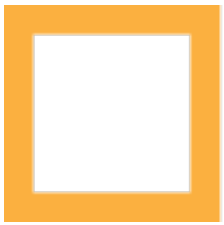
Functional Programming and REST
Functional Programming

Jens Egholm Pedersen and Anders Kalhauge



cphbusiness

Spring 2017

Outline

Imperative programming

Instructing program state

Imperative programming

Instructing program state

- State

Imperative programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Instructing program state

- State
- Statements (action of stating)

Procedural programming

Imperative programming (can) lack structure

Procedural programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Imperative programming (can) lack structure

- Procedures group statements

Procedural programming

Imperative programming (can) lack structure

- Procedures group statements
  == blocks

Procedural programming

Imperative programming (can) lack structure

- Procedures group statements
  == blocks == modules

Procedural programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Imperative programming (can) lack structure

- Procedures group statements
  == blocks == modules == functions (not mathematical)
- Scope

Structures procedures using objects

Object oriented programming

Structures procedures using objects

- Objects

Object oriented programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Structures procedures using objects

- Objects
- Classes

Object oriented programming

Structures procedures using objects

- Objects
- Classes
- Types

Object oriented programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Structures procedures using objects

- Objects
- Classes
- Types (broken)

Object oriented programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Structures procedures using objects

- Objects
- Classes
- Types (broken)
- Inheritance and delegation

Object oriented programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Structures procedures using objects

- Objects
- Classes
- Types (broken)
- Inheritance and delegation
- Polymorphism

Object oriented programming

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Structures procedures using objects

- Objects
- Classes
- Types (broken)
- Inheritance and delegation
- Polymorphism
- Exceptions as control structures

Fix Java!

Clone the `java-exercises` from `cphbus-functional-programming`

`https://github.com/cphbus-functional-programming/java-exercises`

Work on the `FixMe` files in the `breakingjava` folder

Goal: Fix the broken code *without compiling it*!

Untouched by the above misery

Untouched by the above misery

- Types

Functional programming

Untouched by the above misery

- Types
- Pure functions (no side-effects)

Functional programming

Untouched by the above misery

- Types
- Pure functions (no side-effects)
- Recursion

Untouched by the above misery

- Types
- Pure functions (no side-effects)
- Recursion
- Higher-order functions

cphbusiness
COPENHAGEN BUSINESS ACADEMY

In Java

```
Person p = null;
```

cphbusiness
COPENHAGEN BUSINESS ACADEMY

In Java

```
Person p = null;
```

In Elm

```
Person p = Person "Hermann Minkowski"
```

cphbusiness
COPENHAGEN BUSINESS ACADEMY

In Java

```
Person doSomething() {
  fireNuclearMissiles();
  return new Person("Robby the Robot")
}
```

Side-effects

In Java

```
Person doSomething() {
  fireNuclearMissiles();
  return new Person("Robby the Robot")
}
```

In Elm

```
doSomething : Person
doSomething = Person "Isaac Asimov"
```

Exceptions

In Java

```
Person doSomething() {
  throw new RuntimeException("I'm unchecked!");
}
```

Exceptions

In Java

```
Person doSomething() {
  throw new RuntimeException("I'm␣unchecked!");
}
```

In Elm

```
doSomething : Either String Person
doSomething = Left "'Elp!"
```

Maybe

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Maybe

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

If Elm does not have `null` values or exceptions, how do you represent a failure?

Maybe

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

If Elm does not have `null` values or exceptions, how do you represent a failure?

```
type Maybe a
  = Just a
  | Nothing
```

Maybe

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

If Elm does not have `null` values or exceptions, how do you represent a failure?

```
type Maybe a
  = Just a
  | Nothing
```

```
getPerson : Int -> Maybe Person
getPerson id = ...
```

Union types 1/2

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

```
maybeBaby : Maybe String
```

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

```
maybeBaby : Maybe String
```

maybeBaby can now either be

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

```
maybeBaby : Maybe String
```

maybeBaby can now either be `Just String`

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

```
maybeBaby : Maybe String
```

maybeBaby can now either be `Just String` or `Nothing`

Union types 1/2

cphbusiness
COPENHAGEN BUSINESS ACADEMY

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

Is this a union type?

```
int amIUnion = 20;
```

```
type Maybe a
  = Just a
  | Nothing
```

```
maybeBaby : Maybe String
```

`maybeBaby` can now either be `Just String` or `Nothing`

It **cannot be anything else**.

Capturing errors

Capturing errors

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Capturing errors

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Did we forget something?

Capturing errors

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Did we forget something? Yes! The exception!

Capturing errors

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Did we forget something? Yes! The exception!

```
type Either a b
  = Left a          -- The exception
  | Right b         -- The success
```

Capturing errors

In Java

```
Person getPerson(Long id) throws IOException {
  return database.getPersonById(id);
}
```

Did we forget something? Yes! The exception!

```
type Either a b
  = Left a          -- The exception
  | Right b         -- The success
```

```
getPerson : Int -> Either Exception Person
getPerson id = ...
```

Union types 2/2

A union type is a piece of memory which can take the form of one or more values, but only one at the time.

```
type Either a b
  = Left a          -- The exception
  | Right b         -- The success
```

For a HTTP call, what would you expect as input?

For a HTTP call, what would you expect as input?
For a HTTP call, what would you expect as output?

For a HTTP call, what would you expect as input?

For a HTTP call, what would you expect as output?

```
type Result error value
  = Ok value
  | Err error
```

For a HTTP call, what would you expect as input?
For a HTTP call, what would you expect as output?

```
type Result error value
  = Ok value
  | Err error
```

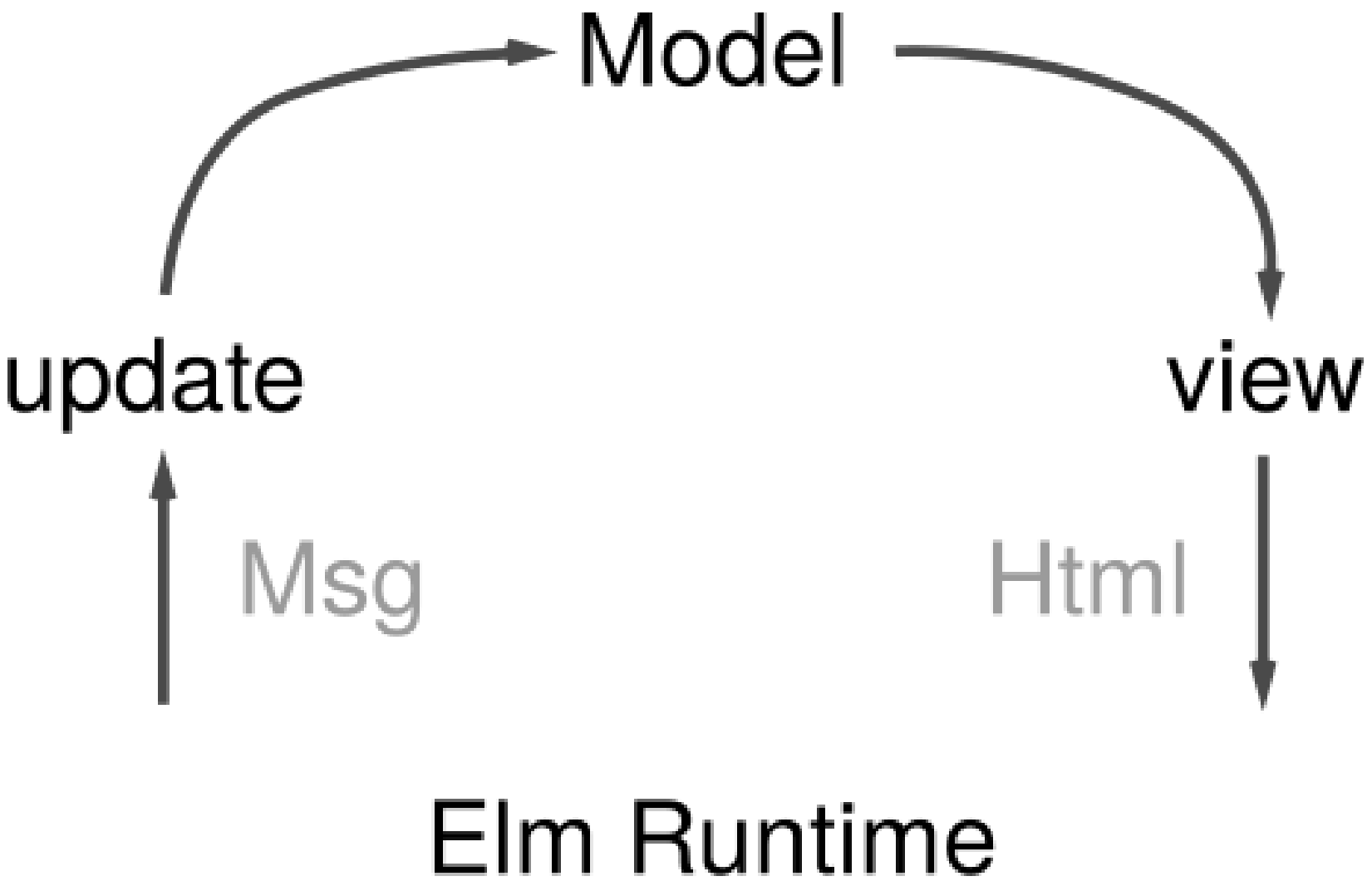
Union type

HTTP request


```
type alias Request a
  = Request a
```

```
getString : String -> Request String
```

HTML updates



HTTP to HTML

cphbusiness
COPENHAGEN BUSINESS ACADEMY

To insert the HTTP result, we have to put it into the HTML page with a `Cmd`

To insert the HTTP result, we have to put it into the HTML page with a `Cmd`

```
type Msg
  = NewContent ?
```

To insert the HTTP result, we have to put it into the HTML page with a `Cmd`

```
type Msg
  = NewContent ?
```

```
type Msg
  = NewContent (Result Http.Error String)
```

Getting the HTTP Result

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

Getting the HTTP Result

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

```
HTTP.send : (Result Error a -> msg) ->
            Request a -> Cmd msg
```

Getting the HTTP Result

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

```
HTTP.send : (Result Error a -> msg) ->
            Request a -> Cmd msg
```

Translated:

- `HTTP.send` takes two parameters

Getting the HTTP Result

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

```
HTTP.send : (Result Error a -> msg) ->
            Request a -> Cmd msg
```

Translated:

- `HTTP.send` takes two parameters
- 1: One function which takes a result and converts it into something else

Getting the HTTP Result

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

```
HTTP.send : (Result Error a -> msg) ->
            Request a -> Cmd msg
```

Translated:

- `HTTP.send` takes two parameters
- 1: One function which takes a result and converts it into something else
- 2: One request which performs the HTTP call

Getting the HTTP Result

Now we have a `HTTP Request` and a way to insert it into our view

But how do we get the `HTTP Result`?

```
HTTP.send : (Result Error a -> msg) ->
            Request a -> Cmd msg
```

Translated:

- `HTTP.send` takes two parameters
- 1: One function which takes a result and converts it into something else
- 2: One request which performs the HTTP call
- `HTTP.send` returns the message extracted from the first function

Putting it together

cphbusiness
COPENHAGEN BUSINESS ACADEMY

```
import Http

type Msg = Click | NewBook (Result Http.Error String)

update : Msg -> Model -> Model
update msg model =
  case msg of
    Click -> ( model, getWarAndPeace )

    NewBook (Ok book) -> ...

    NewBook (Err _) -> ...

getWarAndPeace : Cmd Msg
getWarAndPeace =
  Http.send NewBook <|
    Http.getString "https://example.com/some_book.md"
```

Putting it together

Live coding!

The code from today can be found here: `https://github.com/cphbus-functional-programming/elm-exercises`

REST assignment

Write a server in a language of your choice with two HTTP REST methods:

1. `GET /counter`: increments and returns an integer counter
2. `PUT /counter/{value}`: sets the counter to `value`

REST assignment

Write a server in a language of your choice with two HTTP REST methods:

1. `GET /counter`: increments and returns an integer counter
2. `PUT /counter/{value}`: sets the counter to `value`

Write an Elm client using the model - view - update architecture. Your client must have:

1. A model containing one counter `Model { counter: Int }`
2. A view with two HTML buttons (get and set) as well as the counter in an HTML `H2` element
3. An update part which can 1) get the counter value from your REST service and 2) set the counter to a fixed value of 1

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Write a server in a language of your choice with two HTTP REST methods:

1. `GET /counter`: increments and returns an integer counter
2. `PUT /counter/{value}`: sets the counter to `value`

Write an Elm client using the model - view - update architecture. Your client must have:

1. A model containing one counter `Model { counter: Int }`
2. A view with two HTML buttons (get and set) as well as the counter in an HTML `H2` element
3. An update part which can 1) get the counter value from your REST service and 2) set the counter to a fixed value of 1

Hand in before **27th March 23:59**