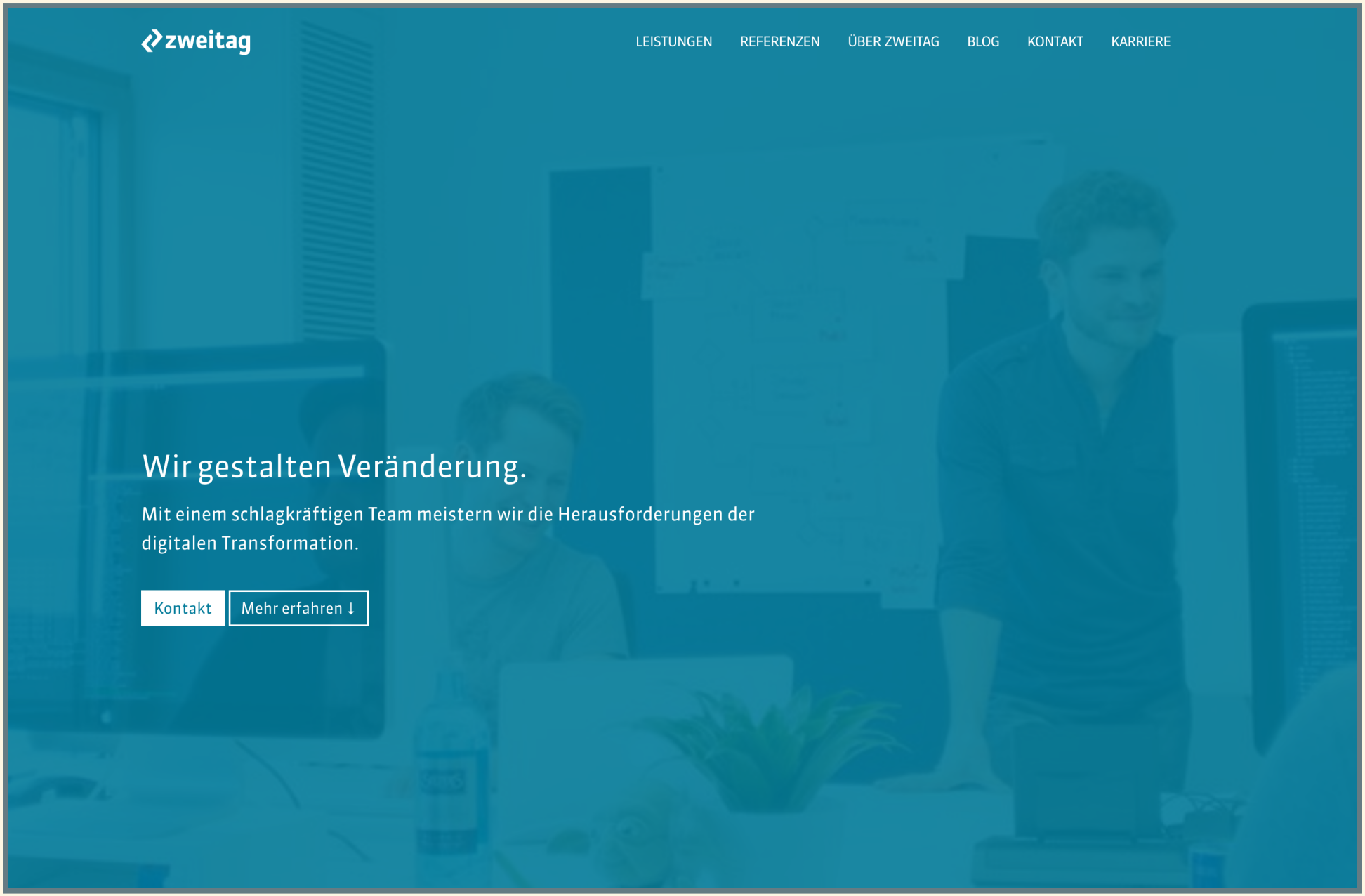
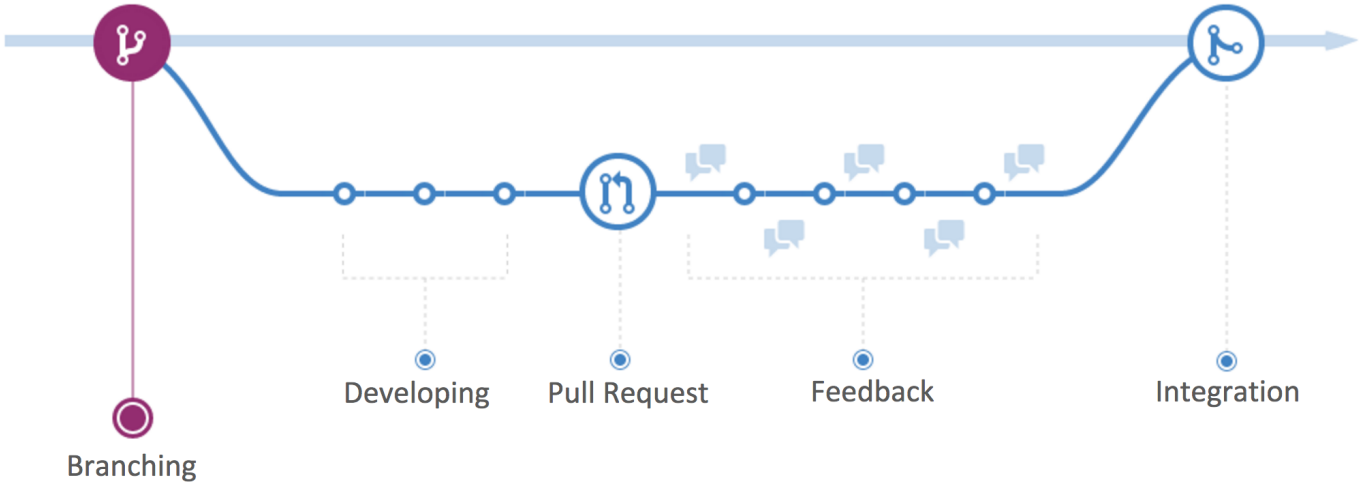
ELIXIR + PHOENIX

WIR

- Patrick
- Heiko


zweitag
LEISTUNGEN
REFERENZEN
ÜBER ZWEITAG
BLOG
KONTAKT
KARRIERE
Wir gestalten Veränderung.
Mit einem schlagkräftigen Team meistern wir die Herausforderungen der digitalen Transformation.
Kontakt
Mehr erfahren ↓

Qualitätsorientierter Entwicklungsprozess



github SOCIAL CODING

Travis

CODE CLIMATE

Feature-orientierter Prozess unterstützt von den richtigen Tools

- Agile Entwicklung
- Test-driven Development (TDD)
- Qualitätssicherung
- Continuous Integration & Delivery

UND IHR?

- Vorkenntnisse in Programmierung?
- Vorkenntnisse in Web-Anwendungen?
- Vorkenntnisse in Elixir und Phoenix?
- Erwartungen an den Workshop?

ORGANISATORISCHES

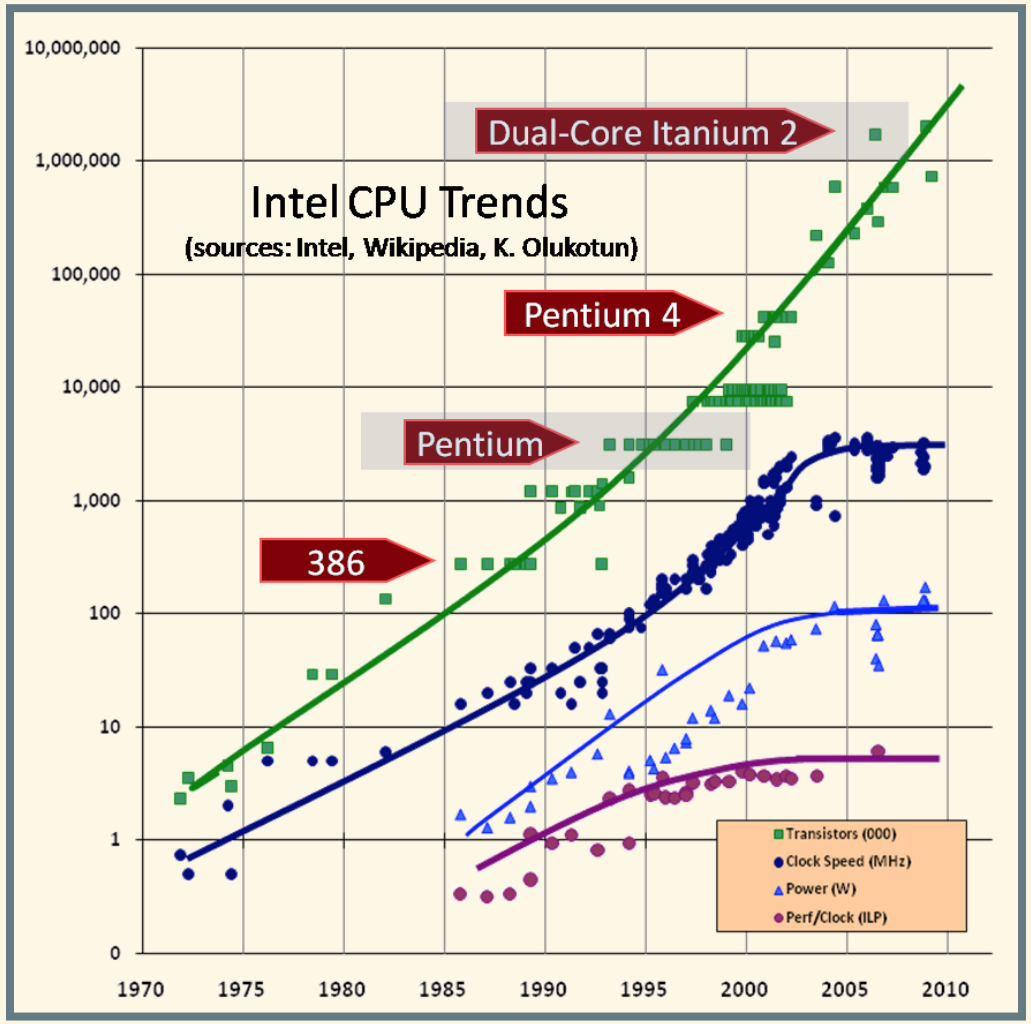
- Pizza-Bestellung: Besondere Wünsche?
- Fotos
- Feedback-Bogen

AGENDA

- Einführung in Elixir
- Hands-On: Elixir
- Mittagessen
- Einführung in Phoenix
- Hands-On: Eine Chat-Web-Anwendung mit Phoenix
- Retrospektive

WARUM ELIXIR?

THE FREE LUNCH IS OVER



http://www.gotw.ca/publications/concurrency-ddj.htm

There are three hard things in computer science: cache invaconcurrlidation, namiencyng things, and off-by-one-errors.

CADE METZ BUSINESS 09.15.15 7:00 AM

WHY WHATSAPP ONLY NEEDS 50 ENGINEERS FOR ITS 900M USERS

SHARE

SHARE 152

TWEET

PIN

COMMENT 46

EMAIL

EARLIER THIS MONTH, in a post to his Facebook page, WhatsApp CEO Jan Koum announced that his company's instant messaging service is now used by more than 900 million people. And then Facebook CEO Mark Zuckerberg promptly responded with two posts of his own. One said "congrats," and the other included a cheeky photo Zuckerberg had taken of Koum as the WhatsApp CEO keyed his 900-million-user post into a smartphone. "Here's an action shot of you writing this update," Zuckerberg wrote.

WhatsApp is owned by Facebook, after Zuckerberg and company paid $19 billion for the startup a little more than a year ago. That means Facebook now runs three of the most popular apps on the internet. Its primary social networking service is used by more than 1.5 billion people worldwide, and Facebook Messenger, an instant messaging service spun off from Facebook proper, spans 700 million. But the 900 million-user milestone announced by Koun is very much a WhatsApp achievement, not a product of the formidable Facebook machine.

MOST POPULAR

TV

John Oliver Sums Up Election 2016 in One Devastating *Last Week*...

17 HOURS

MUSIC

These Are the 6 Albums You Must Listen to Now

1 DAY

https://www.wired.com/2015/09/whatsapp-serves-900-million-users-50-engineers/

DAS VERSPRECHEN

- Erlang VM
- Scalable
- Fault Tolerant
- Tooling

GRUNDLAGEN

BESTANDTEILE

- Module
- Funktionen
- Datenstrukturen
- KEINE Klassen, keine Interfaces

HELLO WORLD

```
defmodule Hello do
  def world do
    IO.puts "hello world"
  end
end
```

```
iex(1)> Hello.world
hello world
:ok
```

IMMUTABILITY

Warum?

```
number = 1
do_something_with(number)
print(number)
```

```
array = [ 1, 2, 3 ]
do_something_with(array)
print(array)
```

IMMUTABILITY

```
iex> name = "elixir"
"elixir"
iex> cap_name = String.capitalize name
"Elixir"
iex> name
"elixir"
```

PATTERN MATCHING (1)

```
iex(1)> [a, b, c] = [1, 2, 3]
[1, 2, 3]
iex(2)> a
1
iex(3)> b
2
iex(4)> c
3
```

```
iex(1)> %{name: name} = %{name: "Heiko", company: "Zweitag"}
%{company: "Zweitag", name: "Heiko"}
iex(2)> name
"Heiko"
```

```
iex(1)> %{name: name} = %{company: "Zweitag"}
** (MatchError) no match of right hand side value: %{company: "Zweitag"}
```

PATTERN MATCHING (2)

```
defmodule SumUntil do
  def sum(0), do: 0
  def sum(number), do: sum(number - 1) + number
end
```

DER PIPE-OPERATOR |>

```
people = DB.find_customers
orders = Orders.for_customers(people)
tax    = sales_tax(orders, 2016)
filing = prepare_filing(tax)
```

```
filing = prepare_filing(sales_tax(Orders.for_customers(DB.find_customers
```

```
filing = DB.find_customers
  |> Orders.for_customers
  |> sales_tax(2016)
  |> prepare_filing
```

ÜBUNG: FIZZBUZZ

1. Write a function that takes three arguments. If the first two are zero, return “FizzBuzz.” If the first is zero, return “Fizz.” If the second is zero, return “Buzz.” Otherwise return the third argument.
2. The operator rem(a, b) returns the remainder after dividing a by b. Write a function that takes a single integer (n) and calls the function in the previous exercise, passing it rem(n,3), rem(n,5), and n. Call it seven times with the arguments 10, 11, 12, and so on. You should get “Buzz, 11, Fizz, 13, 14, FizzBuzz, 16.”

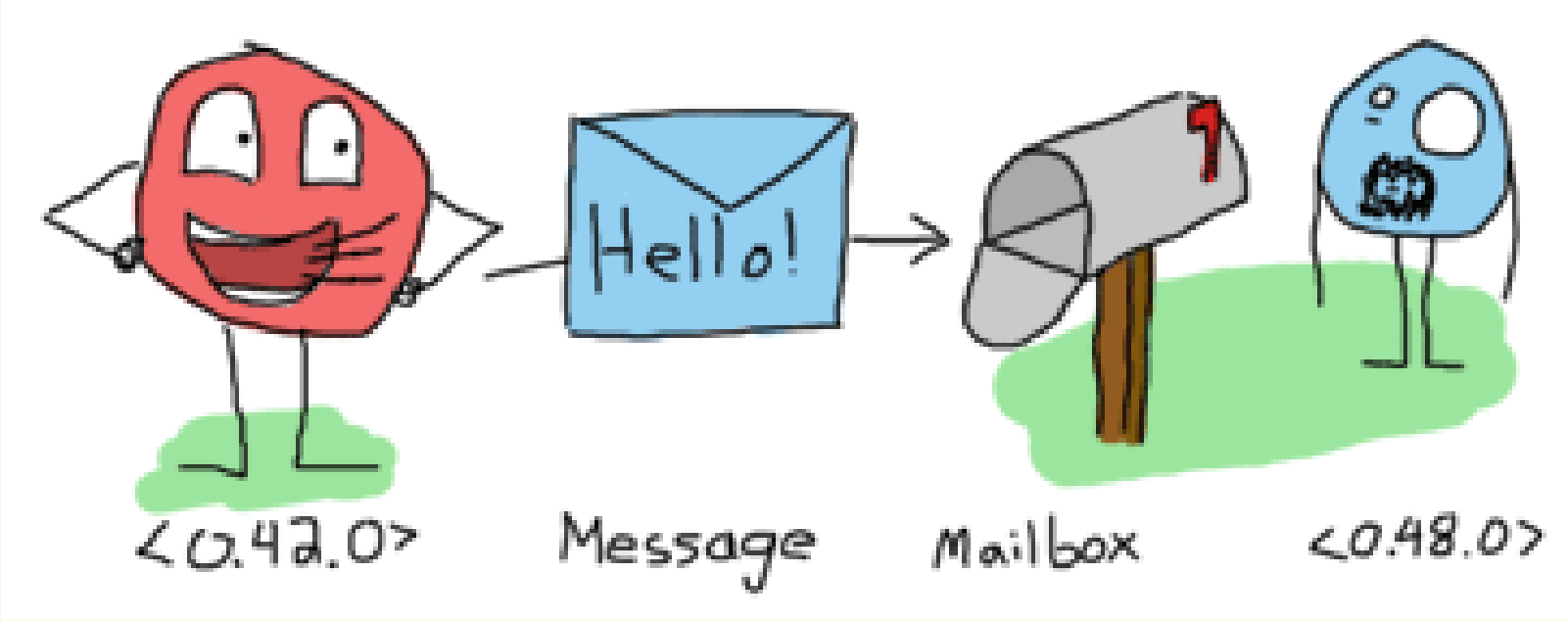
```
ExUnit.start

defmodule FizzBuzzTest do
  use ExUnit.Case, async: true

  test "fizzbuzz" do
    assert FizzBuzz.fizzbuzz(1) == 1
    assert FizzBuzz.fizzbuzz(2) == 2
    assert FizzBuzz.fizzbuzz(3) == "Fizz"
    assert FizzBuzz.fizzbuzz(5) == "Buzz"
    assert FizzBuzz.fizzbuzz(15) == "FizzBuzz"
  end
end
```

CONCURRENCY UND OTP

DAS ACTOR-MODEL



PROZESS-KOMMUNIKATION

```
defmodule Hello do
  def greet do
    receive do
      {sender, msg} ->
        send sender, { :ok, "Hello, #{msg}" }
    end
  end
end

# here's a client
pid = spawn(Hello, :greet, [])
send pid, {self, "World!"}
receive do
  {:ok, message} -> IO.puts message
end
```

PROZESSE ALS SERVER (1)

```
defmodule Counter1 do
  def loop(current_number) do
    receive do
      {from, :next} ->
        send(from, {:ok, current_number})
        loop(current_number + 1)
      end
  end
end

# here's a client
pid = spawn(Counter1, :loop, [1])
send pid, {self, :next}
receive do
  {:ok, number} -> IO.puts number
end
```

PROZESSE ALS SERVER (2)

```
defmodule Counter2 do
  use GenServer

  def handle_call(:next, _from, current_number) do
    {:reply, current_number, current_number + 1}
  end
end

# here's a client
{:ok, pid} = GenServer.start_link(Counter2, 1)
GenServer.call(pid, :next)

# Inspect the state
:observer.start
```

PROZESS-ROLLEN

- Server
- Supervisor
- Application (Component)

TOOLING

IEX

```
iex> 1 + 2
3
```

MIX

```
mix new my_project
* creating README.md
* creating .gitignore
* creating mix.exs
* creating config
* creating config/config.exs
* creating lib
* creating lib/my_project.ex
* creating test
* creating test/test_helper.exs
* creating test/my_project_test.exs

Your Mix project was created successfully.
You can use "mix" to compile it, test it, and more:

  cd my_project
  mix test
```

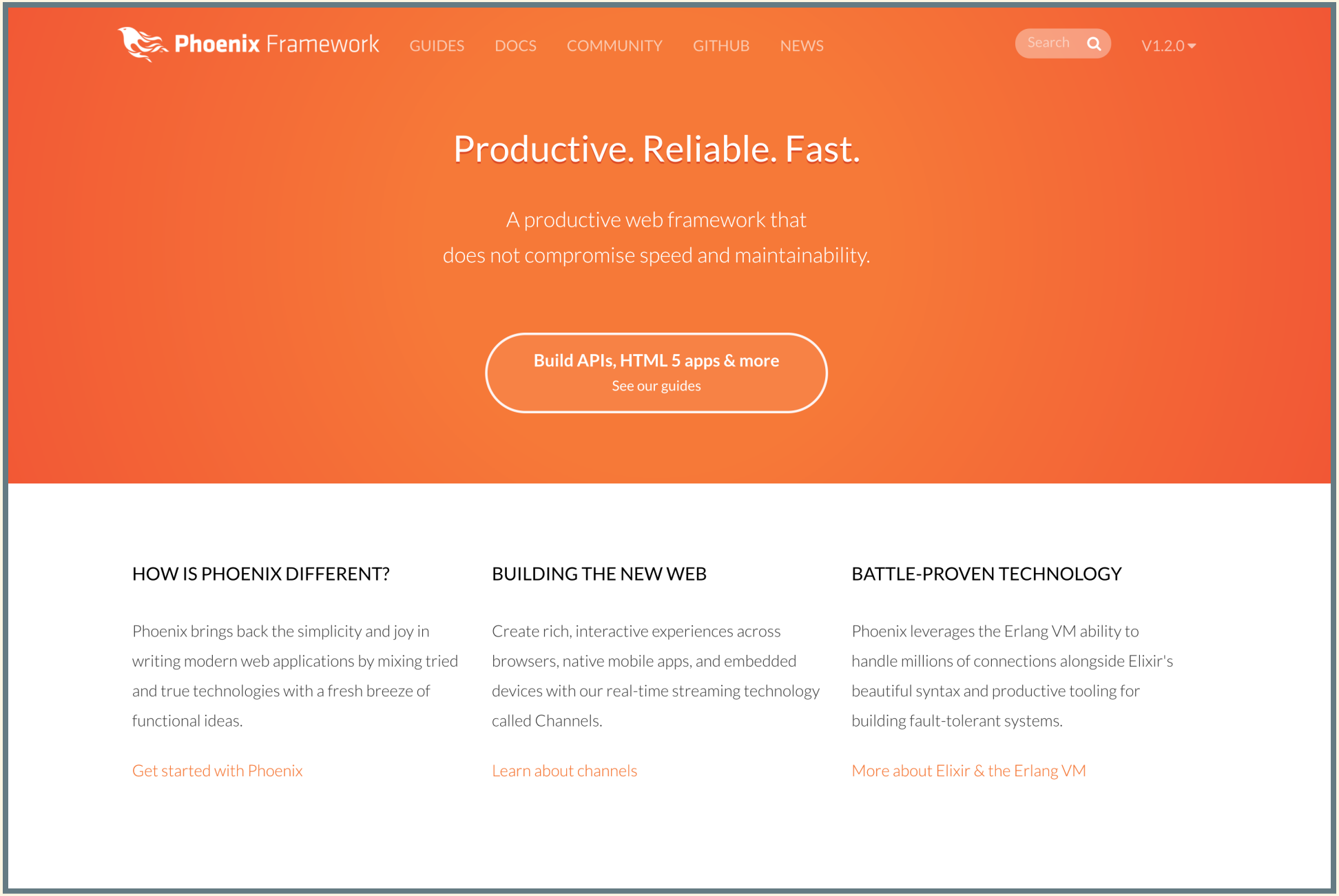
HEX

```
defp deps do
  [{:phoenix, "~> 1.2"},
  {:ecto, "~> 2.0.5"}]
end
```

```
mix deps.get

Running dependency resolution
Dependency resolution completed
  decimal: 1.3.1
  ecto: 2.0.5
  mime: 1.0.1
  phoenix: 1.2.1
  phoenix_pubsub: 1.0.1
  plug: 1.2.2
  poison: 2.2.0
  poolboy: 1.5.1
* Getting phoenix_pubsub (Hex package)
  Checking package (https://repo.hex.pm/tarballs/phoenix_pubsub-1.0.1.ta
  Using locally cached package
* Getting plug (Hex package)
  Checking package (https://repo.hex.pm/tarballs/plug-1.2.2.tar)
  Using locally cached package
```

CORE PROJECTS


Phoenix Framework
GUIDES
DOCS
COMMUNITY
GITHUB
NEWS
Search
V1.2.0
Productive. Reliable. Fast.
A productive web framework that
does not compromise speed and maintainability.
Build APIs, HTML 5 apps & more
See our guides
HOW IS PHOENIX DIFFERENT?
Phoenix brings back the simplicity and joy in writing modern web applications by mixing tried and true technologies with a fresh breeze of functional ideas.
Get started with Phoenix
BUILDING THE NEW WEB
Create rich, interactive experiences across browsers, native mobile apps, and embedded devices with our real-time streaming technology called Channels.
Learn about channels
BATTLE-PROVEN TECHNOLOGY
Phoenix leverages the Erlang VM ability to handle millions of connections alongside Elixir's beautiful syntax and productive tooling for building fault-tolerant systems.
More about Elixir & the Erlang VM

Ecto

Ecto is split into 4 main components:

- `Ecto.Repo` - repositories are wrappers around the data store. Via the repository, we can create, update, destroy and query existing entries. A repository needs an adapter and credentials to communicate to the database
- `Ecto.Schema` - schemas are used to map any data source into an Elixir struct. We will often use them to map tables into Elixir data but that's one of their use cases and not a requirement for using Ecto
- `Ecto.Changeset` - changesets provide a way for developers to filter and cast external parameters, as well as a mechanism to track and validate changes before they are applied to your data
- `Ecto.Query` - written in Elixir syntax, queries are used to retrieve information from a given repository. Queries in Ecto are secure, avoiding common problems like SQL Injection, while still being composable, allowing developers to build queries piece by piece instead of all at once

In the following sections, we will provide an overview of those components and how they interact with each other. Feel free to access their respective module documentation for more specific examples, options and configuration.

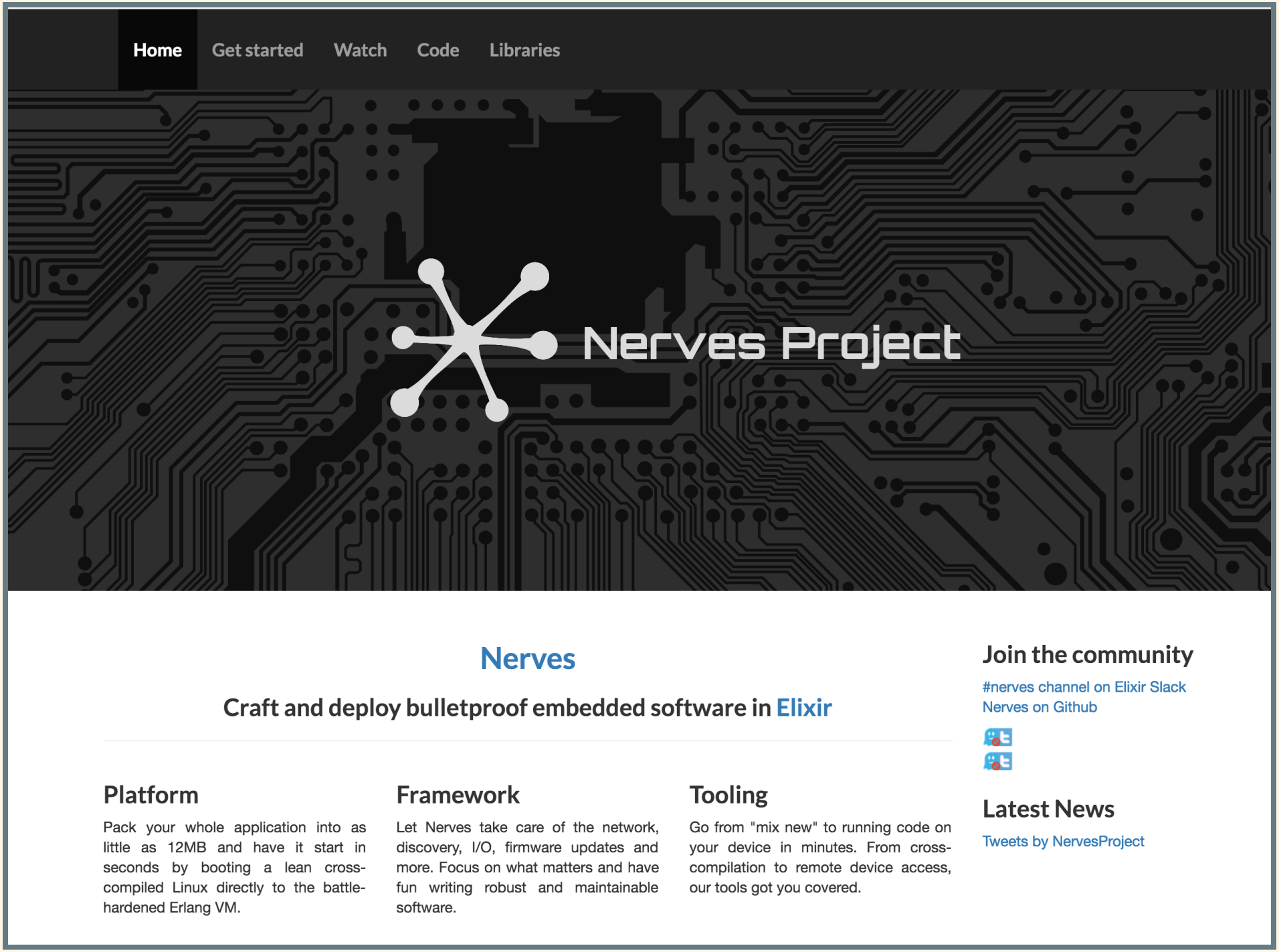
If you want to quickly check a sample application using Ecto, please check the getting started guide and the accompanying sample application.

Repositories

`Ecto.Repo` is a wrapper around the database. We can define a repository as follows:

```
defmodule Repo do
  use Ecto.Repo, otp_app: :my_app
end
```

Where the configuration for the Repo must be in your application environment, usually defined in your `config/config.exs`:


Home
Get started
Watch
Code
Libraries
Nerves Project
Nerves
Craft and deploy bulletproof embedded software in Elixir
Platform
Pack your whole application into as little as 12MB and have it start in seconds by booting a lean cross-compiled Linux directly to the battle-hardened Erlang VM.
Framework
Let Nerves take care of the network, discovery, I/O, firmware updates and more. Focus on what matters and have fun writing robust and maintainable software.
Tooling
Go from "mix new" to running code on your device in minutes. From cross-compilation to remote device access, our tools got you covered.
Join the community
#nerves channel on Elixir Slack
Nerves on Github
Latest News
Tweets by NervesProject