SmallTalk - WebAssembly

alberto.acerbis@intre.it

albertoacerbis

WebAssembly

WebAssembly (abbreviated *Wasm*) is a binary instruction format for a stack-based virtual machine. Wasm is designed as a portable compilation target for programming languages, enabling deployment on the web for client and server applications.

INTr3

A bit of history …

- Sir Tim Berners-Lee: he recognized the need for a technical solution to solve the problem of sharing documents between different operating system or platforms
- JavaScript: Brendan Eich, hired by Netscape to create a Scheme for the browser.
- With NaCl we found a solution that provided sandboxing and performance.
- With PNaCl we found solution platform portability and sandboxing, but not browser portability
- With asm.js we found browser portability and sandboxing, but not performance
- WebAssembly (2015): Brendan Eich

What we need?

- JavaScript is not enough!
- Portability of high-level languages (C/C++/Rust … all)
- We need something to provide software that is
  - Safe
  - Fast
  - Portable
  - Compact
- We need portability at both the code and application levels
- WebAssembly is a target platform with a series of instructions that are vaguely assemblyesque

Interpreted vs Compiled

WebAssembly Text Format

A text format that describes the behavior of a module that is easier for humans to read (.wat)

```
(module
  (func $how_old (param $year_now(i32) (param $year_born i32) (result i32)
    get_local $year_now
    get_local $year_born
    i32.sub)
    (export "how_old" (func $how_old)
  )
)
```


Talk is Cheap
Show me
The Code

WebAssembly Structure

Id	Name	Description
0	Custom	Debugging or metadata information for third-party uses
1	Type	Type definitions used in the modules
2	Import	Imported elements used by a module
3	Function	Type signatures associated with the functions in a module
4	Table	Tables that define indirect, immutable reference used by a module
5	Memory	Linear memory structures used by a module
6	Global	Global variables
7	Export	Exported elements provided by a module
8	Start	An optional start function to initiate a module
9	Element	Elements defined by a module
10	Code	The body of the functions defined by a module
11	Data	The data elements defined by a module
12	Data Count	The number of data elements defined by a module

WASI (WebAssembly System Interface)

- Cross platform applications and games
- Code re-use between platforms and use cases
  - Video-editing, ML, Virtual Reality, Games
- Running applications written in any Wasm/Wasi compilable language on a single runtime
- Containerizing applications and their dependencies
  - This would not be a replacement for containerization, but could be a better option for applications

WASI Goals

- WASI enables wasm module to run well outside of the browser
- Proposals
  - I/O
  - Filesystem
  - Clocks
  - Machine Learning (wasi-nn)

WAGI

- Without networking in WebAssembly we had no way to write services that could load WebAssembly modules to handle http requests.
- WAGI and CGI
  - WAGI abides the CGI 1.1 spec (RFC 3875). It defines a few extra environment variables unique to Wagi, but compatible with the specification.
  - Wagi maps an HTTP path to a Wasm module (es. *http://smalltalk.com/foo/bar* to the Wasm module *helloSmallTalk.wasm.* When /foo/bar is requested, Wagi loads module and executes it just CGI executes its apps.

```
fn main() {
    println("content-type: text/plain");
    println("");
    println("Hello World");
}
```

WebAssembly and .NET

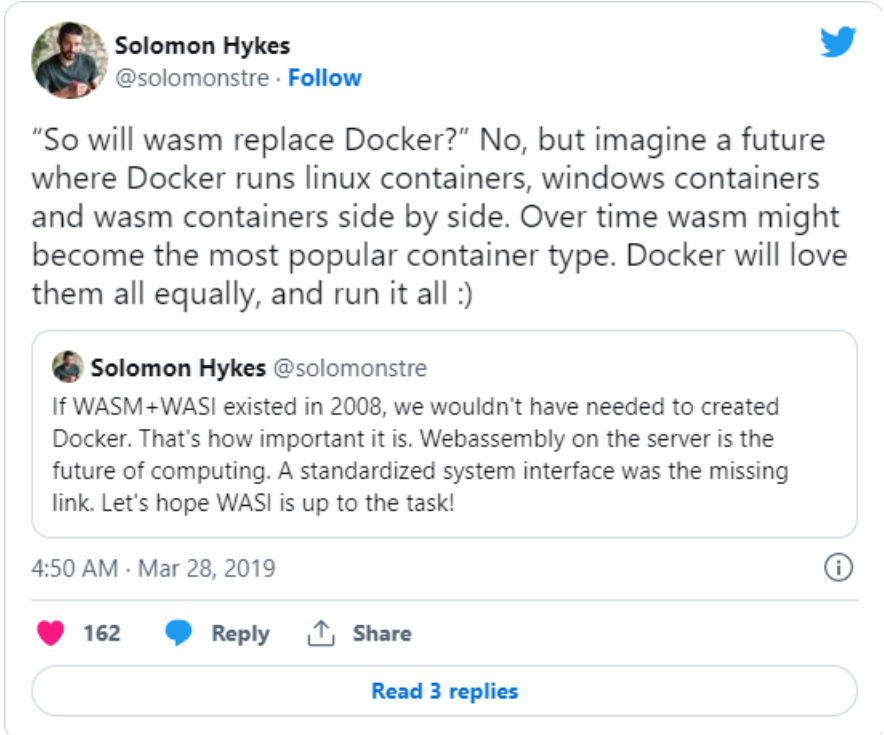
- WebAssembly is a binary exectable format
- C# compiles one binary exceutable formatm .NET bytecode
- Changing it to target another is trickly
  - C# is deeply entwined with .NET standard library
  - C# gets a lot less interesting without NuGet packages
- .NET has taken a different approach to Wasm
- The strategy is to compile the .NET runtime to Wasm bytecode
- Just as Python …
- SteveSandersonMS/dotnet-wasi-sdk:

WebAssembly will replace Docker?

- Wasm is like a container, but with abstraction at a higher level
- Wasm is platform agnostic
- Was runs in an isolated sandbox
- It will be possible to build Wasm-based components

- We can create containers for Windows or for Linux, but not 'universal' containers

WebAssembly and Docker



- Containers and WebAssembly are fast friends, not mortal enemies

[Docker and WebAssembly work well together]

Thanks

alberto.acerbis@intre.it

https://webassembly-studio.kamenokosoft.com/?f=5z61oxzmbch

albertoacerbis