



New Topics

NEW TECHNOLOGY Web Assembly (WASM)

2024. 01. 18

Sungwon Lee
Department of Software Convergence

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.

High Level Language to WASM

- C/C++ to WASM

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

- RUST to WASM

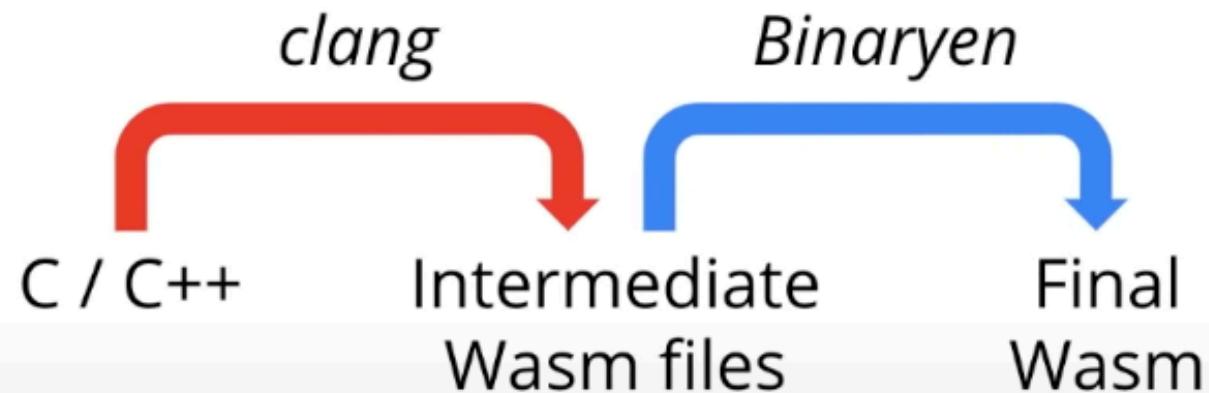
- https://developer.mozilla.org/en-US/docs/WebAssembly/Rust_to_wasm

- Dart/Flutter to WASM

- <https://docs.flutter.dev/platform-integration/web/wasm>

Emscripten pipeline

With native LLVM + WebAssembly



chrome dev summit 2019

Bird' Eye View

High Level Language to WASM

The screenshot shows the Emscripten homepage. At the top, there's a navigation bar with links for Documentation, Downloads, and Community. A "Fork me on GitHub" button is visible. The main content area features a large "emscripten" logo with a green lightning bolt icon. Below the logo, a text block states: "Emscripten is a complete compiler toolchain to WebAssembly, using LLVM, with a special focus on speed, size, and the Web platform." There are three columns: "Porting", "APIs", and "Fast". The "Porting" column describes how it can compile existing C/C++ projects to WebGL. The "APIs" column explains how Emscripten converts OpenGL to WebGL. The "Fast" column highlights the compact and fast output of the compiler. At the bottom, two call-to-action boxes encourage users to learn more or download the SDK.

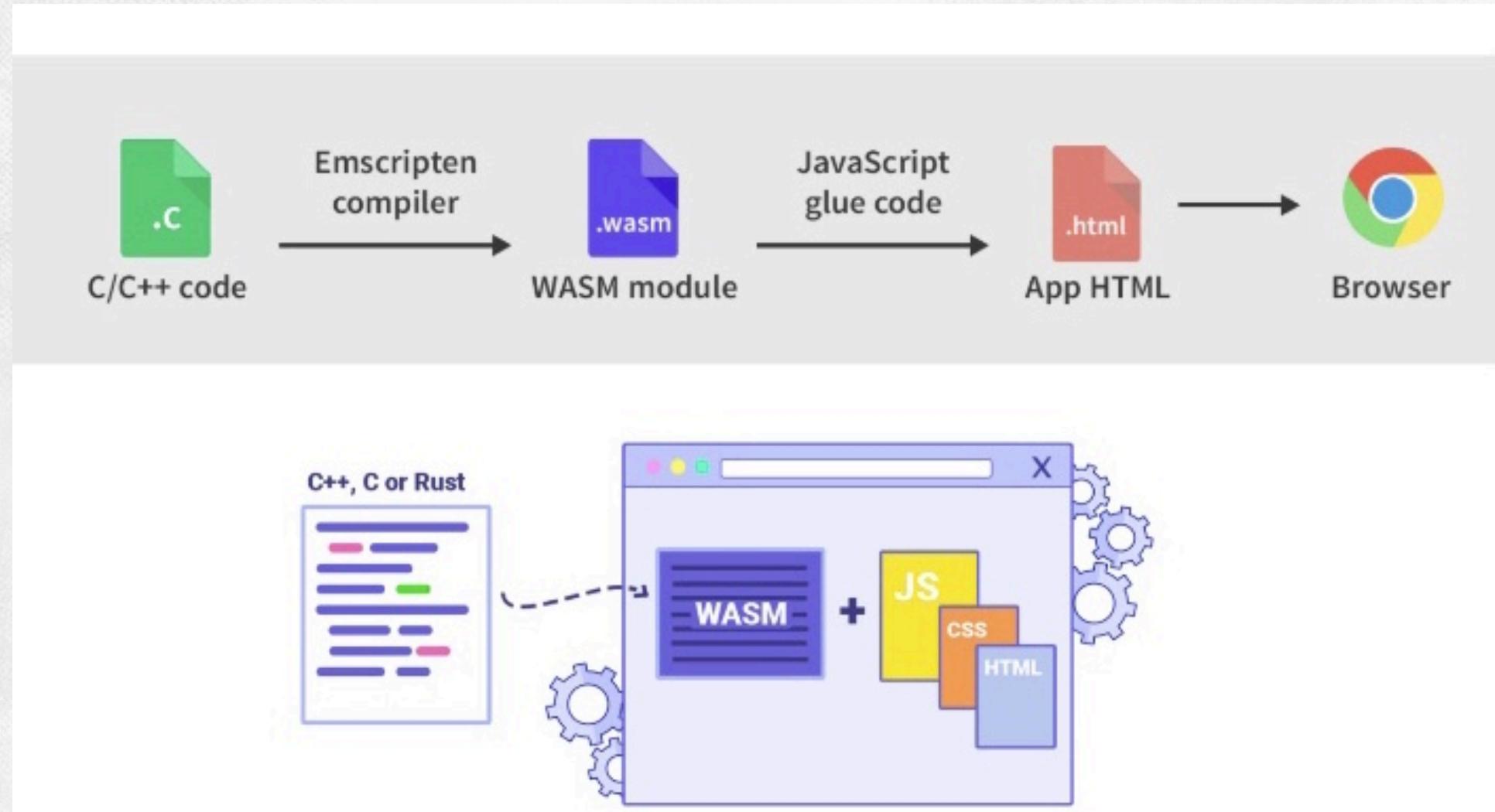
<https://emscripten.org/>

The screenshot shows the GitHub page for the Binaryen repository. The repository has over 6.9k stars and 177 watchers. It includes sections for Readme, Code of conduct, Activity, Custom properties, and Report repository. The Releases section shows version 116 (Latest) from Sep 15, 2023, with 111 releases. The Packages section indicates no packages published. The Contributors section shows 136 contributors with profile icons. A Languages chart at the bottom shows the distribution of code: WebAssembly (61.6%), C++ (29.0%), JavaScript (7.1%), Python (1.3%), C (0.9%), and CMake (0.1%).

<https://github.com/WebAssembly/binaryen>

Bird' Eye View

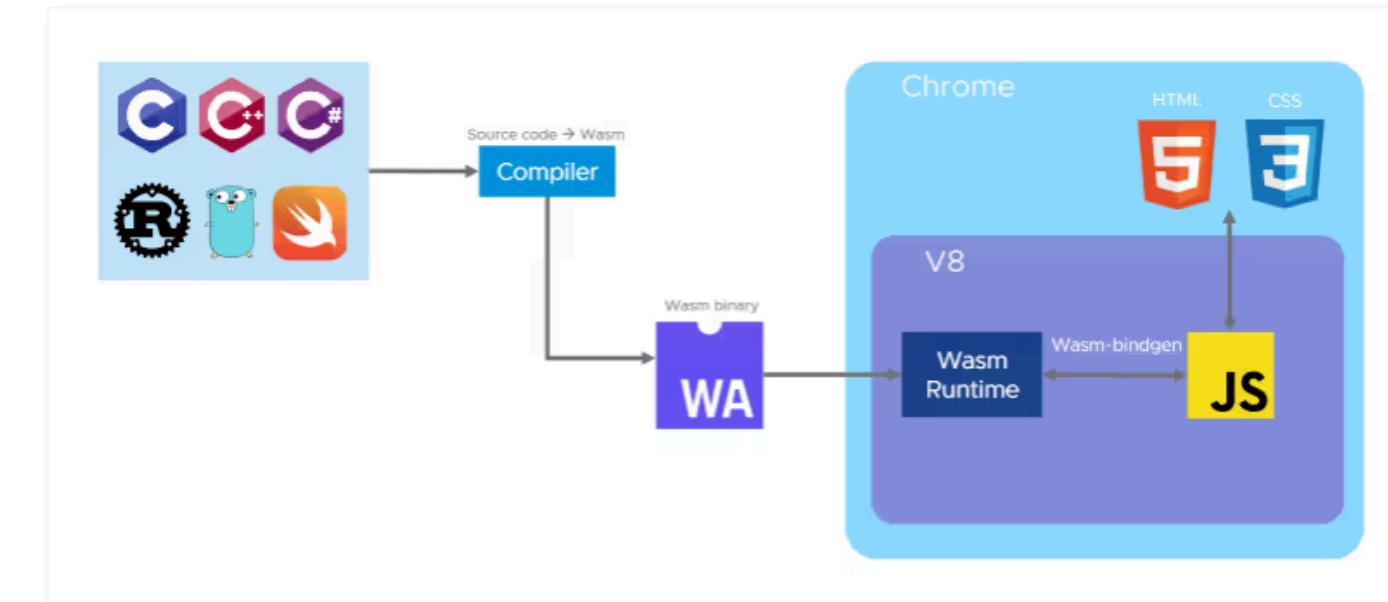
How it works with conventional Web techs?



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.



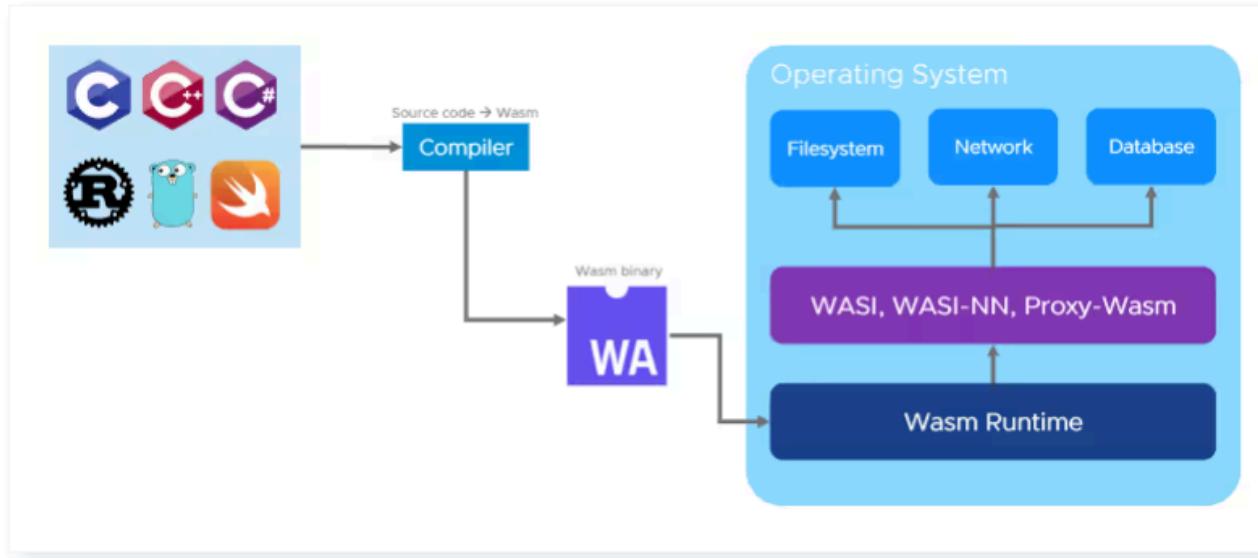
These technologies have allowed traditional desktop apps to run in a browser. And now they can run on any device on which you have a browser. Some notable examples are [Google Earth](#) and the [Open CV](#) library for computer vision.

What's going on?

WASI : Evolution of WASM !!

How does Wasm work on servers?

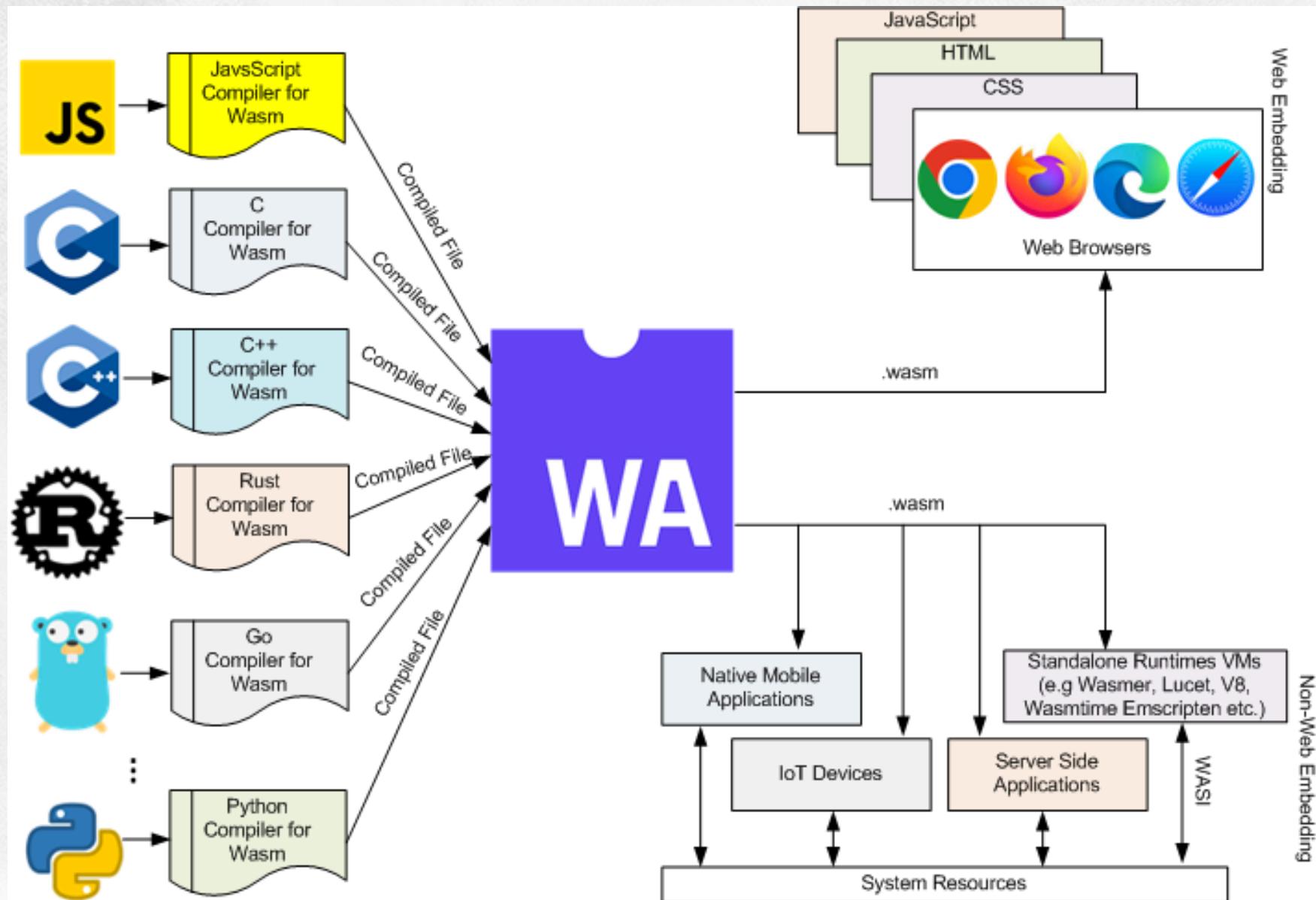
There are Wasm runtimes that can run outside of the browser, including traditional operating systems such as Linux, Windows and macOS. Because they cannot rely on a JavaScript engine being available they communicate with the outside world using different interfaces, such as WASI, the [WebAssembly System 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.

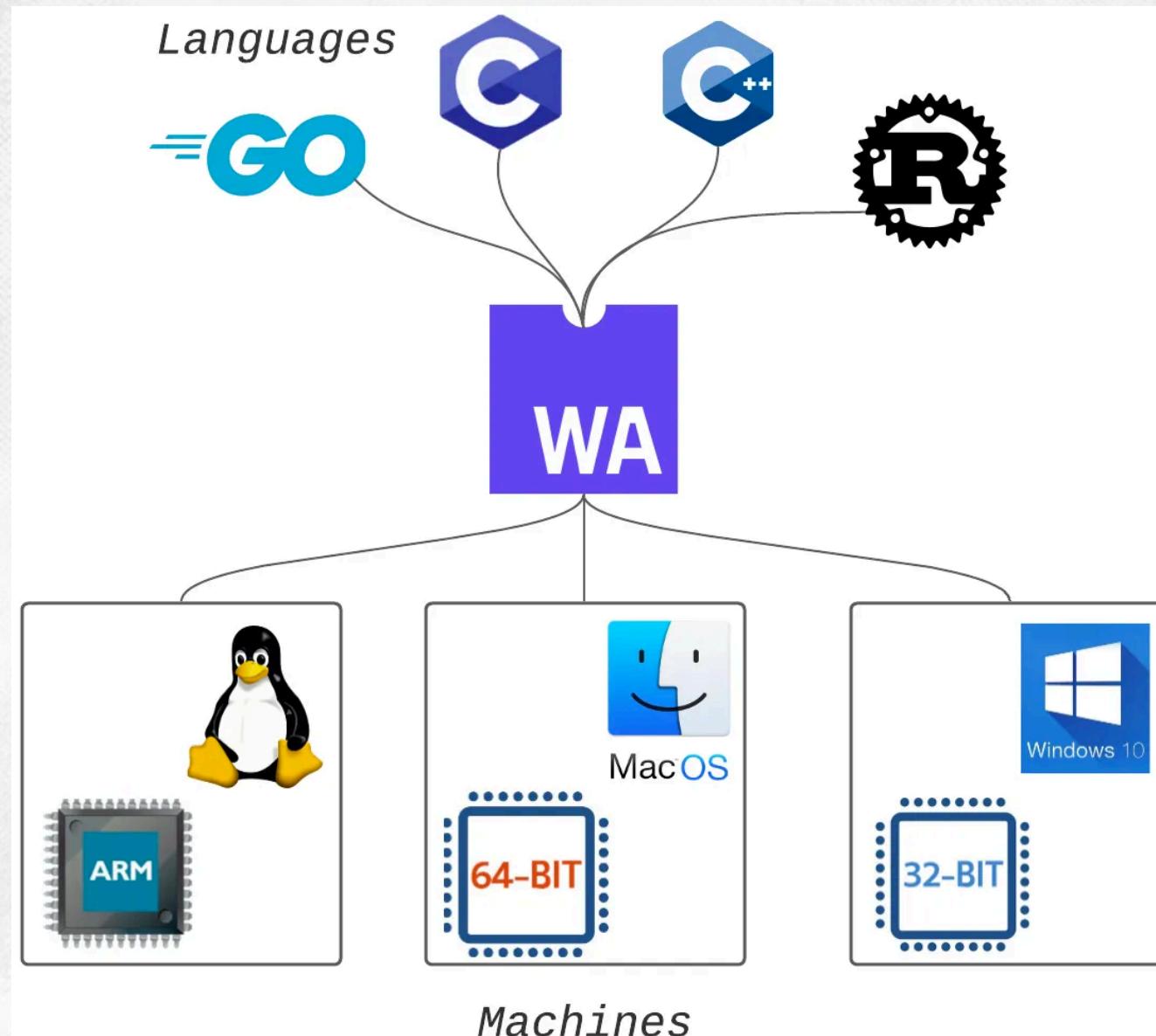
What's going on?

WASM/WASI for Every Domain !!



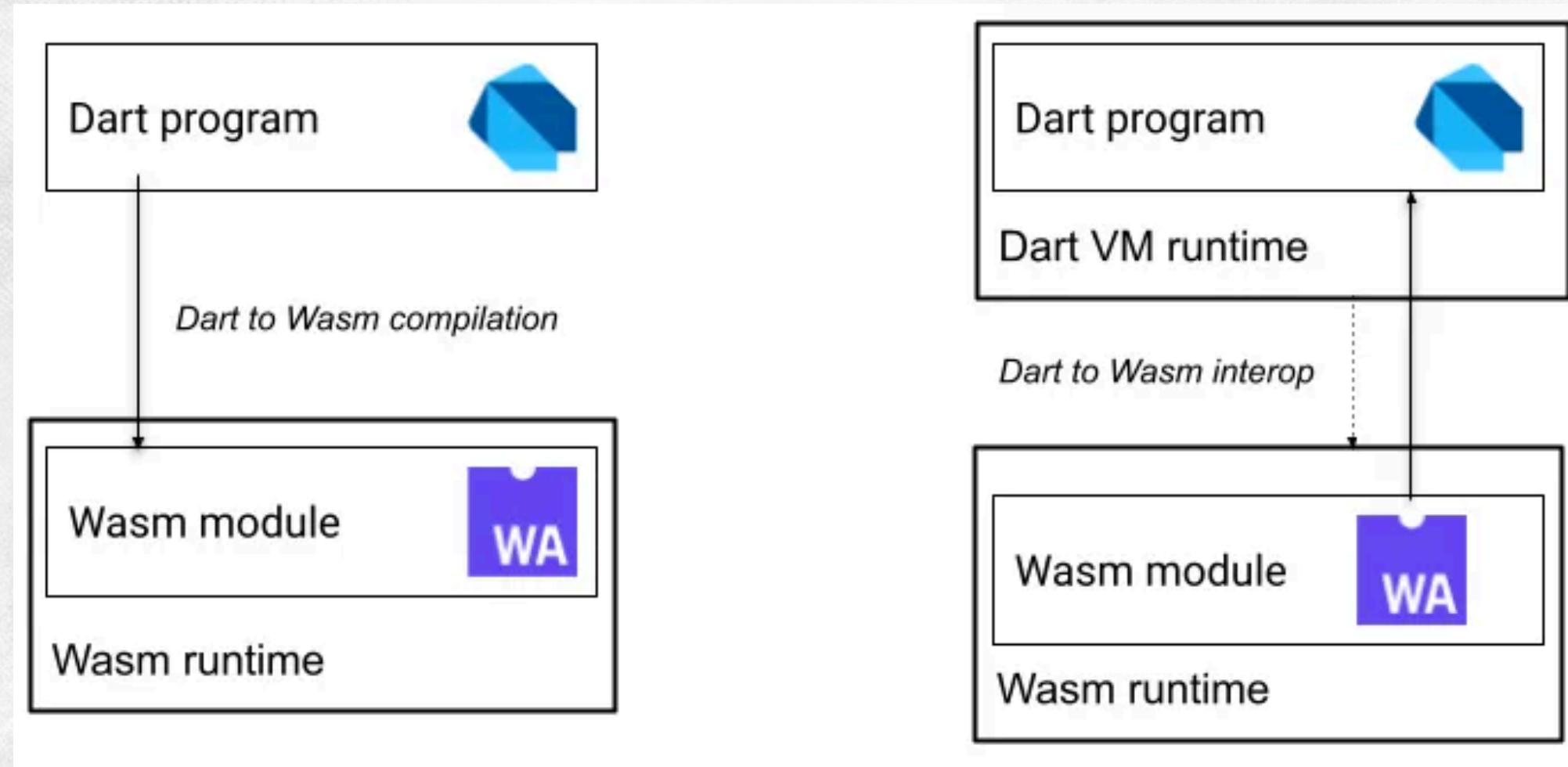
What's going on?

WASM/WASI for Every Technology !!



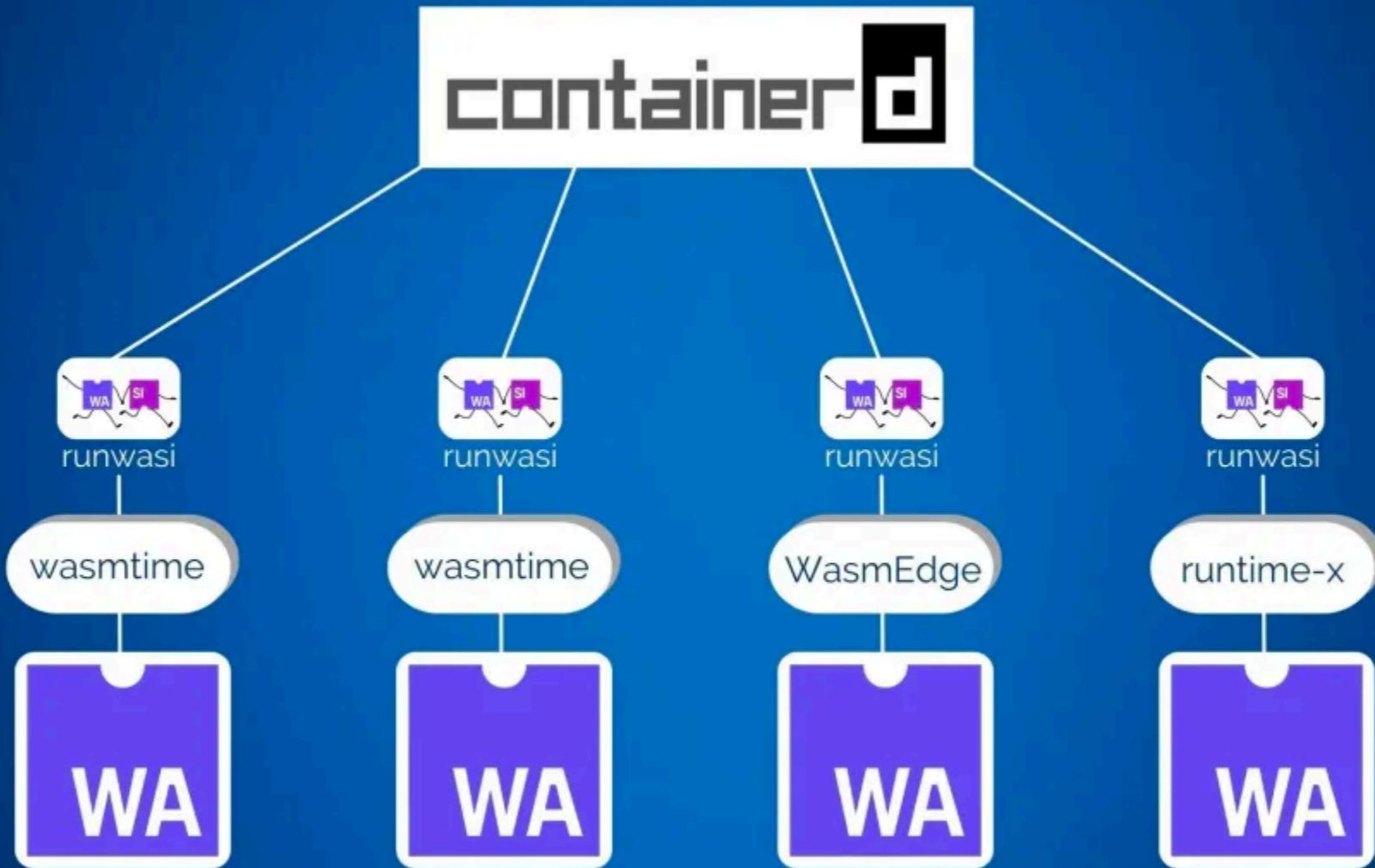
What's going on?

WASM/WASI as Universal App Distribution !!



What's going on?

WASM/WASI with Kubernetes !!



What's going on?

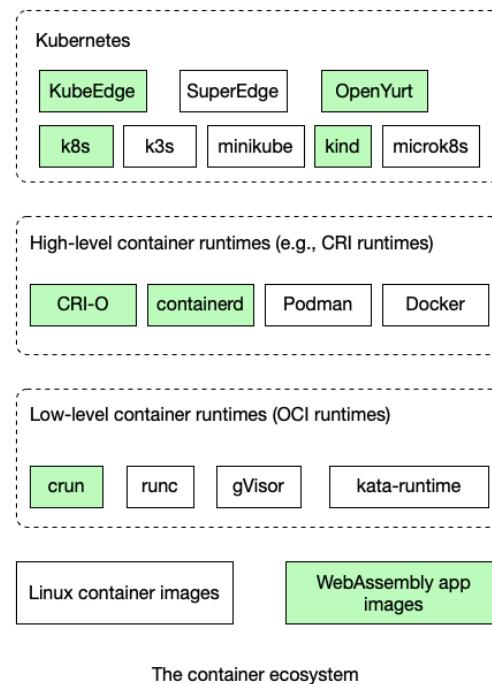
WASM/WASI with Kubernetes !!

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.



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.

What's going on?

WASM/WASI with CNCF !!

The image shows a YouTube video player interface. At the top, there is a navigation bar with the YouTube logo, a search bar, and various icons. Below the video player, there are logos for KubeCon and CloudNativeCon North America 2023. The main content of the video is a presentation slide with a pink background and white text. The title of the slide is "Bringing Cloud Native WASM to the mainstream with WASM Working Group". Below the title, the names "Shivay Lamba, Independent & Kevin Hoffman, Cosmonic" are listed. The video player has a progress bar at the bottom left showing 0:01 / 37:44. At the bottom right, there are video control buttons and a closed caption box in Korean.

Bringing Cloud Native WASM to the mainstream with WASM Working Group - Shivay Lamba & Kevin Hoffman

CNCF [Cloud Native Computing Foundation]

구독자 11만명

구독중 ▾

7

공유

오프라인 저장

클립

비공개 이성원(소프트웨어융합학과) - 1 / 8

나중에 볼 동영상

WebAssembly Universe



Compilers



Runtimes



Tools

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/>

SHORTCUT

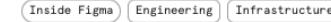
Sign up for Figma 

Maker Stories Working Well Inside Figma Insights Topics ▾ 

June 8, 2017

WebAssembly cut Figma's load time by 3x

 Evan Wallace Co-founder, Figma





Use Cases

Diablo - Experimental (2019)

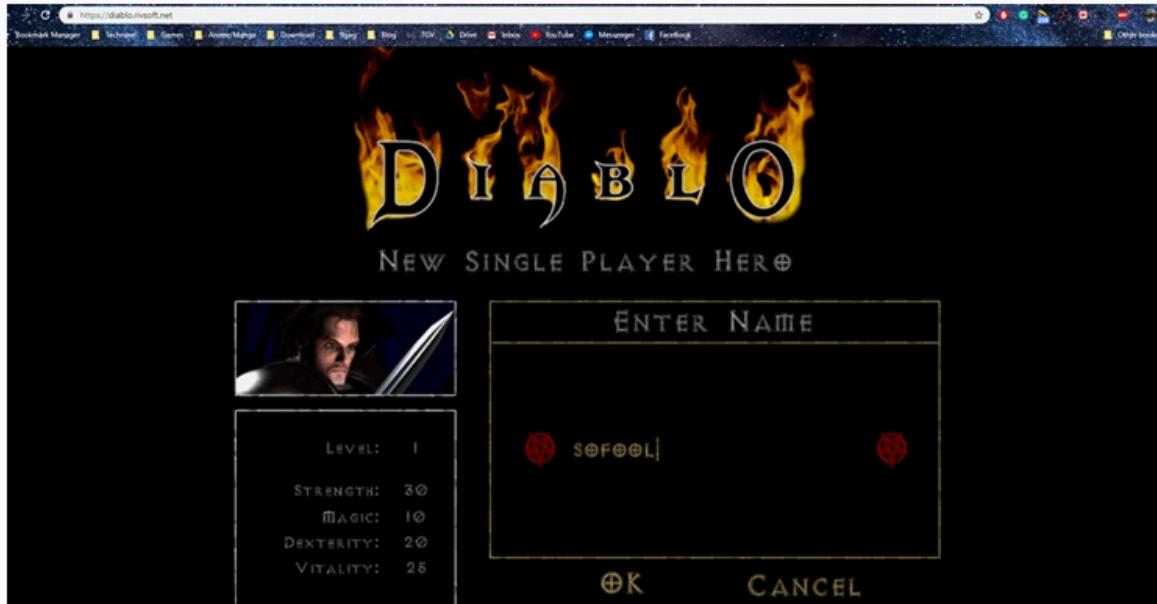
Made with WebAssembly

Home New Projects About

Diabloweb

[Website](#) [Source Code](#)

Diabloweb is Diablo 1 for web browsers!



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

This project uses WebAssembly as it ports the Source code of devilution, 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 recommend Ben Smith's (binji) SFHTML5 Talk on porting C projects to the web. This talk is unrelated to D3Wasm, but can help drive what goes into porting these types of applications.

Date Added: 19. 11. 22.

Use Cases

AutoCAD (2019)

The maintainer of this website has a [Spotify Coding Playlist of their Lo-fi Hip Hop beats!](#)



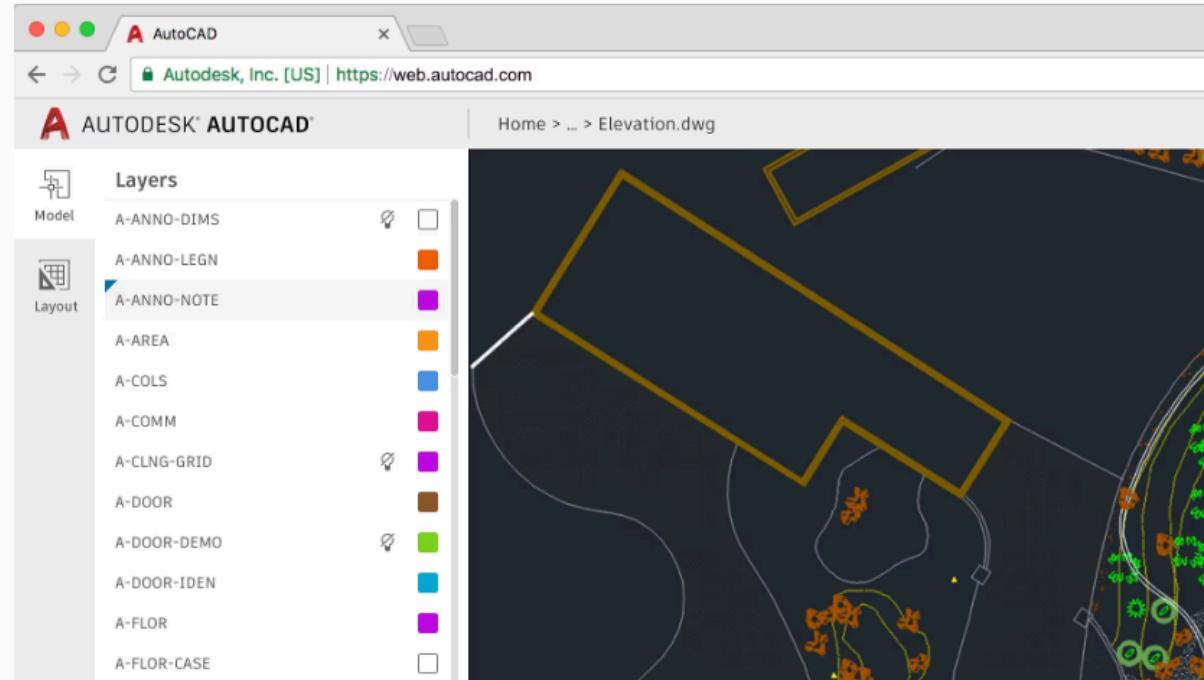
Made with WebAssembly

[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 recommend Ben Smith's ([binji](#)) SFHTML5 Talk on porting C projects to the web. This talk is unrelated to Google Earth, but can help drive what goes into porting these types of applications.

Adobe Acrobat (2020)

The screenshot shows a blog post from the Adobe Tech Blog. The title is "Introducing Acrobat on the Web, Powered by WebAssembly". The author is Tapan Anand, with a profile picture showing him outdoors. The post was published on Feb 22, 2020, and has 626 likes. The content discusses the importance of enhancing the PDF experience on the web for a seamless, multi-device experience, mentioning the Document Cloud PDF Embed API. It also highlights the PDF Embed API's pixel perfect viewing, performance, and ease of integration across major browsers, along with UI customization and integration with Adobe Analytics.

Introducing Acrobat on the Web, Powered by WebAssembly

Tapan Anand · Follow
Published in **Adobe Tech Blog** · 6 min read · Feb 22, 2020

626

PDF documents are a major part of our digital lives and, in an era where we spend most of our time working inside a web browser, enhancing the PDF experience on the web is crucial for providing a seamless, multi-device experience. As the creators of PDF, this led Adobe to envision Acrobat Web; we embarked on our Acrobat Web journey with the introduction of the [Document Cloud PDF Embed API](#) last year.

The PDF Embed API offers Adobe's pixel perfect PDF viewing on the web with the promise of performance and ease of integration on all major browsers. It also offers UI customization and integration with Adobe Analytics. You can see the Embed API in action [here](#).

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

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.

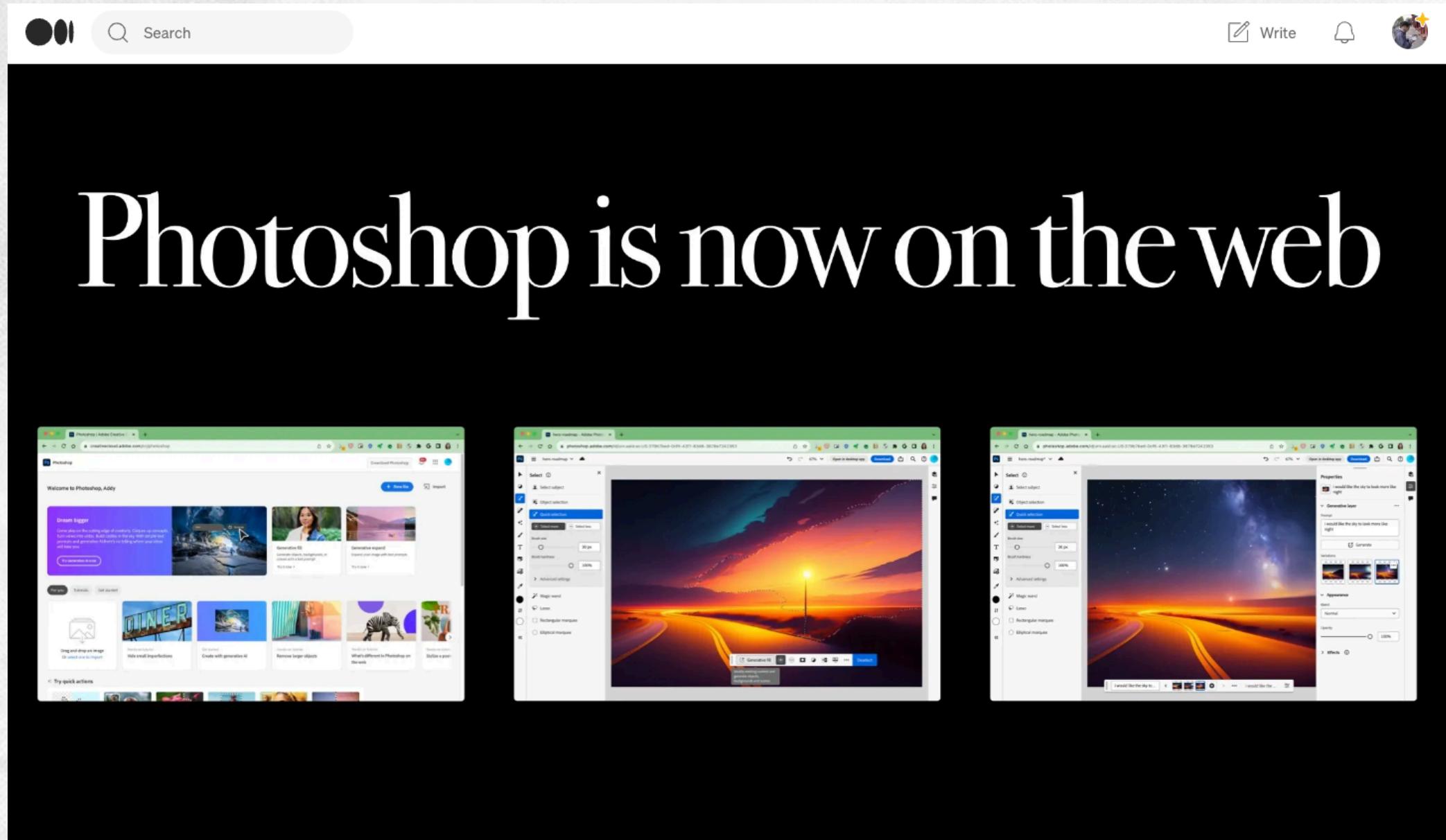
나벨 알 샤마 토마스 나테스타드
GitHub 트위터

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.



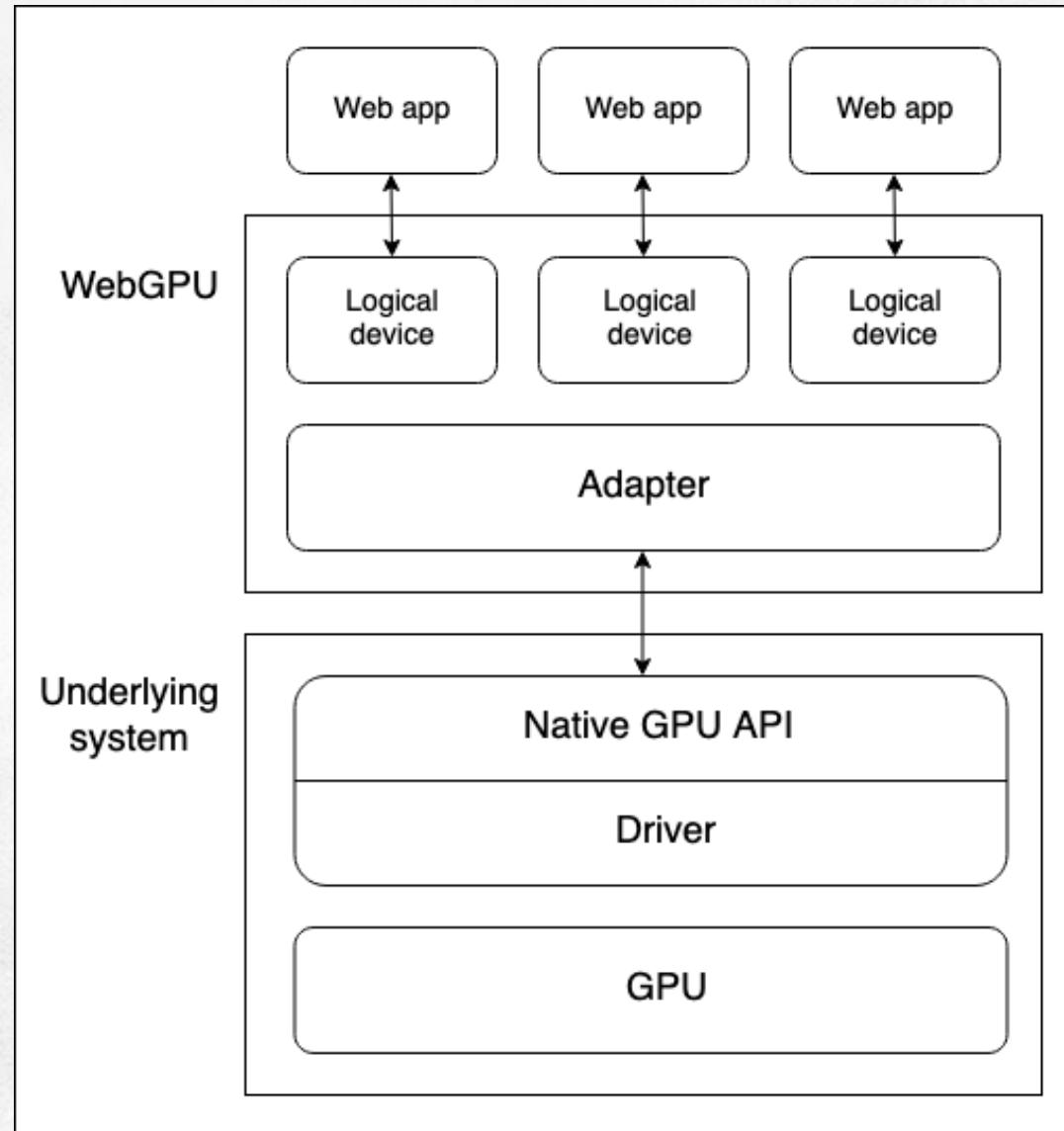
이 페이지의 내용

- 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



And ...

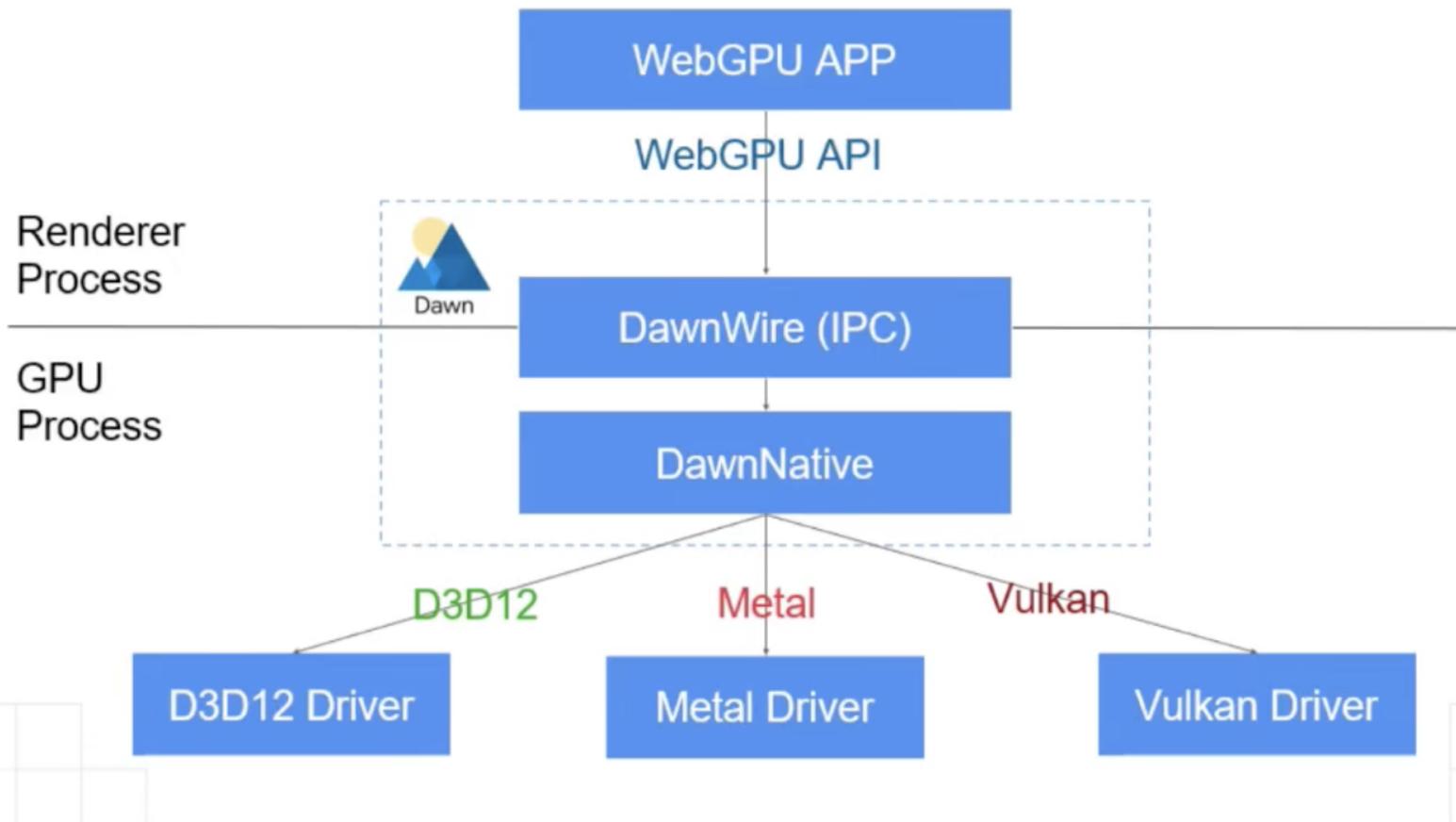
WebGPU



And ...

WebGPU

WebGPU in Chromium



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



Greetings from the Unity Graphics team.

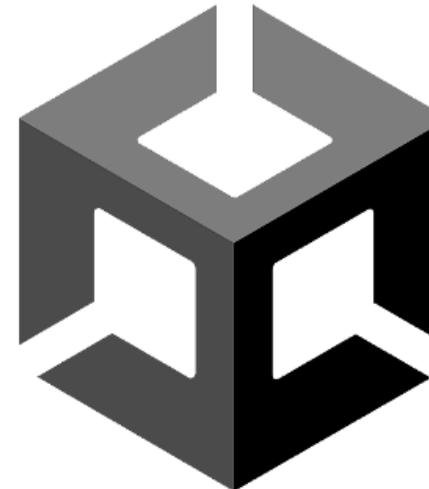
We're excited to announce that starting with Unity 2023.3, the Unity Web Player now provides experimental support for the new WebGPU graphics API.

Unity
Technologies

Joined:
Mar 9, 2022
Posts:
89



WebGPU



The introduction of support for the new WebGPU API marks a significant milestone in web rendering, providing faster performance and more efficient memory usage. This acceleration, combined with the power of modern GPUs, paves the way for unprecedented leaps in graphics rendering quality and performance.

In this post, we share some useful information on how to enable WebGPU, a new rendering backend for Unity, and discuss its features and limitations that should be taken into account.

WebGPU: The future of graphics and compute on the Web



DirectX® 12

Vulkan.

Metal

Use Cases

Unreal - Unofficial (2024)

The image shows a screenshot of a Twitter post from a user named 'moon 🌙/acc' (@spatialweeb). The post includes the text 'The mad lads did it' and 'Unreal Engine 5 ported to WebGPU'. Below the text is a photograph of the interior of a car, specifically the driver's side. Overlaid on the image is a GPU debugger window titled 'Console'. The console window displays various logs and metrics related to the Unreal Engine 5 port, such as frame times (e.g., 19.49 ms), draw counts (e.g., 2,271 ms), and GPU memory usage. The GPU debugger also shows network traffic and other system information. The entire screenshot is framed by a white border, characteristic of a social media post.

← 게시하기

moon 🌙/acc
@spatialweeb

The mad lads did it

Unreal Engine 5 ported to WebGPU

오전 10:42 · 2024년 2월 14일 · 25.9만 조회수

297 재게시 66 인용 2,971 마음에 들어요 428 북마크

言论 icon

分享 icon

喜欢 icon

收藏 icon

428

↑

Python over WebBrowser (WASM)

Write Your First “Hello, World!” in PyScript

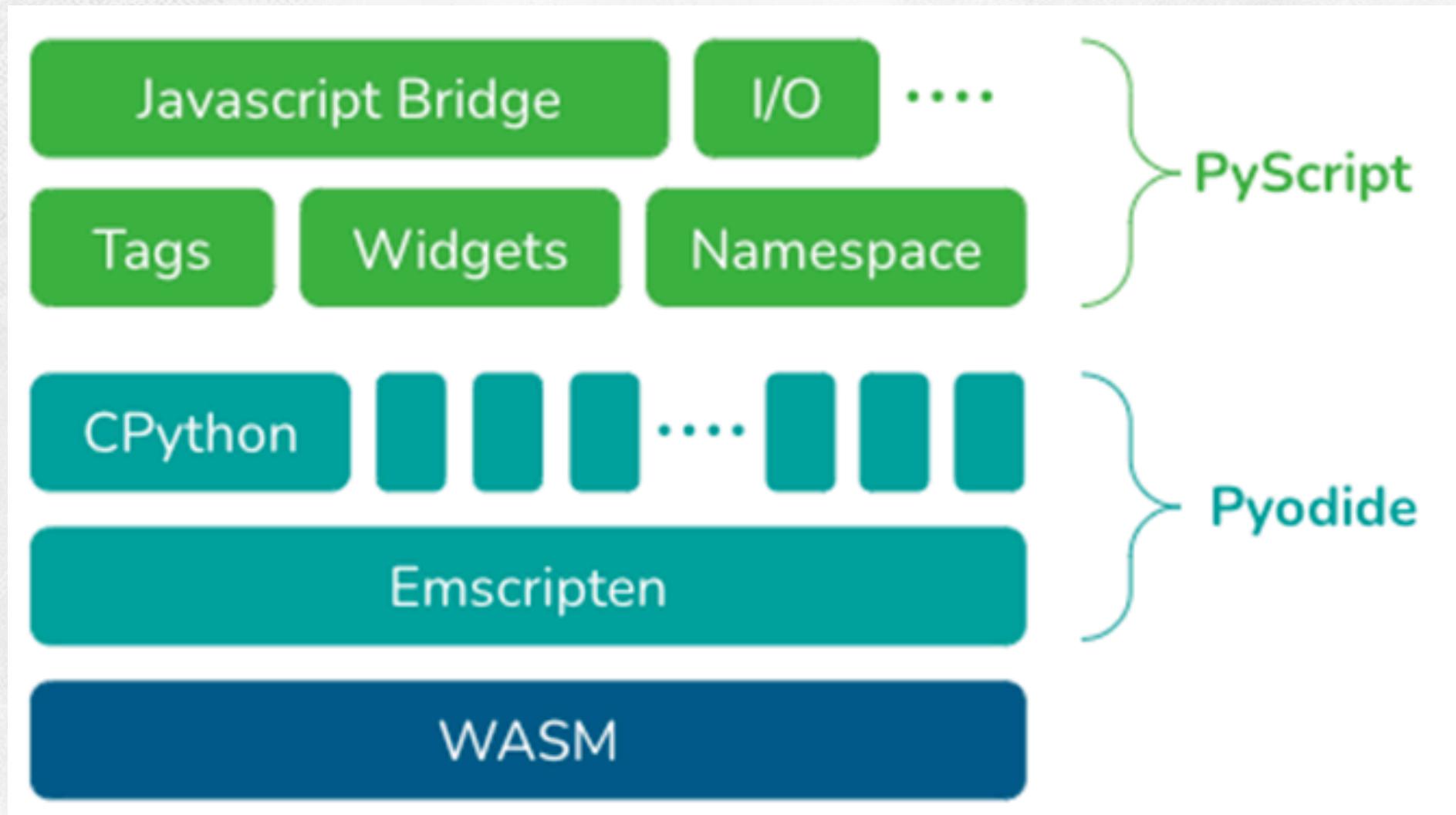
The quickest way to get started with PyScript is by creating a [minimal HTML5 document](#), saving it in a local file such as `hello.html`, and leveraging the two required files hosted on PyScript’s home page:

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.

Python over WebBrowser (WASM)



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