Building a Home Security System with Elixir and Nerves

Arto Bendiken

Nerves?

Elixir?

The Killer App for Elixir?

Nerves (nerves-project.org) is the true killer app of Elixir. It’s not yet widely known outside of the Elixir community, but that is changing quickly (even in this very room, right now).

Nerves is doing to embedded development and Elixir what Rails did for web development and Ruby.

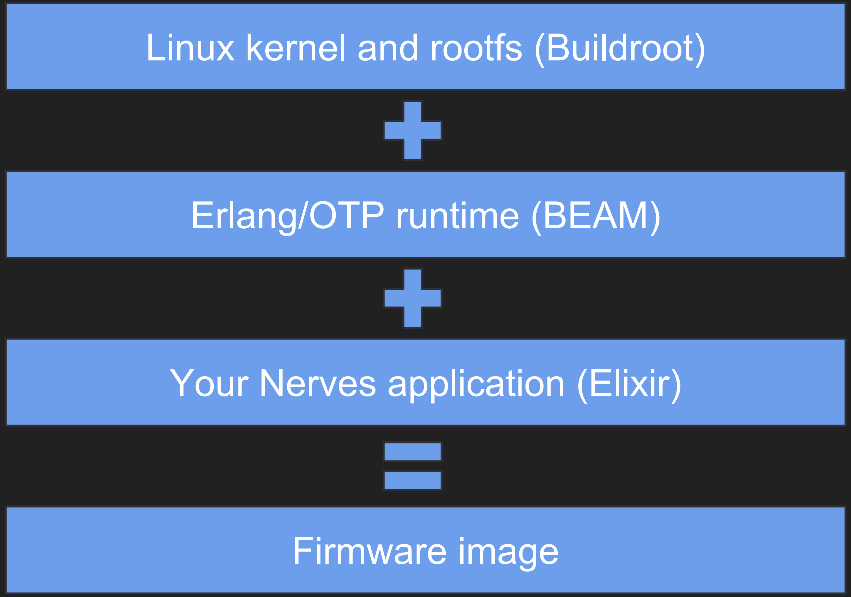
The current statistics from the Elixir community on Slack as of February 2019:
`#phoenix` 21,100+ members, `#nerves` 1,800+ members

What is Nerves?

Nerves (nerves-project.org) enables you to "*craft and deploy bulletproof embedded software in Elixir*".

Nerves is an umbrella project consisting of **tooling and libraries** enabling the development of **robust, reliable firmware** for **smart hardware** devices written in a **high-level** functional language (Elixir) and running on a **carrier-grade** runtime (Erlang/OTP).

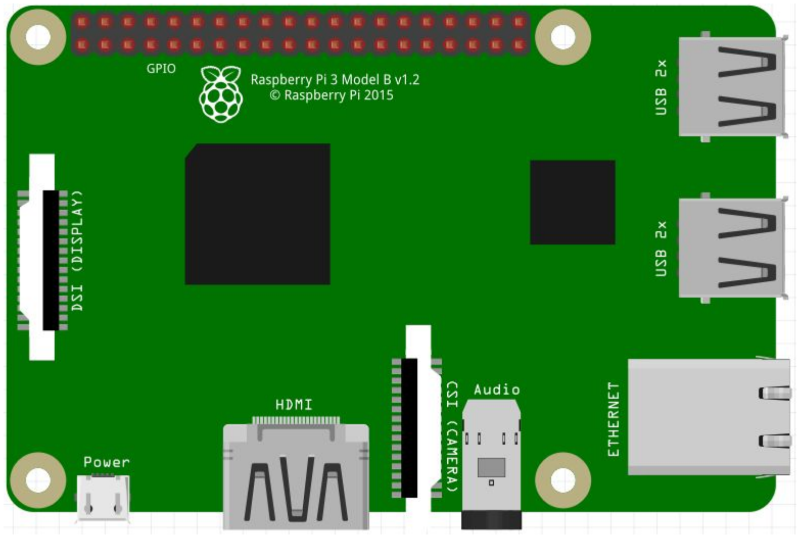
What is Nerves?


Linux kernel and rootfs (Buildroot)
Erlang/OTP runtime (BEAM)
Your Nerves application (Elixir)
Firmware image

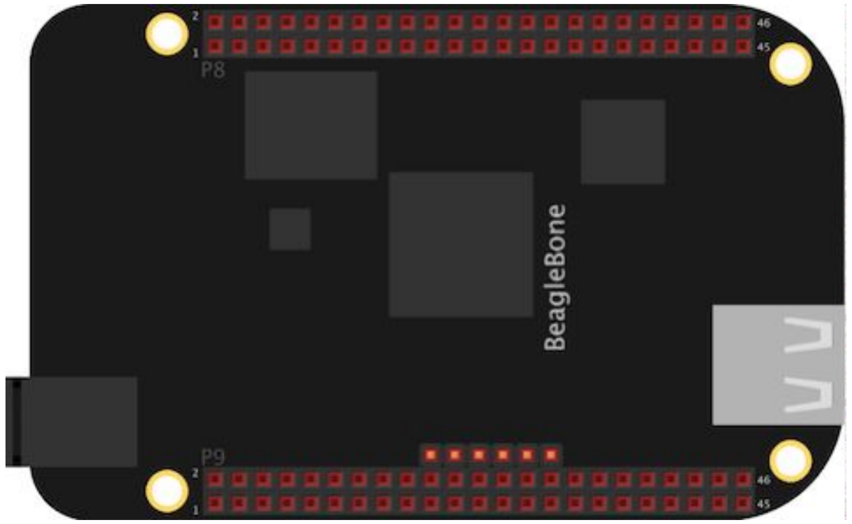
- Produces firmware images as small as 12 MB, which includes the Linux kernel, the Erlang/OTP runtime, and your Nerves application
- Firmware images are burned to an SD card for deployment to target hardware
- The firmware boots in a few seconds: your application code can be running in as little 3-4 seconds after power on
- An incredibly stable runtime platform with low and predictable steady-state RAM consumption: on the order of 8 MB, and your application probably will fit in 32 MB

Hobbyist Hardware for Nerves

Raspberry Pi 3 Model A/B/B+ (RPi3)



BeagleBone Black/Green/Blue (BBB/BBG/BBGW)



Why Elixir for Embedded Development?

- This is what the Erlang platform (OTP, BEAM) was designed for (telecomms switches)—good soft-realtime support
- Pattern matching (with bitstring support) is great for implementing wire protocols robustly
- The real world and real hardware is inherently concurrent, asynchronous, and unpredictable—message passing with the actor model is a good way to model that
- Fault tolerance and error recovery actually matter (just let it crash—so long as the user doesn't notice)

Hobbyist Hardware for Nerves

Raspberry Pi 3 Model B+ (RPi3)

- **$35** MSRP
- 1.4 GHz Cortex-A53 (4× cores)
- 1 GB LPDDR2 RAM
- 4× USB ports, 1× Ethernet port, 1× HDMI port, etc
- 40× GPIO pins
- 802.11ac Wi-Fi (dual band)
- Bluetooth 4.1 and LE

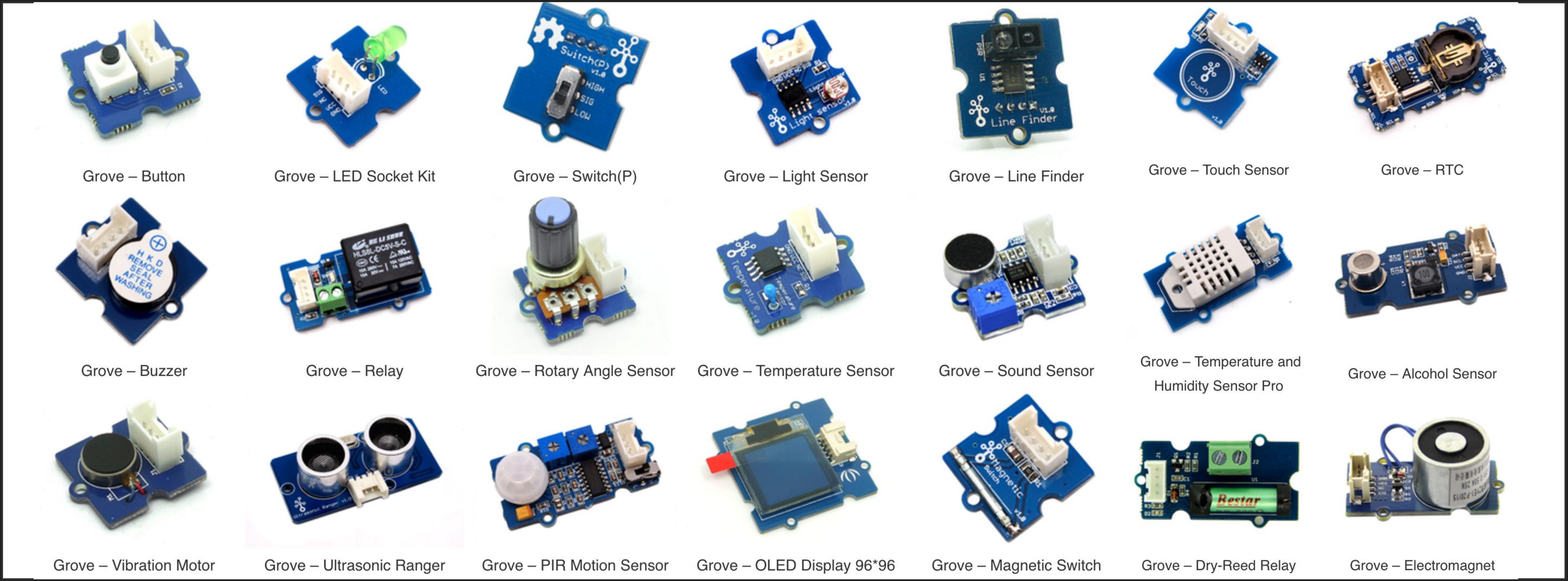
BeagleBone Green (BBG)

- **$44** MSRP
- 1.0 GHz Cortex-A8
- 512 MB DDR3 RAM
- 1× USB port, 1× Ethernet port, micro-HDMI as add-on, etc
- 92× GPIO pins and 2× Grove connectors
- **Open-source hardware**, customizable for production designs

Hobbyist Hardware for Nerves

Seeed Studio's Grove System

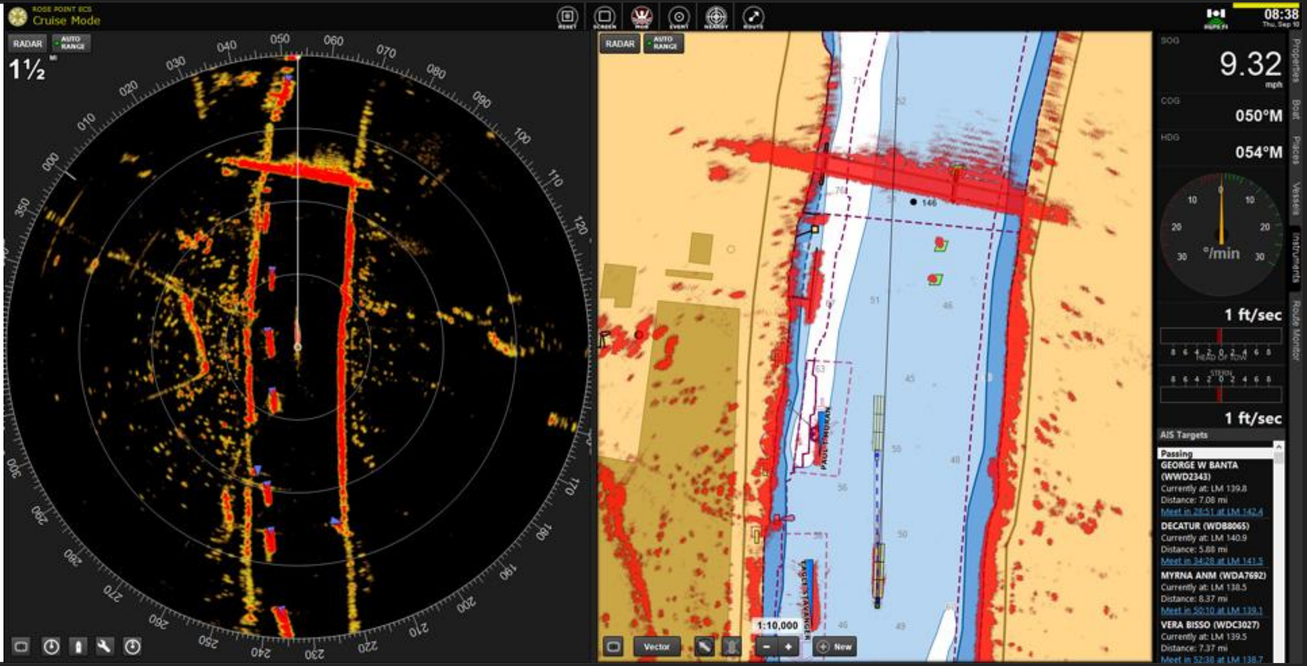
100s of sensors and actuators



Commercial Products Based on Nerves

Commercial Radar Interface

http://www.rosepoint.com/commercial-radar-interface/



AIS Targets

Passing

GEORGE W BANTA (WWD2343)
Currently at: LM 139.8
Distance: 7.08 mi
Meet in 28:51 at LM 142.4

DECATUR (WDB8065)
Currently at: LM 140.9
Distance: 5.88 mi
Meet in 34:28 at LM 141.5

MYRNA ANM (WDA7692)
Currently at: LM 138.5
Distance: 8.37 mi
Meet in 50:10 at LM 139.1

VERA BISSO (WDC3027)
Currently at: LM 139.5
Distance: 7.37 mi
Meet in 52:38 at LM 138.7

Future Products Based on Nerves

Anything and everything…

Your imagination is the limit.

Hello, Blinky

```
$ git clone https://github.com/nerves-project/nerves_examples
```

TERM	DEFINITION
host	The computer on which you are editing source code, compiling, and assembling firmware
target	The platform for which your firmware is built (for example, Raspberry Pi, Raspberry Pi 2, or Beaglebone Black)
toolchain	The tools required to build code for the target, such as compilers, linkers, binutils, and C runtime
system	A lean Buildroot-based Linux distribution that has been customized and cross-compiled for a particular target
assemble	The process of combining system, application, and configuration into a firmware bundle
firmware bundle	A single file that contains an assembled version of everything needed to burn firmware
firmware image	Built from a firmware bundle and contains the partition table, partitions, bootloader, etc.

Project Structure

```
$ tree blinky/
```

Firmware Update

The Slow & Hard Way:

```
$ mix firmware.burn
```

Firmware Update

The Quick & Easy Way:

```
$ mix firmware.gen.script && ./upload.sh
```

Home Alarm System

Work in Progress

Demo

Remote Console

```
$ iex --name me@0.0.0.0 --cookie pivorak --remsh home@nerves.local
```

```
$ ssh nerves.local
```

Remote Debugging

```
iex> Logger.configure(level: :debug)
```

Local Debugging

```
iex> :observer.start()
```

Technologies

- Elixir
- Erlang/OTP
- C++
- OpenCV
- FaceNet
- Dlib
- Nerves
- Linux kernel
- gRPC
- Protocol Buffers
- Python
- Go?
- Ruby?
- etc

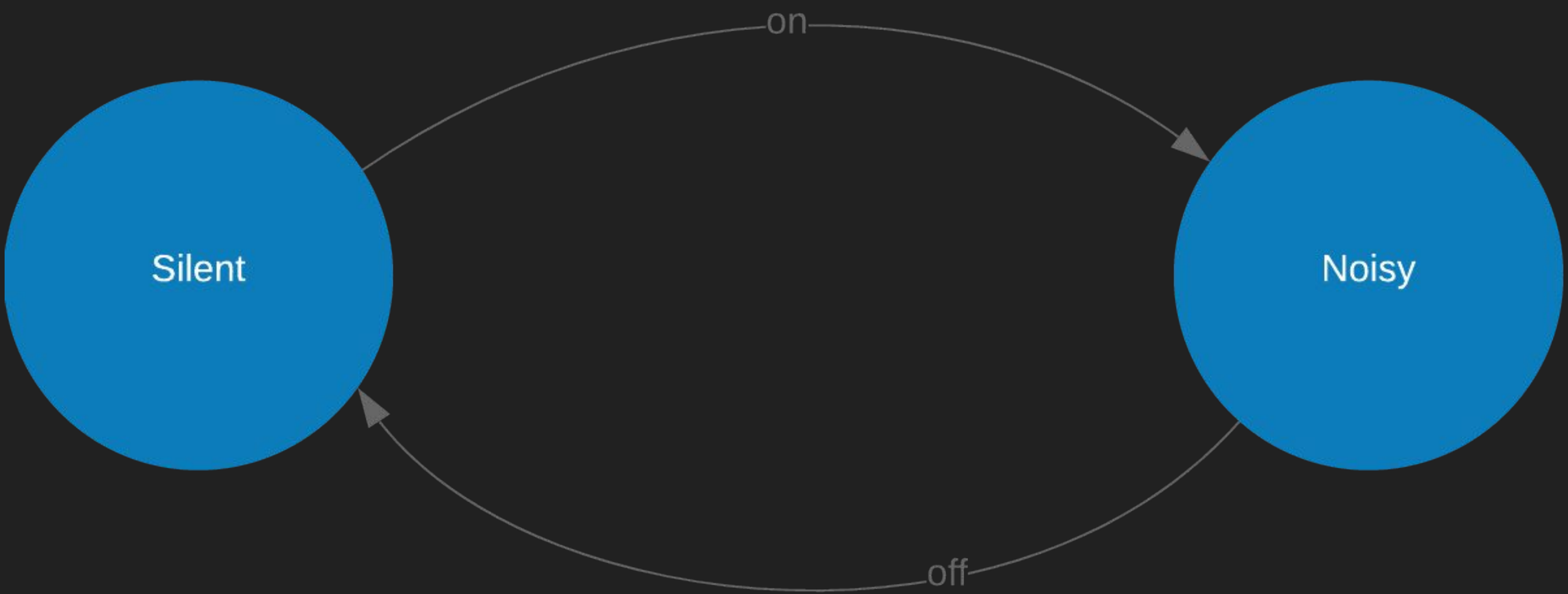
Camera Driver

C++, V4L2, OpenCV → Elixir

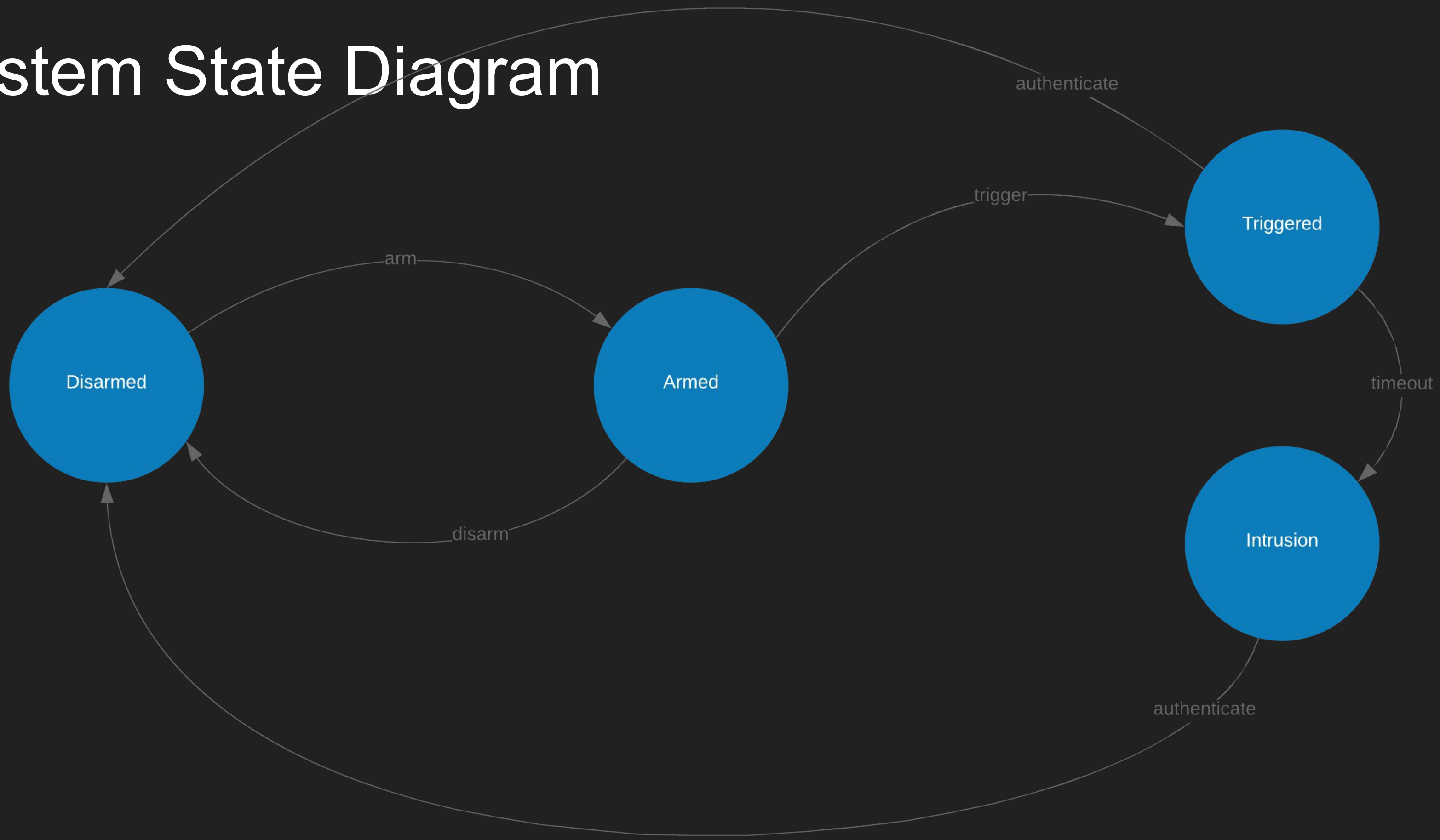
Alarm System

Elixir, GenStateMachine (gen_statem)

Siren State Diagram



System State Diagram



Home Protocol & API

Elixir, gRPC, PB ↔ Python, Go, Ruby, etc

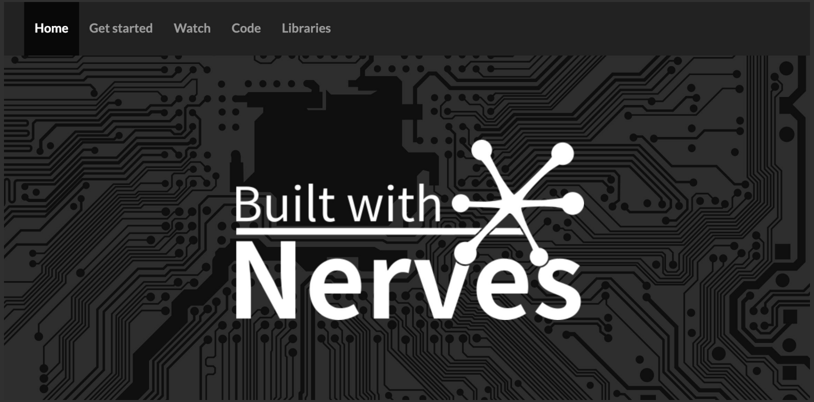
Camera Client

Python, OpenCV, gRPC ↔ gRPC, Elixir

Face Recognition…

OpenCV, Dlib, FaceNet

See



https://nerves-project.org

See Also

gokrazy is a pure-Go userland for your Raspberry Pi 3 appliances

For a long time, we were unhappy with having to care about security issues and Linux distribution maintenance on our various Raspberry Pis.

Then, we had a crazy idea: what if we got rid of memory-unsafe languages and all software we don't strictly need?

Turns out this is feasible. gokrazy is the result.

https://gokrazy.org

Дякую!

Find me at:

https://ar.to & @bendiken