Productive |> Reliable |> Fast

Phoenix framework

Milad Rastian

@slashmili

Why another language/framework?

Why another language/framework?

- Ruby is Slow

Why another language/framework?

- ~~Ruby is Slow~~

Why another language/framework?

- ~~Ruby is Slow~~
- Rails Performance

Why another language/framework?

- ~~Ruby is Slow~~
- ~~Rails Performance~~

Why another language/framework?

- ~~Ruby is Slow~~
- ~~Rails Performance~~
- People are still complaining!

O.K. What are the alternatives?

O.K. What are the alternatives?

- Go

Go?

```
package main

import "github.com/gin-gonic/gin"

func main() {
    r := gin.Default()
    r.GET("/ping", func(c *gin.Context) {
        c.JSON(200, gin.H{
            "message": "hello world",
        })
    })
    r.Run() // listen and server on 0.0.0.0:8080
}
```

O.K. What are the alternatives?

- ~~Go~~

O.K. What are the alternatives?

- ~~Go~~
- Scala / Clojure

Scala/Clojure?

O.K. What are the alternative?

- ~~Go~~
- ~~Scala/Clojure~~

Why do we love Ruby?

- Community
- The joy of programming
- Productivity

You might still say it’s not performing as you want!

Seriously!?

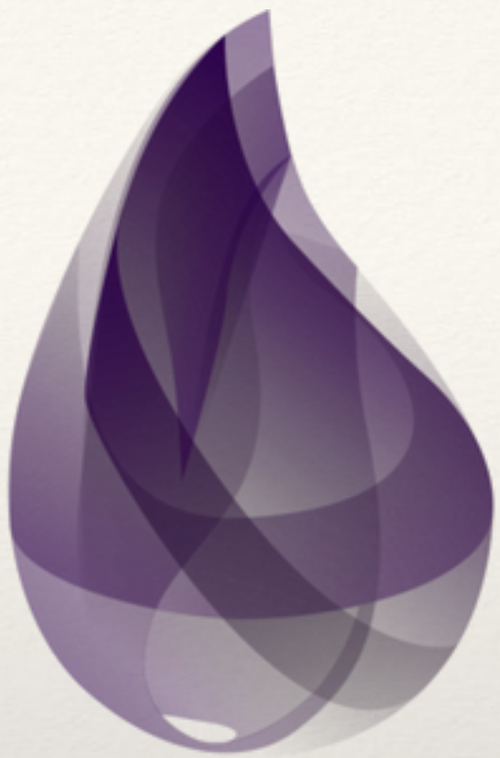
What if?

What if?

- You can have all the benefits of Ruby

What if?

- You can have all the benefits of Ruby
- Without sacrificing performance.



Meet Elixir

Why Elixir/Phoenix?

- Fast

Erlang VM

Distributed

Highly Available

Fault-tolerant

Benchmark

Framework	Throughput (req/s)	Latency (ms)
Phoenix 0.13.1	179,685	0.61ms
Gin (GO)	176,156	0.65ms
Play (Java/Scala)	171,236	1.89ms
Rails	11,903	8.50ms

https://gist.github.com/omnibs/e5e72b31e6bd25caf39a

Why Elixir/Phoenix?

- Fast
- Reliable

Reliable

- Process Monitoring/Supervision
- Recover Failure

Why Elixir/Phoenix?

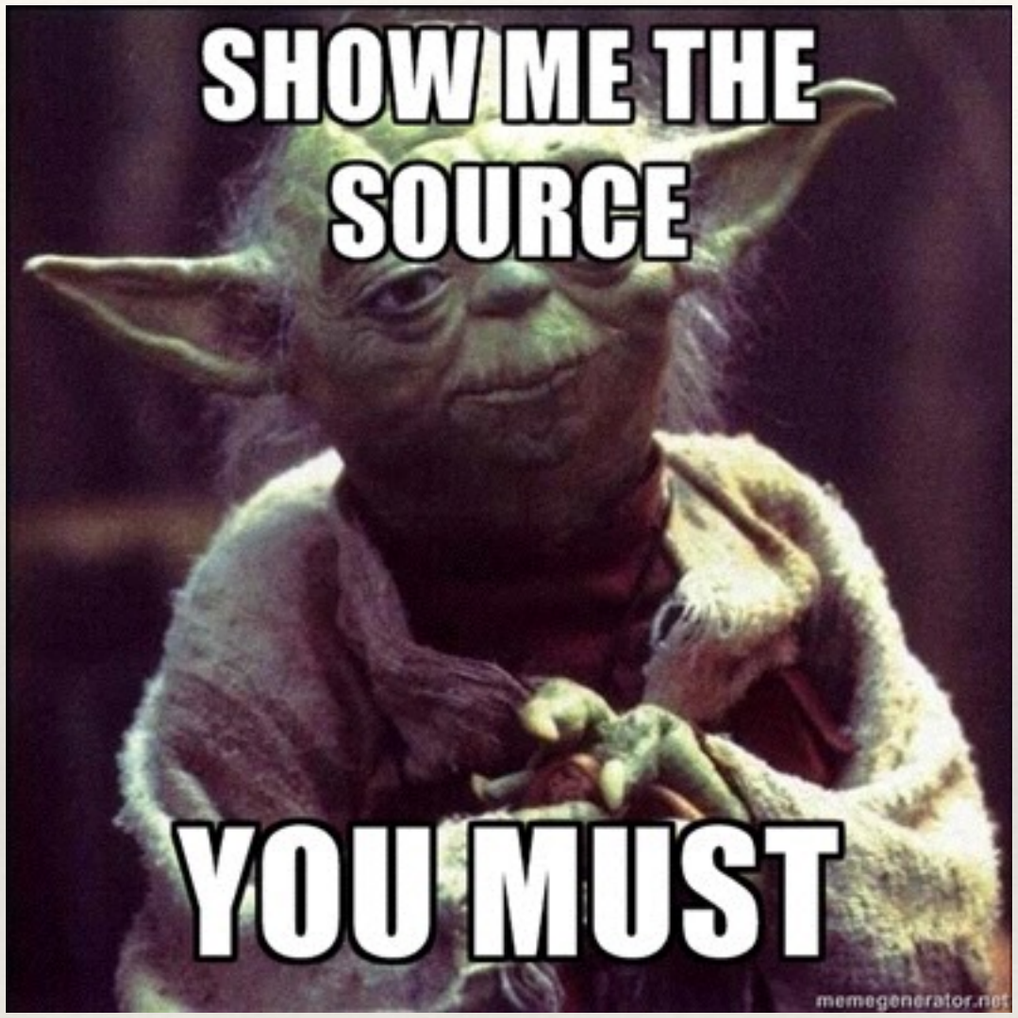
- Fast
- Reliable
- Productive

Productive

```
defmodule Etel.Router do
  use Etel.Web, :router

  pipeline :browser do
    plug :accepts, ["html"]
    plug :fetch_session
    plug :fetch_flash
    plug :protect_from_forgery
    plug :put_secure_browser_headers
  end

  pipeline :api do
    plug :accepts, ["json"]
  end

  scope "/", Etel do
    pipe_through :browser # Use the default browser stack

    get "/", PageController, :index
  end

  # Other scopes may use custom stacks.
  scope "/api", Etel do
    pipe_through :api
  end
end
```

There is no need to trade beautiful, maintainable code

to gain performance

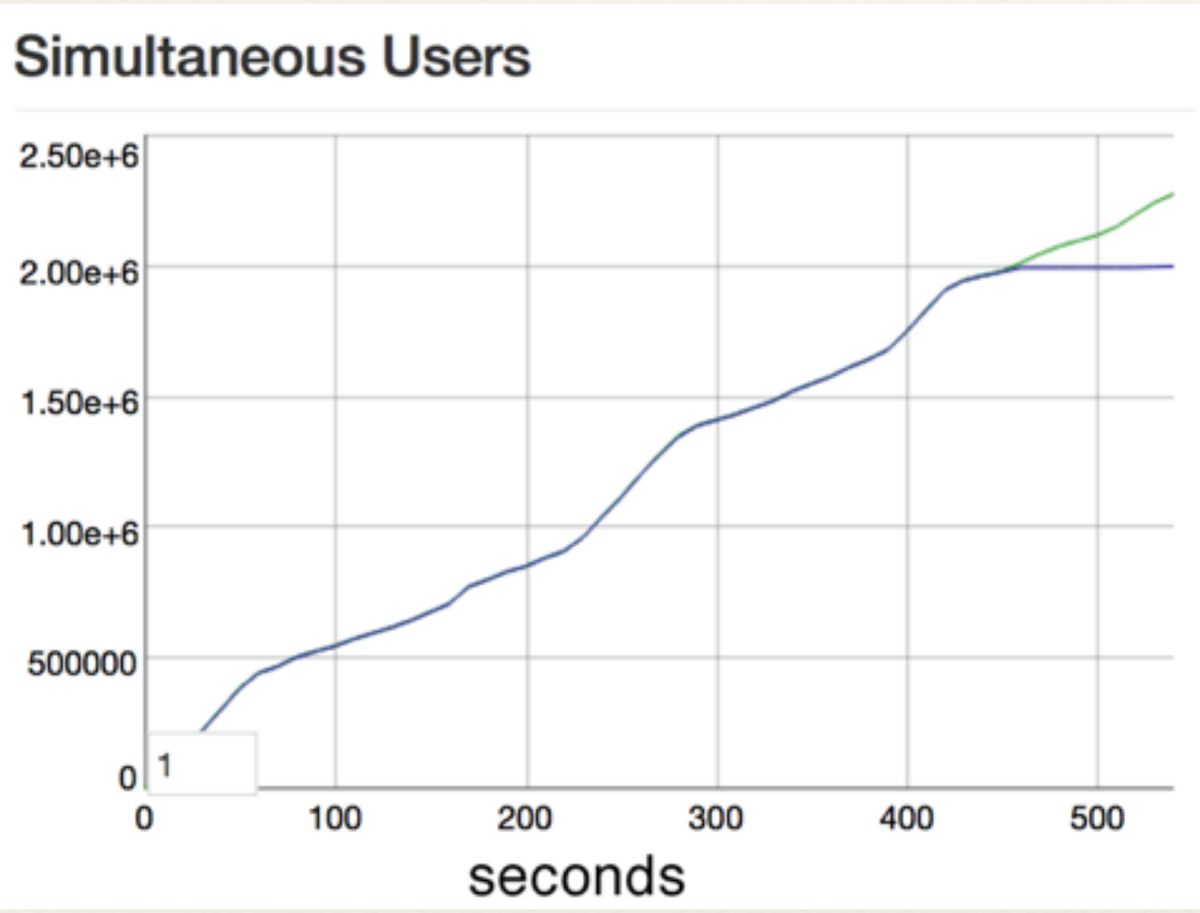


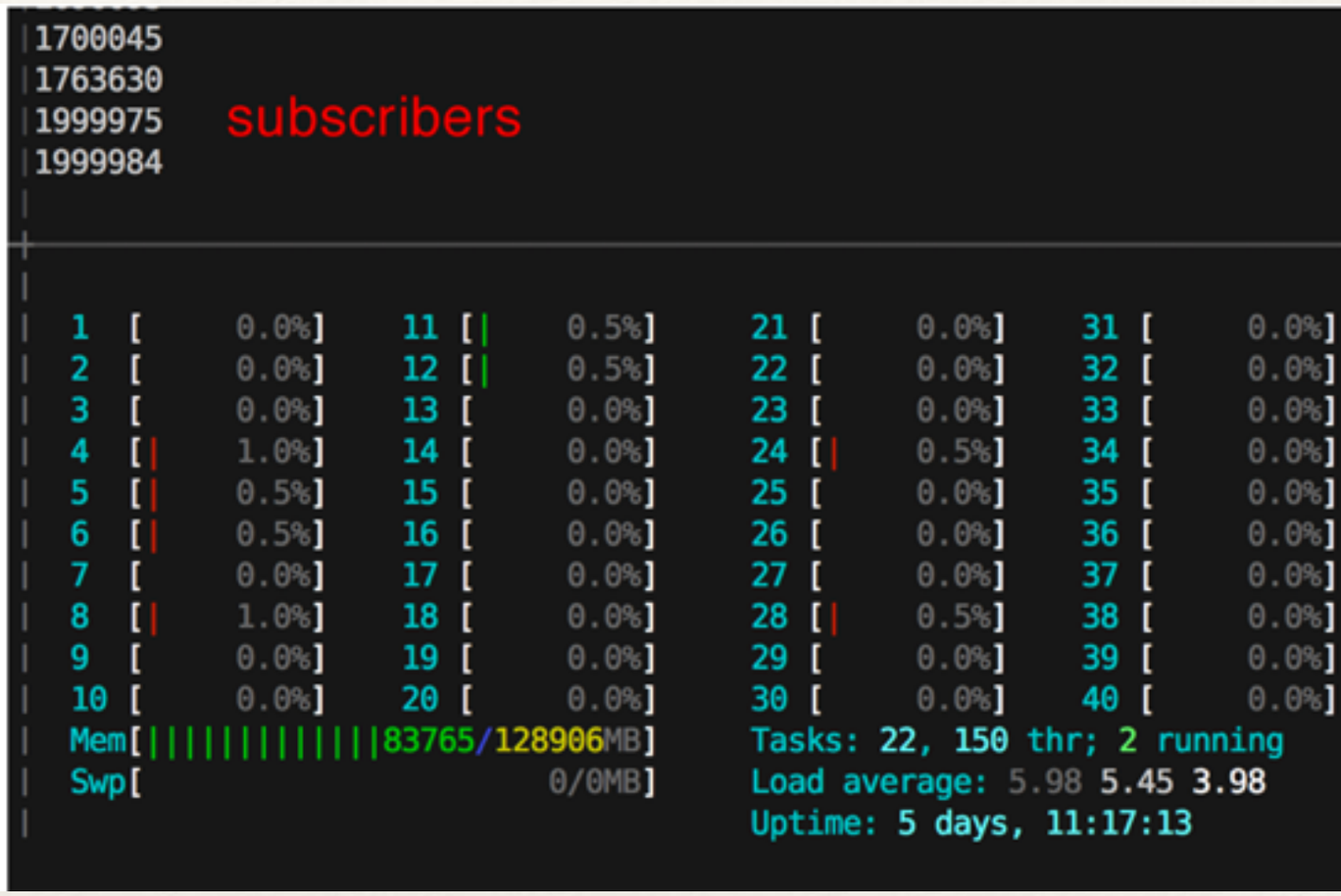
https://github.com/slashmili/talks/tree/master/2016/phoenix-klxrb-march

Static Assets with Brunch

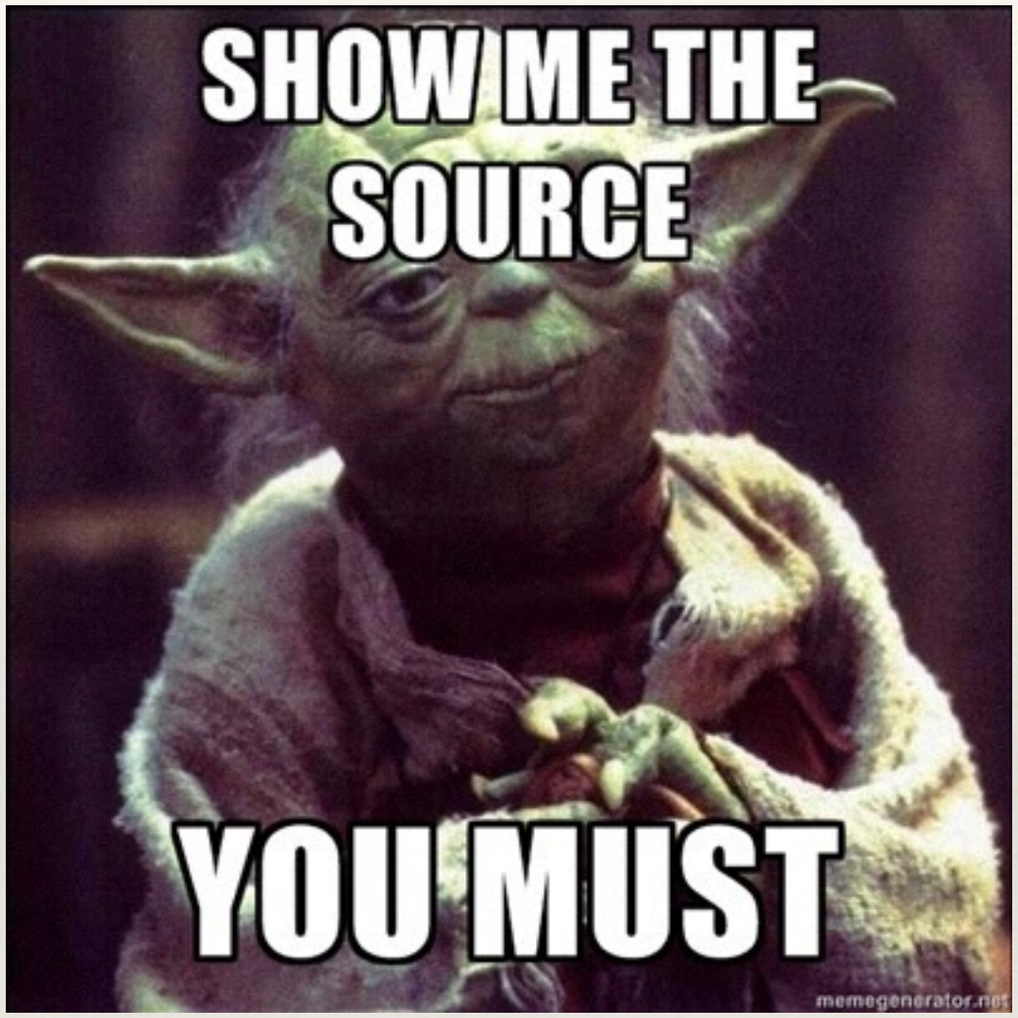
Channels

2 Million Websocket Connections in Phoenix



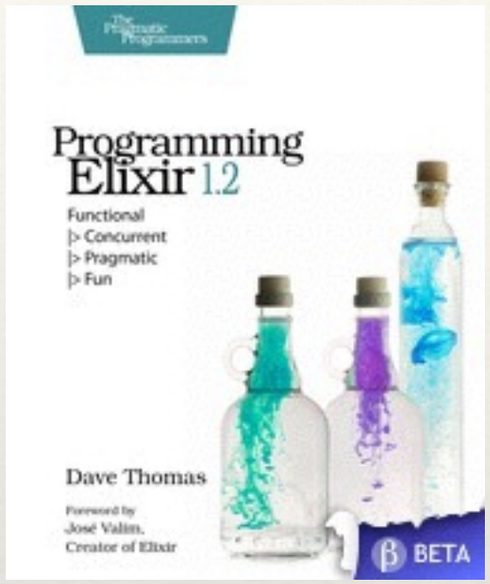


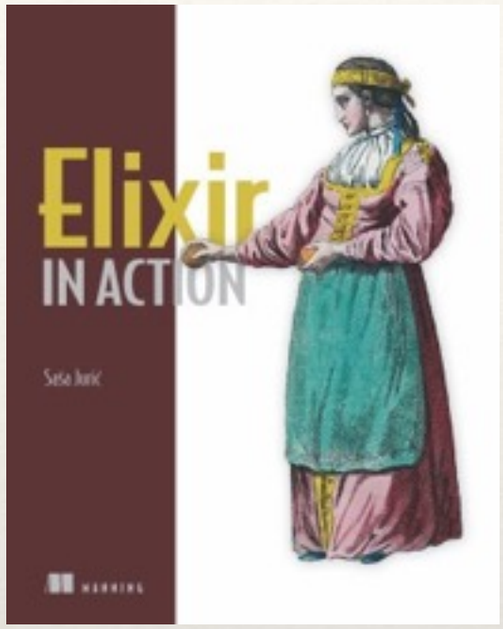
http://www.phoenixframework.org/blog/the-road-to-2-million-websocket-connections

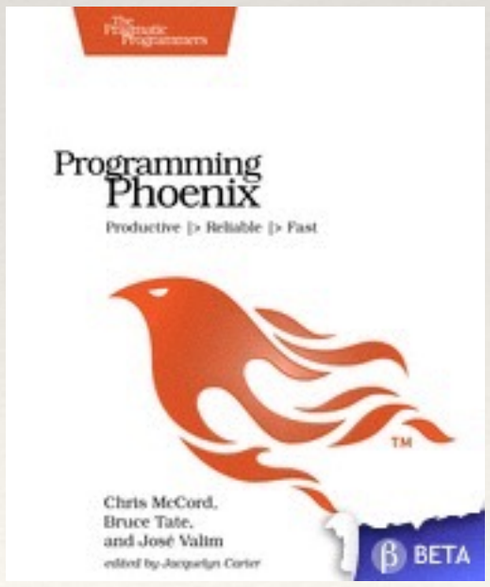


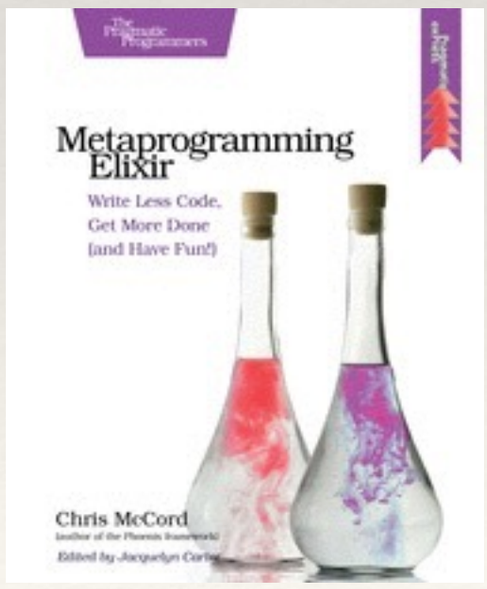
https://github.com/slashmili/talks/tree/master/2016/phoenix-klxrb-march

Books


The Pragmatic Programmers
Programming Elixir 1.2
Functional
|> Concurrent
|> Pragmatic
|> Fun
Dave Thomas
Foreword by
José Valim,
Creator of Elixir
BETA


Elixir
IN ACTION
Saša Jurić


The Pragmatic Programmers
Programming Phoenix
Productive |> Reliable |> Fast
Chris McCord,
Bruce Tate,
and José Valim
edited by Jacquelyn Carter
BETA


The Pragmatic Programmers
Metaprogramming Elixir
Write Less Code,
Get More Done
(and Have Fun!)
Chris McCord

Thanks

slides and codes
http://slashmili.org/talks