PureScript & Halogen

Vladimir Ciobanu

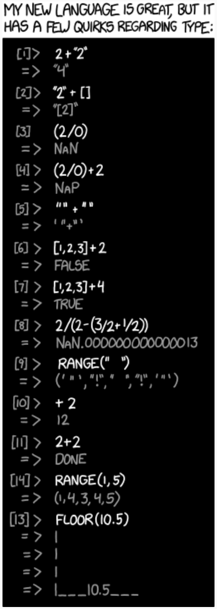
Tuesday, May 8, 2018

Development Lead, Visual Solutions, Mood Media Romania

Overview

Why Not...

Why Not JavaScript?


MY NEW LANGUAGE IS GREAT, BUT IT HAS A FEW QUIRKS REGARDING TYPE:

```
[1]> 2+"2"
 => "4"
[2]> "2"+[]
 => "[2]"
[3]  (2/0)
 => NaN
[4]> (2/0)+2
 => NaP
[5]> "" + ""
 => '"+"'
[6]> [1,2,3]+2
 => FALSE
[7]> [1,2,3]+4
 => TRUE
[8]> 2/(2-(3/2+1/2))
 => NaN.0000000000000013
[9]> RANGE(" ")
 => ('"',"!"," ","!",'"')
[10]> +2
 => 12
[11]> 2+2
 => DONE
[14]> RANGE(1,5)
 => (1,4,3,4,5)
[13]> FLOOR(10.5)
 => |
 => |
 => |
 => |___10.5___
```

- very loose language (coercions, quirks, etc)
- easy to hack something together, hard to keep it sane
- practically inexistent type system
- npm is a mess
- frequent breaking changes

What About TypeScript?

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.

- fails to fix a lot of problems (coercions, quirks are still there)
- type definitions aren’t trivial to keep in sync
- still missing a lot of advanced features (sum types, type classes, dependent types, etc)

Introduction to PureScript

What is PureScript?

PureScript is a strongly-typed functional programming language that compiles to JavaScript.

- Compile to readable JavaScript
- Strong FFI / interoperability with JavaScript
- No runtime (unlike Elm, GHCJS, etc)
- Very good tooling
- Great community
- High-quality libraries

Syntax - Functions

```
addOne :: Int → Int
addOne x = x + 1

head :: List ⇝ Maybe
head = case _ of
  Nil      → Nothing
  Cons x _ → Just x

even :: Int → Boolean
not  :: Boolean → Boolean

notEven :: Int → Boolean
notEven = not <<< even

headNotEven :: List Int → Maybe Boolean
headNotEven = map notEven <<< head
```

JavaScript Output 1/2

Head.purs

```
head :: List ⇝ Maybe
head = case _ of
    Nil      → Nothing
    Cons x _ → Just x
```

output.js

```
var head = function (v) {
    if (v instanceof Data_List_Types.Nil) {
        return Data_Maybe.Nothing.value;
    };
    if (v instanceof Data_List_Types.Cons) {
        return new Data_Maybe.Just(v.value0);
    };
    throw new Error("Failed pattern match at Main..");
};
```

JavaScript Output 2/2

HeadNotEven.purs

```
headNotEven :: List Int → Maybe Boolean
headNotEven = map notEven <<< head
```

output.js

```
var headNotEven = function ($5) {
    return Data_Functor.map
        (Data_Maybe.functorMaybe)(notEven)(head($5));
};
```

Syntax - Records

```
type Person r =
  { name :: String
  , age  :: Int
  | r
  }

isOlderThan :: ∀ r1 r2. Person r1 → Person r2 → Boolean
isOlderThan p1 p2 = p1.age > p2.age

isOlderThan’ :: ∀ r1 r2
              . { age :: Int | r1 }
             → { age :: Int | r2 }
             → Boolean
isOlderThan’ p1 p2 = p1.age > p2.age
```

Effects

```
main :: Eff _ Unit
main = log "Hello world"

getElementById :: String → Eff _ (Maybe Element)
getElementById s
  = (   querySelector (QuerySelector s)
    <<< htmlDocumentToParentNode
    <=< document
    ) =<< window

window :: Eff _ Window
document :: Window -> Eff _ HTMLDocument
htmlDocumentToParentNode :: HTMLDocument → ParentNode
querySelector :: QuerySelector
              → ParentNode
              → Eff _ (Maybe Element)
```

Foreign Function Interface

ParentNode.js

```
exports._querySelector = function (selector) {
  return function (node) {
    return function () {
      return node.querySelector(selector);
}; }; };
```

ParentNode.purs

```
foreign import _querySelector :: QuerySelector
                              → ParentNode
                              → Eff _ (Nullable Element)

querySelector :: QuerySelector
              → ParentNode
              → Eff _ (Maybe Element)
querySelector qs = map toMaybe <<< _querySelector qs
```

My PureScript Workflow

Tooling Demo

- pulp init
- pursuit
- bower install
- vscode

Halogen

Basic Component

```
data Query a = DoNothing a
type Input = Unit; type Message = Void; type State = Unit

component :: ∀ m. H.Component HH.HTML Query Input Message m
component = H.component
  { initialState: id
  , render
  , eval
  , receiver: const Nothing
  }
  where

  render :: State → H.ComponentHTML Query
  render _ = HH.text "Hello, world"

  eval :: Query ⇝ H.ComponentDSL State Query Message m
  eval (DoNothing next) = pure next
```

Render

```
render :: State → H.ComponentHTML Query
render st =
  HH.div
    [ HP.class_ (H.ClassName "messages__middle") ]
    [ HH.div
      [ HP.class_ (H.ClassName "messages") ]
      $ (map (renderMessage st.playing) st.items)
      <> guard st.isSchedule
      [ HH.button
          [ HE.onClick <<< HE.input_ $ Commit
          , HP.class_ <<< H.ClassName $ "messages-btn"
          , HP.disabled $ not st.isDirty
          ]
          [ HH.text "Save" ]
      ]
    ]
```

Eval

```
eval :: Query ~> H.Component HH.HTML Query Input Message MsgM
eval = case _ of
  Initialize next → next <$ do
    H.fork do
      env ← unMessagesEnvironment <$> H.lift ask
      tags → H.lift <<< liftServer $ getTags
      isSchedule ← H.lift $ _.isSchedule <$> get
      case tags of
        Left err → H.lift <<< navigate $ BadSetup
        Right t  → H.put $ Just
          { tags: t
          , selectedTag: SchedOrDmd isSchedule <<< unwrap $ t
          , localeService: env.localeService
          , isSchedule: isSchedule
          }
```

Conclusion

Resources

- **Christopher Allen**, The Haskell Book http://haskellbook.com
- **Phil Freeman**, PureScript by Example http://leanpub.com/pursecript
- **Pursuit**, PureScript Documentation http://pursuit.purescript.org
- **Vladimir Ciobanu**, Halogen Example http://github.com/vladciobanu/purescript-halogen-example
- **Functional Programming Slack** https://fpchat-invite.herokuapp.com/
- **PureScript Discourse** http://purescript-users.ml

We Are Hiring

MOOD: is the global leader for Experience Design services.

The **Visual Solutions** is in charge of our *Digital Signage* product.

We are migrating key components to *Haskell* and *PureScript*.

Are you:

- passionate about writing software?
- interested in learning and use functional programming?

If you answered **yes** to both questions, then you should join our team!

No prior experience with FP is required.

Thank You

Thank you for listening!

cvlad

vladciobanu

cvlad.info