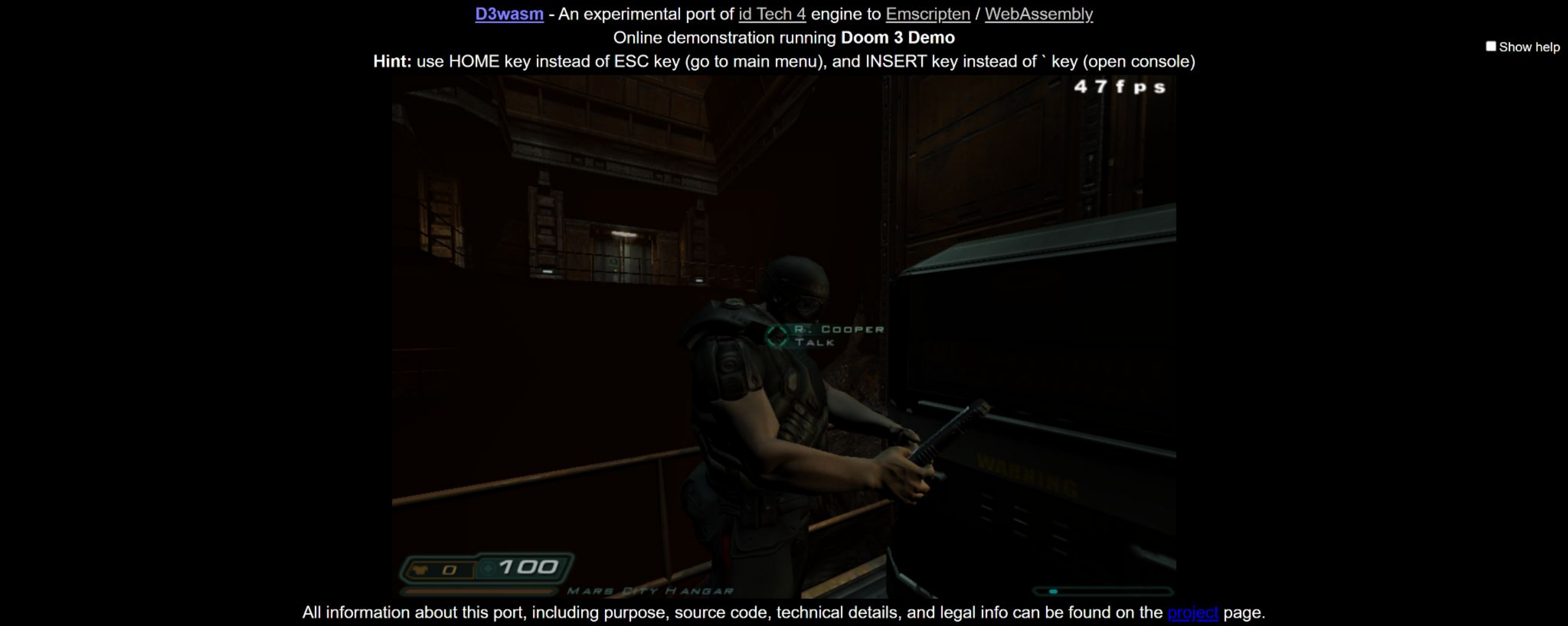
WebAssembly is Cool!

(finally)

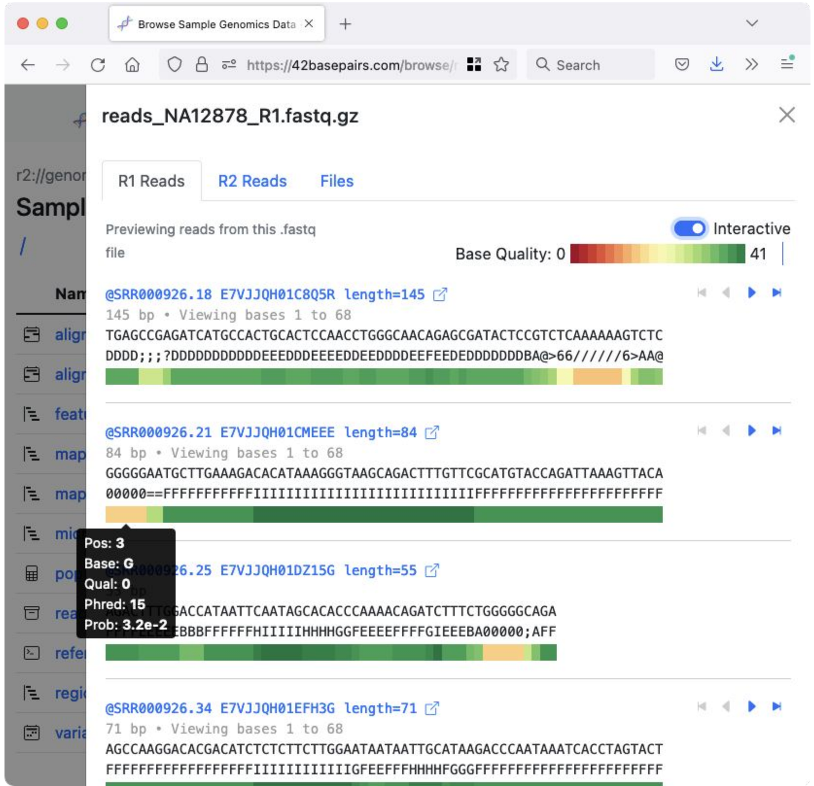


D3wasm - An experimental port of id Tech 4 engine to Emscripten / WebAssembly
Online demonstration running **Doom 3 Demo**
Hint: use HOME key instead of ESC key (go to main menu), and INSERT key instead of ` key (open console)

All information about this port, including purpose, source code, technical details, and legal info can be found on the project page.

Doom at 60fps in your browser

Scientific compute with zero install



reads_NA12878_R1.fastq.gz

R1 Reads | R2 Reads | Files

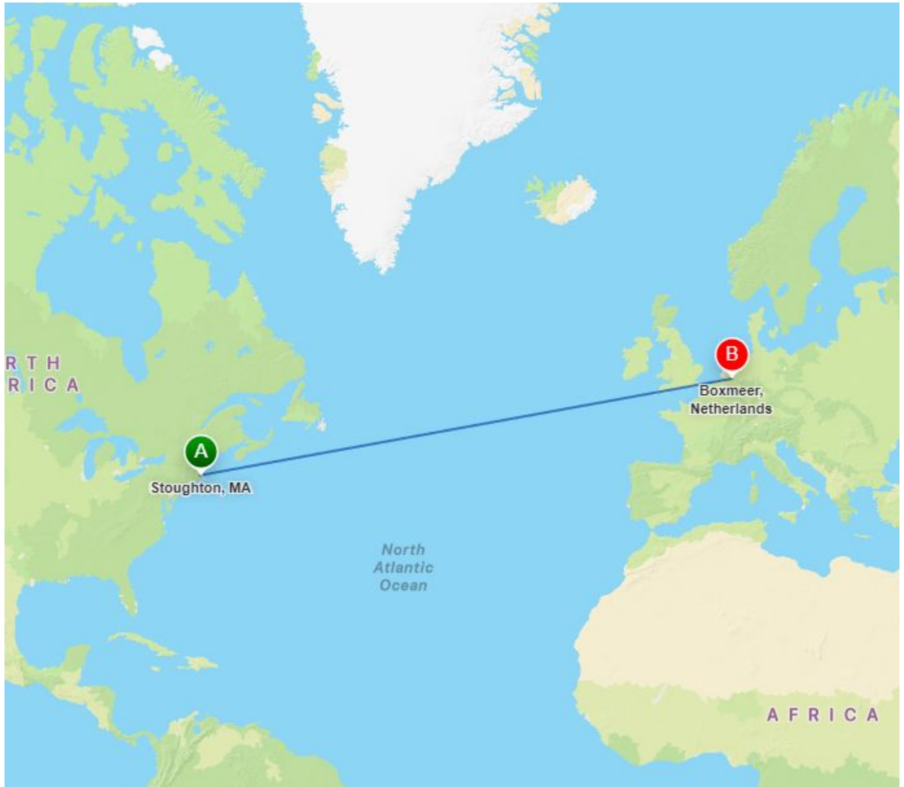
Previewing reads from this .fastq file

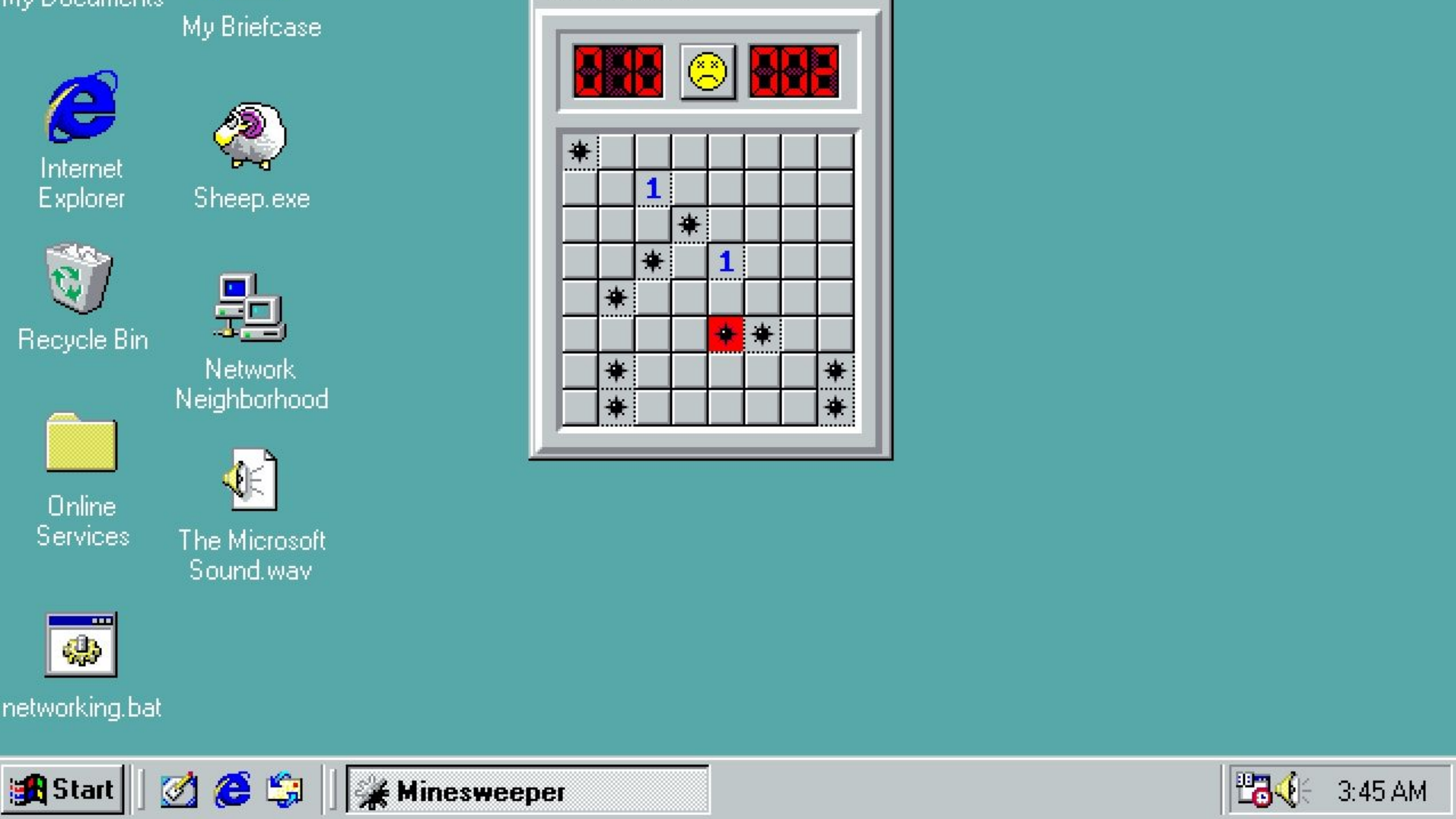
Interactive

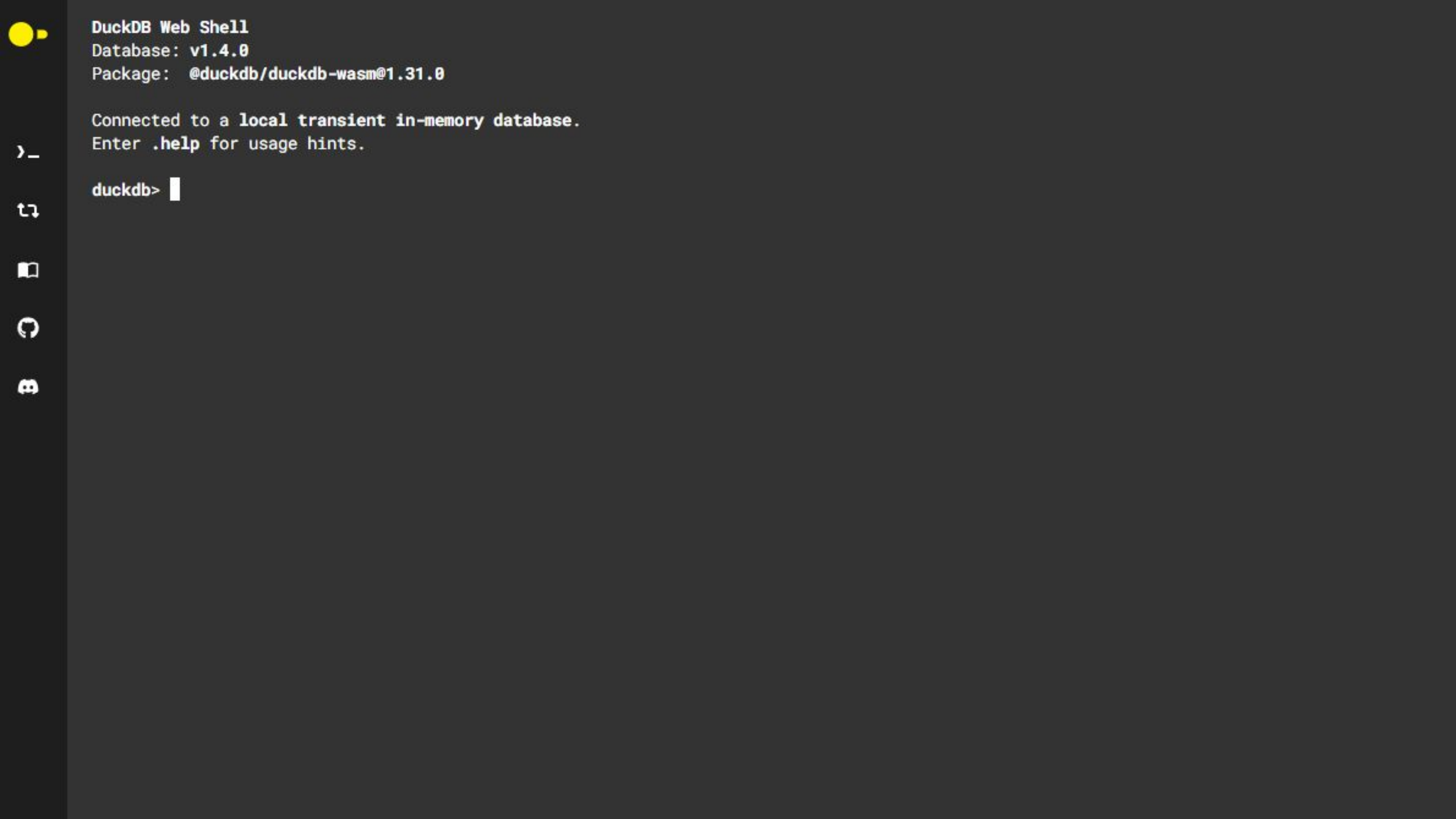
Base Quality: 0 41

```
@SRR000926.18 E7VJJQH01C8Q5R length=145
145 bp • Viewing bases 1 to 68
TGAGCCGAGATCATGCCACTGCACTCCAACCTGGGCAACAGAGCGATACTCCGTCTCAAAAAAGTCTC
DDDD;;;?DDDDDDDDDDDEEEDDDEEEEDDEEDDDDEEFEEDEDDDDDDDBA@>66//////6>AA@

@SRR000926.21 E7VJJQH01CMEEE length=84
84 bp • Viewing bases 1 to 68
GGGGGAATGCTTGAAAGACACATAAAGGGTAAGCAGACTTTGTTCGCATGTACCAGATTAAAGTTACA
00000==FFFFFFFFFFIIIIIIIIIIIIIIIIIIIIIIIIIIIIFFFFFFFFFFFFFFFFFFFFFF

6.25 E7VJJQH01DZ15G length=55
GACCATAATTCAATAGCACACCCAAAACAGATCTTTCTGGGGGCAGA
EBBBFFFFFFHIIIIIHHHHGGFEEEEFFFFGIEEEBA00000;AFF

@SRR000926.34 E7VJJQH01EFH3G length=71
71 bp • Viewing bases 1 to 68
AGCCAAGGACACGACATCTCTCTTCTTGGAATAATAATTGCATAAGACCCAATAAATCACCTAGTACT
FFFFFFFFFFFFFFFFFFFIIIIIIIIIIIGFEEFFFHHHHFGGGFFFFFFFFFFFFFFFFFFFFFF
```

Pos: 3
Base: G
Qual: 0
Phred: 15
Prob: 3.2e-2




My Documents
My Briefcase
Internet Explorer
Sheep.exe
Recycle Bin
Network Neighborhood
Online Services
The Microsoft Sound.wav
networking.bat
1
1
Start
Minesweeper
3:45 AM



```
DuckDB Web Shell
Database: v1.4.0
Package:  @duckdb/duckdb-wasm@1.31.0

Connected to a local transient in-memory database.
Enter .help for usage hints.

duckdb> 
```

So who ***IS*** this guy?

(fair question)

Jakob Heuser

Former Pinterest, LinkedIn

Co-founder, builder, maker

Wasm Enthusiast (obviously)

Setting expectations & ground rules

Pro-Wasm doesn’t mean feeding the hype cycle

See Wasm through a practical lens

Avoid a messy Q&A, let’s talk in small groups after!

Today

~~Guy plays DOOM 3 and Minesweeper~~

~~About that Jakob guy~~

What WebAssembly ISN’T / What WebAssembly IS

Practical use cases

The future & more inspiring stuff

Keep Learning

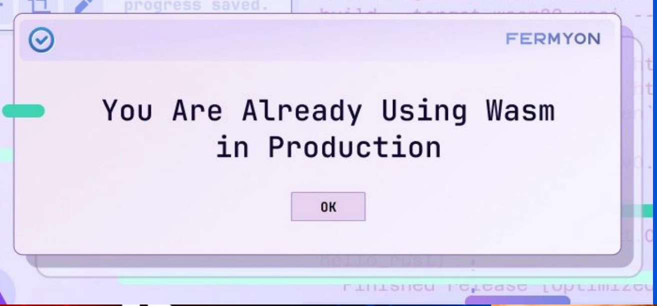
Google I/O - WebAssembly: A new development paradigm for the web (2023)
https://www.youtube.com/watch?v=RcHER-3gFXI

NDC - The WebAssembly Component Model (2024)
https://www.youtube.com/watch?v=_fKPvnhX-vI

Devoxx UK - WebAssembly outside the browser (2024)
https://www.youtube.com/watch?v=We1JKjjTFXI

So what ***ISN'T*** WebAssembly?

(thanks for reading subtitles!)


FERMYON

You Are Already Using Wasm in Production

OK

RAMOTION

Is WebAssembly the Future?

Why WASM is not the futur

Babylon.js

Babylon.js Follow 4 min read · Aug 6, 2021

WASM was a mistake. I just wanted to

code on the web. Everyone praised it. N

uble or quadruple my develop

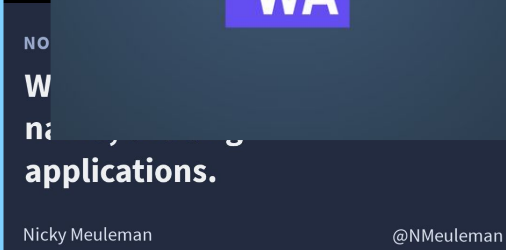
to curse repeatedly at the sc

Building a Containerless Future with WebAssembly

WA

Python Developers

Why WebAssembly is the Future of Browser-Based Python Development


NO

W

na

applications.

Nicky Meuleman @NMeuleman

NOT Java in the browser 2.0

Secure by default (no, really)

Does not include a runtime

No direct system calls

“If Wasm+WASI existed in 2008, we wouldn't have needed to create Docker. That's how important it is.”

Solomon Hykes (co-founder of Docker)

NOT replacing Docker

WebAssembly excels at single tasks

Docker excels at gnarly imperfect software running together

You’re not swapping Docker with Wasm

NOT replacing JavaScript (sorry)

Something must talk to the DOM

Most companies aren’t going to add “and a binary” to JS builds

So what *IS* WebAssembly?

(besides not being for just the web and also not being assembly)

World’s fastest Wasm history lesson

Announced 2015 / Launched 2017 / W3C 2019

Successor to technologies like asm.js & Emscripten

Now (Sept 17) on version 3.0 of the specification

Cool, but what ***IS*** it?

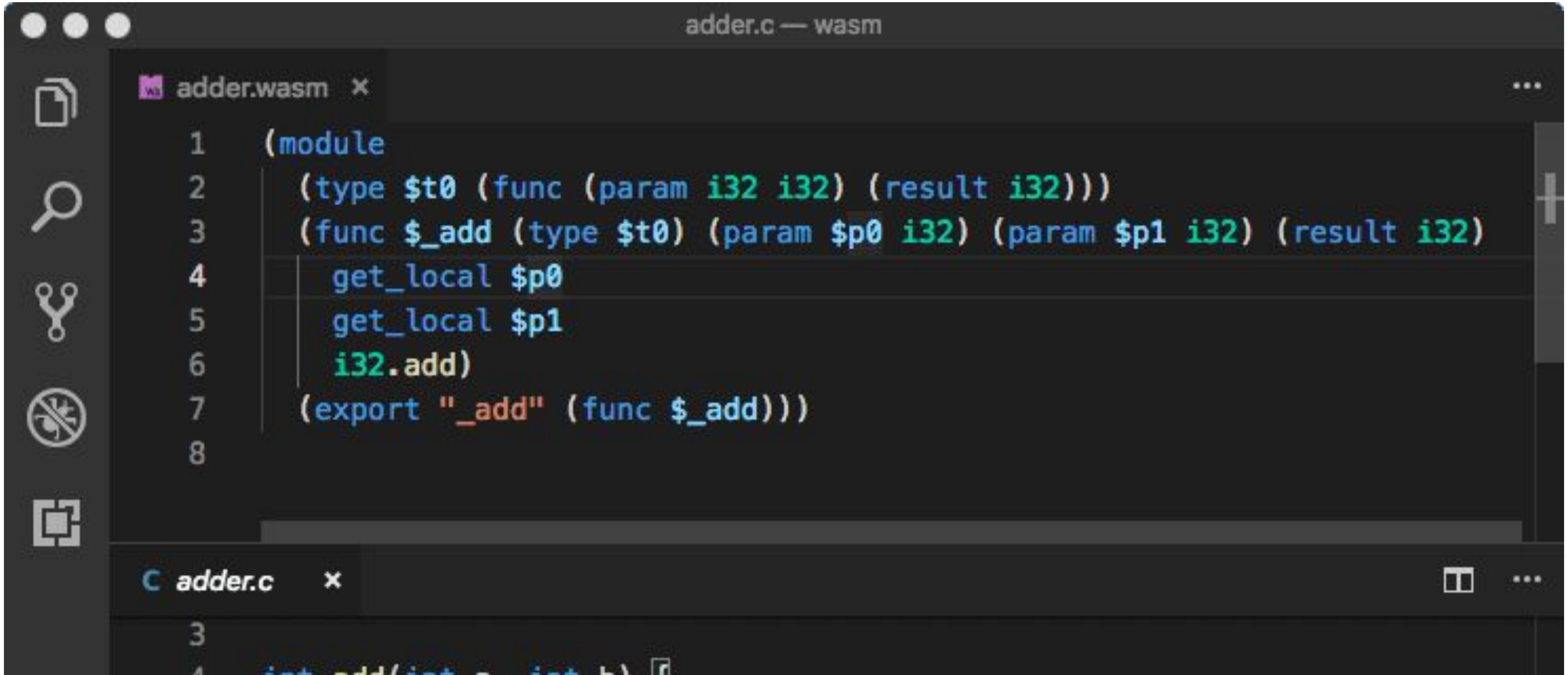
A virtual instruction set architecture (virtual ISA)

Uses Linear Memory with few “types”: i32, i64, f32, f64...

Embeddable in a Host Environment

So let’s go *deeper* into WebAssembly

A virtual ISA with a stack-based design



```
(module
  (type $t0 (func (param i32 i32) (result i32)))
  (func $_add (type $t0) (param $p0 i32) (param $p1 i32) (result i32)
    get_local $p0
    get_local $p1
    i32.add)
  (export "_add" (func $_add)))
```

That compiles to a binary instruction format

TEXTUAL FORMAT

```
(module
  (func $addTwo (param i32 i32)
    (result i32)
    (i32.add
      (get_local 0)
      (get_local 1)))
  (export "addTwo" $addTwo))
```

=

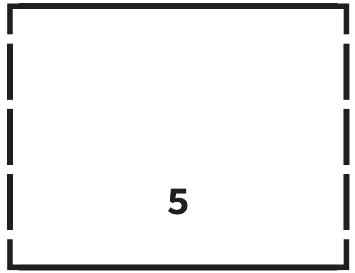
BINARY FORMAT

```
48 83 EC 08
8B CF
8B C1
03 C6
66 90
48 83 C4 08
C3
```

The stack design

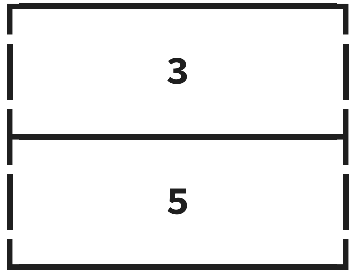
```
PROGRAM: (i32.const 5) (i32.const 3) (i32.add)
```

Step 1: i32.const 5



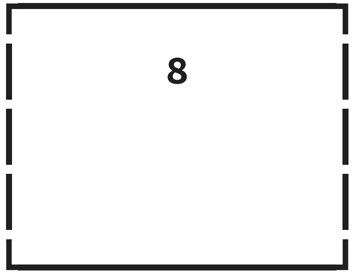
Stack: [5]

Step 2: i32.const 3

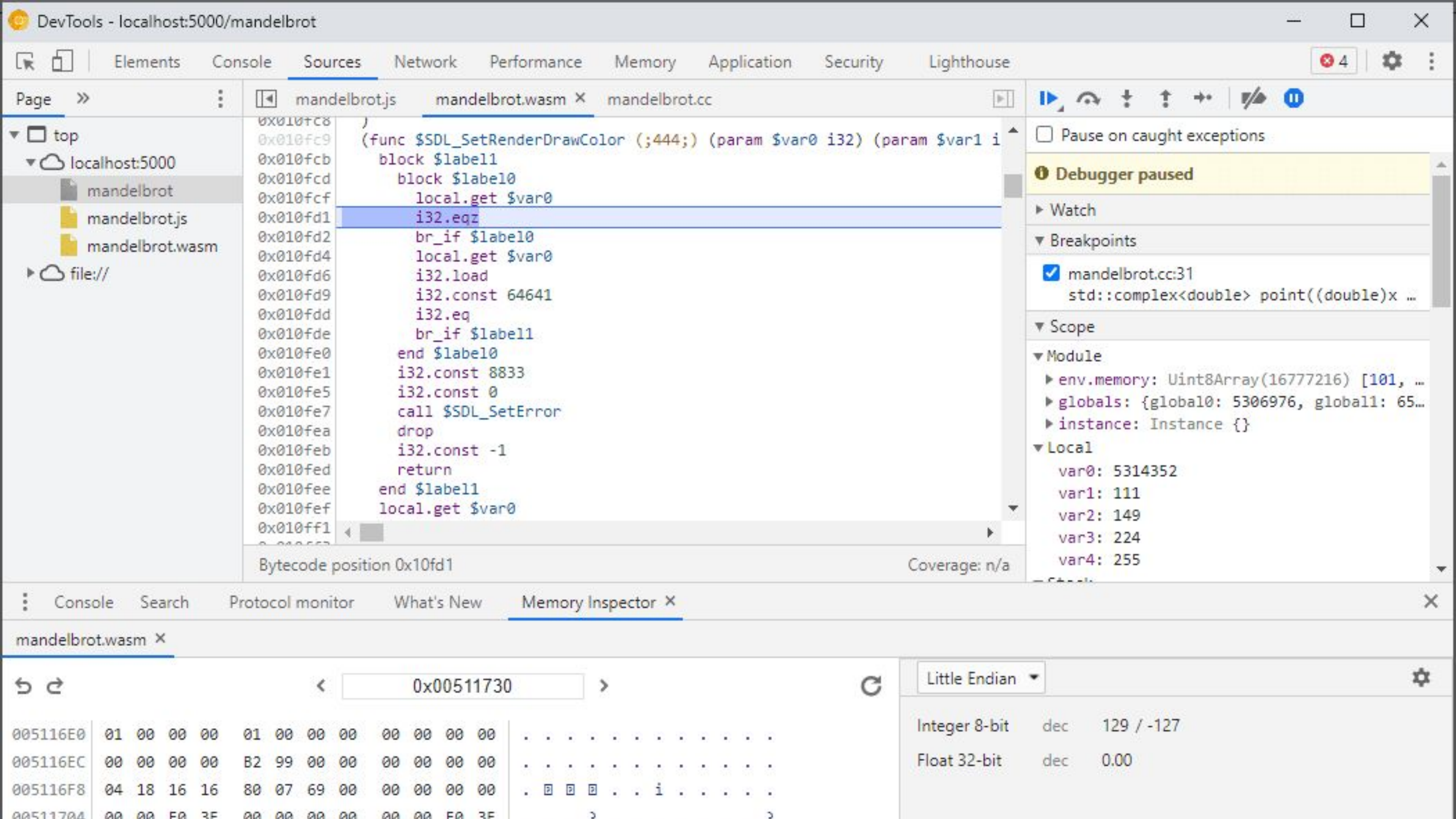


Stack: [5, 3]

Step 3: i32.add



Stack: [8]



```
0x010fc8    )
0x010fc9    (func $SDL_SetRenderDrawColor (;444;) (param $var0 i32) (param $var1 i
0x010fcb      block $label1
0x010fcd        block $label0
0x010fcf          local.get $var0
0x010fd1          i32.eqz
0x010fd2          br_if $label0
0x010fd4          local.get $var0
0x010fd6          i32.load
0x010fd9          i32.const 64641
0x010fdd          i32.eq
0x010fde          br_if $label1
0x010fe0        end $label0
0x010fe1        i32.const 8833
0x010fe5        i32.const 0
0x010fe7        call $SDL_SetError
0x010fea        drop
0x010feb        i32.const -1
0x010fed        return
0x010fee      end $label1
0x010fef      local.get $var0
0x010ff1
```

Bytecode position 0x10fd1   Coverage: n/a

mandelbrot.cc:31
`std::complex<double> point((double)x …`

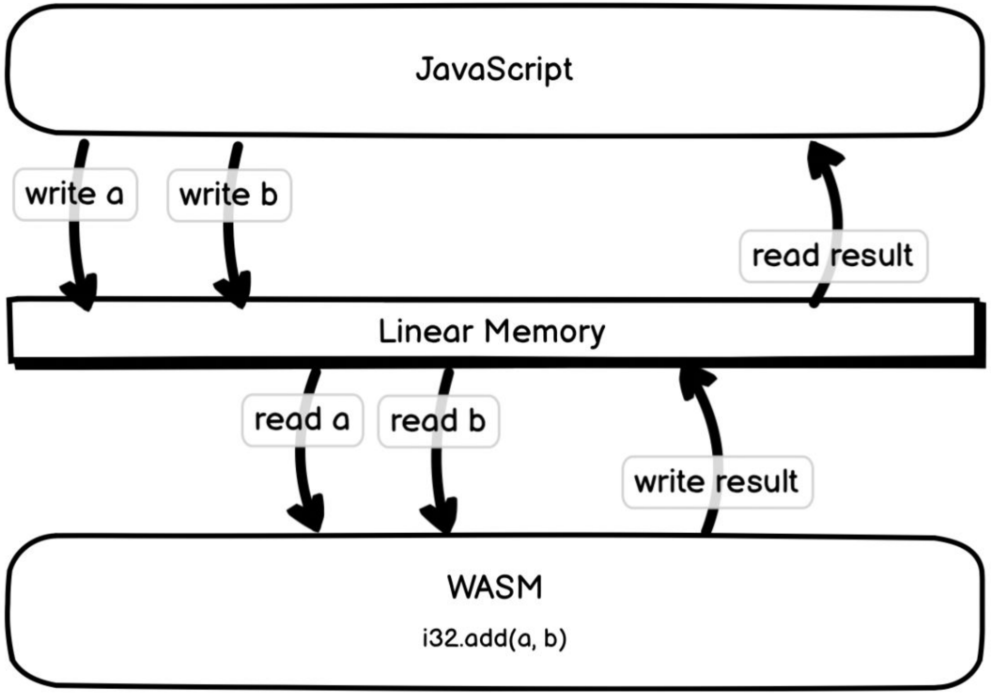
```
▼ Module
  ▶ env.memory: Uint8Array(16777216) [101, …
  ▶ globals: {global0: 5306976, global1: 65…
  ▶ instance: Instance {}
▼ Local
    var0: 5314352
    var1: 111
    var2: 149
    var3: 224
    var4: 255
```

0x00511730

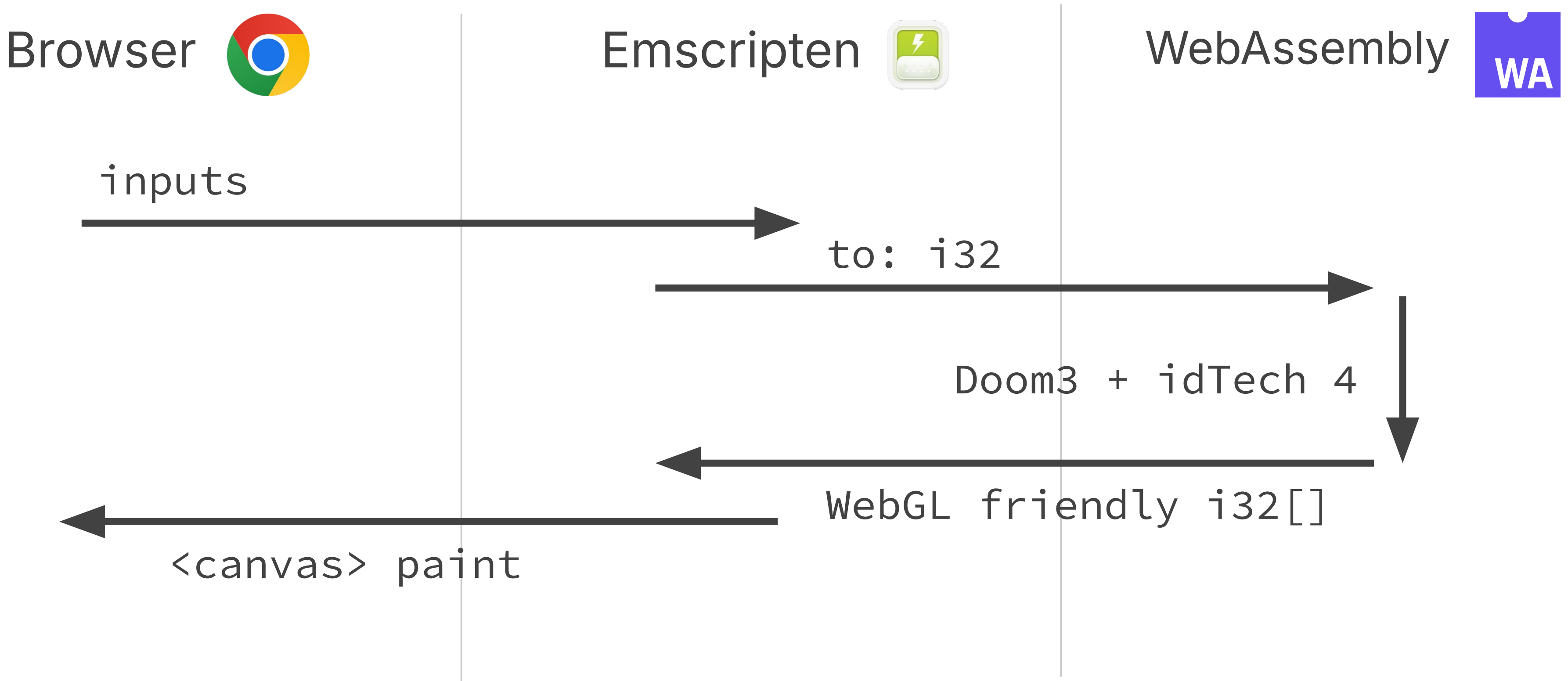
```
005116E0  01 00 00 00  01 00 00 00  00 00 00 00   . . . . . . . . . . . .
005116EC  00 00 00 00  B2 99 00 00  00 00 00 00   . . . . . . . . . . . .
005116F8  04 18 16 16  80 07 69 00  00 00 00 00   . ▯ ▯ ▯ . . i . . . . .
00511704  00 00 F0 3F  00 00 00 00  00 00 F0 3F
```

Integer 8-bit   dec   129 / -127
Float 32-bit   dec   0.00

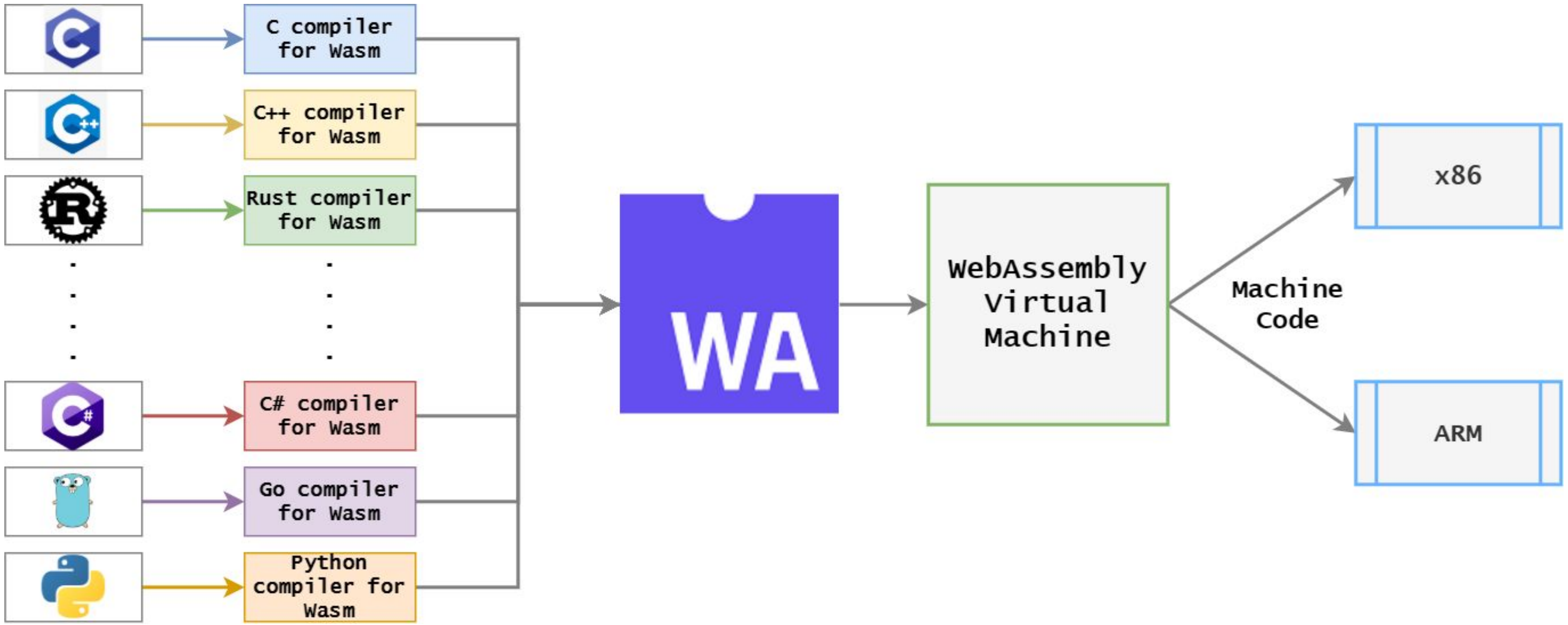
Utilizing Linear Memory



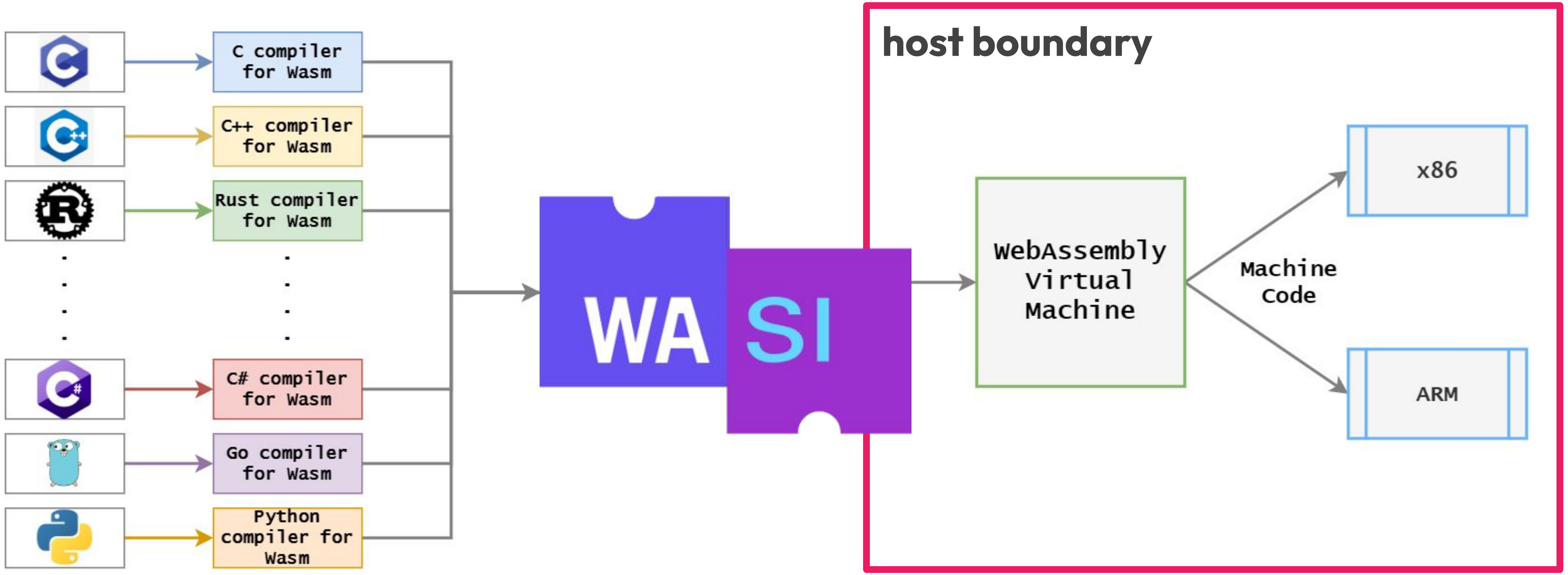
Doom3 is linear memory & WebGL



Gains superpowers through WASI



Embedded in a Host Environment



Without compromising security

Traps at the Wasm level ⇒ Exceptions in host

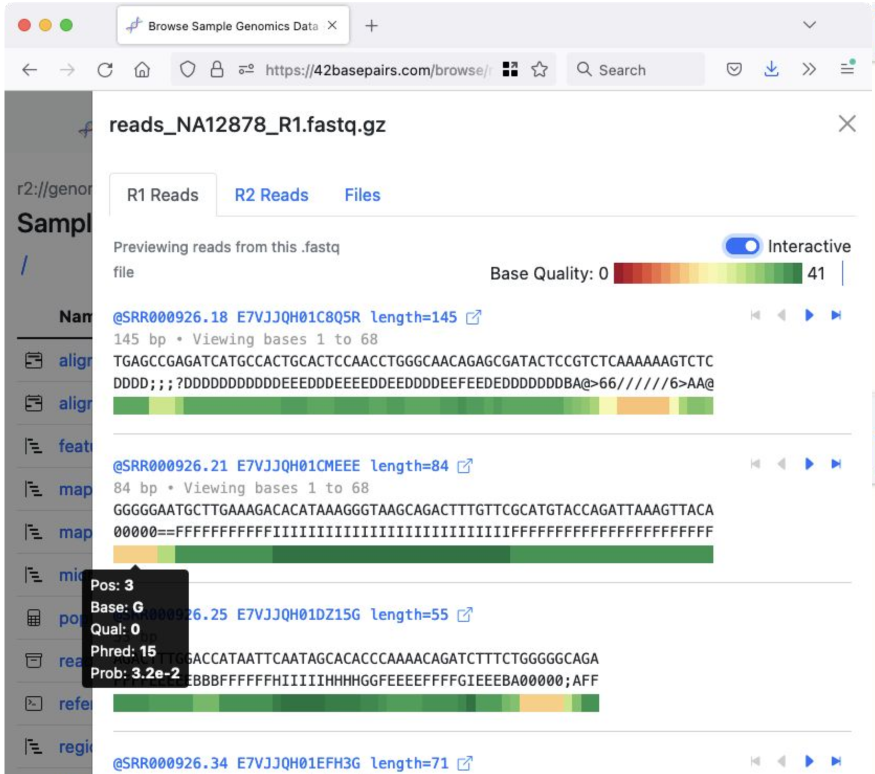
Module isolation puts every Wasm in its own memory

Attack surface area defined by features allowed

WebAssembly at its Best

(the kinds of problems Wasm was meant for)

Scientific computing: BioWasm



reads_NA12878_R1.fastq.gz

R1 Reads | R2 Reads | Files

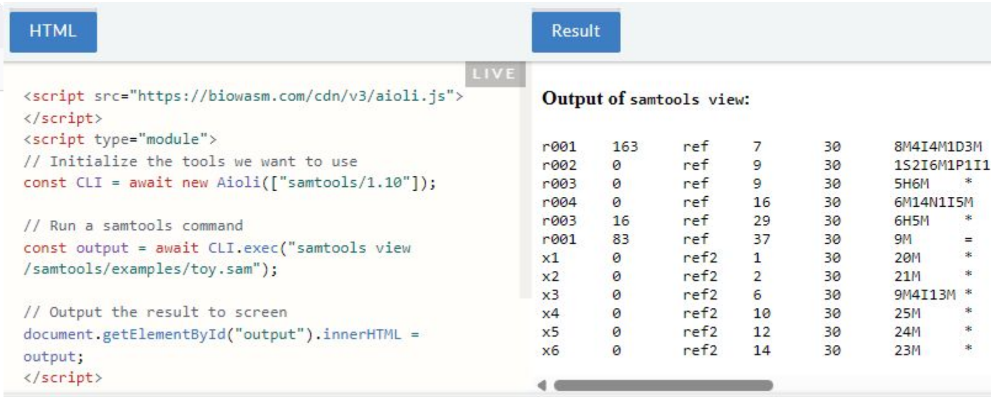
Previewing reads from this .fastq file

Interactive

Base Quality: 0 – 41

```
@SRR000926.18 E7VJJQH01C8Q5R length=145
145 bp • Viewing bases 1 to 68
TGAGCCGAGATCATGCCACTGCACTCCAACCTGGGCAACAGAGCGATACTCCGTCTCAAAAAAGTCTC
DDDD;;;?DDDDDDDDDDDEEEDDDEEEEDDEEDDDDEEFEEDEDDDDDDDBA@>66//////6>AA@

@SRR000926.21 E7VJJQH01CMEEE length=84
84 bp • Viewing bases 1 to 68
GGGGGAATGCTTGAAAGACACATAAAGGGTAAGCAGACTTTGTTCGCATGTACCAGATTAAAGTTACA
00000==FFFFFFFFFFIIIIIIIIIIIIIIIIIIIIIIIIIIIFFFFFFFFFFFFFFFFFFFFFFFF

@SRR000926.25 E7VJJQH01DZ15G length=55
GGACCATAATTCAATAGCACACCCAAAACAGATCTTTCTGGGGGCAGA
EBBBFFFFFFHIIIIIHHHHGGFEEEEFFFFGIEEEBA00000;AFF

@SRR000926.34 E7VJJQH01EFH3G length=71
```

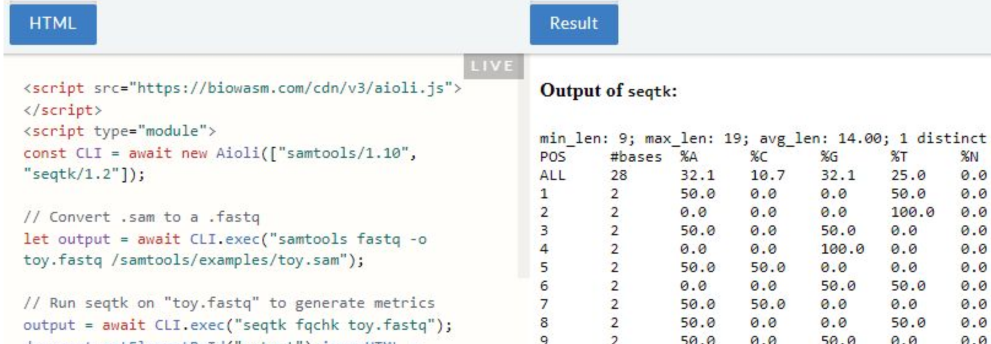
Pos: 3
Base: G
Qual: 0
Phred: 15
Prob: 3.2e-2



```
<script src="https://biowasm.com/cdn/v3/aioli.js">
</script>
<script type="module">
// Initialize the tools we want to use
const CLI = await new Aioli(["samtools/1.10"]);

// Run a samtools command
const output = await CLI.exec("samtools view
/samtools/examples/toy.sam");

// Output the result to screen
document.getElementById("output").innerHTML =
output;
</script>
```

Output of `samtools view`:

```
r001    163     ref     7       30      8M4I4M1D3M
r002    0       ref     9       30      1S2I6M1P1I1
r003    0       ref     9       30      5H6M     *
r004    0       ref     16      30      6M14N1I5M
r003    16      ref     29      30      6H5M     *
r001    83      ref     37      30      9M       =
x1      0       ref2    1       30      20M      *
x2      0       ref2    2       30      21M      *
x3      0       ref2    6       30      9M4I13M  *
x4      0       ref2    10      30      25M      *
x5      0       ref2    12      30      24M      *
x6      0       ref2    14      30      23M      *
```

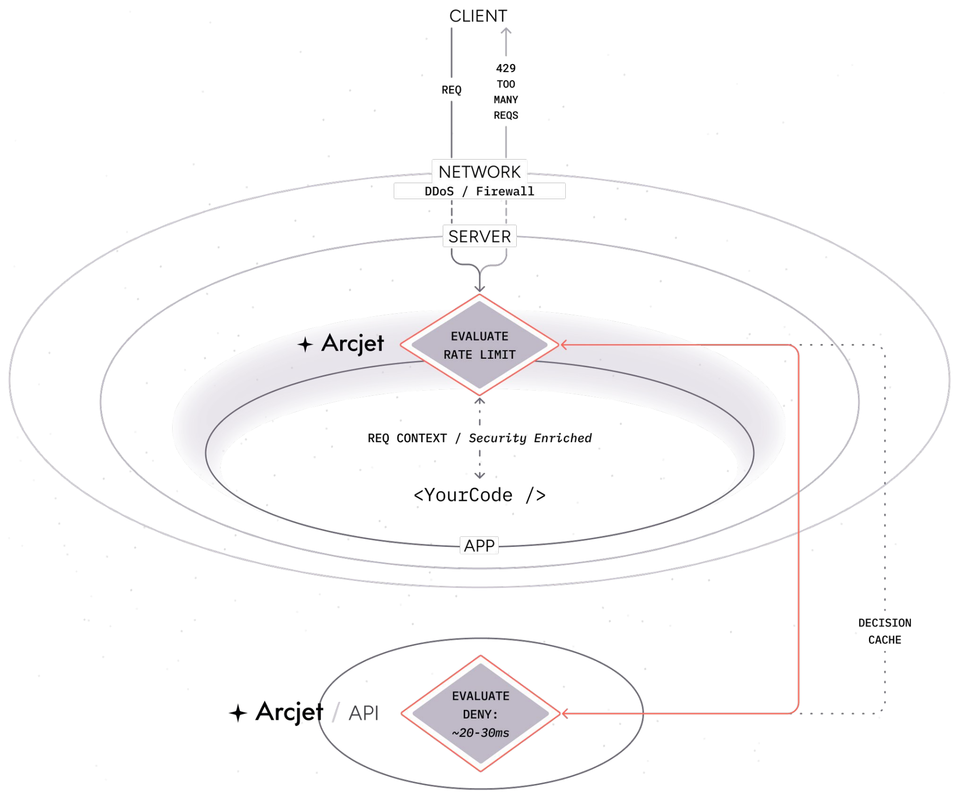


```
<script src="https://biowasm.com/cdn/v3/aioli.js">
</script>
<script type="module">
const CLI = await new Aioli(["samtools/1.10",
"seqtk/1.2"]);

// Convert .sam to a .fastq
let output = await CLI.exec("samtools fastq -o
toy.fastq /samtools/examples/toy.sam");

// Run seqtk on "toy.fastq" to generate metrics
output = await CLI.exec("seqtk fqchk toy.fastq");
```

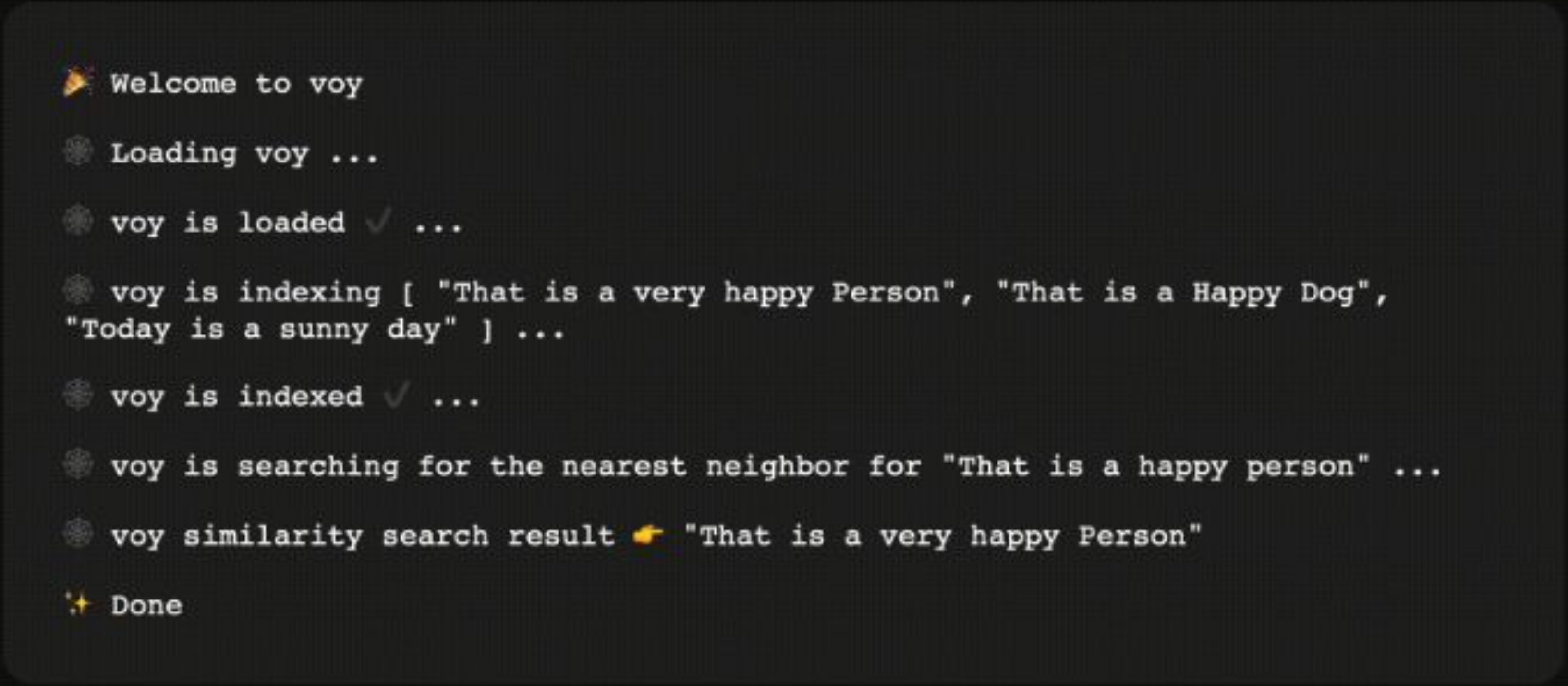
Output of `seqtk`:

```
min_len: 9; max_len: 19; avg_len: 14.00; 1 distinct
POS     #bases  %A      %C      %G      %T      %N
ALL     28      32.1    10.7    32.1    25.0    0.0
1       2       50.0    0.0     0.0     50.0    0.0
2       2       0.0     0.0     0.0     100.0   0.0
3       2       50.0    0.0     50.0    0.0     0.0
4       2       0.0     0.0     100.0   0.0     0.0
5       2       50.0    50.0    0.0     0.0     0.0
6       2       0.0     0.0     50.0    50.0    0.0
7       2       50.0    50.0    0.0     0.0     0.0
8       2       50.0    0.0     0.0     50.0    0.0
9       2       50.0    0.0     50.0    0.0     0.0
```

Security at near-native speeds

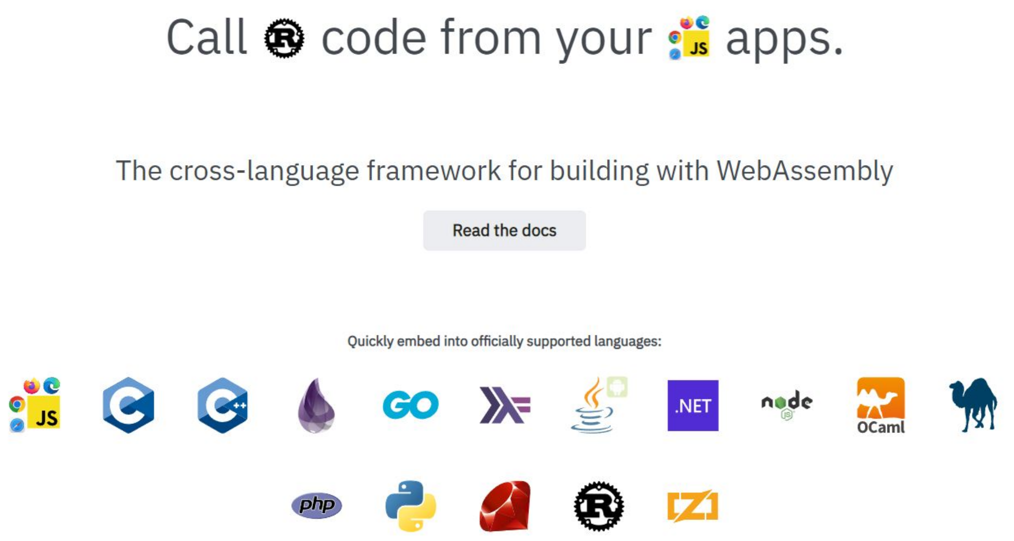


Vector search in edge computing: Voy



```
🎉 Welcome to voy

🕸️ Loading voy ...

🕸️ voy is loaded ✓ ...

🕸️ voy is indexing [ "That is a very happy Person", "That is a Happy Dog",
"Today is a sunny day" ] ...

🕸️ voy is indexed ✓ ...

🕸️ voy is searching for the nearest neighbor for "That is a happy person" ...

🕸️ voy similarity search result 👉 "That is a very happy Person"

✨ Done
```

Beyond the browser: universal runtimes



Call code from your apps.

The cross-language framework for building with WebAssembly

Read the docs

Quickly embed into officially supported languages:

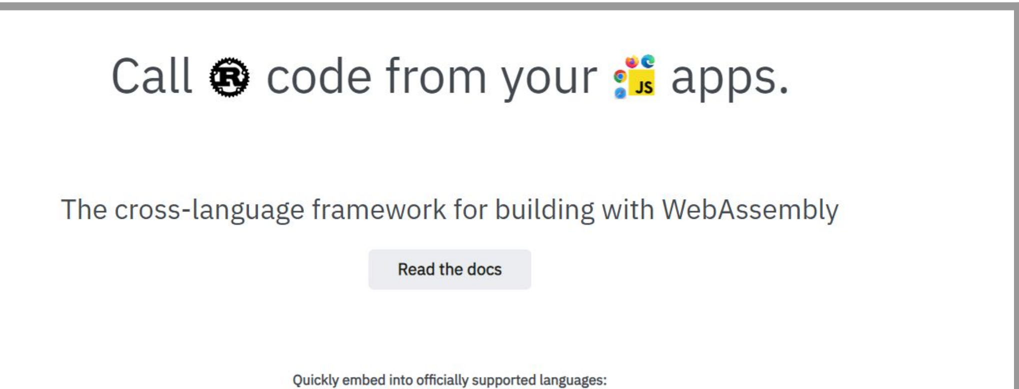
The Future of Wasm

(cool things to keep an eye on)

Portable Compute

PostgreSQL UDFs with **Extism**

Universal build tooling with **Moonrepo**

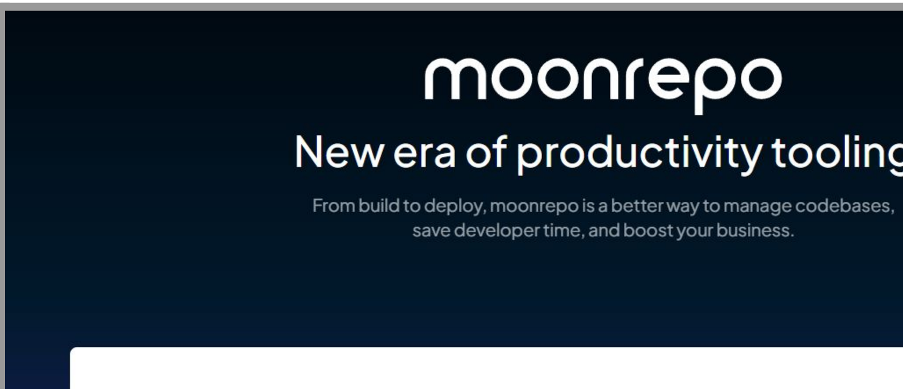


Call code from your apps.

The cross-language framework for building with WebAssembly

Read the docs

Quickly embed into officially supported languages:

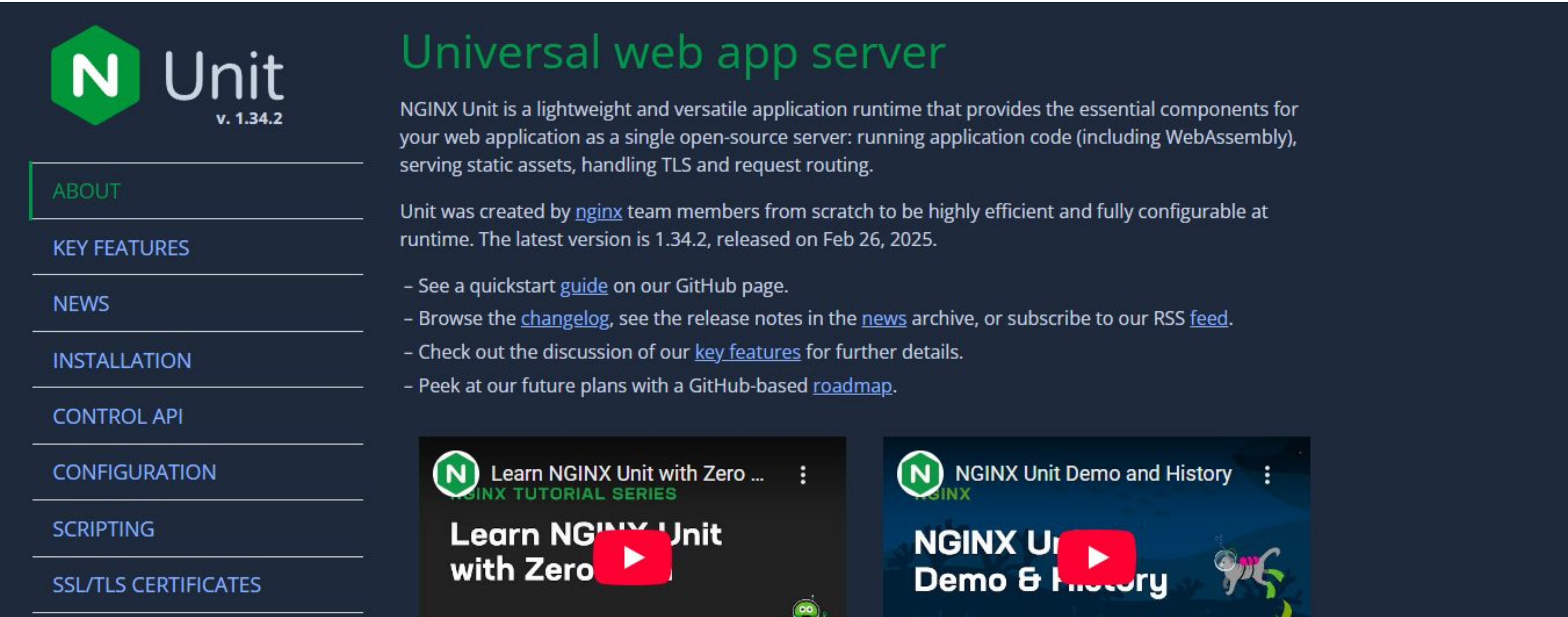


moonrepo

New era of productivity tooling

From build to deploy, moonrepo is a better way to manage codebases, save developer time, and boost your business.

Practical Containerization



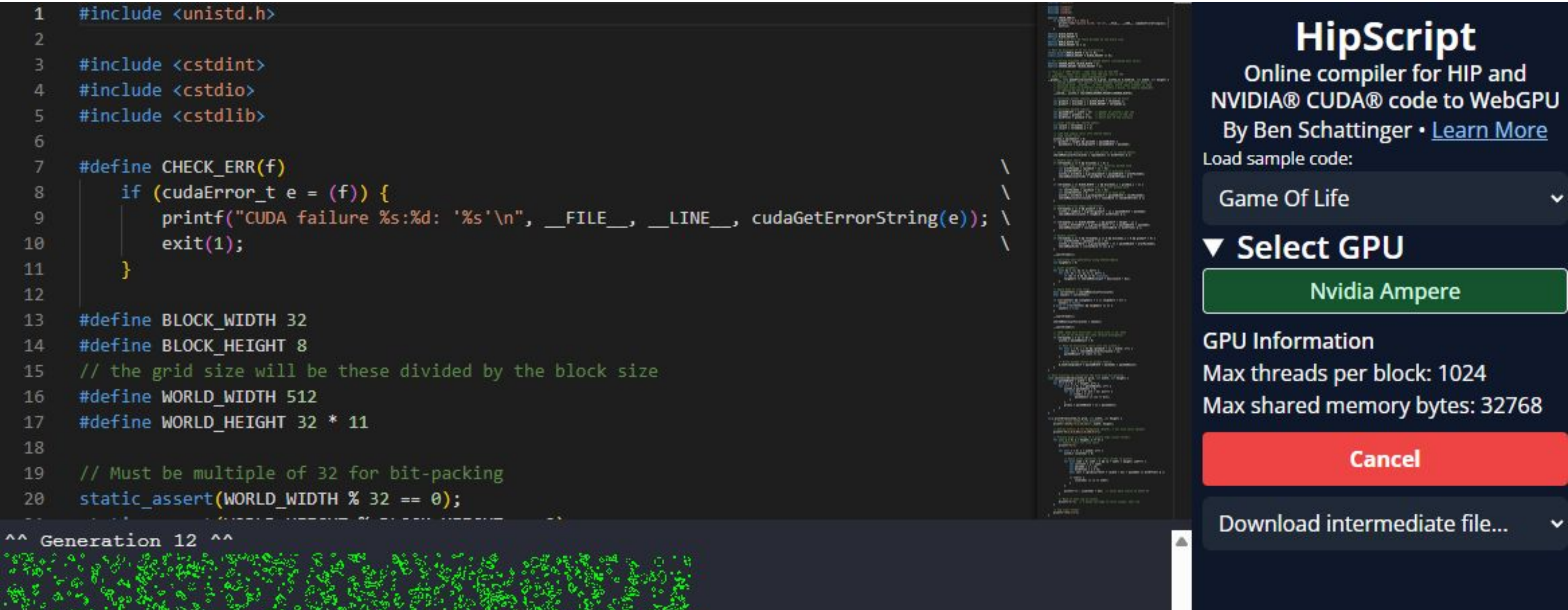
Universal web app server

NGINX Unit is a lightweight and versatile application runtime that provides the essential components for your web application as a single open-source server: running application code (including WebAssembly), serving static assets, handling TLS and request routing.

Unit was created by nginx team members from scratch to be highly efficient and fully configurable at runtime. The latest version is 1.34.2, released on Feb 26, 2025.

- See a quickstart guide on our GitHub page.
- Browse the changelog, see the release notes in the news archive, or subscribe to our RSS feed.
- Check out the discussion of our key features for further details.
- Peek at our future plans with a GitHub-based roadmap.

WebGPU and local LLM



```
#include <unistd.h>

#include <cstdint>
#include <cstdio>
#include <cstdlib>

#define CHECK_ERR(f)                                                          \
    if (cudaError_t e = (f)) {                                                \
        printf("CUDA failure %s:%d: '%s'\n", __FILE__, __LINE__, cudaGetErrorString(e)); \
        exit(1);                                                              \
    }

#define BLOCK_WIDTH 32
#define BLOCK_HEIGHT 8
// the grid size will be these divided by the block size
#define WORLD_WIDTH 512
#define WORLD_HEIGHT 32 * 11

// Must be multiple of 32 for bit-packing
static_assert(WORLD_WIDTH % 32 == 0);
```

```
^^ Generation 12 ^^
```

HipScript

Online compiler for HIP and NVIDIA® CUDA® code to WebGPU

By Ben Schattinger • Learn More

Load sample code: Game Of Life

▼ Select GPU

Nvidia Ampere

GPU Information

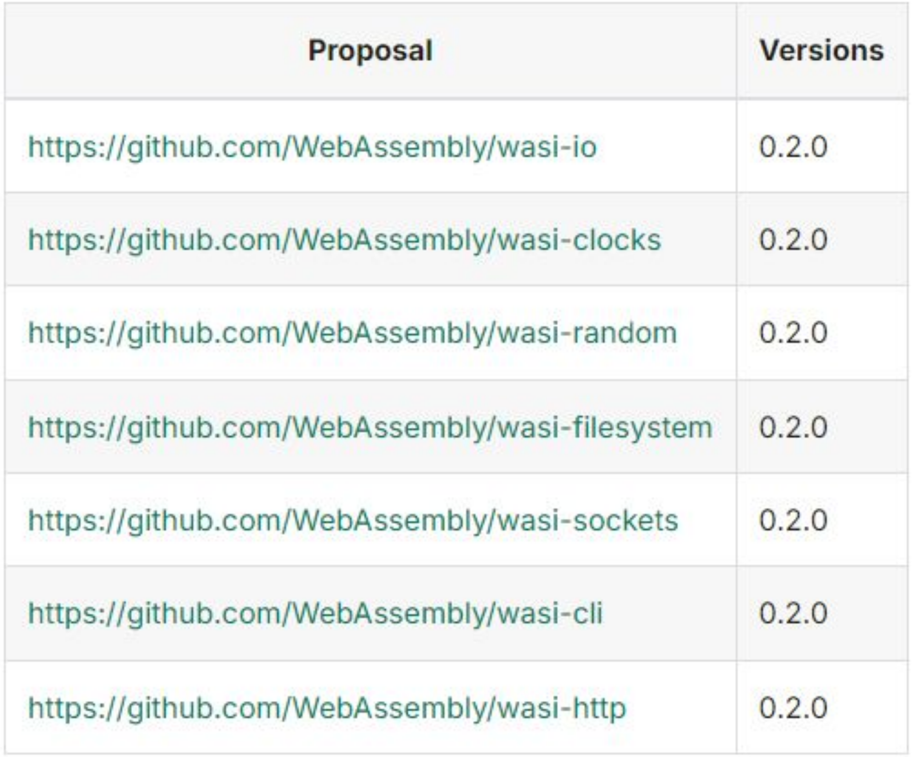
Max threads per block: 1024

Max shared memory bytes: 32768

Cancel

Download intermediate file...

WASI 0.2: Upgraded host interaction (Jan 2024)



Proposal	Versions
https://github.com/WebAssembly/wasi-io	0.2.0
https://github.com/WebAssembly/wasi-clocks	0.2.0
https://github.com/WebAssembly/wasi-random	0.2.0
https://github.com/WebAssembly/wasi-filesystem	0.2.0
https://github.com/WebAssembly/wasi-sockets	0.2.0
https://github.com/WebAssembly/wasi-cli	0.2.0
https://github.com/WebAssembly/wasi-http	0.2.0

Wasm 3.0: Hello features (Sept 2025)

---

64 bit address space

Native Garbage Collector

Fully Deterministic

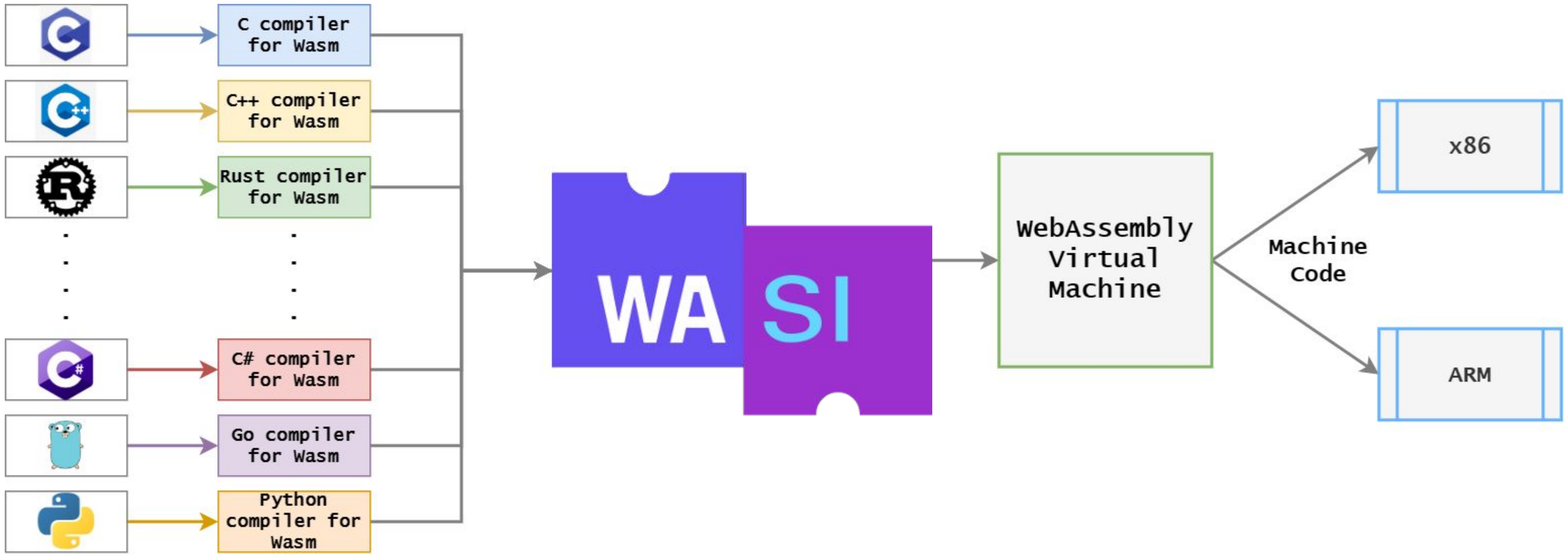
WebAssembly Specification

Release 3.0 (2025-09-17)

Bringing It All Home

(and answering “when do I Wasm?”)

You can Wasm today!


C compiler for Wasm
C++ compiler for Wasm
Rust compiler for Wasm
C# compiler for Wasm
Go compiler for Wasm
Python compiler for Wasm
WA SI
WebAssembly Virtual Machine
Machine Code
x86
ARM

jakob@codedrift.com

Tell him what you liked about this!

Appendix & Links

Examples

Doom3 Demo

42 Base Pairs data

Arcjet Example

Windows 98 Emulated

Examples (2)

BioWasm

Voy

Extism

DuckDB & DuckDB-Wasm

Moonrepo

Examples (3)

Wasmer

Boxer

Wasmtime

Foundational Technologies

asm.js

Emscripten

Specifications

WASI 0.2

Wasm 3.0

Even More Reading

When is WebAssembly Going to Get DOM Support? (hn)

Wasm cut figma's load time by 3x (figma)

Shopify Functions using Wasm (shopify)

WASMs Linear Memory Model (researchgate)

Debugging WebAssembly (chrome)

Awesome Wasm Langs (github)