerPath": "string",
                "readOnly": boolean,
                "sourceVolume": "string"
             }
          ],
          "name": "string",
          "portMappings": [
             {
                "containerPort": number,
                "hostPort": number,
                "protocol": "string"
             }
          ],
          "privileged": boolean,
          "readonlyRootFilesystem": boolean,
          "ulimits": [
             {
                "hardLimit": number,
                "name": "string",
                "softLimit": number
             }
          ],
          "user": "string",
          "volumesFrom": [
             {
                "readOnly": boolean,
                "sourceContainer": "string"
             }
          ],
          "workingDirectory": "string"
       }
    ],
    "family": "string",
    "networkMode": "string",
    "requiresAttributes": [
       {
          "name": "string",
          "value": "string"
       }
    ],
    "revision": number,
    "status": "string",
    "taskDefinitionArn": "string",
```

```
      "taskRoleArn": "string",
      "volumes": [
         {
            "host": {
               "sourcePath": "string"
            },
            "name": "string"
         }
      ]
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

taskDefinition (p. 42)

The full task definition description.

Type: TaskDefinition (p. 141) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request provides descriptive information about the 10th revision of a task definition in the `hello_world` family.

Sample Request

```
POST / HTTP/1.1
```

```
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 36
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DescribeTaskDefinition
X-Amz-Date: 20150429T190902Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "taskDefinition": "hello_world:10"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:09:03 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 574
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "taskDefinition": {
    "containerDefinitions": [
      {
        "cpu": 10,
        "environment": [],
        "essential": true,
        "image": "wordpress",
        "links": [
          "mysql"
        ],
        "memory": 500,
        "mountPoints": [],
        "name": "wordpress",
        "portMappings": [
          {
            "containerPort": 80,
            "hostPort": 80
          }
        ],
        "volumesFrom": []
      },
      {
        "cpu": 10,
        "environment": [
          {
            "name": "MYSQL_ROOT_PASSWORD",
            "value": "password"
          }
        ],
        "essential": true,
        "image": "mysql",
        "memory": 500,
        "mountPoints": [],
        "name": "mysql",
        "portMappings": [],
```

```
        "volumesFrom": []
      }
    ],
    "family": "hello_world",
    "revision": 10,
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-definition/
hello_world:10",
    "volumes": []
  }
}
```

DescribeTasks

Describes a specified task or tasks.

Request Syntax

```
{
   "cluster": "string",
   "tasks": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 47)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the task to describe. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

tasks (p. 47)

A space-separated list of task IDs or full Amazon Resource Name (ARN) entries.

Type: array of Strings

Required: Yes

Response Syntax

```
{
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ],
   "tasks": [
      {
         "clusterArn": "string",
         "containerInstanceArn": "string",
         "containers": [
            {
               "containerArn": "string",
               "exitCode": number,
               "lastStatus": "string",
               "name": "string",
               "networkBindings": [
                  {
                     "bindIP": "string",
                     "containerPort": number,
                     "hostPort": number,
                     "protocol": "string"
```

```
                }
             ],
             "reason": "string",
             "taskArn": "string"
          }
       ],
       "createdAt": number,
       "desiredStatus": "string",
       "lastStatus": "string",
       "overrides": {
          "containerOverrides": [
             {
                "command": [ "string" ],
                "environment": [
                   {
                      "name": "string",
                      "value": "string"
                   }
                ],
                "name": "string"
             }
          ],
          "taskRoleArn": "string"
       },
       "startedAt": number,
       "startedBy": "string",
       "stoppedAt": number,
       "stoppedReason": "string",
       "taskArn": "string",
       "taskDefinitionArn": "string",
       "version": number
    }
 ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

failures (p. 47)

Any failures associated with the call.

Type: array of Failure (p. 126) objects

tasks (p. 47)

The list of tasks.

Type: array of Task (p. 139) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request provides descriptive information about a task with an ID of `1dc5c17a-422b-4dc4-b493-371970c6c4d6` in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 51
X-Amz-Target: AmazonEC2ContainerServiceV20141113.DescribeTasks
X-Amz-Date: 20161121T214915Z
User-Agent: aws-cli/1.11.13 Python/2.7.12 Darwin/16.1.0 botocore/1.4.66
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "tasks": [
    "1dc5c17a-422b-4dc4-b493-371970c6c4d6"
  ]
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 21 Nov 2016 21:49:16 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1238
Connection: keep-alive

x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f
```

```
{
  "failures": [],
  "tasks": [
    {
      "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/1dc5c17a-422b-4dc4-
b493-371970c6c4d6",
      "overrides": {
        "containerOverrides": [
          {
            "name": "simple-app"
          },
          {
            "name": "busybox"
          }
        ]
      },
      "lastStatus": "RUNNING",
      "containerInstanceArn": "arn:aws:ecs:us-east-1:012345678910:container-
instance/5991d8da-1d59-49d2-a31f-4230f9e73140",
      "createdAt": 1476822811.295,
      "version": 0,
      "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
      "startedAt": 1476822833.998,
      "desiredStatus": "RUNNING",
      "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-
definition/console-sample-app-dynamic-ports:1",
      "startedBy": "ecs-svc/9223370560032507596",
      "containers": [
        {
          "containerArn": "arn:aws:ecs:us-
east-1:012345678910:container/4df26bb4-f057-467b-a079-961675296e64",
          "taskArn": "arn:aws:ecs:us-
east-1:012345678910:task/1dc5c17a-422b-4dc4-b493-371970c6c4d6",
          "lastStatus": "RUNNING",
          "name": "simple-app",
          "networkBindings": [
            {
              "protocol": "tcp",
              "bindIP": "0.0.0.0",
              "containerPort": 80,
              "hostPort": 32774
            }
          ]
        },
        {
          "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
e09064f7-7361-4c87-8ab9-8d073bbdbcb9",
          "taskArn": "arn:aws:ecs:us-
east-1:012345678910:task/1dc5c17a-422b-4dc4-b493-371970c6c4d6",
          "lastStatus": "RUNNING",
          "name": "busybox",
          "networkBindings": []
        }
      ]
    }
  ]
}
```

DiscoverPollEndpoint

Note
This action is only used by the Amazon EC2 Container Service agent, and it is not intended for use outside of the agent.

Returns an endpoint for the Amazon EC2 Container Service agent to poll for updates.

Request Syntax

```
{
   "cluster": "string",
   "containerInstance": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 51)
The cluster that the container instance belongs to.

Type: String

Required: No

containerInstance (p. 51)
The container instance ID or full Amazon Resource Name (ARN) of the container instance. The ARN contains the `arn:aws:ecs` namespace, followed by the region of the container instance, the AWS account ID of the container instance owner, the `container-instance` namespace, and then the container instance ID. For example, `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:container-instance/`*`container_instance_ID`*.

Type: String

Required: No

Response Syntax

```
{
   "endpoint": "string",
   "telemetryEndpoint": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

endpoint (p. 51)
The endpoint for the Amazon ECS agent to poll.

Type: String

telemetryEndpoint (p. 51)
The telemetry endpoint for the Amazon ECS agent.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

ListClusters

Returns a list of existing clusters.

Request Syntax

```
{
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

maxResults (p. 53)

The maximum number of cluster results returned by `ListClusters` in paginated output. When this parameter is used, `ListClusters` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListClusters` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListClusters` returns up to 100 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 53)

The `nextToken` value returned from a previous paginated `ListClusters` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

Response Syntax

```
{
   "clusterArns": [ "string" ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

clusterArns (p. 53)

The list of full Amazon Resource Name (ARN) entries for each cluster associated with your account.

Type: array of Strings

nextToken (p. 53)

The `nextToken` value to include in a future `ListClusters` request. When the results of a `ListClusters` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists the clusters for your account.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListClusters
X-Amz-Date: 20150429T170621Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
```

```
Date: Wed, 29 Apr 2015 17:06:21 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 126
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "clusterArns": [
    "arn:aws:ecs:us-east-1:012345678910:cluster/My-cluster",
    "arn:aws:ecs:us-east-1:012345678910:cluster/default"
  ]
}
```

ListContainerInstances

Returns a list of container instances in a specified cluster.

Request Syntax

```
{
   "cluster": "string",
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 56)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the container instances to list. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

maxResults (p. 56)

The maximum number of container instance results returned by `ListContainerInstances` in paginated output. When this parameter is used, `ListContainerInstances` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListContainerInstances` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListContainerInstances` returns up to 100 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 56)

The `nextToken` value returned from a previous paginated `ListContainerInstances` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

Response Syntax

```
{
   "containerInstanceArns": [ "string" ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

containerInstanceArns (p. 56)
The list of container instances with full Amazon Resource Name (ARN) entries for each container instance associated with the specified cluster.

Type: array of Strings

nextToken (p. 56)
The `nextToken` value to include in a future `ListContainerInstances` request. When the results of a `ListContainerInstances` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException
The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException
The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException
These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists the container instances in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-west-2.amazonaws.com
Accept-Encoding: identity
```

```
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListContainerInstances
X-Amz-Date: 20150429T175306Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 17:53:06 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 492
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:012345678910:container-
instance/14e8cce9-0b16-4af4-bfac-a85f7587aa98",
    "arn:aws:ecs:us-west-2:012345678910:container-
instance/23bbf61b-45b4-4a4f-b90c-c86290f066d6",
    "arn:aws:ecs:us-west-2:012345678910:container-instance/
bd0abd43-94ce-4909-9750-0dcc471ca4cb",
    "arn:aws:ecs:us-west-2:012345678910:container-instance/
c967b2ee-68ea-415b-b220-5936b26e6a04",
    "arn:aws:ecs:us-west-2:012345678910:container-instance/
f5ec555b-8da4-48e1-8427-0e03c3674a29"
  ]
}
```

ListServices

Lists the services that are running in a specified cluster.

Request Syntax

```
{
   "cluster": "string",
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 59)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the services to list. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

maxResults (p. 59)

The maximum number of container instance results returned by `ListServices` in paginated output. When this parameter is used, `ListServices` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListServices` request with the returned `nextToken` value. This value can be between 1 and 10. If this parameter is not used, then `ListServices` returns up to 10 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 59)

The `nextToken` value returned from a previous paginated `ListServices` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

Response Syntax

```
{
   "nextToken": "string",
   "serviceArns": [ "string" ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

nextToken (p. 59)
The `nextToken` value to include in a future `ListServices` request. When the results of a `ListServices` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

serviceArns (p. 59)
The list of full Amazon Resource Name (ARN) entries for each service associated with the specified cluster.

Type: array of Strings

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException
The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException
The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException
These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists the services in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
```

```
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListServices
X-Amz-Date: 20150429T191342Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:13:42 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 138
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "serviceArns": [
    "arn:aws:ecs:us-east-1:012345678910:service/hello_world",
    "arn:aws:ecs:us-east-1:012345678910:service/ecs-simple-service"
  ]
}
```

ListTaskDefinitionFamilies

Returns a list of task definition families that are registered to your account (which may include task definition families that no longer have any `ACTIVE` task definition revisions).

You can filter out task definition families that do not contain any `ACTIVE` task definition revisions by setting the `status` parameter to `ACTIVE`. You can also filter the results with the `familyPrefix` parameter.

Request Syntax

```
{
   "familyPrefix": "string",
   "maxResults": number,
   "nextToken": "string",
   "status": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

familyPrefix (p. 62)

The `familyPrefix` is a string that is used to filter the results of `ListTaskDefinitionFamilies`. If you specify a `familyPrefix`, only task definition family names that begin with the `familyPrefix` string are returned.

Type: String

Required: No

maxResults (p. 62)

The maximum number of task definition family results returned by `ListTaskDefinitionFamilies` in paginated output. When this parameter is used, `ListTaskDefinitions` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListTaskDefinitionFamilies` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListTaskDefinitionFamilies` returns up to 100 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 62)

The `nextToken` value returned from a previous paginated `ListTaskDefinitionFamilies` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

status (p. 62)

The task definition family status with which to filter the `ListTaskDefinitionFamilies` results. By default, both `ACTIVE` and `INACTIVE` task definition families are listed. If this parameter is set to

`ACTIVE`, only task definition families that have an `ACTIVE` task definition revision are returned. If this parameter is set to `INACTIVE`, only task definition families that do not have any `ACTIVE` task definition revisions are returned. If you paginate the resulting output, be sure to keep the `status` value constant in each subsequent request.

Type: String

Valid Values: `ACTIVE | INACTIVE | ALL`

Required: No

Response Syntax

```
{
   "families": [ "string" ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

families (p. 63)

The list of task definition family names that match the `ListTaskDefinitionFamilies` request.

Type: array of Strings

nextToken (p. 63)

The `nextToken` value to include in a future `ListTaskDefinitionFamilies` request. When the results of a `ListTaskDefinitionFamilies` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Examples

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists all of the task definition families in your account in the current region.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListTaskDefinitionFamilies
X-Amz-Date: 20150429T191650Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:16:51 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 270
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "families": [
    "console-sample-app",
    "ecs-demo",
    "ecs-private",
    "hello_world",
    "hpcc",
    "hpcc-t2-medium",
    "image-dedupe",
    "node-dedupe",
    "port-mappings",
    "redis-volumes-from",
    "sleep360",
    "terrible-timer",
    "test-volumes-from",
    "tt-empty",
    "tt-empty-2vol",
    "tt-empty-volumes",
    "web-timer"
  ]
}
```

Example

This example request lists all of the task definition families in your account in the current region that begin with `hpcc`.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 24
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListTaskDefinitionFamilies
X-Amz-Date: 20150429T191825Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "familyPrefix": "hpcc"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:18:25 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 38
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "families": [
    "hpcc",
    "hpcc-t2-medium"
  ]
}
```

ListTaskDefinitions

Returns a list of task definitions that are registered to your account. You can filter the results by family name with the `familyPrefix` parameter or by status with the `status` parameter.

Request Syntax

```
{
   "familyPrefix": "string",
   "maxResults": number,
   "nextToken": "string",
   "sort": "string",
   "status": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

familyPrefix (p. 66)

The full family name with which to filter the `ListTaskDefinitions` results. Specifying a `familyPrefix` limits the listed task definitions to task definition revisions that belong to that family.

Type: String

Required: No

maxResults (p. 66)

The maximum number of task definition results returned by `ListTaskDefinitions` in paginated output. When this parameter is used, `ListTaskDefinitions` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListTaskDefinitions` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListTaskDefinitions` returns up to 100 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 66)

The `nextToken` value returned from a previous paginated `ListTaskDefinitions` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

sort (p. 66)

The order in which to sort the results. Valid values are `ASC` and `DESC`. By default (`ASC`), task definitions are listed lexicographically by family name and in ascending numerical order by revision so that the newest task definitions in a family are listed last. Setting this parameter to `DESC` reverses the sort order on family name and revision so that the newest task definitions in a family are listed first.

Type: String

Valid Values: `ASC` | `DESC`

Required: No

status (p. 66)

The task definition status with which to filter the `ListTaskDefinitions` results. By default, only `ACTIVE` task definitions are listed. By setting this parameter to `INACTIVE`, you can view task definitions that are `INACTIVE` as long as an active task or service still references them. If you paginate the resulting output, be sure to keep the `status` value constant in each subsequent request.

Type: String

Valid Values: `ACTIVE` | `INACTIVE`

Required: No

Response Syntax

```
{
   "nextToken": "string",
   "taskDefinitionArns": [ "string" ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

nextToken (p. 67)

The `nextToken` value to include in a future `ListTaskDefinitions` request. When the results of a `ListTaskDefinitions` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

taskDefinitionArns (p. 67)

The list of task definition Amazon Resource Name (ARN) entries for the `ListTaskDefinitions` request.

Type: array of Strings

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists all of the task definitions in the `hello_world` family.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 31
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListTaskDefinitions
X-Amz-Date: 20150429T192041Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "familyPrefix": "hello_world"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:20:41 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 695
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "taskDefinitionArns": [
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:1",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:2",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:3",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:4",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:5",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:6",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:7",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:8",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:9",
    "arn:aws:ecs:us-east-1:012345678910:task-definition/hello_world:10"
  ]
}
```

ListTasks

Returns a list of tasks for a specified cluster. You can filter the results by family name, by a particular container instance, or by the desired status of the task with the `family`, `containerInstance`, and `desiredStatus` parameters.

Recently-stopped tasks might appear in the returned results. Currently, stopped tasks appear in the returned results for at least one hour.

Request Syntax

```
{
   "cluster": "string",
   "containerInstance": "string",
   "desiredStatus": "string",
   "family": "string",
   "maxResults": number,
   "nextToken": "string",
   "serviceName": "string",
   "startedBy": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 69)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the tasks to list. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstance (p. 69)

The container instance ID or full Amazon Resource Name (ARN) of the container instance with which to filter the `ListTasks` results. Specifying a `containerInstance` limits the results to tasks that belong to that container instance.

Type: String

Required: No

desiredStatus (p. 69)

The task desired status with which to filter the `ListTasks` results. Specifying a `desiredStatus` of `STOPPED` limits the results to tasks that ECS has set the desired status to `STOPPED`, which can be useful for debugging tasks that are not starting properly or have died or finished. The default status filter is `RUNNING`, which shows tasks that ECS has set the desired status to `RUNNING`.

> **Note**
> Although you can filter results based on a desired status of `PENDING`, this will not return any results because ECS never sets the desired status of a task to that value (only a task's `lastStatus` may have a value of `PENDING`).

Type: String

Valid Values: `RUNNING | PENDING | STOPPED`

Required: No

family (p. 69)

The name of the family with which to filter the `ListTasks` results. Specifying a `family` limits the results to tasks that belong to that family.

Type: String

Required: No

maxResults (p. 69)

The maximum number of task results returned by `ListTasks` in paginated output. When this parameter is used, `ListTasks` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListTasks` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListTasks` returns up to 100 results and a `nextToken` value if applicable.

Type: Integer

Required: No

nextToken (p. 69)

The `nextToken` value returned from a previous paginated `ListTasks` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is `null` when there are no more results to return.

> **Note**
> This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

serviceName (p. 69)

The name of the service with which to filter the `ListTasks` results. Specifying a `serviceName` limits the results to tasks that belong to that service.

Type: String

Required: No

startedBy (p. 69)

The `startedBy` value with which to filter the task results. Specifying a `startedBy` value limits the results to tasks that were started with that value.

Type: String

Required: No

Response Syntax

```
{
   "nextToken": "string",
   "taskArns": [ "string" ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

nextToken (p. 70)

The `nextToken` value to include in a future `ListTasks` request. When the results of a `ListTasks` request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

taskArns (p. 70)

The list of task Amazon Resource Name (ARN) entries for the `ListTasks` request.

Type: array of Strings

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

ServiceNotFoundException

The specified service could not be found. You can view your available services with ListServices (p. 59). Amazon ECS services are cluster-specific and region-specific.

HTTP Status Code: 400

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request lists all of the tasks in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListTasks
X-Amz-Date: 20150429T192615Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:26:16 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 330
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "taskArns": [
    "arn:aws:ecs:us-east-1:012345678910:task/0b69d5c0-
d655-4695-98cd-5d2d526d9d5a",
    "arn:aws:ecs:us-east-1:012345678910:task/51a01bdf-d00e-487e-
ab14-7645330b6207",
    "arn:aws:ecs:us-east-1:012345678910:task/b0b28bb8-2be3-4810-
b52b-88df129d893c",
    "arn:aws:ecs:us-east-1:012345678910:task/c09f0188-7f87-4b0f-
bfc3-16296622b6fe"
  ]
}
```

RegisterContainerInstance

Note
This action is only used by the Amazon EC2 Container Service agent, and it is not intended for use outside of the agent.

Registers an EC2 instance into the specified cluster. This instance becomes available to place containers on.

Request Syntax

```
{
   "attributes": [
      {
         "name": "string",
         "value": "string"
      }
   ],
   "cluster": "string",
   "containerInstanceArn": "string",
   "instanceIdentityDocument": "string",
   "instanceIdentityDocumentSignature": "string",
   "totalResources": [
      {
         "doubleValue": number,
         "integerValue": number,
         "longValue": number,
         "name": "string",
         "stringSetValue": [ "string" ],
         "type": "string"
      }
   ],
   "versionInfo": {
      "agentHash": "string",
      "agentVersion": "string",
      "dockerVersion": "string"
   }
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

attributes (p. 73)

The container instance attributes that this container instance supports.

Type: array of Attribute (p. 112) objects

Required: No

cluster (p. 73)

The short name or full Amazon Resource Name (ARN) of the cluster with which to register your container instance. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstanceArn (p. 73)
The Amazon Resource Name (ARN) of the container instance (if it was previously registered).
Type: String
Required: No

instanceIdentityDocument (p. 73)
The instance identity document for the EC2 instance to register. This document can be found by running the following command from the instance: `curl http://169.254.169.254/latest/dynamic/instance-identity/document/`
Type: String
Required: No

instanceIdentityDocumentSignature (p. 73)
The instance identity document signature for the EC2 instance to register. This signature can be found by running the following command from the instance: `curl http://169.254.169.254/latest/dynamic/instance-identity/signature/`
Type: String
Required: No

totalResources (p. 73)
The resources available on the instance.
Type: array of Resource (p. 135) objects
Required: No

versionInfo (p. 73)
The version information for the Amazon ECS container agent and Docker daemon running on the container instance.
Type: VersionInfo (p. 145) object
Required: No

Response Syntax

```
{
   "containerInstance": {
      "agentConnected": boolean,
      "agentUpdateStatus": "string",
      "attributes": [
         {
            "name": "string",
            "value": "string"
         }
      ],
      "containerInstanceArn": "string",
      "ec2InstanceId": "string",
      "pendingTasksCount": number,
      "registeredResources": [
         {
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "remainingResources": [
         {
```

```
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "runningTasksCount": number,
      "status": "string",
      "version": number,
      "versionInfo": {
         "agentHash": "string",
         "agentVersion": "string",
         "dockerVersion": "string"
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

containerInstance (p. 74)

The container instance that was registered.

Type: ContainerInstance (p. 121) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

RegisterTaskDefinition

Registers a new task definition from the supplied `family` and `containerDefinitions`. Optionally, you can add data volumes to your containers with the `volumes` parameter. For more information about task definition parameters and defaults, see Amazon ECS Task Definitions in the *Amazon EC2 Container Service Developer Guide*.

You can specify an IAM role for your task with the `taskRoleArn` parameter. When you specify an IAM role for a task, its containers can then use the latest versions of the AWS CLI or SDKs to make API requests to the AWS services that are specified in the IAM policy associated with the role. For more information, see IAM Roles for Tasks in the *Amazon EC2 Container Service Developer Guide*.

You can specify a Docker networking mode for the containers in your task definition with the `networkMode` parameter. The available network modes correspond to those described in Network settings in the Docker run reference.

Request Syntax

```
{
   "containerDefinitions": [
      {
         "command": [ "string" ],
         "cpu": number,
         "disableNetworking": boolean,
         "dnsSearchDomains": [ "string" ],
         "dnsServers": [ "string" ],
         "dockerLabels": {
            "string" : "string"
         },
         "dockerSecurityOptions": [ "string" ],
         "entryPoint": [ "string" ],
         "environment": [
            {
               "name": "string",
               "value": "string"
            }
         ],
         "essential": boolean,
         "extraHosts": [
            {
               "hostname": "string",
               "ipAddress": "string"
            }
         ],
         "hostname": "string",
         "image": "string",
         "links": [ "string" ],
         "logConfiguration": {
            "logDriver": "string",
            "options": {
               "string" : "string"
            }
         },
         "memory": number,
         "memoryReservation": number,
         "mountPoints": [
            {
               "containerPath": "string",
               "readOnly": boolean,
```

```
                "sourceVolume": "string"
            }
         ],
         "name": "string",
         "portMappings": [
            {
               "containerPort": number,
               "hostPort": number,
               "protocol": "string"
            }
         ],
         "privileged": boolean,
         "readonlyRootFilesystem": boolean,
         "ulimits": [
            {
               "hardLimit": number,
               "name": "string",
               "softLimit": number
            }
         ],
         "user": "string",
         "volumesFrom": [
            {
               "readOnly": boolean,
               "sourceContainer": "string"
            }
         ],
         "workingDirectory": "string"
      }
   ],
   "family": "string",
   "networkMode": "string",
   "taskRoleArn": "string",
   "volumes": [
      {
         "host": {
            "sourcePath": "string"
         },
         "name": "string"
      }
   ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

containerDefinitions (p. 76)

A list of container definitions in JSON format that describe the different containers that make up your task.

Type: array of ContainerDefinition (p. 115) objects

Required: Yes

family (p. 76)

You must specify a `family` for a task definition, which allows you to track multiple versions of the same task definition. The `family` is used as a name for your task definition. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

Type: String

Required: Yes

networkMode (p. 76)

The Docker networking mode to use for the containers in the task. The valid values are `none`, `bridge`, and `host`.

The default Docker network mode is `bridge`. If the network mode is set to `none`, you cannot specify port mappings in your container definitions, and the task's containers do not have external connectivity. The `host` network mode offers the highest networking performance for containers because they use the host network stack instead of the virtualized network stack provided by the `bridge` mode; however, exposed container ports are mapped directly to the corresponding host port, so you cannot take advantage of dynamic host port mappings or run multiple instantiations of the same task on a single container instance if port mappings are used.

For more information, see Network settings in the *Docker run reference*.

Type: String

Valid Values: `bridge | host | none`

Required: No

taskRoleArn (p. 76)

The short name or full Amazon Resource Name (ARN) of the IAM role that containers in this task can assume. All containers in this task are granted the permissions that are specified in this role. For more information, see IAM Roles for Tasks in the *Amazon EC2 Container Service Developer Guide*.

Type: String

Required: No

volumes (p. 76)

A list of volume definitions in JSON format that containers in your task may use.

Type: array of Volume (p. 146) objects

Required: No

Response Syntax

```
{
   "taskDefinition": {
      "containerDefinitions": [
         {
            "command": [ "string" ],
            "cpu": number,
            "disableNetworking": boolean,
            "dnsSearchDomains": [ "string" ],
            "dnsServers": [ "string" ],
            "dockerLabels": {
               "string" : "string"
            },
            "dockerSecurityOptions": [ "string" ],
            "entryPoint": [ "string" ],
            "environment": [
               {
                  "name": "string",
                  "value": "string"
               }
```

```
            ],
            "essential": boolean,
            "extraHosts": [
               {
                  "hostname": "string",
                  "ipAddress": "string"
               }
            ],
            "hostname": "string",
            "image": "string",
            "links": [ "string" ],
            "logConfiguration": {
               "logDriver": "string",
               "options": {
                  "string" : "string"
               }
            },
            "memory": number,
            "memoryReservation": number,
            "mountPoints": [
               {
                  "containerPath": "string",
                  "readOnly": boolean,
                  "sourceVolume": "string"
               }
            ],
            "name": "string",
            "portMappings": [
               {
                  "containerPort": number,
                  "hostPort": number,
                  "protocol": "string"
               }
            ],
            "privileged": boolean,
            "readonlyRootFilesystem": boolean,
            "ulimits": [
               {
                  "hardLimit": number,
                  "name": "string",
                  "softLimit": number
               }
            ],
            "user": "string",
            "volumesFrom": [
               {
                  "readOnly": boolean,
                  "sourceContainer": "string"
               }
            ],
            "workingDirectory": "string"
         }
      ],
      "family": "string",
      "networkMode": "string",
      "requiresAttributes": [
         {
            "name": "string",
            "value": "string"
```

```
            }
         ],
         "revision": number,
         "status": "string",
         "taskDefinitionArn": "string",
         "taskRoleArn": "string",
         "volumes": [
            {
               "host": {
                  "sourcePath": "string"
               },
               "name": "string"
            }
         ]
      }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

taskDefinition (p. 78)

The full description of the registered task definition.

Type: TaskDefinition (p. 141) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request registers a task definition in the `hello_world` family with the `host` networking mode.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 486
X-Amz-Target: AmazonEC2ContainerServiceV20141113.RegisterTaskDefinition
X-Amz-Date: 20150429T193109Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "networkMode": "host",
  "containerDefinitions": [
    {
      "name": "wordpress",
      "links": [
        "mysql"
      ],
      "image": "wordpress",
      "essential": true,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "memory": 500,
      "cpu": 10
    },
    {
      "name": "mysql",
      "image": "mysql",
      "cpu": 10,
      "environment": [
        {
          "name": "MYSQL_ROOT_PASSWORD",
          "value": "password"
        }
      ],
      "memory": 500,
      "essential": true
    }
  ],
  "family": "hello_world"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 12 Aug 2016 22:17:20 GMT
Content-Type: application/x-amz-json-1.1
```

```
Content-Length: 714
Connection: keep-alive
x-amzn-RequestId: 896d7e0f-60da-11e6-8e21-55c97a4b6423

{
  "taskDefinition": {
    "containerDefinitions": [
      {
        "cpu": 10,
        "environment": [],
        "essential": true,
        "image": "wordpress",
        "links": [
          "mysql"
        ],
        "memory": 500,
        "mountPoints": [],
        "name": "wordpress",
        "portMappings": [
          {
            "containerPort": 80,
            "hostPort": 80,
            "protocol": "tcp"
          }
        ],
        "volumesFrom": []
      },
      {
        "cpu": 10,
        "environment": [
          {
            "name": "MYSQL_ROOT_PASSWORD",
            "value": "password"
          }
        ],
        "essential": true,
        "image": "mysql",
        "memory": 500,
        "mountPoints": [],
        "name": "mysql",
        "portMappings": [],
        "volumesFrom": []
      }
    ],
    "family": "hello_world",
    "networkMode": "host",
    "requiresAttributes": [
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      }
    ],
    "revision": 4,
    "status": "ACTIVE",
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-definition/
hello_world:4",
    "volumes": []
  }
}
```

RunTask

Start a task using random placement and the default Amazon ECS scheduler. To use your own scheduler or place a task on a specific container instance, use `StartTask` instead.

Important
The `count` parameter is limited to 10 tasks per call.

Request Syntax

```
{
   "cluster": "string",
   "count": number,
   "overrides": {
      "containerOverrides": [
         {
            "command": [ "string" ],
            "environment": [
               {
                  "name": "string",
                  "value": "string"
               }
            ],
            "name": "string"
         }
      ],
      "taskRoleArn": "string"
   },
   "startedBy": "string",
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 83)
The short name or full Amazon Resource Name (ARN) of the cluster on which to run your task. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

count (p. 83)
The number of instantiations of the specified task to place on your cluster.

Important
The `count` parameter is limited to 10 tasks per call.

Type: Integer

Required: No

overrides (p. 83)
A list of container overrides in JSON format that specify the name of a container in the specified task definition and the overrides it should receive. You can override the default command for a container (that is specified in the task definition or Docker image) with a `command` override. You can also override existing environment variables (that are specified in the task definition or Docker image) on a container or add new environment variables to it with an `environment` override.

Note
A total of 8192 characters are allowed for overrides. This limit includes the JSON formatting characters of the override structure.

Type: TaskOverride (p. 143) object

Required: No

startedBy (p. 83)

An optional tag specified when a task is started. For example if you automatically trigger a task to run a batch process job, you could apply a unique identifier for that job to your task with the `startedBy` parameter. You can then identify which tasks belong to that job by filtering the results of a ListTasks (p. 69) call with the `startedBy` value. Up to 36 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

If a task is started by an Amazon ECS service, then the `startedBy` parameter contains the deployment ID of the service that starts it.

Type: String

Required: No

taskDefinition (p. 83)

The `family` and `revision` (`family:revision`) or full Amazon Resource Name (ARN) of the task definition to run. If a `revision` is not specified, the latest `ACTIVE` revision is used.

Type: String

Required: Yes

Response Syntax

```
{
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ],
   "tasks": [
      {
         "clusterArn": "string",
         "containerInstanceArn": "string",
         "containers": [
            {
               "containerArn": "string",
               "exitCode": number,
               "lastStatus": "string",
               "name": "string",
               "networkBindings": [
                  {
                     "bindIP": "string",
                     "containerPort": number,
                     "hostPort": number,
                     "protocol": "string"
                  }
               ],
               "reason": "string",
               "taskArn": "string"
            }
         ],
         "createdAt": number,
         "desiredStatus": "string",
```

```
         "lastStatus": "string",
         "overrides": {
            "containerOverrides": [
               {
                  "command": [ "string" ],
                  "environment": [
                     {
                        "name": "string",
                        "value": "string"
                     }
                  ],
                  "name": "string"
               }
            ],
            "taskRoleArn": "string"
         },
         "startedAt": number,
         "startedBy": "string",
         "stoppedAt": number,
         "stoppedReason": "string",
         "taskArn": "string",
         "taskDefinitionArn": "string",
         "version": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

failures (p. 84)

Any failures associated with the call.

Type: array of Failure (p. 126) objects

tasks (p. 84)

A full description of the tasks that were run. Each task that was successfully placed on your cluster are described here.

Type: array of Task (p. 139) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request runs the latest `ACTIVE` revision of the `hello_world` task definition family in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 45
X-Amz-Target: AmazonEC2ContainerServiceV20141113.RunTask
X-Amz-Date: 20161121T215740Z
User-Agent: aws-cli/1.11.13 Python/2.7.12 Darwin/16.1.0 botocore/1.4.66
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "count": 1,
  "taskDefinition": "hello_world"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 21 Nov 2016 21:57:40 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1025
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "failures": [],
  "tasks": [
    {
      "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
      "containerInstanceArn": "arn:aws:ecs:us-east-1:012345678910:container-
instance/4c543eed-f83f-47da-b1d8-3d23f1da4c64",
      "containers": [
        {
```

```
          "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
e76594d4-27e1-4c74-98b5-46a6435eb769",
          "lastStatus": "PENDING",
          "name": "wordpress",
          "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/
fdf2c302-468c-4e55-b884-5331d816e7fb"
        },
        {
          "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
b19106ea-4fa8-4f1d-9767-96922c82b070",
          "lastStatus": "PENDING",
          "name": "mysql",
          "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/
fdf2c302-468c-4e55-b884-5331d816e7fb"
        }
      ],
      "createdAt": 1479765460.842,
      "desiredStatus": "RUNNING",
      "lastStatus": "PENDING",
      "overrides": {
        "containerOverrides": [
          {
            "name": "wordpress"
          },
          {
            "name": "mysql"
          }
        ]
      },
      "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/fdf2c302-468c-4e55-
b884-5331d816e7fb",
      "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-
definition/hello_world:6",
      "version": 1
    }
  ]
}
```

StartTask

Starts a new task from the specified task definition on the specified container instance or instances. To use the default Amazon ECS scheduler to place your task, use `RunTask` instead.

Important
The list of container instances to start tasks on is limited to 10.

Request Syntax

```
{
   "cluster": "string",
   "containerInstances": [ "string" ],
   "overrides": {
      "containerOverrides": [
         {
            "command": [ "string" ],
            "environment": [
               {
                  "name": "string",
                  "value": "string"
               }
            ],
            "name": "string"
         }
      ],
      "taskRoleArn": "string"
   },
   "startedBy": "string",
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 88)
The short name or full Amazon Resource Name (ARN) of the cluster on which to start your task. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstances (p. 88)
The container instance IDs or full Amazon Resource Name (ARN) entries for the container instances on which you would like to place your task.

Important
The list of container instances to start tasks on is limited to 10.

Type: array of Strings

Required: Yes

overrides (p. 88)
A list of container overrides in JSON format that specify the name of a container in the specified task definition and the overrides it should receive. You can override the default command for a

container (that is specified in the task definition or Docker image) with a `command` override. You can also override existing environment variables (that are specified in the task definition or Docker image) on a container or add new environment variables to it with an `environment` override.

Note
A total of 8192 characters are allowed for overrides. This limit includes the JSON formatting characters of the override structure.

Type: TaskOverride (p. 143) object

Required: No

startedBy (p. 88)

An optional tag specified when a task is started. For example if you automatically trigger a task to run a batch process job, you could apply a unique identifier for that job to your task with the `startedBy` parameter. You can then identify which tasks belong to that job by filtering the results of a ListTasks (p. 69) call with the `startedBy` value. Up to 36 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

If a task is started by an Amazon ECS service, then the `startedBy` parameter contains the deployment ID of the service that starts it.

Type: String

Required: No

taskDefinition (p. 88)

The `family` and `revision` (`family:revision`) or full Amazon Resource Name (ARN) of the task definition to start. If a `revision` is not specified, the latest `ACTIVE` revision is used.

Type: String

Required: Yes

Response Syntax

```
{
   "failures": [
      {
         "arn": "string",
         "reason": "string"
      }
   ],
   "tasks": [
      {
         "clusterArn": "string",
         "containerInstanceArn": "string",
         "containers": [
            {
               "containerArn": "string",
               "exitCode": number,
               "lastStatus": "string",
               "name": "string",
               "networkBindings": [
                  {
                     "bindIP": "string",
                     "containerPort": number,
                     "hostPort": number,
                     "protocol": "string"
                  }
               ],
               "reason": "string",
               "taskArn": "string"
            }
```

```
         ],
         "createdAt": number,
         "desiredStatus": "string",
         "lastStatus": "string",
         "overrides": {
            "containerOverrides": [
               {
                  "command": [ "string" ],
                  "environment": [
                     {
                        "name": "string",
                        "value": "string"
                     }
                  ],
                  "name": "string"
               }
            ],
            "taskRoleArn": "string"
         },
         "startedAt": number,
         "startedBy": "string",
         "stoppedAt": number,
         "stoppedReason": "string",
         "taskArn": "string",
         "taskDefinitionArn": "string",
         "version": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

failures (p. 89)

Any failures associated with the call.

Type: array of Failure (p. 126) objects

tasks (p. 89)

A full description of the tasks that were started. Each task that was successfully placed on your container instances are described here.

Type: array of Task (p. 139) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request starts the latest `ACTIVE` revision of the `hello_world` task definition family in the default cluster on the container instance with the ID `4c543eed-f83f-47da-b1d8-3d23f1da4c64`.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 97
X-Amz-Target: AmazonEC2ContainerServiceV20141113.StartTask
X-Amz-Date: 20161121T220032Z
User-Agent: aws-cli/1.11.13 Python/2.7.12 Darwin/16.1.0 botocore/1.4.66
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "containerInstances": [
    "4c543eed-f83f-47da-b1d8-3d23f1da4c64"
  ],
  "taskDefinition": "hello_world"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 21 Nov 2016 22:00:32 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1025
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "failures": [],
  "tasks": [
    {
```

```
      "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
      "containerInstanceArn": "arn:aws:ecs:us-east-1:012345678910:container-
instance/4c543eed-f83f-47da-b1d8-3d23f1da4c64",
      "containers": [
        {
          "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
e76594d4-27e1-4c74-98b5-46a6435eb769",
          "lastStatus": "PENDING",
          "name": "wordpress",
          "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/
fdf2c302-468c-4e55-b884-5331d816e7fb"
        },
        {
          "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
b19106ea-4fa8-4f1d-9767-96922c82b070",
          "lastStatus": "PENDING",
          "name": "mysql",
          "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/
fdf2c302-468c-4e55-b884-5331d816e7fb"
        }
      ],
      "createdAt": 1479765460.842,
      "desiredStatus": "RUNNING",
      "lastStatus": "PENDING",
      "overrides": {
        "containerOverrides": [
          {
            "name": "wordpress"
          },
          {
            "name": "mysql"
          }
        ]
      },
      "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/fdf2c302-468c-4e55-
b884-5331d816e7fb",
      "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-
definition/hello_world:6",
      "version": 1
    }
  ]
}
```

StopTask

Stops a running task.

When StopTask (p. 93) is called on a task, the equivalent of `docker stop` is issued to the containers running in the task. This results in a `SIGTERM` and a 30-second timeout, after which `SIGKILL` is sent and the containers are forcibly stopped. If the container handles the `SIGTERM` gracefully and exits within 30 seconds from receiving it, no `SIGKILL` is sent.

Request Syntax

```
{
   "cluster": "string",
   "reason": "string",
   "task": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 93)

The short name or full Amazon Resource Name (ARN) of the cluster that hosts the task to stop. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

reason (p. 93)

An optional message specified when a task is stopped. For example, if you are using a custom scheduler, you can use this parameter to specify the reason for stopping the task here, and the message will appear in subsequent DescribeTasks (p. 47) API operations on this task. Up to 255 characters are allowed in this message.

Type: String

Required: No

task (p. 93)

The task ID or full Amazon Resource Name (ARN) entry of the task to stop.

Type: String

Required: Yes

Response Syntax

```
{
   "task": {
      "clusterArn": "string",
      "containerInstanceArn": "string",
      "containers": [
         {
            "containerArn": "string",
            "exitCode": number,
            "lastStatus": "string",
            "name": "string",
            "networkBindings": [
```

```
                {
                   "bindIP": "string",
                   "containerPort": number,
                   "hostPort": number,
                   "protocol": "string"
                }
             ],
             "reason": "string",
             "taskArn": "string"
          }
       ],
       "createdAt": number,
       "desiredStatus": "string",
       "lastStatus": "string",
       "overrides": {
          "containerOverrides": [
             {
                "command": [ "string" ],
                "environment": [
                   {
                      "name": "string",
                      "value": "string"
                   }
                ],
                "name": "string"
             }
          ],
          "taskRoleArn": "string"
       },
       "startedAt": number,
       "startedBy": "string",
       "stoppedAt": number,
       "stoppedReason": "string",
       "taskArn": "string",
       "taskDefinitionArn": "string",
       "version": number
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

task (p. 93)

The task that was stopped.

Type: Task (p. 139) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request stops a task with the ID `a126249b-b7e4-4b06-9d8f-1b56e75a99b5` in the default cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 88
X-Amz-Target: AmazonEC2ContainerServiceV20141113.StopTask
X-Amz-Date: 20161121T220318Z
User-Agent: aws-cli/1.11.13 Python/2.7.12 Darwin/16.1.0 botocore/1.4.66
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "task": "1dc5c17a-422b-4dc4-b493-371970c6c4d6"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 21 Nov 2016 22:03:18 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1260
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "task": {
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
```

```
    "containerInstanceArn": "arn:aws:ecs:us-east-1:012345678910:container-
instance/5991d8da-1d59-49d2-a31f-4230f9e73140",
    "containers": [
      {
        "containerArn": "arn:aws:ecs:us-
east-1:012345678910:container/4df26bb4-f057-467b-a079-961675296e64",
        "lastStatus": "RUNNING",
        "name": "simple-app",
        "networkBindings": [
          {
            "bindIP": "0.0.0.0",
            "containerPort": 80,
            "hostPort": 32774,
            "protocol": "tcp"
          }
        ],
        "taskArn": "arn:aws:ecs:us-
east-1:012345678910:task/1dc5c17a-422b-4dc4-b493-371970c6c4d6"
      },
      {
        "containerArn": "arn:aws:ecs:us-east-1:012345678910:container/
e09064f7-7361-4c87-8ab9-8d073bbdbcb9",
        "lastStatus": "RUNNING",
        "name": "busybox",
        "networkBindings": [],
        "taskArn": "arn:aws:ecs:us-
east-1:012345678910:task/1dc5c17a-422b-4dc4-b493-371970c6c4d6"
      }
    ],
    "createdAt": 1476822811.295,
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
    "overrides": {
      "containerOverrides": [
        {
          "name": "simple-app"
        },
        {
          "name": "busybox"
        }
      ]
    },
    "startedAt": 1476822833.998,
    "startedBy": "ecs-svc/9223370560032507596",
    "stoppedReason": "Task stopped by user",
    "taskArn": "arn:aws:ecs:us-east-1:012345678910:task/1dc5c17a-422b-4dc4-
b493-371970c6c4d6",
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-definition/
console-sample-app-dynamic-ports:1",
    "version": 0
  }
}
```

SubmitContainerStateChange

Note
This action is only used by the Amazon EC2 Container Service agent, and it is not intended for use outside of the agent.

Sent to acknowledge that a container changed states.

Request Syntax

```
{
   "cluster": "string",
   "containerName": "string",
   "exitCode": number,
   "networkBindings": [
      {
         "bindIP": "string",
         "containerPort": number,
         "hostPort": number,
         "protocol": "string"
      }
   ],
   "reason": "string",
   "status": "string",
   "task": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 97)
The short name or full Amazon Resource Name (ARN) of the cluster that hosts the container.
Type: String
Required: No

containerName (p. 97)
The name of the container.
Type: String
Required: No

exitCode (p. 97)
The exit code returned for the state change request.
Type: Integer
Required: No

networkBindings (p. 97)
The network bindings of the container.
Type: array of NetworkBinding (p. 133) objects
Required: No

reason (p. 97)
The reason for the state change request.
Type: String
Required: No

status (p. 97)
The status of the state change request.
Type: String
Required: No

task (p. 97)
The task ID or full Amazon Resource Name (ARN) of the task that hosts the container.
Type: String
Required: No

Response Syntax

```
{
   "acknowledgment": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

acknowledgment (p. 98)
Acknowledgement of the state change.
Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException
These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.
HTTP Status Code: 400

ServerException
These errors are usually caused by a server issue.
HTTP Status Code: 500

SubmitTaskStateChange

Note
This action is only used by the Amazon EC2 Container Service agent, and it is not intended for use outside of the agent.

Sent to acknowledge that a task changed states.

Request Syntax

```
{
   "cluster": "string",
   "reason": "string",
   "status": "string",
   "task": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 99)
The short name or full Amazon Resource Name (ARN) of the cluster that hosts the task.
Type: String
Required: No

reason (p. 99)
The reason for the state change request.
Type: String
Required: No

status (p. 99)
The status of the state change request.
Type: String
Required: No

task (p. 99)
The task ID or full Amazon Resource Name (ARN) of the task in the state change request.
Type: String
Required: No

Response Syntax

```
{
   "acknowledgment": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

acknowledgment (p. 99)

Acknowledgement of the state change.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

UpdateContainerAgent

Updates the Amazon ECS container agent on a specified container instance. Updating the Amazon ECS container agent does not interrupt running tasks or services on the container instance. The process for updating the agent differs depending on whether your container instance was launched with the Amazon ECS-optimized AMI or another operating system.

`UpdateContainerAgent` requires the Amazon ECS-optimized AMI or Amazon Linux with the `ecs-init` service installed and running. For help updating the Amazon ECS container agent on other operating systems, see Manually Updating the Amazon ECS Container Agent in the *Amazon EC2 Container Service Developer Guide*.

Request Syntax

```
{
   "cluster": "string",
   "containerInstance": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 101)

The short name or full Amazon Resource Name (ARN) of the cluster that your container instance is running on. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

containerInstance (p. 101)

The container instance ID or full Amazon Resource Name (ARN) entries for the container instance on which you would like to update the Amazon ECS container agent.

Type: String

Required: Yes

Response Syntax

```
{
   "containerInstance": {
      "agentConnected": boolean,
      "agentUpdateStatus": "string",
      "attributes": [
         {
            "name": "string",
            "value": "string"
         }
      ],
      "containerInstanceArn": "string",
      "ec2InstanceId": "string",
      "pendingTasksCount": number,
      "registeredResources": [
```

```
         {
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "remainingResources": [
         {
            "doubleValue": number,
            "integerValue": number,
            "longValue": number,
            "name": "string",
            "stringSetValue": [ "string" ],
            "type": "string"
         }
      ],
      "runningTasksCount": number,
      "status": "string",
      "version": number,
      "versionInfo": {
         "agentHash": "string",
         "agentVersion": "string",
         "dockerVersion": "string"
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

containerInstance (p. 101)

The container instance for which the container agent was updated.

Type: ContainerInstance (p. 121) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

MissingVersionException

Amazon ECS is unable to determine the current version of the Amazon ECS container agent on the container instance and does not have enough information to proceed with an update. This could be because the agent running on the container instance is an older or custom version that does not use our version information.

HTTP Status Code: 400

NoUpdateAvailableException

There is no update available for this Amazon ECS container agent. This could be because the agent is already running the latest version, or it is so old that there is no update path to the current version.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

UpdateInProgressException

There is already a current Amazon ECS container agent update in progress on the specified container instance. If the container agent becomes disconnected while it is in a transitional stage, such as `PENDING` or `STAGING`, the update process can get stuck in that state. However, when the agent reconnects, it resumes where it stopped previously.

HTTP Status Code: 400

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example updates the container agent version for the container instance with the ID `53ac7152-dcd1-4102-81f5-208962864132` in the `update` cluster.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 82
X-Amz-Target: AmazonEC2ContainerServiceV20141113.UpdateContainerAgent
X-Amz-Date: 20150528T152756Z
User-Agent: aws-cli/1.7.30 Python/2.7.9 Darwin/14.3.0
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "cluster": "update",
  "containerInstance": "53ac7152-dcd1-4102-81f5-208962864132"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Thu, 28 May 2015 15:27:54 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1033
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "containerInstance": {
    "agentConnected": true,
    "agentUpdateStatus": "PENDING",
...
    "versionInfo": {
      "agentHash": "4023248",
      "agentVersion": "1.0.0",
      "dockerVersion": "DockerVersion: 1.5.0"
    }
  }
}
```

UpdateService

Modifies the desired count, deployment configuration, or task definition used in a service.

You can add to or subtract from the number of instantiations of a task definition in a service by specifying the cluster that the service is running in and a new `desiredCount` parameter.

You can use UpdateService (p. 105) to modify your task definition and deploy a new version of your service.

You can also update the deployment configuration of a service. When a deployment is triggered by updating the task definition of a service, the service scheduler uses the deployment configuration parameters, `minimumHealthyPercent` and `maximumPercent`, to determine the deployment strategy.

If the `minimumHealthyPercent` is below 100%, the scheduler can ignore the `desiredCount` temporarily during a deployment. For example, if your service has a `desiredCount` of four tasks, a `minimumHealthyPercent` of 50% allows the scheduler to stop two existing tasks before starting two new tasks. Tasks for services that *do not* use a load balancer are considered healthy if they are in the `RUNNING` state; tasks for services that *do* use a load balancer are considered healthy if they are in the `RUNNING` state and the container instance it is hosted on is reported as healthy by the load balancer.

The `maximumPercent` parameter represents an upper limit on the number of running tasks during a deployment, which enables you to define the deployment batch size. For example, if your service has a `desiredCount` of four tasks, a `maximumPercent` value of 200% starts four new tasks before stopping the four older tasks (provided that the cluster resources required to do this are available).

When UpdateService (p. 105) stops a task during a deployment, the equivalent of `docker stop` is issued to the containers running in the task. This results in a `SIGTERM` and a 30-second timeout, after which `SIGKILL` is sent and the containers are forcibly stopped. If the container handles the `SIGTERM` gracefully and exits within 30 seconds from receiving it, no `SIGKILL` is sent.

When the service scheduler launches new tasks, it attempts to balance them across the Availability Zones in your cluster with the following logic:

- Determine which of the container instances in your cluster can support your service's task definition (for example, they have the required CPU, memory, ports, and container instance attributes).
- Sort the valid container instances by the fewest number of running tasks for this service in the same Availability Zone as the instance. For example, if zone A has one running service task and zones B and C each have zero, valid container instances in either zone B or C are considered optimal for placement.
- Place the new service task on a valid container instance in an optimal Availability Zone (based on the previous steps), favoring container instances with the fewest number of running tasks for this service.

Request Syntax

```
{
   "cluster": "string",
   "deploymentConfiguration": {
      "maximumPercent": number,
      "minimumHealthyPercent": number
   },
   "desiredCount": number,
   "service": "string",
   "taskDefinition": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 148).

The request accepts the following data in JSON format.

cluster (p. 105)
The short name or full Amazon Resource Name (ARN) of the cluster that your service is running on. If you do not specify a cluster, the default cluster is assumed.

Type: String

Required: No

deploymentConfiguration (p. 105)
Optional deployment parameters that control how many tasks run during the deployment and the ordering of stopping and starting tasks.

Type: DeploymentConfiguration (p. 125) object

Required: No

desiredCount (p. 105)
The number of instantiations of the task to place and keep running in your service.

Type: Integer

Required: No

service (p. 105)
The name of the service to update.

Type: String

Required: Yes

taskDefinition (p. 105)
The `family` and `revision` (`family:revision`) or full Amazon Resource Name (ARN) of the task definition to run in your service. If a `revision` is not specified, the latest `ACTIVE` revision is used. If you modify the task definition with `UpdateService`, Amazon ECS spawns a task with the new version of the task definition and then stops an old task after the new version is running.

Type: String

Required: No

Response Syntax

```
{
   "service": {
      "clusterArn": "string",
      "createdAt": number,
      "deploymentConfiguration": {
         "maximumPercent": number,
         "minimumHealthyPercent": number
      },
      "deployments": [
         {
            "createdAt": number,
            "desiredCount": number,
            "id": "string",
            "pendingCount": number,
            "runningCount": number,
            "status": "string",
            "taskDefinition": "string",
            "updatedAt": number
```

```
            }
         ],
         "desiredCount": number,
         "events": [
            {
               "createdAt": number,
               "id": "string",
               "message": "string"
            }
         ],
         "loadBalancers": [
            {
               "containerName": "string",
               "containerPort": number,
               "loadBalancerName": "string",
               "targetGroupArn": "string"
            }
         ],
         "pendingCount": number,
         "roleArn": "string",
         "runningCount": number,
         "serviceArn": "string",
         "serviceName": "string",
         "status": "string",
         "taskDefinition": "string"
      }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

service (p. 106)

The full description of your service following the update call.

Type: Service (p. 136) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 150).

ClientException

These errors are usually caused by a client action, such as using an action or resource on behalf of a user that doesn't have permission to use the action or resource, or specifying an identifier that is not valid.

HTTP Status Code: 400

ClusterNotFoundException

The specified cluster could not be found. You can view your available clusters with ListClusters (p. 53). Amazon ECS clusters are region-specific.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server issue.

HTTP Status Code: 500

ServiceNotActiveException

The specified service is not active. You cannot update a service that is not active. If you have previously deleted a service, you can re-create it with CreateService (p. 6).

HTTP Status Code: 400

ServiceNotFoundException

The specified service could not be found. You can view your available services with ListServices (p. 59). Amazon ECS services are cluster-specific and region-specific.

HTTP Status Code: 400

Example

In the following example or examples, the Authorization header contents (`AUTHPARAMS`) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the *AWS General Reference*.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example request updates the `hello_world` service to a desired count of 3.

Sample Request

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 45
X-Amz-Target: AmazonEC2ContainerServiceV20141113.UpdateService
X-Amz-Date: 20150429T194543Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "service": "hello_world",
  "desiredCount": 3
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Wed, 29 Apr 2015 19:45:43 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 13376
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
  "service": {
    "clusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/default",
    "deploymentConfiguration": {
        "maximumPercent": 200,
```

```
        "minimumHealthyPercent": 100
    },
    "deployments": [
      {
        "createdAt": 1430333711.033,
        "desiredCount": 3,
        "id": "ecs-svc/9223370606521064774",
        "pendingCount": 0,
        "runningCount": 0,
        "status": "PRIMARY",
        "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-
definition/hello_world:10",
        "updatedAt": 1430336267.173
      }
    ],
    "desiredCount": 3,
    "events": [],
    "loadBalancers": [],
    "pendingCount": 0,
    "runningCount": 0,
    "serviceArn": "arn:aws:ecs:us-east-1:012345678910:service/hello_world",
    "serviceName": "hello_world",
    "status": "ACTIVE",
    "taskDefinition": "arn:aws:ecs:us-east-1:012345678910:task-definition/
hello_world:10"
  }
}
```

Data Types

The Amazon EC2 Container Service API contains several data types that various actions use. This section describes each data type in detail.

Note
The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

- Volume (p. 146)
- VolumeFrom (p. 147)

Attribute

The attributes applicable to a container instance when it is registered.

Contents

name

The name of the container instance attribute.

Type: String

Required: Yes

value

The value of the container instance attribute (at this time, the value here is `Null`, but this could change in future revisions for expandability).

Type: String

Required: No

Cluster

A regional grouping of one or more container instances on which you can run task requests. Each account receives a default cluster the first time you use the Amazon ECS service, but you may also create other clusters. Clusters may contain more than one instance type simultaneously.

Contents

activeServicesCount
The number of services that are running on the cluster in an `ACTIVE` state. You can view these services with ListServices (p. 59).
Type: Integer
Required: No

clusterArn
The Amazon Resource Name (ARN) that identifies the cluster. The ARN contains the `arn:aws:ecs` namespace, followed by the region of the cluster, the AWS account ID of the cluster owner, the `cluster` namespace, and then the cluster name. For example, `arn:aws:ecs:`*`region`*`:`*`012345678910`*`:cluster/`*`test`* ..
Type: String
Required: No

clusterName
A user-generated string that you use to identify your cluster.
Type: String
Required: No

pendingTasksCount
The number of tasks in the cluster that are in the `PENDING` state.
Type: Integer
Required: No

registeredContainerInstancesCount
The number of container instances registered into the cluster.
Type: Integer
Required: No

runningTasksCount
The number of tasks in the cluster that are in the `RUNNING` state.
Type: Integer
Required: No

status
The status of the cluster. The valid values are `ACTIVE` or `INACTIVE`. `ACTIVE` indicates that you can register container instances with the cluster and the associated instances can accept tasks.
Type: String
Required: No

Container

A Docker container that is part of a task.

Contents

containerArn
The Amazon Resource Name (ARN) of the container.
Type: String
Required: No

exitCode
The exit code returned from the container.
Type: Integer
Required: No

lastStatus
The last known status of the container.
Type: String
Required: No

name
The name of the container.
Type: String
Required: No

networkBindings
The network bindings associated with the container.
Type: array of NetworkBinding (p. 133) objects
Required: No

reason
A short (255 max characters) human-readable string to provide additional detail about a running or stopped container.
Type: String
Required: No

taskArn
The Amazon Resource Name (ARN) of the task.
Type: String
Required: No

ContainerDefinition

Container definitions are used in task definitions to describe the different containers that are launched as part of a task.

Contents

command

The command that is passed to the container. This parameter maps to `Cmd` in the Create a container section of the Docker Remote API and the `COMMAND` parameter to docker run. For more information, see https://docs.docker.com/engine/reference/builder/#cmd.

Type: array of Strings

Required: No

cpu

The number of `cpu` units reserved for the container. A container instance has 1,024 `cpu` units for every CPU core. This parameter specifies the minimum amount of CPU to reserve for a container, and containers share unallocated CPU units with other containers on the instance with the same ratio as their allocated amount. This parameter maps to `CpuShares` in the Create a container section of the Docker Remote API and the `--cpu-shares` option to docker run.

> **Note**
> You can determine the number of CPU units that are available per EC2 instance type by multiplying the vCPUs listed for that instance type on the Amazon EC2 Instances detail page by 1,024.

For example, if you run a single-container task on a single-core instance type with 512 CPU units specified for that container, and that is the only task running on the container instance, that container could use the full 1,024 CPU unit share at any given time. However, if you launched another copy of the same task on that container instance, each task would be guaranteed a minimum of 512 CPU units when needed, and each container could float to higher CPU usage if the other container was not using it, but if both tasks were 100% active all of the time, they would be limited to 512 CPU units.

The Docker daemon on the container instance uses the CPU value to calculate the relative CPU share ratios for running containers. For more information, see CPU share constraint in the Docker documentation. The minimum valid CPU share value that the Linux kernel allows is 2; however, the CPU parameter is not required, and you can use CPU values below 2 in your container definitions. For CPU values below 2 (including null), the behavior varies based on your Amazon ECS container agent version:

- **Agent versions less than or equal to 1.1.0:** Null and zero CPU values are passed to Docker as 0, which Docker then converts to 1,024 CPU shares. CPU values of 1 are passed to Docker as 1, which the Linux kernel converts to 2 CPU shares.
- **Agent versions greater than or equal to 1.2.0:** Null, zero, and CPU values of 1 are passed to Docker as 2.

Type: Integer

Required: No

disableNetworking

When this parameter is true, networking is disabled within the container. This parameter maps to `NetworkDisabled` in the Create a container section of the Docker Remote API.

Type: Boolean

Required: No

dnsSearchDomains

A list of DNS search domains that are presented to the container. This parameter maps to `DnsSearch` in the Create a container section of the Docker Remote API and the `--dns-search` option to docker run.

Type: array of Strings

Required: No

dnsServers

A list of DNS servers that are presented to the container. This parameter maps to `Dns` in the Create a container section of the Docker Remote API and the `--dns` option to docker run.

Type: array of Strings

Required: No

dockerLabels

A key/value map of labels to add to the container. This parameter maps to `Labels` in the Create a container section of the Docker Remote API and the `--label` option to docker run. This parameter requires version 1.18 of the Docker Remote API or greater on your container instance. To check the Docker Remote API version on your container instance, log into your container instance and run the following command: `sudo docker version | grep "Server API version"`

Type: String to String map

Required: No

dockerSecurityOptions

A list of strings to provide custom labels for SELinux and AppArmor multi-level security systems. This parameter maps to `SecurityOpt` in the Create a container section of the Docker Remote API and the `--security-opt` option to docker run.

> **Note**
> The Amazon ECS container agent running on a container instance must register with the `ECS_SELINUX_CAPABLE=true` or `ECS_APPARMOR_CAPABLE=true` environment variables before containers placed on that instance can use these security options. For more information, see Amazon ECS Container Agent Configuration in the *Amazon EC2 Container Service Developer Guide*.

Type: array of Strings

Required: No

entryPoint

> **Important**
> Early versions of the Amazon ECS container agent do not properly handle `entryPoint` parameters. If you have problems using `entryPoint`, update your container agent or enter your commands and arguments as `command` array items instead.

The entry point that is passed to the container. This parameter maps to `Entrypoint` in the Create a container section of the Docker Remote API and the `--entrypoint` option to docker run. For more information, see https://docs.docker.com/engine/reference/builder/#entrypoint.

Type: array of Strings

Required: No

environment

The environment variables to pass to a container. This parameter maps to `Env` in the Create a container section of the Docker Remote API and the `--env` option to docker run.

> **Important**
> We do not recommend using plain text environment variables for sensitive information, such as credential data.

Type: array of KeyValuePair (p. 129) objects

Required: No

essential

If the `essential` parameter of a container is marked as `true`, and that container fails or stops for any reason, all other containers that are part of the task are stopped. If the `essential` parameter of a container is marked as `false`, then its failure does not affect the rest of the containers in a task. If this parameter is omitted, a container is assumed to be essential.

All tasks must have at least one essential container. If you have an application that is composed of multiple containers, you should group containers that are used for a common purpose into

components, and separate the different components into multiple task definitions. For more information, see Application Architecture in the *Amazon EC2 Container Service Developer Guide*.

Type: Boolean

Required: No

extraHosts

A list of hostnames and IP address mappings to append to the `/etc/hosts` file on the container. This parameter maps to `ExtraHosts` in the Create a container section of the Docker Remote API and the `--add-host` option to docker run.

Type: array of HostEntry (p. 127) objects

Required: No

hostname

The hostname to use for your container. This parameter maps to `Hostname` in the Create a container section of the Docker Remote API and the `--hostname` option to docker run.

Type: String

Required: No

image

The image used to start a container. This string is passed directly to the Docker daemon. Images in the Docker Hub registry are available by default. Other repositories are specified with *`repository-url/image:tag`* . Up to 255 letters (uppercase and lowercase), numbers, hyphens, underscores, colons, periods, forward slashes, and number signs are allowed. This parameter maps to `Image` in the Create a container section of the Docker Remote API and the `IMAGE` parameter of docker run.

- Images in official repositories on Docker Hub use a single name (for example, `ubuntu` or `mongo`).
- Images in other repositories on Docker Hub are qualified with an organization name (for example, `amazon/amazon-ecs-agent`).
- Images in other online repositories are qualified further by a domain name (for example, `quay.io/assemblyline/ubuntu`).

Type: String

Required: No

links

The `link` parameter allows containers to communicate with each other without the need for port mappings, using the `name` parameter and optionally, an `alias` for the link. This construct is analogous to `name:alias` in Docker links. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed for each `name` and `alias`. For more information on linking Docker containers, see https://docs.docker.com/engine/userguide/networking/default_network/dockerlinks/. This parameter maps to `Links` in the Create a container section of the Docker Remote API and the `--link` option to docker run.

> **Important**
> Containers that are collocated on a single container instance may be able to communicate with each other without requiring links or host port mappings. Network isolation is achieved on the container instance using security groups and VPC settings.

Type: array of Strings

Required: No

logConfiguration

The log configuration specification for the container. This parameter maps to `LogConfig` in the Create a container section of the Docker Remote API and the `--log-driver` option to docker run. By default, containers use the same logging driver that the Docker daemon uses; however the container may use a different logging driver than the Docker daemon by specifying a log driver with this parameter in the container definition. To use a different logging driver for a container, the log system must be configured properly on the container instance (or on a different log server for remote logging options). For more information on the options for different supported log drivers, see Configure logging drivers in the Docker documentation.

Note
Amazon ECS currently supports a subset of the logging drivers available to the Docker daemon (shown in the LogConfiguration (p. 131) data type). Currently unsupported log drivers may be available in future releases of the Amazon ECS container agent.

This parameter requires version 1.18 of the Docker Remote API or greater on your container instance. To check the Docker Remote API version on your container instance, log into your container instance and run the following command: `sudo docker version | grep "Server API version"`

Note
The Amazon ECS container agent running on a container instance must register the logging drivers available on that instance with the `ECS_AVAILABLE_LOGGING_DRIVERS` environment variable before containers placed on that instance can use these log configuration options. For more information, see Amazon ECS Container Agent Configuration in the *Amazon EC2 Container Service Developer Guide*.

Type: LogConfiguration (p. 131) object

Required: No

memory

The hard limit (in MiB) of memory to present to the container. If your container attempts to exceed the memory specified here, the container is killed. This parameter maps to `Memory` in the Create a container section of the Docker Remote API and the `--memory` option to docker run.

You must specify a non-zero integer for one or both of `memory` or `memoryReservation` in container definitions. If you specify both, `memory` must be greater than `memoryReservation`. If you specify `memoryReservation`, then that value is subtracted from the available memory resources for the container instance on which the container is placed; otherwise, the value of `memory` is used.

The Docker daemon reserves a minimum of 4 MiB of memory for a container, so you should not specify fewer than 4 MiB of memory for your containers.

Type: Integer

Required: No

memoryReservation

The soft limit (in MiB) of memory to reserve for the container. When system memory is under heavy contention, Docker attempts to keep the container memory to this soft limit; however, your container can consume more memory when it needs to, up to either the hard limit specified with the `memory` parameter (if applicable), or all of the available memory on the container instance, whichever comes first. This parameter maps to `MemoryReservation` in the Create a container section of the Docker Remote API and the `--memory-reservation` option to docker run.

You must specify a non-zero integer for one or both of `memory` or `memoryReservation` in container definitions. If you specify both, `memory` must be greater than `memoryReservation`. If you specify `memoryReservation`, then that value is subtracted from the available memory resources for the container instance on which the container is placed; otherwise, the value of `memory` is used.

For example, if your container normally uses 128 MiB of memory, but occasionally bursts to 256 MiB of memory for short periods of time, you can set a `memoryReservation` of 128 MiB, and a `memory` hard limit of 300 MiB. This configuration would allow the container to only reserve 128 MiB of memory from the remaining resources on the container instance, but also allow the container to consume more memory resources when needed.

Type: Integer

Required: No

mountPoints

The mount points for data volumes in your container. This parameter maps to `Volumes` in the Create a container section of the Docker Remote API and the `--volume` option to docker run.

Type: array of MountPoint (p. 132) objects

Required: No

name

The name of a container. If you are linking multiple containers together in a task definition, the `name` of one container can be entered in the `links` of another container to connect the containers. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed. This parameter maps to `name` in the Create a container section of the Docker Remote API and the `--name` option to docker run.

Type: String

Required: No

portMappings

The list of port mappings for the container. Port mappings allow containers to access ports on the host container instance to send or receive traffic. This parameter maps to `PortBindings` in the Create a container section of the Docker Remote API and the `--publish` option to docker run. If the network mode of a task definition is set to `none`, then you cannot specify port mappings. If the network mode of a task definition is set to `host`, then host ports must either be undefined or they must match the container port in the port mapping.

> **Note**
> After a task reaches the `RUNNING` status, manual and automatic host and container port assignments are visible in the **Network Bindings** section of a container description of a selected task in the Amazon ECS console, or the `networkBindings` section DescribeTasks (p. 47) responses.

Type: array of PortMapping (p. 134) objects

Required: No

privileged

When this parameter is true, the container is given elevated privileges on the host container instance (similar to the `root` user). This parameter maps to `Privileged` in the Create a container section of the Docker Remote API and the `--privileged` option to docker run.

Type: Boolean

Required: No

readonlyRootFilesystem

When this parameter is true, the container is given read-only access to its root file system. This parameter maps to `ReadonlyRootfs` in the Create a container section of the Docker Remote API and the `--read-only` option to `docker run`.

Type: Boolean

Required: No

ulimits

A list of `ulimits` to set in the container. This parameter maps to `Ulimits` in the Create a container section of the Docker Remote API and the `--ulimit` option to docker run. Valid naming values are displayed in the Ulimit (p. 144) data type. This parameter requires version 1.18 of the Docker Remote API or greater on your container instance. To check the Docker Remote API version on your container instance, log into your container instance and run the following command: `sudo docker version | grep "Server API version"`

Type: array of Ulimit (p. 144) objects

Required: No

user

The user name to use inside the container. This parameter maps to `User` in the Create a container section of the Docker Remote API and the `--user` option to docker run.

Type: String

Required: No

volumesFrom

Data volumes to mount from another container. This parameter maps to `VolumesFrom` in the Create a container section of the Docker Remote API and the `--volumes-from` option to docker run.

Type: array of VolumeFrom (p. 147) objects

Required: No

workingDirectory

The working directory in which to run commands inside the container. This parameter maps to `WorkingDir` in the Create a container section of the Docker Remote API and the `--workdir` option to docker run.

Type: String

Required: No

ContainerInstance

An EC2 instance that is running the Amazon ECS agent and has been registered with a cluster.

Contents

agentConnected

This parameter returns `true` if the agent is actually connected to Amazon ECS. Registered instances with an agent that may be unhealthy or stopped return `false`, and instances without a connected agent cannot accept placement requests.

Type: Boolean

Required: No

agentUpdateStatus

The status of the most recent agent update. If an update has never been requested, this value is `NULL`.

Type: String

Valid Values: `PENDING | STAGING | STAGED | UPDATING | UPDATED | FAILED`

Required: No

attributes

The attributes set for the container instance by the Amazon ECS container agent at instance registration.

Type: array of Attribute (p. 112) objects

Required: No

containerInstanceArn

The Amazon Resource Name (ARN) of the container instance. The ARN contains the `arn:aws:ecs` namespace, followed by the region of the container instance, the AWS account ID of the container instance owner, the `container-instance` namespace, and then the container instance ID. For example, `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:container-instance/`*`container_instance_ID`* .

Type: String

Required: No

ec2InstanceId

The EC2 instance ID of the container instance.

Type: String

Required: No

pendingTasksCount

The number of tasks on the container instance that are in the `PENDING` status.

Type: Integer

Required: No

registeredResources

For most resource types, this parameter describes the registered resources on the container instance that are in use by current tasks. For port resource types, this parameter describes the ports that were reserved by the Amazon ECS container agent when it registered the container instance with Amazon ECS.

Type: array of Resource (p. 135) objects

Required: No

remainingResources

For most resource types, this parameter describes the remaining resources of the container instance that are available for new tasks. For port resource types, this parameter describes the ports that are reserved by the Amazon ECS container agent and any containers that have reserved port mappings; any port that is not specified here is available for new tasks.

Type: array of Resource (p. 135) objects

Required: No

runningTasksCount

The number of tasks on the container instance that are in the `RUNNING` status.

Type: Integer

Required: No

status

The status of the container instance. The valid values are `ACTIVE` or `INACTIVE`. `ACTIVE` indicates that the container instance can accept tasks.

Type: String

Required: No

version

The version counter for the container instance. Every time a container instance experiences a change that triggers a CloudWatch event, the version counter is incremented. If you are replicating your Amazon ECS container instance state with CloudWatch events, you can compare the version of a container instance reported by the Amazon ECS APIs with the version reported in CloudWatch events for the container instance (inside the `detail` object) to verify that the version in your event stream is current.

Type: Long

Required: No

versionInfo

The version information for the Amazon ECS container agent and Docker daemon running on the container instance.

Type: VersionInfo (p. 145) object

Required: No

ContainerOverride

The overrides that should be sent to a container.

Contents

command

The command to send to the container that overrides the default command from the Docker image or the task definition.

Type: array of Strings

Required: No

environment

The environment variables to send to the container. You can add new environment variables, which are added to the container at launch, or you can override the existing environment variables from the Docker image or the task definition.

Type: array of KeyValuePair (p. 129) objects

Required: No

name

The name of the container that receives the override.

Type: String

Required: No

Deployment

The details of an Amazon ECS service deployment.

Contents

createdAt
The Unix timestamp for when the service was created.
Type: Timestamp
Required: No

desiredCount
The most recent desired count of tasks that was specified for the service to deploy or maintain.
Type: Integer
Required: No

id
The ID of the deployment.
Type: String
Required: No

pendingCount
The number of tasks in the deployment that are in the `PENDING` status.
Type: Integer
Required: No

runningCount
The number of tasks in the deployment that are in the `RUNNING` status.
Type: Integer
Required: No

status
The status of the deployment. Valid values are `PRIMARY` (for the most recent deployment), `ACTIVE` (for previous deployments that still have tasks running, but are being replaced with the `PRIMARY` deployment), and `INACTIVE` (for deployments that have been completely replaced).
Type: String
Required: No

taskDefinition
The most recent task definition that was specified for the service to use.
Type: String
Required: No

updatedAt
The Unix timestamp for when the service was last updated.
Type: Timestamp
Required: No

DeploymentConfiguration

Optional deployment parameters that control how many tasks run during the deployment and the ordering of stopping and starting tasks.

Contents

maximumPercent

The upper limit (as a percentage of the service's `desiredCount`) of the number of tasks that are allowed in the `RUNNING` or `PENDING` state in a service during a deployment. The maximum number of tasks during a deployment is the `desiredCount` multiplied by the `maximumPercent`/100, rounded down to the nearest integer value.

Type: Integer

Required: No

minimumHealthyPercent

The lower limit (as a percentage of the service's `desiredCount`) of the number of running tasks that must remain in the `RUNNING` state in a service during a deployment. The minimum healthy tasks during a deployment is the `desiredCount` multiplied by the `minimumHealthyPercent`/100, rounded up to the nearest integer value.

Type: Integer

Required: No

Failure

A failed resource.

Contents

arn
The Amazon Resource Name (ARN) of the failed resource.
Type: String
Required: No

reason
The reason for the failure.
Type: String
Required: No

HostEntry

Hostnames and IP address entries that are added to the `/etc/hosts` file of a container via the `extraHosts` parameter of its ContainerDefinition (p. 115).

Contents

hostname

The hostname to use in the `/etc/hosts` entry.

Type: String

Required: Yes

ipAddress

The IP address to use in the `/etc/hosts` entry.

Type: String

Required: Yes

HostVolumeProperties

Details on a container instance host volume.

Contents

sourcePath

The path on the host container instance that is presented to the container. If this parameter is empty, then the Docker daemon has assigned a host path for you. If the `host` parameter contains a `sourcePath` file location, then the data volume persists at the specified location on the host container instance until you delete it manually. If the `sourcePath` value does not exist on the host container instance, the Docker daemon creates it. If the location does exist, the contents of the source path folder are exported.

Type: String

Required: No

KeyValuePair

A key and value pair object.

Contents

name

The name of the key value pair. For environment variables, this is the name of the environment variable.

Type: String

Required: No

value

The value of the key value pair. For environment variables, this is the value of the environment variable.

Type: String

Required: No

LoadBalancer

Details on a load balancer that is used with a service.

Contents

containerName
The name of the container (as it appears in a container definition) to associate with the load balancer.
Type: String
Required: No

containerPort
The port on the container to associate with the load balancer. This port must correspond to a `containerPort` in the service's task definition. Your container instances must allow ingress traffic on the `hostPort` of the port mapping.
Type: Integer
Required: No

loadBalancerName
The name of the load balancer.
Type: String
Required: No

targetGroupArn
The full Amazon Resource Name (ARN) of the Elastic Load Balancing target group associated with a service.
Type: String
Required: No

LogConfiguration

Log configuration options to send to a custom log driver for the container.

Contents

logDriver

The log driver to use for the container. The valid values listed for this parameter are log drivers that the Amazon ECS container agent can communicate with by default.

> **Note**
> If you have a custom driver that is not listed above that you would like to work with the Amazon ECS container agent, you can fork the Amazon ECS container agent project that is available on GitHub and customize it to work with that driver. We encourage you to submit pull requests for changes that you would like to have included. However, Amazon Web Services does not currently provide support for running modified copies of this software.

This parameter requires version 1.18 of the Docker Remote API or greater on your container instance. To check the Docker Remote API version on your container instance, log into your container instance and run the following command: `sudo docker version | grep "Server API version"`

Type: String

Valid Values: `json-file | syslog | journald | gelf | fluentd | awslogs | splunk`

Required: Yes

options

The configuration options to send to the log driver. This parameter requires version 1.19 of the Docker Remote API or greater on your container instance. To check the Docker Remote API version on your container instance, log into your container instance and run the following command: `sudo docker version | grep "Server API version"`

Type: String to String map

Required: No

MountPoint

Details on a volume mount point that is used in a container definition.

Contents

containerPath

The path on the container to mount the host volume at.

Type: String

Required: No

readOnly

If this value is `true`, the container has read-only access to the volume. If this value is `false`, then the container can write to the volume. The default value is `false`.

Type: Boolean

Required: No

sourceVolume

The name of the volume to mount.

Type: String

Required: No

NetworkBinding

Details on the network bindings between a container and its host container instance. After a task reaches the `RUNNING` status, manual and automatic host and container port assignments are visible in the `networkBindings` section of DescribeTasks (p. 47) API responses.

Contents

bindIP
The IP address that the container is bound to on the container instance.
Type: String
Required: No

containerPort
The port number on the container that is be used with the network binding.
Type: Integer
Required: No

hostPort
The port number on the host that is used with the network binding.
Type: Integer
Required: No

protocol
The protocol used for the network binding.
Type: String
Valid Values: `tcp | udp`
Required: No

PortMapping

Port mappings allow containers to access ports on the host container instance to send or receive traffic. Port mappings are specified as part of the container definition. After a task reaches the `RUNNING` status, manual and automatic host and container port assignments are visible in the `networkBindings` section of DescribeTasks (p. 47) API responses.

Contents

containerPort

The port number on the container that is bound to the user-specified or automatically assigned host port. If you specify a container port and not a host port, your container automatically receives a host port in the ephemeral port range (for more information, see `hostPort`). Port mappings that are automatically assigned in this way do not count toward the 100 reserved ports limit of a container instance.

Type: Integer

Required: No

hostPort

The port number on the container instance to reserve for your container. You can specify a non-reserved host port for your container port mapping, or you can omit the `hostPort` (or set it to `0`) while specifying a `containerPort` and your container automatically receives a port in the ephemeral port range for your container instance operating system and Docker version.

The default ephemeral port range is 49153 to 65535, and this range is used for Docker versions prior to 1.6.0. For Docker version 1.6.0 and later, the Docker daemon tries to read the ephemeral port range from `/proc/sys/net/ipv4/ip_local_port_range`; if this kernel parameter is unavailable, the default ephemeral port range is used. You should not attempt to specify a host port in the ephemeral port range, because these are reserved for automatic assignment. In general, ports below 32768 are outside of the ephemeral port range.

The default reserved ports are 22 for SSH, the Docker ports 2375 and 2376, and the Amazon ECS container agent ports 51678 and 51679. Any host port that was previously specified in a running task is also reserved while the task is running (after a task stops, the host port is released).The current reserved ports are displayed in the `remainingResources` of DescribeContainerInstances (p. 32) output, and a container instance may have up to 100 reserved ports at a time, including the default reserved ports (automatically assigned ports do not count toward the 100 reserved ports limit).

Type: Integer

Required: No

protocol

The protocol used for the port mapping. Valid values are `tcp` and `udp`. The default is `tcp`.

Type: String

Valid Values: `tcp | udp`

Required: No

Resource

Describes the resources available for a container instance.

Contents

doubleValue
When the `doubleValue` type is set, the value of the resource must be a double precision floating-point type.
Type: Double
Required: No

integerValue
When the `integerValue` type is set, the value of the resource must be an integer.
Type: Integer
Required: No

longValue
When the `longValue` type is set, the value of the resource must be an extended precision floating-point type.
Type: Long
Required: No

name
The name of the resource, such as `CPU`, `MEMORY`, `PORTS`, or a user-defined resource.
Type: String
Required: No

stringSetValue
When the `stringSetValue` type is set, the value of the resource must be a string type.
Type: array of Strings
Required: No

type
The type of the resource, such as `INTEGER`, `DOUBLE`, `LONG`, or `STRINGSET`.
Type: String
Required: No

Service

Details on a service within a cluster

Contents

clusterArn

The Amazon Resource Name (ARN) of the cluster that hosts the service.

Type: String

Required: No

createdAt

The Unix timestamp for when the service was created.

Type: Timestamp

Required: No

deploymentConfiguration

Optional deployment parameters that control how many tasks run during the deployment and the ordering of stopping and starting tasks.

Type: DeploymentConfiguration (p. 125) object

Required: No

deployments

The current state of deployments for the service.

Type: array of Deployment (p. 124) objects

Required: No

desiredCount

The desired number of instantiations of the task definition to keep running on the service. This value is specified when the service is created with CreateService (p. 6), and it can be modified with UpdateService (p. 105).

Type: Integer

Required: No

events

The event stream for your service. A maximum of 100 of the latest events are displayed.

Type: array of ServiceEvent (p. 138) objects

Required: No

loadBalancers

A list of Elastic Load Balancing load balancer objects, containing the load balancer name, the container name (as it appears in a container definition), and the container port to access from the load balancer.

Type: array of LoadBalancer (p. 130) objects

Required: No

pendingCount

The number of tasks in the cluster that are in the `PENDING` state.

Type: Integer

Required: No

roleArn

The Amazon Resource Name (ARN) of the IAM role associated with the service that allows the Amazon ECS container agent to register container instances with an Elastic Load Balancing load balancer.

Type: String

Required: No

runningCount
The number of tasks in the cluster that are in the `RUNNING` state.

Type: Integer

Required: No

serviceArn
The Amazon Resource Name (ARN) that identifies the service. The ARN contains the `arn:aws:ecs` namespace, followed by the region of the service, the AWS account ID of the service owner, the `service` namespace, and then the service name. For example, `arn:aws:ecs:`*`region`*`:`*`012345678910`*`:service/`*`my-service`* .

Type: String

Required: No

serviceName
The name of your service. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed. Service names must be unique within a cluster, but you can have similarly named services in multiple clusters within a region or across multiple regions.

Type: String

Required: No

status
The status of the service. The valid values are `ACTIVE`, `DRAINING`, or `INACTIVE`.

Type: String

Required: No

taskDefinition
The task definition to use for tasks in the service. This value is specified when the service is created with CreateService (p. 6), and it can be modified with UpdateService (p. 105).

Type: String

Required: No

ServiceEvent

Details on an event associated with a service.

Contents

createdAt
The Unix timestamp for when the event was triggered.
Type: Timestamp
Required: No

id
The ID string of the event.
Type: String
Required: No

message
The event message.
Type: String
Required: No

Task

Details on a task in a cluster.

Contents

clusterArn
The Amazon Resource Name (ARN) of the cluster that hosts the task.
Type: String
Required: No

containerInstanceArn
The Amazon Resource Name (ARN) of the container instances that host the task.
Type: String
Required: No

containers
The containers associated with the task.
Type: array of Container (p. 114) objects
Required: No

createdAt
The Unix timestamp for when the task was created (the task entered the `PENDING` state).
Type: Timestamp
Required: No

desiredStatus
The desired status of the task.
Type: String
Required: No

lastStatus
The last known status of the task.
Type: String
Required: No

overrides
One or more container overrides.
Type: TaskOverride (p. 143) object
Required: No

startedAt
The Unix timestamp for when the task was started (the task transitioned from the `PENDING` state to the `RUNNING` state).
Type: Timestamp
Required: No

startedBy
The tag specified when a task is started. If the task is started by an Amazon ECS service, then the `startedBy` parameter contains the deployment ID of the service that starts it.
Type: String
Required: No

stoppedAt
The Unix timestamp for when the task was stopped (the task transitioned from the `RUNNING` state to the `STOPPED` state).
Type: Timestamp
Required: No

stoppedReason

The reason the task was stopped.

Type: String

Required: No

taskArn

The Amazon Resource Name (ARN) of the task.

Type: String

Required: No

taskDefinitionArn

The Amazon Resource Name (ARN) of the task definition that creates the task.

Type: String

Required: No

version

The version counter for the task. Every time a task experiences a change that triggers a CloudWatch event, the version counter is incremented. If you are replicating your Amazon ECS task state with CloudWatch events, you can compare the version of a task reported by the Amazon ECS APIs with the version reported in CloudWatch events for the task (inside the `detail` object) to verify that the version in your event stream is current.

Type: Long

Required: No

TaskDefinition

Details of a task definition.

Contents

containerDefinitions
A list of container definitions in JSON format that describe the different containers that make up your task. For more information about container definition parameters and defaults, see Amazon ECS Task Definitions in the *Amazon EC2 Container Service Developer Guide*.
Type: array of ContainerDefinition (p. 115) objects
Required: No

family
The family of your task definition, used as the definition name.
Type: String
Required: No

networkMode
The Docker networking mode to use for the containers in the task. The valid values are `none`, `bridge`, and `host`.
If the network mode is `none`, the containers do not have external connectivity. The default Docker network mode is `bridge`. The `host` network mode offers the highest networking performance for containers because it uses the host network stack instead of the virtualized network stack provided by the `bridge` mode.
For more information, see Network settings in the *Docker run reference*.
Type: String
Valid Values: `bridge | host | none`
Required: No

requiresAttributes
The container instance attributes required by your task.
Type: array of Attribute (p. 112) objects
Required: No

revision
The revision of the task in a particular family. The revision is a version number of a task definition in a family. When you register a task definition for the first time, the revision is `1`; each time you register a new revision of a task definition in the same family, the revision value always increases by one (even if you have deregistered previous revisions in this family).
Type: Integer
Required: No

status
The status of the task definition.
Type: String
Valid Values: `ACTIVE | INACTIVE`
Required: No

taskDefinitionArn
The full Amazon Resource Name (ARN) of the task definition.
Type: String
Required: No

taskRoleArn
The Amazon Resource Name (ARN) of the IAM role that containers in this task can assume. All containers in this task are granted the permissions that are specified in this role.
Type: String

Required: No

volumes

The list of volumes in a task. For more information about volume definition parameters and defaults, see Amazon ECS Task Definitions in the *Amazon EC2 Container Service Developer Guide*.

Type: array of Volume (p. 146) objects

Required: No

TaskOverride

The overrides associated with a task.

Contents

containerOverrides

One or more container overrides sent to a task.

Type: array of ContainerOverride (p. 123) objects

Required: No

taskRoleArn

The Amazon Resource Name (ARN) of the IAM role that containers in this task can assume. All containers in this task are granted the permissions that are specified in this role.

Type: String

Required: No

Ulimit

The `ulimit` settings to pass to the container.

Contents

hardLimit
The hard limit for the ulimit type.
Type: Integer
Required: Yes

name
The `type` of the `ulimit`.
Type: String
Valid Values: `core | cpu | data | fsize | locks | memlock | msgqueue | nice | nofile | nproc | rss | rtprio | rttime | sigpending | stack`
Required: Yes

softLimit
The soft limit for the ulimit type.
Type: Integer
Required: Yes

VersionInfo

The Docker and Amazon ECS container agent version information about a container instance.

Contents

agentHash
The Git commit hash for the Amazon ECS container agent build on the amazon-ecs-agent GitHub repository.
Type: String
Required: No

agentVersion
The version number of the Amazon ECS container agent.
Type: String
Required: No

dockerVersion
The Docker version running on the container instance.
Type: String
Required: No

Volume

A data volume used in a task definition.

Contents

host

The contents of the `host` parameter determine whether your data volume persists on the host container instance and where it is stored. If the host parameter is empty, then the Docker daemon assigns a host path for your data volume, but the data is not guaranteed to persist after the containers associated with it stop running.

Type: HostVolumeProperties (p. 128) object

Required: No

name

The name of the volume. Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed. This name is referenced in the `sourceVolume` parameter of container definition `mountPoints`.

Type: String

Required: No

VolumeFrom

Details on a data volume from another container.

Contents

readOnly

If this value is `true`, the container has read-only access to the volume. If this value is `false`, then the container can write to the volume. The default value is `false`.

Type: Boolean

Required: No

sourceContainer

The name of the container to mount volumes from.

Type: String

Required: No

Common Parameters

The following table lists the parameters that all actions use for signing Signature Version 4 requests. Any action-specific parameters are listed in the topic for that action. To view sample requests, see Examples of Signed Signature Version 4 Requests or Signature Version 4 Test Suite in the *Amazon Web Services General Reference*.

Action
The action to be performed.

Type: string

Required: Yes

Version
The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm
The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential
The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service. For a list of services that support AWS Security Token Service, go to Using Temporary Security Credentials to Access AWS in *Using Temporary Security Credentials*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Common Errors

This section lists the common errors that all actions return. Any action-specific errors are listed in the topic for the action.

IncompleteSignature
The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure
The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction
The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId
The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination
Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue
An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter
The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString
The query string contains a syntax error.

HTTP Status Code: 404

MissingAction
The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken
The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter
A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired
The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired
The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable
The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

Throttling
The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError
The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400