Concepts of programming languages

Rust

Tobias van Driessel, Tom Freijsen, Ruben Schenkhuizen, Floris Schild, Mats Veldhuizen

Universiteit Utrecht

[Faculty of **Science**
Information and Computing Sciences]

Presentation Schedule

- Background
- Design principles
- What problems does it solve and how?
- Practical details

Background

- Personal project of Mozilla employee
- Sponsored by Mozilla Research
- First stable release (1.0) in 2015
- Used in Firefox and by Dropbox
- Most loved programming language - SO Developer Survey

Quick Overview

- Statically typed
- Functional **and** Imperative paradigms
- Strict language

Beautiful Quotes

Mozilla Research:

"Rust is a systems programming language that runs blazingly fast, prevents segfaults, and guarantees thread safety."

"It fuses the expressive and intuitive syntax of high-level languages with the control and performance of a low-level language."

Features

Features

- zero-cost abstractions

Features

- zero-cost abstractions
- move semantics

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching
- type inference

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching
- type inference
- minimal runtime

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching
- type inference
- minimal runtime
- efficient C bindings

Features

- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching
- type inference
- minimal runtime
- efficient C bindings

First slide! Please use Markdown to write your slides.

This makes sure that slides will be consistent – and easy for me to edit in the future.

Start a new slide with by beginning a new line three dashes `---`.

For example:

```
---

My contents

---
```

Titles

You can use the hash symbol # to make the title of a slide.

Subtitle

You can use more than one hash symbol ## to have subtitles on your slide.

- Bullet lists
- are pretty easy
- too!

Emphasis

You can include a word in asterisks to add *emphasis* or two asterisks to make it **bold**.

That is:

```
*emphasis* vs **bold**
```

Images

Please include any images in the `img` subdirectory.

You can refer to images using the usual markdown syntax:

My caption

Staged builds

This is quite easy

Staged builds

This is quite easy

Just insert `. . .` on a new line when you want the slide to appear incrementally.

Code

You can use backticks to include inline code such as `x` or `y`.

Use three backticks to introduce a code block:

```
main = print "Hello world!"
```

Syntax highlighting

There are syntax highlighting options for the most widely used languages.

```
foo y = let x = 4 in x + z
  where
  z = 12
```

Making slides

I've included a Makefile to build slides.

You will need to have the Haskell tool `pandoc` installed:

```
> cabal install pandoc
> make
```

Working with markdown

You may want to install the markdown mode for emacs (or some other editor of choice).

I've included some file local variables at the bottom of this file – you may find them useful.

Inline latex

You can always use *inline* LaTeXcommands if you want.

But try to avoid this if you can.

Most Markdown commands should suffice.

LaTeXis useful for formula's

$$\tau + x = \sigma \tag{1}$$

Or inline formulas, enclosed in dollar symbols like so $\tau + x$.

Questions

If you can’t get things to work, don’t hesitate to get in touch!