

WEBASSEMBLY

PÄR WENÅKER

CADEC 2020.02.02 | [CALLISTAENTERPRISE.SE](https://callistaenterprise.se)

CALLISTA



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



The screenshot shows the GitHub interface for the repository 'feresr/super-mario-bros'. The repository is public and has 41 stars, 11 forks, and 2 issues. The main content area displays a file tree with folders like '.idea', 'assets', 'cmake', 'include', 'readme', 'src', and 'vendor/glad', and files like '.gitignore', 'CMakeLists.txt', 'README.md', and 'maplayout'. The README is expanded, showing the title 'Super Mario Bros' and the text 'Made for educational purposes. No game-engine, only C++ and SDL2.' The right sidebar contains an 'About' section with a description, a link to 'feresr.github.io', and tags like 'game', 'mario', 'sdl2', 'retro', 'game-development', 'platformer', 'snes', 'bros', 'retrogaming', and 'super'. It also lists 'Releases' (none published), 'Packages' (none published), and 'Languages' (C++ 93.0%, CMake 6.8%).

github.com/feresr/super-mario-bros

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search / Sign in Sign up

feresr / super-mario-bros Public

Notifications Fork 11 Star 41

<> Code Issues 2 Pull requests 1 Actions Projects Wiki Security Insights

master 4 branches 0 tags Go to file Code

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 purposes. 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

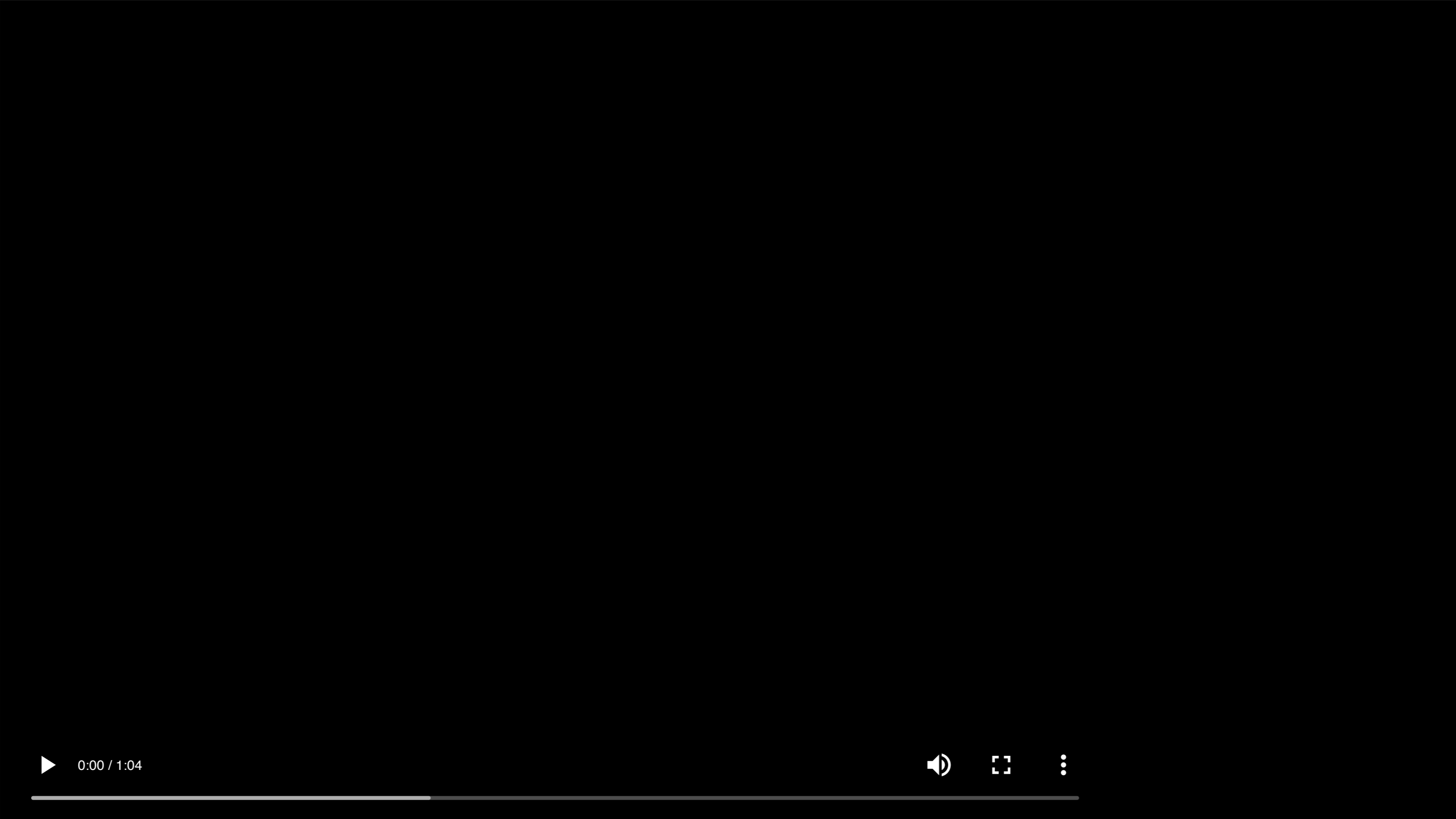
Releases
No releases published

Packages
No packages published

Languages
C++ 93.0% CMake 6.8%

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

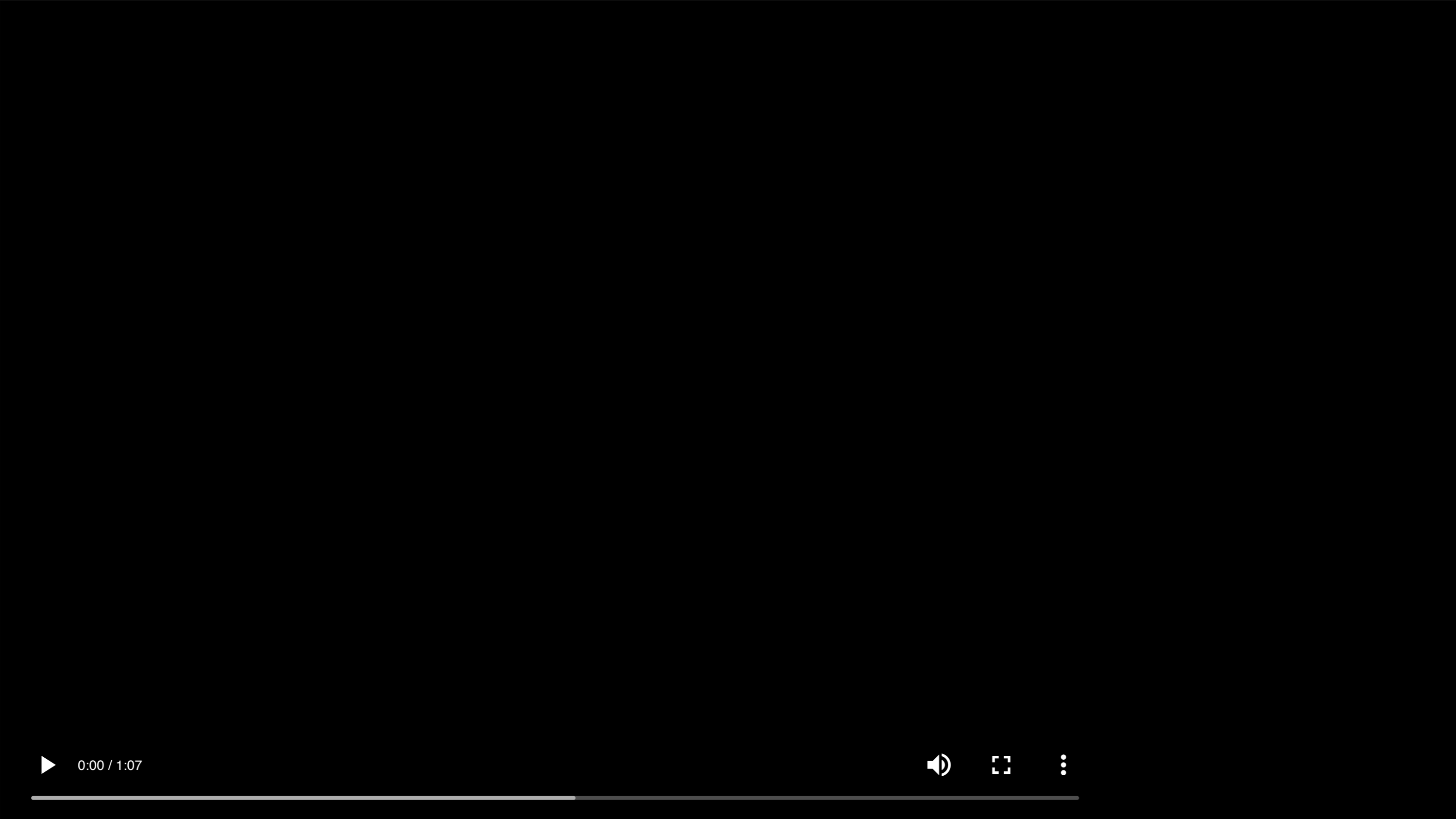
DEMO - SUPER MARIO BROS



DEMO - SUPER MARIO BROS

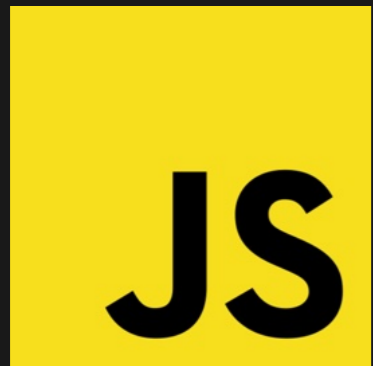


DEMO - SUPER MARIO BROS



BACKGROUND - WEB PLATFORM





JavaScript has performance problems when used for more intense tasks.



Introducing new functionality requires standardization efforts.



The web platform is a separate target platform and ecosystem.

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."

November 2017

MOZILLA

WebAssembly support now shipping in all major browsers

📅 NOVEMBER 13, 2017

👤 JUDY MCCONNELL

WHAT IS WEBASSEMBLY?



WHAT IS WEBASSEMBLY ?

```
30 textinit ldx #00 ; init display text
31 lda text1, x
32 sta charline12, x
33 lda text2, x
34 sta charline13, x
35 inx
36 cpx #40
37 bne textinit+2
38
39 lda initcolourmap1, x
40 sta colmapline12, x
41 lda initcolourmap2, x
42 sta colmapline13, x
43 inx
44 cpx #40
45 bne colourinit+2
46
47
48 lda #255 ; enable all sprites
49 sta spriteenable
50 sta spritemulti ; enable multicolour on all
```

WHAT IS WEBASSEMBLY? - SPECIFICATION



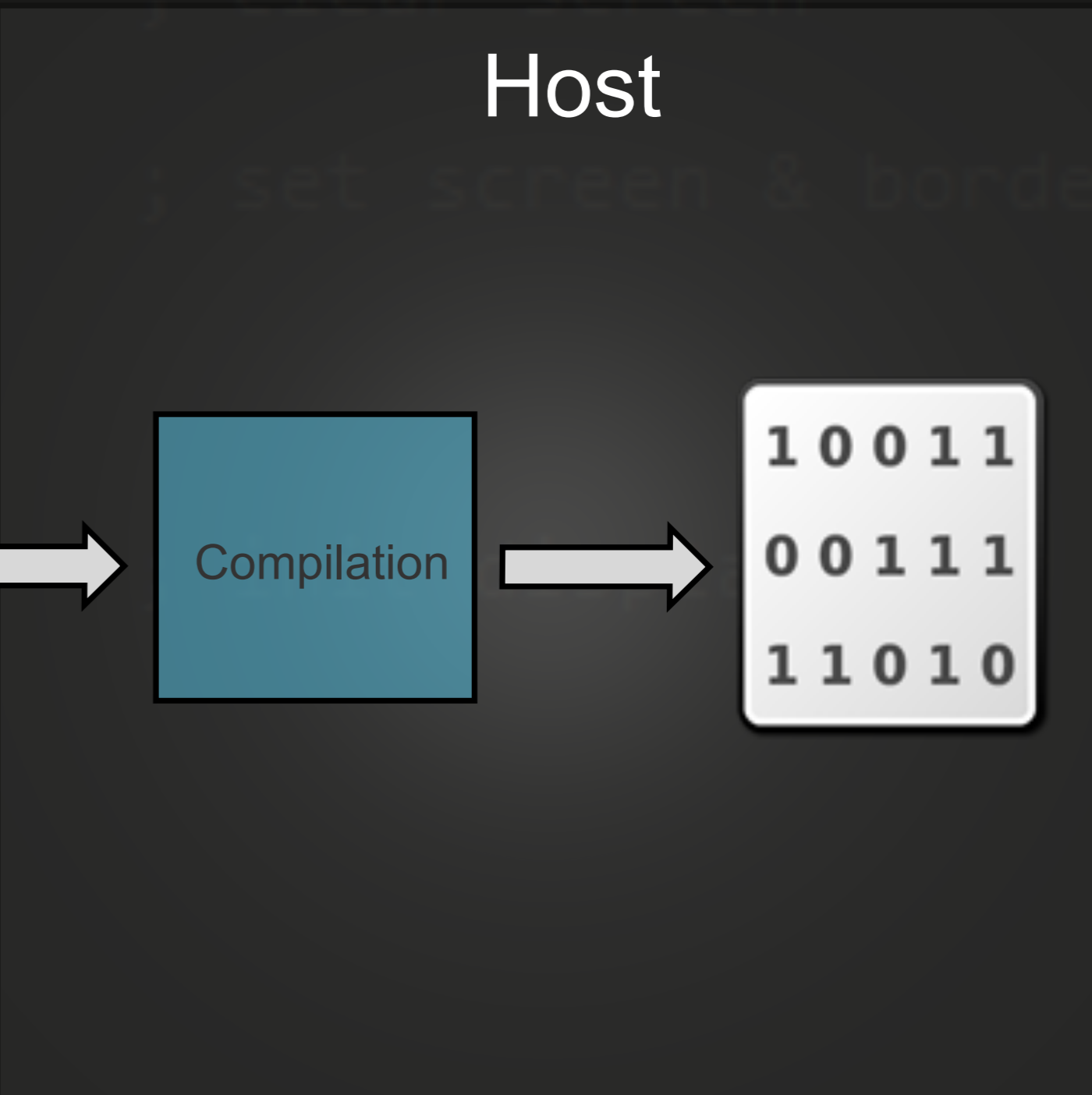
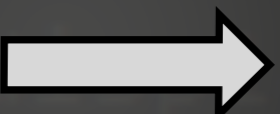
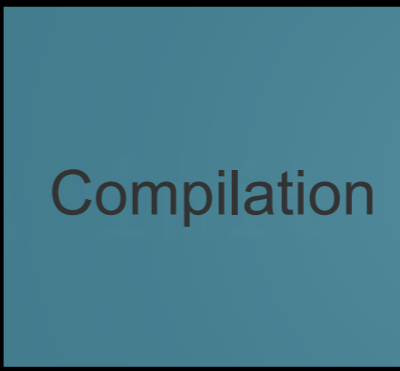
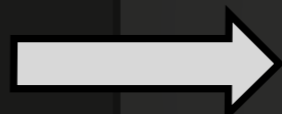
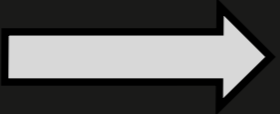
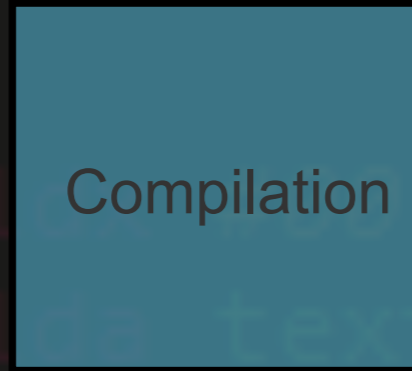
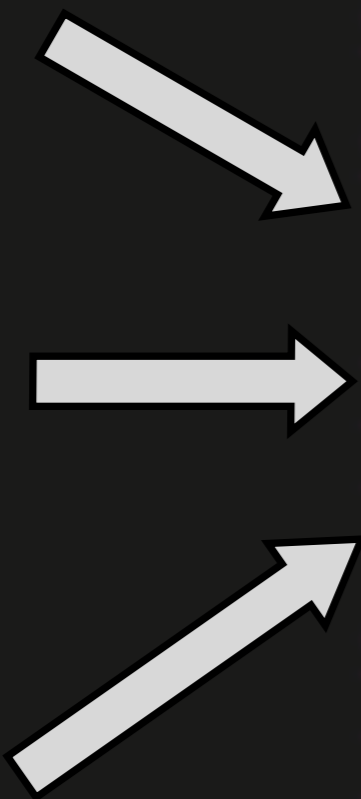
WebAssembly (WASM) is a specification

WHAT IS WEBASSEMBLY? - COMPILATION TARGET



WASM is a compilation target for other programming languages.

WHAT IS WEBASSEMBLY - COMPILATION CHAIN



```
jsr $e544  
lda #$00  
sta $d020  
sta $d021  
text1, x  
sta charline12, x  
lda text2, x  
sta charline13, x  
inx  
cpx #40  
bne textinit+2  
colourinit  
idx #00
```

```
routine_start (49152)  
clear_screen  
set_screen_color  
init_text_colours
```



```
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 interrupt
sta $dd0d
and $d011 ; clear high bit of raster
sta $d011
```

Kompileras när det stömmas

WHAT IS WEBASSEMBLY - DESIGN GOALS



- Portable
- Small
- Fast
- Safe
- Debuggable



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

A magnifying glass is positioned over an open dictionary, focusing on the word 'lang'. The background is a dark, textured surface. The text in the dictionary is partially legible, showing various definitions and etymologies for the word 'lang'.

DETAILS OF THE WASM MODULE



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

WASM LOW-LEVEL INTRODUCTION

```
1 import { int inc(v int) } from host;  
2  
3 export int add_inc(a int, b int) {  
4     return inc(a + b)  
5 }
```

WASM LOW-LEVEL INTRODUCTION

```
1 import { inc(v int) } from host;  
2  
3 export int add_inc(a int, b int) {  
4     return inc(a + b)  
5 }
```

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

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```

WEBASSEMBLY TEXT FORMAT

```
1 (module
2   (type (;0;) (func (param i32) (result i32)))
3   (type (;1;) (func (param i32 i32) (result i32)))
4   (import "host" "inc" (func (;0;) (type 0)))
5   (func (;1;) (type 1) (param i32 i32) (result i32)
6     local.get 0
7     local.get 1
8     i32.add
9     call 0)
10  (memory (;0;) 1)
11  (export "add_inc" (func 1))
12  (export "mem" (memory 0))
13 )
```


WEBASSEMBLY JAVASCRIPT EMBEDDING

```
1 var importObj = {
2   host: {
3     inc: (v) => v + 1,
4   }
5 };
6
7 const response = await fetch('add_inc.wasm')
8 const buffer = await response.arrayBuffer()
9 const { module, instance } =
10   await WebAssembly.instantiate(buffer, importObj)
11 console.log(instance.exports.add_inc(1, 2))
```

WEBASSEMBLY JAVASCRIPT EMBEDDING

```
1 var importObj = {
2   host: {
3     inc: (v) => v + 1,
4   }
5 };
6
7 const response = await fetch('add_inc.wasm')
8 const buffer = await response.arrayBuffer()
9 const { module, instance } =
10   await WebAssembly.instantiate(buffer, importObj)
11 console.log(instance.exports.add_inc(1, 2))
```

WEBASSEMBLY JAVASCRIPT EMBEDDING

```
1 var importObj = {
2   host: {
3     inc: (v) => v + 1,
4   }
5 };
6
7 const response = await fetch('add_inc.wasm')
8 const buffer = await response.arrayBuffer()
9 const { module, instance } =
10   await WebAssembly.instantiate(buffer, importObj)
11 console.log(instance.exports.add_inc(1, 2))
```

WEBASSEMBLY JAVASCRIPT EMBEDDING

```
1 var importObj = {
2   host: {
3     inc: (v) => v + 1,
4   }
5 };
6
7 const response = await fetch('add_inc.wasm')
8 const buffer = await response.arrayBuffer()
9 const { module, instance } =
10   await WebAssembly.instantiate(buffer, importObj)
11 console.log(instance.exports.add_inc(1, 2))
```

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

Tool Chains



```
(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 i32))
```

```
    get_local $p1
```

```
    get_local $p0
```

```
    i32.add
```

```
    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

```
(module  
  (type (;0;) (func (param i32)))  
  (type (;1;) (func))  
  (type (;2;) (func (param i32 i32) (result i32)))  
  (import "js" "print" (func (;0;) (type 0)))  
  (export "print" (func (;0;) (type 1)))  
  (export "memory" (memory (;0;) 65536))  
  (i32.const 66576)  
  (i32.load offset=66576)  
  (i32.store offset=1024)  
  (func (;3;) (type 2) (param i32 i32) (result i32)  
    (local i32)
```



WebAssembly System Interface

"WebAssembly: Neither Web Nor Assembly"

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)
    (local i32)
    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)
```

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

WA SI

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

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

WA SI

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

An illustration of a wrench and a hammer crossed in an 'X' shape. The wrench is light gray and the hammer is dark gray. The word 'APPLICATIONS' is written in large, bold, white capital letters across the center of the tools.

APPLICATIONS



WASM IN THE CLOUD

- Speed

- Speed
- **Size**

- Speed
- Security
- Size

- Speed
- Size
- Portability
- Security

- Speed
- Size
- Security
- Portability

fastly[®]

fastly[®]



netlify

fastly[®]


 netlify



CLOUDFLARE[®]

fastly®

 netlify


CLOUDFLARE®

CALLISTA



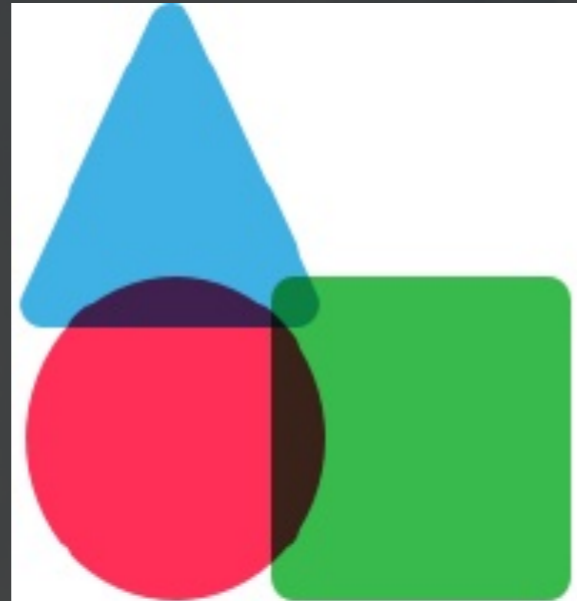
Istio



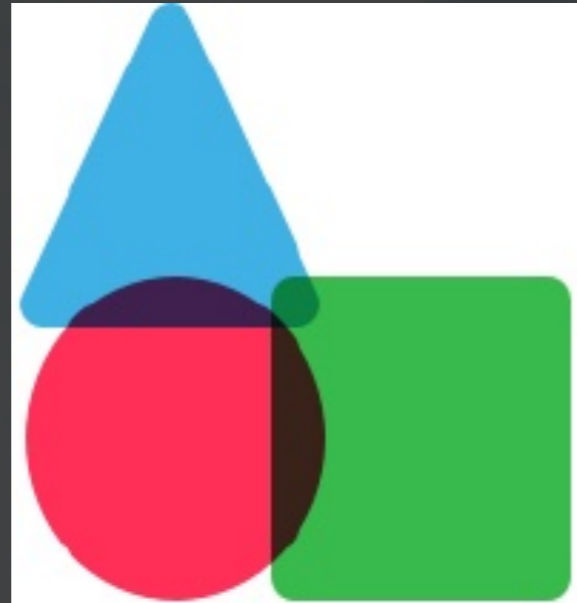
OPENRESTY



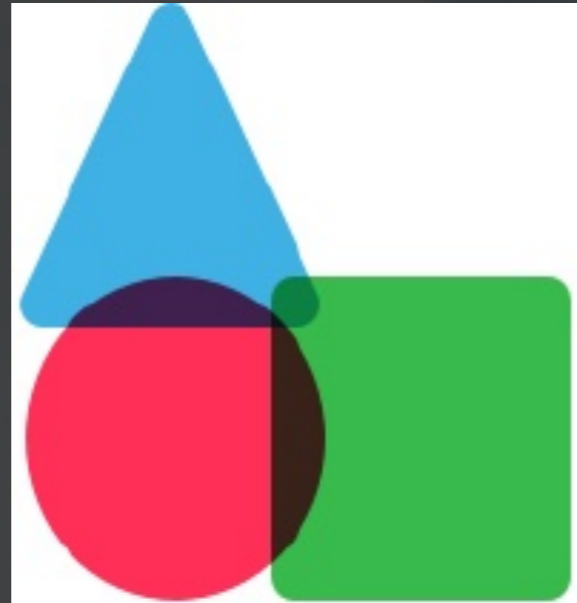
envoy



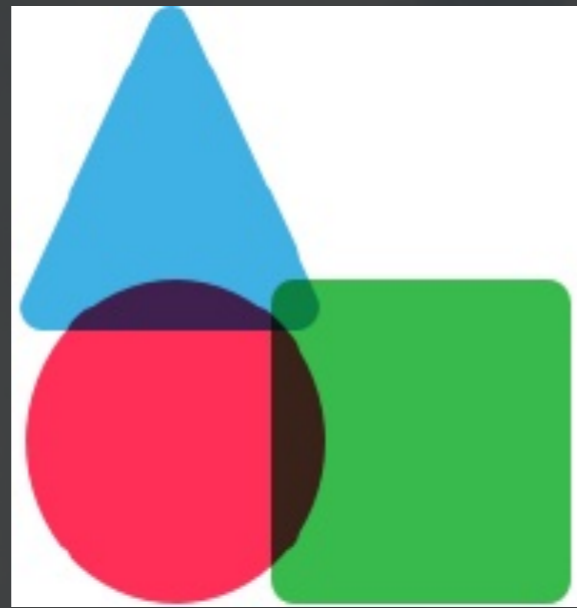
Deis Labs



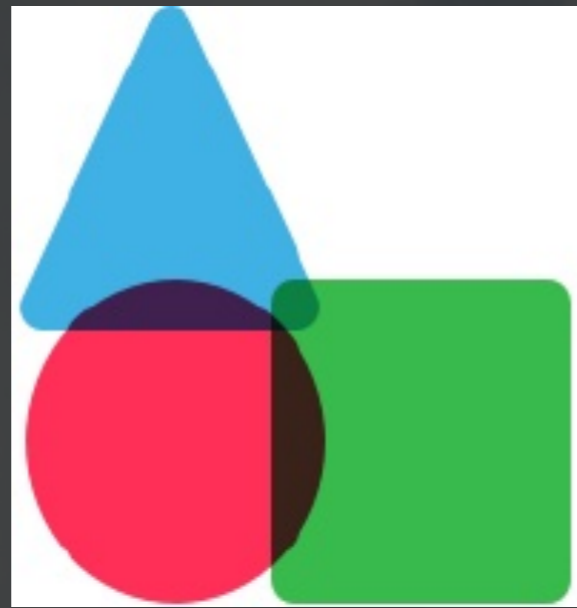
- Krustlets



- Krustlets
- Hippo



- Krustlets
- Hippo
- WAGI



- Krustlets
- Hippo
- WAGI



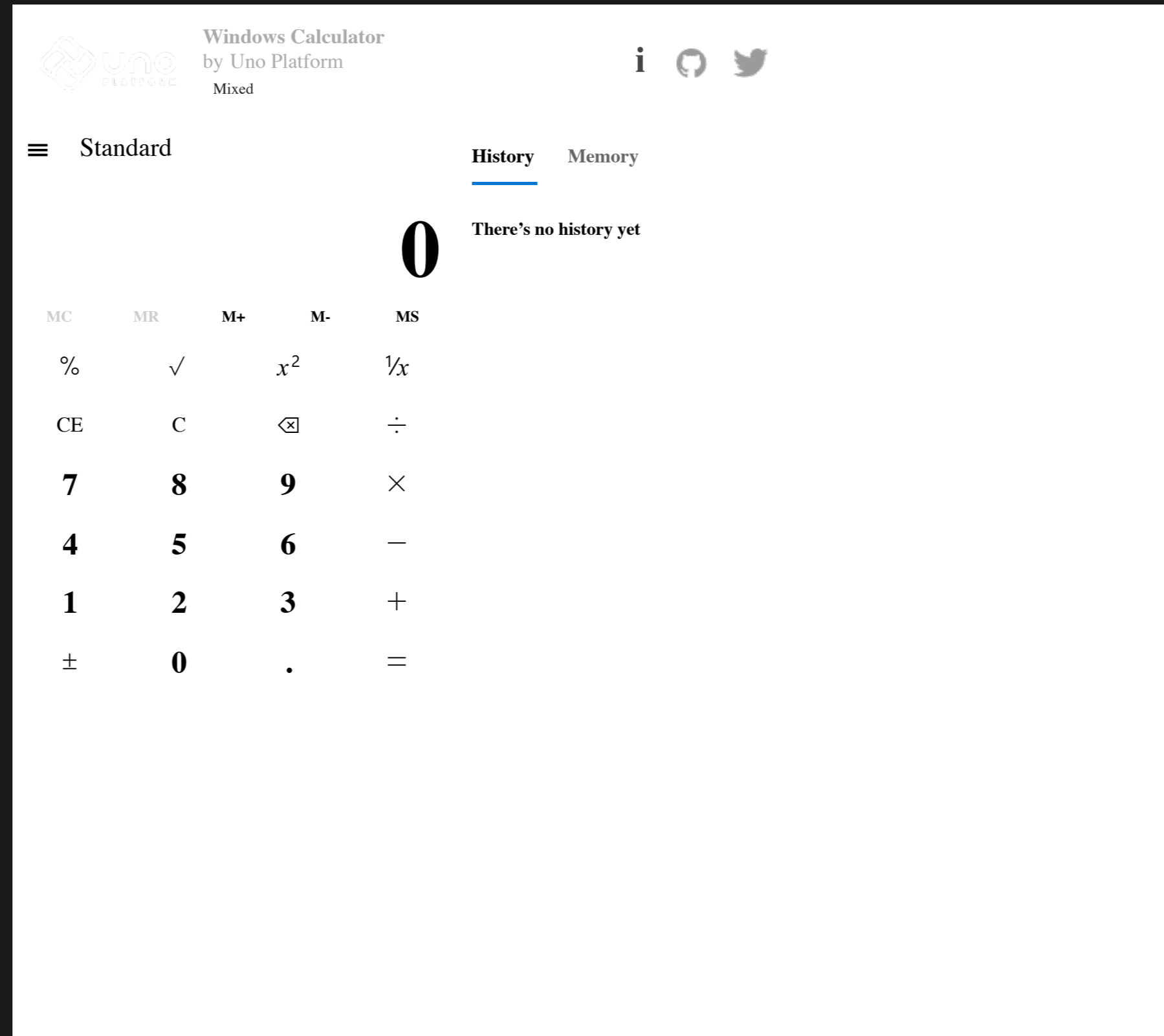
wasmcloud



```
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();
```


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

<https://calculator.platform.uno>



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

KEEP AN EYE ON WEBASSEMBLY!

