WebAssembly becoming the biggest platform

Sven Sauleau

2018

Sven Sauleau

Sven Sauleau

@svensauleau

"The Web is already the biggest platform and WebAssembly will expand it to rest of the world."

— Sven Sauleau, 5min ago

An issue with JavaScript ?

Sorry JavaScript

It's not a good compilation target.

WebAssembly has different use cases[1]

[1]VPN, databases, games, platform emulation, VM, ...

WebAssembly

- Deterministic and easier to reason about
- Designed to be Safe to execute

WebAssembly

"[...] memory is disjoint from code space, the execution stack, and the engine's data structures; therefore compiled programs cannot corrupt their execution environment, jump to arbitrary locations, or perform other undefined behavior"

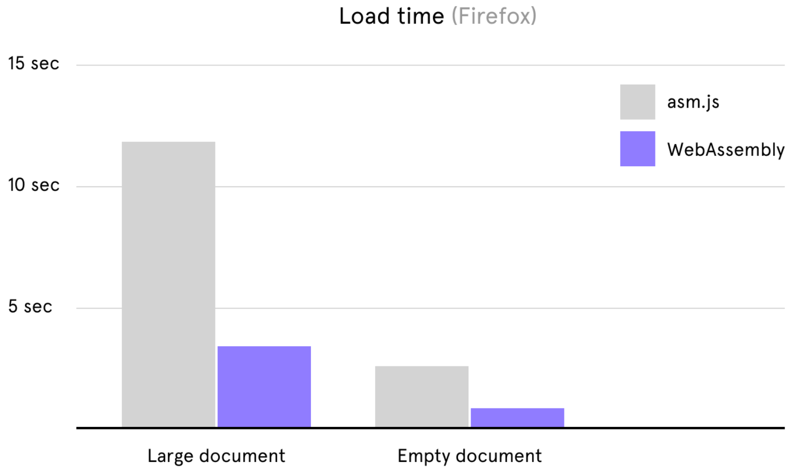
— Bringing the Web up to Speed with WebAssembly

Executes fast

- Near native speed (and JIT soon)
- Compilers optimize the code Ahead Of Time

Note: interop with JS can be slow

Compact and easy to decode



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

— Figma, medium

Safe and efficient representation

- Compact and easy to decode
- Streamable and parallelizable

How to use it?

Usage

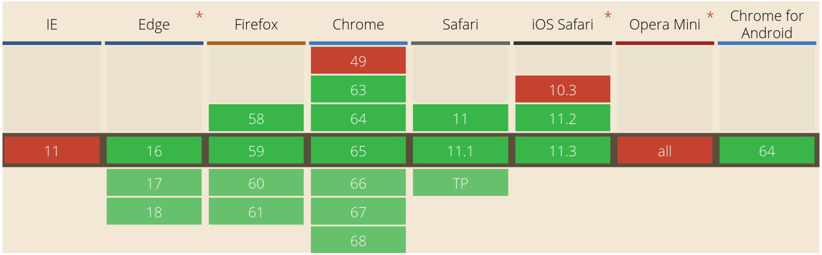
```
$ my-dopi-compiler --target=wasm32 file
```

Languages [2]

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

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

Browser support



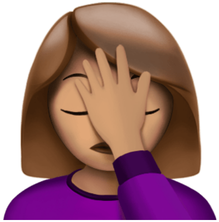
IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android
			🟥 49				
			🟩 63		🟥 10.3		
		🟩 58	🟩 64	🟩 11	🟩 11.2		
🟥 11	🟩 16	🟩 59	🟩 65	🟩 11.1	🟩 11.3	🟥 all	🟩 64
	🟩 17	🟩 60	🟩 66	🟩 TP			
	🟩 18	🟩 61	🟩 67				
			🟩 68				

Is it going to kill JavaScript ?

No.

- WebAssembly will not replace JavaScript
- Other languages will target WebAssembly instead

Will we compile JavaScript to WebAssembly?



Please no.

- JavaScript is very dynamic
- it lacks of static typing
- Lots of check at runtime

AssemblyScript: A TypeScript to WebAssembly compiler [3]

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

[3] AssemblyScript.org

Walt: is an alternative syntax for WebAssembly [4]

```
export function fibonacci(n: i32): i32 {
  if (n <= 0) return 0;
  if (n == 1) return 1;

  return fibonacci(n - 1) + fibonacci(n - 2);
}
```

[4] https://github.com/ballercat/walt

The incoming parts

Builtin garbage collection

- You can ship your own
- Will track JavaScript refs as well as WebAssembly

Threads

Native threads with concurrency primitives [5].

[5] Locks, Atomics, etc.

Direct access to browser APIs

But you can import any JavaScript function

Write to the console

```
const importObject = {
  env: {
    log: msg => console.log(msg),
  }
};
```

WASM or WAST

WASM: WebAssembly Binary Format [6]

[6] .wasm

WAST: WebAssembly "Script" Format [7]

[7] .wast

Tooling

Toolchain for WebAssembly (in JavaScript)[8]:

- WAST/WASM parser/printer
- Interpreter
- AST introspection/manipulation
- WAST/WASM optimizations

[8] https://github.com/xtuc/webassemblyjs

DCE in Webpack

```
bin = editWithAST(ast, bin, {

    ModuleExport(path) {
        const usedName = module.isUsed(name);

        if (!usedName) {
            path.remove();
        }
    }

});
```

Webpack integration demo

Slides on

reacteu-2018.ralf.cc

Pictures by Ben Heine (Pencil Vs Camera)