UNIVERSITY OF
Southampton

UNIVERSITY OF Southampton

asm.js and WebAssembly

COMP3220 Web Infrastructure

Dr Nicholas Gibbins – nmg@ecs.soton.ac.uk

A little history of Web scripting

Client-side web scripting first investigated by Netscape in 1995

- Licence Java from Sun Microsystems (i.e. Java applets)
- Use Scheme as a scripting language (simple language in the Lisp family)

Brendan Eich of Netscape chose a third option: create a new language

- C-like syntax
- Weak, dynamic typing
- Object-oriented with prototype-based inheritance
- Multiparadigm – supports functional programming as well as imperative

(and on the tenth day he rested)

A little history of Web scripting

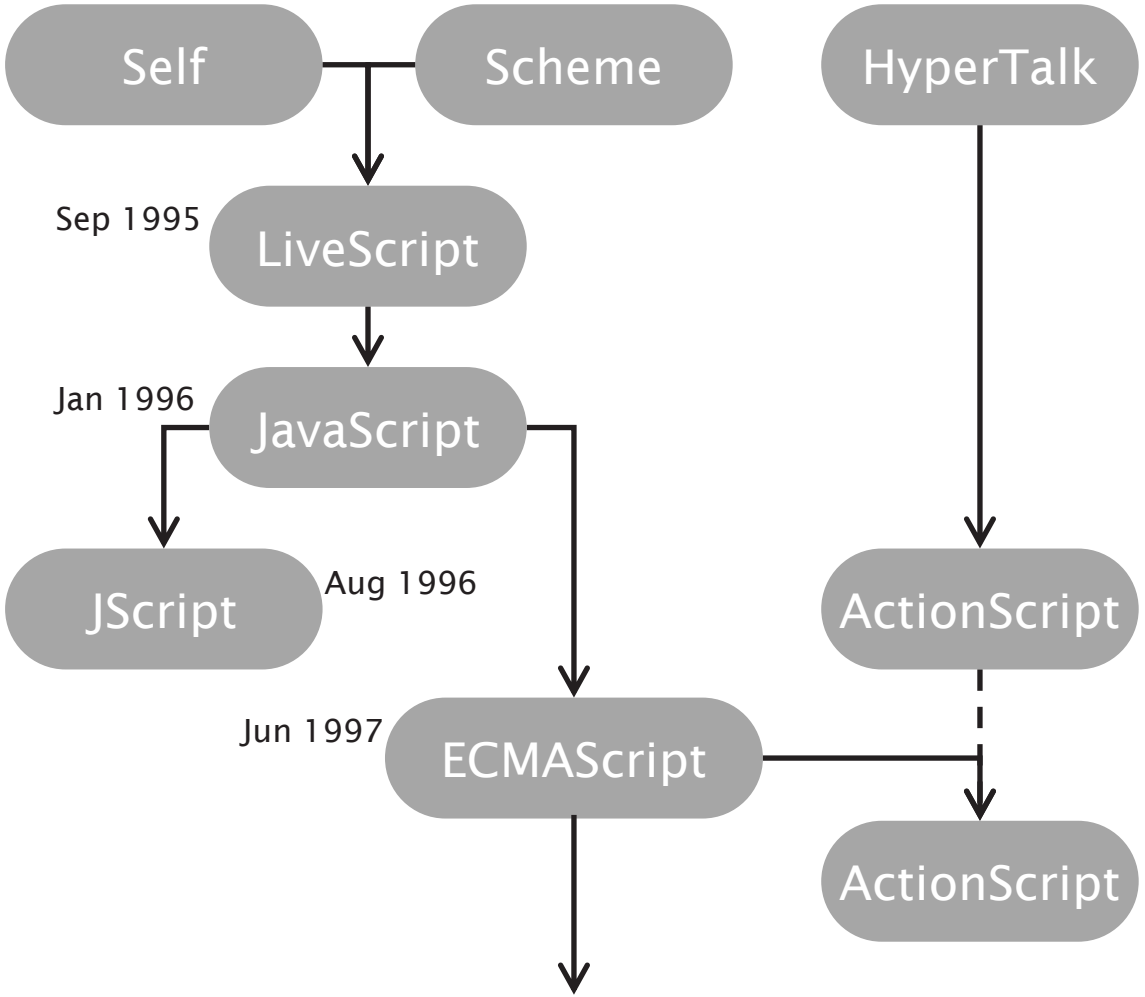
JavaScript released in Netscape 2 in 1996

Reverse engineered by Microsoft as JScript

Submitted to ECMA for standardisation

ECMA-262 published in 1997

JavaScript, JScript and ActionScript as implementations of ECMAScript



A little history of Web scripting

As originally designed, JavaScript allowed only very simple page manipulation

- Event handlers only for forms - `onClick()`, etc
- Early versions predate DOM – `document.write()`

Introduction of XMLHttpRequest in Gecko in 2000 led to growth of AJAX

- Increasingly complex single-page applications
- Efficiency of executing JavaScript as limiting factor

Google's V8 engine introduced JIT compilation of JavaScript in 2008

- Significant performance improvements – compile to intermediate bytecode or native code
- JavaScript for server-side programming (node.js)

Garrett, J.J. (2005) *AJAX: A New Approach to Web Applications*. Available online at: https://web.archive.org/web/20150910072359/http://adaptivepath.org/ideas/ajax-new-approach-web-applications/

asm.js

Strict subset of ECMAScript 2015 (ES6) used as target for source-to-source compilers

- Write in C, compile to asm.js, run in any browser
- Allows APIs such as OpenGL and SDL to be used within web pages
- No garbage collection (typed array for memory), static typing (enabling AOT compilation)

Typically runs at ~2/3 the speed of native code

Compiler implementation (emscripten) as LLVM backend

- Straightforward support for other languages: compile to LLVM IR, compile LLVM IR to asm.js

http://asmjs.org/
Zakai, A. (2013) *Big Web app? Compile it!* Available at: https://kripken.github.io/mloc_emscripten_talk/

Web Assembly

asm.js improves runtime performance, but parsing JavaScript syntax is costly

The solution: adopt a standard bytecode

WebAssembly is a bytecode for a stack-based virtual machine

- cf. the stack-based machine used by SeaMonkey or the register machine used by V8

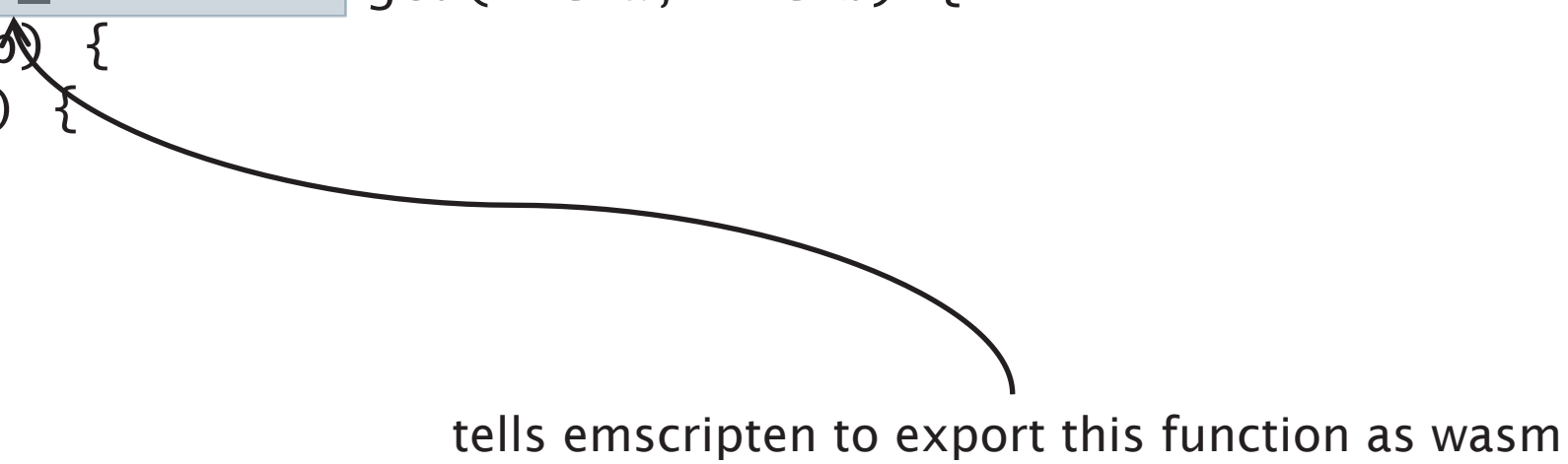
Compile $LANGUAGE to .wasm using emscripten, as for asm.js

Eich, B. (2015) *From asm.js to WebAssembly*. Available at: https://brendaneich.com/2015/06/from-asm-js-to-webassembly/

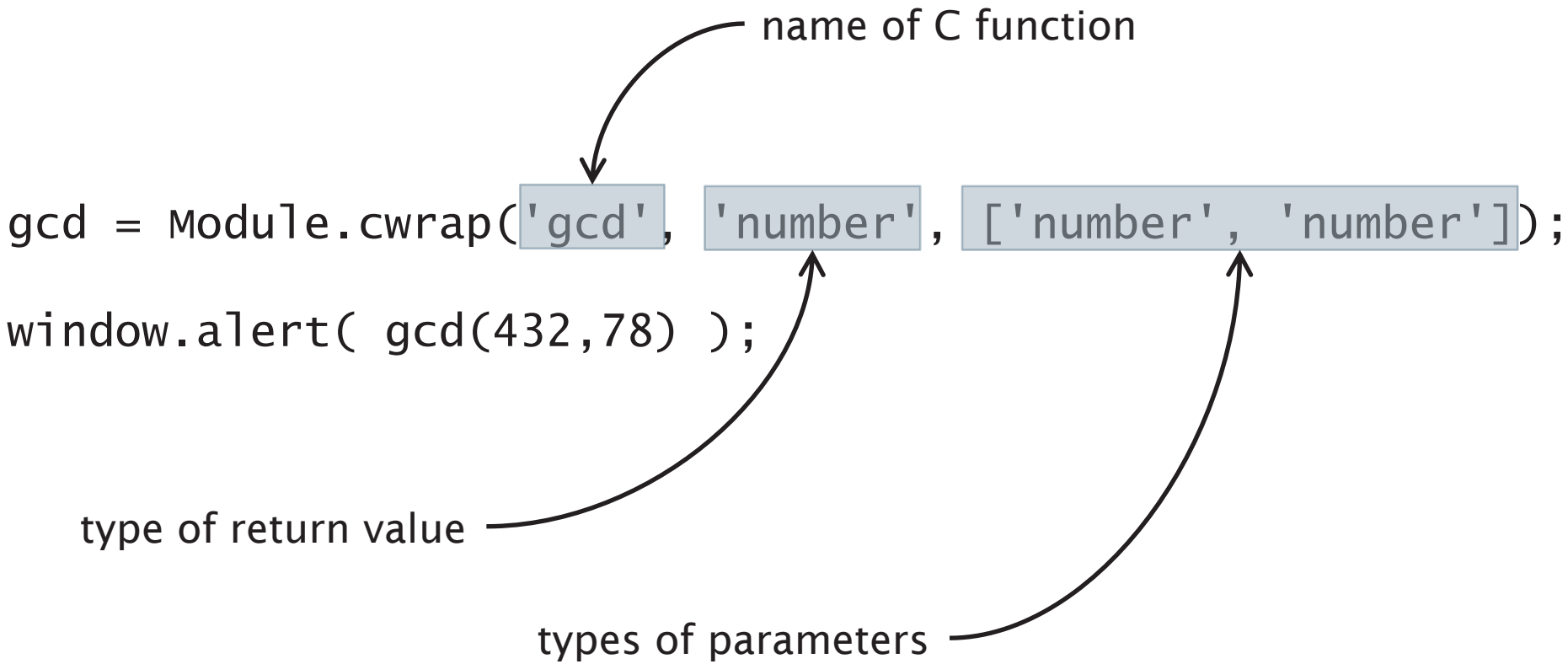
C function...

```
#include <emscripten.h>


int EMSCRIPTEN_KEEPALIVE gcd(int a, int b) {
  while(a != b) {
    if (a > b) {
      a -= b;
    } else {
      b -= a;
    }
  }
  return a;
}
```



...called from JavaScript



Further Reading

ECMA-262 (ECMAScript standard)

https://www.ecma-international.org/publications/standards/Ecma-262-arch.htm

asm.js Specification

http://asmjs.org/

emscripten compiler

https://github.com/emscripten-core/emscripten

WebAssembly Specification

https://webassembly.org/

WebAssembly Binary Toolkit ("wabbit")

https://github.com/WebAssembly/wabt

UNIVERSITY OF Southampton

Next Lecture: Web Formats