WEBASSEMBLY

PÄR WENÅKER

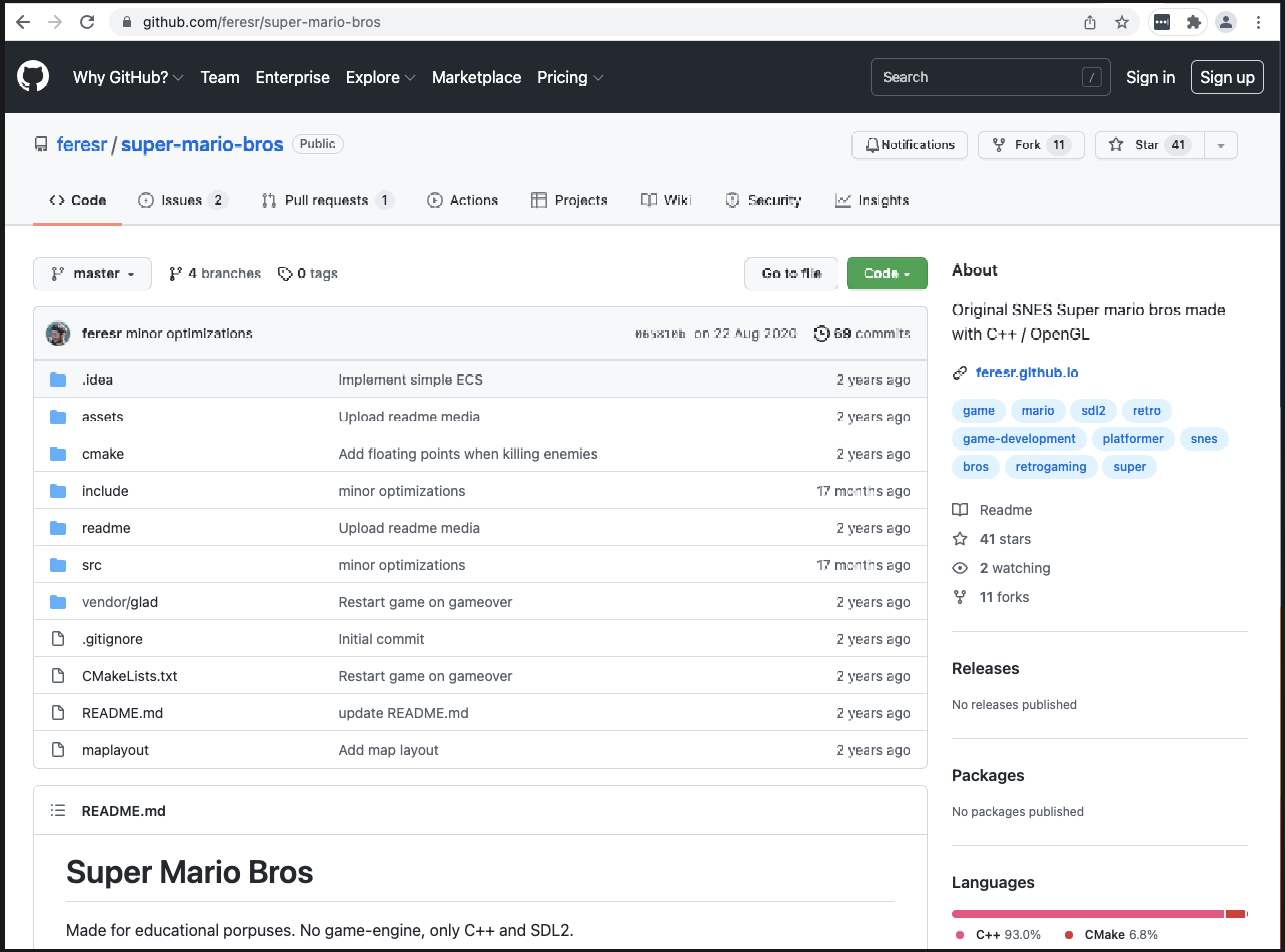
CADEC 2020.02.02 | CALLISTAENTERPRISE.SE

CALLISTA

What if you have a program not written in JavaScript and want to run it on the web?

CALLISTA

DEMO



github.com/feresr/super-mario-bros

feresr / super-mario-bros Public

4 branches 0 tags

feresr minor optimizations 065810b on 22 Aug 2020 69 commits

.idea	Implement simple ECS	2 years ago
assets	Upload readme media	2 years ago
cmake	Add floating points when killing enemies	2 years ago
include	minor optimizations	17 months ago
readme	Upload readme media	2 years ago
src	minor optimizations	17 months ago
vendor/glad	Restart game on gameover	2 years ago
.gitignore	Initial commit	2 years ago
CMakeLists.txt	Restart game on gameover	2 years ago
README.md	update README.md	2 years ago
maplayout	Add map layout	2 years ago

README.md

Super Mario Bros

Made for educational porpuses. No game-engine, only C++ and SDL2.

About

Original SNES Super mario bros made with C++ / OpenGL

feresr.github.io

game mario sdl2 retro game-development platformer snes bros retrogaming super

Readme
41 stars
2 watching
11 forks

Releases

No releases published

Packages

No packages published

Languages

● C++ 93.0% ● CMake 6.8%

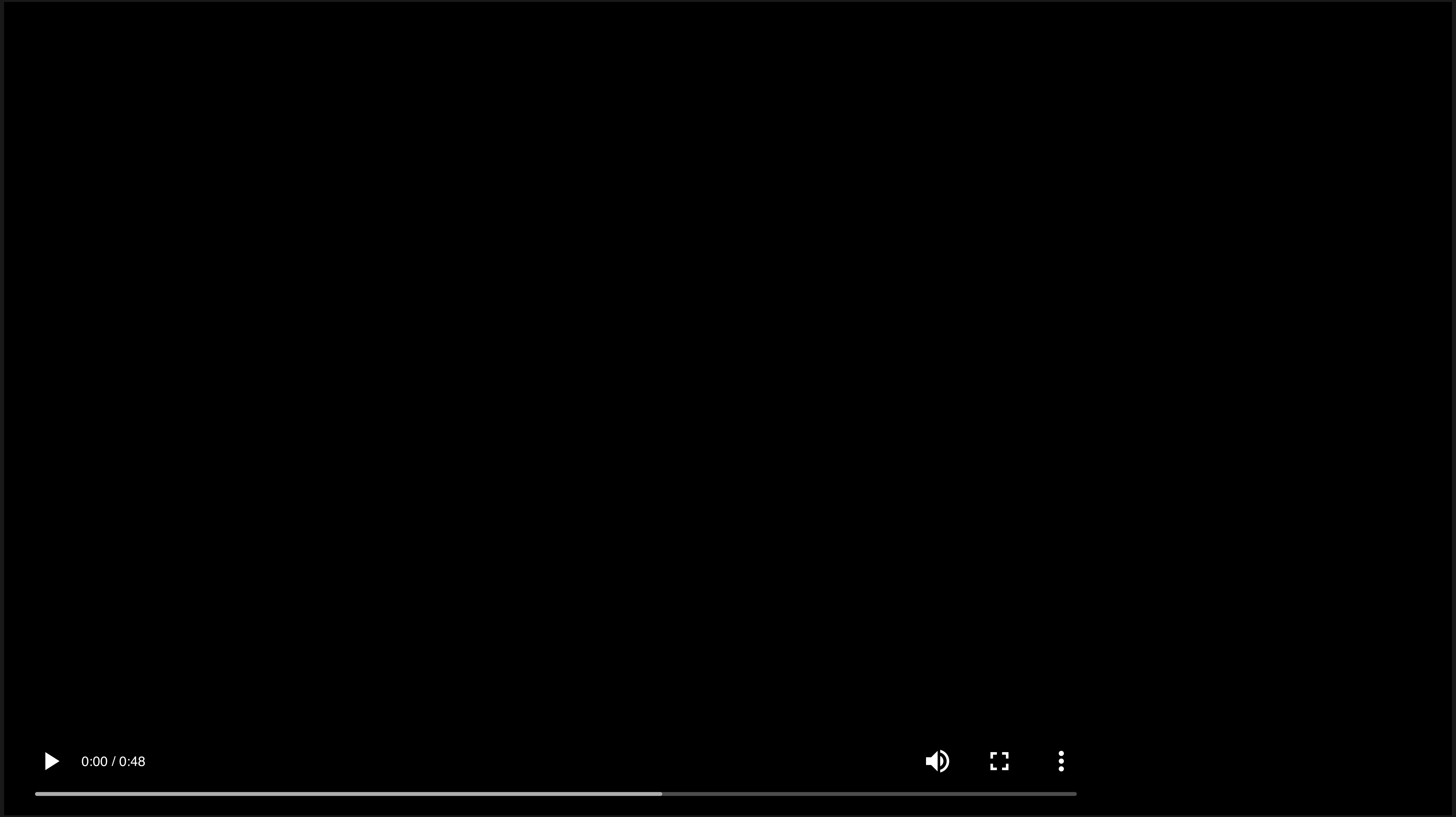
https://github.com/feresr/super-mario-bros

CALLISTA

DEMO - SUPER MARIO BROS

CALLISTA

DEMO - SUPER MARIO BROS



CALLISTA

DEMO - SUPER MARIO BROS

CALLISTA

BACKGROUND - WEB PLATFORM

BACKGROUND - JAVASCRIPT

JavaScript has performance problems when used for more intense tasks.

CALLISTA

BACKGROUND - STANDARDISATION

Introducing new functionality requires standardization efforts.

BACKGROUND - CODE REUSE

The web platform is a separate target platform and ecosystem.

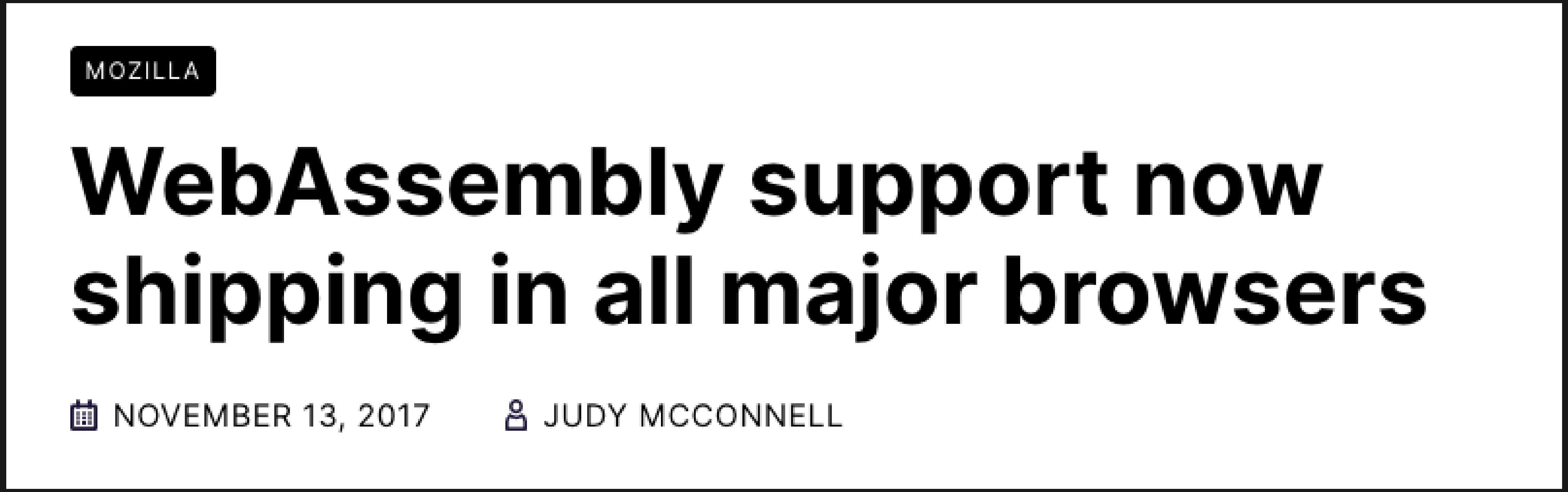
BACKGROUND - HISTORY

June 2015

"Mozilla, Chromium, Edge & Webkit started working on a new standard, WebAssembly, that defines a portable, size- and load-efficient format and execution model specifically designed to serve as a compilation target for the Web."

CALLISTA

November 2017


MOZILLA

WebAssembly support now shipping in all major browsers

NOVEMBER 13, 2017 JUDY MCCONNELL

WHAT IS WEBASSEMBLY ?

```
textinit    ldx #00                 ; init display text
            lda text1, x
            sta charline12, x
            lda text2, x
            sta charline13, x
            inx
            cpx #40
            bne textinit+2

            ldx #00                 ; init colour
            lda initcolourmap1, x
            sta colmapline12, x
            lda initcolourmap2, x
            sta colmapline13, x
            inx
            cpx #40
            bne colourinit+2

            lda #255                ; enable all sprites
            sta spriteenable
            sta spritemulti         ; enable multicolour on all
```

CALLISTA

WHAT IS WEBASSEMBLY? - SPECIFICATION

WebAssembly (WASM) is a specification

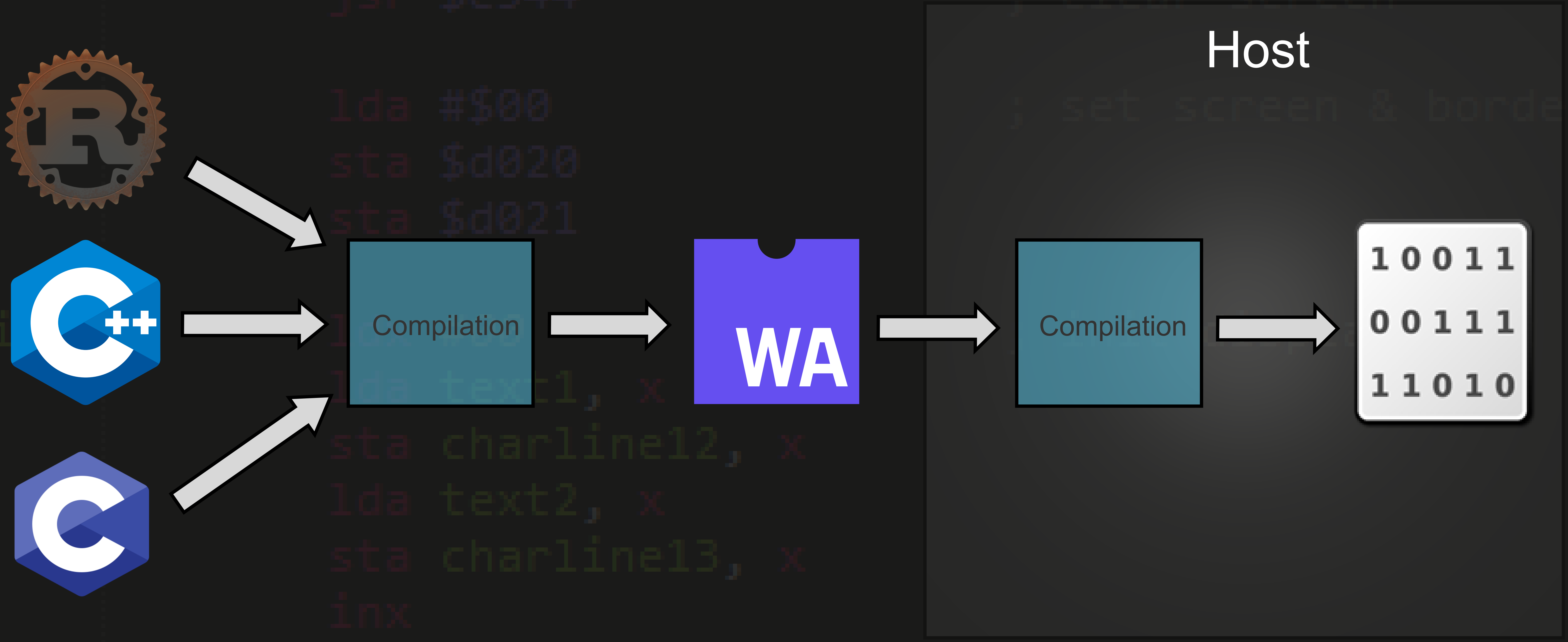
CALLISTA

WHAT IS WEBASSEMBLY? - COMPILATION TARGET

WASM is a compilation target for other programming languages.

CALLISTA

WHAT IS WEBASSEMBLY - COMPILATION CHAIN



```
lda initcolourmap1, x
sta colmapline12, x
lda initcolourmap2, x
sta colmapline13, x
inx
cpx #40
bne colourinit+2

lda #255                        ; enable all sprites
sta spriteenable
sta spritemulti                 ; enable multicolour on a

sei                             ; set up interrupt
lda #$7f
sta $dc0d                       ; turn off the CIA intern
sta $dd0d
and $d011                       ; clear high bit of raste
sta $d011
```

Kompileras när det stömmas

CALLISTA

WHAT IS WEBASSEMBLY - DESIGN GOALS

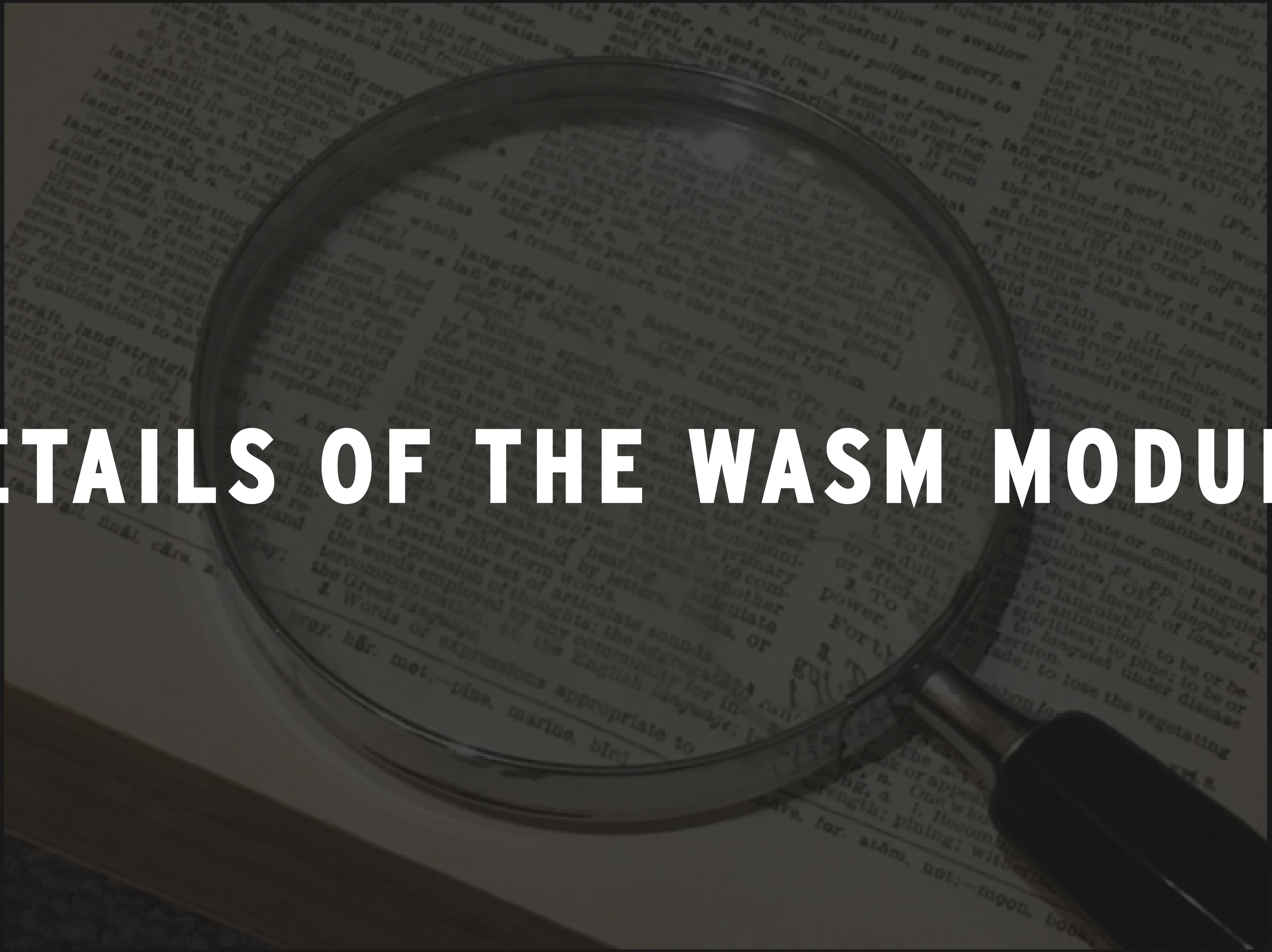
WA

- Portable
- Small
- Fast
- Safe
- Debuggable

W3C SPECIFICATIONS

- Core Specification
  - WebAssembly
- Embedder Specifications
  - JavaScript Embedding
  - Web Embedding

CALLISTA



DETAILS OF THE WASM MODULE

WASM LOW-LEVEL INTRODUCTION

- Type safe
- Low-level instructions
- Export & import functions
- Export & import linear memory
- Data types i32, i64, f32 & f64

CALLISTA

WASM LOW-LEVEL INTRODUCTION

```
import { int inc(v int) } from host;

export int add_inc(a int, b int) {
  return inc(a + b)
}
```

CALLISTA

WASM LOW-LEVEL INTRODUCTION

```
import { inc(v int) } from host;

export int add_inc(a int, b int) {
  return inc(a + b)
}
```

CALLISTA

Wasm Binary Module

```
00000000: 0061 736d 0100 0000 010c 0260 017f 017f .asm.......`....
00000010: 6002 7f7f 017f 020c 0104 686f 7374 0369 `.........host.i
00000020: 6e63 0000 0302 0101 0503 0100 0107 1102 nc..............
00000030: 0761 6464 5f69 6e63 0001 036d 656d 0200 .add_inc...mem..
00000040: 0a0b 0109 0020 0020 016a 1000 0b ..... . .j...
```

Wasm Module Sections

```
Section Details:

Type[2]:
- type[0] (i32) -> i32
- type[1] (i32, i32) -> i32
Import[1]:
- func[0] sig=0 <inc> <- host.inc
Function[1]:
- func[1] sig=1 <add_inc>
Memory[1]:
- memory[0] pages: initial=1
Export[2]:
- func[1] <add_inc> -> "add_inc"
- memory[0] -> "mem"
Code[1]:
- func[1] size=9
```

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY TEXT FORMAT

```
(module
  (type (;0;) (func (param i32) (result i32)))
  (type (;1;) (func (param i32 i32) (result i32)))
  (import "host" "inc" (func (;0;) (type 0)))
  (func (;1;) (type 1) (param i32 i32) (result i32)
    local.get 0
    local.get 1
    i32.add
    call 0)
  (memory (;0;) 1)
  (export "add_inc" (func 1))
  (export "mem" (memory 0))
)
```

CALLISTA

WEBASSEMBLY JAVASCRIPT EMBEDDING

```javascript
var importObj = {
  host: {
    inc: (v) => v + 1,
  }
};

const response = await fetch('add_inc.wasm')
const buffer = await response.arrayBuffer()
const { module, instance } =
  await WebAssembly.instantiate(buffer, importObj)
console.log(instance.exports.add_inc(1, 2))
```

CALLISTA

WEBASSEMBLY JAVASCRIPT EMBEDDING

```javascript
var importObj = {
  host: {
    inc: (v) => v + 1,
  }
};

const response = await fetch('add_inc.wasm')
const buffer = await response.arrayBuffer()
const { module, instance } =
  await WebAssembly.instantiate(buffer, importObj)
console.log(instance.exports.add_inc(1, 2))
```

CALLISTA

WEBASSEMBLY JAVASCRIPT EMBEDDING

```
var importObj = {
  host: {
    inc: (v) => v + 1,
  }
};

const response = await fetch('add_inc.wasm')
const buffer = await response.arrayBuffer()
const { module, instance } =
  await WebAssembly.instantiate(buffer, importObj)
console.log(instance.exports.add_inc(1, 2))
```

CALLISTA

WEBASSEMBLY JAVASCRIPT EMBEDDING

```javascript
var importObj = {
  host: {
    inc: (v) => v + 1,
  }
};

const response = await fetch('add_inc.wasm')
const buffer = await response.arrayBuffer()
const { module, instance } =
  await WebAssembly.instantiate(buffer, importObj)
console.log(instance.exports.add_inc(1, 2))
```

CALLISTA

WEBASSEMBLY - SECURITY

- A WASM module has no access to the host by default.
- The host provides the WASM module capabilities through imports.

CALLISTA

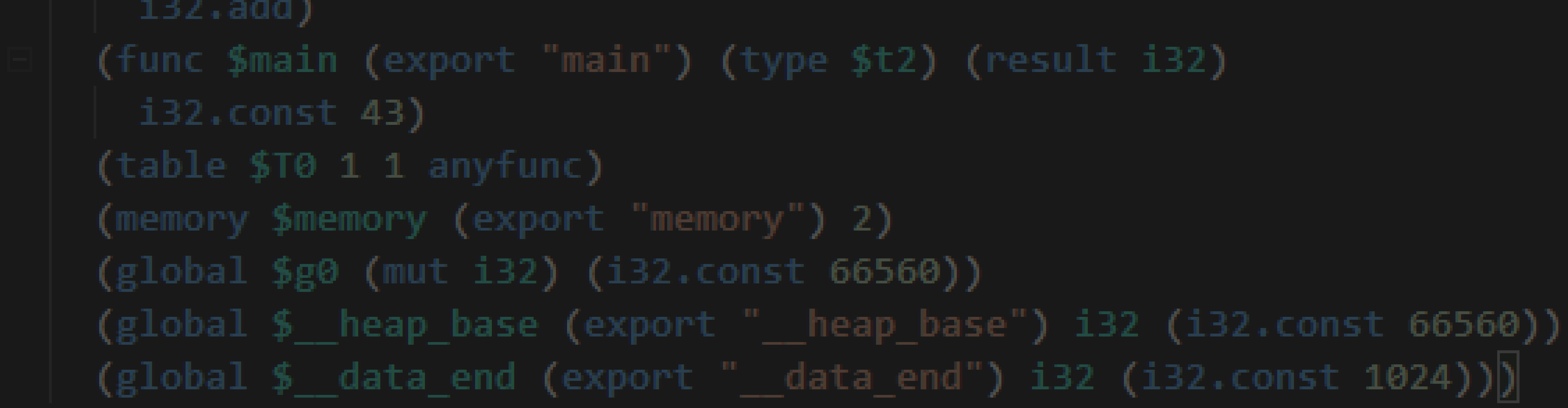
Tool Chains

emscripten

wasm-pack

AssemblyScript

```
(module
  (type $t0 (func))
  (type $t1 (func (param i32 i32) (result i32)))
  (type $t2 (func (result i32)))
  (func $__wasm_call_ctors (type $t0))
  (func $myAdd (export "myAdd") (type $t1) (param $p0 i32) (param $p1 i32) (result i
    get_local $p1
    get_local $p0
```



```
  i32.add)
(func $main (export "main") (type $t2) (result i32)
  i32.const 43)
(table $T0 1 1 anyfunc)
(memory $memory (export "memory") 2)
(global $g0 (mut i32) (i32.const 66560))
(global $__heap_base (export "__heap_base") i32 (i32.const 66560))
(global $__data_end (export "__data_end") i32 (i32.const 1024)))
```

WASI

WebAssembly System Interface

"WebAssembly: Neither Web Nor Assembly"

```
(module
  (type (;0;) (func (param i32)))
  (type (;1;) (func))
  (type (;2;) (func (param i32 i32) (result i32)))
  (import "js" "print" (func (;0;) (type 0)))
  (          ) (type 1)

  (          1)
    i32.cons
    i32.const 0
    i32.load offset=66576
    i32.store offset=1024)
  (func (;3;) (type 2) (param i32 i32) (result i32)
    (local i32)
```

CALLISTA

WASI

```
(module
  (type (;0;) (func (param i32)))
  (type (;1;) (func))
  (type (;2;) (func (param i32 i32) (result i32)))
  (import "js" "print" (func (;0;) (type 0)))
  (func (;1;) (type 1)
    i32.const 0
    i32.const 0
    i32.load offset=66576
    i32.store offset=1024)
  (func (;3;) (type 2) (param i32 i32) (result i32)
    (local i32)
```

WA SI

"Define an abstract and modular operating system that maintains the WASM portability and security model."

WASI

```
(module
  (type (;0;) (func (param i32)))
  (type (;1;) (func))
  (type (;2;) (func (param i32 i32) (result i32)))
  (import "js" "print" (func (;0;) (type 0)))
  (func (;1;) (type 1)

  (                 ype 1)
    i32.co
    i32.const 0
    i32.load offset=66576
    i32.store offset=1024)
  (func (;3;) (type 2) (param i32 i32) (result i32)
    (local i32)
```

WASI

"Define a component model that enables integration between WASM modules."

CALLISTA

BYTECODE ALLIANCE

"...cross-industry collaborative mission to create a secure, performant, cross-platform and cross-device future of computing."

BYTECODE ALLIANCE

Mozilla, Fastly, Intel, and Red Hat
Google, Amazon, Microsoft

APPLICATIONS

CALLISTA

WASM IN THE CLOUD

- Speed

- Speed
- Size

- Speed
- **Security**
- Size

- Speed
- Size
- **Portability**
- Security

- Speed
- Size
- Security
- Portability

fastly®

fastly®

netlify

CLOUD - EDGE

fastly

netlify

CLOUDFLARE

CLOUD - EDGE

fastly

netlify

CLOUDFLARE

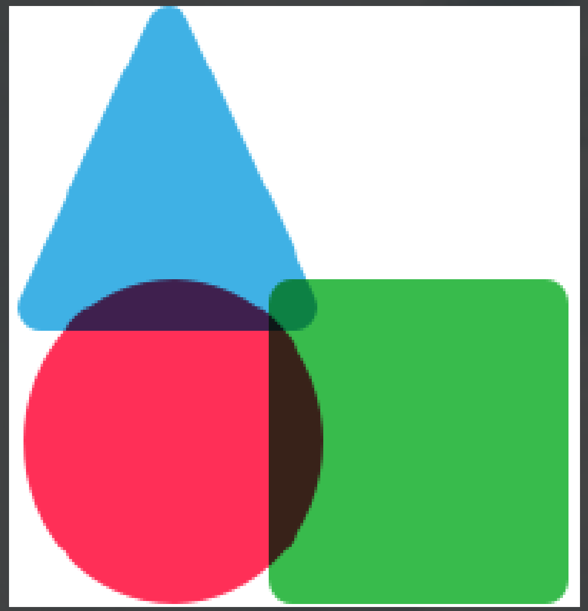
CLOUD - WASM PROXY

Istio

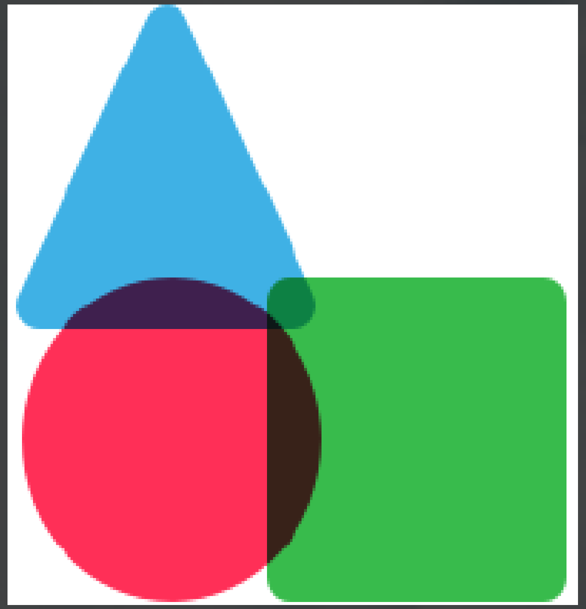
OPENRESTY

envoy

Deis Labs



- Krustlets



- Krustlets
- Hippo

CLOUD - KUBERNETES

- Krustlets
- Hippo
- WAGI

CLOUD - KUBERNETES

- Krustlets
- Hippo
- WAGI

wasmcloud

CLOUD - APPLICATION FRAMEWORKS

wasmcloud

ATMO

JAVA - GRAALVM

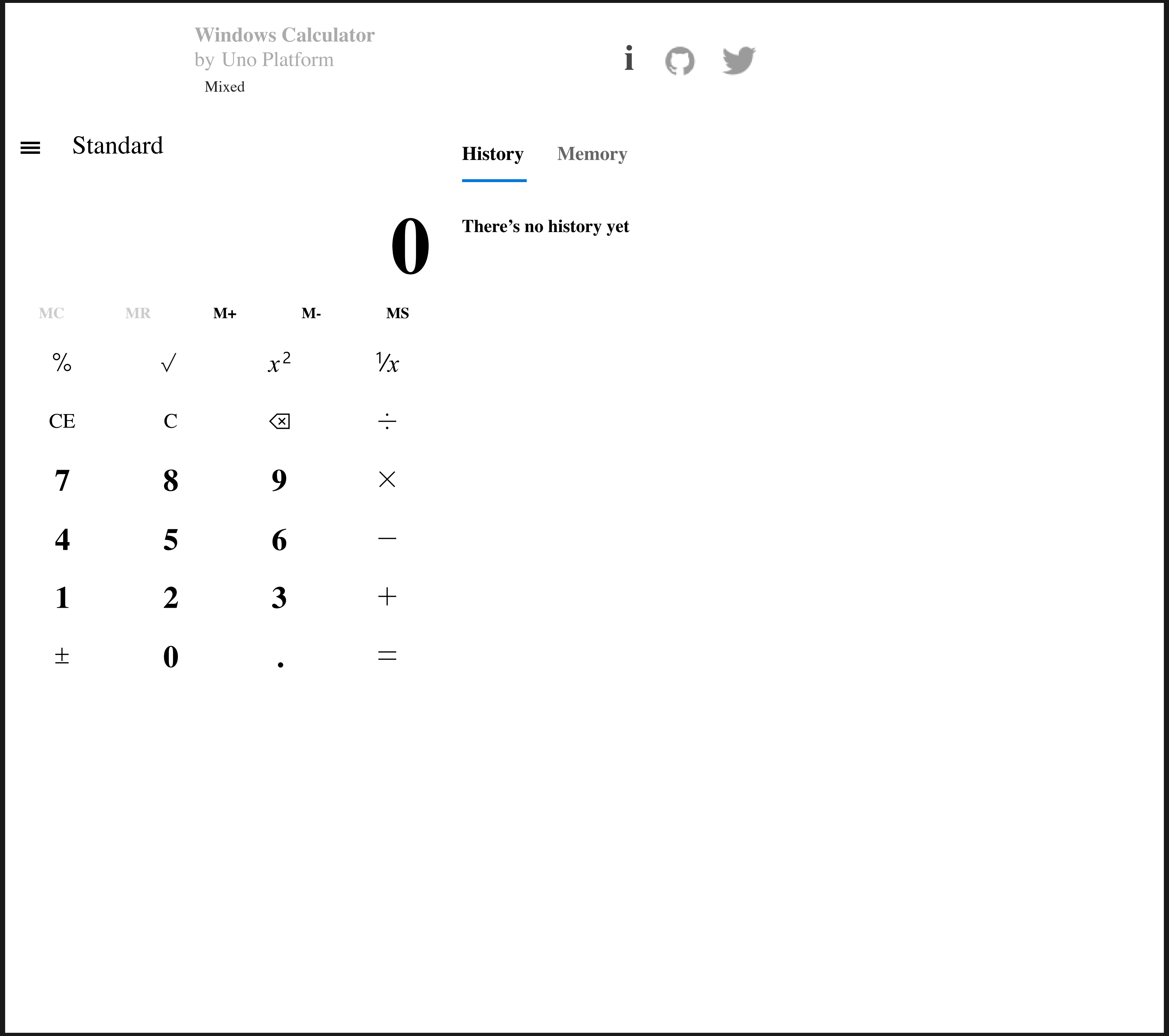
```java
byte[] binary = readAllBytes(new File("floyd.wasm").toPath());
Context.Builder contextBuilder =
  Context.newBuilder("wasm");
Source.Builder sourceBuilder =
  Source.newBuilder("wasm", ByteSequence.create(binary), "floyd");
Source source = sourceBuilder
  .build();
Context context = contextBuilder
  .option("wasm.Builtins", "wasi_snapshot_preview1")
  .build();
context.eval(source);
Value mainFunction = context.getBindings("wasm").getMember("main").getMember("run");
mainFunction.execute();
```

CALLISTA

"The first and only UI Platform for single-codebase applications for Windows, WebAssembly, iOS, macOS, Android and Linux"

https://calculator.platform.uno

UNO PLATFORM



There's no history yet

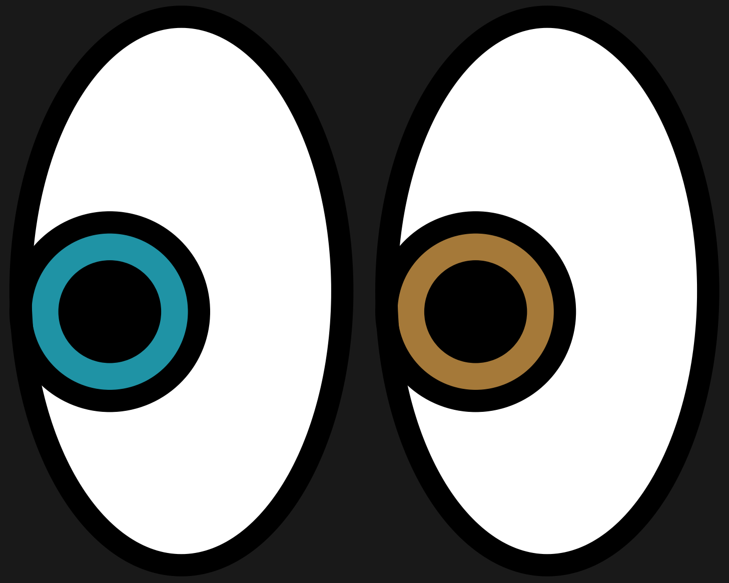
CALLISTA

OTHER

- AutoCad
- Photoshop
- Tensorflow JS
- SQL JS - SqlLite
- 1Password
- Figma

CALLISTA

KEEP AN EYE ON WEBASSEMBLY!



CALLISTA