Grades.HS

Abraham Hinteregger, BSc

Vienna University of Technology

30.06.2014

Gewünschte Features

- Notenstatistik
- ohne Excel
- verschiedene Notensysteme (at,ch,us, . . .)

Umsetzung

- Typ für Fach und Note mit verschiedenen Untertypen
- Zugriff mittels lens Package
- State entspricht einer Liste aus Fächern
- Typ für verschiedene Manipulationen von State
- Typ für verschiedene Arten der Schnittberechnung

applyAction

```
-- | Applies an action to a list of subjects and
     returns the modified list.
applyAction :: LAction -> [LSubject]-> [LSubject]
applyAction (RemRes i) s= s & ix i . result .~ Nothing
applyAction (AddRes i r) s= s & ix i . result .~Just r
applyAction (AddSub n) s= s ++ [n]
applyAction _ s= s
```

GUI & State

- GUI mit Threepenny-GUI
- State mit IORef
  - Durchreichen der Referenz und des Fensters
  - Bei Änderungen IOModifyRef
  - Manipulation des Fensters (nicht inkrementell sondern immer vollständig

main :: IO()

```
main :: IO()
main = do
startGUI config (setup state)

setup :: [Maybe LSubject] -> Window -> UI ()
setup s w = void $ do
    io   <- liftIO $ newIORef (catMaybes s)
    view <- mkView (w,io) s
    getBody w UI.# set UI.children [view]
```

Positiv

- Keine Probleme mit verschiedenen Package Versionen o.Ä.
- Debuggen mit :t und :info ist recht angenehm
- Recht einfach nachzuvollziehen wo etwas schiefgeht.

Negativ

- Dokumentation von Threepenny-GUI und FRP etwas dürftig
- Instanzierung von `Read a` extrem mühsam
- An Fehlerbehandlung gescheitert

Sonstiges

- GUI- Code bläht sich recht schnell auf (möglicherweise meinem Stil geschuldet)
- gigantische Binaries