New Topics

NEW TECHNOLOGY
Web Assembly (WASM)

2024. 01. 18

Sungwon Lee

Department of Software Convergence

경희대학교
KYUNG HEE UNIVERSITY

WebAssembly

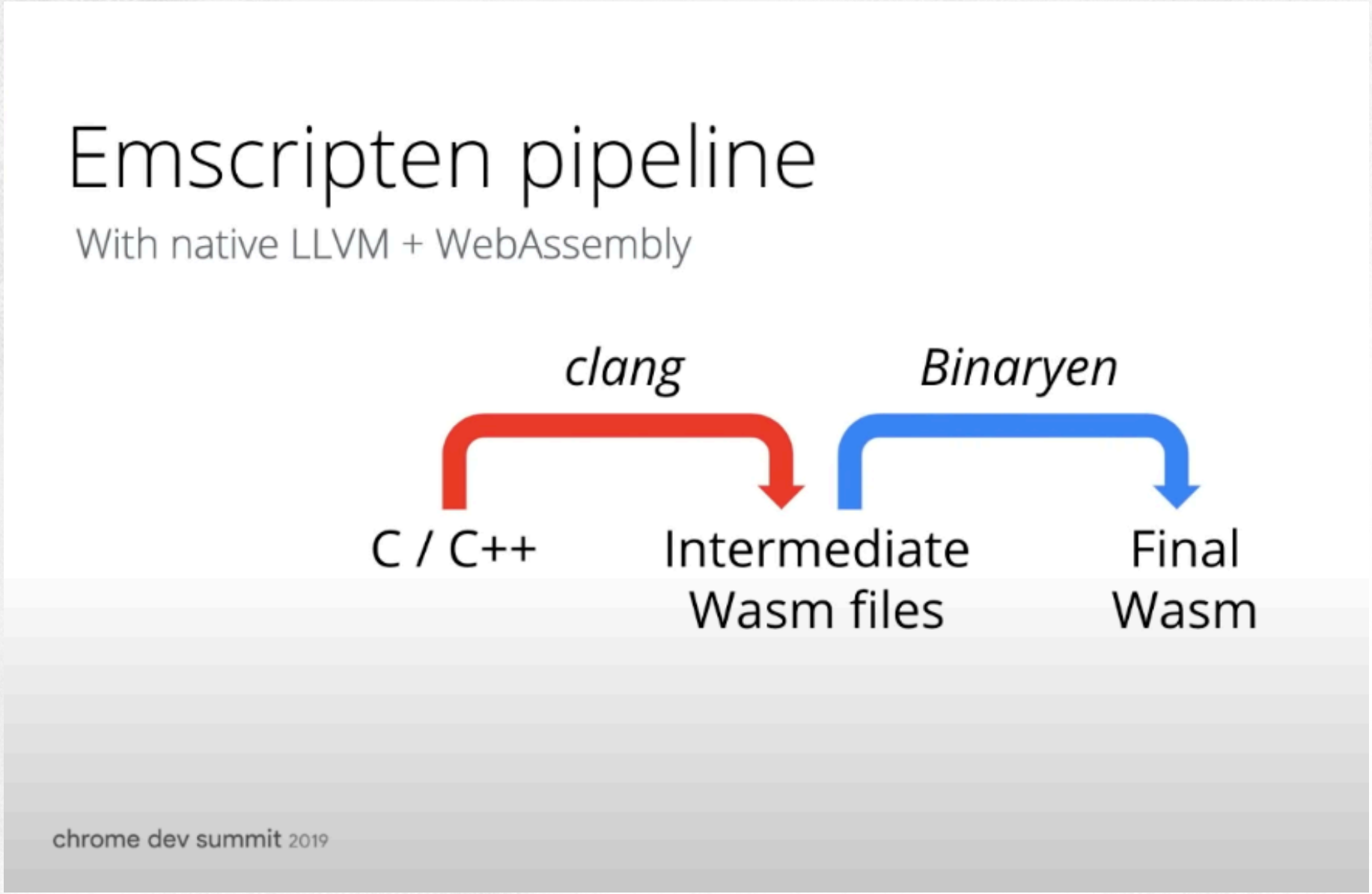
- WebAssembly is a type of code that can be run in modern web browsers — it is a low-level assembly-like language with a compact binary format that runs with near-native performance and provides languages such as C/C++, C# and Rust with a compilation target so that they can run on the web.
- It is also designed to run alongside JavaScript, allowing both to work together.

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

High Level Language to WASM

- C/C++ to WASM
  - https://developer.mozilla.org/en-US/docs/WebAssembly/C_to_wasm
- RUST to WASM
  - https://developer.mozilla.org/en-US/docs/WebAssembly/Rust_to_wasm
- Dart/Flutter to WASM
  - https://docs.flutter.dev/platform-integration/web/wasm

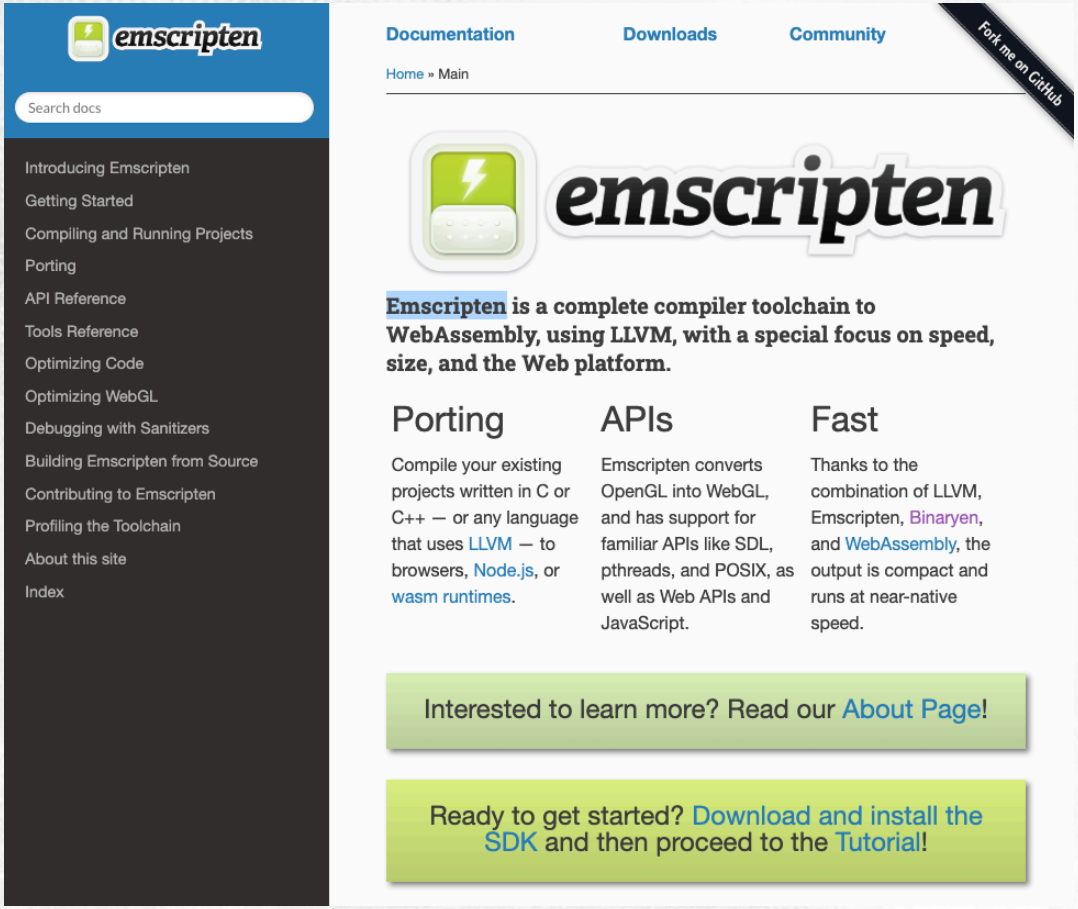
How it works?



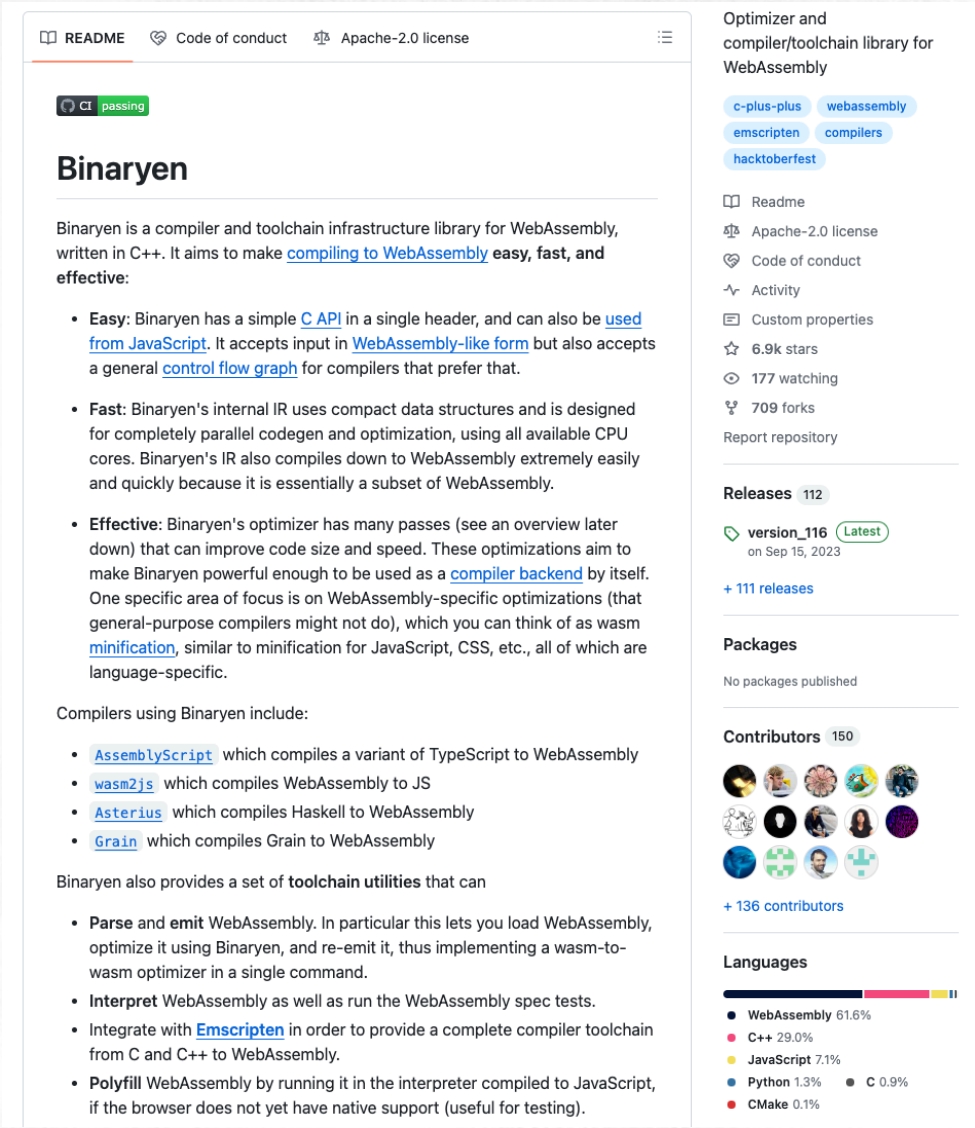
REFERENCE: https://www.youtube.com/watch?v=kZrl91SPSpc&t=1012s

Bird' Eye View

High Level Language to WASM

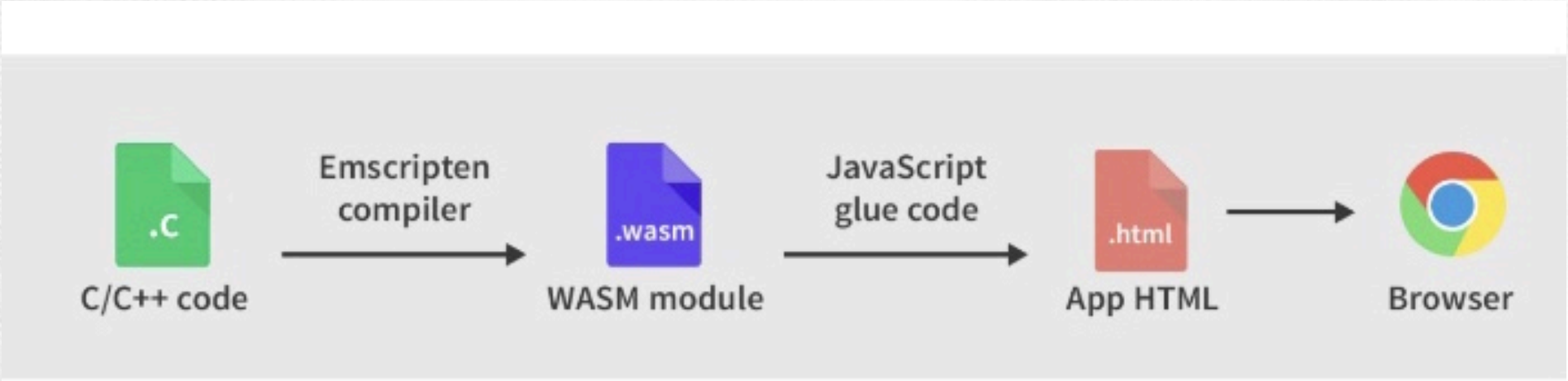


https://emscripten.org/



https://github.com/WebAssembly/binaryen

How it works with conventional Web techs?

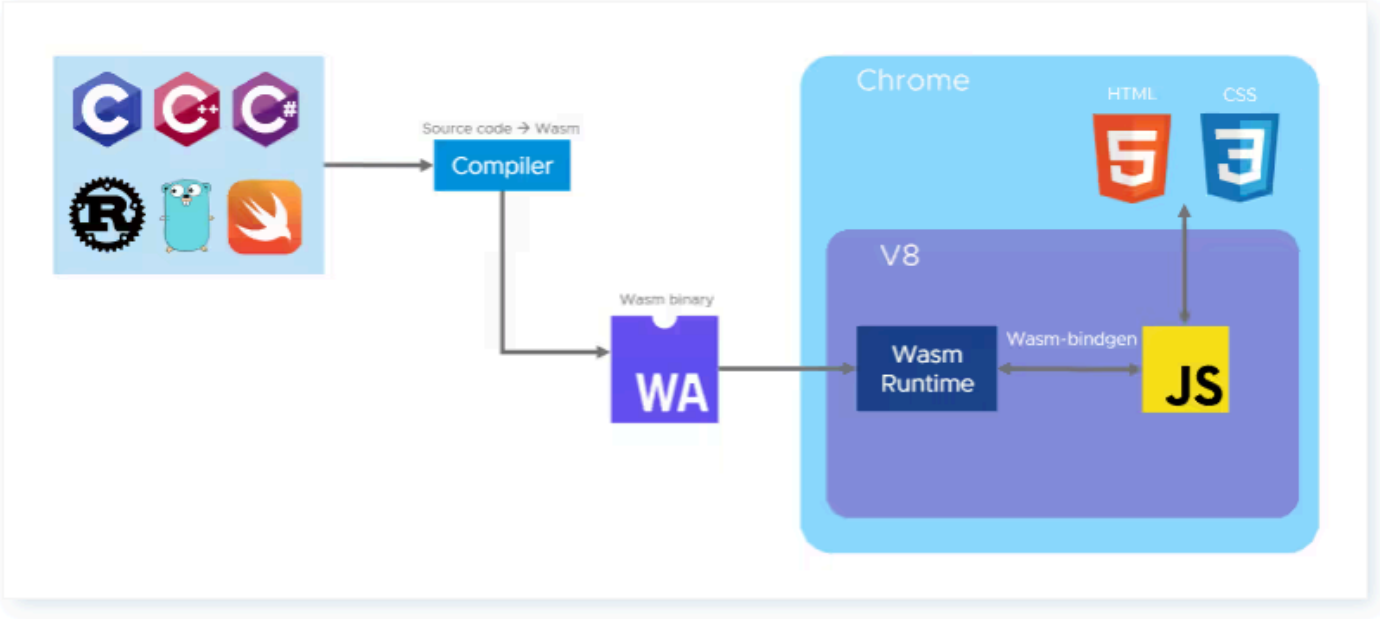


REFERENCE: https://jaehoon-daddy.tistory.com/27

How it works with conventional Web techs?

How does Wasm work in browsers?

Browser engines integrate a Wasm virtual machine, usually called a Wasm runtime, which can run the Wasm binary instructions. There are compiler toolchains (like Emscripten) that can compile source code to the Wasm target. This allows for legacy applications to be ported to a browser and directly communicate with the JS code that runs in client-side Web applications.



These technologies have allowed traditional desktop apps to run in a browser. And now they can run on any device on which you have a browser. Some notable examples are Google Earth and the Open CV library for computer vision.

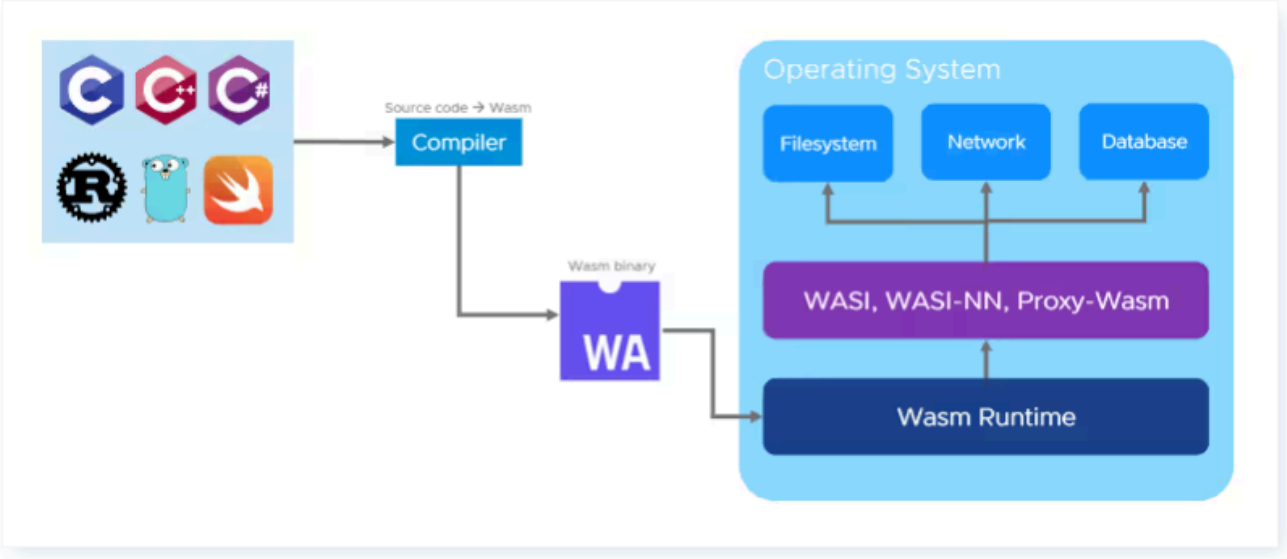
REFERENCE: https://wasmlabs.dev/articles/docker-without-containers/

What's going on?

WASI : Evolution of WASM !!

How does Wasm work on servers?

There are Wasm runtimes that can run outside of the browser, including traditional operating systems such as Linux, Windows and macOS. Because they cannot rely on a JavaScript engine being available they communicate with the outside world using different interfaces, such as WASI, the WebAssembly System Interface. These runtimes allow Wasm applications to interact with their host system in a similar (but not quite the same) way as POSIX. Projects like WASI SDK and wasi-libc help people compile existing POSIX-compliant applications to WebAssembly.

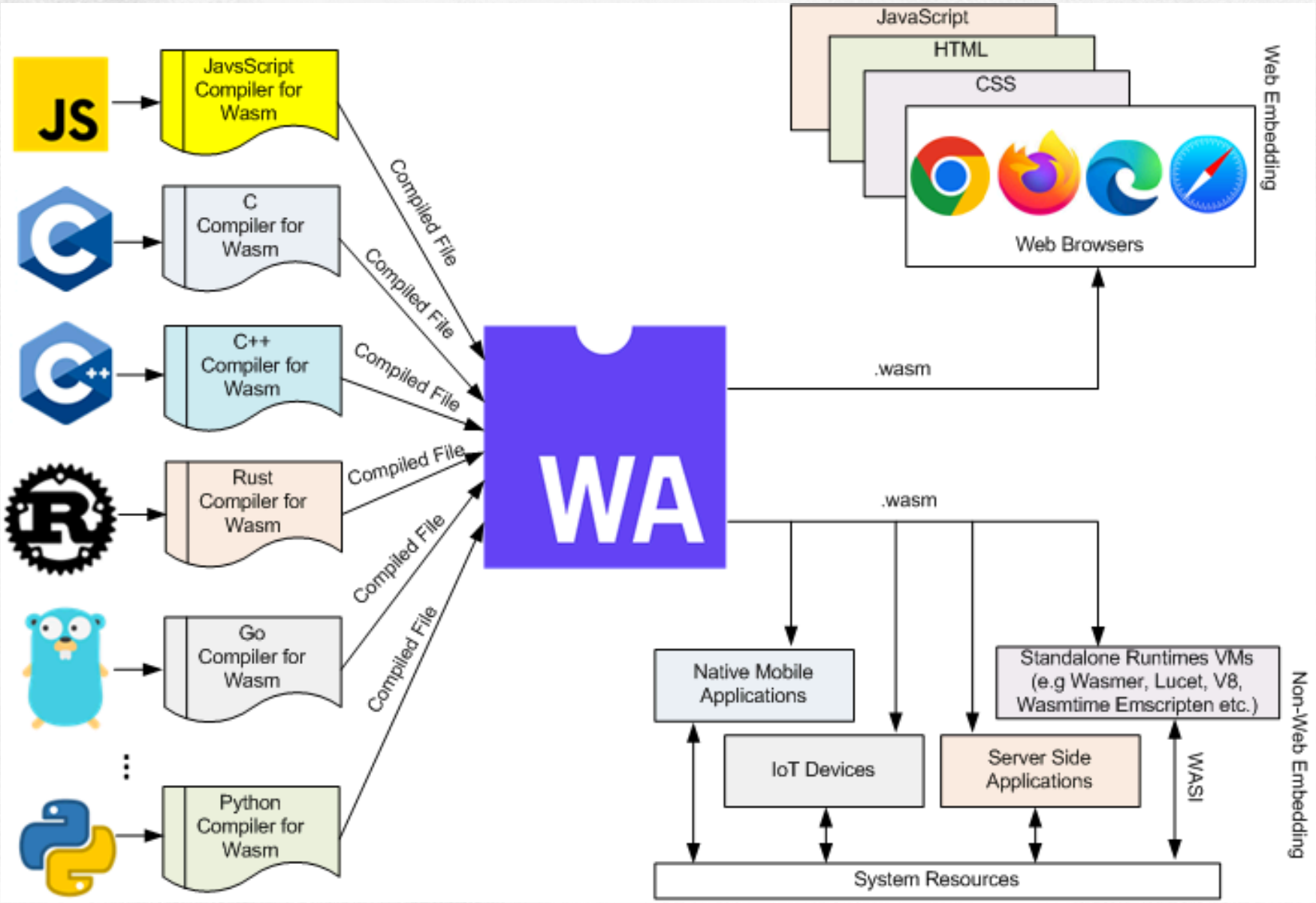


You only need to compile an application into a Wasm module once, and then you can run the exact same binary everywhere.

REFERENCE: https://wasmlabs.dev/articles/docker-without-containers/

What's going on?

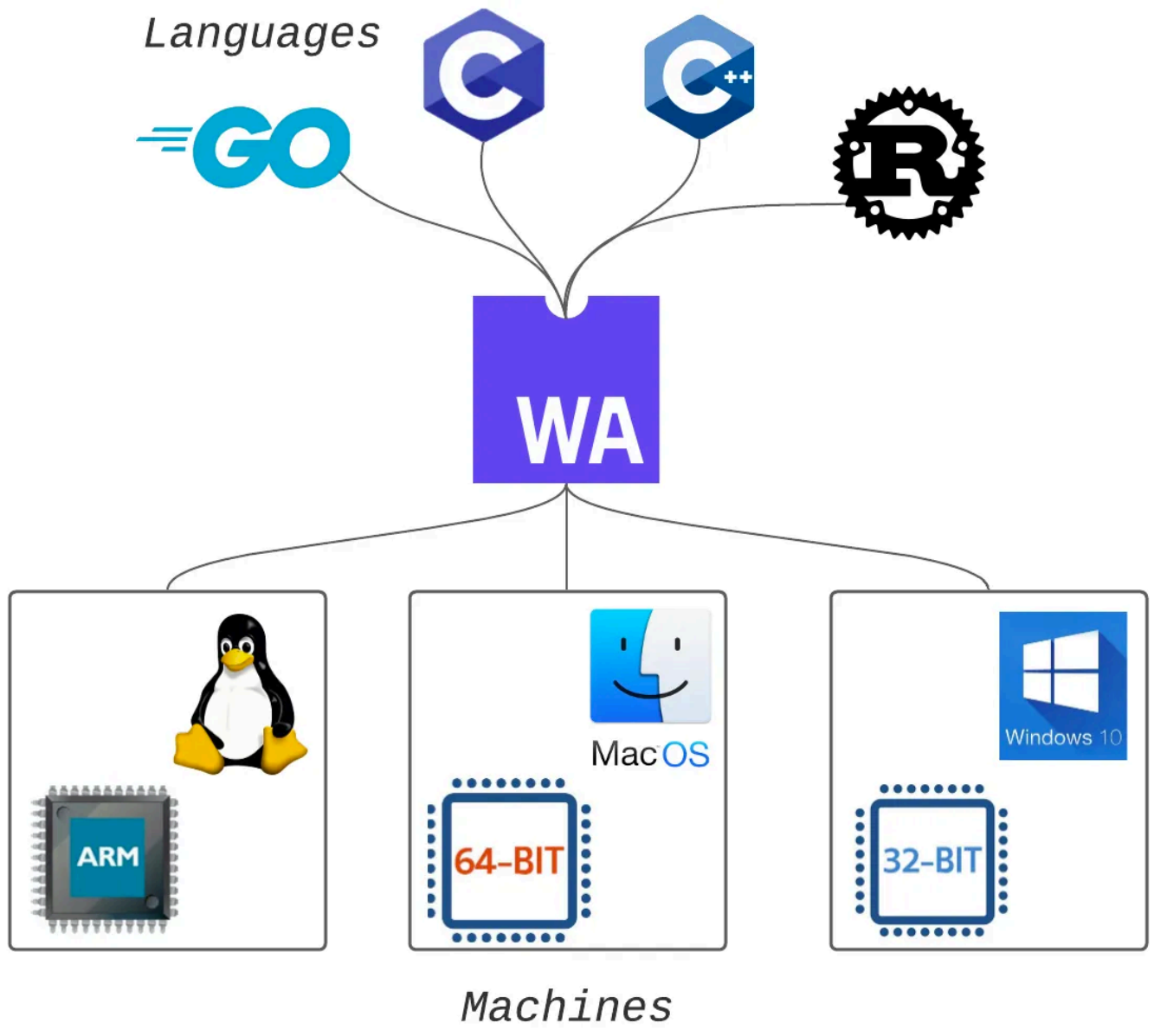
WASM/WASI for Every Domain !!



REFERENCE: https://www.researchgate.net/figure/WebAssembly-data-flow-architecture_fig1_373229823

What's going on?

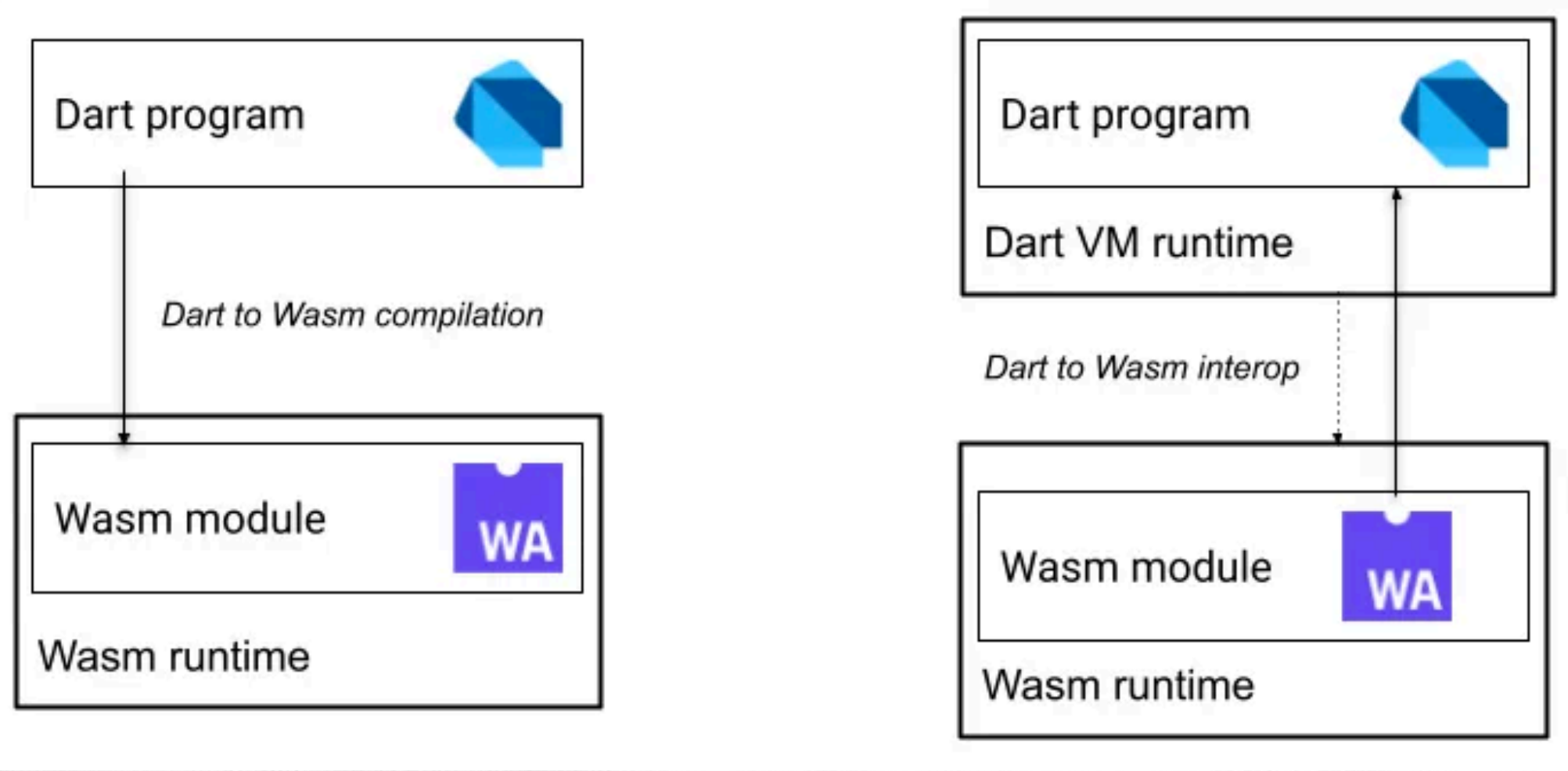
WASM/WASI for Every Technology !!



REFERENCE: https://cloudconstructed.substack.com/p/webassembly-a-primer

What's going on?

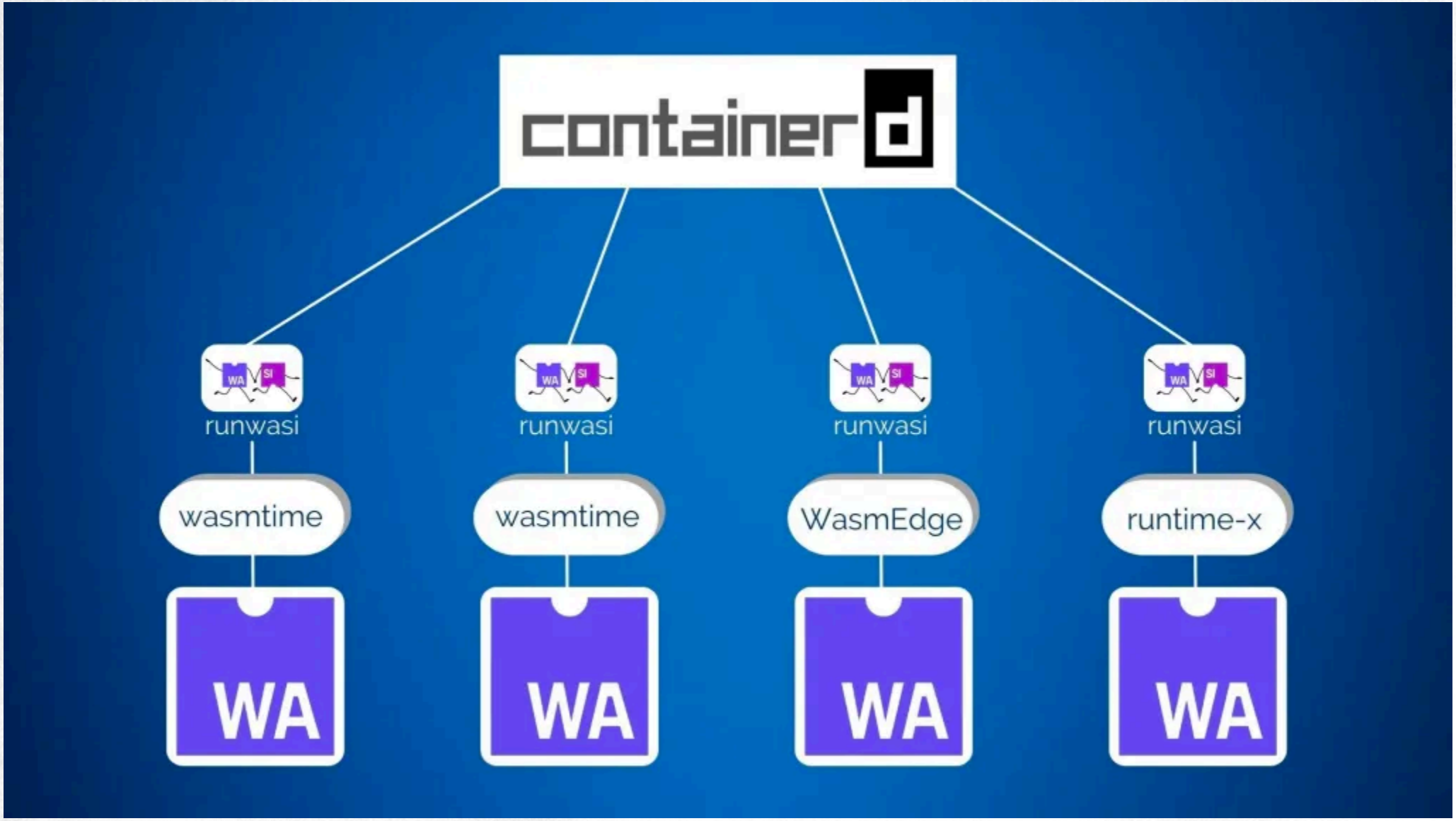
WASM/WASI as Universal App Distribution !!



REFERENCE: https://medium.com/dartlang/experimenting-with-dart-and-wasm-ef7f1c065577

What's going on?

WASM/WASI with Kubernetes !!



REFERENCE: https://nigelpoulton.com/webassembly-on-kubernetes-everything-you-need-to-know/

What's going on?

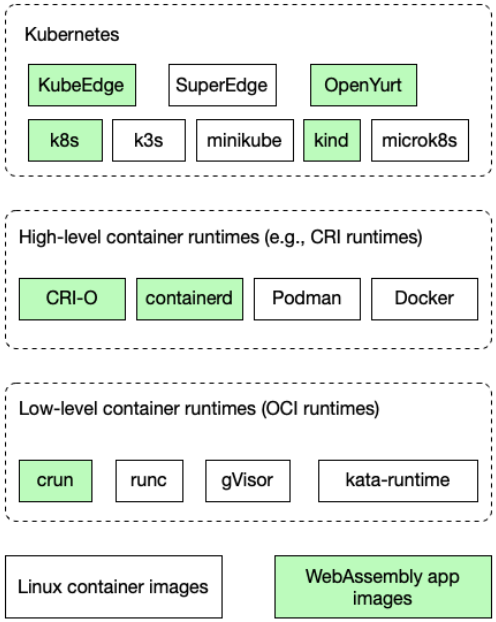
WASM/WASI with Kubernetes !!

WasmEdge in Kubernetes

Developers can leverage container tools such as Kubernetes, Docker and CRI-O to deploy, manage, and run lightweight WebAssembly applications. In this chapter, we will demonstrate how Kubernetes ecosystem tools work with WasmEdge WebAssembly applications.

Compared with Linux containers, WebAssembly could be 100x faster at startup, have a much smaller memory and disk footprint, and have a better-defined safety sandbox. However, the trade-off is that WebAssembly requires its own language SDKs, and compiler toolchains, making it a more constrained developer environment than Linux containers. WebAssembly is increasingly used in Edge Computing scenarios where it is difficult to deploy Linux containers or when the application performance is vital.

One of the great advantages of Linux application containers is the rich ecosystem of tools. The good news is that you can use the exact same tools to manage WebAssembly applications, enabling Linux containers and WebAssembly apps to run side-by-side in the same system.



The container ecosystem

Compared with Linux containers, WebAssembly could be 100x faster at startup, have a much smaller memory and disk footprint, and have a better-defined safety sandbox.

REFERENCE: https://wasmedge.org/book/en/use_cases/kubernetes.html

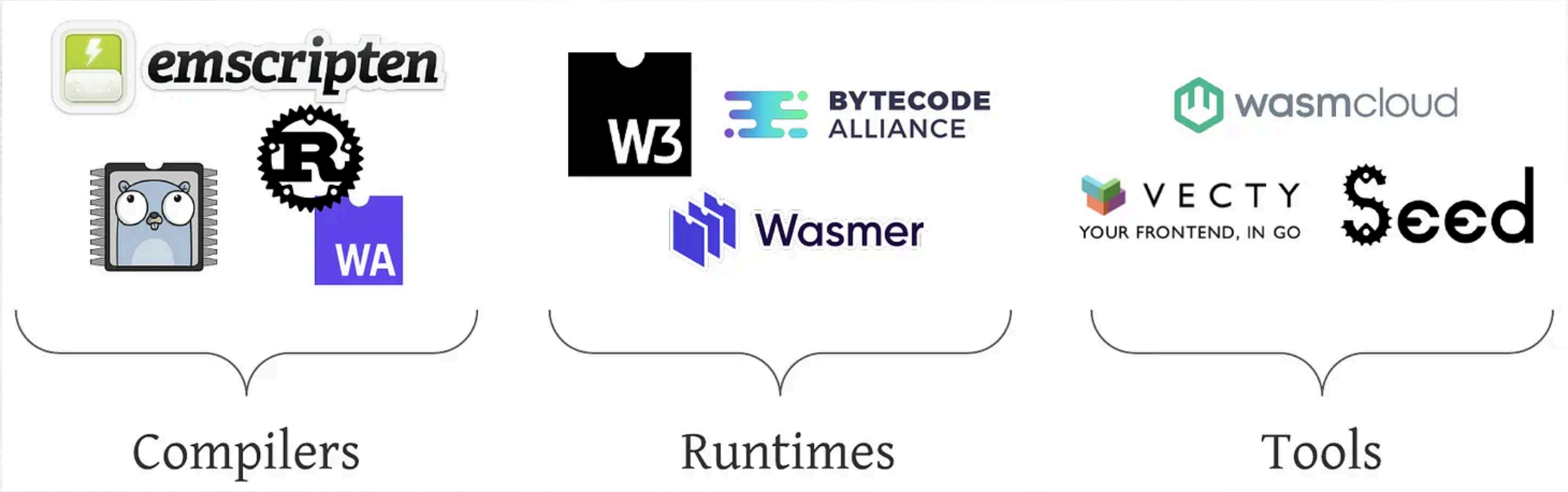
What's going on?

WASM/WASI with CNCF !!

REFERENCE: https://www.youtube.com/watch?v=b1675UWMLas&list=WL&index=2

Ecosystem

WebAssembly Universe



REFERENCE: https://cloudconstructed.substack.com/p/webassembly-a-primer

Ecosystem

WebAssembly Universe

- WebAssembly (Official) : https://webassembly.org/
- WebAssembly : https://developer.mozilla.org/en-US/docs/WebAssembly
- Bytecode Alliance : https://bytecodealliance.org/
- WASI : https://wasi.dev/
- Wasmer : https://wasmer.io/
- WASIX : https://wasix.org/
- WasmEdge : https://wasmedge.org/

Use Cases

Figma (2017)

SHORTCUT

Sign up for Figma | Maker Stories | Working Well | Inside Figma | Insights | Topics

June 8, 2017

WebAssembly cut Figma's load time by 3x

Evan Wallace Co-founder, Figma

Inside Figma | Engineering | Infrastructure

REFERENCE: https://www.figma.com/blog/webassembly-cut-figmas-load-time-by-3x/

Use Cases

Diablo - Experimental (2019)

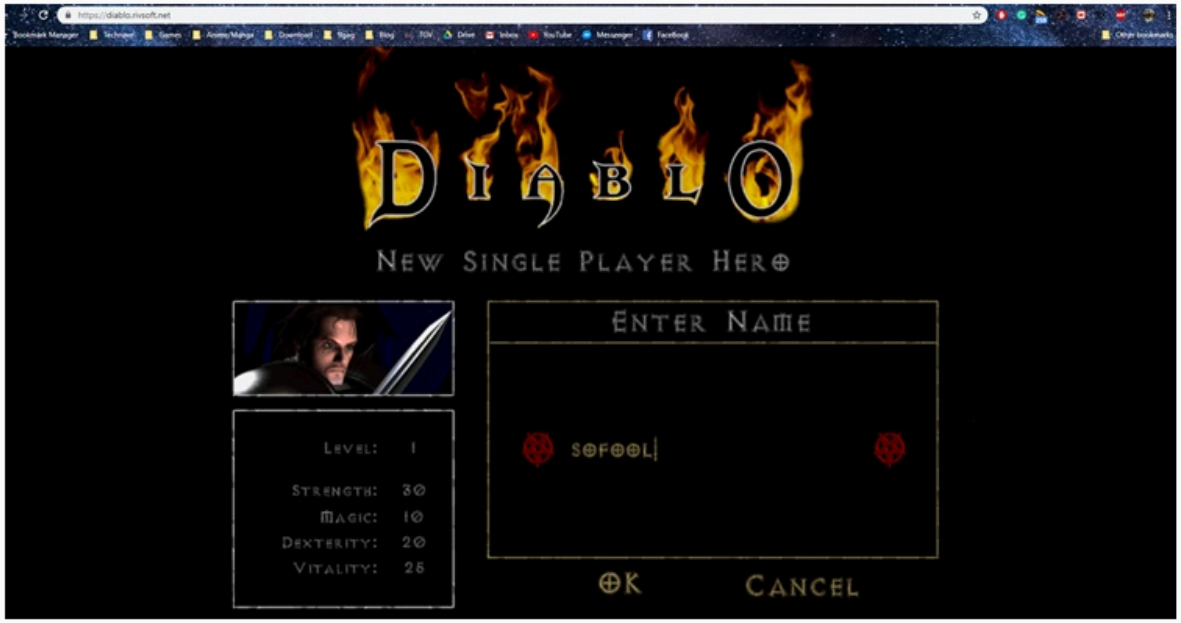
Made with WebAssembly

Home New Projects About

Diabloweb

Website Source Code

Diabloweb is Diablo 1 for web browsers!



This project uses WebAssembly as it ports the Source code of deviloution, using Emscripten. This is quite notable, as it proves that WebAssembly can bring these large C/C++ codebases using Emscripten to the web, to run impressive 3d games.

If you are interested in porting C/C++ libraries or applications, I'd highly reccomend Ben Smith's (binji) SFHTML5 Talk on porting C projects to the web. This talk is unrealted to D3Wasm, but can help drive what goes into porting these types of applications.

Date Added: 19. 11. 22.

Visit !!!! : https://d07riv.github.io/diabloweb/

REFERENCE: https://madewithwebassembly.com/showcase/diabloweb/

Use Cases

AutoCAD (2019)

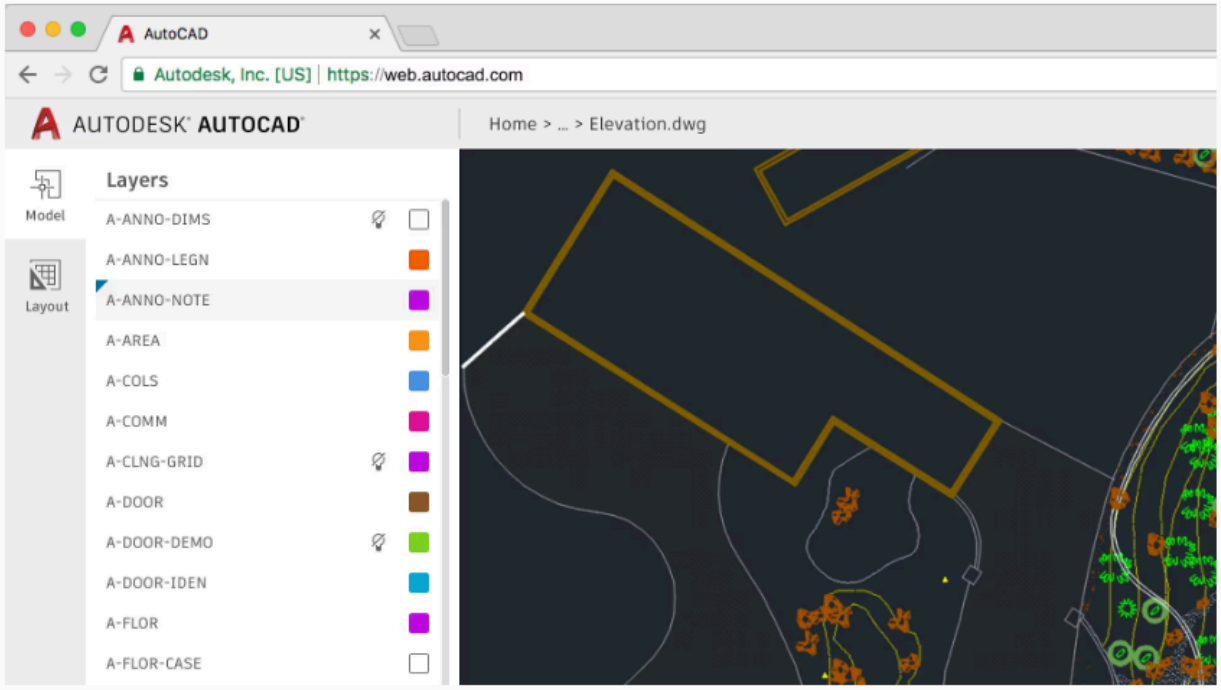
The maintainer of this website has a Spotify Coding Playlist of their Lo-fi Hip Hop beats!

WA **Made with WebAssembly** Home New Projects About

AutoCAD Web App

Website

AutoCAD is a design tool for creating 2D and 3D drawings.



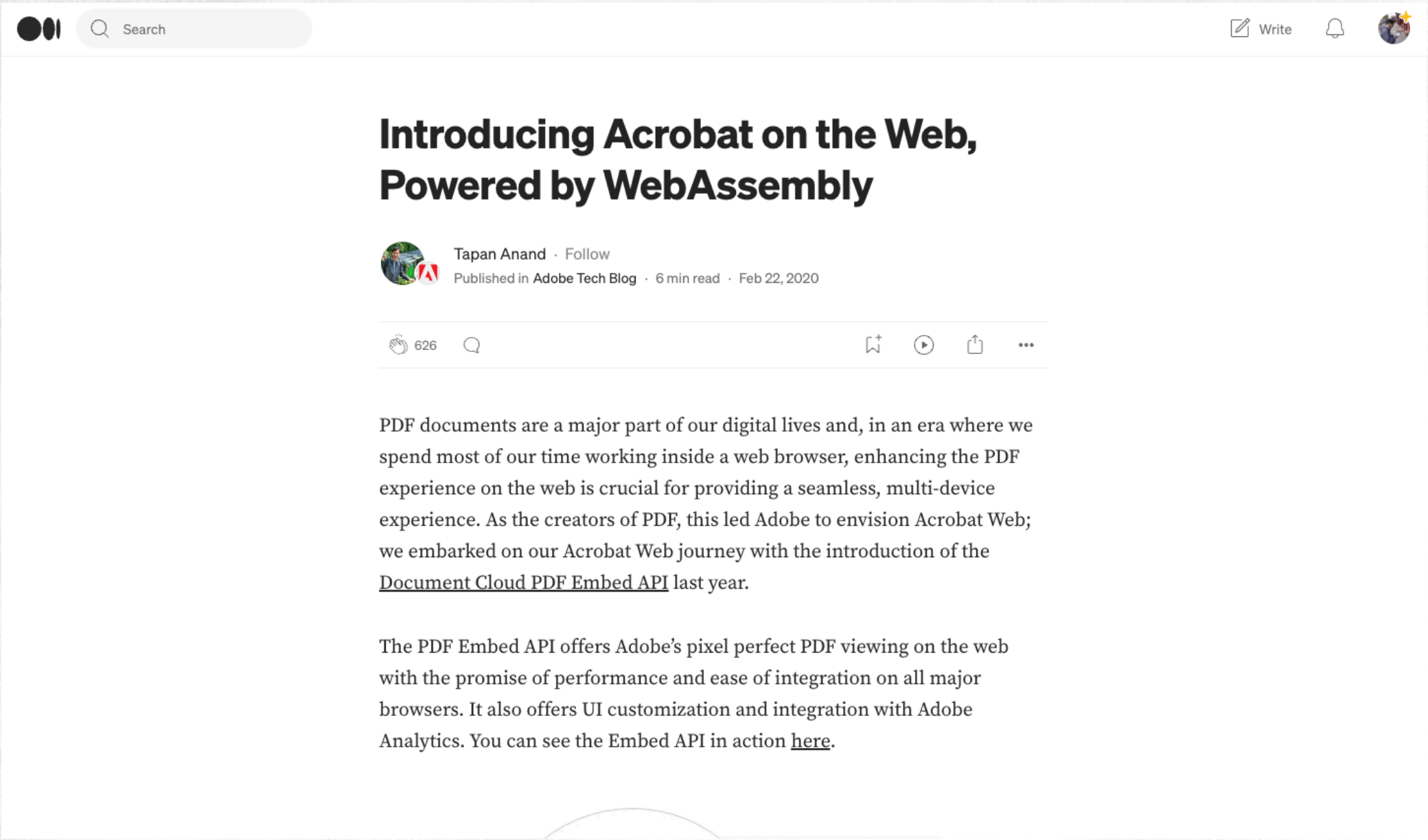
The AutoCAD web app uses emscripten to port pieces from the > 35 years old native application for AutoCAD, to the web! This is quite notable, as it proves that WebAssembly can bring these large C/C++ codebases using Emscripten to the web, to run large computationally intensive desktop applications on the web!

If you are interested in porting C/C++ libraries, I'd highly reccomend Ben Smith's (binji) SFHTML5 Talk on porting C projects to the web. This talk is unrealted to Google Earth, but can help drive what goes into porting these types of applications.

REFERENCE: https://madewithwebassembly.com/showcase/autocad/

Use Cases

Adobe Acrobat (2020)



REFERENCE: https://blog.developer.adobe.com/acrobat-on-the-web-powered-by-webassembly-782385e4947e

Use Cases

Adobe Photoshop @Beta (2021)

web.dev 정보 블로그 기사 알아보기 탐색 패턴 우수사례 검색 Language 로그인

translated by Google 이 페이지는 Cloud Translation API를 통해 번역되었습니다. SWITCH TO ENGLISH

홈 > Articles 도움이 되었나요?

Photoshop의 웹 여정

이 페이지의 내용
- Photoshop을 웹에 사용하게 된 이유
- Photoshop을 웹으로 모은 방법
- Emscripten으로 WebAssembly 포팅
- WebAssembly 디버깅
- 고성능 스토리지
- 캔버스의 P3 색상 공간
- 웹 구성요소 및 Lit
- Workbox에서 서비스 워커 캐싱
- 웹용 Adobe의 다음 단계

브라우저에서 Photoshop처럼 복잡한 소프트웨어를 실행한다는 아이디어는 불과 몇 년 전만 해도 상상하기 어려웠을 것입니다. 그러나 여러 가지 새로운 웹 기술을 사용하여 Adobe에서 이제 Photoshop의 공개 베타를 웹에 제공합니다.

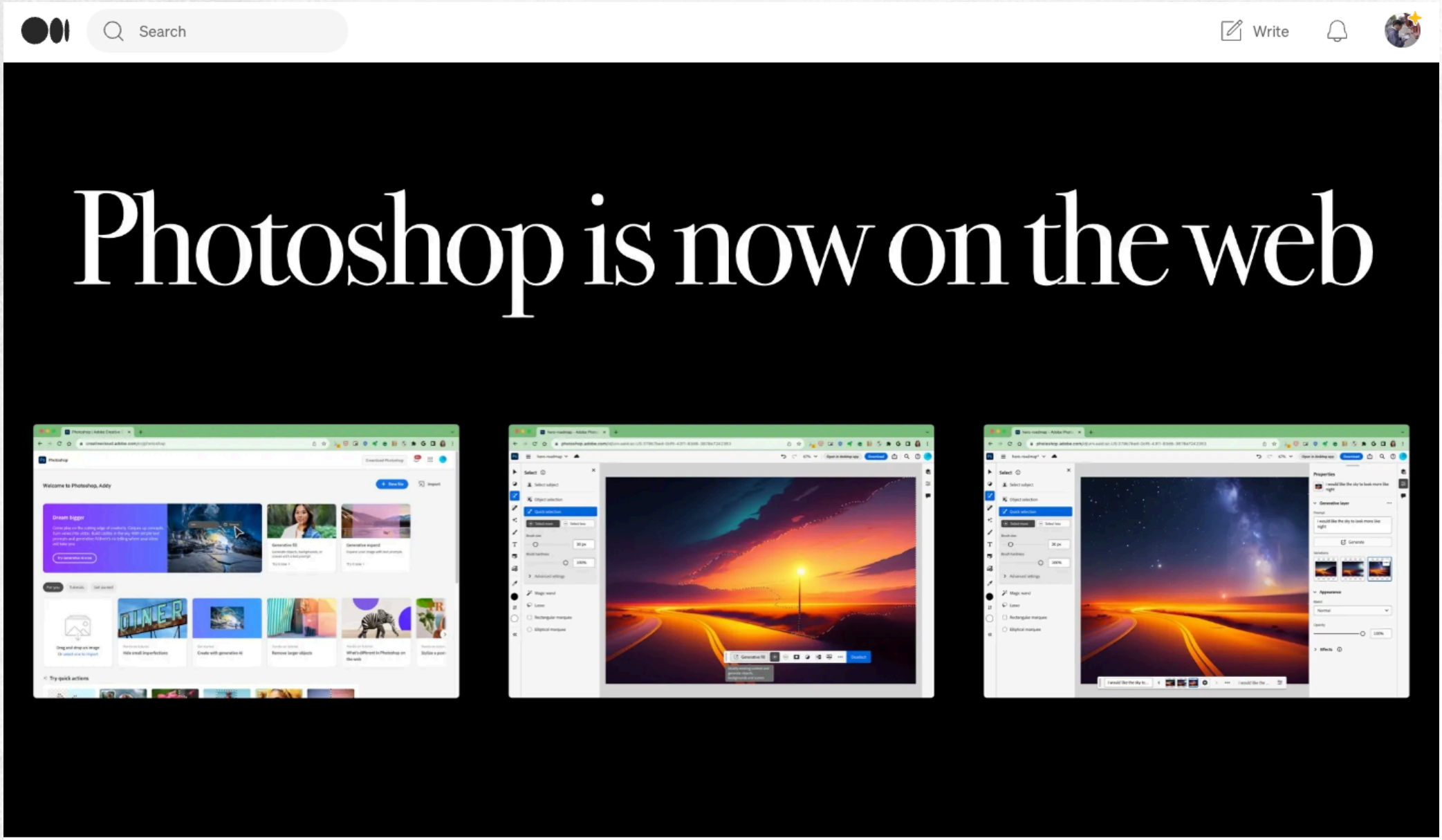
나벨 알 샤마
GitHub

토마스 나테스타드
트위터

지난 3년 동안 Chrome은 브라우저의 한계를 뛰어넘고자 하는 웹 애플리케이션을 지원하기 위해 노력해 왔습니다. 이러한 웹 애플리케이션 중 하나가 Photoshop입니다. 브라우저에서 Photoshop처럼 복잡한 소프트웨어를 실행한다는 아이디어는 불과 몇 년 전만 해도 상상하기 어려웠을 것입니다. 그러나 여러 가지 새로운 웹 기술을 사용하여 Adobe에서 이제 Photoshop의 공개 베타를 웹에 제공합니다.

Use Cases

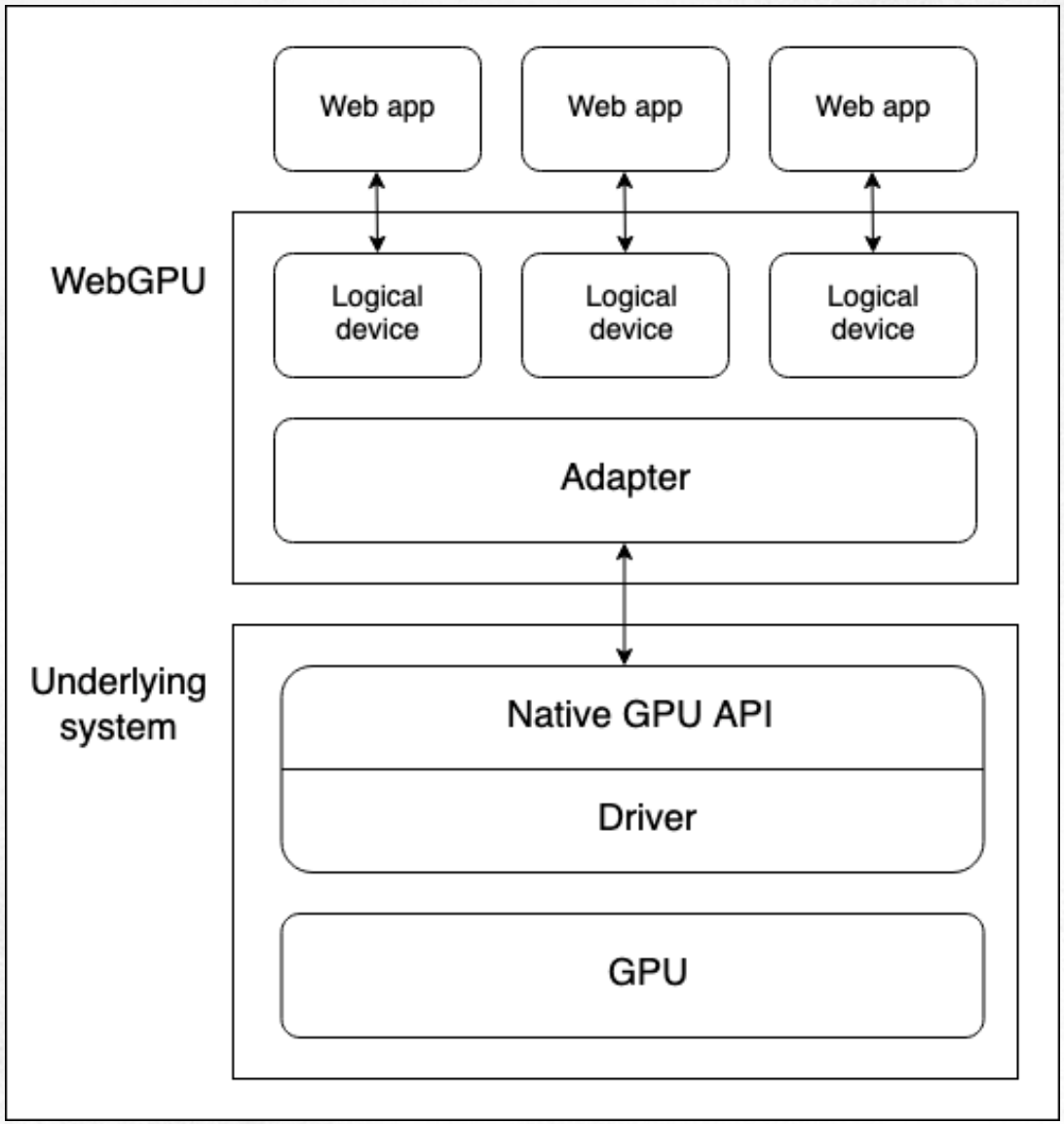
Adobe Photoshop (2023)



REFERENCE: https://medium.com/@addyosmani/photoshop-is-now-on-the-web-38d70954365a

And …

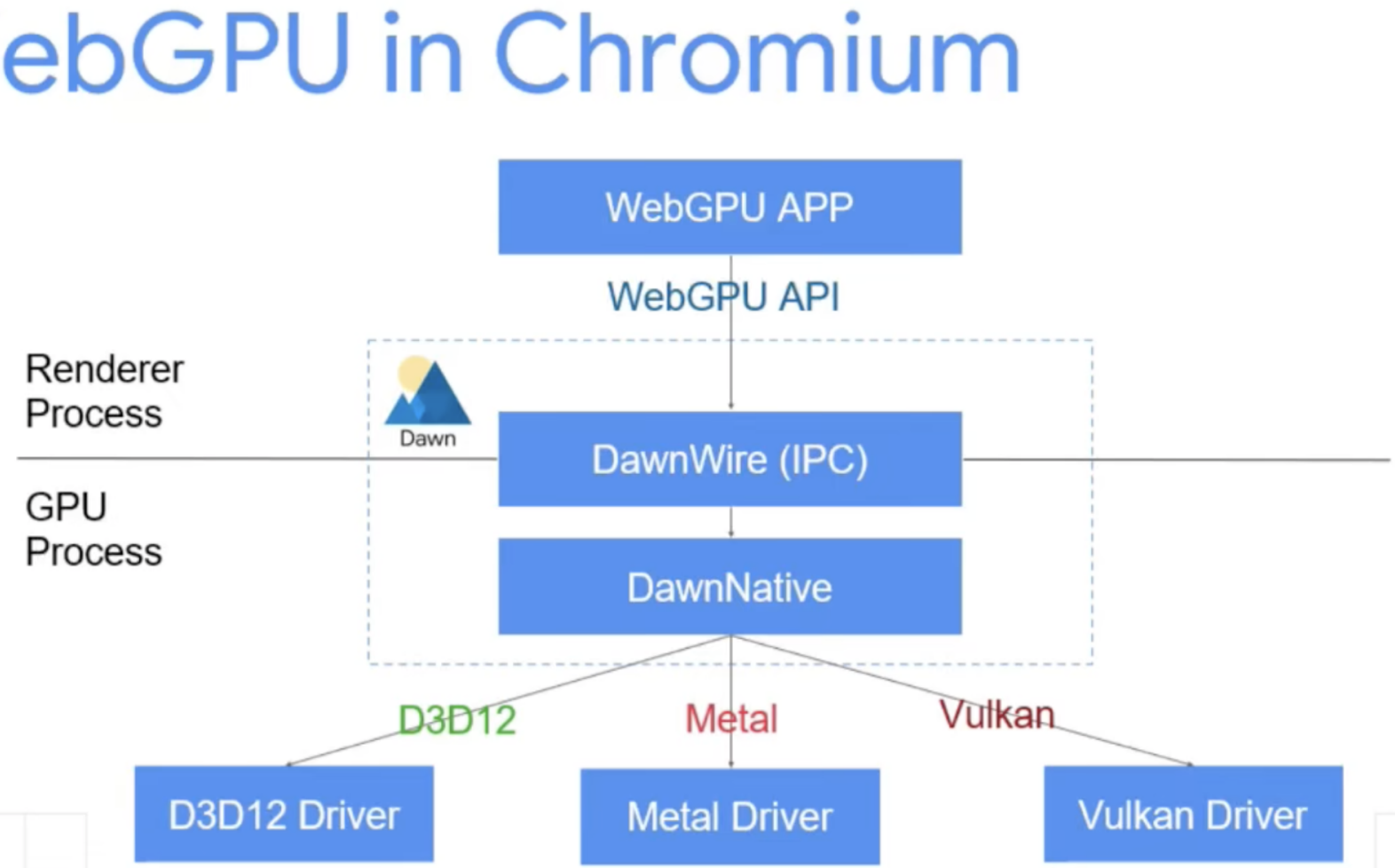
WebGPU



REFERENCE: https://developer.mozilla.org/en-US/docs/Web/API/WebGPU_API

And …

WebGPU


WebGPU in Chromium
WebGPU APP
WebGPU API
Renderer Process
Dawn
DawnWire (IPC)
GPU Process
DawnNative
D3D12
Metal
Vulkan
D3D12 Driver
Metal Driver
Vulkan Driver
GDG
Google Developers

Use Cases

Unity - Experimental (2023)

Official Early access to the new WebGPU backend in Unity 2023.3

rendering

Page 1 of 5 1 2 3 4 5 Next >

dnach

Unity Technologies

Joined: Mar 9, 2022

Posts: 89

Greetings from the Unity Graphics team.

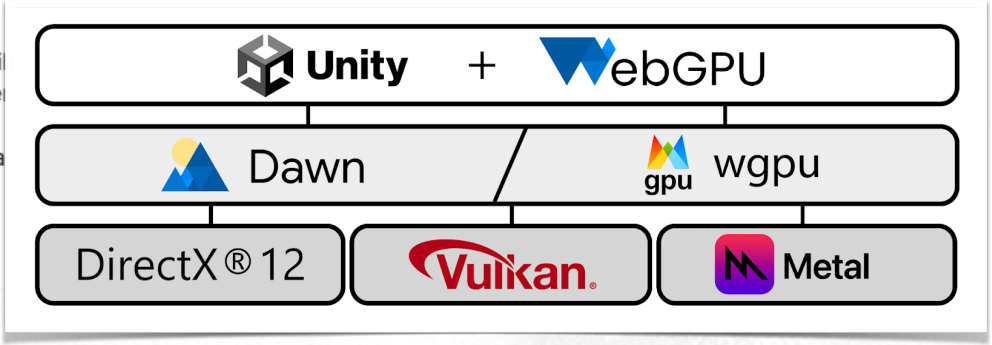
We're excited to announce that starting with Unity 2023.3, the Unity Web Player now provides experimental support for the new WebGPU graphics API.

WebGPU

The introduction of support for the new WebGPU API marks a significant mi
acceleration, and paves the way for unprecedented leaps in graphics rende

In this post, we share some useful information on how to enable WebGPU, a
limitations that should be taken into account.

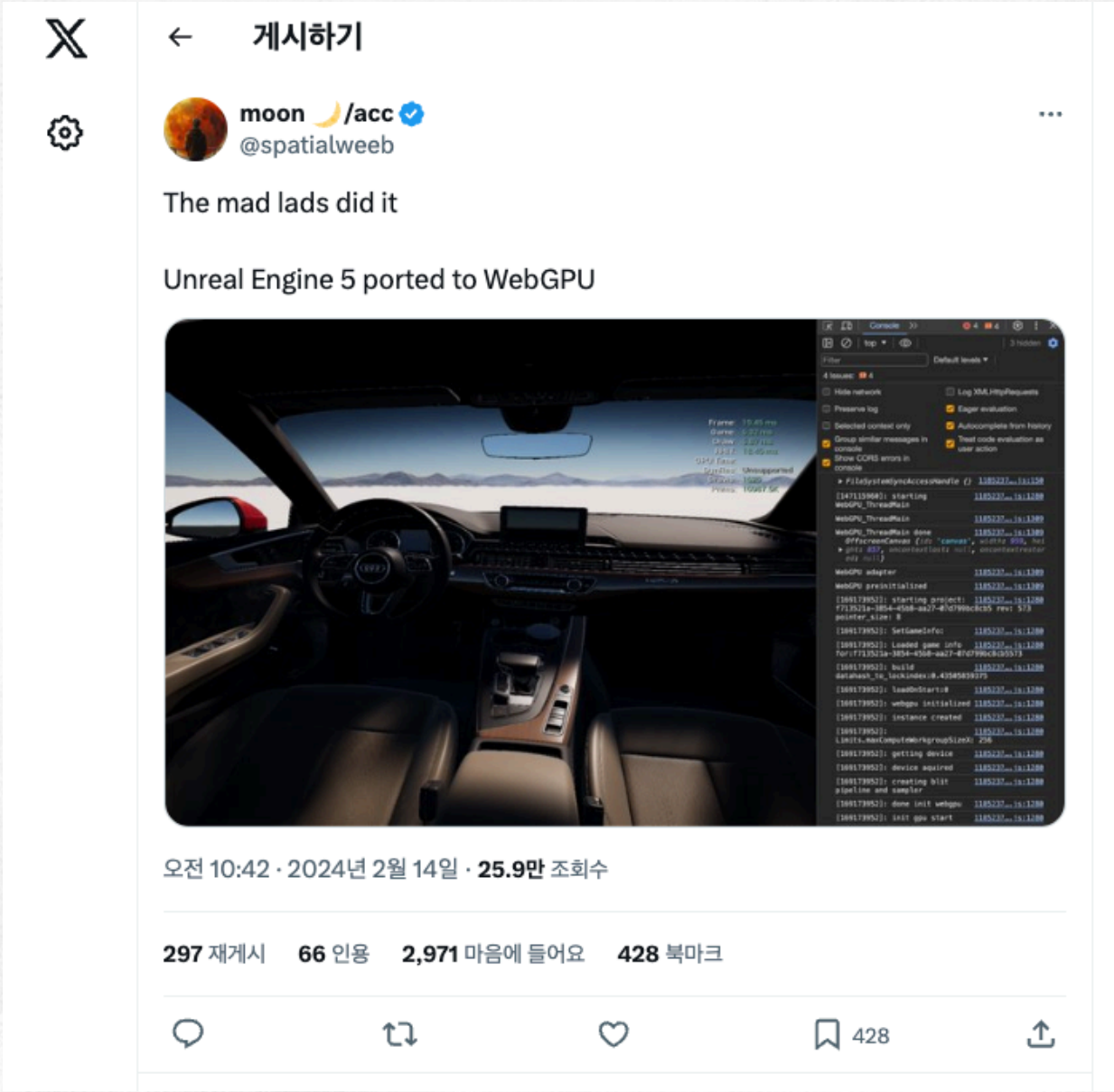
WebGPU: The future of graphics and compute on the Web



REFERENCE: https://forum.unity.com/threads/early-access-to-the-new-webgpu-backend-in-unity-2023-3.1516621/

Use Cases

Unreal - Unofficial (2024)



One More

Python over WebBrowser (WASM)

Write Your First “Hello, World!” in PyScript

The quickest way to get started with PyScript is by creating a minimal HTML5 document, saving it in a local file such as `hello.html`, and leveraging the two required files hosted on PyScript’s home page:

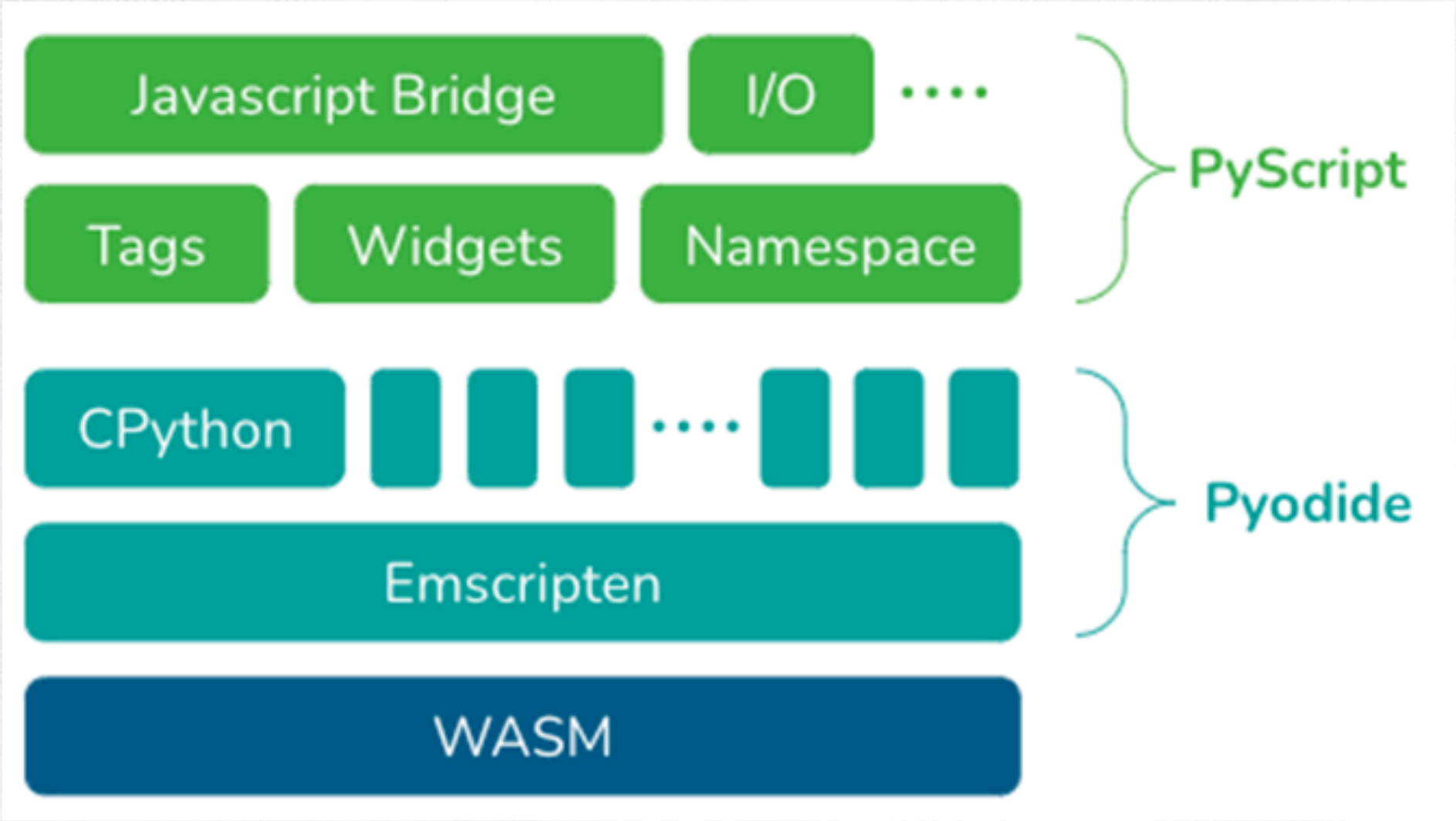
HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Hello, World!</title>
  <link rel="stylesheet" href="https://pyscript.net/alpha/pyscript.css" />
  <script defer src="https://pyscript.net/alpha/pyscript.js"></script>
</head>
<body>
  <py-script>print("Hello, World!")</py-script>
</body>
</html>
```

The first file, `pyscript.css`, provides default styling for PyScript’s visual components that you’ll explore later, as well as the loader splash screen. The second file, `pyscript.js`, contains JavaScript that bootstraps the Python runtime and adds custom elements like `<py-script>`, which can hold Python instructions such as a call to the `print()` function.

One More

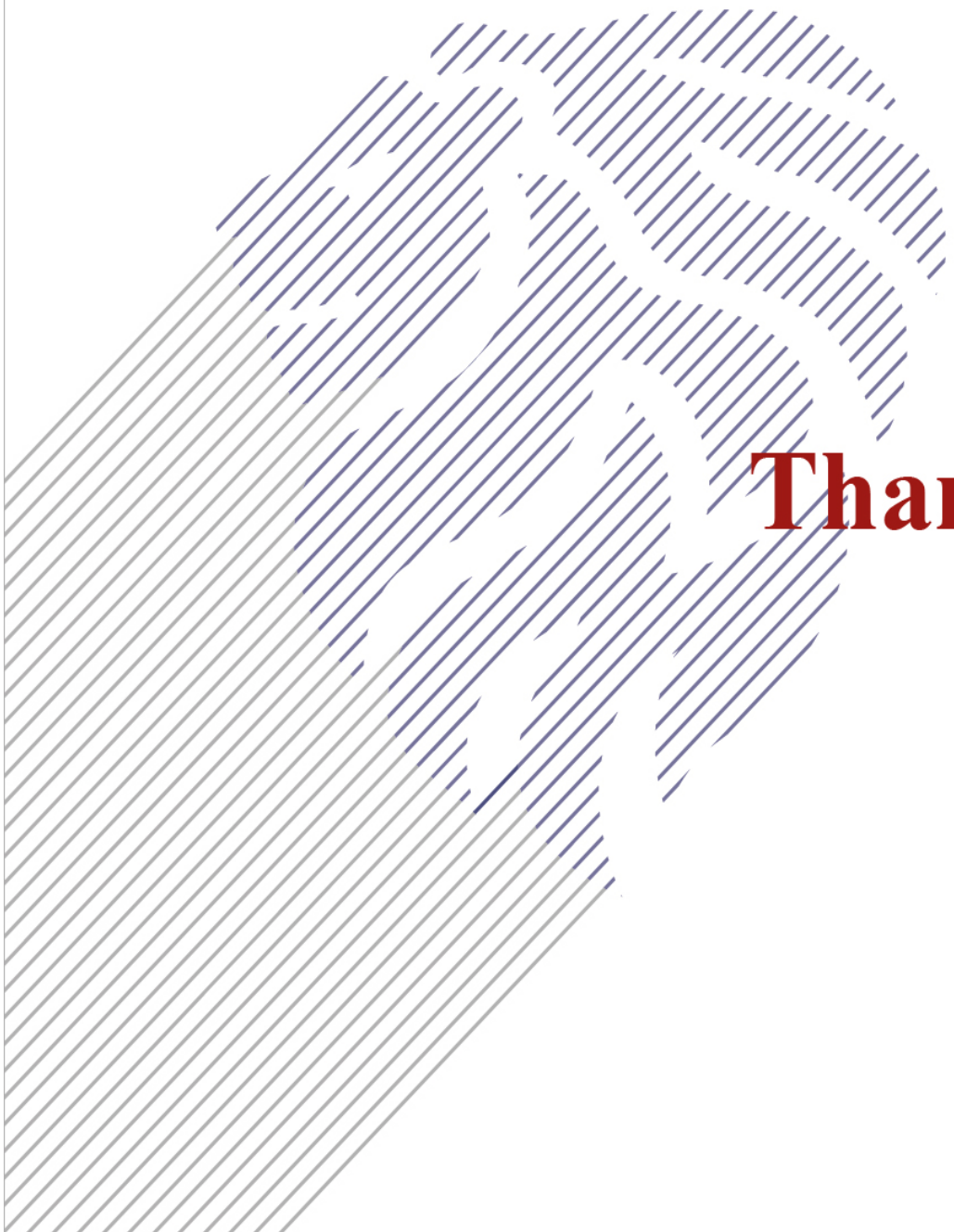
Python over WebBrowser (WASM)



One More

Python over WebBrowser (WASM)

- 경희의 오늘 (Opensource) : http://mobilelab.khu.ac.kr/
- Anaconda : https://www.anaconda.com/blog/pyscript-updates-bytecode-alliance-pyodide-and-micropython
- Anaconda : https://www.anaconda.com/blog/pyscript-python-in-the-browser
- Pyodide : https://pyodide.org/en/stable/#
- PyScript : https://pyscript.net/
- Real Python Tutorial : https://realpython.com/pyscript-python-in-browser/



Thank you

경희대학교
KYUNG HEE UNIVERSITY