Balsamiq Mockups 3 Project Edit View Window Help Wed 2:46 PM

01 - Cards Outline - elixir - [.../Google Drive/Mocks/elixir.bmpr]

elixir

Quick Add

Cards

create_deck	Create an array of playing cards
shuffle	Shuffle an array of playing cards
deal	Create a 'hand' of cards
contains?	Given a deck and a single card, figure out if the card is in the deck
save	Save a collection of cards to a file on the local machine
load	Load a collection of cards from the local machine

```
→  prod elixir
Usage: elixir [options] [.exs file] [data]

  -v                              Prints version and exits
  -e COMMAND                      Evaluates the given command (*)
  -r FILE                         Requires the given files/patterns (*
)
  -S SCRIPT                       Finds and executes the given script
in PATH
  -pr FILE                        Requires the given files/patterns in
 parallel (*)
  -pa PATH                        Prepends the given path to Erlang co
de path (*)
  -pz PATH                        Appends the given path to Erlang cod
e path (*)
```

Balsamiq Mockups 3 Project Edit View Window Help Wed 2:46 PM

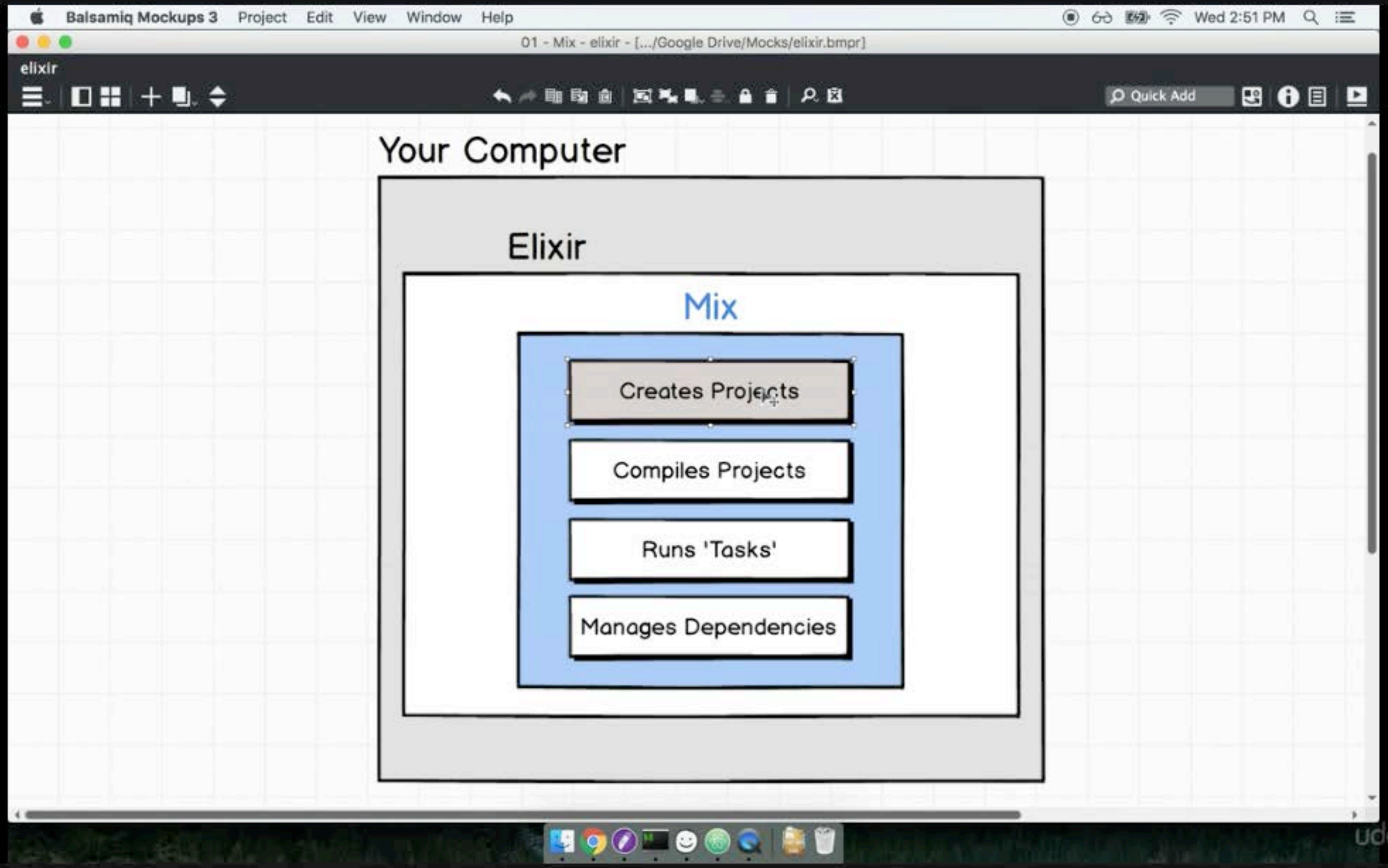
01 - Cards Outline - elixir - [.../Google Drive/Mocks/elixir.bmpr]

elixir

Quick Add

Cards

create_deck	Create an array of playing cards
shuffle	Shuffle an array of playing cards
deal	Create a 'hand' of cards
contains?	Given a deck and a single card, figure out if the card is in the deck
save	Save a collection of cards to a file on the local machine
load	Load a collection of cards from the local machine


Balsamiq Mockups 3
Project
Edit
View
Window
Help
Wed 2:51 PM
01 - Mix - elixir - [.../Google Drive/Mocks/elixir.bmpr]
elixir
Quick Add
Your Computer
Elixir
Mix
Creates Projects
Compiles Projects
Runs 'Tasks'
Manages Dependencies

iTerm Shell Edit View Profiles Toolbelt Window Help Wed 2:51 PM

1. stephengrider@stephens-MacBook-Pro: ~/workspace/ElixirWorkspace/prod (zsh)

```
→  prod mix new cards
```

```
* creating config/config.exs
* creating lib
* creating lib/cards.ex
* creating test
* creating test/test_helper.exs
* creating test/cards_test.exs

Your Mix project was created successfully.
You can use "mix" to compile it, test it, and more:

    cd cards
    mix test

Run "mix help" for more commands.

➜  prod
```

```
* creating lib
* creating lib/cards.ex
* creating test
* creating test/test_helper.exs
* creating test/cards_test.exs

Your Mix project was created successfully.
You can use "mix" to compile it, test it, and more:

    cd cards
    mix test

Run "mix help" for more commands.

→  prod cd cards
→  cards
```

```
* creating test
* creating test/test_helper.exs
* creating test/cards_test.exs

Your Mix project was created successfully.
You can use "mix" to compile it, test it, and more:

    cd cards
    mix test

Run "mix help" for more commands.

➜  prod cd cards
➜  cards ls
README.md config    lib       mix.exs   test
➜  cards
```

```
* creating test/test_helper.exs
* creating test/cards_test.exs

Your Mix project was created successfully.
You can use "mix" to compile it, test it, and more:

    cd cards
    mix test

Run "mix help" for more commands.

→  prod cd cards
→  cards ls
README.md config    lib       mix.exs   test
→  cards atom .
→  cards
```

Atom File Edit View Selection Find Packages Window Help

Wed 2:53 PM

cards.ex — /Users/stephengrider/workspace/ElixirWorkspace/prod/cards

cards

config

lib

cards.ex

test

.gitignore

mix.exs

README.md

cards.ex

```
1  defmodule Cards do
2  end
3
```

File 0 Project 0 ✔ No issues lib/cards.ex 3:1 LF UTF-8 Elixir 6 updates

Atom File Edit View Selection Find Packages Window Help

Wed 2:55 PM

cards.ex — /Users/stephengrider/workspace/ElixirWorkspace/prod/cards

- cards
  - config
  - lib
    - cards.ex
  - test
  - .gitignore
  - mix.exs
  - README.md

cards.ex

```
1  defmodule Cards do
2
3    def hello do
4      "hi there!"
5    end
6  end
7
```

File 0 | Project 0 | ✔ No issues | lib/cards.ex* 4:3 | LF UTF-8 Elixir | 6 updates

```
→  cards iex -S mix
Erlang/OTP 18 [erts-7.3] [source] [64-bit] [smp:4:4] [async-threads:10] [hipe] [kernel-poll:false] [dtrace]

Compiling 1 file (.ex)
Generated cards app
Interactive Elixir (1.3.1) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)>
```

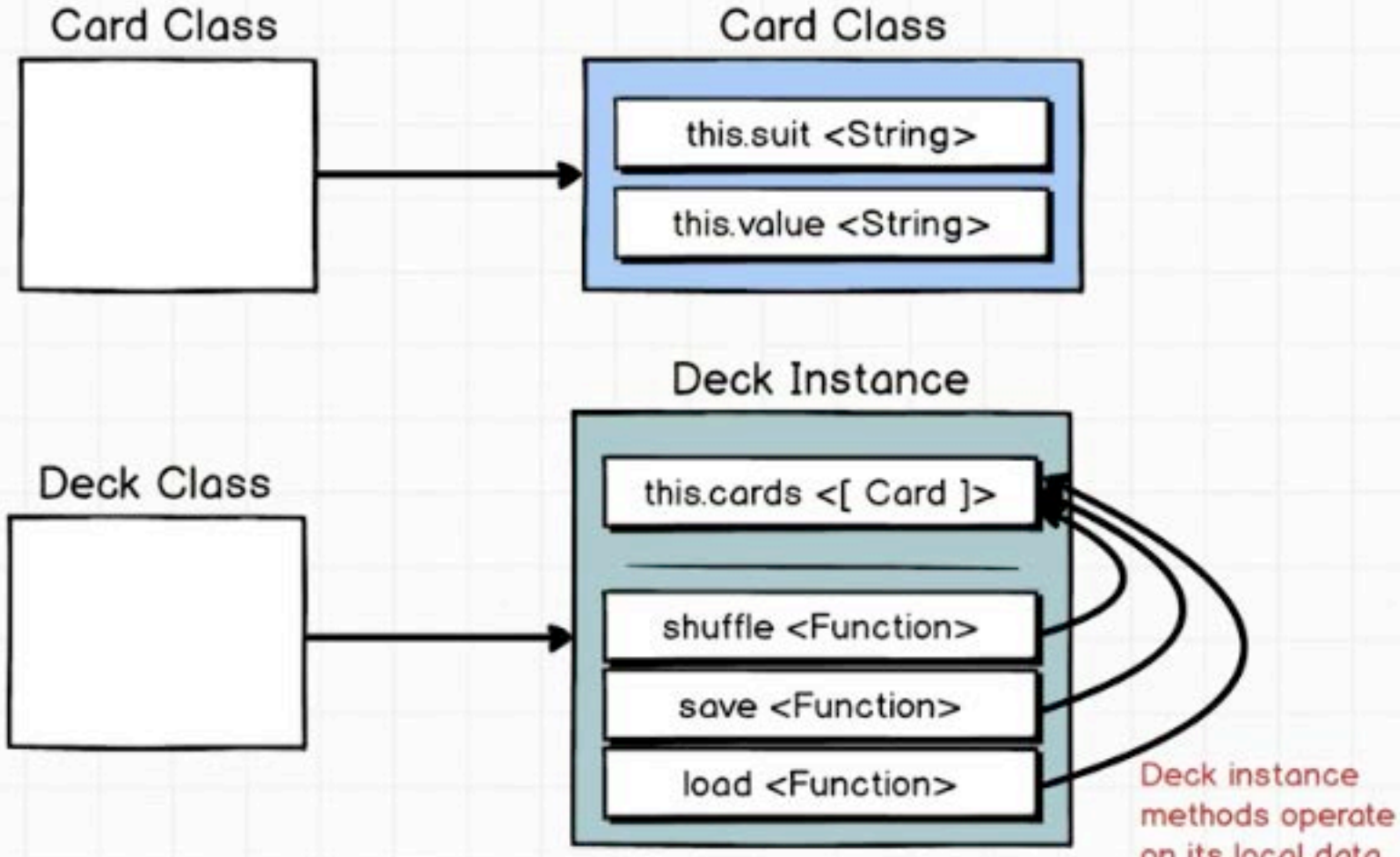
iTerm Shell Edit View Profiles Toolbelt Window Help Wed 2:57 PM

1. iex -S mix (beam.smp)

```
→  cards iex -S mix
Erlang/OTP 18 [erts-7.3] [source] [64-bit] [smp:4:4] [async-thread
s:10] [hipe] [kernel-poll:false] [dtrace]

Compiling 1 file (.ex)
Generated cards app
Interactive Elixir (1.3.1) - press Ctrl+C to exit (type h() ENTER
for help)
iex(1)> 1 + 1
2
iex(2)> Cards
Cards
iex(3)> Cards.hello
"hi there!"
iex(4)>
```

File 0 Project 0 ✓ No Issues lib/cards.ex 6:1 LF UTF-8 Elixir 6 updates

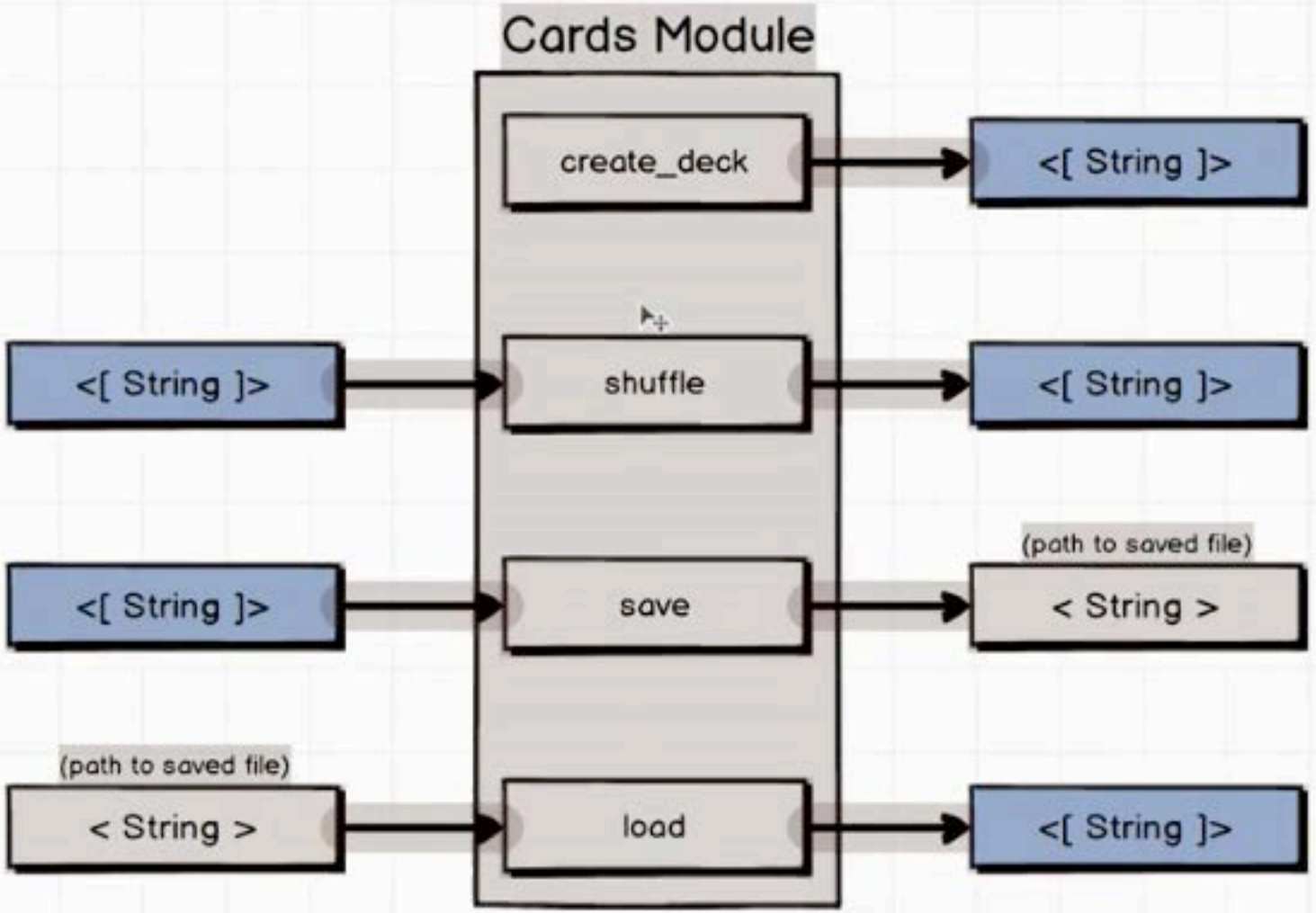
Cards - OO Approach

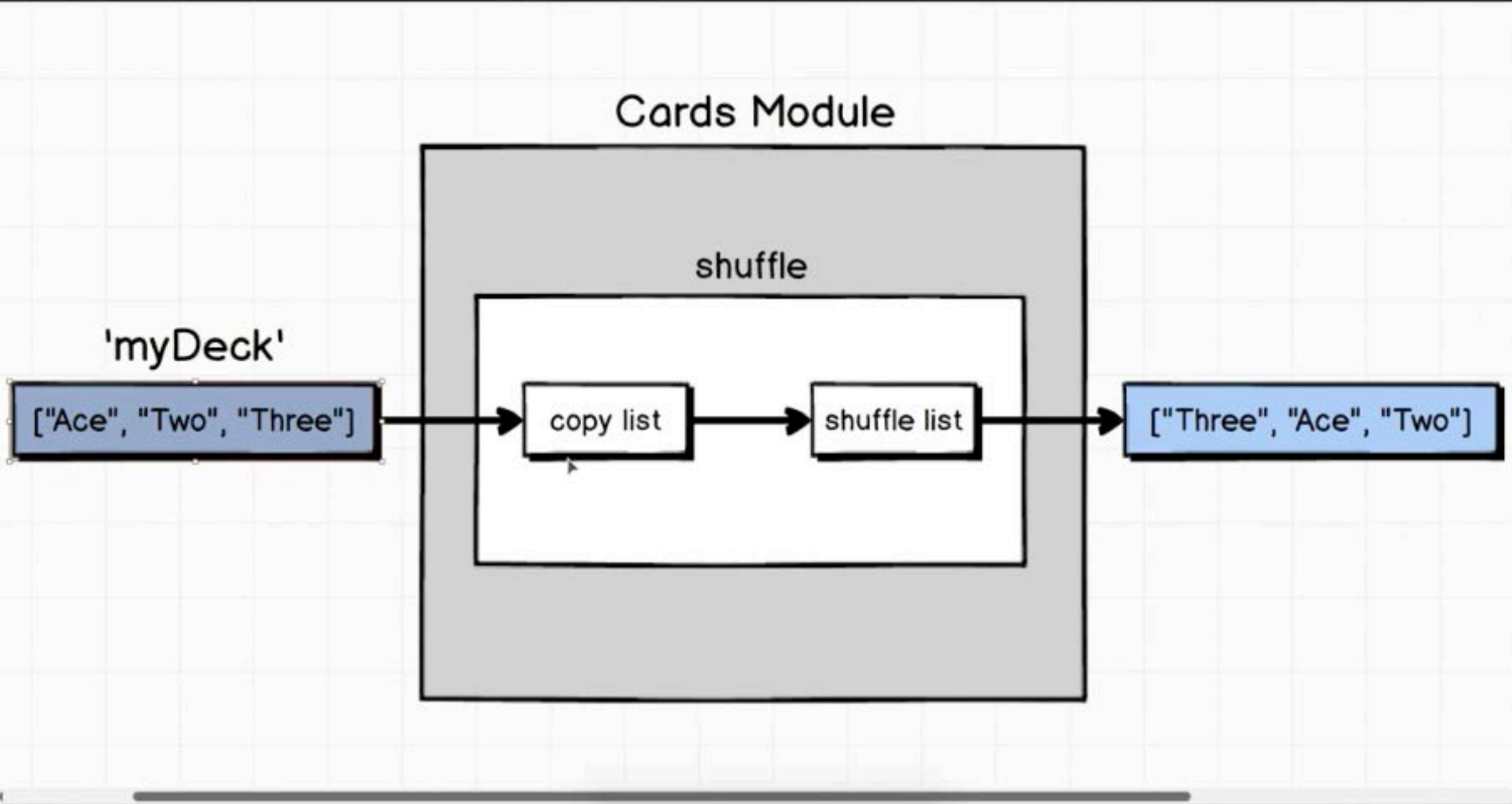


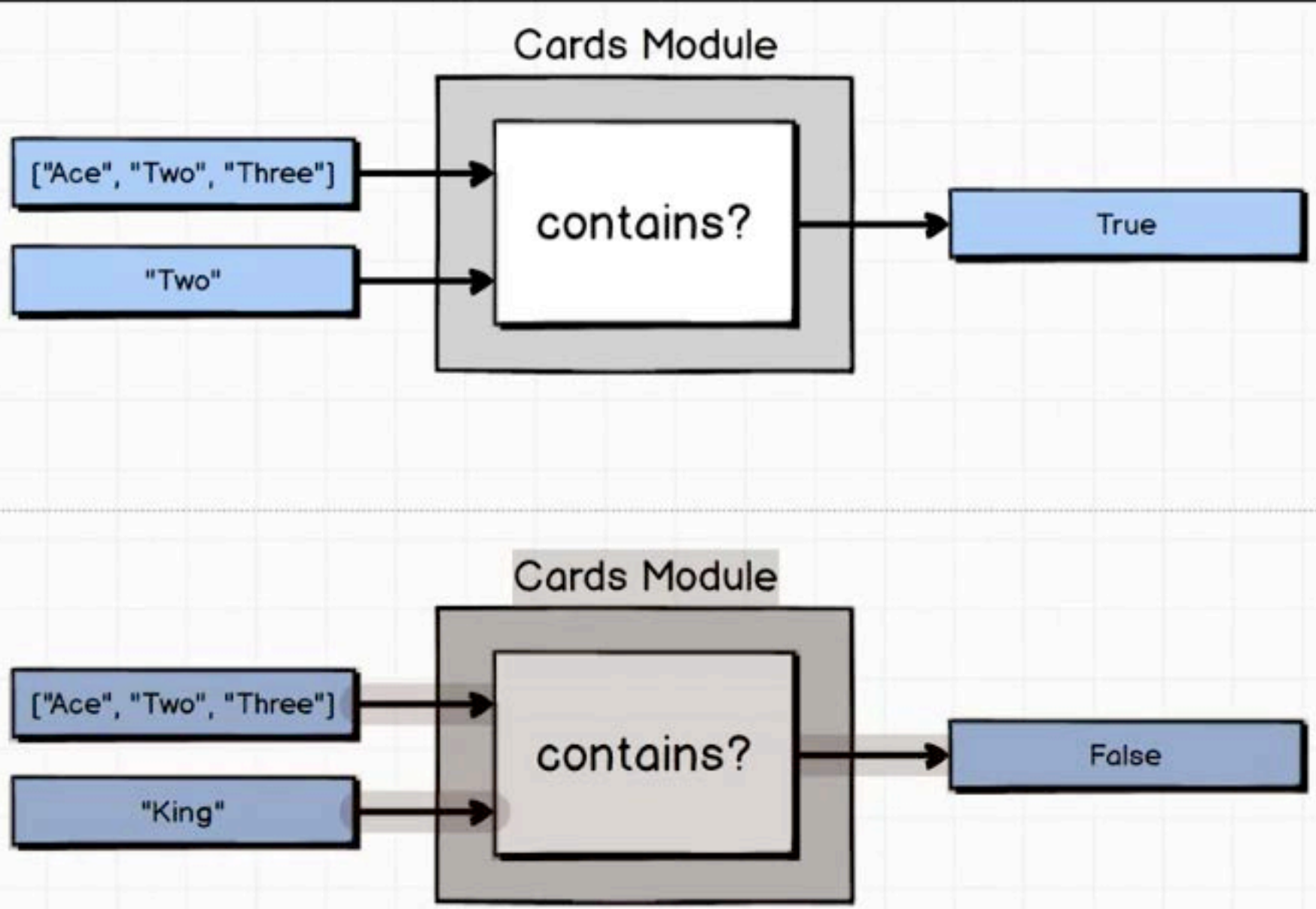
Cards - OO Approach

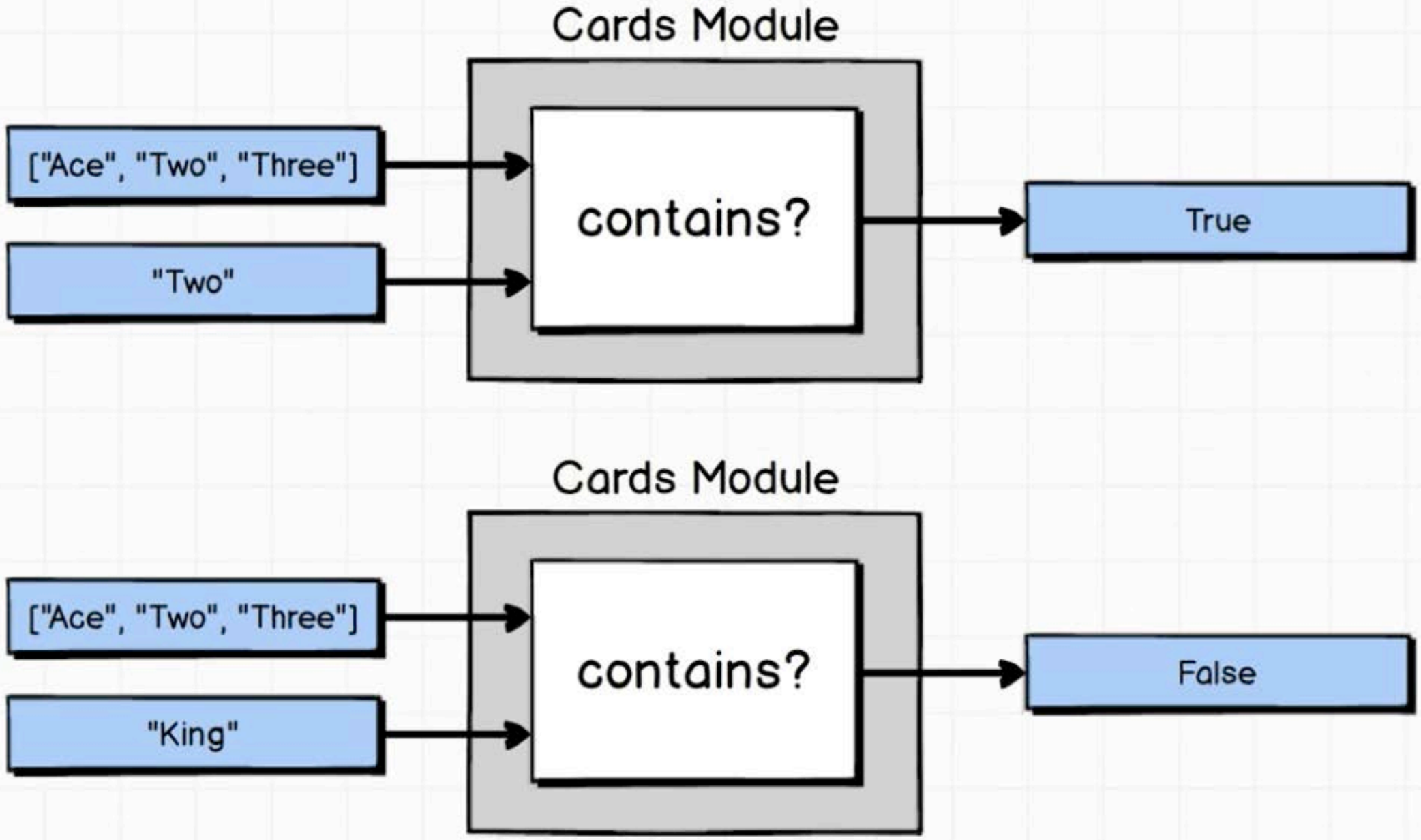
```
deck = new Deck;    <Deck>

deck.cards          [ <Card1>, <Card2>, <Card3>]

deck.shuffle        [ <Card2>, <Card1>, <Card3>]

deck.deal           <Deck[ <Card1> ] >
```

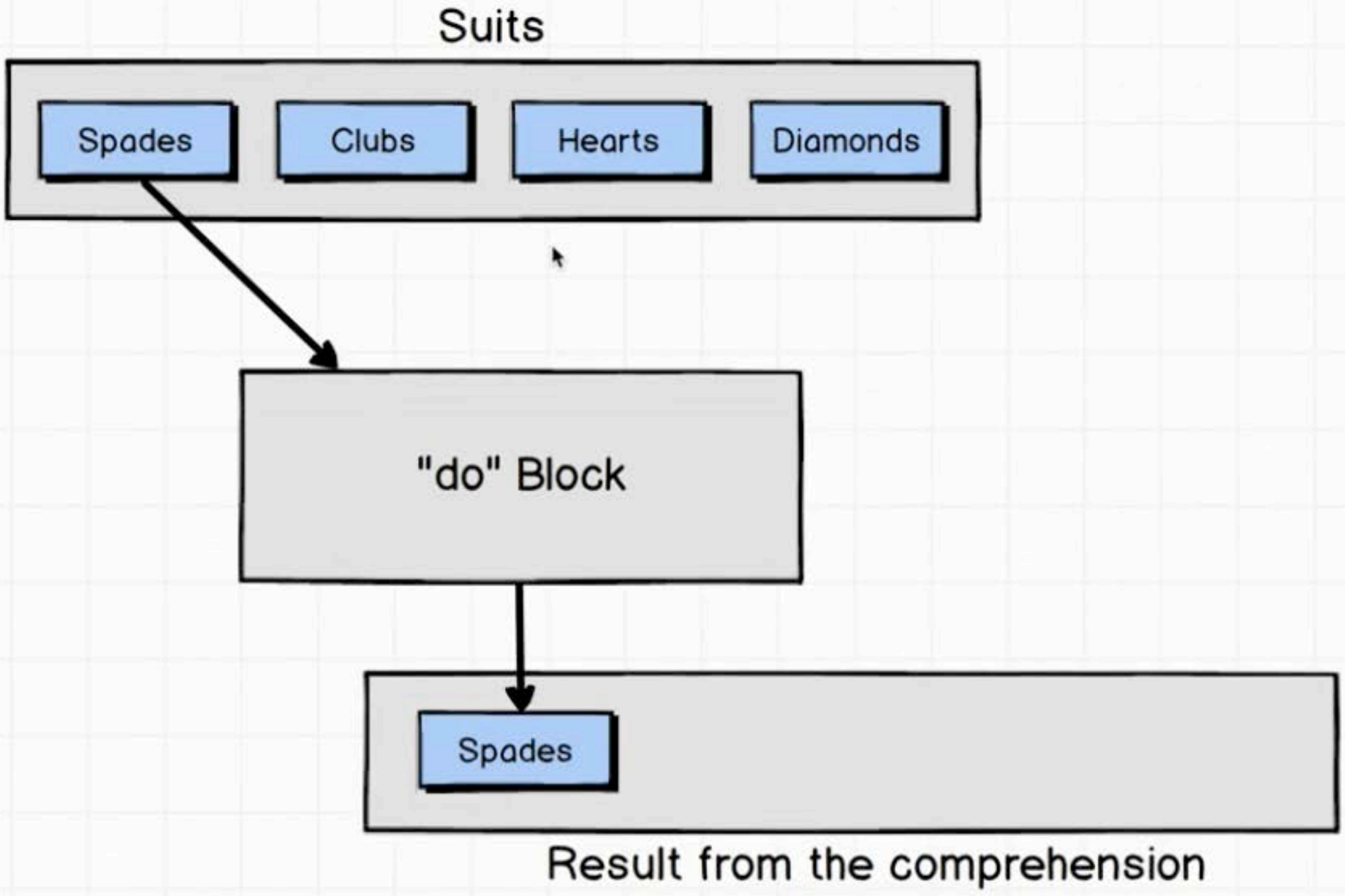
Cards - FP Approach

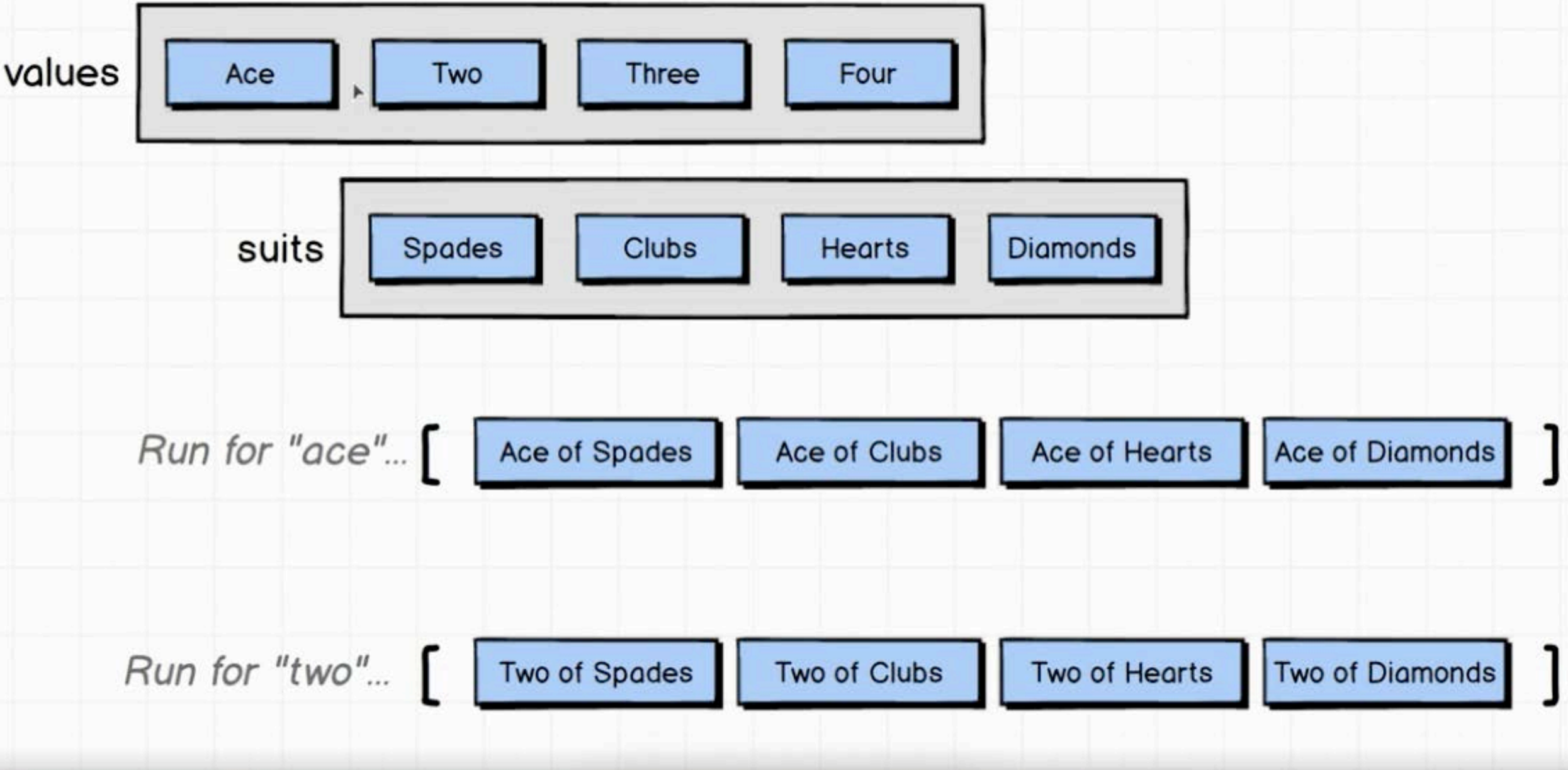


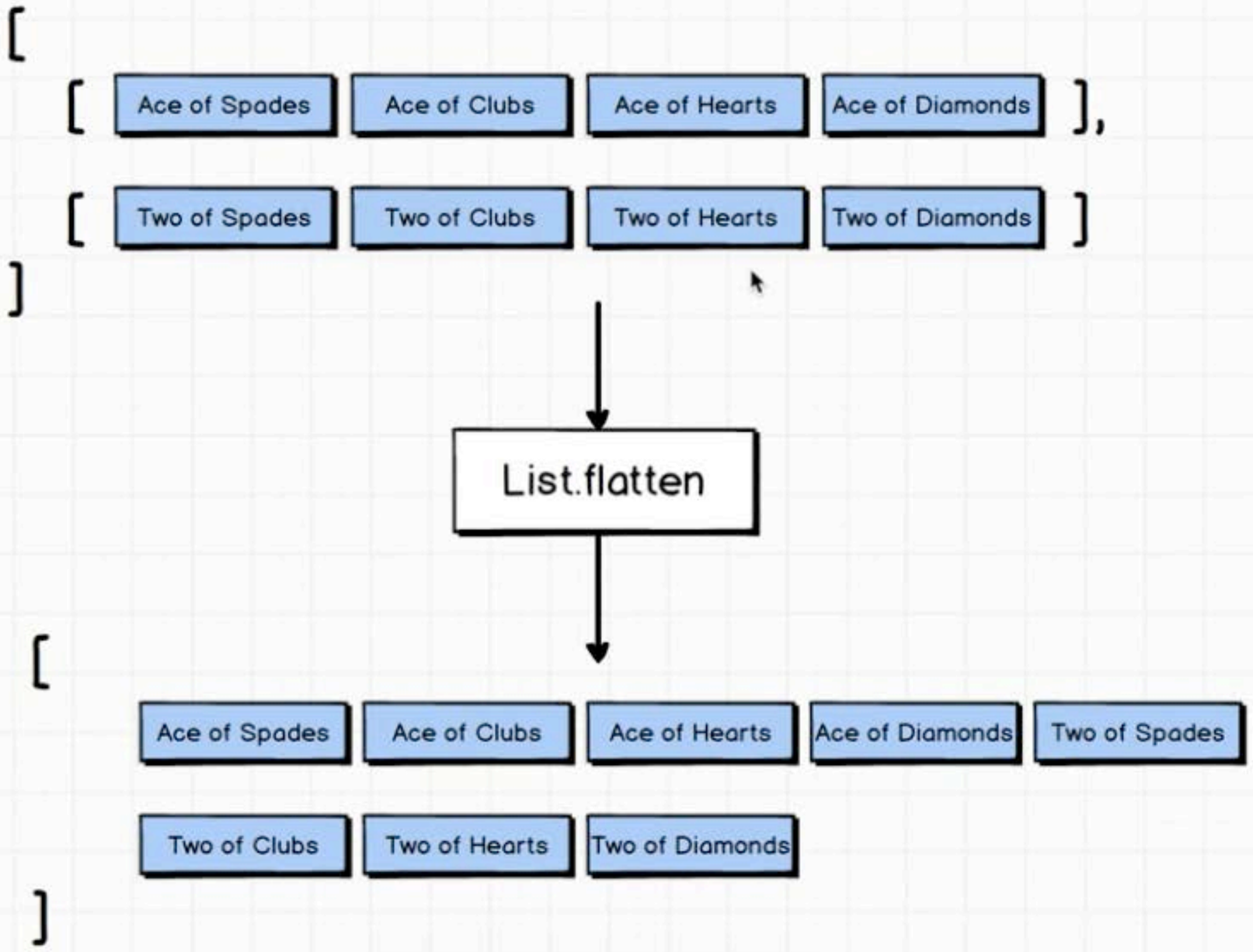

Cards Module
shuffle
'myDeck'
["Ace", "Two", "Three"]
copy list
shuffle list
["Three", "Ace", "Two"]

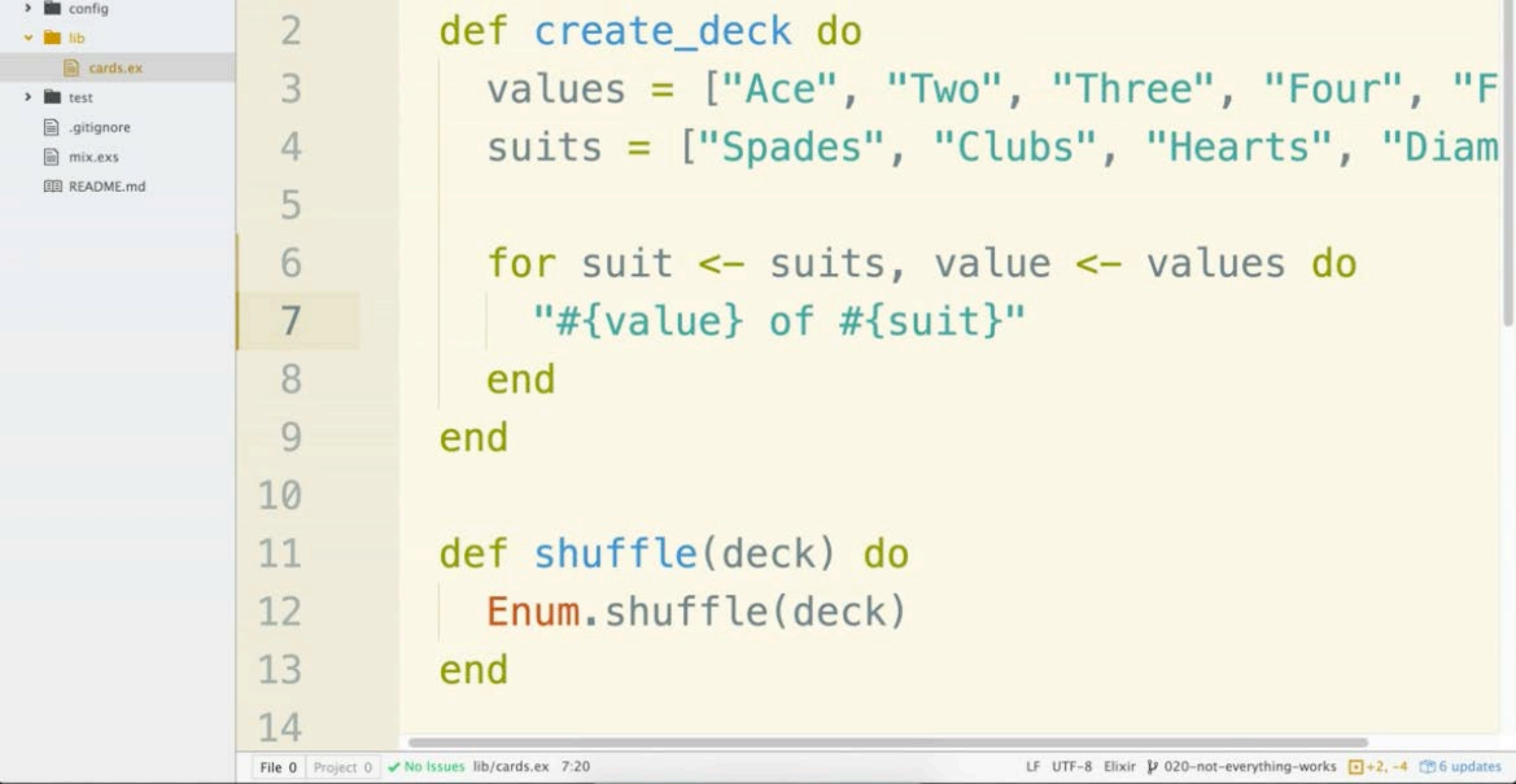

Cards Module
["Ace", "Two", "Three"]
"Two"
contains?
True
Cards Module
["Ace", "Two", "Three"]
"King"
contains?
False

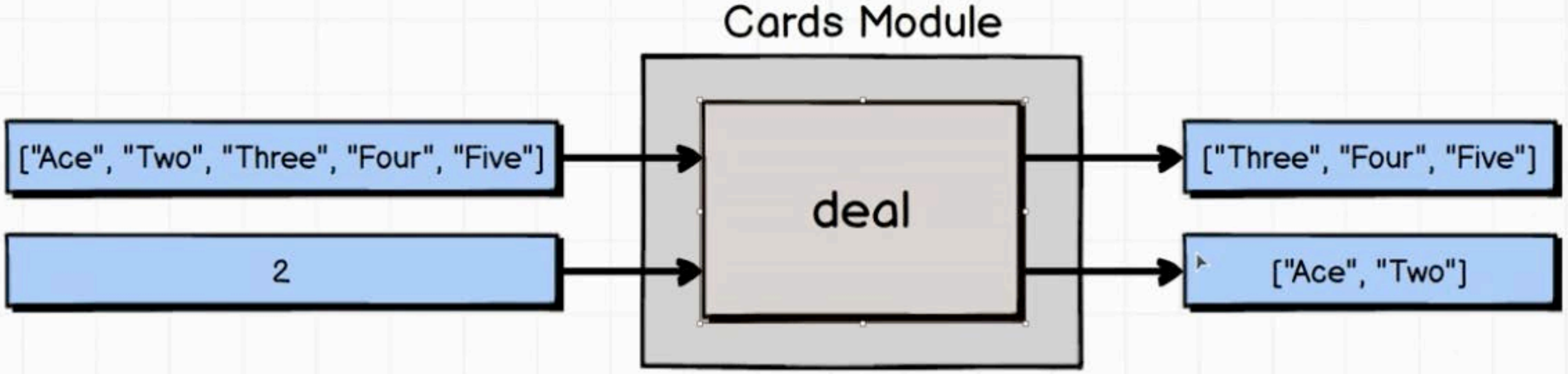

Cards Module
["Ace", "Two", "Three"]
"Two"
contains?
True
Cards Module
["Ace", "Two", "Three"]
"King"
contains?
False


Suits
Spades
Clubs
Hearts
Diamonds
"do" Block
Spades
Result from the comprehension


values
Ace
Two
Three
Four
suits
Spades
Clubs
Hearts
Diamonds
Run for "ace"...
Ace of Spades
Ace of Clubs
Ace of Hearts
Ace of Diamonds
Run for "two"...
Two of Spades
Two of Clubs
Two of Hearts
Two of Diamonds


[
[Ace of Spades Ace of Clubs Ace of Hearts Ace of Diamonds],
[Two of Spades Two of Clubs Two of Hearts Two of Diamonds]
]
List.flatten
[
Ace of Spades Ace of Clubs Ace of Hearts Ace of Diamonds Two of Spades
Two of Clubs Two of Hearts Two of Diamonds
]


config
lib
cards.ex
test
.gitignore
mix.exs
README.md
def create_deck do
values = ["Ace", "Two", "Three", "Four", "F
suits = ["Spades", "Clubs", "Hearts", "Diam
for suit <- suits, value <- values do
"#{value} of #{suit}"
end
end
def shuffle(deck) do
Enum.shuffle(deck)
end
File 0
Project 0
No Issues
lib/cards.ex 7:20
LF UTF-8 Elixir
020-not-everything-works
+2, -4
6 updates


Cards Module
["Ace", "Two", "Three", "Four", "Five"]
2
deal
["Three", "Four", "Five"]
["Ace", "Two"]

Sorts the enumerable by the given function

sort_by(enumerable, mapper, sorter \\ &<=/2)

Sorts the mapped results of the enumerable according to the `sorter` *function*

split(enumerable, count)

Splits the `enumerable` *into two enumerables, leaving* `count` *elements in the first one. If* `count` *is a negative number, it starts counting from the back to the beginning of the enumerable*

split_while(enumerable, fun)

Splits enumerable in two at the position of the element for which `fun` *returns* `false` *for the first time*

sum(enumerable)

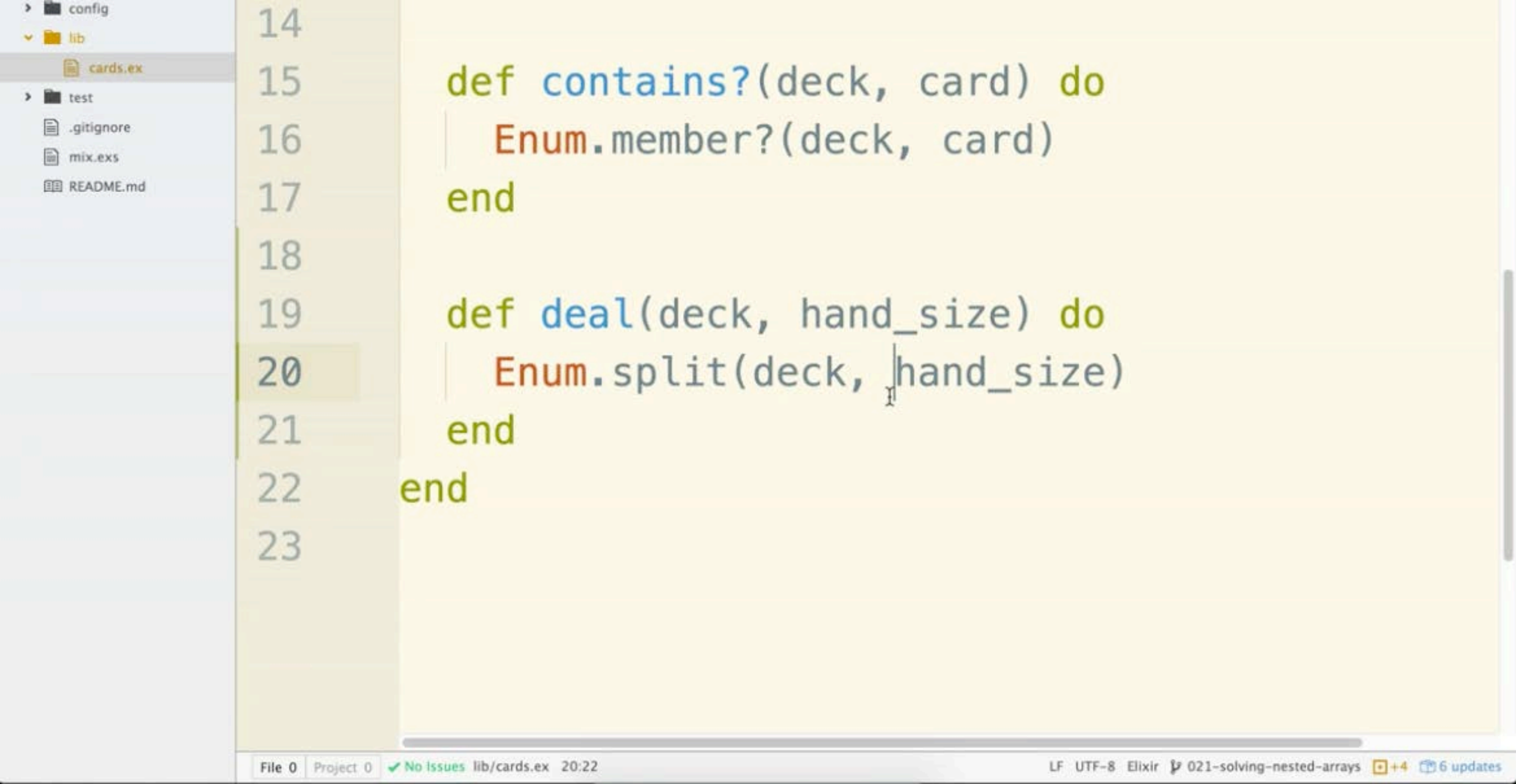
Returns the sum of all elements

take(enumerable, count)

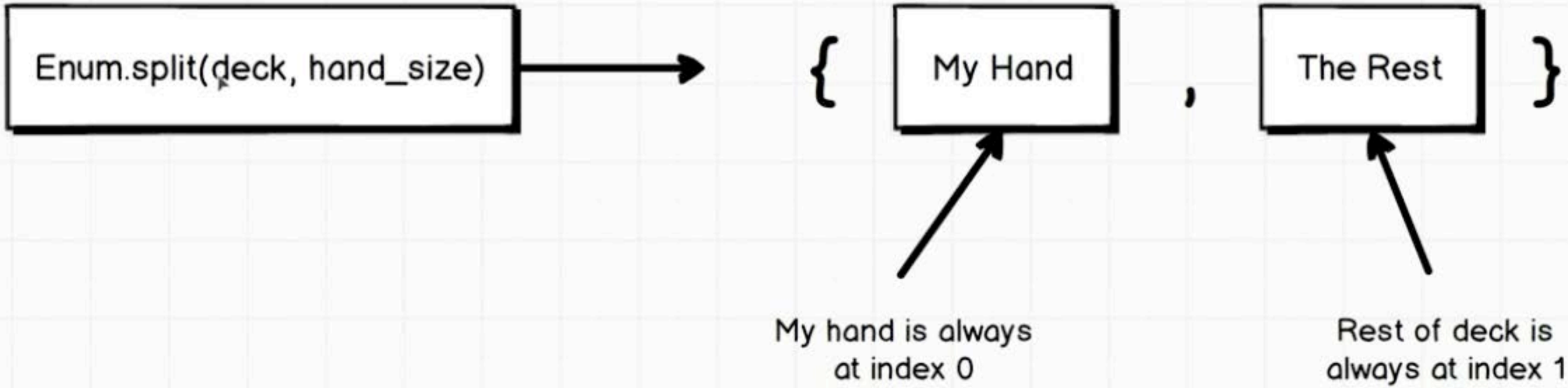
Be aware that a negative `count` implies the `enumerable` will be enumerated twice: once to calculate the position, and a second time to do the actual splitting.

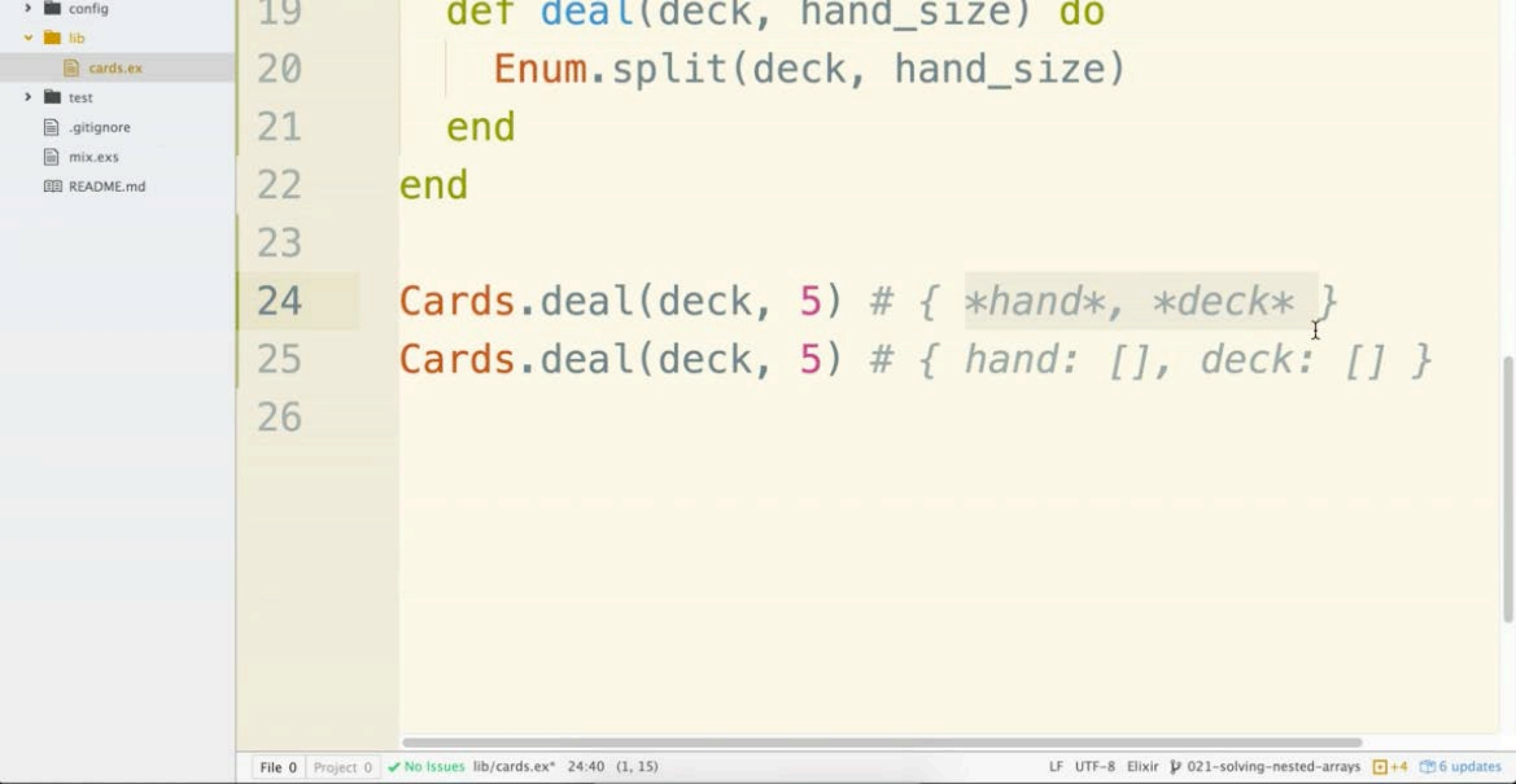
Examples

```
iex> Enum.split([1, 2, 3], 2)
{[1, 2], [3]}

iex> Enum.split([1, 2, 3], 10)
{[1, 2, 3], []}

iex> Enum.split([1, 2, 3], 0)
{[], [1, 2, 3]}

iex> Enum.split([1, 2, 3], -1)
```


config
lib
cards.ex
test
.gitignore
mix.exs
README.md
14
15 def contains?(deck, card) do
16 Enum.member?(deck, card)
17 end
18
19 def deal(deck, hand_size) do
20 Enum.split(deck, hand_size)
21 end
22 end
23
File 0
Project 0
No Issues
lib/cards.ex 20:22
LF UTF-8 Elixir
021-solving-nested-arrays
+4
6 updates

```
 "Three of Diamonds", "Four of Diamonds",
 "Five of Diamonds"]
iex(22)> Cards.deal(deck, 5)
{["Ace of Spades", "Two of Spades", "Three of Spades",
  "Four of Spades", "Five of Spades"],
 ["Ace of Clubs", "Two of Clubs", "Three of Clubs",
  "Four of Clubs", "Five of Clubs", "Ace of Hearts",
  "Two of Hearts", "Three of Hearts", "Four of Hearts",
  "Five of Hearts", "Ace of Diamonds", "Two of Diamonds",

  "Three of Diamonds", "Four of Diamonds",
  "Five of Diamonds"]}
iex(23)>
```


Enum.split(deck, hand_size)
{
My Hand
,
The Rest
}
My hand is always at index 0
Rest of deck is always at index 1


config
lib
cards.ex
test
.gitignore
mix.exs
README.md
19 def deal(deck, hand_size) do
20 Enum.split(deck, hand_size)
21 end
22 end
23
24 Cards.deal(deck, 5) # { *hand*, *deck* }
25 Cards.deal(deck, 5) # { hand: [], deck: [] }
26
File 0 Project 0 No issues lib/cards.ex* 24:40 (1, 15)
LF UTF-8 Elixir 021-solving-nested-arrays +4 6 updates