

# Get Started with Web Assembly and Rust

Diwanshi Pandey |  @diwanship |  @diwanshi



# Web Assembly

What

Why

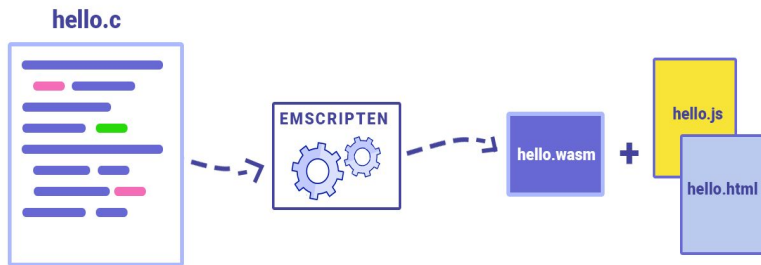
How

# What is WebAssembly?

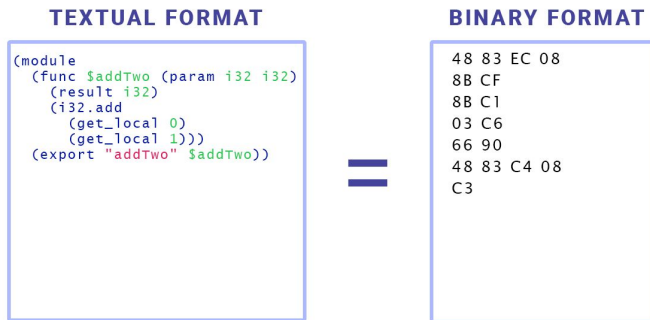
WebAssembly (abbreviated *Wasm*) is a binary instruction format for a stack-based virtual machine. Wasm is designed as a portable target for compilation of high-level languages like C/C++/Rust, enabling deployment on the web for client and server applications.

# In layman's terms

WebAssembly is a new type of code that can be run in modern web browsers and provides new features and major gains in performance.



Compiling C/C++ code to WebAssembly with Emscripten



WebAssembly textual and binary format

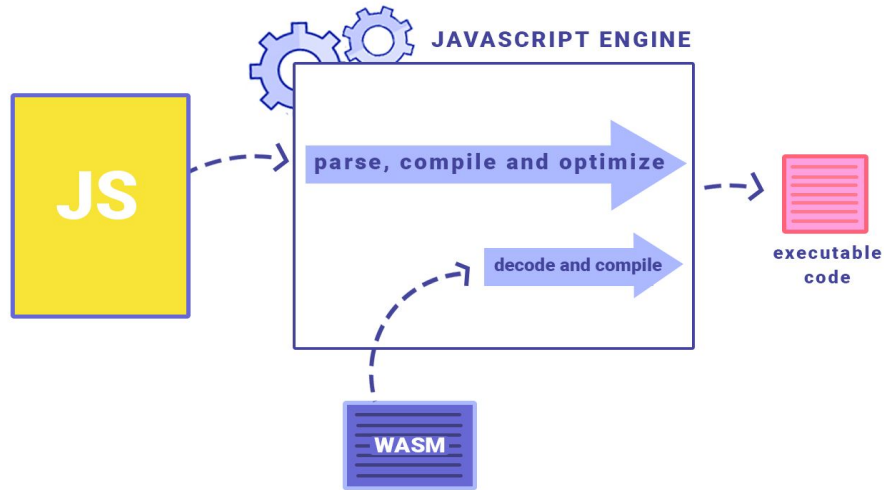
# Why we need it?

Speed

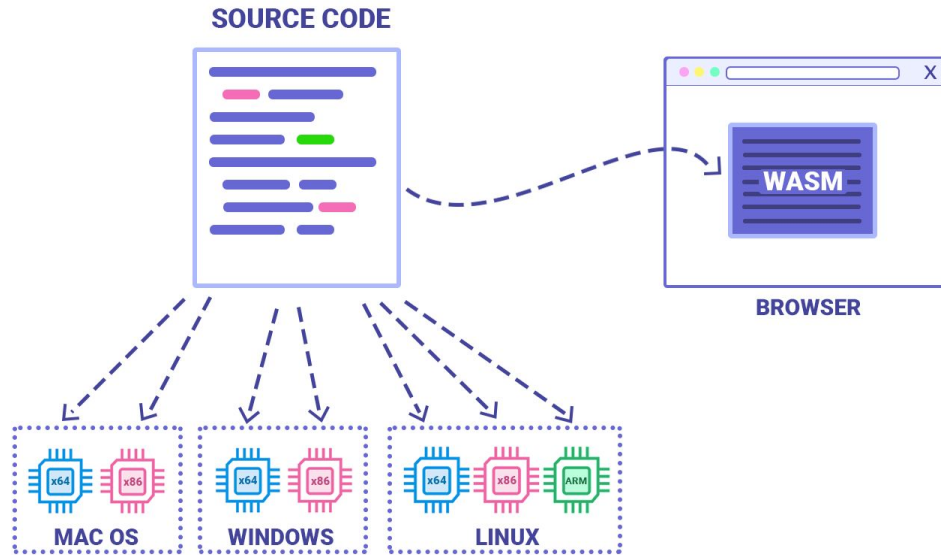
Portability

Flexibility

# Speed



# Portability & Flexibility



# Scenarios where WASM can be really efficient

Video Games

Video Editing

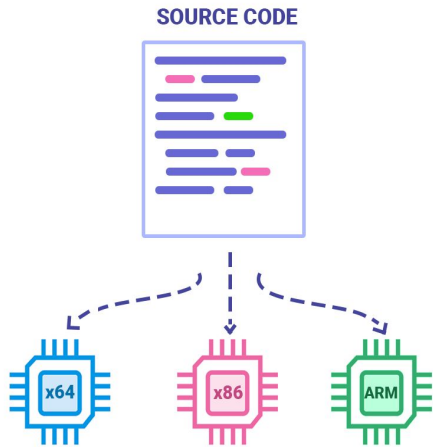
3D rendering

Music Production

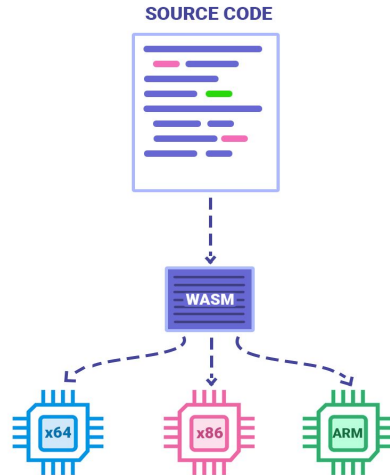
Artificial Intelligence



# How it works

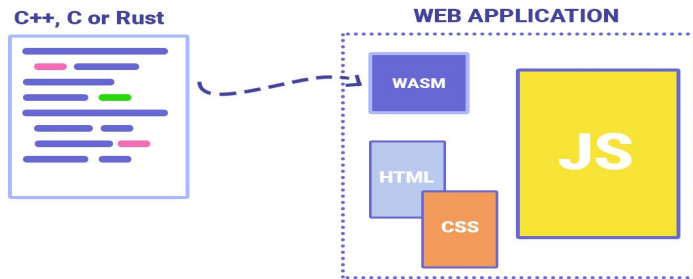


Compiling source code for different processor architectures



WebAssembly as an intermediary compiler target

# How it works



# How can we use it?

Write → Compile → Include → Run

# Demo using Rust

A basic Rust program running in the browser.

Tools Used:

**cargo** - package manager for Rust

**wasm-pack** - compiles rust code to wasm and produce packaging for npm

**wasm-bindgen** - provides a bridge between the types of JavaScript and Rust

```
Diwanshis-MacBook-Pro:~ dpandey$ cargo new --lib hello-wasm
```

```
Created library `hello-wasm` package
```

```
Diwanshis-MacBook-Pro:~ dpandey$ cd hello-wasm/
```

```
Diwanshis-MacBook-Pro:hello-wasm dpandey$ ls
```

```
Cargo.toml      src
```

```
Diwanshis-MacBook-Pro:hello-wasm dpandey$ ls -R
```

```
Cargo.toml      src
```

```
./src:
```

```
lib.rs
```

```
Diwanshis-MacBook-Pro:hello-wasm dpandey$
```

## EXPLORER

## ▲ OPEN EDITORS

Cargo.toml

U

## ▲ HELLO-WASM

src

lib.rs

U

.gitignore

U

Cargo.toml

U

## Cargo.toml ×

```
1  [package]
2  name = "hello-wasm"
3  version = "0.1.0"
4  authors = ["Diwanshi Pandey <diwanshipandey@gmail.com>"]
5  description = "A sample project with wasm-pack"
6  license = "MIT/Apache-2.0"
7  repository = "https://github.com/diwanshi/hello-wasm"
8
9  [lib]
10 crate-type = ["cdylib"]
11
12 [dependencies]
13 wasm-bindgen = "0.2"
```



## EXPLORER

## OPEN EDITORS

index.html rust-wasm-www

lib.rs hello-wasm/src U

## UNTITLED (WORKSPACE)

rust-wasm-www

node\_modules

index.html

index.js

package-lock.json

package.json

webpack.config.js

hello-wasm

pkg

src

lib.rs U

target

.gitignore U

Cargo.lock

Cargo.toml U

wasm-rust-triangle

wasm-rust-triangle-www



index.html

lib.rs



```
1 extern crate wasm_bindgen;
2
3 use wasm_bindgen::prelude::*;
4
5 #[wasm_bindgen]
6 extern {
7     pub fn alert(s: &str);
8 }
9
10 #[wasm_bindgen]
11 pub fn greet(name: &str) {
12     alert(&format!("Hello, {}! \n Welcome to COSCUP 2019 Taiwan", name));
13 }
```

Diwanshis-MacBook-Pro:hello-wasm dpandey\$ wasm-pack build --scope diwanshipandey

[1/10] 🦀 Checking `rustc` version...

[2/10] 🔧 Checking crate configuration...

[3/10] 🎯 Adding WASM target...

info: component 'rust-std' for target 'wasm32-unknown-unknown' is up to date

[4/10] 🌀 Compiling to WASM...

Updating crates.io index

Compiling proc-macro2 v0.4.29

Compiling unicode-xid v0.1.0

Compiling syn v0.15.33

Compiling wasm-bindgen-shared v0.2.43

Compiling cfg-if v0.1.7

Compiling lazy\_static v1.3.0

Compiling bumpalo v2.4.1

Compiling wasm-bindgen v0.2.43

Compiling log v0.4.6

Compiling quote v0.6.12

Compiling wasm-bindgen-backend v0.2.43

Compiling wasm-bindgen-macro-support v0.2.43

Compiling wasm-bindgen-macro v0.2.43

Compiling hello-wasm v0.1.0 (/Users/dpandey/hello-wasm)

Finished release [optimized] target(s) in 44.95s

[5/10] 📁 Creating a pkg directory...

[6/10] 📄 Writing a package.json...

[7/10] 👤 Copying over your README...

⚠️ [WARN]: origin crate has no README

[8/10] 👤 Copying over your LICENSE...

📄 [INFO]: License key is set in Cargo.toml but no LICENSE file(s) were found; Please add the LICENSE file(s) to your project directory

[9/10] ⬇️ Installing wasm-bindgen...

[10/10] 🏃 Running WASM-bindgen...

🌟 Done in 45 seconds

📦 Your wasm pkg is ready to publish at ./pkg.



```
Diwanshis-MacBook-Pro:hello-wasm dpandey$ ls
```

```
Cargo.lock    Cargo.toml    pkg           src           target
```

```
Diwanshis-MacBook-Pro:hello-wasm dpandey$ ls pkg
```

```
hello_wasm.d.ts    hello_wasm_bg.d.ts    package.json
```

```
hello_wasm.js      hello_wasm_bg.wasm
```

```
Diwanshis-MacBook-Pro:hello-wasm dpandey$ █
```



EXPLORER

hello\_wasm\_bg.wasm x



OPEN EDITORS

hello\_wasm\_bg.wasm pkg

HELLO-WASM

pkg

.gitignore

hello\_wasm\_bg.d.ts

hello\_wasm\_bg.wasm

hello\_wasm.d.ts

hello\_wasm.js

package.json

src

target

.gitignore

Cargo.lock

Cargo.toml

The file will not be displayed in the editor because it is either binary, very large or uses an unsupported text encoding.

## EXPLORER

## OPEN EDITORS

JS hello\_wasm.js pkg

## HELLO-WASM

## pkg

.gitignore

TS hello\_wasm\_bg.d.ts

WA hello\_wasm\_bg.wasm

TS hello\_wasm.d.ts

JS hello\_wasm.js

package.json

## src

## target

.gitignore

Cargo.lock

Cargo.toml


## GITLENS

JS hello\_wasm.js x

```
1 import * as wasm from './hello_wasm_bg';
2
3 let cachedTextDecoder = new TextDecoder('utf-8');
4
5 let cachegetUint8Memory = null;
6 function getUint8Memory() {
7     if (cachegetUint8Memory === null || cachegetUint8Memory.buffer !== wasm.memory.buffer) {
8         cachegetUint8Memory = new Uint8Array(wasm.memory.buffer);
9     }
10    return cachegetUint8Memory;
11 }
12
13 function getStringFromWasm(ptr, len) {
14     return cachedTextDecoder.decode(getUint8Memory().subarray(ptr, ptr + len));
15 }
16
17 export function __wbg_alert_a5a2f68cc09adc6e(arg0, arg1) {
18     let varg0 = getStringFromWasm(arg0, arg1);
19     alert(varg0);
20 }
21
22 let WASM_VECTOR_LEN = 0;
23
24 let cachedTextEncoder = new TextEncoder('utf-8');
25
26 let passStringToWasm;
27 if (typeof cachedTextEncoder.encodeInto === 'function') {
28     passStringToWasm = function(arg) {
29
30         let size = arg.length;
31         let ptr = wasm.__wbindgen_malloc(size);
32         let writeOffset = 0;
33         while (true) {
34             const view = getUint8Memory().subarray(ptr + writeOffset, ptr + size);
35             const { read, written } = cachedTextEncoder.encodeInto(arg, view);
```





## EXPLORER

## OPEN EDITORS

 hello\_wasm.d.ts pkg




## HELLO-WASM

## pkg

 .gitignore hello\_wasm\_bg.d.ts hello\_wasm\_bg.wasm hello\_wasm.d.ts hello\_wasm.js package.json

## src

## target

 .gitignore Cargo.lock Cargo.toml hello\_wasm.d.ts x

```
1  /* tslint:disable */
2  /**
3   * @param {string} name
4   * @returns {void}
5   */
6  export function greet(name: string): void;
7
```

Diwanshis-MacBook-Pro:pkg dpandey\$ npm publish --access=public

npm notice

npm notice  @diwanshipandey/hello-wasm@0.1.0

npm notice === Tarball Contents ===

npm notice 488B package.json

npm notice 21.2kB hello\_wasm\_bg.wasm

npm notice 114B hello\_wasm.d.ts

npm notice 2.2kB hello\_wasm.js

npm notice === Tarball Details ===

npm notice name: @diwanshipandey/hello-wasm

npm notice version: 0.1.0

npm notice package size: 10.2 kB

npm notice unpacked size: 24.1 kB

npm notice shasum: 27ab923d5984fb1399d43ac26413b9ab0bbb0141

npm notice integrity: sha512-pb3CBPnF54rak[...]qHUV0mxqN7s0Q==

npm notice total files: 4

npm notice

+ @diwanshipandey/hello-wasm@0.1.0

Diwanshis-MacBook-Pro:pkg dpandey\$



## EXPLORER

## ▲ OPEN EDITORS

package.json rust-wasm-www 1

## ▲ UNTITLED (WORKSPACE)

▶ ● hello-wasm ●

▲ ○ rust-wasm-www ●

package.json 1

package.json x

```
1 {
2   "scripts": {
3     "serve": "webpack-dev-server"
4   },
5   "dependencies": {
6     "@diwanshipandey/hello-wasm": "^0.1.0"
7   },
8   "devDependencies": {
9     "webpack": "^4.25.1",
10    "webpack-cli": "^3.1.2",
11    "webpack-dev-server": "^3.1.10"
12  }
13 }
```



## EXPLORER

## ▲ OPEN EDITORS

webpack.config.js rust-wasm-www

## ▲ UNTITLED (WORKSPACE)

▶ ● hello-wasm ●

▲ ○ rust-wasm-www

package.json

webpack.config.js

webpack.config.js x

```
1  const path = require('path');
2  module.exports = {
3    entry: './index.js',
4    output: {
5      path: path.resolve(__dirname, 'dist'),
6      filename: 'index.js',
7    },
8    mode: 'development'
9  };
```



## EXPLORER

## OPEN EDITORS

index.html rust-wasm-www

## UNTITLED (WORKSPACE)

rust-wasm-www

node\_modules

index.html

index.js

package-lock.json

package.json

webpack.config.js

hello-wasm

wasm-rust-triangle

wasm-rust-triangle-www

index.html x

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>hello-wasm example for COSCUP 2019 Taiwan</title>
6   </head>
7   <body>
8     <h2 style="text-align: center">你好 COSCUP 2019</h2>
9     <h4 style="text-align: center">
10       这只是谷歌翻译欢迎信息的一种愚蠢和天真的尝试。请注意下面的警告框，这是我们在浏览器上运行的RUST代码
11     </h4>
12     <script src="./index.js"></script>
13   </body>
14 </html>
```





## EXPLORER

## OPEN EDITORS

JS index.js rust-wasm-www

## UNTITLED (WORKSPACE)

rust-wasm-www

node\_modules

index.html

JS index.js

package-lock.json

package.json

webpack.config.js

hello-wasm

wasm-rust-triangle

wasm-rust-triangle-www

JS index.js x

```
1  const js = import("./node_modules/@diwanshipandey/hello-wasm/hello_wasm.js");
2  js.then(js => {
3    |   js.greet("Fellow Open Source Warriors!!!");
4  });
```

```
Diwanshis-MacBook-Pro:~ dpandey$ cd rust-wasm-www/  
Diwanshis-MacBook-Pro:rust-wasm-www dpandey$ npm install
```

```
> fsevents@1.2.9 install /Users/dpandey/rust-wasm-www/node_modules/fsevents  
> node install
```

```
node-pre-gyp WARN Using needle for node-pre-gyp https download  
[fsevents] Success: "/Users/dpandey/rust-wasm-www/node_modules/fsevents/lib/binding/Release/node-v57-darwin-x64/fse.node" is installed via remote
```

```
> webpack-cli@3.3.2 postinstall /Users/dpandey/rust-wasm-www/node_modules/webpack-cli  
> node ./bin/opencollective.js
```

Thanks for using Webpack!  
Please consider donating to our [Open Collective](#)  
to help us maintain this package.

👉 Donate: <https://opencollective.com/webpack/donate>

```
npm notice created a lockfile as package-lock.json. You should commit this file.  
npm WARN rust-wasm-www No description  
npm WARN rust-wasm-www No repository field.  
npm WARN rust-wasm-www No license field.
```

```
added 596 packages from 363 contributors and audited 8679 packages in 29.909s  
found 0 vulnerabilities
```

```
Diwanshis-MacBook-Pro:rust-wasm-www dpandey$ ls  
index.html      node_modules    package.json  
index.js        package-lock.json  webpack.config.js  
Diwanshis-MacBook-Pro:rust-wasm-www dpandey$
```

```
.../www — one@api-one-portal:~/one-portal-server — node - npm TERM_PROGRAM=Apple_Terminal TERM=xterm-256color SHELL=/bin/bash
Diwanshis-MacBook-Pro:rust-wasm-www dpandey$ npm run serve

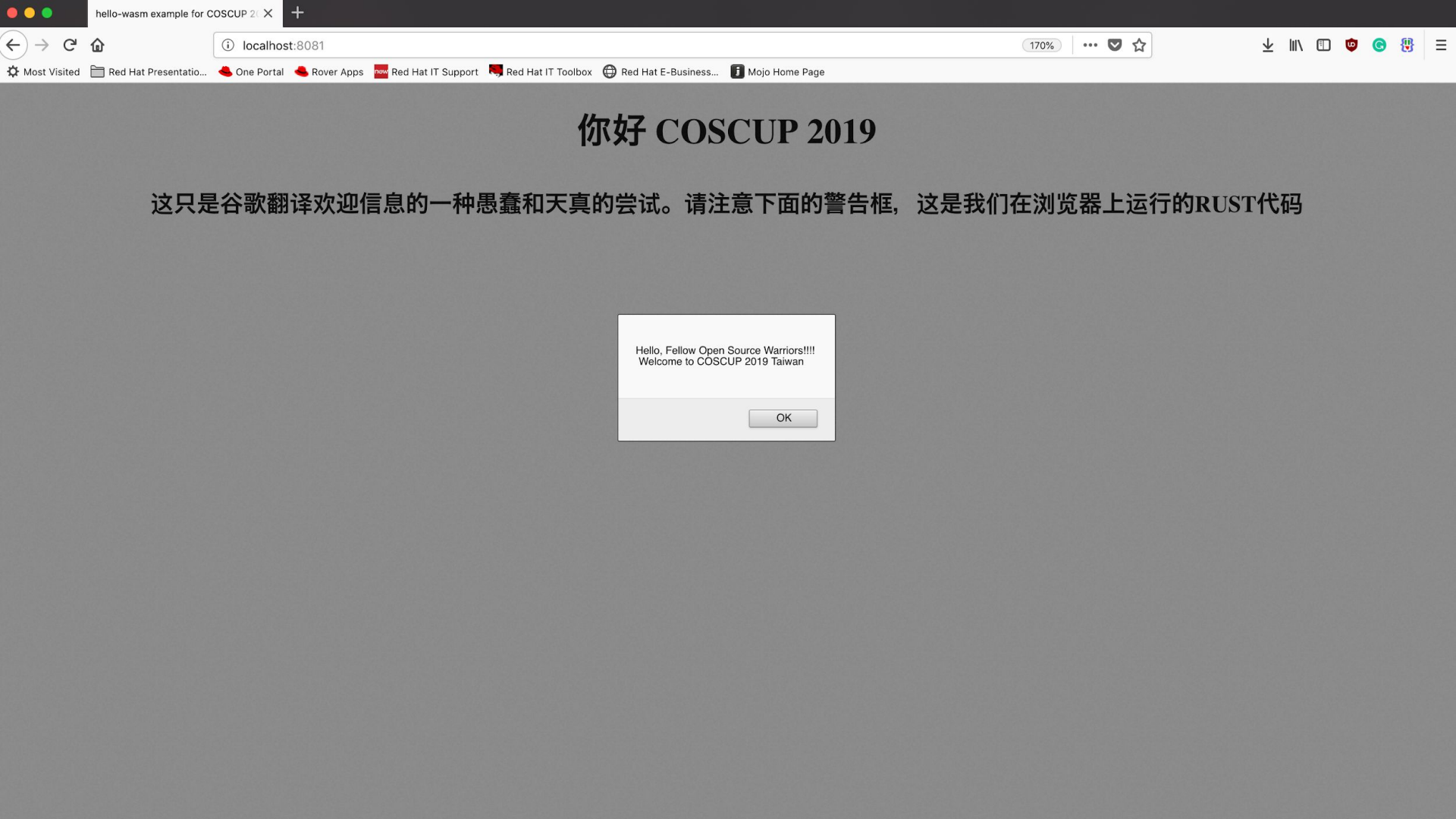
> @ serve /Users/dpandey/rust-wasm-www
> webpack-dev-server

i [wds]: Project is running at http://localhost:8080/
i [wds]: webpack output is served from /
i [wdm]: Hash: 1273d38d004298cedfa0
Version: webpack 4.30.0
Time: 644ms
Built at: 2019-05-08 00:08:42

      Asset      Size  Chunks  Chunk Names
0.index.js      4.82 KiB      0  [emitted]
87aeab2d36370a3271f7.module.wasm unknown size      0  [emitted]
index.js        355 KiB    main  [emitted]  main

Entrypoint main = index.js
[0] multi (webpack)-dev-server/client?http://localhost ./index.js 40 bytes {main} [built]
[./index.js] 130 bytes {main} [built]
[./node_modules/@diwanshipandey/hello-wasm/hello_wasm.js] 2.18 KiB {0} [built]
[./node_modules/@diwanshipandey/hello-wasm/hello_wasm_bg.wasm] 20.7 KiB {0} [built]
[./node_modules/ansi-html/index.js] 4.16 KiB {main} [built]
[./node_modules/events/events.js] 13.3 KiB {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 7.68 KiB {main} [built]
[./node_modules/querystring-es3/index.js] 127 bytes {main} [built]
[./node_modules/url/url.js] 22.8 KiB {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost] (webpack)-dev-server/client?http://localhost 8.26 KiB {main} [built]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.59 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.05 KiB {main} [built]
[./node_modules/webpack-dev-server/node_modules/strip-ansi/index.js] (webpack)-dev-server/node_modules/strip-ansi/index.js 161 bytes {main} [built]
[./node_modules/webpack/hot sync ^\\.\\.log$] (webpack)/hot sync nonrecursive ^\\.\\.log$ 170 bytes {main} [built]
[./node_modules/webpack/hot/emitter.js] (webpack)/hot/emitter.js 75 bytes {main} [built]
+ 13 hidden modules

i [wdm]: Compiled successfully.
```



# 你好 COSCUP 2019

这只是谷歌翻译欢迎信息的一种愚蠢和天真的尝试。请注意下面的警告框，这是我们在浏览器上运行的RUST代码

Hello, Fellow Open Source Warriors!!!!  
Welcome to COSCUP 2019 Taiwan

OK



## EXPLORER

## OPEN EDITORS

index.html rust-wasm-www

lib.rs hello-wasm/src U

## UNTITLED (WORKSPACE)

rust-wasm-www

node\_modules

index.html

index.js

package-lock.json

package.json

webpack.config.js

hello-wasm

pkg

src

lib.rs U

target

.gitignore U

Cargo.lock

Cargo.toml U

wasm-rust-triangle

wasm-rust-triangle-www



index.html

lib.rs



```
1  extern crate wasm_bindgen;
2
3  use wasm_bindgen::prelude::*;
4
5  #[wasm_bindgen]
6  extern {
7      pub fn alert(s: &str);
8  }
9
10 #[wasm_bindgen]
11 pub fn greet(name: &str) {
12     alert(&format!("Hello, {}! \n Welcome to COSCUP 2019 Taiwan", name));
13 }
```

# Rust and WASM Resources

Rust and WASM book: <https://rustwasm.github.io/docs/book/>

Image Credits: <https://blog.logrocket.com/webassembly-how-and-why-559b7f96cd71>

Compiling Rust to WASM: [https://developer.mozilla.org/en-US/docs/WebAssembly/Rust\\_to\\_wasm](https://developer.mozilla.org/en-US/docs/WebAssembly/Rust_to_wasm)

# Lin Clark's Cartoon Intro to WASM



Questions?



# Thank You!

Diwanshi Pandey |  @diwanship |  @diwanshi

