elixir

OTP - Basics

Ruben Amortegui
@ramortegui
http://rubenamortegui.com
https://github.com/ramortegui

Agenda

- OTP
- Concepts
- OTP Basics
  - GenServer
  - Supervisor
  - Application
- Case EcCart
- Summary

Elixir – OTP

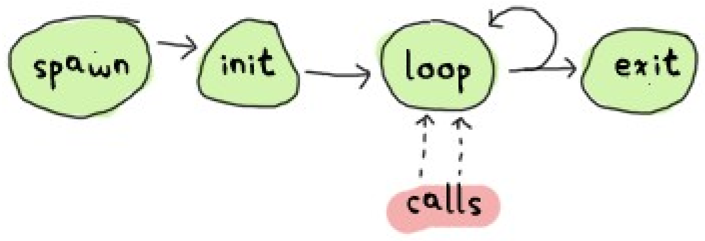
- OTP stands for Open Telecom Platform.
- Is a general purpose tool for developing and managing large systems.
- Provides tools, libraries, conventions and defines a structure for your application.

Elixir – About OTP

- Features included in Erlang/Elixir/OTP:
  - Erlang interpreter and compiler
  - Standard libraries
  - Dialyzer, a static analysis tool
  - Mnesia, a distributed database
  - Erlang Term Storage (ETS)
  - A debugger
  - An event tracer
  - A release management tool (hot swap)

Elixir – Concepts

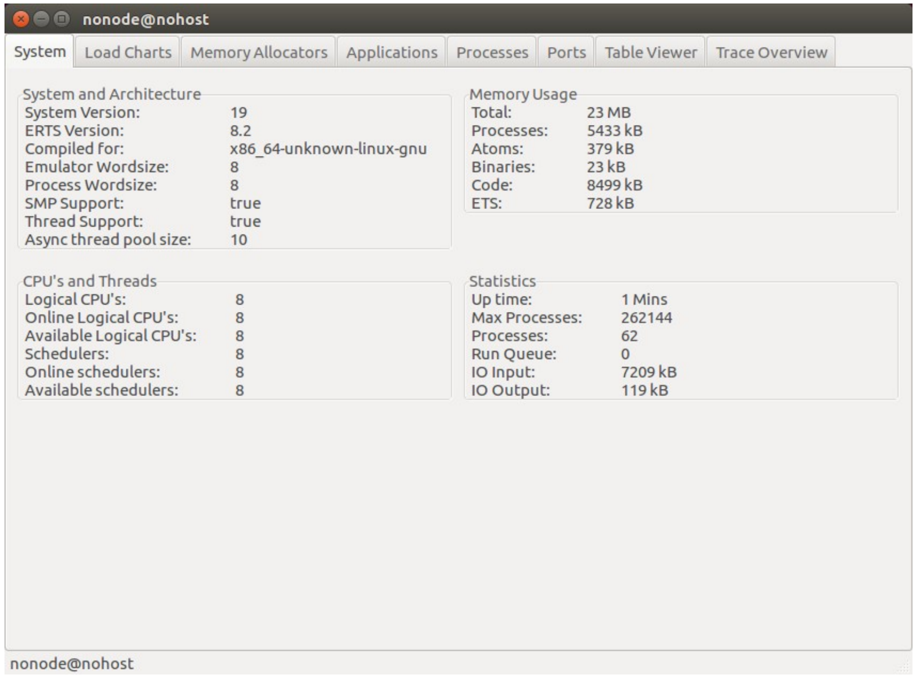
- Process
  - Fundamental part of concurrency
  - Are light weight
  - Doesn’t share memory
  - Implemented using tail recursion



Taken from: http://learnyousomeerlang.com/event-handlers

Elixir – Concepts

- Process



System and Architecture

```
System Version:          19
ERTS Version:            8.2
Compiled for:            x86_64-unknown-linux-gnu
Emulator Wordsize:       8
Process Wordsize:        8
SMP Support:             true
Thread Support:          true
Async thread pool size:  10
```

Memory Usage

```
Total:       23 MB
Processes:   5433 kB
Atoms:       379 kB
Binaries:    23 kB
Code:        8499 kB
ETS:         728 kB
```

CPU's and Threads

```
Logical CPU's:            8
Online Logical CPU's:     8
Available Logical CPU's:  8
Schedulers:               8
Online schedulers:        8
Available schedulers:     8
```

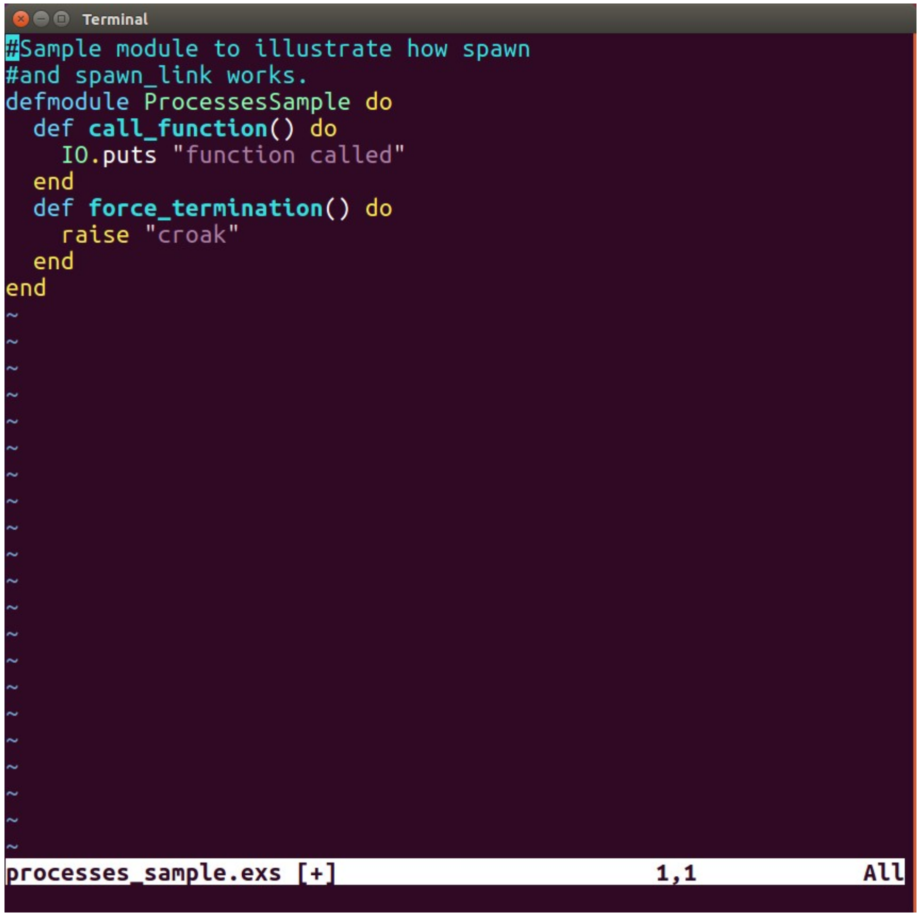
Statistics

```
Up time:         1 Mins
Max Processes:   262144
Processes:       62
Run Queue:       0
IO Input:        7209 kB
IO Output:       119 kB
```

nonode@nohost

Elixir – Concepts

- Process (recap spawn, spawn_link)

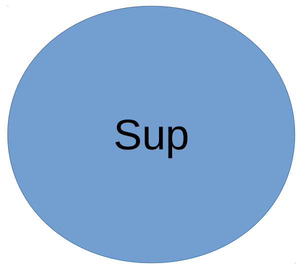


```
#Sample module to illustrate how spawn
#and spawn_link works.
defmodule ProcessesSample do
  def call_function() do
    IO.puts "function called"
  end
  def force_termination() do
    raise "croak"
  end
end
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
processes_sample.exs [+]                               1,1           All
```

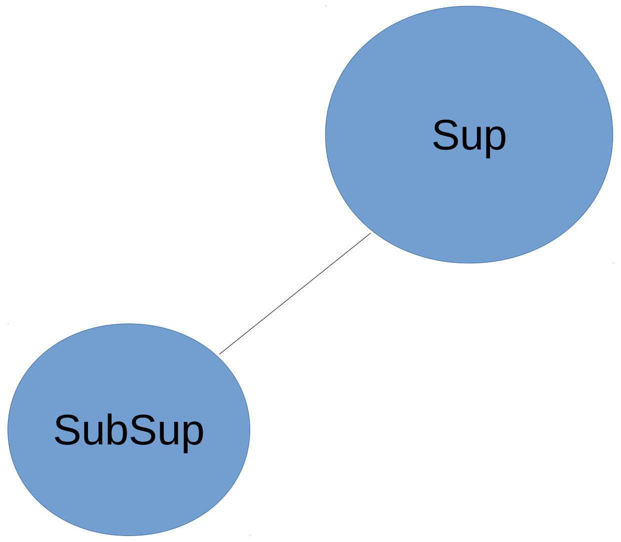
Elixir – Concepts

- Behaviours
  - Code that implements a common pattern.
  - OTP provides:
    - GenServer
    - Supervisor
    - Application

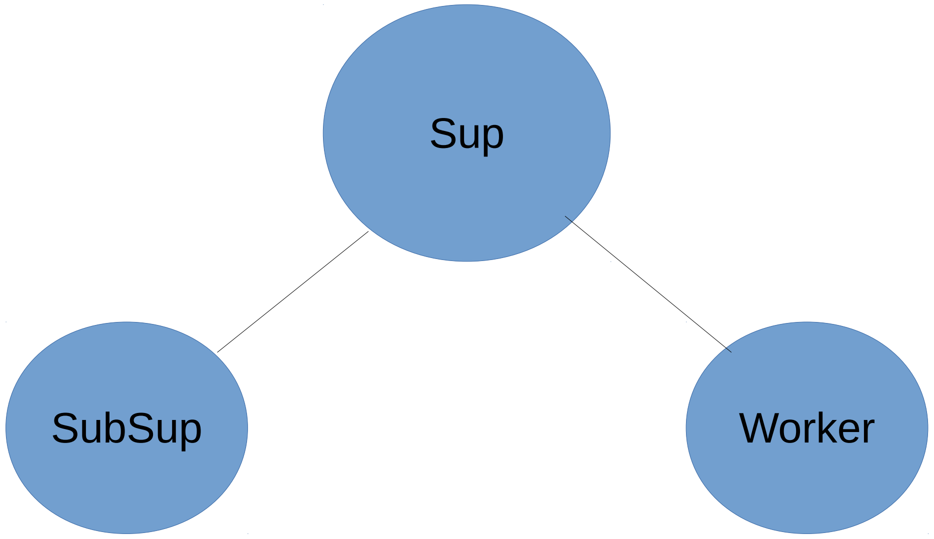
Elixir – About OTP



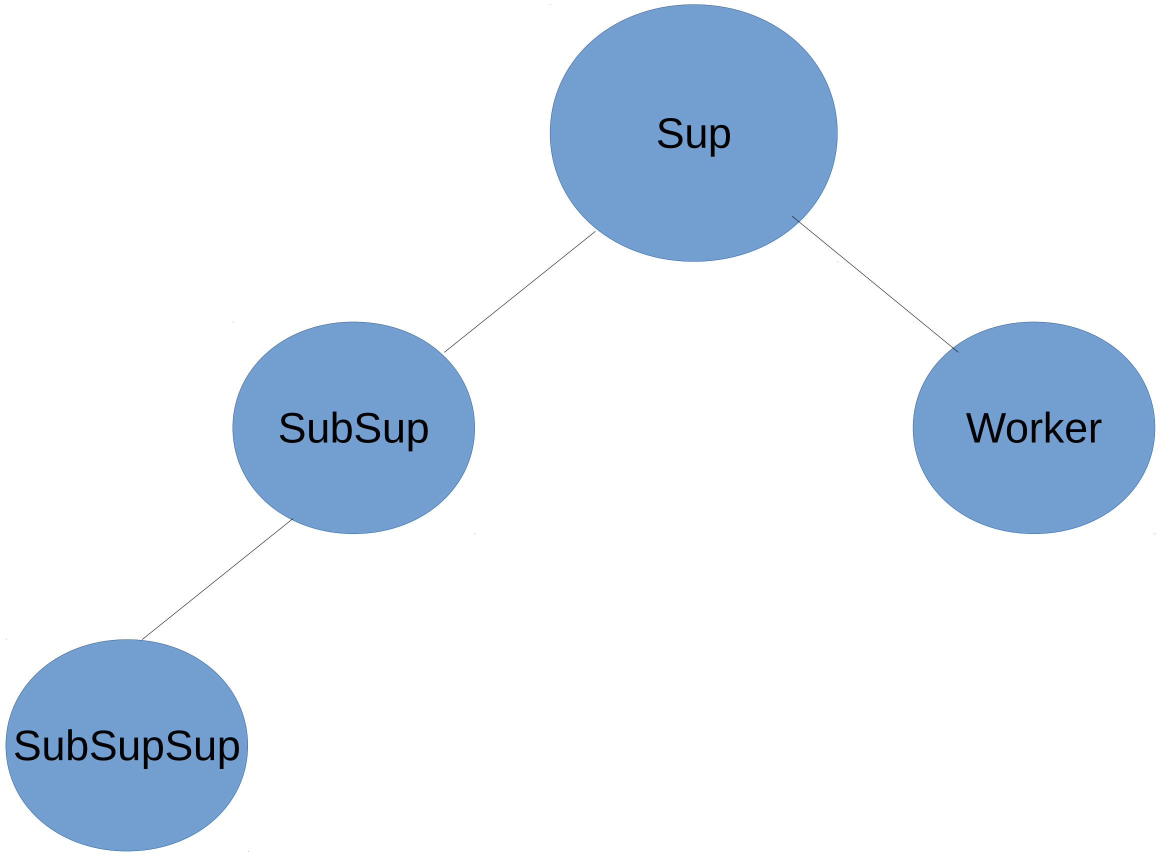
Elixir – About OTP



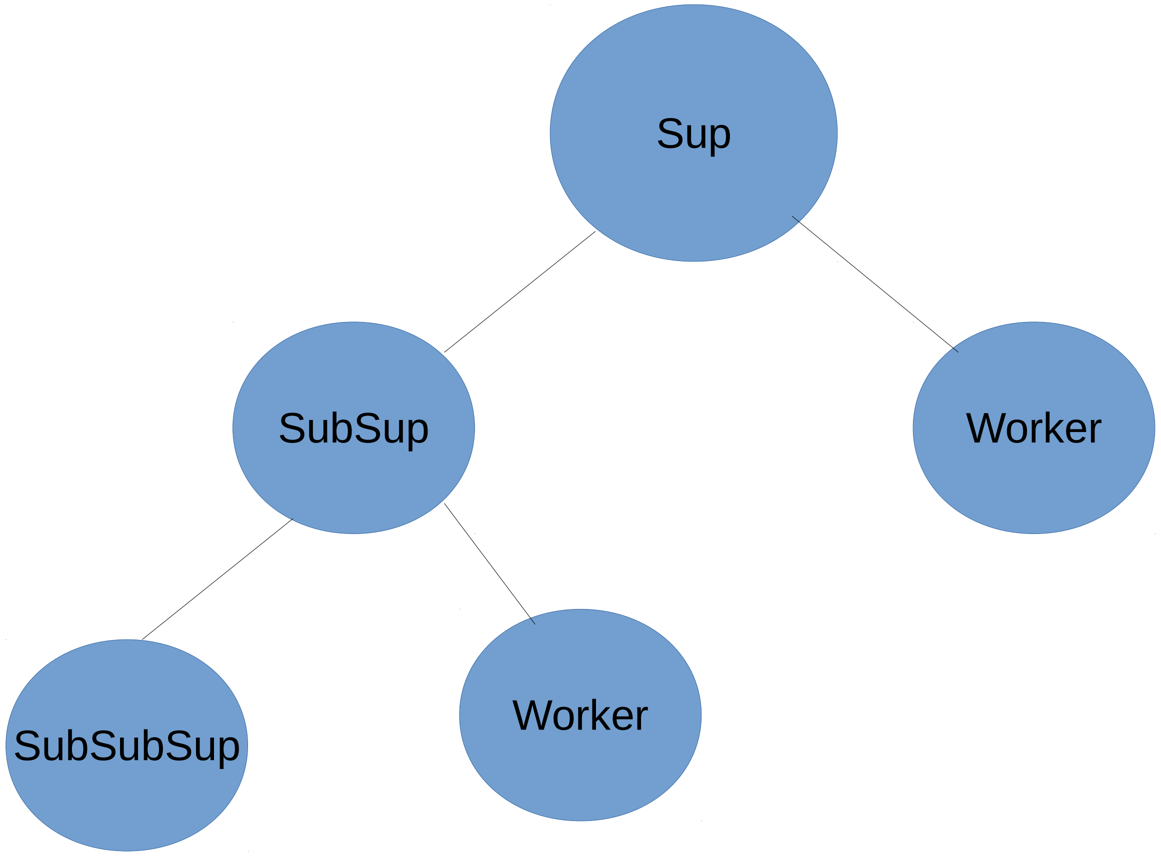
Elixir – About OTP



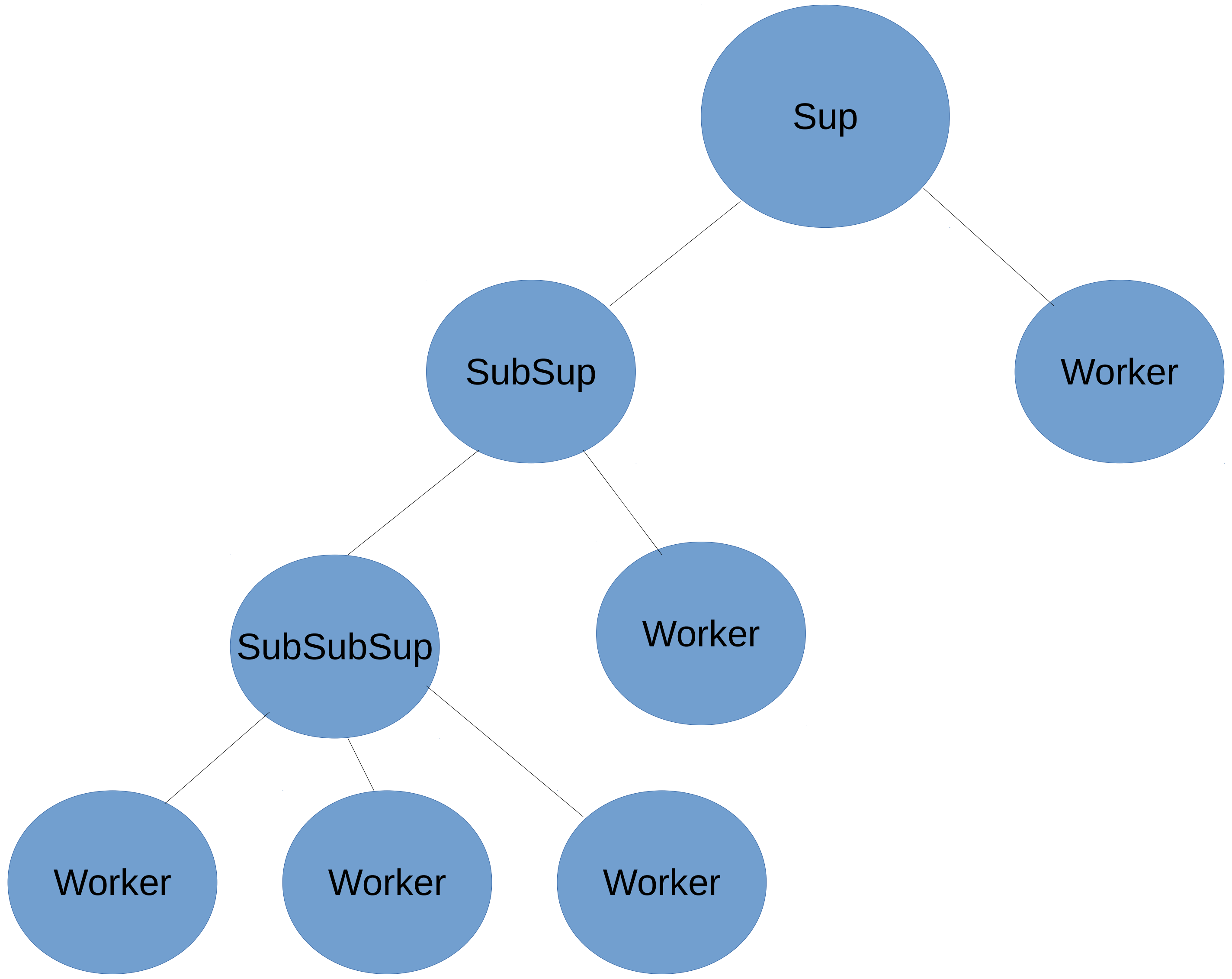
Elixir – About OTP



Elixir – About OTP



Elixir – About OTP



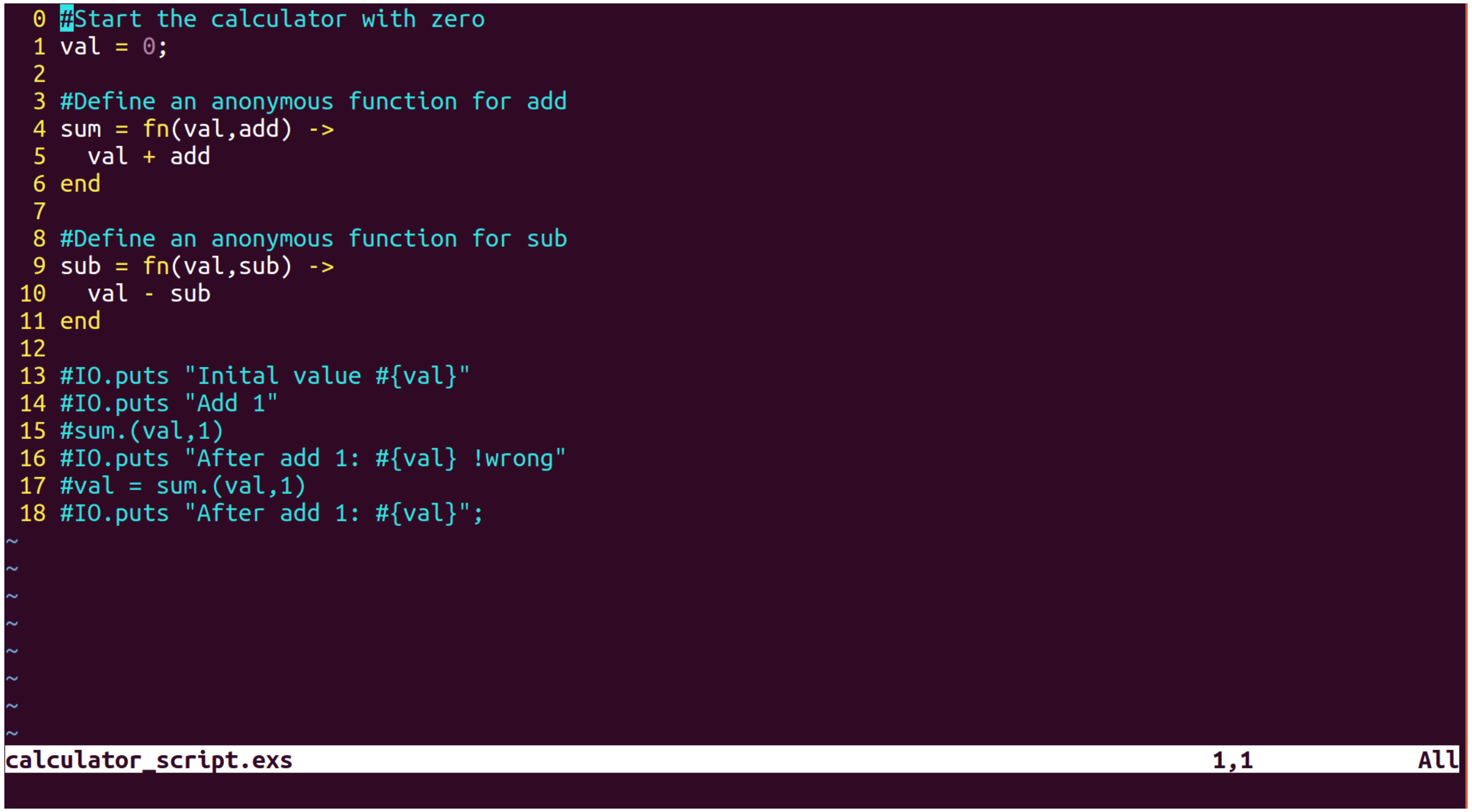
Elixir – About OTP

- Erlang implements Actor model
  - Actor can communicate with others by messages.
  - Actor encapsulates the state.
  - When and Actor is processing messages, the actor can designate a new state.

Elixir – OTP case

- Use a calculator of example of how can evolve on a basic OTP architecture.

Elixir – Script Calculator

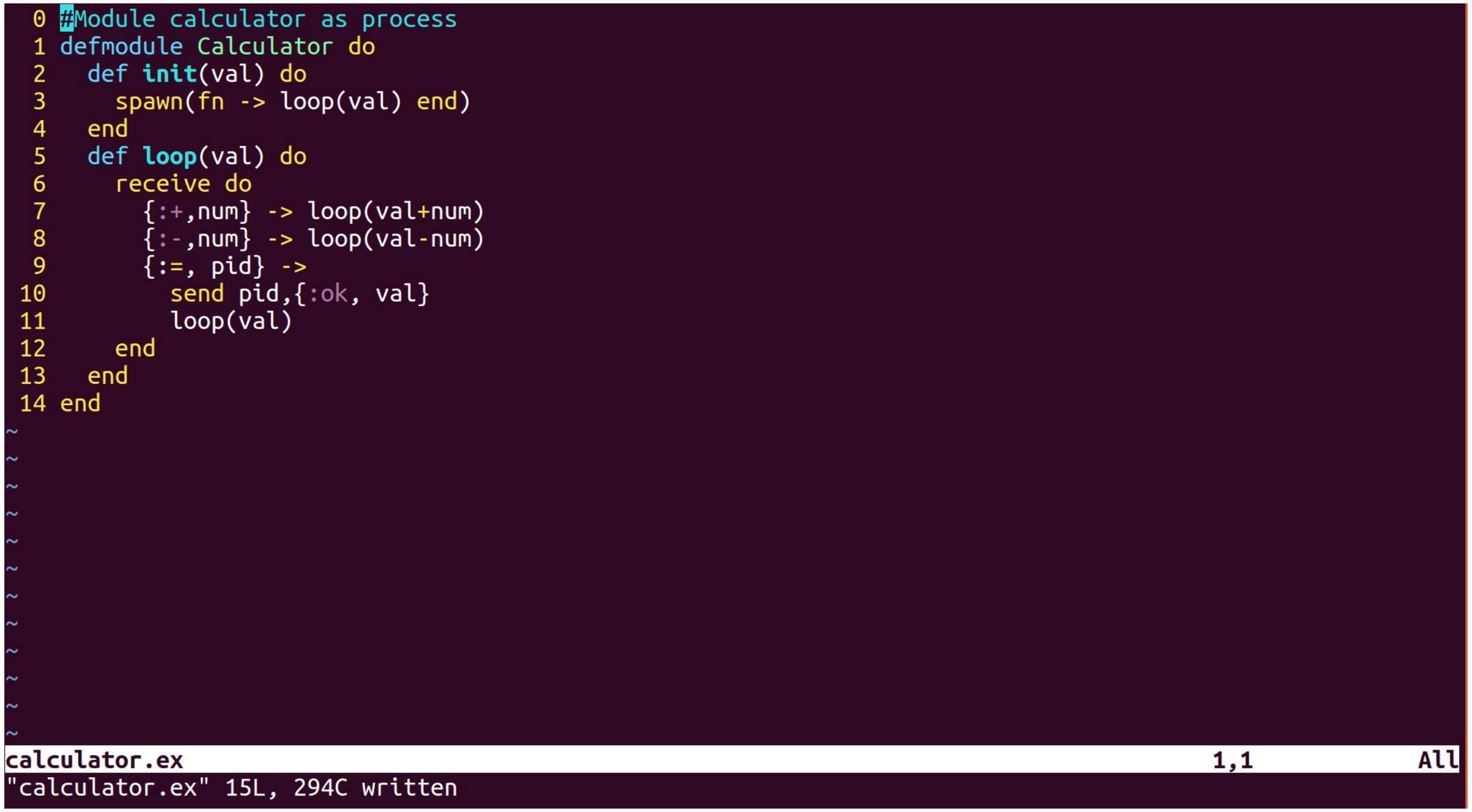


```
#Start the calculator with zero
val = 0;

#Define an anonymous function for add
sum = fn(val,add) ->
  val + add
end

#Define an anonymous function for sub
sub = fn(val,sub) ->
  val - sub
end

#IO.puts "Inital value #{val}"
#IO.puts "Add 1"
#sum.(val,1)
#IO.puts "After add 1: #{val} !wrong"
#val = sum.(val,1)
#IO.puts "After add 1: #{val}";
~
~
~
~
~
~
~
~
calculator_script.exs                                         1,1           All
```

Elixir – Script Calculator

- Issues
  - Doesn’t maintain state
  - No communicaton with other processes

Elixir – Module Calculator



```
#Module calculator as process
defmodule Calculator do
  def init(val) do
    spawn(fn -> loop(val) end)
  end
  def loop(val) do
    receive do
      {:+,num} -> loop(val+num)
      {:-,num} -> loop(val-num)
      {:=, pid} ->
        send pid,{:ok, val}
        loop(val)
    end
  end
end
~
~
~
~
~
~
~
~
~
~
~
~
calculator.ex                                                                1,1           All
"calculator.ex" 15L, 294C written
```

Elixir – Module Calculator

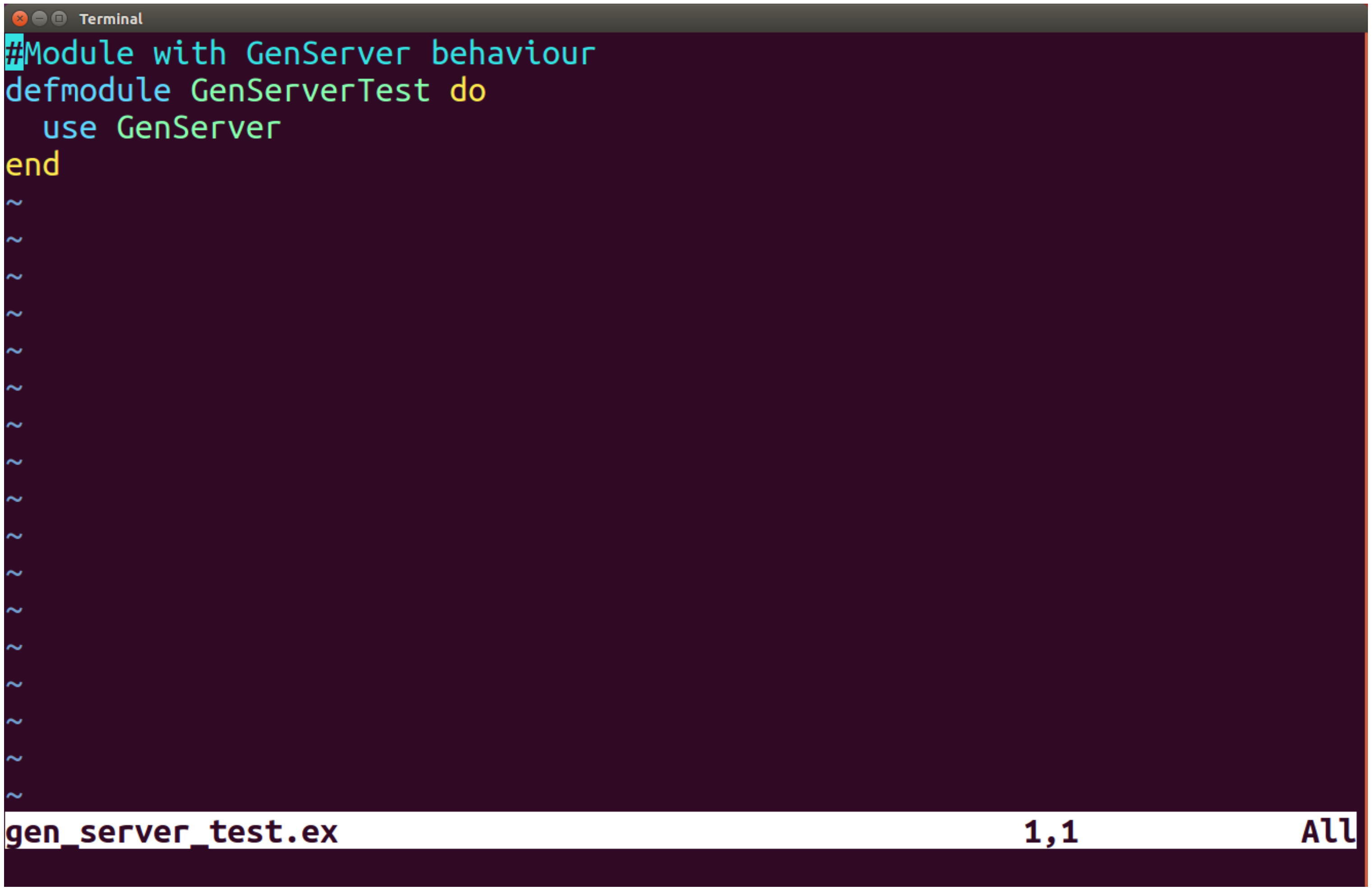
- Maintain state
- Communication with other processes
- What is missing?
  - Message Box
  - Naming
  - Distribution
  - Concurrency
  - Fault Tolerance/recovery

Elixir – GenServer

- Provides callback functions
- Manage inbox messages
- Alias registration
- Integration with OTP behaviours

Elixir – GenServer

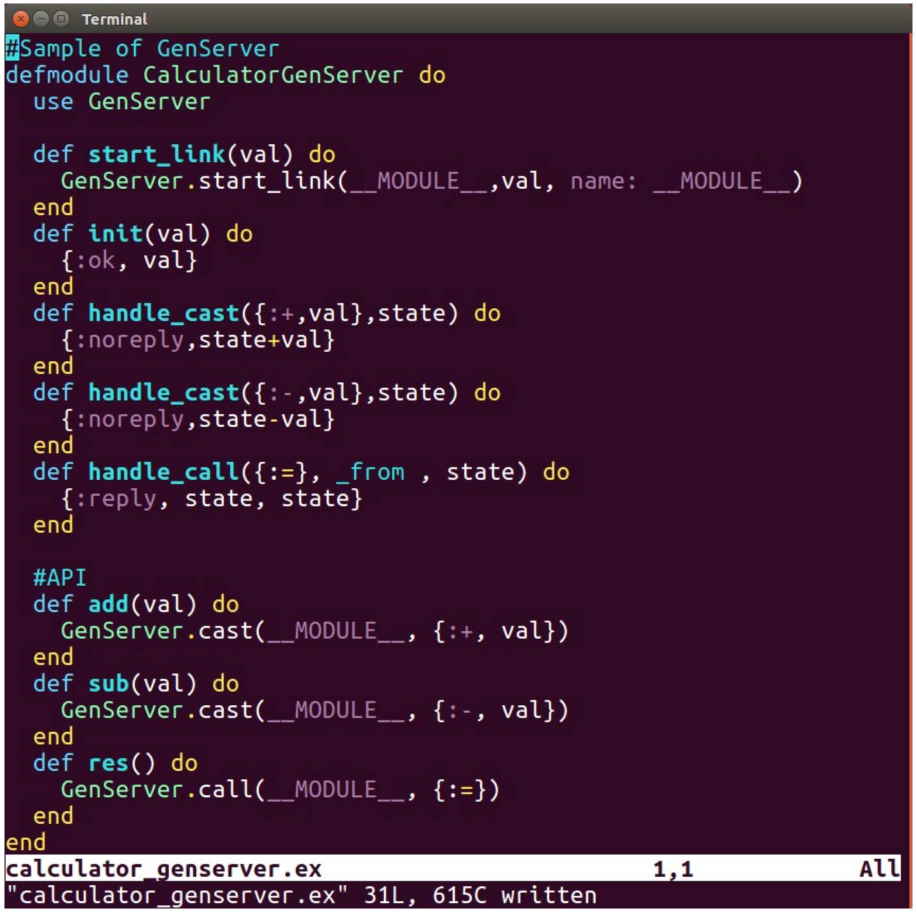
•



Terminal

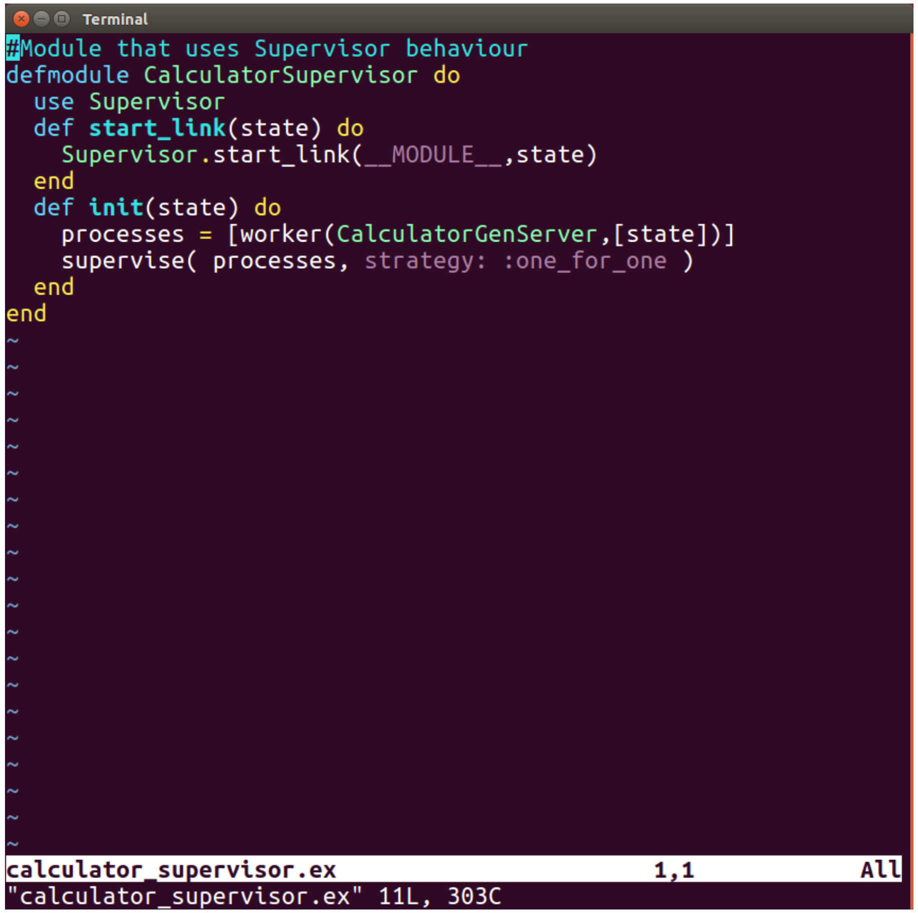
```
#Module with GenServer behaviour
defmodule GenServerTest do
  use GenServer
end
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
gen_server_test.ex                                    1,1           All
```

Elixir – GenServer Calculator


Terminal

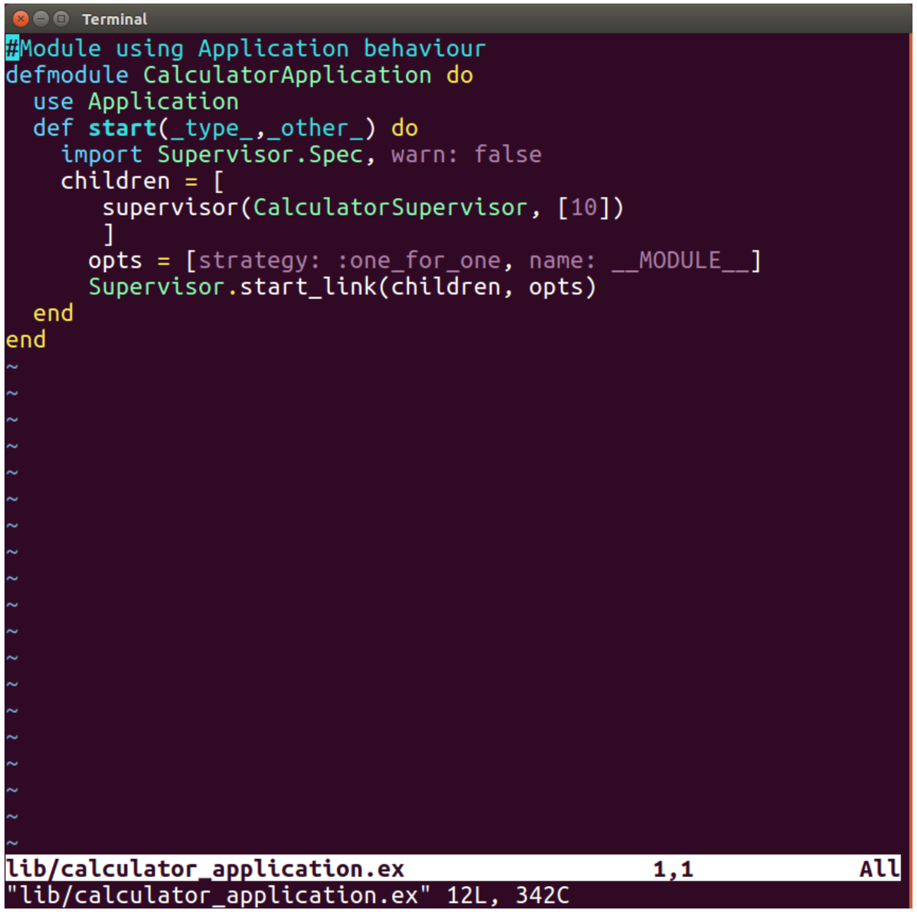
```
#Sample of GenServer
defmodule CalculatorGenServer do
  use GenServer

  def start_link(val) do
    GenServer.start_link(__MODULE__,val, name: __MODULE__)
  end
  def init(val) do
    {:ok, val}
  end
  def handle_cast({:+,val},state) do
    {:noreply,state+val}
  end
  def handle_cast({:-,val},state) do
    {:noreply,state-val}
  end
  def handle_call({:=}, _from , state) do
    {:reply, state, state}
  end

  #API
  def add(val) do
    GenServer.cast(__MODULE__, {:+, val})
  end
  def sub(val) do
    GenServer.cast(__MODULE__, {:-, val})
  end
  def res() do
    GenServer.call(__MODULE__, {:=})
  end
end
calculator_genserver.ex                                  1,1            All
"calculator_genserver.ex" 31L, 615C written
```

Elixir – Supervisor Calculator


Terminal

```
#Module that uses Supervisor behaviour
defmodule CalculatorSupervisor do
  use Supervisor
  def start_link(state) do
    Supervisor.start_link(__MODULE__,state)
  end
  def init(state) do
    processes = [worker(CalculatorGenServer,[state])]
    supervise( processes, strategy: :one_for_one )
  end
end
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
calculator_supervisor.ex                              1,1            All
"calculator_supervisor.ex" 11L, 303C
```

Elixir – Application Calculator



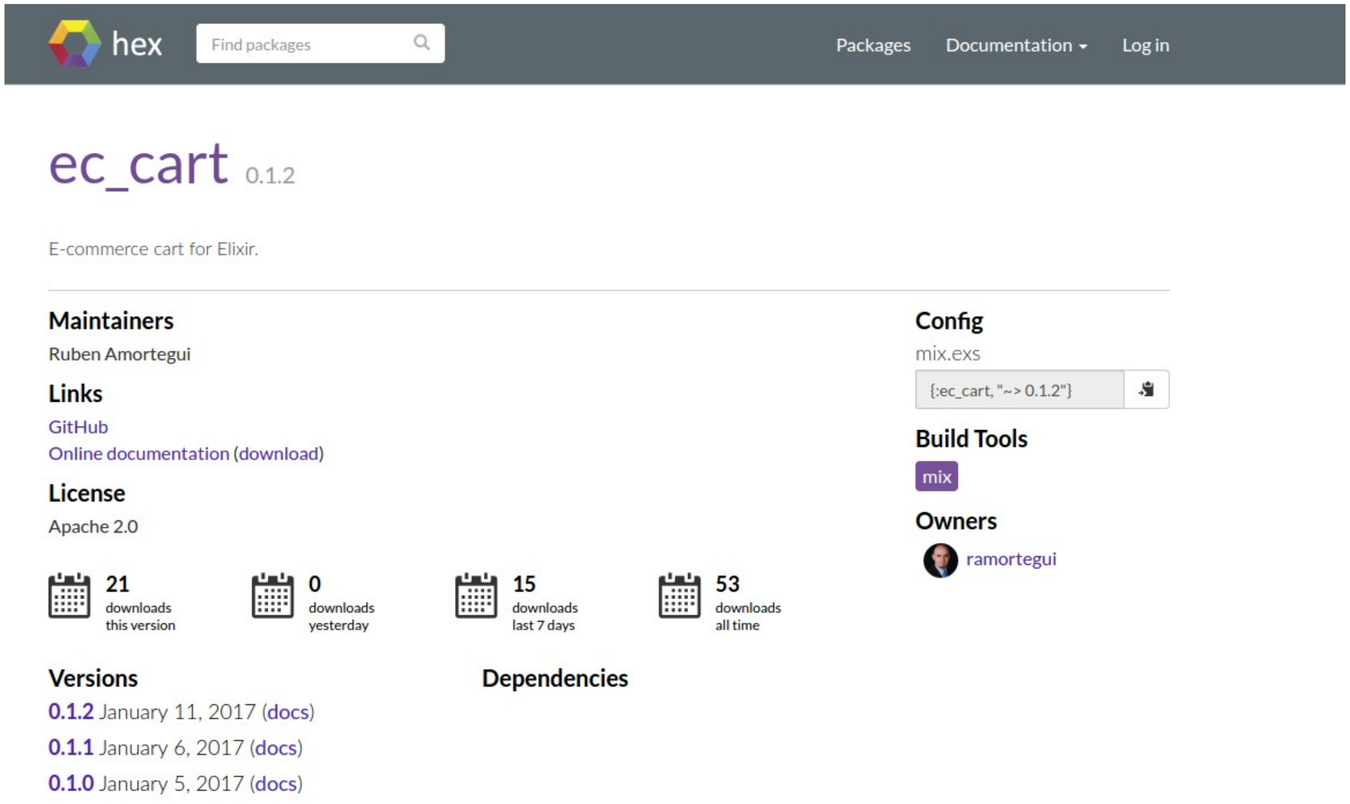
```
#Module using Application behaviour
defmodule CalculatorApplication do
  use Application
  def start(_type_,_other_) do
    import Supervisor.Spec, warn: false
    children = [
       supervisor(CalculatorSupervisor, [10])
       ]
      opts = [strategy: :one_for_one, name: __MODULE__]
      Supervisor.start_link(children, opts)
  end
end
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
lib/calculator_application.ex                       1,1            All
"lib/calculator_application.ex" 12L, 342C
```

Elixir – Summary

- OTP is easy to implement.
- OTP behaviours has been battle tested for years.
- The behaviours on OTP makes hard tasks really easy.

Elixir – EcCart

- https://hex.pm/packages/ec_cart



hex | Find packages | Packages | Documentation | Log in

ec_cart 0.1.2

E-commerce cart for Elixir.

Maintainers
Ruben Amortegui

Links
GitHub
Online documentation (download)

License
Apache 2.0

21 downloads this version
0 downloads yesterday
15 downloads last 7 days
53 downloads all time

Versions
0.1.2 January 11, 2017 (docs)
0.1.1 January 6, 2017 (docs)
0.1.0 January 5, 2017 (docs)

Dependencies

Config
mix.exs
```
{:ec_cart, "~> 0.1.2"}
```

Build Tools
mix

Owners
ramortegui

References

- Jurić, S. (2015). Elixir in action. Shelter Island, NY: Manning Publications.
- Thomas, D. (2016). Programming Elixir 1.3: functional, concurrent, pragmatic, fun. Releigh, NC: Pragmatic Bookshelf.
- Tan Wei Hao, B.(2017). The little Elixir & OTP Guidebook. Shelter Island, NY: Manning Publications.

Thanks!

Q & A?

@ramortegui

http://rubenamortegui.com

https://github.com/ramortegui