Introduction to Haskell & some category theory

Wellington Functional Programming

Finlay Thompson

26 March 2015

DRAGONFLY
Data Science

Introduction

Haskell is strange ?

Haskell is strange ?

A functional programming language

Haskell is strange ?

A functional programming language

With lazy evaluation

Haskell is strange ?

A functional programming language

With lazy evaluation

Pure, with no side effects

Haskell is strange ?

A functional programming language

With lazy evaluation

Pure, with no side effects

Fairly old

Haskell is strange ?

A functional programming language

With lazy evaluation

Pure, with no side effects

Fairly old, fairly odd

DRAGONFLY
Data Science

Haskell is hard ?

Haskell is hard ?

My program won't compile,
and I don't know why ?

Haskell is hard ?

My program won't compile,
and I don't know why ?

The tutorials online are
confusing.

Haskell is hard ?

My program won't compile,
and I don't know why ?

The tutorials online are
confusing.

Oh god, I am reading math !

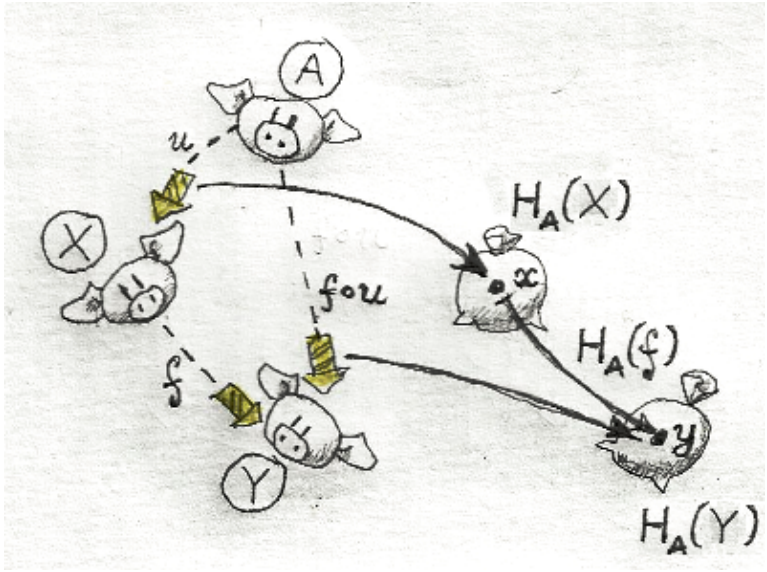
Haskell is hard ?

My program won't compile, and I don't know why ?

The tutorials online are confusing.

Oh god, I am reading math !

Huh ?



DRAGONFLY
Data Science

Haskell is impractical ?

Haskell is impractical ?

Strong type system gets in the way

Haskell is impractical ?

Strong type system gets in the way

Hard to install, and find good libraries

Haskell is impractical ?

Strong type system gets in the way

Hard to install, and find good libraries

Impossible to find other developers

Haskell is impractical ?

Strong type system gets in the way

Hard to install, and find good libraries

Impossible to find other developers

DRAGONFLY
Data Science

But, Haskell is awesome!

But, Haskell is awesome!

Is Haskell weird ?

But, Haskell is awesome!

Is Haskell weird ?

- No, just different. Its the other languages that are weird.

But, Haskell is awesome!

Is Haskell weird ?

- No, just different. Its the other languages that are weird.

Is Haskell hard ?

But, Haskell is awesome!

Is Haskell weird ?

- No, just different. Its the other languages that are weird.

Is Haskell hard ?

- No, it makes you think differently, which is good.

But, Haskell is awesome!

Is Haskell weird ?

- No, just different. Its the other languages that are weird.

Is Haskell hard ?

- No, it makes you think differently, which is good.

Is Haskell impractical ?

But, Haskell is awesome!

Is Haskell weird ?

- No, just different. Its the other languages that are weird.

Is Haskell hard ?

- No, it makes you think differently, which is good.

Is Haskell impractical ?

- Hackage has thousands of libraries
- Haskell is fast, and getting faster

To learn Haskell,
it helps to learn a little category theory.

To learn Haskell,
it helps to learn a little category theory.

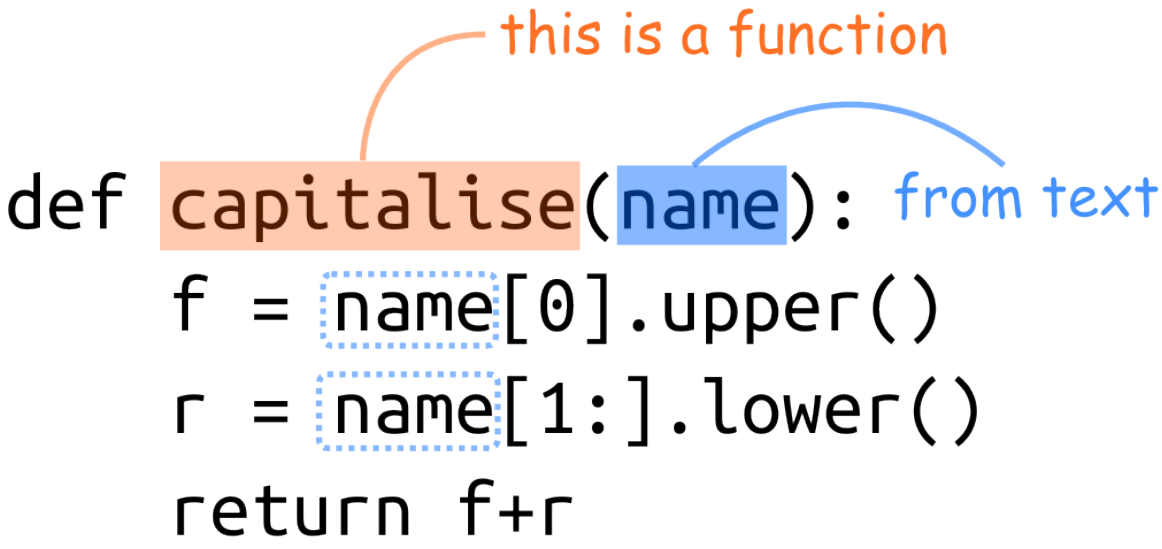
Actually, I reckon you already know category theory!

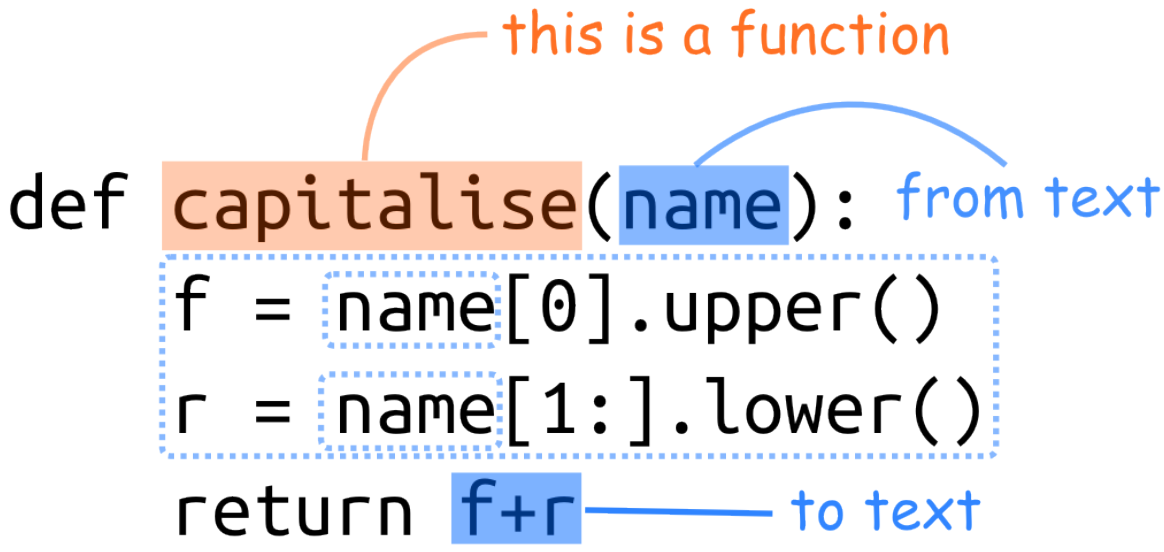
Anatomy of a function

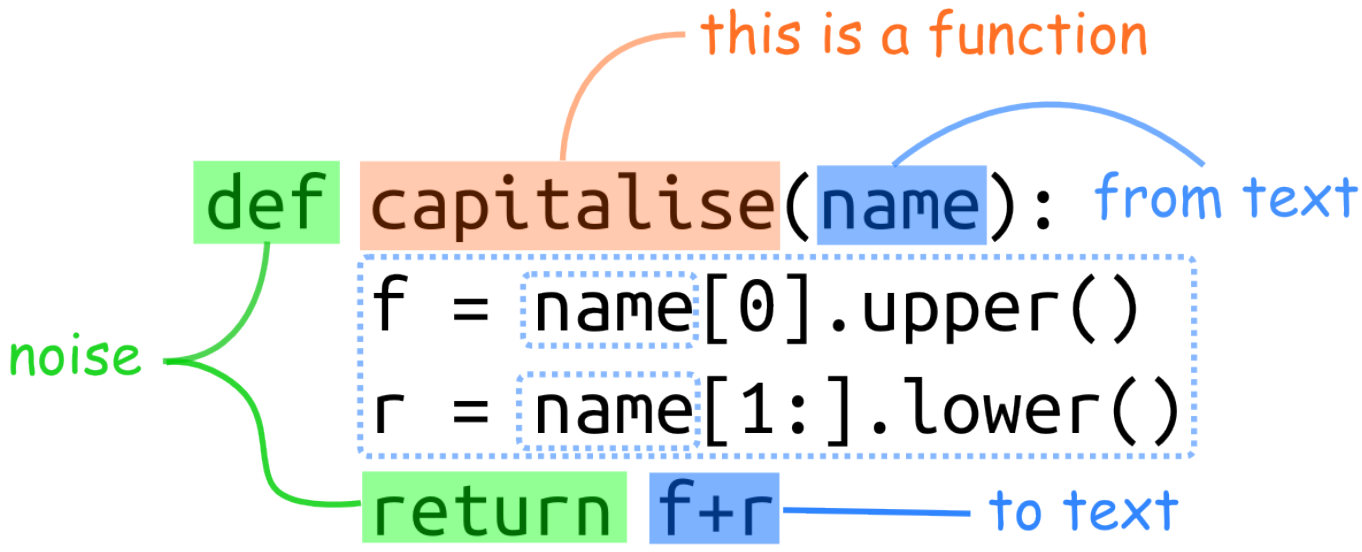
```
def capitalise(name):
    f = name[0].upper()
    r = name[1:].lower()
    return f+r
```

this is a function

```
def capitalise(name):
    f = name[0].upper()
    r = name[1:].lower()
    return f+r
```


this is a function
def capitalise(name):
from text
f = name[0].upper()
r = name[1:].lower()
return f+r

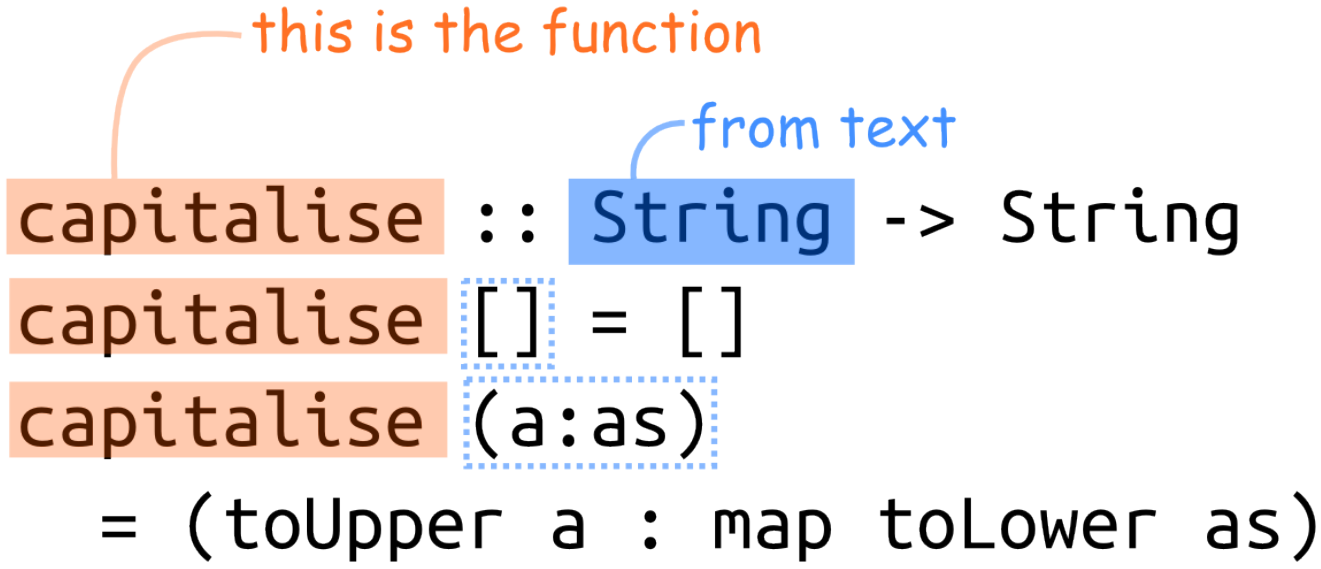

this is a function
def capitalise(name):
from text
f = name[0].upper()
r = name[1:].lower()
return f+r
to text

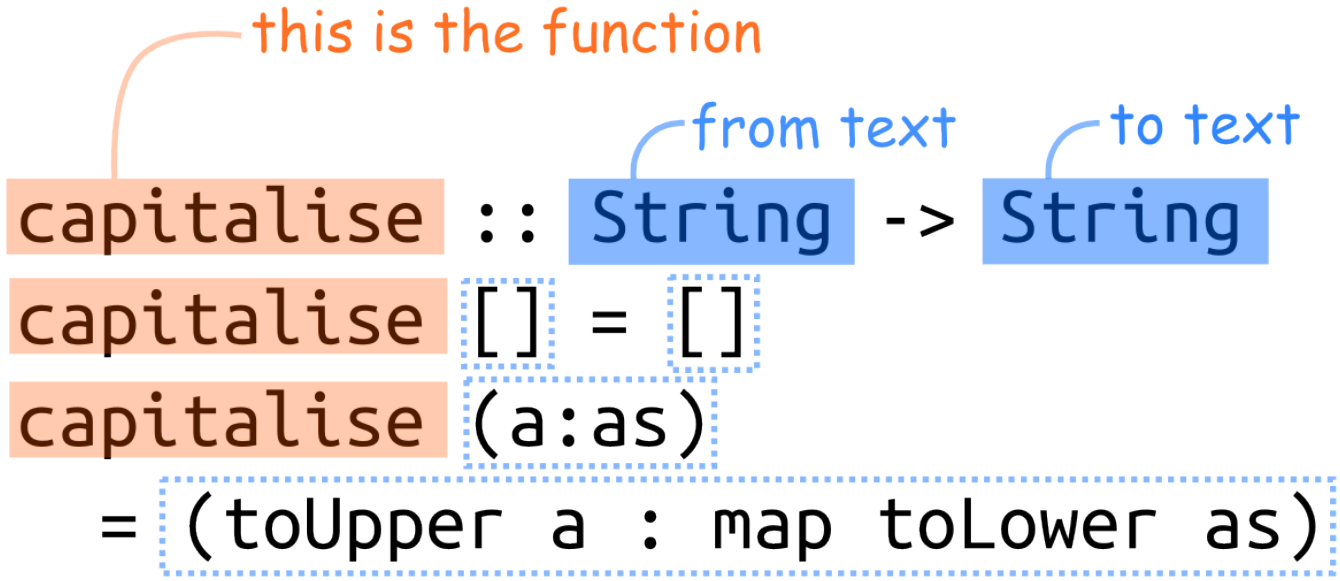

this is a function
def capitalise(name): from text
f = name[0].upper()
r = name[1:].lower()
noise
return f+r
to text

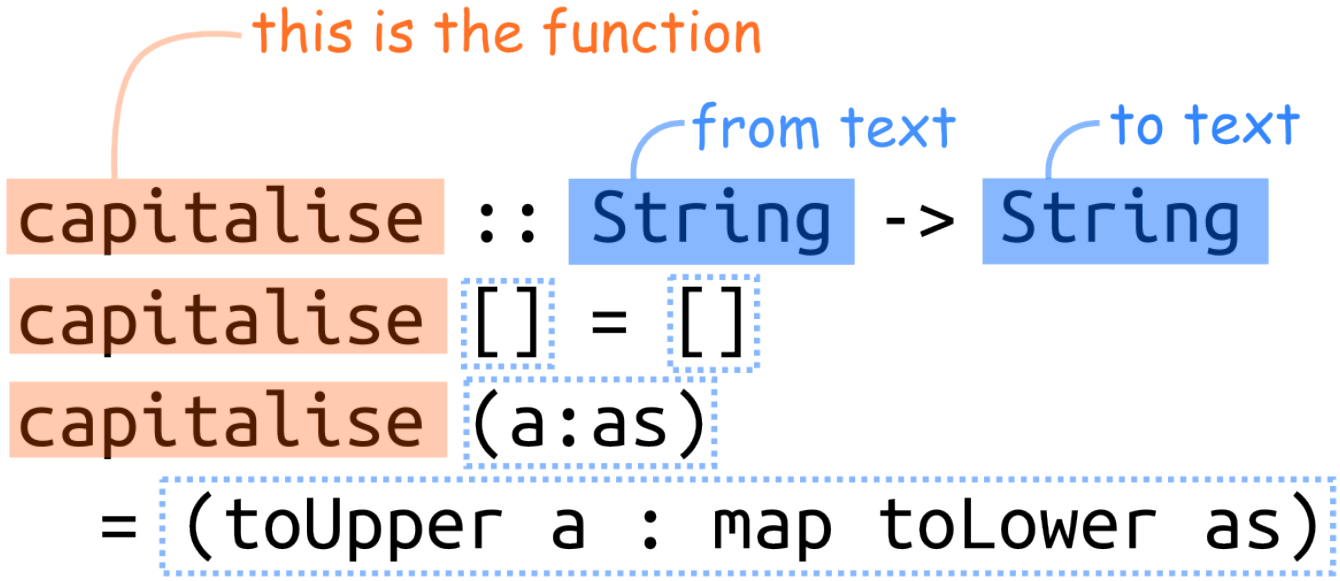
```
capitalise :: String -> String
capitalise [] = []
capitalise (a:as)
  = (toUpper a : map toLower as)
```

this is the function

```
capitalise :: String -> String
capitalise [] = []
capitalise (a:as)
   = (toUpper a : map toLower as)
```


this is the function
from text
capitalise :: String -> String
capitalise [] = []
capitalise (a:as)
= (toUpper a : map toLower as)


this is the function
from text
to text
capitalise :: String -> String
capitalise [] = []
capitalise (a:as)
= (toUpper a : map toLower as)


this is the function
from text
to text
capitalise :: String -> String
capitalise [] = []
capitalise (a:as)
= (toUpper a : map toLower as)

no noise now !

A little category theory


DRAGONFLY
Data Science

Programming patterns


DRAGONFLY
Data Science

Functors


DRAGONFLY
Data Science

Monads


DRAGONFLY
Data Science