Cloudinary

Supercharge your JavaScript with Wasm

Tamas Piros

Cloudinary

Hi 👋 I am

Tamas Piros

Developer Evangelist
Cloudinary

Director
Full Stack Training

Google Developer Expert
Web Technologies

@tpiros

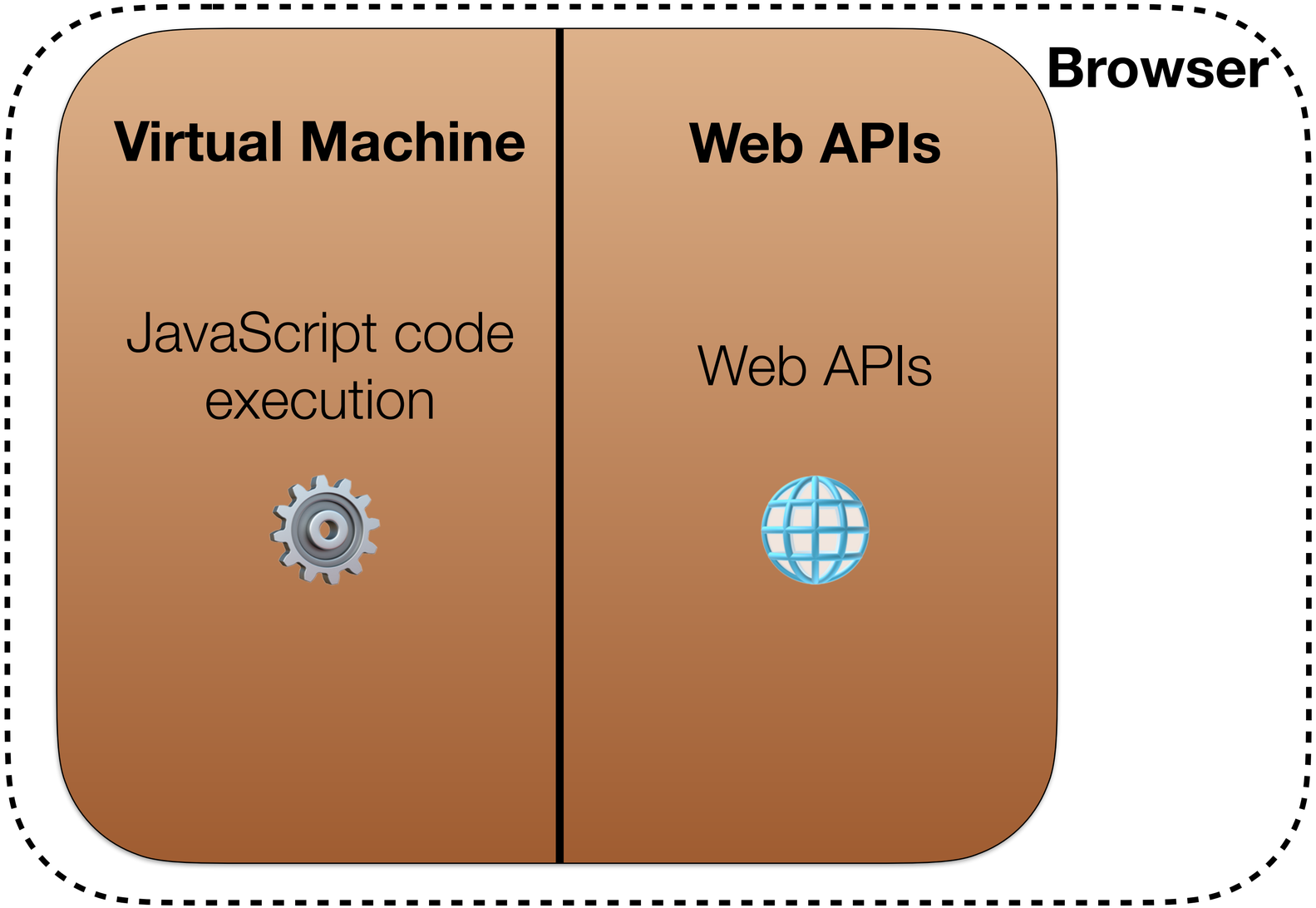
Cloudinary

Why do programmers leave their job?

Because they don’t get a raise

a raise = arrays 😉

Web Platform (as of 2018)



The Web is progressing at an incredible pace

JavaScript is great for leveraging the ecosystem

But has its limits

Cloudinary

It’s very difficult to achieve low-level tasks without a performance impact

Cloudinary

WebAssembly was created in 2015*

* asm.js predates WebAssembly (2013) - allowed apps written in C to run as web apps

Cloudinary

Since 2019 (Dec) WebAssembly is a W3C recommendation

WebAssembly is a low-level assembly-like language with a compact binary format that runs with near-native performance and provides languages such as C/C++ and Rust with a compilation target so that they can run on the web.

https://developer.mozilla.org/en-US/docs/WebAssembly

Cloudinary

Run native apps on the web

Cloudinary

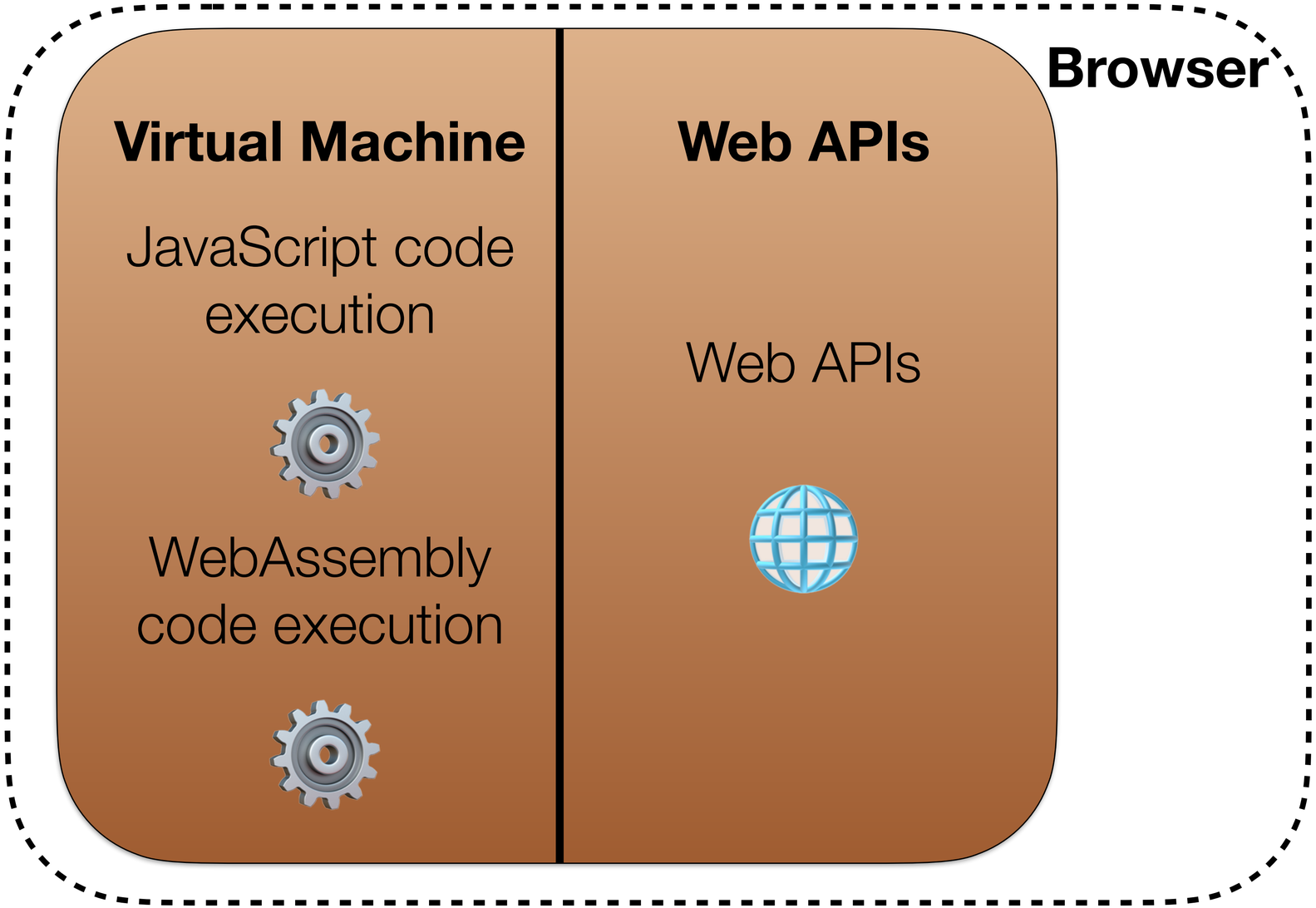
WebAssembly functions can be exposed to JavaScript

Cloudinary

WebAssembly is not here to replace JavaScript.

It’s here to enhance / augment it.

Web Platform (today)



WebAssembly JavaScript API

Loading module (compiled WebAssembly binary)

Create new memory and table instances

Instance: Module + Memory & Table - just like an ES2015 module

Process of creating wasm

Write code in C or C++ (or any other LLVM supported languages)

Use Emscripten or use direct compile targets to produce .wasm

Load & consume via JavaScript

Cloudinary

Process of creating wasm

Write code using .NET languages, Java, Ruby or Go

Compile to .wasm

Load & consume via JavaScript

Languages that compile to .wasm

.Net, C, C++, C#, D, F#, Go, Java, PHP, Python, TypeScript

… and a lot more https://github.com/appcypher/awesome-wasm-langs

Cloudinary

Demo Time 🥳

Resources

- Emscripten (https://emscripten.org)
- MDN WebAssembly (https://developer.mozilla.org/en-US/docs/WebAssembly)
- Sample Repository (https://github.com/tpiros/wasm-samples)
- Wasm by example (https://wasmbyexample.dev)
- Running Doom via wasm (https://wasm.continuation-labs.com/d3demo/)
- Super Marion via wasm (https://medium.com/@bokuweb17/writing-an-nes-emulator-with-rust-and-webassembly-d64de101c49d)
- Squoosh.app (https://squoosh.app)
  - Case study: https://developers.google.com/web/updates/2019/02/hotpath-with-wasm

Cloudinary

Thank you

@tpiros