µSvcs with Elixir

Rodrigo Nonose

EmCasa

µSvcs with Elixir

Rodrigo Nonose

EmCasa

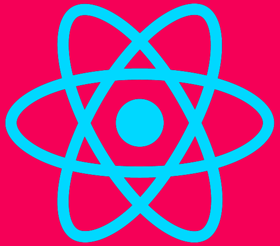
ORDEM E PROGRESSO

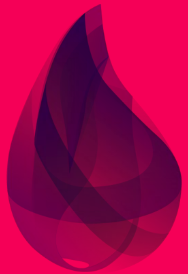
ORDEM E PROGRESSO

ORDEM E PROGRESSO

emcasa.com

emcasa

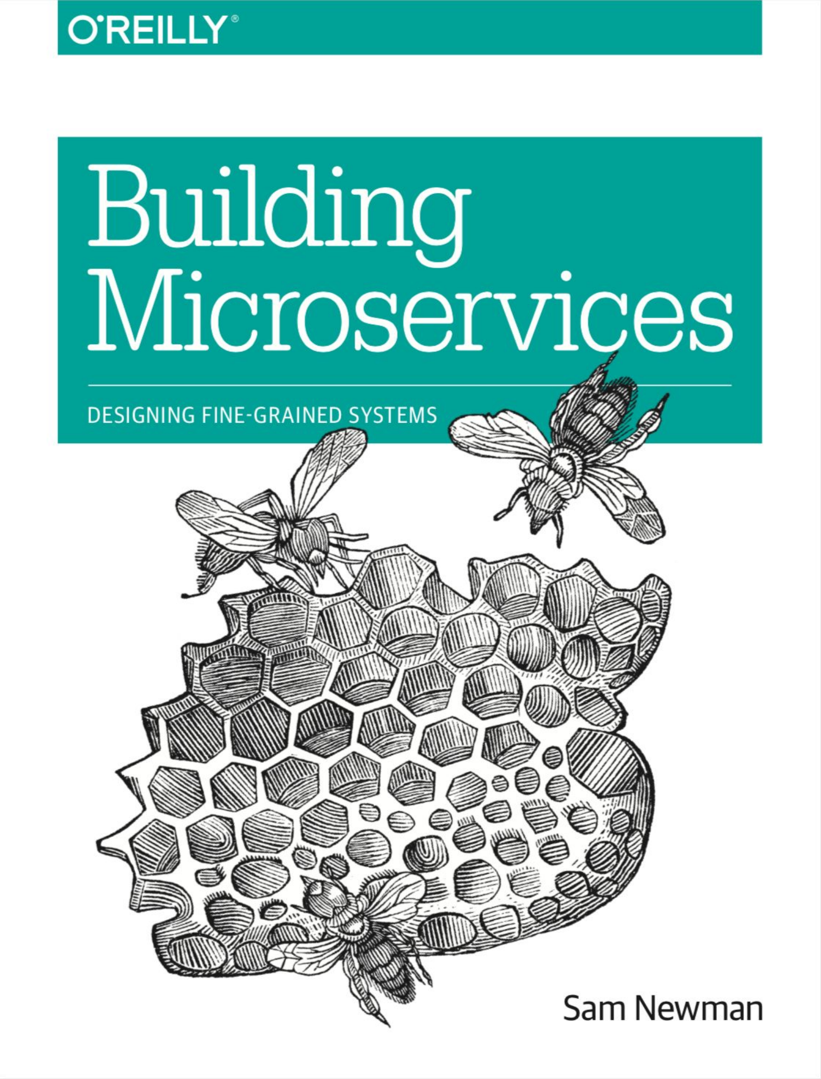




EmCasa

µSvcs

EmCasa



O'REILLY®

Building Microservices

DESIGNING FINE-GRAINED SYSTEMS

Sam Newman

Replaceability

Single Responsibility Principle

Composability

Organizational Alignment

Ease of Deployment

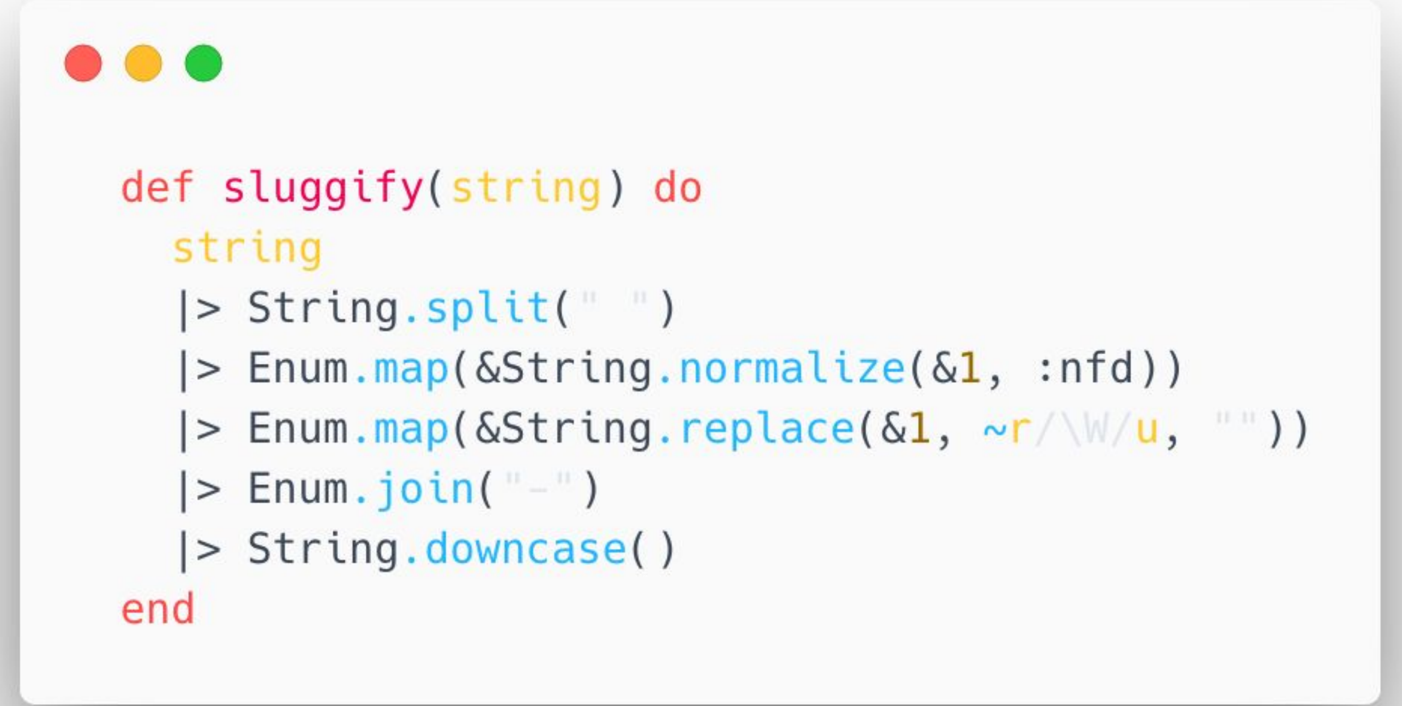
Scaling

Autonomy

Resilience

Technological Heterogeneity

Elixir



```
def sluggify(string) do
  string
  |> String.split(" ")
  |> Enum.map(&String.normalize(&1, :nfd))
  |> Enum.map(&String.replace(&1, ~r/\W/u, ""))
  |> Enum.join("-")
  |> String.downcase()
end
```

~~Replaceability~~

Single Responsibility Principle

~~Composability~~

Organizational Alignment

Ease of Deployment

Scaling

Autonomy

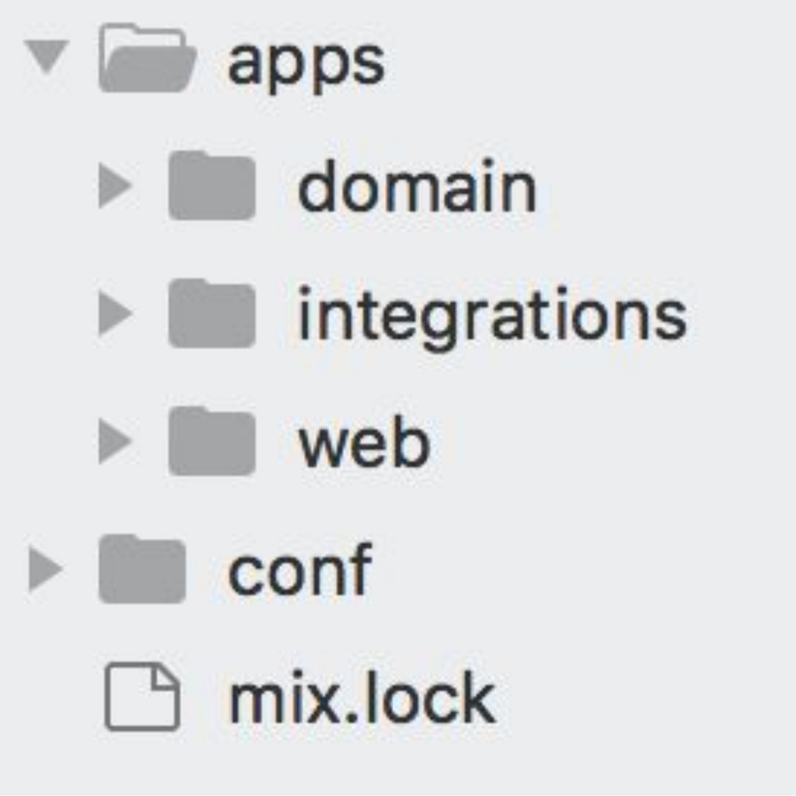
Resilience

Technological Heterogeneity

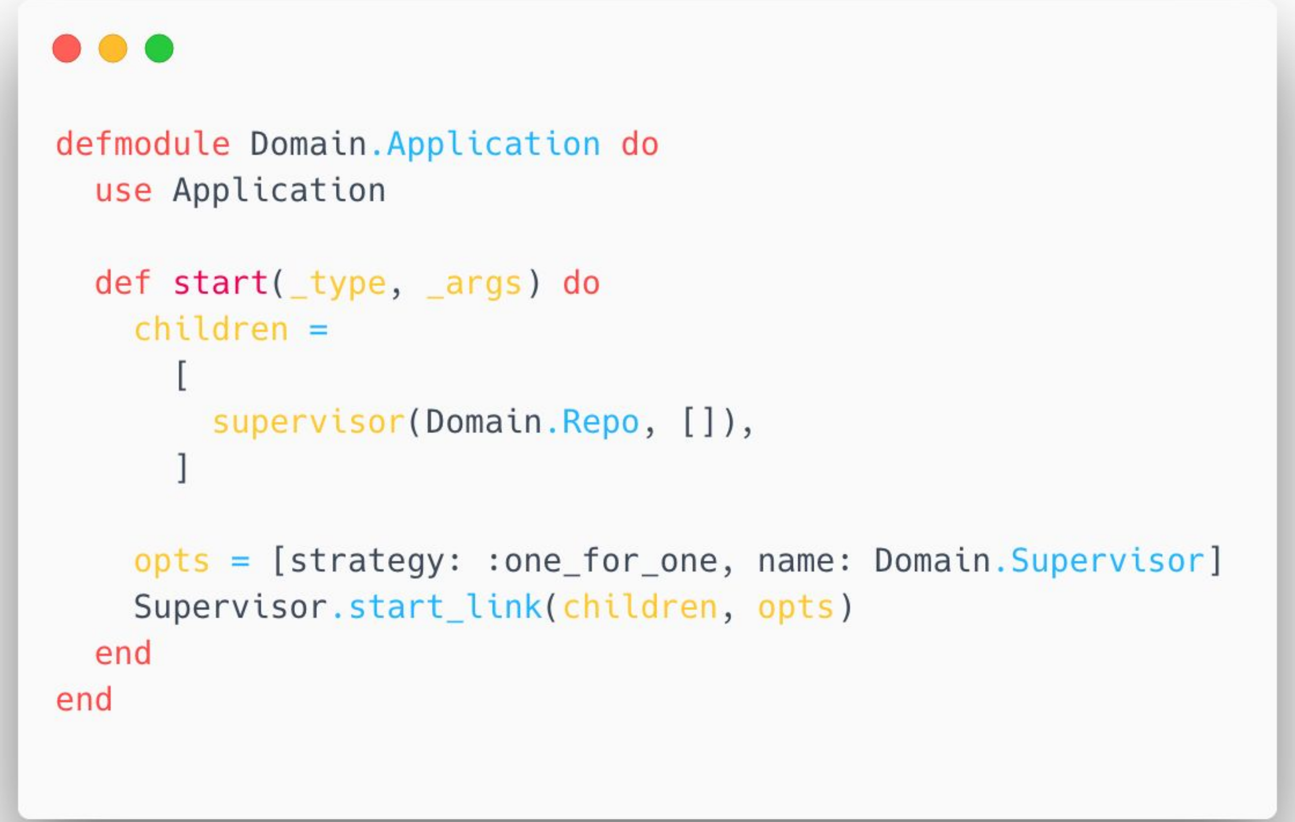
Umbrella Applications

EmCasa

Umbrella applications



```
▼ apps
    ▶ domain
    ▶ integrations
    ▶ web
▶ conf
  mix.lock
```

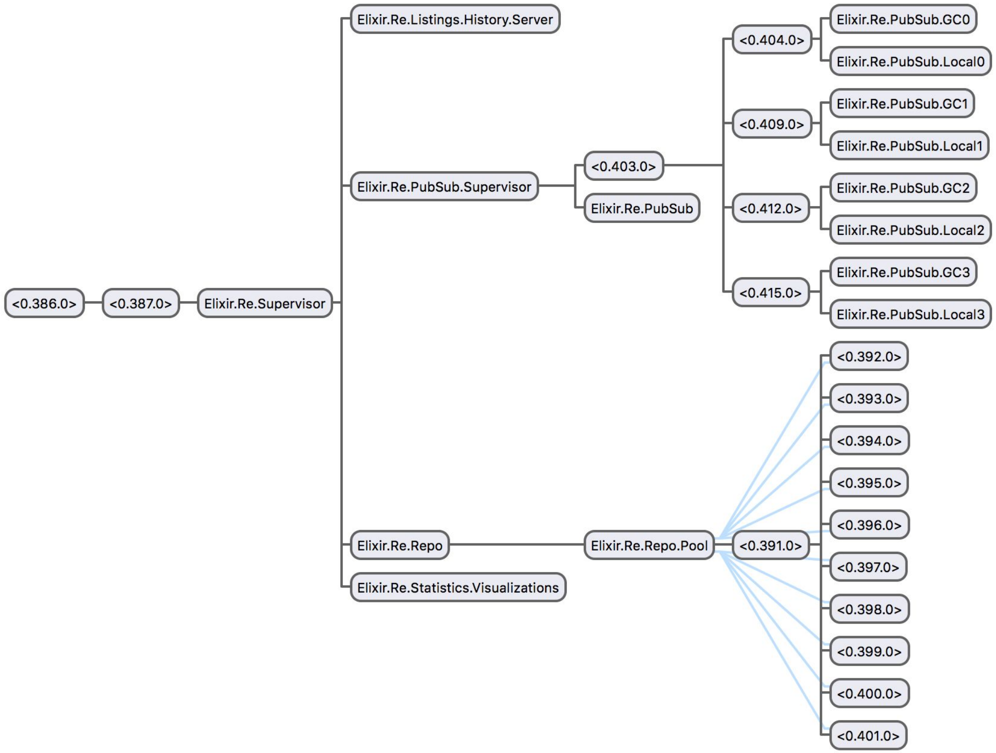


```elixir
defmodule Domain.Application do
  use Application

  def start(_type, _args) do
    children =
      [
        supervisor(Domain.Repo, []),
      ]

    opts = [strategy: :one_for_one, name: Domain.Supervisor]
    Supervisor.start_link(children, opts)
  end
end
```

Umbrella applications



<0.386.0>

<0.387.0>

Elixir.Re.Supervisor

Elixir.Re.Listings.History.Server

Elixir.Re.PubSub.Supervisor

<0.403.0>

<0.404.0>

Elixir.Re.PubSub.GC0

Elixir.Re.PubSub.Local0

<0.409.0>

Elixir.Re.PubSub.GC1

Elixir.Re.PubSub.Local1

<0.412.0>

Elixir.Re.PubSub.GC2

Elixir.Re.PubSub.Local2

<0.415.0>

Elixir.Re.PubSub.GC3

Elixir.Re.PubSub.Local3

Elixir.Re.PubSub

Elixir.Re.Repo

Elixir.Re.Repo.Pool

<0.391.0>

<0.392.0>

<0.393.0>

<0.394.0>

<0.395.0>

<0.396.0>

<0.397.0>

<0.398.0>

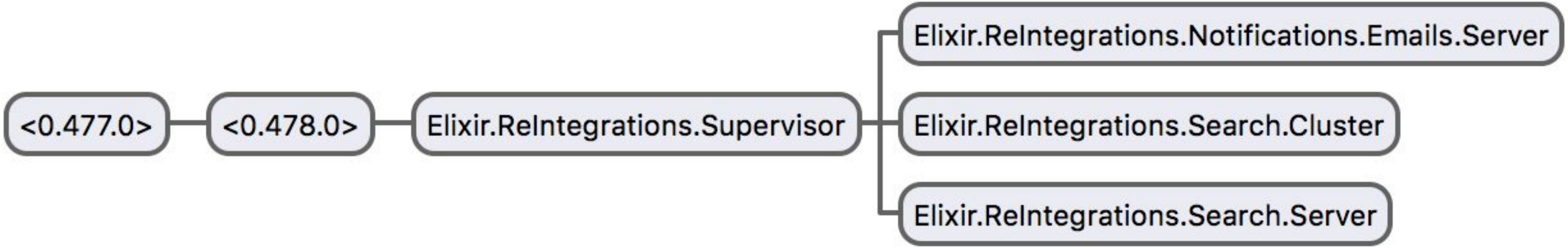
<0.399.0>

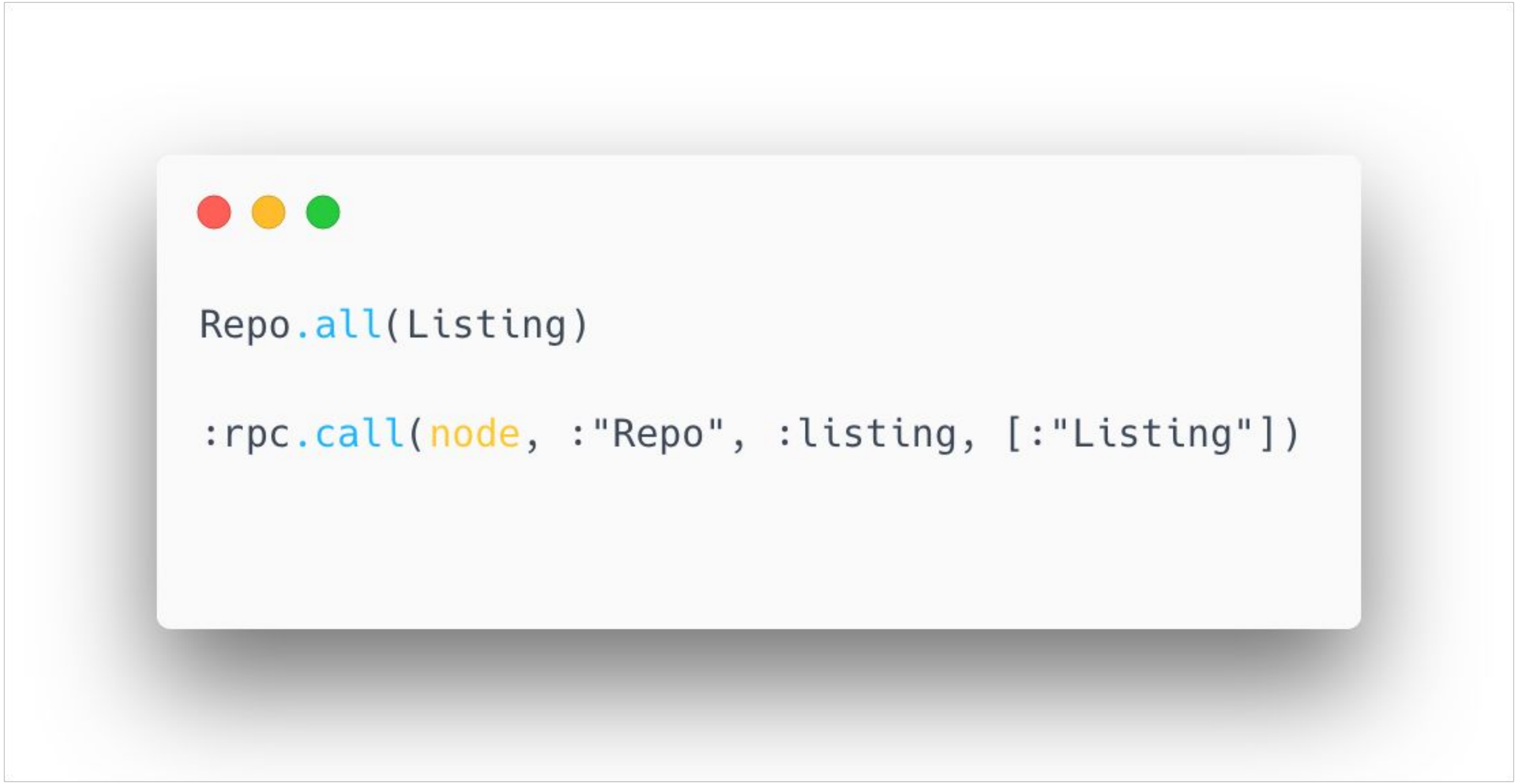
<0.400.0>

<0.401.0>

Elixir.Re.Statistics.Visualizations

Umbrella applications


<0.477.0>
<0.478.0>
Elixir.ReIntegrations.Supervisor
Elixir.ReIntegrations.Notifications.Emails.Server
Elixir.ReIntegrations.Search.Cluster
Elixir.ReIntegrations.Search.Server



```
Repo.all(Listing)

:rpc.call(node, :"Repo", :listing, [:"Listing"])
```

~~Replaceability~~

~~Single Responsibility Principle~~

~~Composability~~

~~Organizational Alignment~~

~~Ease of Deployment~~

Scaling

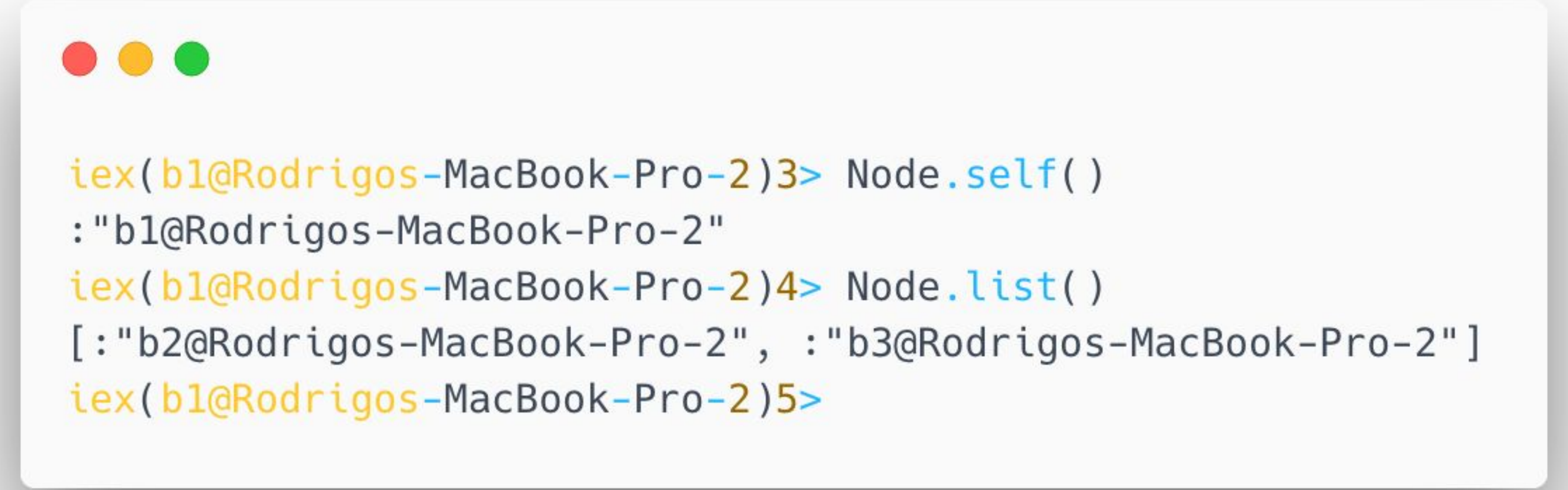
~~Autonomy~~

Resilience

Technological Heterogeneity

Clustering

EmCasa



```
iex(b1@Rodrigos-MacBook-Pro-2)3> Node.self()
:"b1@Rodrigos-MacBook-Pro-2"
iex(b1@Rodrigos-MacBook-Pro-2)4> Node.list()
[:"b2@Rodrigos-MacBook-Pro-2", :"b3@Rodrigos-MacBook-Pro-2"]
iex(b1@Rodrigos-MacBook-Pro-2)5>
```

Clustering

- Node takeover
- Node failover
- Load-balancing and process distribution
- Stateful process hand-off
- Work stealing
- Cluster healing

~~Replaceability~~

~~Single Responsibility Principle~~

~~Composability~~

~~Organizational Alignment~~

~~Ease of Deployment~~

~~Scaling~~

~~Resilience~~

~~Autonomy~~

Technological Heterogeneity

Ports

EmCasa

~~Replaceability~~

~~Single Responsibility Principle~~

~~Composability~~

~~Organizational Alignment~~

~~Ease of Deployment~~

~~Scaling~~

~~Autonomy~~

~~Resilience~~

~~Technological Heterogeneity~~

Domain-Driven Design

EmCasa

By Frits Ahlefeldt

Thank you

rhnonose

backendwizard

EmCasa