Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Week 11
GUI Development

Glenn.Strong@scss.tcd.ie
https://scss.tcd.ie/Glenn.Strong/

GUI Development

As a change of pace, let's look at a library for doing something practical.

Threepenny GUI by Heinrich Apfelmus is a library for creating user interfaces. It is implemented as a FRP DSL.

Functional Reactive Programming

- Threepenny GUI is based on the earlier "Reactive Banana" FRP library, but simplifies it greatly.
- Creates a browser-based UI
- Unlike, say, Scotty this is intended for use as a *local* web server with a tight interaction loop.

GUI Development

- The library is designed around a weak form of what Gill et. al. called a “Remote Monad”.
- Programs expressed in a reactive style by connecting up event flows.
- Haskell code specifies an HTML document. Threepenny converts this to JavaScript that executes on the page

```haskell
import Graphics.UI.Threepenny

main :: IO ()
main = do
  startGUI defaultConfig showMessage

showMessage :: Window -> UI ()
showMessage window = do
  getBody window #+ [string "Hello, world!"]
  return ()
```

GUI Development

```
startGUI :: Config -> (Window -> UI ()) -> IO ()
```

- “Config” is a value that configures the server (lets us specify port numbers, etc)
- “Window” represents the DOM “window” object
- “UI” is a monad (based on a transformed IO monad)

GUI Development

Reactivity can come from event responses:

```
buttonUI :: Window -> UI ()
buttonUI window = do
  button <- UI.button #+ [string "Click me"]
  getBody window #+ [return button]

  on UI.click button $ \_ -> do
    getBody window #+ [ UI.div #+ [ string "You clicked me!"] ]
```

GUI Development

There is a JavaScript FFI, and JQuery integration

```
import qualified Graphics.UI.Threepenny as UI
import Graphics.UI.Threepenny.Core

import Graphics.UI.Threepenny.JQuery

main :: IO ()
main = startGUI defaultConfig setup

setup :: Window -> UI ()
setup w = do
   return w # set title "fadeIn - fadeOut"
   button <- UI.button # set text "Click me to make me \
                                   \fade out and in!"
   getBody w #+ [column [UI.string "Demonstration of \
                                \jQuery’s animate() function"
                          ,element button]]
    on UI.click button $ \_ -> do
        fadeOut button 400 Swing $ do
            runUI w $ fadeIn button 400 Swing $ return ()
```

GUI Development

- There are facilities for graphics (most easily via the canvas), and animation (for example via UI.timer)
- State can be managed either:
  - Explicitly through the IO monad
  - Using “Reactive.Threepenny”

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

End of part 1

Glenn.Strong@scss.tcd.ie
https://scss.tcd.ie/Glenn.Strong/

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Thank you

glenn.Strong@scss.tcd.ie
https://scss.tcd.ie/Glenn.Strong/