English

WebAssembly becoming the biggest platform

Sven Sauleau

2019

Sven Sauleau @svensauleau



Igalia's compiler team.

The Web (previously)


Java Setup - Progress
Java
ORACLE
Status: Registering Java Runtime Environment
3 Billion Devices Run Java
Computers, Printers, Routers, Cell Phones, BlackBerry,
Kindle, Parking Meters, Public Transportation Passes, ATMs,
Credit Cards, Home Security Systems, Cable Boxes, TVs...
ORACLE

Java: “write once, run anywhere”

Desktop apps.

Web pages.

Android.

Smart Card (SIM, credit card, . . .).

The Web (currently)

Side note[1]

Java

May 23, 1995

23 years ago

JavaScript

December 4, 1995

23 years ago

[1] source: Wikipedia

JavaScript became mainstream on the web

All good,
but suddenly...

The <blink> tag
stopped working.

JavaScript, what happened?

Loading time

Fetching.

Parsing source.

Compiling + optimizing $\xrightarrow{\infty}$ reoptimizing.

Performance

Dynamic and untyped.

Complex runtime.

Managed memory.

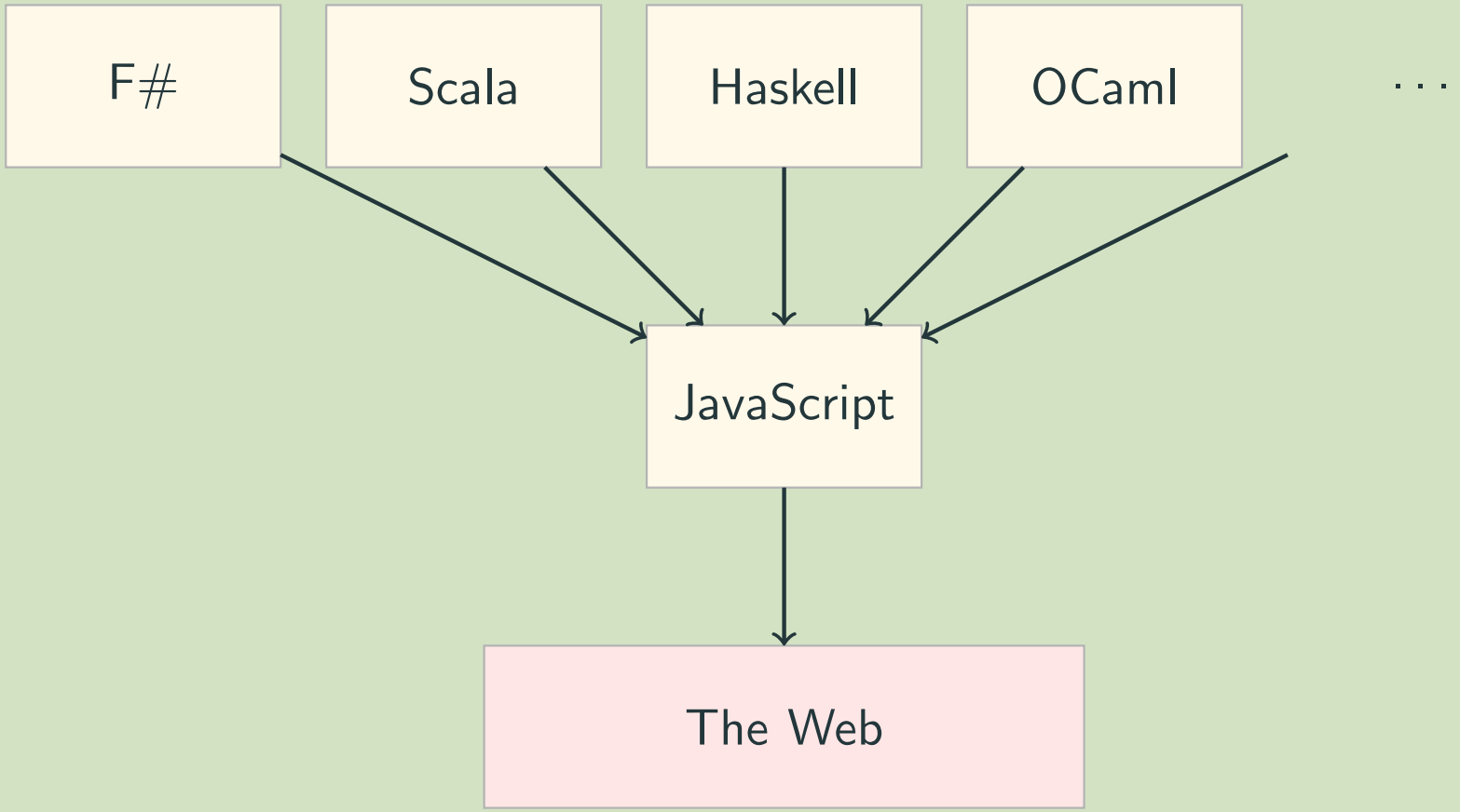
A few optimizations

Static analysis is difficult.

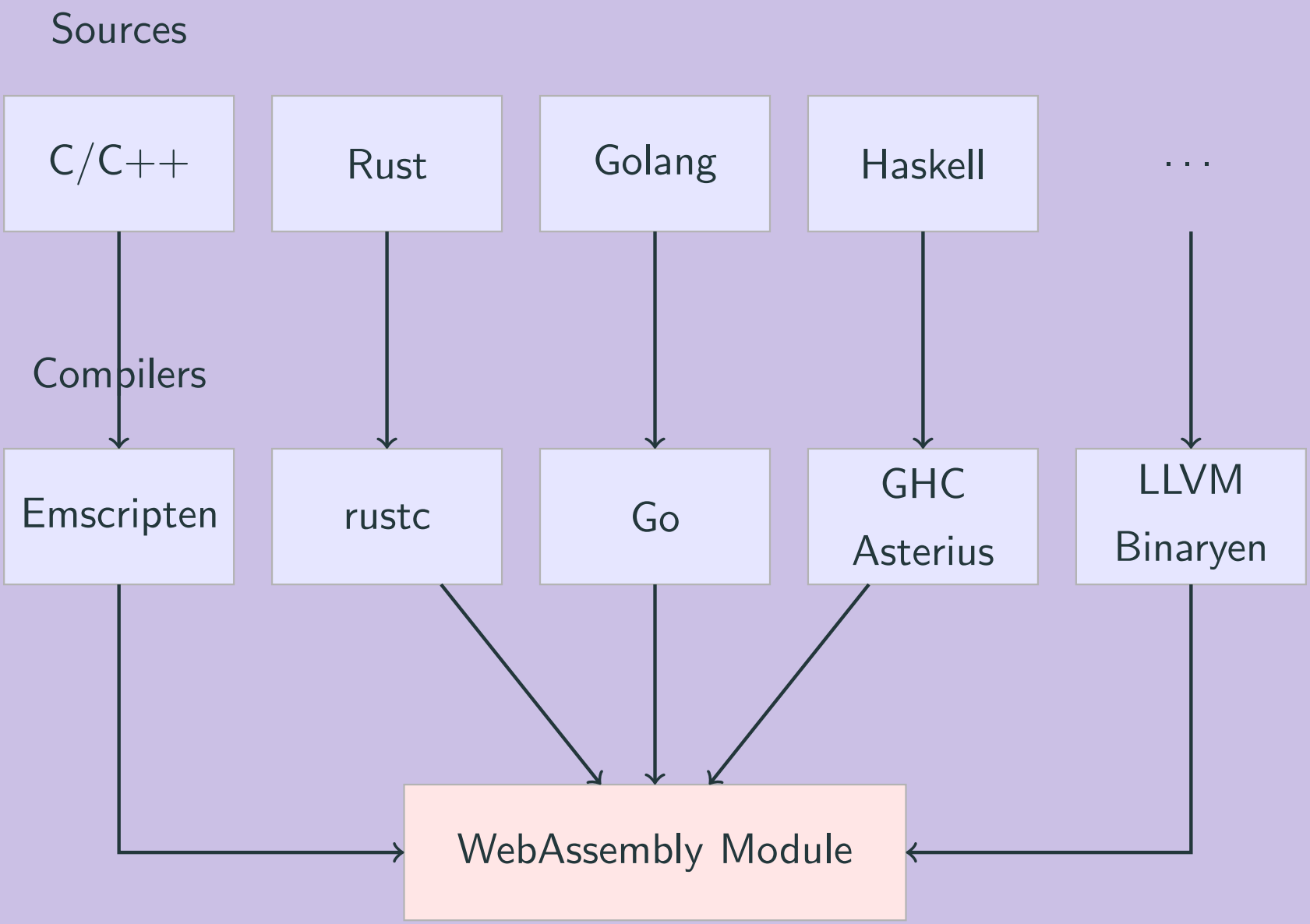
Minification.

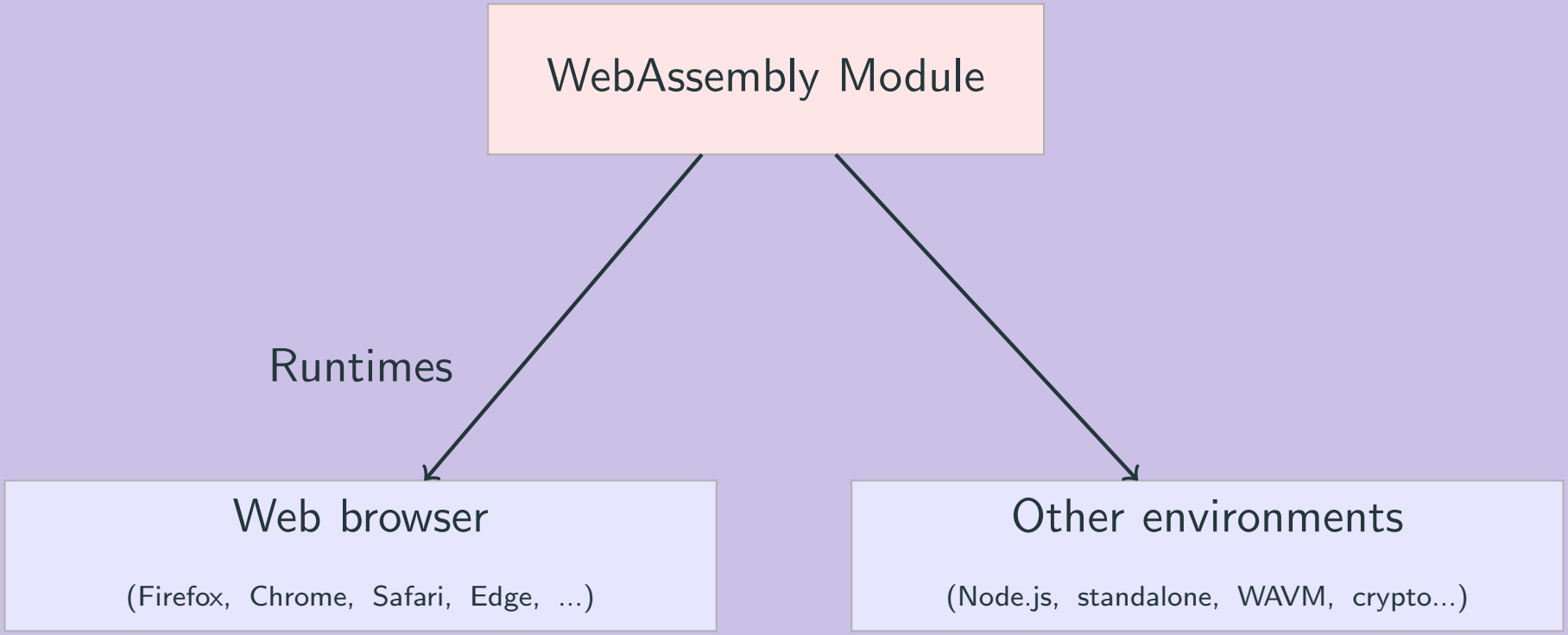
Tree shaking.

Became a compilation target


F#
Scala
Haskell
OCaml
· · ·
JavaScript
The Web

WebAssembly, at the rescue?


Sources
C/C++
Rust
Golang
Haskell
· · ·
Compilers
Emscripten
rustc
Go
GHC
Asterius
LLVM
Binaryen
WebAssembly Module


WebAssembly Module
Runtimes
Web browser
(Firefox, Chrome, Safari, Edge, ...)
Other environments
(Node.js, standalone, WAVM, crypto...)

WebAssembly is a

safe,

portable,

low-level

format.

Replace JavaScript with Wasm?

No!

JavaScript

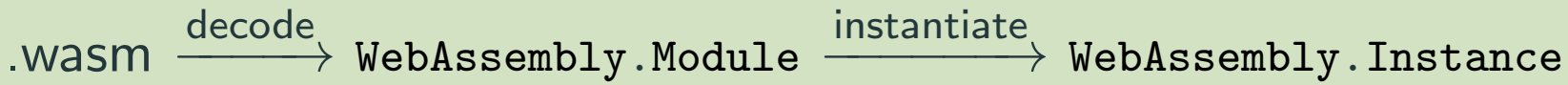
Simple.

Accessible.

Easy to Debug and Test.

WebAssembly is designed to be a complement to, not replacement of, JavaScript.

Steps


.wasm
decode
WebAssembly.Module
instantiate
WebAssembly.Instance

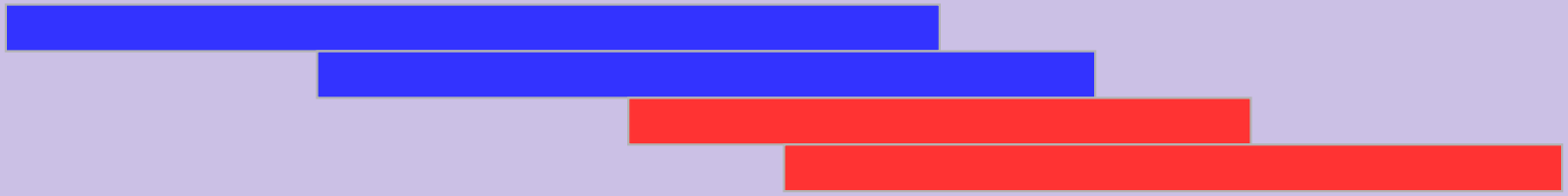
Efficient representation

Compact and easy to decode.

Streamable and parallelizable.

"Streamable"

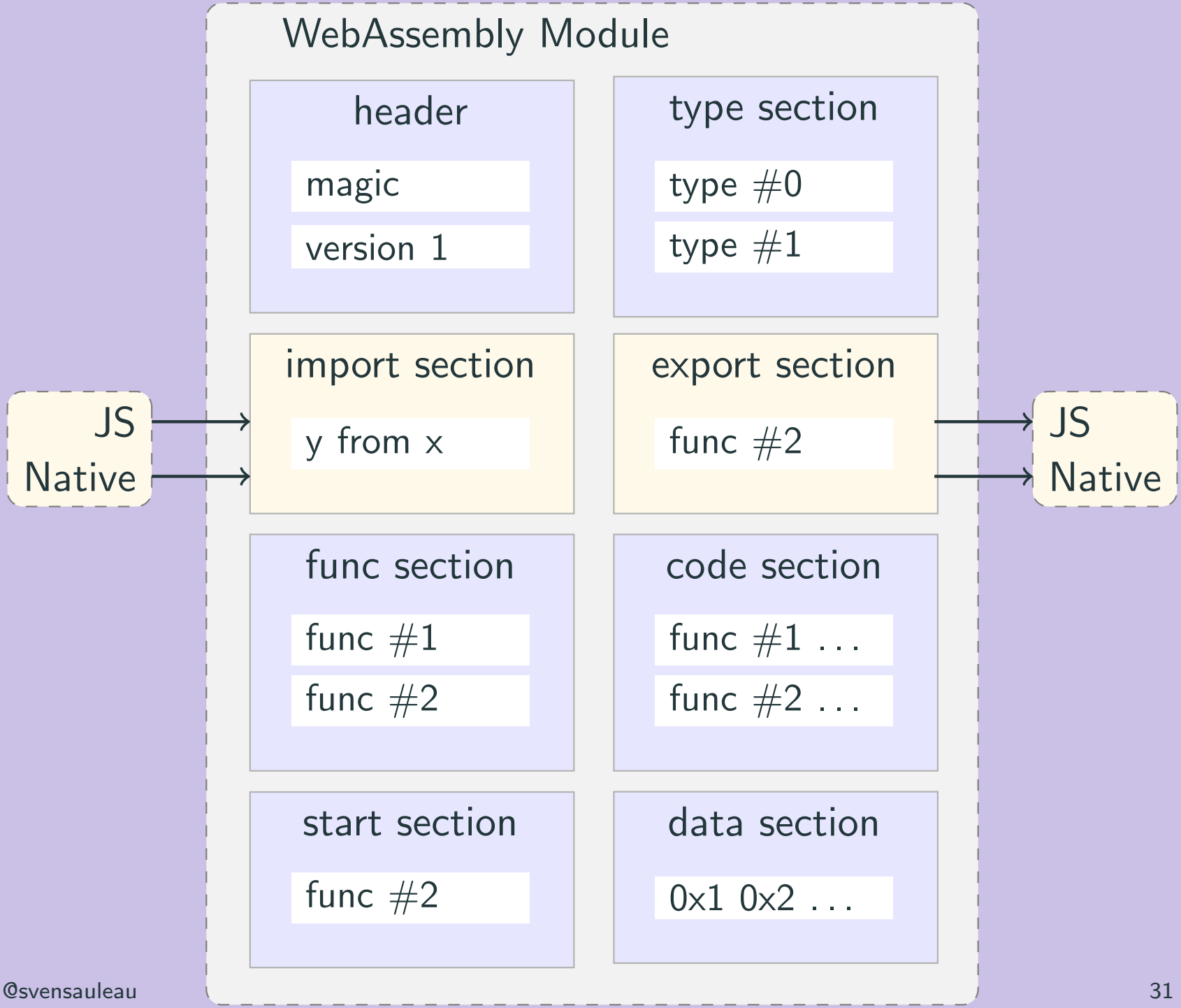
Download → decode → instantiate → compile.



.wasm

00 61 73 6d 01 00 00 00 01 07 01 60
02 7f 7f 01 7f 03 02 01 00 07 0a 01
06 61 64 64 54 77 6f 00 00 0a 09 01
07 00 20 00 20 01 6a 0b 00 19 04 6e
61 6d 65 01 09 01 00 06 61 64 64 54
77 6f 02 07 01 00 02 00 00 01 00 ...


WebAssembly Module
header
magic
version 1
type section
type #0
type #1
func section
func #1
func #2
code section
func #1 ...
func #2 ...
start section
func #2
data section
0x1 0x2 ...


WebAssembly Module
header
magic
version 1
type section
type #0
type #1
import section
y from x
export section
func #2
JS
Native
JS
Native
func section
func #1
func #2
code section
func #1 ...
func #2 ...
start section
func #2
data section
0x1 0x2 ...

```
1  const binary = ...;
2
3  const module = WebAssembly.Module(binary);
4  const instance = WebAssembly.Instance(
5    binary, importObject);
6
7  instance.exports.somefunc();
```

It's a virtual machine

Register-based
(x86, . . .)

```
1 mov %eax, 0x1
2 mov %rax, 0x1
3 add %eax, %rax
```

VS

Stack-based
(WebAssembly, call stack, . . .)

```
1 i32.const 1
2 i32.const 1
3 i32.add
```

Textual representation

module.wast:

```
1 (module
2   (func (export "addTwo") (param i32 i32) (result i32)
3     (get_local 0)
4     (get_local 1)
5     (i32.add)
6   )
7 )
```

Performance

WebAssembly is fast

Compiled to machine code.

Static analysis.

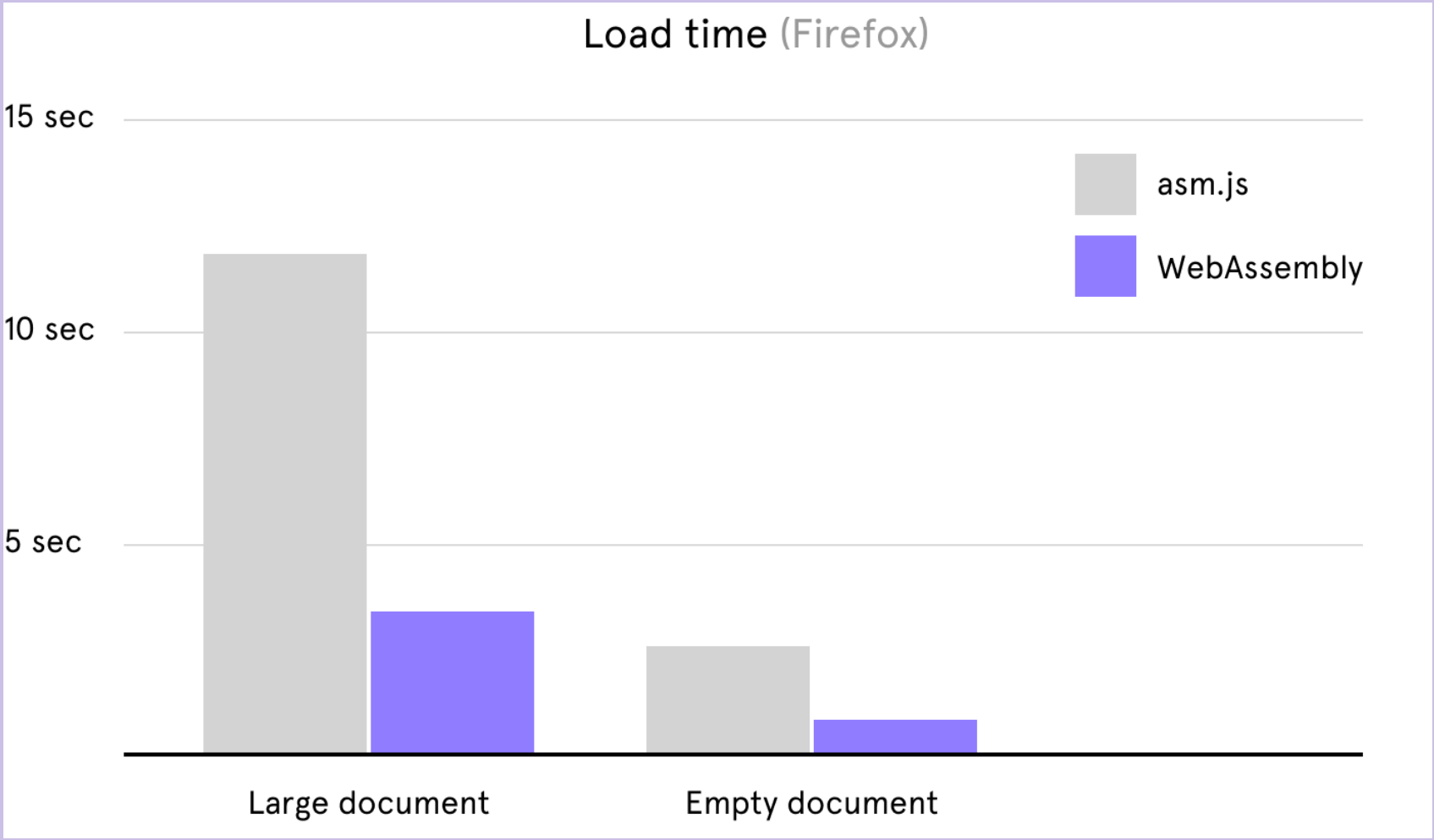
Optimized Ahead Of Time.

Crossing the boundary[2]

Can require a conversion.

Can require checks.

[2] https://hacks.mozilla.org/2018/10/calls-between-javascript-and-webassembly-are-finally-fast-%F0%9F%8E%89/



"Our load time improved by more than 3x [...]"

— Figma, medium

How to use it?

Usage

Likely:

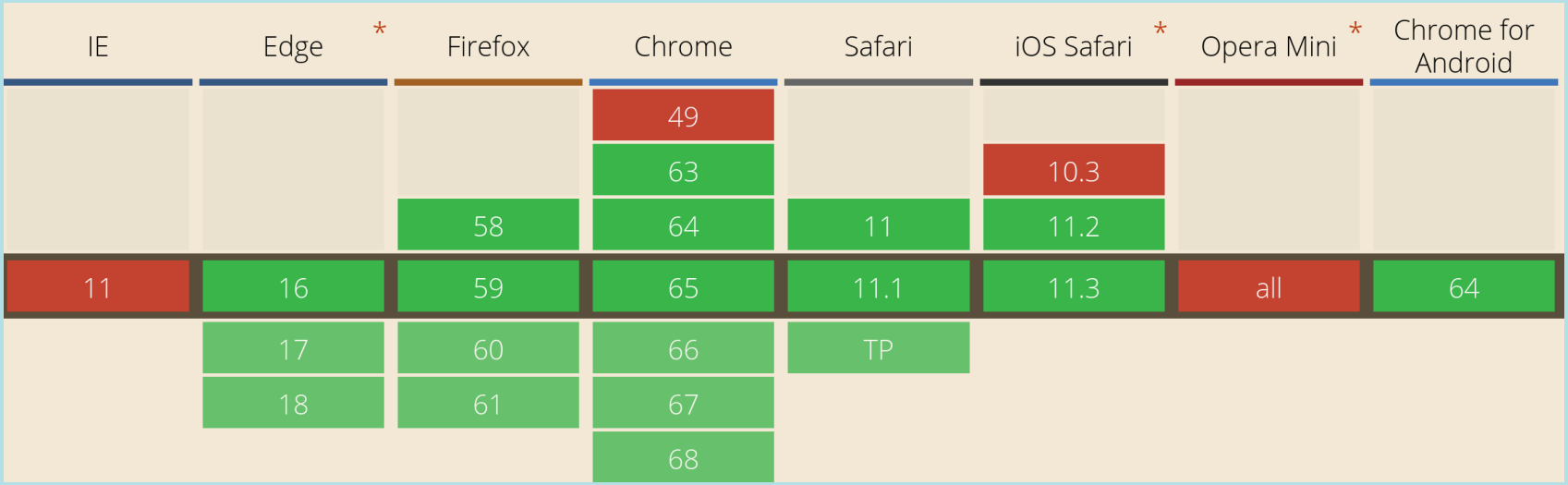
```
$ compiler -x --target=wasm file
```

Languages [3]

- .Net
- Astro
- Brainfuck
- C / C# / C++
- Elixir
- Faust
- Forest
- Forth
- Haskell
- Golang
- Java
- Kotlin/Native
- Kou
- Lua
- OCaml
- Plorth
- Rust
- Turboscript
- Wah
- Wracket
- Xlang

[3] https://github.com/appcypher/awesome-wasm-langs

Browser support



Browser (no) support

WebAssembly → JS compiler

WebAssembly/binaryen

WebAssembly interpreter written in JavaScript

(WIP) xtuc/webassemblyjs

Other usages

Example Cloudflare

Run code on edge on each request.

Links: main.c, blog + demo

Crypto

"ewasm is a restricted subset of WASM to be used for contracts in Ethereum."

— ewasm

Unity and Unreal Engine use WebAssembly

Links: Funky Karts, AngryBots

What does it mean for JavaScript?

ES Module Integration [4]

Import JS modules and values from Wasm.

Export JS module from Wasm.

[4] https://github.com/WebAssembly/esm-integration

with Webpack

module.c

```
1 #include <strings.h>
2 #include <webassembly.h>
3
4 EXPORT void test() {
5     console_log("Hi");
6 }
```

index.js

```
1 import("./module.c")
2   .then(({test}) => {
3       test();
4   });
```

JS-like languages

AssemblyScript: TypeScript → WebAssembly compiler [5]

```
1 export function add(a: i32, b: i32): i32 {
2   return a + b;
3 }
```

[5] AssemblyScript.org

Work-In-Progress

Wasm BigInt [6][7]

WebAssembly:

```
(module
  (func (export "fn") (param i64) (result i64)
    (get_local 0)
  )
)
```

JavaScript:

```
exports.fn(42n) === 42n
```

[6] https://sauleau.com/notes/wasm-bigint.html
[7] https://github.com/WebAssembly/JS-BigInt-integration

Host Bindings [8]

Integration with the host

Import Web APIs.

Manipulate JavaScript + DOM objects.

[8] https://github.com/webassembly/host-bindings

Garbage collection [9]

- Data structures
- Reference types
- Support more languages

[9] https://github.com/webassembly/gc

Threading [10]

- Native threads
- Shared memory
- Atomics

[10] https://github.com/webassembly/threads

Demo

Thanks