Lec 07 - Elm Svg Graphics

CS 1XA3

Feb 27, 2018

Record Types

- Unique data structure to Elm (not in Haskell). Similar to structures used in Javascript
- See http://elm-lang.org/docs/records for a comprehensive overview of them
- Example

```
type alias Pos = {x : Int,y : Int}

-- assign a literal value
pos = {x = 5, y = 6}
-- update by element
pos2 = {pos | x = pos.x+1 }
```

Elm - SVG Graphics

- Html5 allows embedding of svg graphics directly:
  See
  https://www.w3schools.com/graphics/svg_inhtml.asp
- Just like Html, Elm provides functions corresponding to Svg
  See
  http://package.elm-lang.org/packages/elm-lang/svg/2.0.0/Svg
- Need to install in your working directory with

```
elm package install elm-lang/svg
```

Svg Graphics Elements

- Common Svg Elements include:

```
circle : List (Attribute msg) -> List (Svg msg)
                              -> Svg msg
rect : List (Attribute msg) -> List (Svg msg)
                            -> Svg msg
text : String -> Svg msg
```

- These functions return Svg msg, convert to Html msg with

```
svg : List (Attribute msg) -> List (Svg msg)
                           -> Html msg
```

Embedding Svg in Html

In order to render an Svg element in our view function, we need to convert to Html using the svg function

Example

```
import Svg exposing (..)
import Svg.Attributes exposing (..)

view : Model -> Html Msg
view model =
   svg [width "600",height "600"]
       [circle [cx "300",cy "300", r "50", fill "red"] []]
```

AnimationFrame

- You can use Subscriptions that pass a time value to render animations
- The Time module in elm-lang/core is one option, however it will render shaky
- Use AnimationFrame for Subscriptions that sync with the browsers natural rendering rate
- Need to install in your working directory with

  ```
  elm package install elm-lang/animationframe
  ```

AnimationFrame Subscriptions

- AnimationFrame provides two subscriptions, each in lockstep to the browsers natural rendering speed

  ```
  times : (Time -> msg) -> Sub msg
  -- current time

  diffs : (Time -> msg) -> Sub msg
  -- time diffs between animation frames
  ```

- They are parameterized using Time

  ```
  type alias Time = Float
  ```

Example: Animating a Circle

```
import AnimationFrame as Anim

type alias Model = { pos: { x : Int
                          , y : Int } }
type Msg = Tick Float

subscriptions model = Anim.times Tick

update (Tick time) model =
-- Use time to change model.pos

view model =
-- render circle with model.pos
```