A Haskell Platform for Creating Progressive Web Applications

A midterm progress report

Jakub Zárybnický

2019-01-29

Outline

Assignment

- My task:
  - *"A Haskell Platform for Creating Progressive Web Applications"*

Assignment

- My task:
  - *"A Haskell Platform for Creating Progressive Web Applications"*
- What's a PWA?
  - new buzzword from Google
  - an *almost* native app
    - load a website
    - save to phone homepage like an app
    - available offline, perhaps with data sync
    - use push notifications, device APIs

Assignment

- My task:
  - *"A Haskell Platform for Creating Progressive Web Applications"*
- What's a PWA?
  - new buzzword from Google
  - an *almost* native app
    - load a website
    - save to phone homepage like an app
    - available offline, perhaps with data sync
    - use push notifications, device APIs
- What's a Haskell?
  - purely functional programming language with type inference and lazy evaluation
  - *"If it compiles, it works."*

Motivation

- The trend of FP on the frontend, motivated by Elm, PureScript
- Sharing common code between the client and the server
- Rapidly growing area in the Haskell ecosystem, many companies involved, OSS contributors:
  - IOHK (Cardano) - iohk-ops, distributed processing
  - Obsidian - Reflex, Obelisk, Rhyolite
  - Tweag - Asterius, Inline-js
  - QFPL - Reflex-workshop, UI components
- Growing, but still not established - missing tools and libraries
- More immediate need - I have clients with projects that are waiting on this thesis' results

What have I been doing?

- How have I been working?
  - in part research, type and API design
  - in part extracting relevant parts from my existing Haskell applications

What have I been doing?

- How have I been working?
  - in part research, type and API design
  - in part extracting relevant parts from my existing Haskell applications
- What worked out?
  - survey of web frameworks all around the programming world
  - survey of the Haskell ecosystem around web development
  - prototype of a full-stack application
  - prototype of a JAM-stack-alike
  - prototype of an offline-capable client application
  - prototype of a frontend debugger toolbar (quite limited so far)

What haven't I been doing?

What haven't I been doing?

- What didn't work out?
  - attempts at a blog with regular updates - a failure this far
  - a mile long to-do list and a backlog of ideas and research topics

What haven't I been doing?

- What didn't work out?
  - attempts at a blog with regular updates - a failure this far
  - a mile long to-do list and a backlog of ideas and research topics
- Non-goals:
  - write the thesis itself
  - write publishable code

Technologies

The main ones:

- Haskell
  - *"Haskell is a purely functional programming language with type inference and lazy evaluation."* – Wikipedia
  - *"If it compiles, it works."*
- Nix
  - *"Nix package manager, a "purely functional" package and configuration manager for computer systems"* – Wikipedia
  - *"Describe your end result, then magic happens"* ☺
  - Tools build on top:
    - Nix = package manager
    - NixOS = operating system
    - NixOps = cloud deployment tool

Haskell example (Servant)

```haskell
type HackageAPI =
  "users" :> Get '[JSON] [User] :<|>
  "user" :> Capture "login" Login :> Get '[JSON] User :<|>
  "packages" :> Get '[JSON] [Package]

getUsers :: Handler [User]
getUser :: Login -> Handler User
getPackages :: Handler [Package]

server :: Server HackageApi
server = getUsers :<|> getUser :<|> getPackages

getUsers :<|> getUser :<|> getPackages =
  client @HackageApi "http://hackage.haskell.org"
```

NixOps example

```
{
  network.description = "Web server";

  webserver = { config, pkgs, ... }: {
    services.httpd.enable = true;
    services.httpd.adminAddr = "alice@example.org";
    services.httpd.documentRoot =
      "${pkgs.valgrind.doc}/share/doc/valgrind/html";
    networking.firewall.allowedTCPPorts = [ 80 ];

    deployment.targetEnv = "virtualbox";
  };
}
```

Next tasks

Wrapping up unfinished tasks:

- finish article drafts and publish them
- finish extracting useful patterns from my applications

Starting work on new areas:

- ServiceWorker wrapper or template
- push notifications
- pre-rendering (build- or runtime)
- CLI tool
- type design for data channel/synchronization

Finishing up

- four more months until the publication deadline
- tons of work left, mile-long lists of tasks and ideas
- basics are well underway
- many stretch goals