A Haskell Platform for Creating Progressive Web Applications

A midterm progress report

Jakub Zárybnický

2019-01-29

Full-stack Haskell?

- Why Haskell? Why on the frontend?

Full-stack Haskell?

- Why Haskell? Why on the frontend?
  - Compile-to-JavaScript languages accepted (see Elm)
  - Code sharing between server and client (business logic, entities)

Full-stack Haskell?

- Why Haskell? Why on the frontend?
  - Compile-to-JavaScript languages accepted (see Elm)
  - Code sharing between server and client (business logic, entities)
  - Many companies use it, and also contribute to OSS:
    - Obsidian - Reflex, Obelisk, Rhyolite
    - Tweag - Asterius, Inline-js
    - QFPL - Reflex-workshop, many UI components
    - IOHK (Cardano) - iohk-ops

PWA?

- New buzzword from Google
- Recent web development trend
- An *almost* native app
  - Load a website
  - Click "Add to homepage"
  - Use like a native application

PWA features

- *Progressive Web App Checklist* – Google

PWA features

- *Progressive Web App Checklist* – Google
- Required:
  - Responsive and fast ("Native experience")
  - Available offline
  - Push notifications
  - Device APIs

PWA features

- *Progressive Web App Checklist* – Google
- Required:
  - Responsive and fast ("Native experience")
  - Available offline
  - Push notifications
  - Device APIs
- Optional:
  - Full functionality when offline
  - Data sync
  - Pre-rendered

PWA...

- Technologies:
  - Service Workers
  - Web App Manifest
  - History API
  - Web Share API
  - Network Information API
  - Credential Management API

What have I been doing?

- Research:
  - PWA developers' expectations (across languages)
  - Current Haskell libraries and tooling

What have I been doing?

- Research:
  - PWA developers' expectations (across languages)
  - Current Haskell libraries and tooling
- Prototypes:
  - A full-stack application
  - An offline-capable application
  - A frontend debugger toolbar
  - CI/CD for Haskell

What haven't I been doing?

What haven't I been doing?

- Regular writing
  - Blog
  - Progress reports

What haven't I been doing?

- Regular writing
  - Blog
  - Progress reports
    - Oops...

What's next?

- Must-haves:
  - Push notifications (server & client library)
  - Browser routing (client library)
  - Service Worker wrapper (build tool, client library)

What's next?

- Must-haves:
  - Push notifications (server & client library)
  - Browser routing (client library)
  - Service Worker wrapper (build tool, client library)
- Nice-to-haves:
  - Pre-rendering (server library)
  - Data sync (server & client library)

Wrapping up

- Thanks for listening!
- `https://github.com/zarybnicky/thesis`