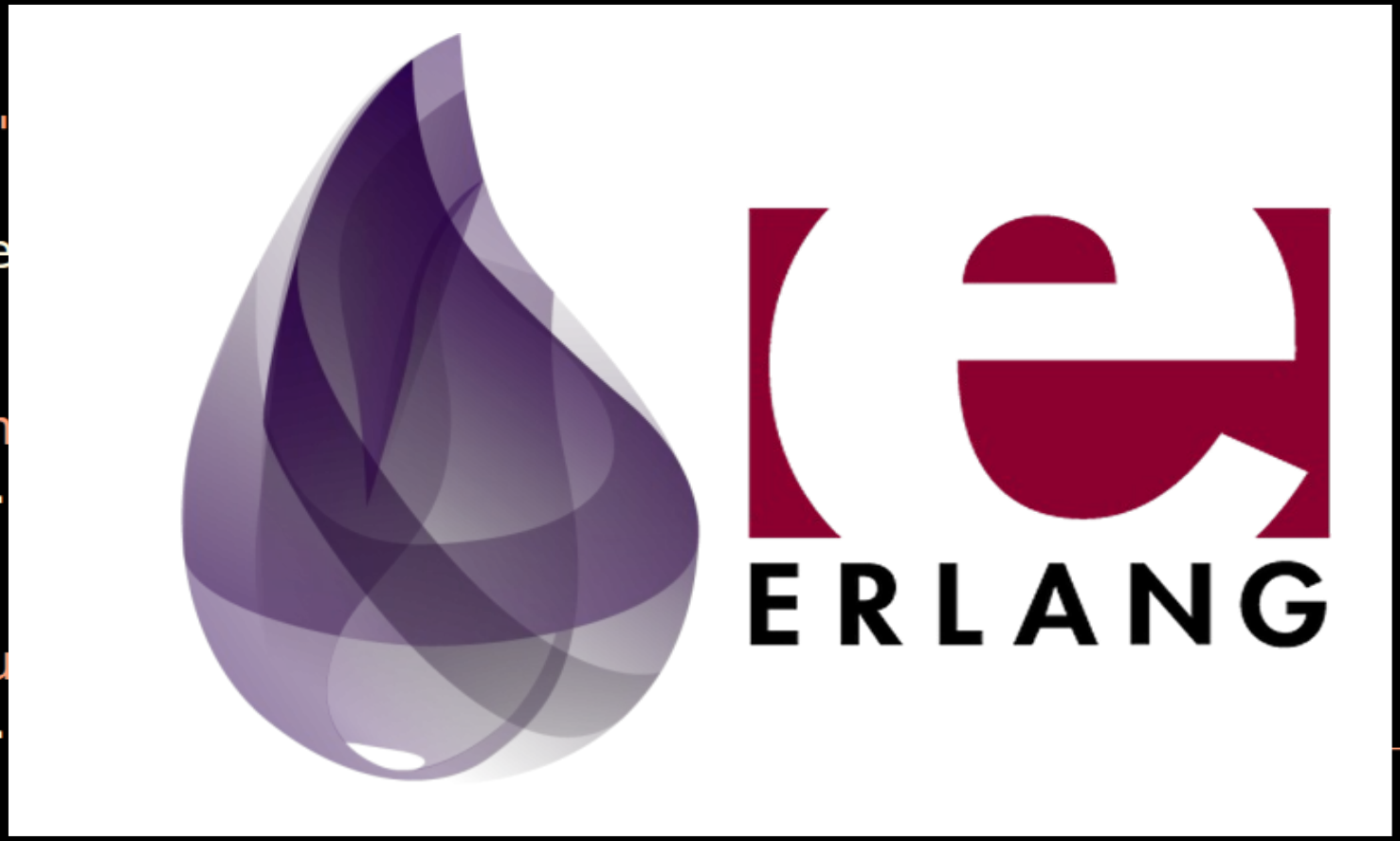

Ruby's influence
over the Elixir language
by Paolo Montrasio
paolo.montrasio@connettiva.eu
http://connettiva.eu/rubyday

```
defmodule ApplicationRouter do
  use Dynamo.Router

  prepare do
    conn.fetch([:cookies, :params])
  end

  get "/" do
    conn = conn.assign(:title, "Welcome to Dynamo!")
    render conn, "index.html"
  end

  get "/hello/world" do
    conn.resp(200, "Hello world")
  end

  put "/users/:user_id" do
    conn.resp 200, "Got user id: #{conn.params[:user_id]}"
  end
end
```

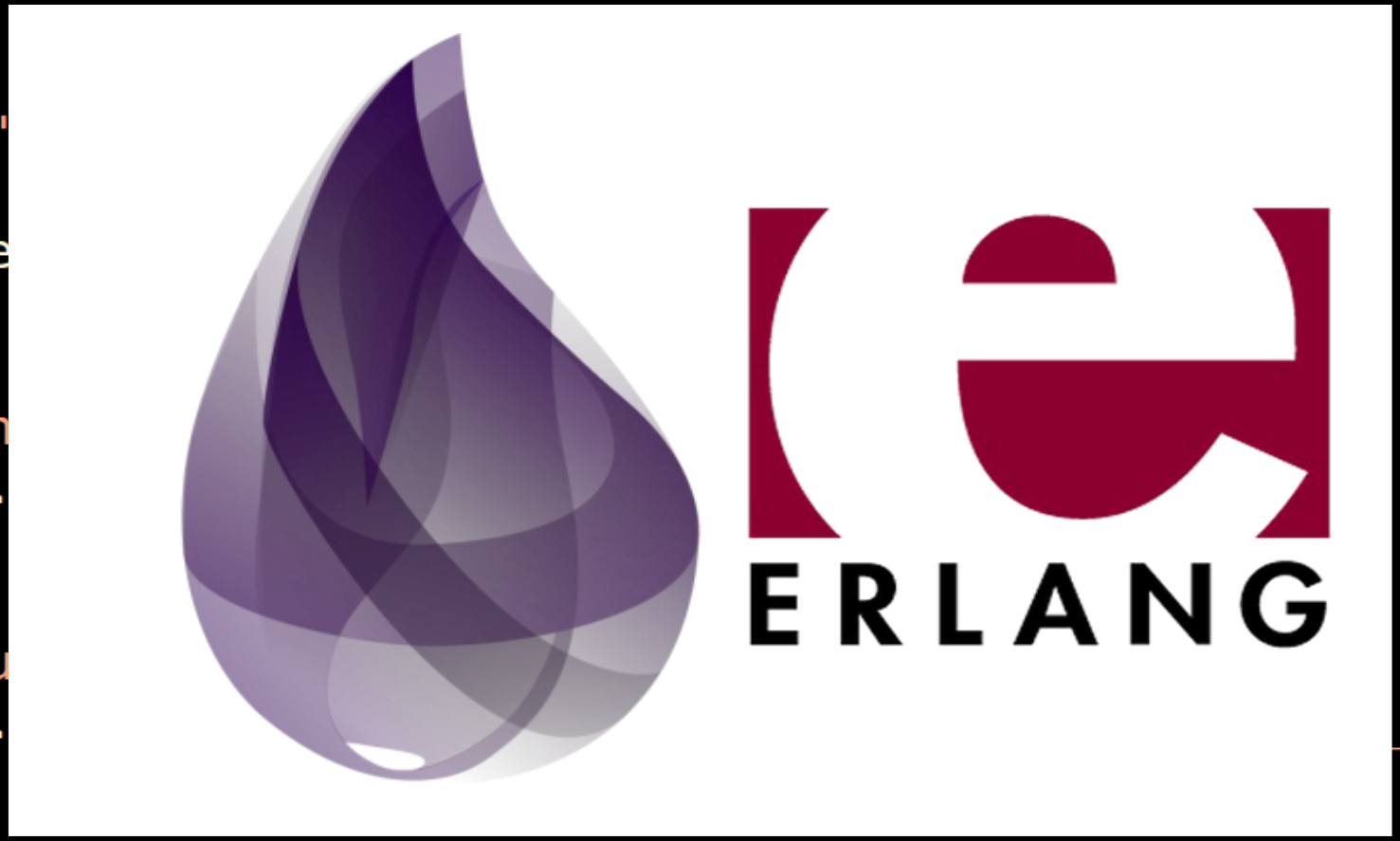
```
defmodule ApplicationRouter do
  use Dynamo.Router

  prepare do
    conn.fetch([:cookies, :
  end

  get "/" do
    conn = conn.a                                  to Dynamo!")
    render conn,
  end

  get "/hello/
    conn.resp(
  end

  put "/users/
    conn.resp 20                    id: #{conn.params[:user_id]}"
  end
end
```

```
defmodule ApplicationRouter do
  use Dynamo.Router

  prepare do
    conn.fetch([:cookies, :params])
  end

  get "/"
    conn
    rende
  end

  get "/h
    conn.
  end

  put "/u
    conn.                                          _id]}"
  end
end
```

```
defmodule ApplicationRouter do
  use Dynamo.Router

  prepare do
    conn.fetch([:cookies, :params])
  end

  get "/"
    conn
    rende
  end

  get "/h
    conn.
  end

  put "/u
    conn.                                          _id]}"
  end
end
```



```
defmodule ApplicationRouter do
  use Dynamo.Router

  prepare do
    conn.fetch([:cookies, :params])
  end

  get "/"
    conn
    rende
  end

  get "/h
    conn.
  end

  put "/u
    conn.                                          _id]}"
  end
end
```



```
# this is a comment

true false nil

parentheses are, optional

&& || !    # work on all data types
and or not # work only on booleans

$ iex
iex(1)> !2
false
iex(2)> not 2
** (ArgumentError) argument error
    :erlang.not(2)
iex(2)>
```

```elixir
if condition do
  ...
end

unless condition do
  ...
end

# one liners
if condition, do: ...
unless condition, do: ...

iex(2)> if !nil, do: "Look ma, one line!"
"Look ma, one line!"
```

```
# atoms are Ruby's symbols
:atoms, :start, :with, :a, :colon

# many functions return atoms
```

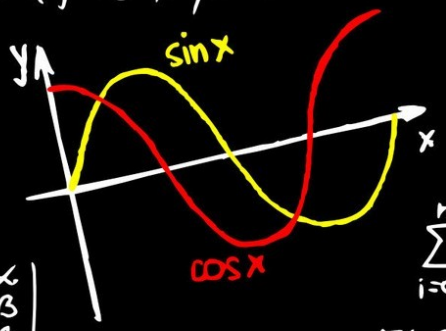
```
$ iex
> x = 1
1
> y = 2
2
> "#{x}, #{y}, 3"
"1, 2, 3"
> IO.puts "#{x}, #{y}, 3"
1, 2, 3
:ok
```

```
$ irb
> x = 1
 => 1
> y = 2
 => 2
> "#{x}, #{y}, 3"
 => "1, 2, 3"
> puts "#{x}, #{y}, 3"
1, 2, 3
 => nil
```

```
# atoms are Ruby's symbols
:atoms, :start, :with, :a, :colon

# many functions return

$ iex
> x = 1                                    1
1
> y = 2                                    2
2
> "#{x}, #{y                                x}, #{y}, 3"
"1, 2, 3"                                  1, 2, 3"
> IO.puts "#                                s "#{x}, #{y}, 3"
1, 2, 3                                    , 3
:ok                                      => nil
```


Differences

$x^3+x^2+y^3+z^3+xyz-6=0$



$grad f = \left(\frac{\partial f}{\partial x}; \frac{\partial f}{\partial y}\right)$

$tg x \cdot ctg x = 1$

$2x^2yy' + y^2 = 2$

$x_1 = -11p, x_2 = -p, x_3 = 7p, p \in \mathbb{R}$

$Y_{i+1} = Y_i + b \cdot k_2$

$B = \begin{pmatrix} 2 & 1 & -1 & 0 \\ 3 & 0 & 1 & 2 \end{pmatrix}$

$a^2 = b^2 + c^2 - 2bc\cos\alpha$

$tg\frac{x}{2} = \frac{1-\cos x}{\sin x} = \frac{\sin x}{1+\cos x}$

$X_2 = \begin{pmatrix} -\alpha \\ \beta \\ -\gamma \\ -\delta \end{pmatrix}$

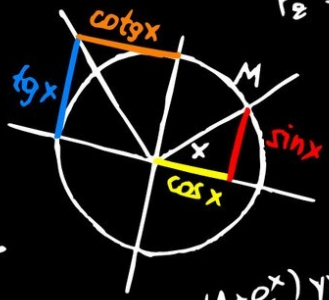
$\sum_{i=0}^{n} (p_2(x_i) - y_i)^2$

$tg 2x = \frac{2tgx}{1-tg^2x}$

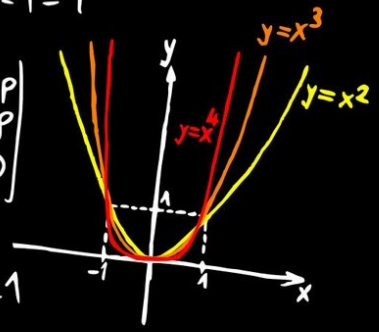
$tgx = \frac{\sin x}{\cos x}$

$\lambda x - y + z = 1$
$x + \lambda y + z = \lambda$
$x + y + \lambda z = \lambda^2$

$F_z = 2xyz - 1 = 1$



$X_1 = \begin{pmatrix} 2p \\ -p \\ 0 \end{pmatrix}$



$\iiint_M z\,dx\,dy\,dz = \int_0^{2\pi}\left(\int_0^2\left(\int_{\frac{1}{2}r}^{1} r\,r\,dr\right)dz\right)d\varphi$

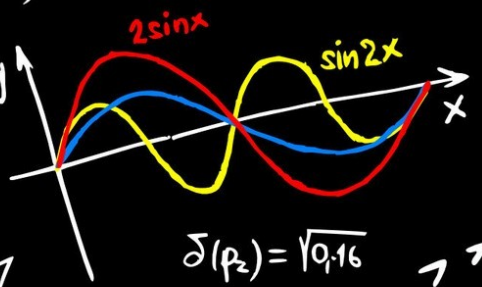
$\lim_{n\to+\infty} \frac{\sqrt{n^3+1}+n}{\sqrt[3]{3n^2+2n-1}}$

$2arctgx - x = 0, I = (1,10)$

$\frac{a}{\sin\alpha} = \frac{b}{\sin\beta} = \frac{c}{\sin\gamma}$

$(1+e^x)yy' = e^x$
$y(1) = 1$

$\int_{-\pi/2}^{\pi/2} \sin^4 x \cdot \cos^3 x\,dx$



$y = \sqrt[3]{x+1}; x = tgt$

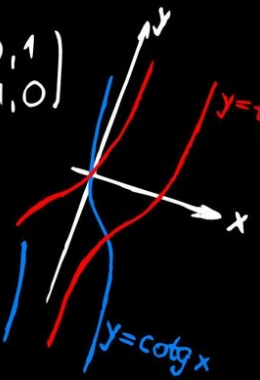
$X_1 = \begin{pmatrix} \alpha+\beta x \\ \beta \end{pmatrix}$

$\cos 2x = \cos^2 x - \sin^2 x$

$\cos^2\alpha + \cos^2\beta + \cos^2\gamma = 1$

$\delta(p_2) = \sqrt{0.16}$

$C = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$



$A+B+C = 8$
$-3A-7B+2C = -10,3$
$-18A+6B-3C = 15$

$\frac{\partial z}{\partial x} = 2; \frac{\partial z}{\partial y} = 0$

$\vec{n} = (F_x'; F_y'; F_z')$

$a^2 + b^2 = c^2$

$\alpha, \beta, \gamma \in C$

$\sin^2 x + \cos^2 x = 1$

$\int R\left(x, \sqrt[n]{\frac{ax+b}{cx+d}}\right)dx$

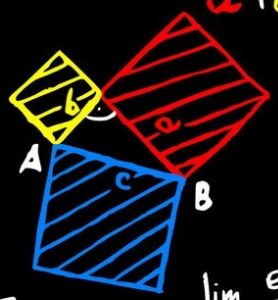
$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 0$

$f(x) = 2^{-x} + 1, \varepsilon = 0.005$

$\lambda_2 = i\sqrt{14}$

$\frac{\sin x}{x} \le \frac{x}{x} = 1$

$\sin 2x = 2\sin x \cdot \cos x$



$e^z - xyz = e; A[0;e;1]$

$\frac{2x}{x^2+2y^2} = 2$

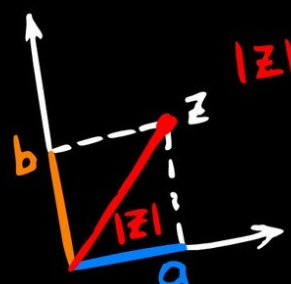
$z = \frac{1}{x}\arcsin\frac{\sqrt{2}}{2}$

$\eta_1 = \lambda_1^2 - 3\lambda_1 + 1 \neq 0$

$\lim_{x\to 0} \frac{e^{2x}-1}{5x} = \frac{2}{5}$

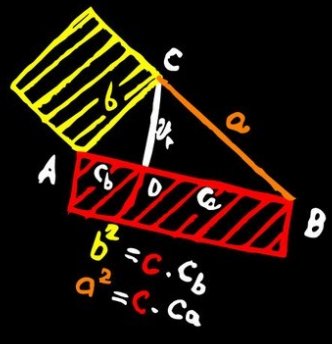
$|\alpha| + |\beta| \neq 0; \gamma \neq 0$

$|z| = \sqrt{a^2+b^2}$



$\sin(x+y) = \sin x\cos y + \cos x \sin y$

$y' - \frac{\sqrt{y}}{x+2} = 0; y(0) = 1$



$D\left(\frac{\partial f}{\partial x}\right) = 16 - x^2 + 16y^2 - 4z > 0$

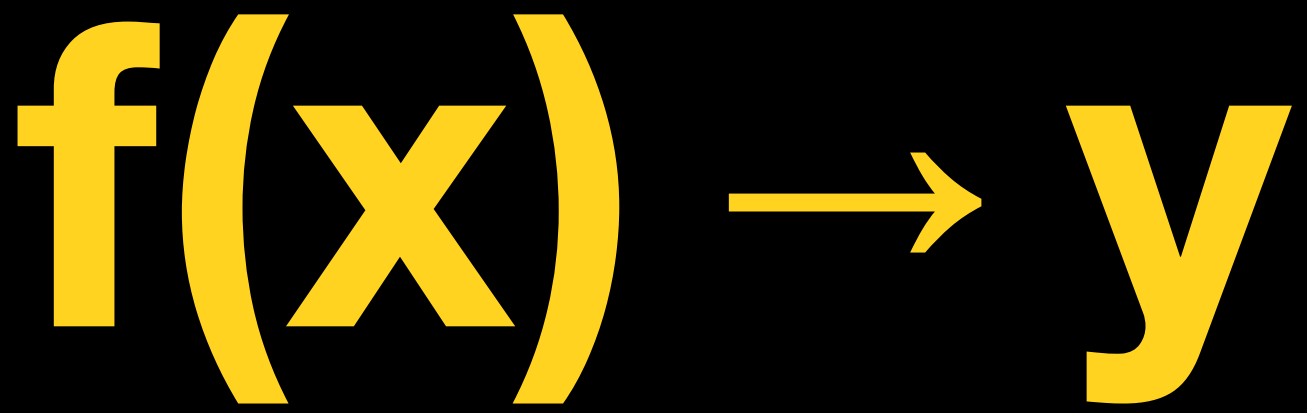
$A = \begin{pmatrix} x & 1+x^2 & 1 \\ y & 1+y^2 & 1 \\ z & 1+z^2 & 1 \end{pmatrix}; x=0, y=1, z=2$

$\cos\varphi = \frac{(1;0)\cdot\left(\frac{1}{2\sqrt{3}}; \frac{1}{4\sqrt{3}}\right)}{\sqrt{\frac{1}{12}+\frac{1}{48}}}$

$\oint 3x^{7} + 1{,}66x^{-0{,}17}\,dx$

$\lim_{n\to+\infty}\left(1+\frac{3}{n}\right)^n$

$A = [1;0;3]$


f(x) → y

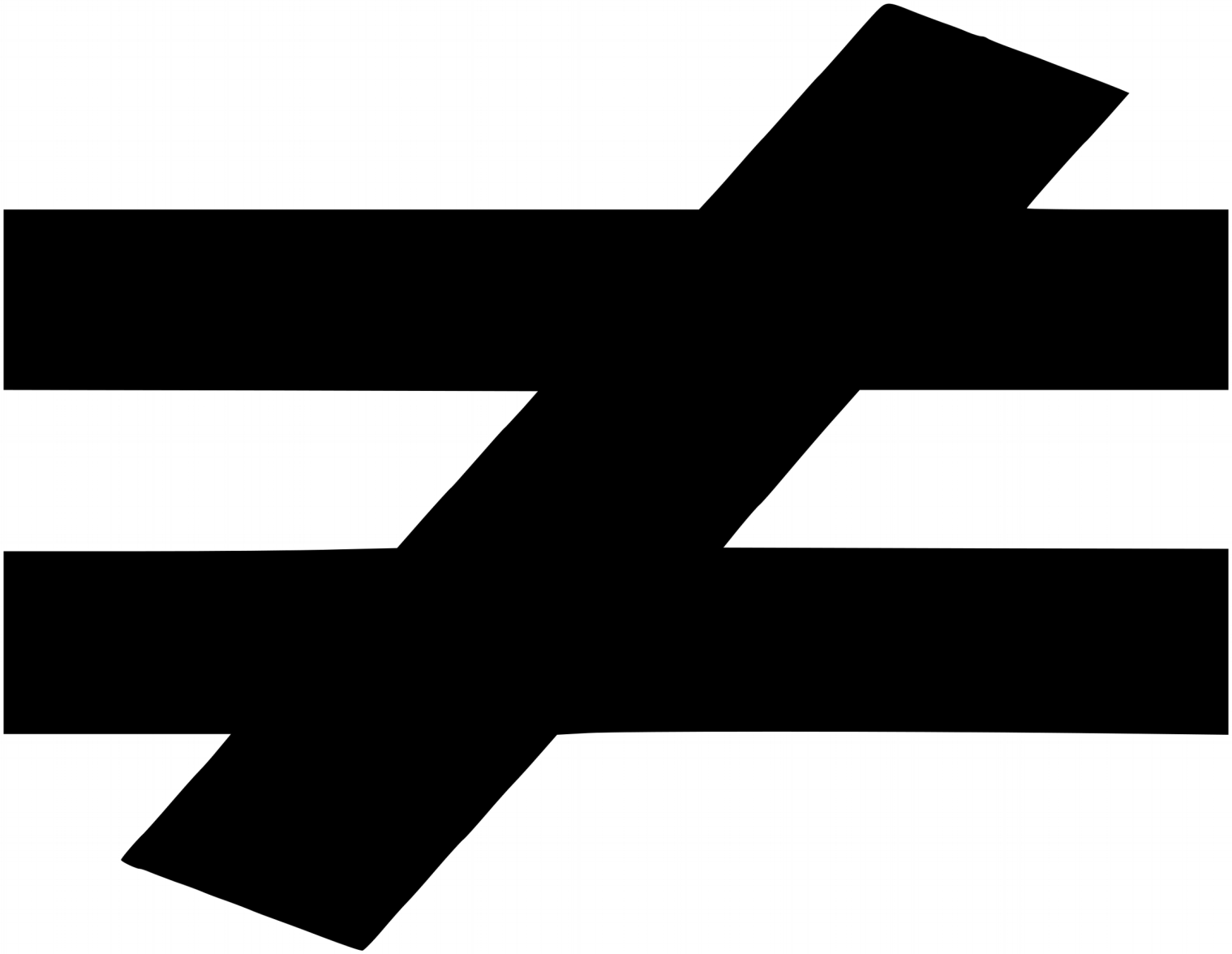
```
# Variables are IMMUTABLE

iex(1)> x = "abcd"
"abcd"
iex(2)> x = String.replace(x, "a", "A")
"Abcd"
iex(3)> x
"Abcd"

# But it mutated!

$ erl
1> X = 1.
1
2> X = 2.
** exception error: no match of right hand side value 2
```

```ruby
# The OO way
Ruby.is.an.object.oriented.language

# The functional way (Yoda)
language(oriented(object(an(is(Ruby)))))

# Ruby
[1, 2, [3, 4], 5].flatten.reverse.map{|n| n*n}
 => [25, 16, 9, 4, 1]

# Elixir
Enum.map(Enum.reverse(List.flatten([1, 2, [3, 4], 5])),
         fn n -> n * n end)
[25, 16, 9, 4, 1]
```


I IZ SAD

```ruby
# Ruby
[1, 2, [3, 4], 5].flatten.reverse.map{|n| n*n}
 => [25, 16, 9, 4, 1]

# Elixir
[1, 2, [3, 4], 5] |> List.flatten |>
  Enum.reverse |> Enum.map(fn n -> n * n end)
```

```elixir
# Elixir has no loops
defmodule AreYou do
  def bored? do
    Float.floor(:random.uniform * 5, 0) == 0.0
  end
end

defmodule Listen do
  def to_me do
    IO.puts "Listen to me"
    # Recursion with tail optimization
    unless AreYou.bored?, do: Listen.to_me
  end
end

:random.seed(:os.timestamp)
Listen.to_me
```



```
# match operator on tuples

iex(1)> c = :cont
:cont
iex(2)> {:stop, x} = {c, 3}
** (MatchError) no match of right hand side
   value: {:cont, 3}

iex(2)> c = :stop
:stop
iex(3)> {:stop, x} = {c, 3}
{:stop, 3}
iex(4)> x
3
```

```elixir
# read a file
case File.read "/home/me/elixir/doc.txt" do
  {:ok, content}  -> do_something_with(content)
  {:error, error} -> log(error)
end

# get a web page
:inets.start()
{:ok, {status, headers, content}} =
  :httpc.request "http://www.example.com"
{:ok, file} = File.open "index.html", [:write, :utf8]
IO.binwrite file, content
File.close file
```


Strings

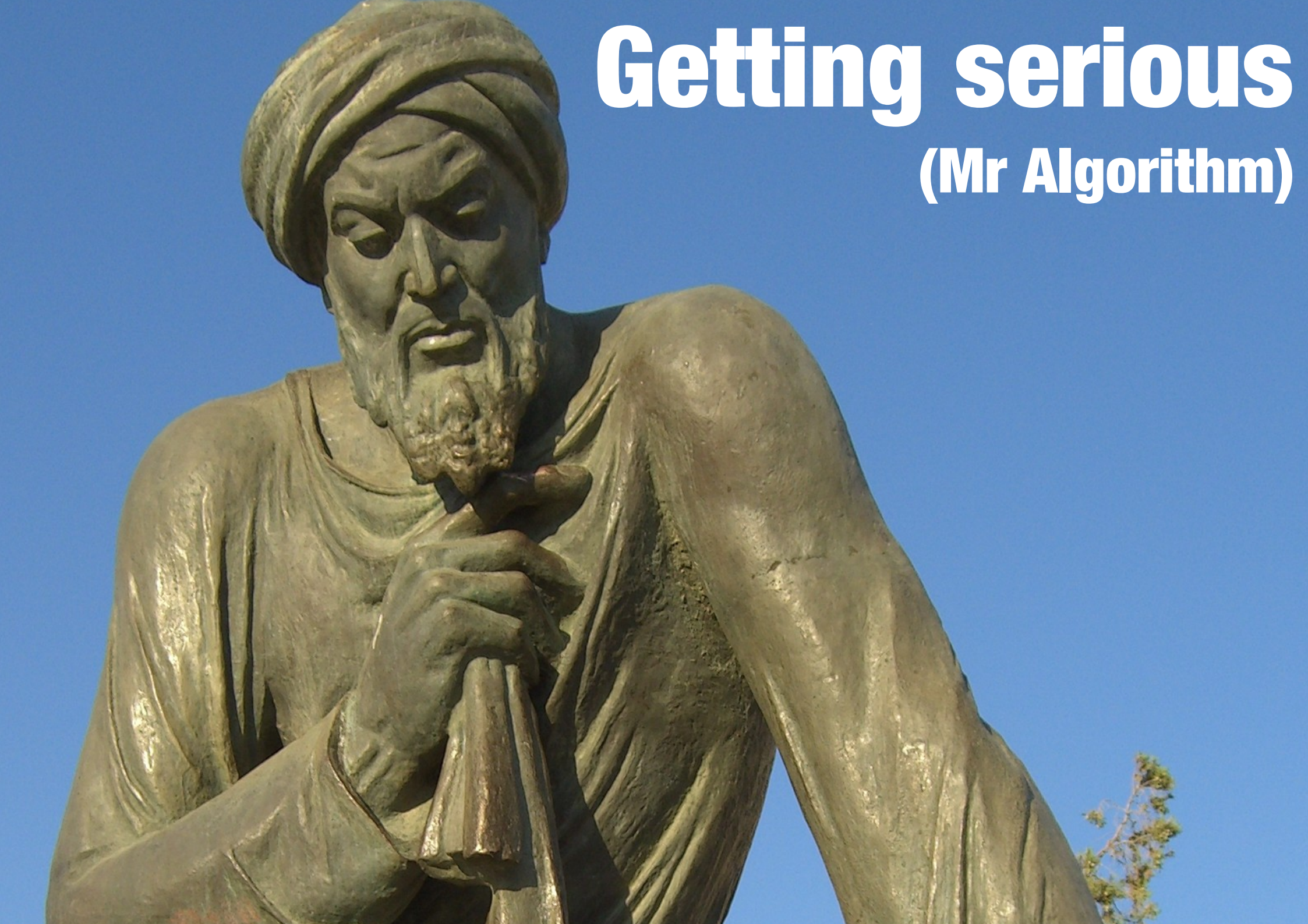
```
iex(1)> "a" == "a"
true
iex(2)> 'a' == 'a'
true
iex(3)> "a" == 'a'
false
iex(4)> "è utf8" # a string
"è utf8"
iex(5)> 'è utf8' # list of characters
[232, 32, 117, 116, 102, 56] # [ ] are lists
iex(6)> to_char_list "è utf8"
[232, 32, 117, 116, 102, 56]
iex(7)> to_string 'è utf8'
"è utf8"
iex(8)> IO.puts '#{x}, #{y}' # '' interpolate
1, 2
:ok
iex(9)> "foo" <> "bar"       # ugly as hell
"foobar"
```

Lists and arrays

```
# This is a list not an array
iex(1)> x = [1, 2, 3]
[1, 2, 3]

# Concatenation
iex(2)> [1, 2, 3] ++ [4, 5, 6]
[1, 2, 3, 4, 5, 6]

# Head and tail of a list
iex(3)> [y|z] = x
[1, 2, 3]
iex(4)> y   # head
1
iex(5)> z   # tail
[2, 3]
iex(6)> List.last(x)
3
```

```elixir
keyword_list = [{:a, "a"}, {:b, "b"}]
map = %{:a => "a", 2 => "b"}

@moduledoc
@doc

defmodule Language do
  defstruct name: "Ruby", age: 19
end

# comprehension, not a loop
for dir  <- dirs,
    file <- File.ls!(dir),
    path = Path.join(dir, file),
    File.regular?(path) do
  File.rm!(path)
end
```


12
1
2
3
4
5
6
7
8
9
10
11



Getting serious

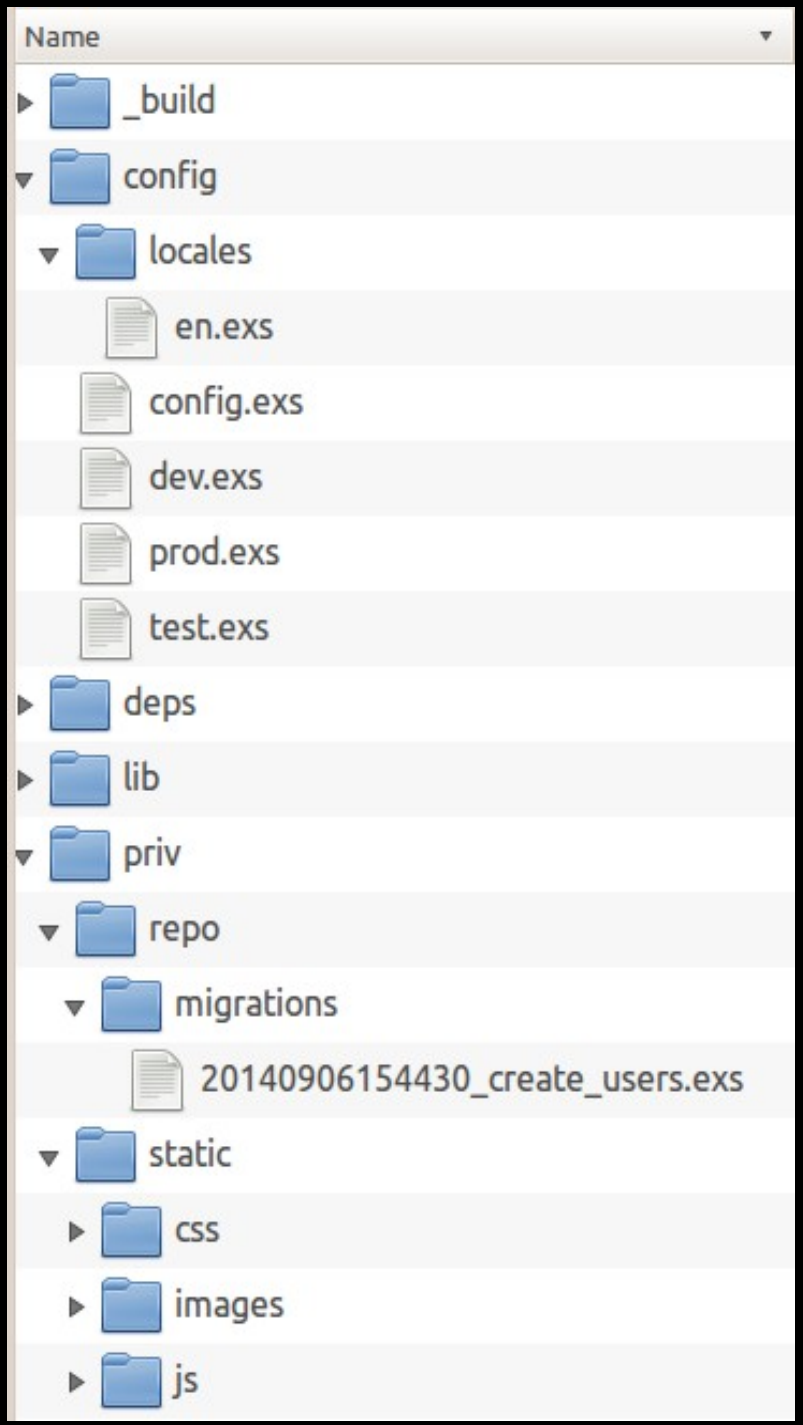
(Mr Algorithm)

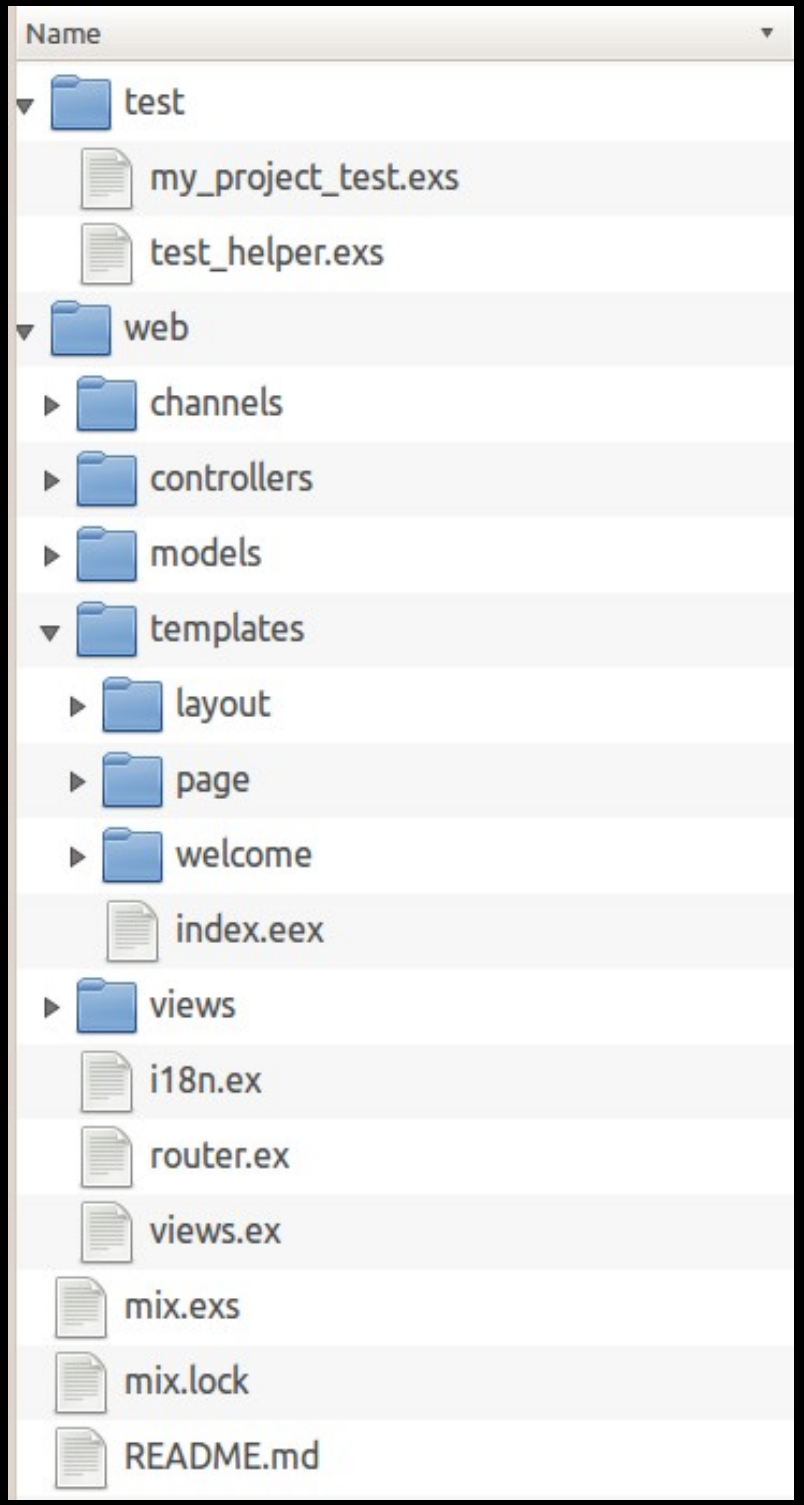
```
#!/usr/bin/env elixir
defmodule Server do
  def echo do
    # actor based concurrency
    receive do
      {client, message} -> IO.puts "server: #{message}"
          send client, message
    end
    echo # Tail recursion http://xkcd.com/1270/
  end
end

# could spawn to a different machine by Node.connect
pid = spawn fn -> Server.echo end
send pid, {self, "Hi"}
receive do
  message -> IO.puts "client: #{message}"
end

$ ./spawn.exs
server: Hi
client: Hi
```

```
$ git clone https://github.com/phoenixframework/phoenix.git
$ cd phoenix

# mix is rake + bundle
# mix.exs is Rakefile + Gemfile
# mix.lock is Gemfile.lock
$ mix do deps.get, compile

# From this directory! Important!
$ mix phoenix.new my_project ~/my_project

$ cd ~/my_project
$ mix do deps.get, compile  # bundle
$ mix phoenix.start         # rails s

http://localhost:4000

$ mix help                  # rake -T
$ mix phoenix.routes        # rake routes
```

```elixir
defmodule MyProject.Mixfile do
  use Mix.Project

  def project do
    [ app: :my_project,
      version: "0.0.1",
      elixir: "~> 1.0.0-rc1",
      elixirc_paths: ["lib", "web"],
      deps: deps ]
  end

  def application do
    [
      mod: { MyProject, [] },
      applications: [:phoenix, :cowboy, :logger]
    ]
  end

  defp deps do
    [
      {:phoenix, "0.4.0"},
      {:cowboy, "~> 1.0.0"}
    ]
  end
end
```

```elixir
%{"cowboy": {:package, "1.0.0"},
  "cowlib": {:package, "1.0.0"},
  "linguist": {:package, "0.1.2"},
  "phoenix": {:package, "0.4.0"},
  "plug": {:package, "0.7.0"},
  "poison": {:package, "1.0.3"},
  "ranch": {:package, "1.0.0"}}
```


Name
_build
config
locales
en.exs
config.exs
dev.exs
prod.exs
test.exs
deps
lib
priv
repo
migrations
20140906154430_create_users.exs
static
css
images
js


Name
test
my_project_test.exs
test_helper.exs
web
channels
controllers
models
templates
layout
page
welcome
index.eex
views
i18n.ex
router.ex
views.ex
mix.exs
mix.lock
README.md

Rails	Phoenix
controllers	controllers
models	models
views	templates
helpers	views but they also render templates
config env files	config
config/routes.rb	web/router.ex
lib	lib
	channels bidirectional controllers, websockets

app/view/layouts/application.html.erb

```
<%= yield %>
```

web/templates/layout/application.html.eex

```
<%= @inner %>
```


DIY
WD-40

```
$ psql -U postgres
# create role my_project login password 'password';
CREATE ROLE
# create database my_project owner my_project
encoding='UTF8' lc_collate='en_US.UTF-8' lc_ctype='en_US.UTF-8';
CREATE DATABASE
# grant all on database my_project to my_project;
GRANT
# \q

$ vi mix.exs
defp deps do
  ...
  {:postgrex, ">= 0.5.0"}
  ...
end
```

```
$ vi mix.exs
defp deps do
  ...
  {:ecto, "~> 0.2.0"}
  ...
end

$ vi lib/my_project/repo.ex
  defmodule Repo do
    use Ecto.Repo, adapter: Ecto.Adapters.Postgres

    def conf do
      parse_url "ecto://my_project:postgres@localhost/my_project"
    end

    def priv do
      app_dir(:my_project, "priv/repo")
    end
  end

$ mix ecto.gen.migration Repo create_users
Compiled lib/my_project/repo.ex
Generated my_project.app
* creating priv/repo/migrations
* creating priv/repo/migrations/20140906154430_create_users.exs
```

```
# No DSL yet!
# The up and down functions must return the SQL to execute

$ vi priv/repo/migrations/20140906154430_create_users.exs
defmodule Repo.Migrations.CreateUsers do
  use Ecto.Migration

  def up do
    "CREATE TABLE users(id serial primary key, content varchar(140))"
  end

  def down do
    "DROP TABLE users"
  end
end

$ mix ecto.migrate Repo
* running UP _build/dev/lib/my_project/priv/repo/migrations/201409061
54430_create_users.exs
```

```
$ vi web/router.ex

defmodule MyProject.Router do
  use Phoenix.Router

  scope alias: MyProject do
    get "/", WelcomeController, :index, as: :root
  end

  resources "/users", UsersController do
    resources "/pages", PagesController
  end

  scope path: "/admin", alias: MyProject.Admin, helper: "admin" do
    resources "/users", UsersController
  end

end
```

```
$ vi web/router.ex

defmodule MyProject.Router do
  use Phoenix.Router

  scope alias: MyProject do
    get "/", WelcomeController, :index, as: :root
  end

  resources "/users", UsersController do
```

```
get  "/users",          UserController, :index
get  "/users/:id",      UserController, :show
get  "/users/new",      UserController, :new
post "/users",          UserController, :create
get  "/users/:id/edit", UserController, :edit
put  "/users/:id",      UserController, :update
delete "/users/:id",    UserController, :destroy
```

```
en
```

No Devise

Enter mail Password Sign in

Forgot your password? Didn't receive activation instructions?

DIY with the cookie session store

```elixir
defmodule MyProject.User do
  use Ecto.Model
  import Ecto.Query

  schema "users" do
    field :email,    :string
    field :password, :string
  end

  validate user,
    email:    present(),
    password: present()

  def encrypt_password(plaintext) do
    :base64.encode(:crypto.hash(:sha256, to_char_list(plaintext)))
  end

  def find(email, plaintext_password) do
    encrypted_password = encrypt_password(plaintext_password)
    query = from u in MyProject.User,
      where: u.email == ^email and u.password == ^encrypted_password,
      select: u
    Repo.all(query)
  end
end
```

```elixir
defmodule MyProject.Admin.UsersController do
  use Phoenix.Controller
  require Logger
  alias MyProject.User
  require Authentication
  require AdminsOnly

  plug Authentication
  plug AdminsOnly

  def show(conn, _params) do
    %{ "id" => user_id } = _params
    { user_id, _ } = Integer.parse(user_id)
    user = Repo.get(User, user_id)
    render conn, "show", user: user
  end
end
```

```
<ul>
<%= if @current_user == nil do %>
  <li><a href="<%= Router.sessions_path(:new) %>">Login</a></li>
<% else %>
  <li><a href="<%= Router.users_path(:show, @current_user.id) %>">
      <%= @current_user.email %></a></li>
  <li><a href="<%= Router.sessions_path(:destroy) %>">Logout</a></li>
<% end %>
</ul>

<%= for notice <- Flash.get_all(@conn, :notice) do %>
  <div class="container">
    <div class="row">
      <div class="alert alert-success" role="alert">
        <p><%= notice %></p>
      </div>
    </div>
  </div>
<% end %>
```

The equal in <%= if cond do %> is important


Is it any better?

```
# i7-4700MQ laptop 4 cores, 2 threads per core

$ iex
Erlang/OTP 17 [erts-6.0] [source] [64-bit] [smp:8:8]
[async-threads:10] [hipe] [kernel-poll:false]

Interactive Elixir (1.0.0-rc1) - press Ctrl+C to exit
(type h() ENTER for help)
iex(1)>

# very light weight processes

# spawn processes vs create objects
# send messages between processes
```


RAND
<>
T
RND
GOSUB
★★
H
SQR
REM
STEP
E
TAN
BREAK
£
SPACE
REM
STEP
E
TAN
NEXT
<
N
NOT
DIM
SLOW
D
ARCTAN