Hello, `noweb`!

An exploration of literate Idris programming via `noweb`.

Eric Bailey

January 6, 2017

Abstract

What follows is an attempt at using `noweb` to write a literate program in Idris. While Idris provides some literate programming support of its own, it's rather basic (like Literate Haskell), and doesn't allow users to present code chunks out of order or do any sort of cross-referencing.

Contents

1 hello.idr Outline

`hello.idr` is rather simple. It consists of a module declaration with a type signature and definition for the `main` function.

2a ⟨* 2a⟩≡
⟨*Module declaration* 2b⟩

⟨*main type signature* 2d⟩
⟨*main definition* 2e⟩

2 Module Declaration

Declare the module `Main`, including a docstring, which is another `noweb` chunk.

2b ⟨*Module declaration* 2b⟩≡ (2a)
```
||| ⟨Hello message 2c⟩
module Main
```

The docstring above consists of the following message, which we'll also print later, using a `noweb` reference.

2c ⟨*Hello message* 2c⟩≡ (2)
```
Hello, noweb!
```

3 The main Function

`main` is an `IO` action that doesn't return any value, i.e.

2d ⟨*main type signature* 2d⟩≡ (2a)
```
main : IO ()
```

Output the message to stdout with a trailing newline.

2e ⟨*main definition* 2e⟩≡ (2a)
```
main = putStrLn "⟨Hello message 2c⟩"
```

4 Chunks

⟨* 2a⟩ 2a
⟨*Hello message* 2c⟩ 2b, 2c, 2e
⟨*main definition* 2e⟩ 2a, 2e
⟨*main type signature* 2d⟩ 2a, 2d
⟨*Module declaration* 2b⟩ 2a, 2b