Getting stuff done in Haste

A brief introduction to web development
with Haskell

Anton Ekblad
antonek@chalmers.se

What we'll do today

- Write a Pong clone with Haste
  - See web development in action
  - Use simple 2D graphics
  - Practice asking questions
- Talk about research on Haste
  - Practice asking even more questions

What is Haste?

- Two major parts
  - A Haskell compiler targeting JavaScript
  - A set of Haskell web development libraries
- By your powers combined a Haskell dialect for the web!

Fun things to do with Haste

- 2D graphics with Canvas
- 3D graphics with WebGL
- More powerful GUI with jQuery, React, etc.
- Easy client/server programs
- Add a feature to the libraries or compiler

Web development basics

- Web pages are trees of elements
- Elements have
  - Properties
  - Style attributes
  - Children
  - Event handlers
- HTML + CSS is the usual starting point

And now for something completely different

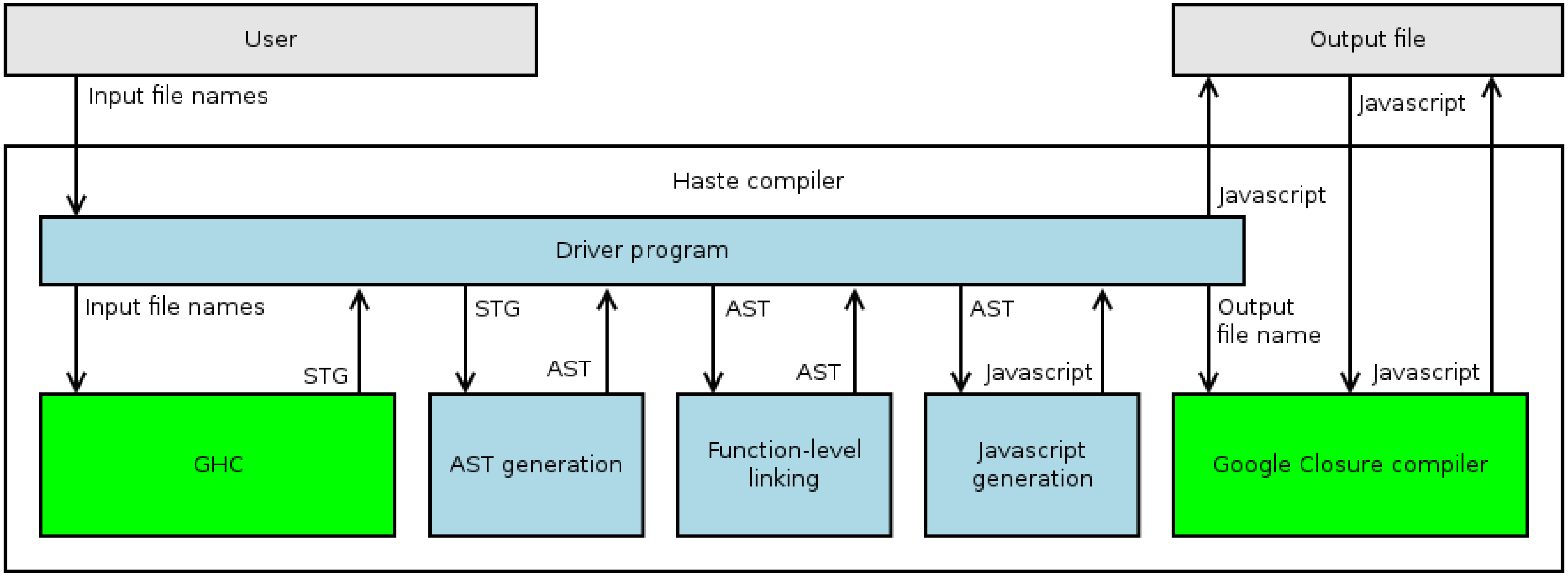
Why Haste?

- Web development is the future
- ...but it is a JavaScript monoculture!
- What to do if JS doesn't fit your problem?
  - Symbolic computations
  - Concurrent programs
  - Large, complex systems
  - Domain-specific languages

Does anyone actually use this?

- Education
  - FP intro course, this course
- Research tool
  - Interest in using Haste for doing NLP, Agda, etc. in the browser
  - Enhancing security in web languages
- Industry
  - In use at several small companies

How does it work?



Program slicing

- Webb apps traditionally built in layers
  - Often different languages
  - Hampers code reuse
  - Error-prone communication
- Can we do away with that?

Interfacing with the outside world

- Traditionally, very low level
  - C-level interop
  - Restricted to primitive types (`int`, `char`, etc.)
  - Unsuitable for high-level targets like JS
- Can we do away with that?

Information flow control

- Web apps use *lots* of third party code!
  - jQuery, Google Analytics, Angular, etc.
  - Pulled in from `$DEITY` knows where
  - May leak user data to just about anywhere
- Can we do something about that?

Compiling to crazy architectures

- JS is very different from a traditional CPU
    - High level features like closures and GC
    - No pointers/raw memory access
    - No arbitrary jumps
- How do we deal with that?

Read more

- http //haste-lang.org
  - Video tutorials
  - Documentation
  - Useful libraries
  - Publications
  - Source code, mailing list, IRC channel