GSoC: Text/UTF-8

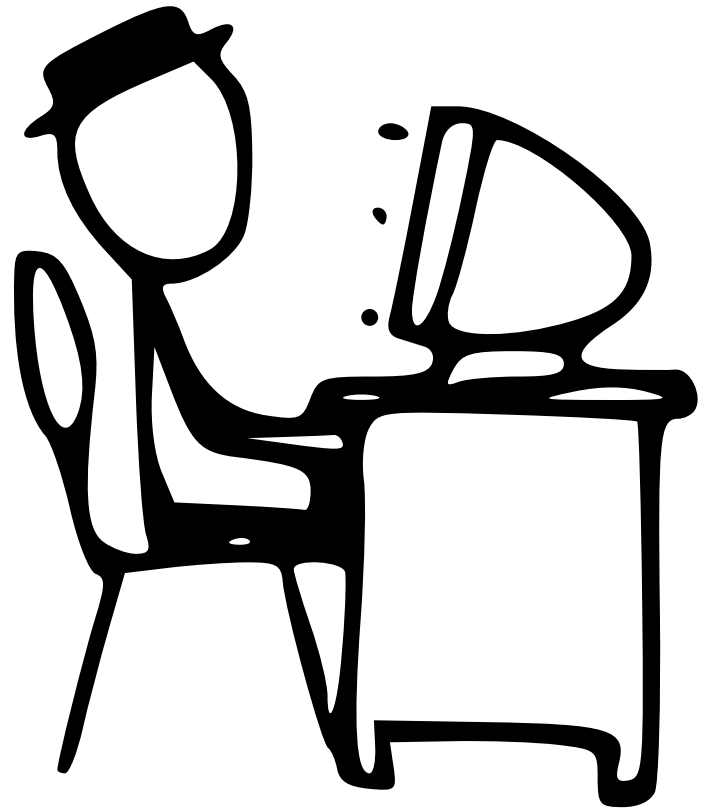
CamHac

Jasper Van der Jeugt

August 14, 2011

Hello!

My name is Jasper
Student at UGent
I write Haskell
GhentFPG
`@jaspervdj`
`jaspervdj.be`



Overview

Credit where credit is due

Bryan O'Sullivan
Edward Kmett
Johan Tibell
Tom Harper

Overview

Introduction

Use `String` with care

Unicode crash course

Results

Overview

Introduction

Use `String` with care

Unicode crash course

Results

Use `String` with care

Number of unicode characters?

17 planes
Each plane: 2^{16} characters

$\log_2(17 * 2^{16}) = 20.087...$
21 bits per character

Use `String` with care

`data` Char = C# **Char**#

`C#`: word

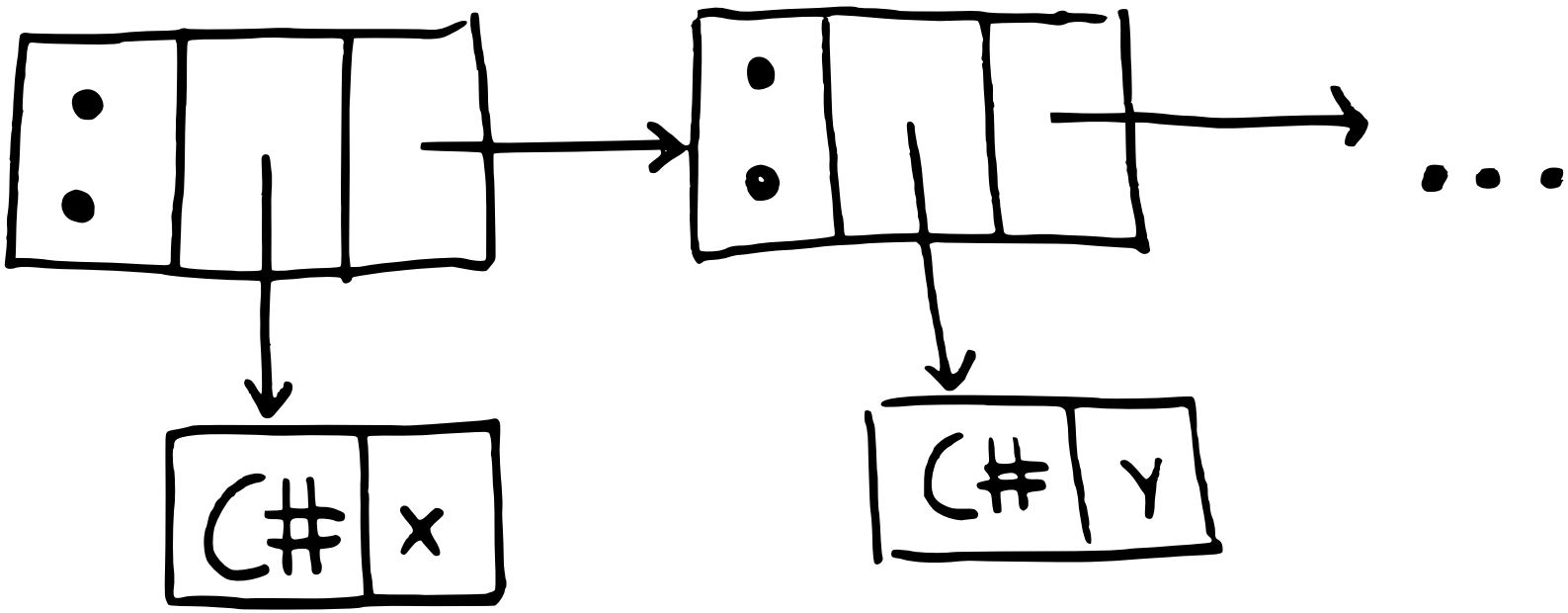
`Char#`: 32 bits

Use `String` with care

```
type String = [Char]
```

```
data [] a = [] | a : [a]
```

Use `String` with care



Use `String` with care

Two points:

1. Use strict arrays
2. Don't use a 32-bit encoding

Overview

Introduction
Use `String` with care
Unicode crash course
Results

Unicode crash course

This is UTF-8

0xxx xxxx

110x xxxx <cb>

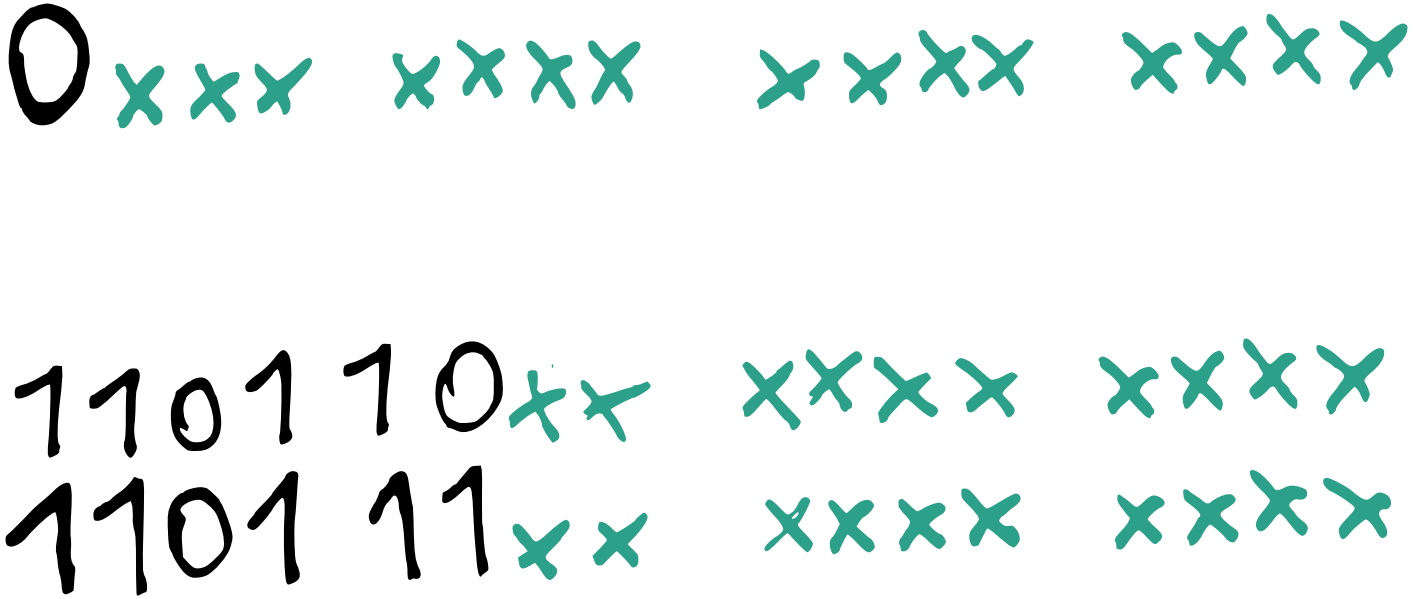
1110 xxxx <cb> <cb>

11110xxx <cb> <cb> <cb>

<cb> = 10xx xxxx

Unicode crash course

This is UTF-16



Unicode crash course

Two points:

1. Some things are inherently faster using UTF-8
2. Some things are inherently faster using UTF-16

Unicode crash course

Results depend on:

1. The application (e.g. lots of processing vs. static in-memory database)
2. The language (e.g. English vs. Japanese)

Overview

Introduction
Use `String` with care
Unicode crash course
Results

Results: pure functions

```
stream   :: Text -> Stream Char
unstream :: Stream Char -> Text

map :: (Char -> Char)
    -> Text -> Text
map f =
    unstream . map' f . stream
```

Results: pure functions

```
map :: (Char -> Char)
    -> Text -> Text
map f =
    unstream . map' f . stream
```

"*Generally*" a little slower for UTF-8

Results: applications

```
program :: ByteString
        -> ByteString
program = encodeUtf8
        . map someF
        . decodeUtf8
```

"*Generally*" a little faster for UTF-8

Results

Other advantages

Integrate with C libraries (e.g. `pcre`)
Reduced memory usage
Fast output path (for e.g. `aeson`)

Results

GSoC: progress

Conversion:	done
Optimization:	done*
Summary report:	in progress
Switch:	?

**always room for improvement*

Questions?