PROGRAMMING IN HASKELL

Chapter 1 - Introduction

Acknowledgement

▮ Slides generously provided by Prof Hutton, University of Nottingham.

▮ Related textbook:

  ▮ Hutton. 2016. Programming in Haskell (2nd edition). Cambridge University Press.

▮ Some additional/revisions added 'silently'.

What is a Functional Language?

Opinions differ, and it is difficult to give a precise definition, but generally speaking:

- Functional programming is <u>style</u> of programming in which the basic method of computation is the application of functions to arguments;

- A functional language is one that <u>supports</u> and <u>encourages</u> the functional style.

Example

Summing the integers 1 to 10 in Java:

```java
int total = 0;
for (int i = 1; i ≤ 10; i++)
    total = total + i;
```

The computation method is <u>variable assignment</u>.

Example

Summing the integers 1 to 10 in Haskell:

```
sum [1..10]
```

The computation method is <u>function application</u>.

The λ-Calculus as a Programming Language

- The λ-calculus can be seen as a simple programming language.
- Advantages
    1. It has a simple syntax and a precise semantics;
    2. Computation is defined as the application of just three rules;
    3. It is free from "architectural" influences;
    4. It is mathematical foundations enable proofs over programs.

The λ-Calculus as a Programming Language

▌ λ-calculus: formalise semantics of other programming languages

▌ λ-calculus: basis for functional programming.

From λ-Calculus to Functional Programming

- Disadvantages of the λ-calculus
  - Programs written as λ-expressions can be long;
  - It may be hard to follow all the substitutions.
- Solution: higher-level language on top of the λ-calculus.
- Resulting class of languages: functional programming languages.
- Functional programming = λ-calculus +"syntactic sugar".

The λ-Calculus as a Programming Language

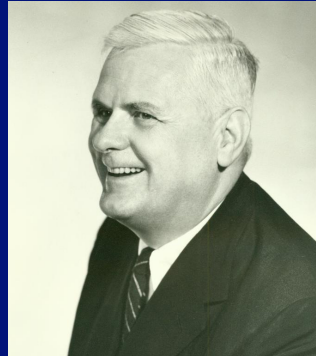- λ-calculus: formalise semantics of other programming languages

- λ-calculus: basis for functional programming.

Relevance

- λ-calculus ties into issues about FSAs, Turing-machines, and the treatment of infinite data....

- A functional programming language relates to these issues....
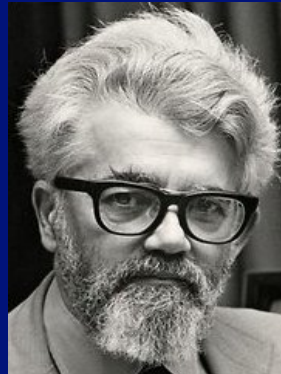
Historical Background

1930s:

Alonzo Church develops the <u>lambda calculus,</u>
a simple but powerful theory of functions.

Historical Background

1950s:

John McCarthy develops <u>Lisp</u>, the first functional language, with some influences from the lambda calculus, but retaining variable assignments.
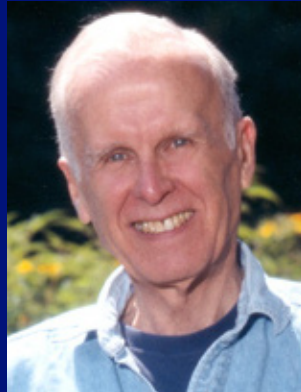
Historical Background

1960s:

Peter Landin develops <u>ISWIM</u>, the first *pure* functional language, based strongly on the lambda calculus, with no assignments.

Historical Background

1970s:

John Backus develops <u>FP</u>, a functional language that emphasizes *higher-order functions* and *reasoning about programs*.

Historical Background

1970s:

Robin Milner and others develop <u>ML</u>, the first modern functional language, which introduced *type inference* and *polymorphic types*.

Historical Background

1970s - 1980s:

David Turner develops a number of *lazy* functional languages, culminating in the <u>Miranda</u> system.

Historical Background

1987:

An international committee starts the development of <u>Haskell</u>, a standard lazy functional language.

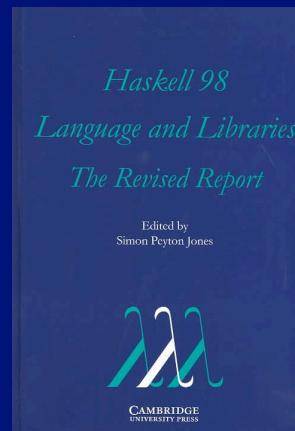# Historical Background

1990s:

Phil Wadler and others develop *type classes* and *monads,* two of the main innovations of Haskell.

Historical Background

2003:

The committee publishes the Haskell Report, defining a stable version of the language; an updated version was published in 2010.

Historical Background

2010-date:

Standard distribution, library support, new language features, development tools, use in industry, influence on other languages, etc.

A Taste of Haskell

```
f []       = []
f (x:xs) = f ys ++ [x] ++ f zs
            where
                ys = [a | a ← xs, a ≤ x]
                zs = [b | b ← xs, b > x]
```

?

Related Material

- General Haskell site:
  - https://www.haskell.org/
- Documentation (tutorials etc):
  - https://www.haskell.org/documentation
- See the downloads link in the general site for the Haskell Platform.