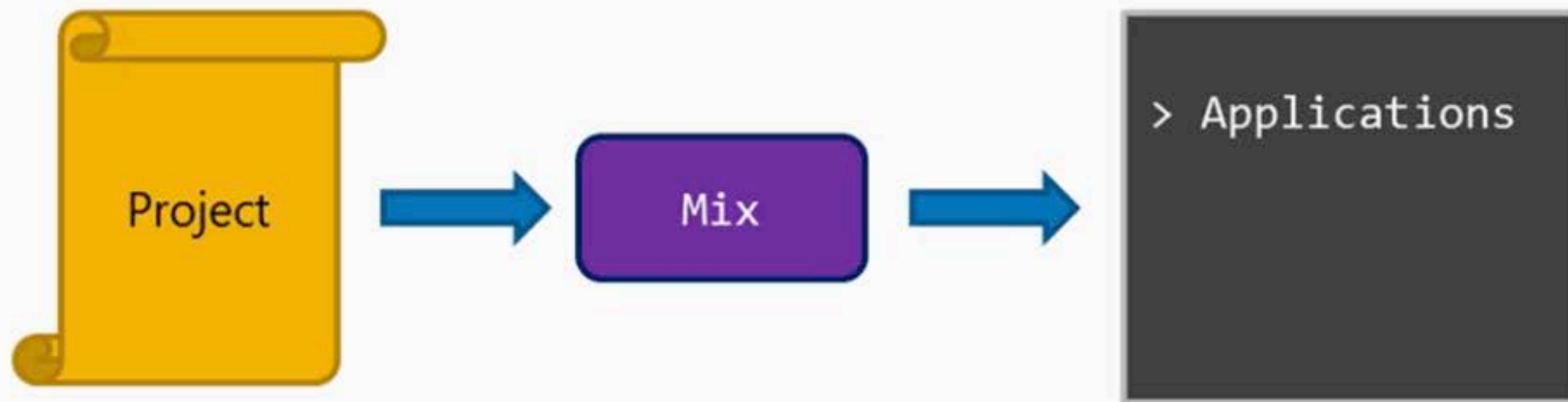


Releases

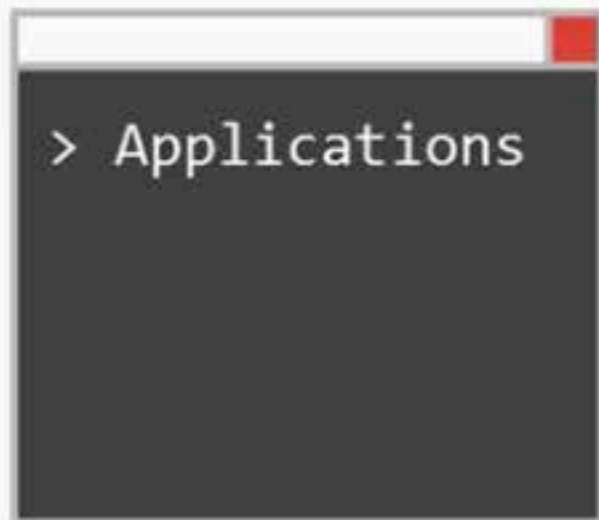
In this Video, we are going to take a look at...

- Mix environments
- Setting up distillery
- Releasing

Mix Environments

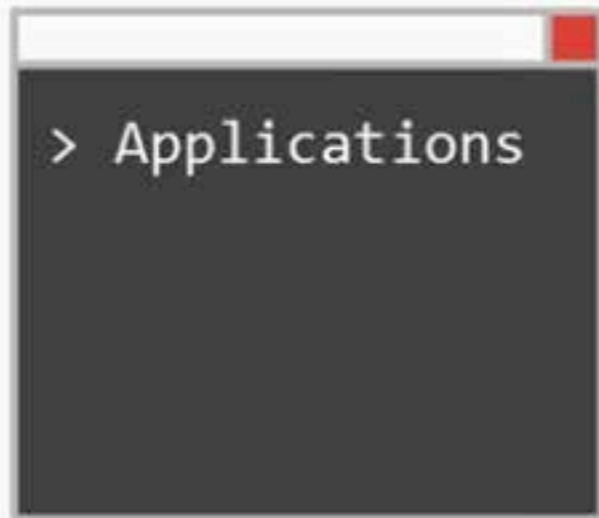


Mix Environments



Test

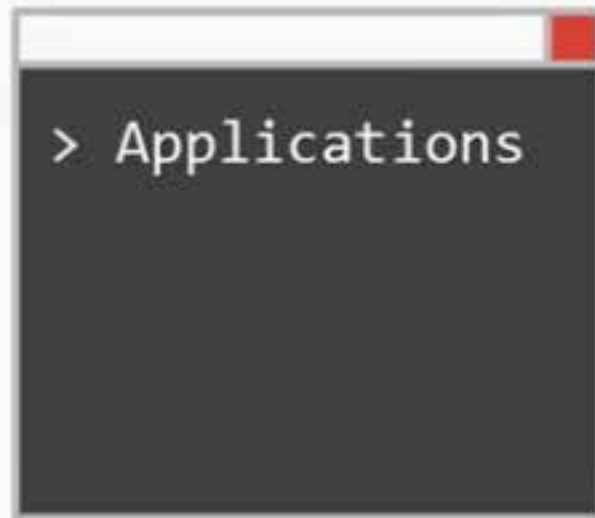
Mix Environments



Test

Development

Mix Environments

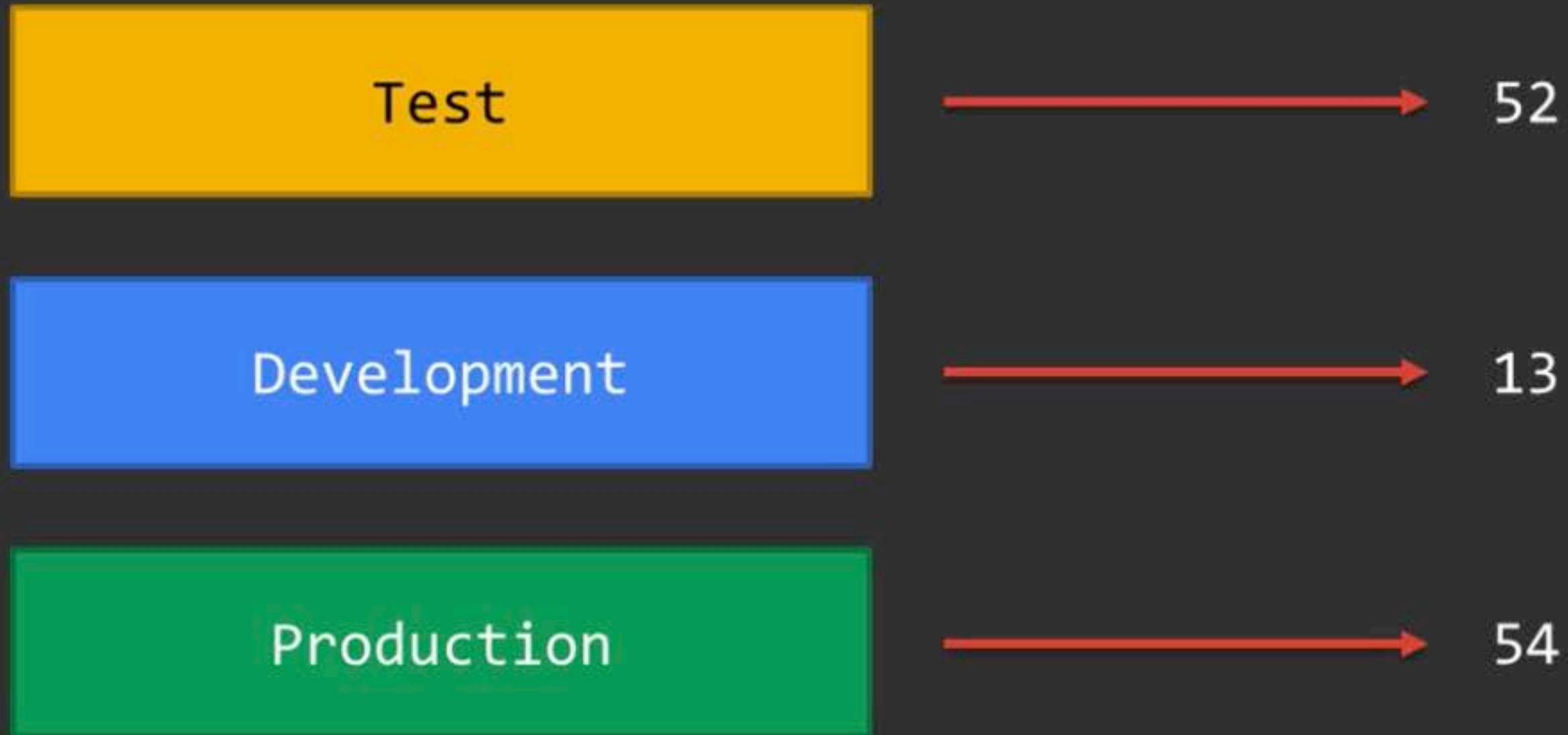


Test

Development

Production

Card Deck Size



Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

+ apps/

+ config/

+ deps/

+ rel/

.gitignore

README.md

mix.exs

mix.lock

[3/10] card_deck (D:6 F:4)



- 0

scratch

Text

utf-8 | 1: 0

All


```
defmodule Deck do
  use GenServer

  def start_link() do
    GenServer.start_link(__MODULE__, [], name: __MODULE__)
  end

  def take_card() do
    GenServer.call(__MODULE__, {:take_card})
  end

  def init(_) do
    {:ok, Enum.shuffle(1..52)}
  end

  def handle_call({:take_card}, _from, [card|deck]) do
    {:reply, card, deck}
  end
end
```

```
defmodule Deck do
  use GenServer

  @deck_size Application.get_env(:deck, :size)

  def start_link() do
    GenServer.start_link(__MODULE__, [], name: __MODULE__)
  end

  def take_card() do
    GenServer.call(__MODULE__, {:take_card})
  end

  def init(_) do
    {:ok, Enum.shuffle(1..52)}
  end

  def handle_call({:take_card}, _from, [card|deck]) do
    {:reply, card, deck}
  end
end
```

```

defmodule Deck do
  use GenServer

  @deck_size Application.get_env(:deck, :size)

  def start_link() do
    GenServer.start_link(__MODULE__, [], name: __MODULE__)
  end

  def take_card() do
    GenServer.call(__MODULE__, {:take_card})
  end

  def init(_) do
    {:ok, Enum.shuffle(1..@deck_size)}
  end

  def handle_call({:take_card}, _from, [card|deck]) do
    {:reply, card, deck}
  end
end

```

1 - 397 **deck.ex** Elixir alchemist (Y)(S)(P)(A)(K) Git:feature/environments Mod

Wrote /Users/jpoverclock/Development/Elixir/Projects/card_deck/apps/deck/lib/deck.ex

x


```
# and access this configuration in your application as:
#
#   Application.get_env(:deck, :key)
#
# You can also configure a 3rd-party app:
#
#   config :logger, level: :info
#
# It is also possible to import configuration files, relative to this
# directory. For example, you can emulate configuration per environment
# by uncommenting the line below and defining dev.exs, test.exs and such.
# Configuration from the imported file will override the ones defined
# here (which is why it is important to import them last).
#
import_config "#{Mix.env}.exs"
```

Press ? for neotree help

```
<ent/Elixir/Projects/card_deck/
+.git/
+_build/
-apps/
  +api/
  -deck/
    -config/
      config.exs
    +lib/
    +test/
      .gitignore
      README.md
```

[1/6] deck (D:3 F:3)

->prod.e

neotree-create-node (C-h: Go up one level)

[?] prod.e

```
# It is also possible to import configuration file\
s, relative to this
# directory. For example, you can emulate configur\
ation per environment
# by uncommenting the line below and defining dev.\
exs, test.exs and such.
# Configuration from the imported file will overri\
de the ones defined
# here (which is why it is important to import the\
m last).
#
import_config "#{Mix.env}.exs"
```

① - 1.1k config.exs Elixir unix | 30: 0 B

Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

- apps/

+ api/

- deck/

- config/

config.exs

dev.exs

prod.exs

test.exs

+ lib/

+ test/

.gitignore

README.md

mix.exs

+ config/

+ deps/

+ rel/

.gitignore

README.md

[3/4] config (F:4)



- 0

test.exs

Elixir

utf-8

| 1: 0

All

Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

- apps/

+ api/

- deck/

- config/

config.exs

dev.exs

prod.exs

test.exs

+ lib/

+ test/

.gitignore

README.md

mix.exs

+ config/

+ deps/

+ rel/

.gitignore

README.md

[2/4] config (F:4)

use Mix.Config

config :deck, size: 13

1

*

38

dev.exs

Elixir

alchemist

Y

S

P

A

K

Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

- apps/

+ api/

- deck/

- config/

config.exs

dev.exs

prod.exs

test.exs

+ lib/

+ test/

.gitignore

README.md

mix.exs

+ config/

+ deps/

+ rel/

.gitignore

README.md

[3/4] config (F:4)

use Mix.Config

config :deck, size: 52

1

* 38

prod.exs

Elixir

alchemist

Y

S

P

A

K

Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

- apps/

+ api/

- deck/

- config/

config.exs

dev.exs

prod.exs

test.exs

+ lib/

+ test/

.gitignore

README.md

mix.exs

+ config/

+ deps/

+ rel/

.gitignore

README.md

[5/5] config (F:5)

:w

use Mix.Config

config :deck, size: 54

1

* 38 test.exs

Elixir

alchemist

Y S P a K

```

defmodule Api.Router do
  use Plug.Router

  @suites ["Spades", "Clubs", "Diamonds", "Hearts"]
  @cards ["Ace", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "\
Ten", "Jack", "Queen", "King"]


  plug :match
  plug :dispatch

  get "/card" do
    card = Deck.take_card()
    |> convert()

    conn
    |> send_resp(200, card)
  end

  def convert(card) when card > 52 do
    "Joker"
  end
end

```

1 - 552 **router.ex** Elixir alchemist  Git:feature/environments Mod
 Wrote /Users/jpoverclock/Development/Elixir/Projects/card_deck/apps/api/lib/api/rou\
 ter.ex

→ **card_deck** **git:(feature/environments)** **X** iex -S mix

Erlang/OTP 20 [erts-9.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:10]
] [hipe] [kernel-poll:false] [dtrace]

Interactive Elixir (1.5.1) - press Ctrl+C to exit (type h() ENTER for help)

iex(1)> Deck.take_card()

6

iex(2)> Deck.take_card()

11

iex(3)> █

```
11
iex(3)> Deck.take_card()
4
iex(4)> Deck.take_card()
7
iex(5)> Deck.take_card()
8
iex(6)> Deck.take_card()
9
iex(7)> Deck.take_card()
12
iex(8)> Deck.take_card()
3
iex(9)> Deck.take_card()
5
iex(10)> Deck.take_card()
10
iex(11)> Deck.take_card()
13
iex(12)> Deck.take_card()
2
iex(13)> Deck.take_card()
1
iex(14)> █
```



```
(stdlib) gen_server.erl:665: :gen_server.handle_msg/6
(stdlib) proc_lib.erl:247: :proc_lib.init_p_do_apply/3
(elixir) lib/gen_server.ex:774: GenServer.call/3
iex(14)>
14:20:37.334 [error] GenServer Deck terminating
** (FunctionClauseError) no function clause matching in Deck.handle_call/3
    (deck) lib/deck.ex:18: Deck.handle_call({:take_card}, {#PID<0.309.0>, #Reference
<0.2633520126.1562902531.12900>}, [])
    (stdlib) gen_server.erl:636: :gen_server.try_handle_call/4
    (stdlib) gen_server.erl:665: :gen_server.handle_msg/6
    (stdlib) proc_lib.erl:247: :proc_lib.init_p_do_apply/3
Last message (from #PID<0.309.0>): {:take_card}
State: []
Client #PID<0.309.0> is alive
    (stdlib) gen.erl:169: :gen.do_call/4
    (elixir) lib/gen_server.ex:771: GenServer.call/3
    (stdlib) erl_eval.erl:670: :erl_eval.do_apply/6
    (elixir) src/elixir.erl:239: :elixir.eval_forms/4
    (iex) lib/iex/evaluator.ex:219: IEx.Evaluator.handle_eval/5
    (iex) lib/iex/evaluator.ex:200: IEx.Evaluator.do_eval/3
    (iex) lib/iex/evaluator.ex:178: IEx.Evaluator.eval/3
    (iex) lib/iex/evaluator.ex:77: IEx.Evaluator.loop/1
```



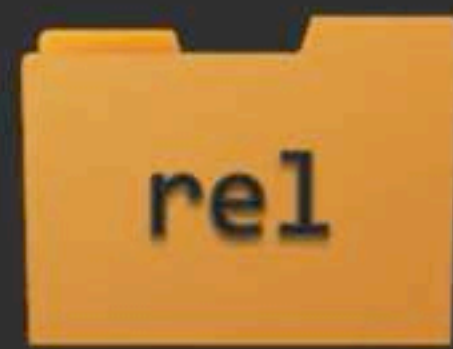
```
iex -S mix
```

Distillery

A tool that allows release builds to be generated



```
mix release --init
```

config.exs

```
MIX_ENV=prod mix release
```

Press ? for neotree help

<ent/Elixir/Projects/card_deck/

+ .git/

+ _build/

+ apps/

+ config/

+ deps/

.gitignore

README.md

mix.exs

mix.lock

```
# This file is responsible for configuring your ap\
plication
```

```
# and its dependencies with the aid of the Mix.Con\
fig module.
```

```
use Mix.Config
```

```
# By default, the umbrella project as well as each\
child
```

```
# application will require this configuration file\
, ensuring
```

```
# they all use the same configuration. While one c\
ould
```

```
# configure all applications here, we prefer to de\
legate
```

```
# back to each application for organization purpos\
es.
```

```
import_config "../apps/*/config/config.exs"
```

```
# Sample configuration (overrides the imported con\
figuration above):
```

```
#
```

```
# config :logger, :console,
```

```
defmodule WebDeck.Mixfile do
  use Mix.Project

  def project do
    [
      apps_path: "apps",
      start_permanent: Mix.env == :prod,
      deps: deps()
    ]
  end

  # Dependencies listed here are available only for this
  # project and cannot be accessed from applications inside
  # the apps folder.
  #
  # Run "mix help deps" for examples and options.
  defp deps do
    [
      {:distillery, "~> 1.5", runtime: false}
    ]
  end
end
```

```
→ card_deck git:(feature/release) ✗ mix deps.get
```



```
→ card_deck git:(feature/release) X mix deps.get
```

```
Running dependency resolution...
```

```
Dependency resolution completed:
```

```
cowboy 1.1.2
```

```
cowlib 1.0.2
```

```
distillery 1.5.2
```

```
mime 1.1.0
```

```
plug 1.4.3
```

```
ranch 1.3.2
```

```
* Getting distillery (Hex package)
```

```
Checking package (https://repo.hex.pm/tarballs/distillery-1.5.2.tar)
```

```
Using locally cached package
```

```
→ card_deck git:(feature/release) X █
```

```
→ card_deck git:(feature/release) X mix deps.get
```

```
Running dependency resolution...
```

```
Dependency resolution completed:
```

```
cowboy 1.1.2
```

```
cowlib 1.0.2
```

```
distillery 1.5.2
```

```
mime 1.1.0
```

```
plug 1.4.3
```

```
ranch 1.3.2
```

```
* Getting distillery (Hex package)
```

```
Checking package (https://repo.hex.pm/tarballs/distillery-1.5.2.tar)
```

```
Using locally cached package
```

```
→ card_deck git:(feature/release) X mix release.init
```

```
==> distillery
```

```
Compiling 19 files (.ex)
```

```
Generated distillery app
```

An example config file has been placed in `rel/config.exs`, review it,
make edits as needed/desired, and then run ``mix release`` to build the release

```
→ card_deck git:(feature/release) X █
```

```
→ card_deck git:(feature/release) X mix deps.get
```

```
Running dependency resolution...
```

```
Dependency resolution completed:
```

```
cowboy 1.1.2
```

```
cowlib 1.0.2
```

```
distillery 1.5.2
```

```
mime 1.1.0
```

```
plug 1.4.3
```

```
ranch 1.3.2
```

```
* Getting distillery (Hex package)
```

```
Checking package (https://repo.hex.pm/tarballs/distillery-1.5.2.tar)
```

```
Using locally cached package
```

```
→ card_deck git:(feature/release) X mix release.init
```

```
==> distillery
```

```
Compiling 19 files (.ex)
```

```
Generated distillery app
```

An example config file has been placed in rel/config.exs, review it,
make edits as needed/desired, and then run `mix release` to build the release

```
→ card_deck git:(feature/release) X emacs -nw
```



```
# Import all plugins from `rel/plugins`
# They can then be used by adding `plugin MyPlugin` to
# either an environment, or release definition, where
# `MyPlugin` is the name of the plugin module.
Path.join(["rel", "plugins", "*.exs"])
|> Path.wildcard()
|> Enum.map(&Code.eval_file(&1))

use Mix.Releases.Config,
  # This sets the default release built by `mix release`
  default_release: :default,
  # This sets the default environment used by `mix release`
  default_environment: Mix.env()

# For a full list of config options for both releases
# and environments, visit https://hexdocs.pm/distillery/configuration.html

# You may define one or more environments in this file,
# an environment's settings will override those of a release
# when building in that environment, this combination of release
# and environment configuration is called a profile
```

```
environment :prod do
  set include_erts: true
  set include_src: false
  set cookie: : "v&XjHA8s_Y/|<pRw(EDFP}1Nv4gJt/Ten$aC^s%7CfV{w~X@~X_C*}aN(:7ouu@2"
end

# You may define one or more releases in this file.
# If you have not set a default release, or selected one
# when running `mix release`, the first release in the file
# will be used by default

release :card_deck do
  set version: "0.1.0"
  set applications: [
    :runtime_tools,
    api: :permanent,
    deck: :permanent
  ]
end
```


→ **card_deck git:(feature/release) X** MIX_ENV=prod mix release

==> Compiling ranch

==> Failed to restore /Users/jpoverclock/Development/Elixir/Projects/card_deck/deps/ranch/.rebar3/erlinfo file. Discarding it.

==> Compiling cowlib

==> Failed to restore /Users/jpoverclock/Development/Elixir/Projects/card_deck/deps/cowlib/.rebar3/erlinfo file. Discarding it.

src/cow_multipart.erl:392: Warning: crypto:rand_bytes/1 is deprecated and will be removed in a future release; use crypto:strong_rand_bytes/1

█

```
Generated mime app
==> plug
Compiling 1 file (.erl)
Compiling 44 files (.ex)
Generated plug app
==> distillery
Compiling 19 files (.ex)
Generated distillery app
==> deck
Compiling 3 files (.ex)
Generated deck app
==> api
Compiling 3 files (.ex)
Generated api app
==> Assembling release..
==> Building release card_deck:0.1.0 using environment prod
==> Including ERTS 9.0.4 from /usr/local/Cellar/erlang/20.0.4/lib/erlang/erts-9.0.4
==> Packaging release..
==> Release successfully built!
```

You can run it in one of the following ways:

Interactive: `_build/prod/rel/card_deck/bin/card_deck console`

Foreground: `_build/prod/rel/card_deck/bin/card_deck foreground`

Daemon: `_build/prod/rel/card_deck/bin/card_deck start`

→ `card_deck git:(feature/release) X`

→ **card_deck git:(feature/release) X** curl http://localhost:4000/card

Six of Clubs%

→ **card_deck git:(feature/release) X** curl http://localhost:4000/card

Eight of Spades%

→ **card_deck git:(feature/release) X** curl http://localhost:4000/card

Summary

- Explored agents and tasks
- Discussed umbrella applications
- Explained releases

Credits

Author

João Gonçalves

Reviewer

Paulo Pereira

Video Acquisition Editor

Mandar Raorane

Content Development Editor

Dwayne D'Souza

Video Editor

Rupali Parab

Video Quality Editor

Pranali Gore

Distribution Executive

Chaitali Naikwadi

Project Co-ordinator

Vedha Ramamoorthy