

# The rise of non-JavaScript frameworks using WebAssembly

Boyan Mihaylov  
@boyanio  
boyan.io

WebAssembly (WASM) is compiler  
target for programs on the Web

```
C:\wasm>type index.c
```

```
#include <stdio.h>
```

```
int main(void) {  
    printf("Hello, cool people!\n");  
    return 0;  
}
```

```
C:\wasm>clang index.c
```

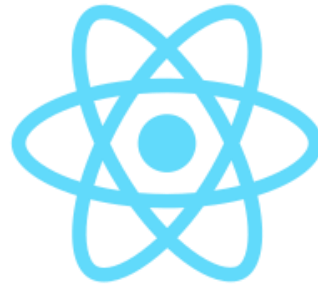
```
C:\wasm>a.exe
```

```
Hello, cool people!
```

```
C:\wasm>emcc -o a.js index.c
```

```
C:\wasm>node a.js
```

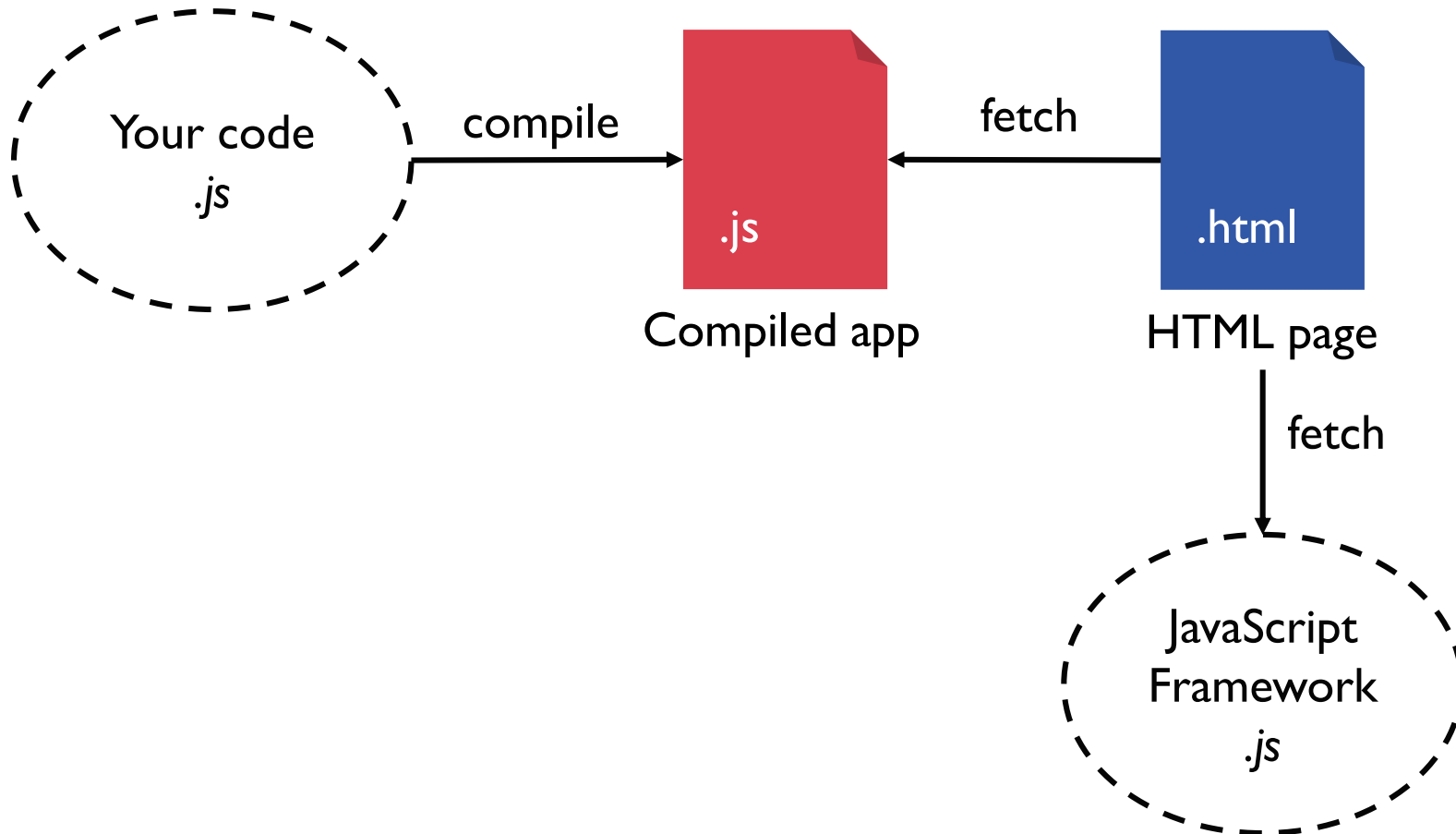
```
Hello, cool people!
```



# The Web of JavaScript frameworks



# JavaScript frameworks architecture

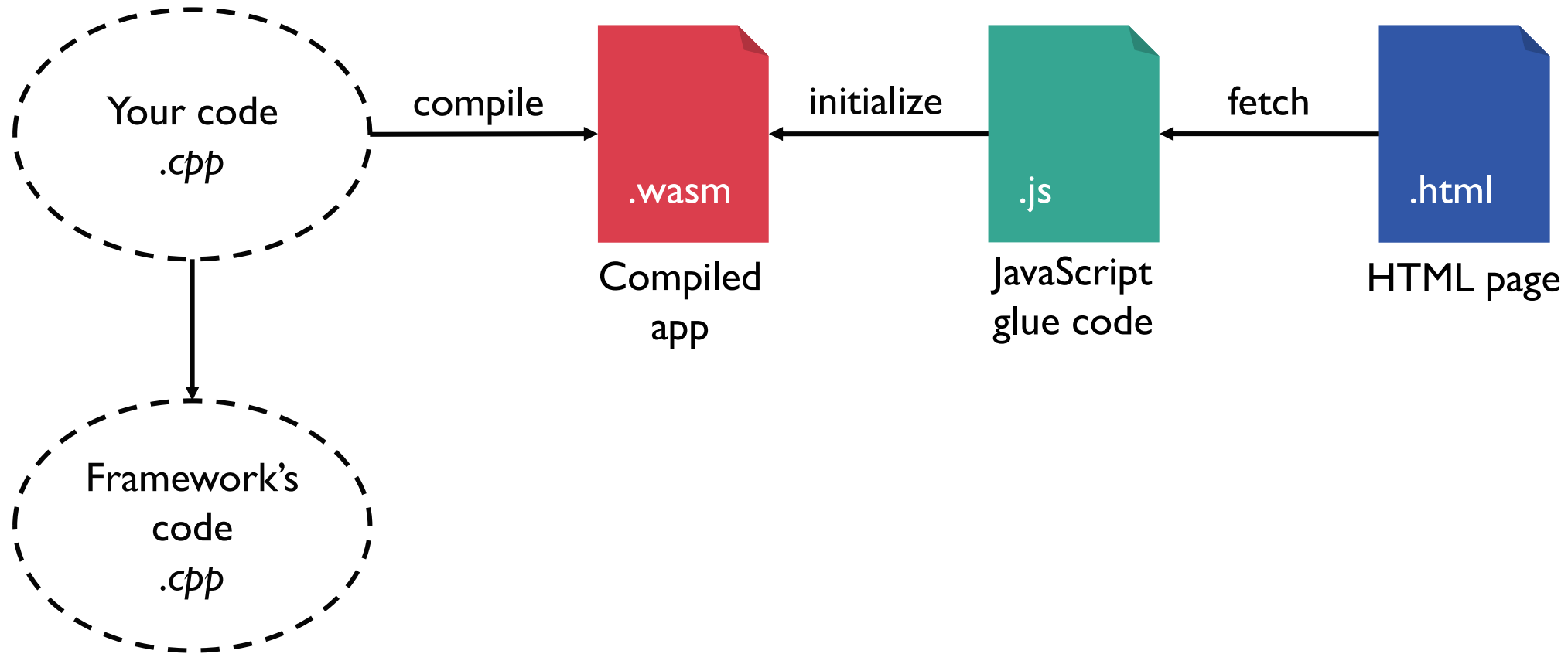


How would WebAssembly  
influence the way we do  
Web development today?

Rewriting existing JavaScript  
frameworks into a language  
that can be compiled to  
WebAssembly



# WebAssembly-compiled frameworks





C++/Python → JavaScript → C++/Python



The graphic features a cartoon character with purple hair, wearing a brown flight suit and goggles, pointing upwards. In the background, there are blue clouds, a large white moon, and several interlocking gears. An orange square with the letters 'JS' is in the top left corner.

**JavaScript Coding**

Half Day Camp | Grades 7-8

Learn JavaScript and build your own games for web and mobile platforms

# No direct DOM access

## index.c

```
extern void createElement(void);

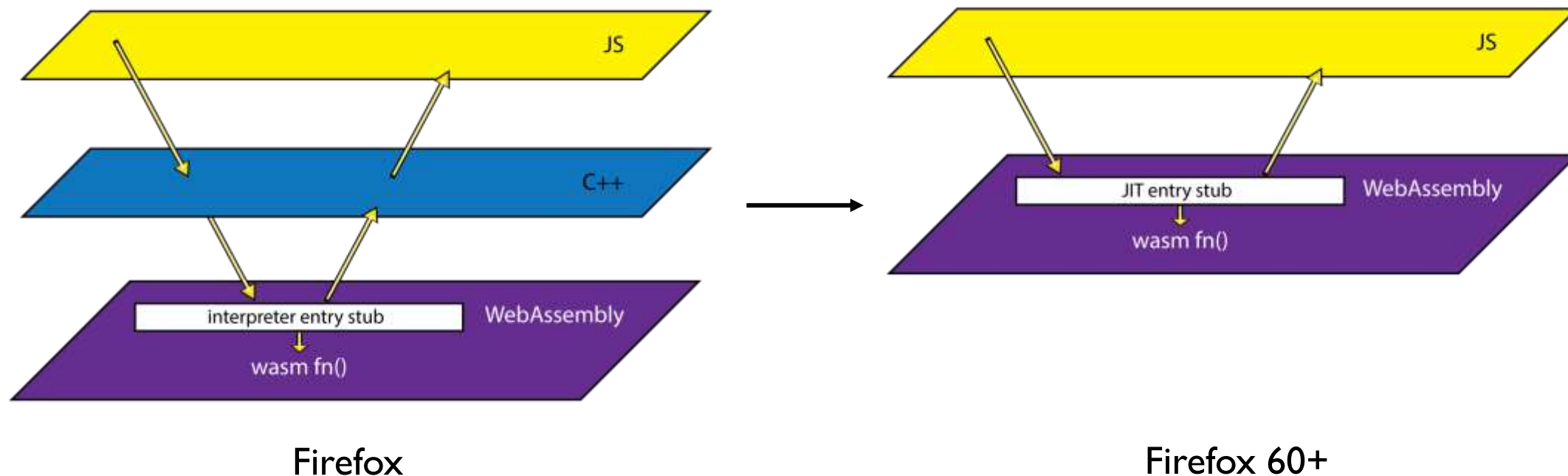
int main(void)
{
    createElement();
    createElement();
    ...
    return 0;
}
```

## main.js

```
const imports = {
    createElement: () => {
        document.createElement('div');
    }
};

WebAssembly.instantiate(..., imports);
```

# JavaScript → WebAssembly overhead



# Easier to create fast native mobile apps



Rewriting *parts* of existing  
JavaScript frameworks into  
a language that can be compiled  
to WebAssembly



2



# Languages

- Java: [RxJava](#)
- JavaScript: [RxJS](#)
- C#: [Rx.NET](#)
- C#(Unity): [UniRx](#)
- Scala: [RxScala](#)
- Clojure: [RxClojure](#)
- C++: [RxCpp](#)
- Lua: [RxLua](#)
- Ruby: [Rx.rb](#)
- Python: [RxPY](#)
- Go: [RxGo](#)
- Groovy: [RxGroovy](#)
- JRuby: [RxJRuby](#)
- Kotlin: [RxKotlin](#)
- Swift: [RxSwift](#)
- PHP: [RxPHP](#)
- Elixir: [reaxive](#)
- Dart: [RxDart](#)

# LibSass - Sass compiler written in C++

Currently maintained by Marcel Greter (@mgreter) and Michael Mifsud (@xzyfer)

Originally created by Aaron Leung (@akhleung) and Hampton Catlin (@hcatlin)

build passing build passing coverage 86% open issues 4% issue resolution 2 d bountysource \$130 in 0 bounties slack 2/146

[LibSass](#) is just a library! If you want to use LibSass to compile Sass, you need an implementer. Some implementations are only bindings into other programming languages. But most also ship with a command line interface (CLI) you can use directly. There is also [SassC](#), which is the official lightweight CLI tool built by the same people as LibSass.

## Excerpt of "sanctioned" implementations:

- <https://github.com/sass/node-sass> (Node.js)
- <https://github.com/sass/perl-libsass> (Perl)
- <https://github.com/sass/libsass-python> (Python)
- <https://github.com/wellington/go-libsass> (Go)
- <https://github.com/sass/sassc-ruby> (Ruby)
- <https://github.com/sass/libsass-net> (C#)
- <https://github.com/medialize/sass.js> (JS)
- <https://github.com/bit3/jsass> (Java)

This list does not say anything about the quality of either the listed or not listed [implementations](#)!

The authors of the listed projects above are just known to work regularly together with LibSass developers.

 vuejs / vue

 Watch ▾

4,966

★ Star

95,870

Fork

14,116

<> Code

! Issues 130

🔗 Pull requests 76

📁 Projects 0

📊 Insights

Webassembly integration. Split the core into two parts. #8193

<https://github.com/vuejs/vue/issues/8193>

 glimmerjs / glimmer-vm

 Watch ▾

77

★ Star

861

Fork

111

<> Code

! Issues 65

🔗 Pull requests 12

📁 Projects 0

📖 Wiki

📊 Insights

Initial stab at porting `asm/stack.ts` to Rust #752

<https://github.com/glimmerjs/glimmer-vm/pull/752>



Writing custom components  
in a language that can be  
compiled to WebAssembly



3

# Angular & WebAssembly

A collection of examples of how WebAssembly can be used with Angular



[Home](#) [GitHub](#) [Twitter](#)

Fibonacci battlefield

Console logger

Text to ASCII art converter

Bitmap to ASCII art converter

3D cube

Proof of work

# Angular & WebAssembly

<https://boyan.io/angular-wasm/>

# Emergence of new non-JavaScript frameworks using WebAssembly



4

# Blazor

Full-stack web development with C# and WebAssembly

🚀 [Get Started](#)



## Build a Web UI with C#

Blazor is an experimental .NET web framework using C# and HTML that runs in the browser.

[What is Blazor?](#)

**Blazor**  
<https://blazor.net>

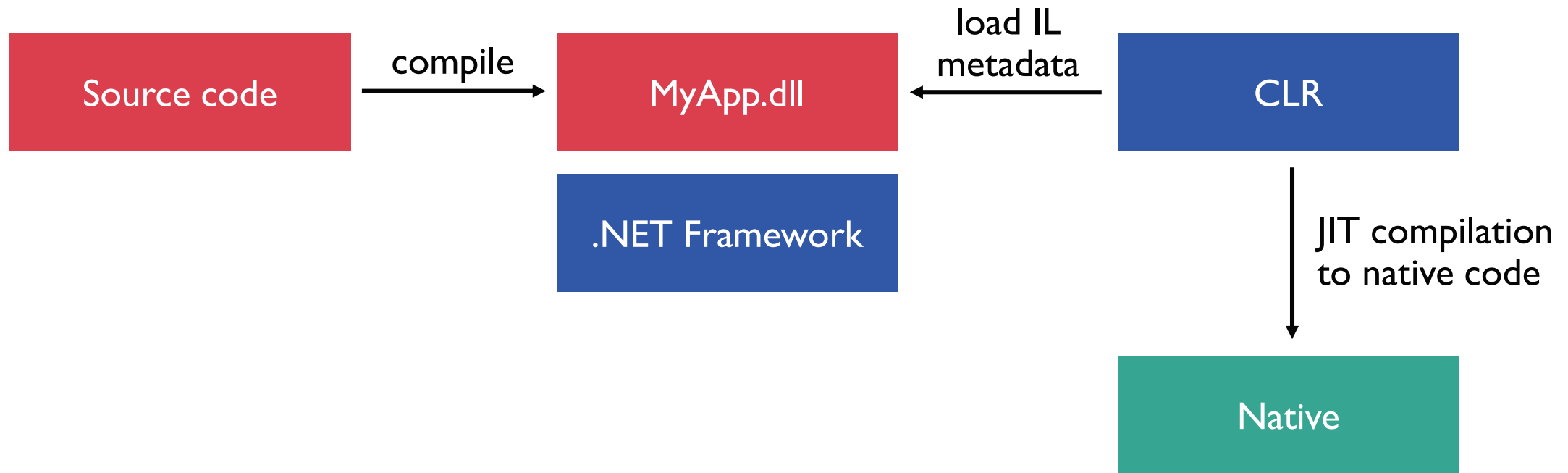
## Full-stack .NET

Do full-stack .NET development using stable and consistent tools, languages, and APIs both in the browser and on the server.

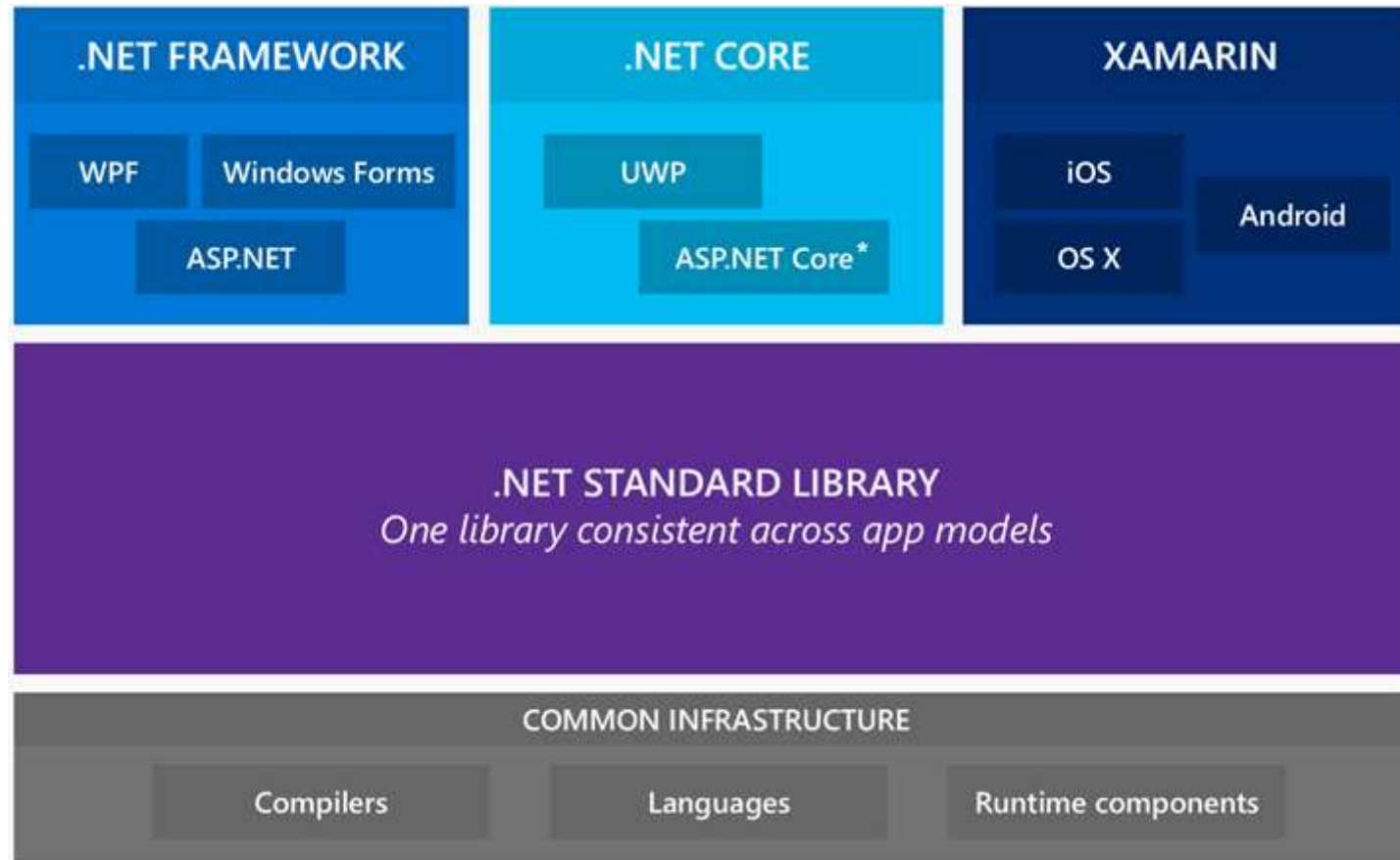
[Learn more about the .NET platform](#)



# Traditional .NET architecture



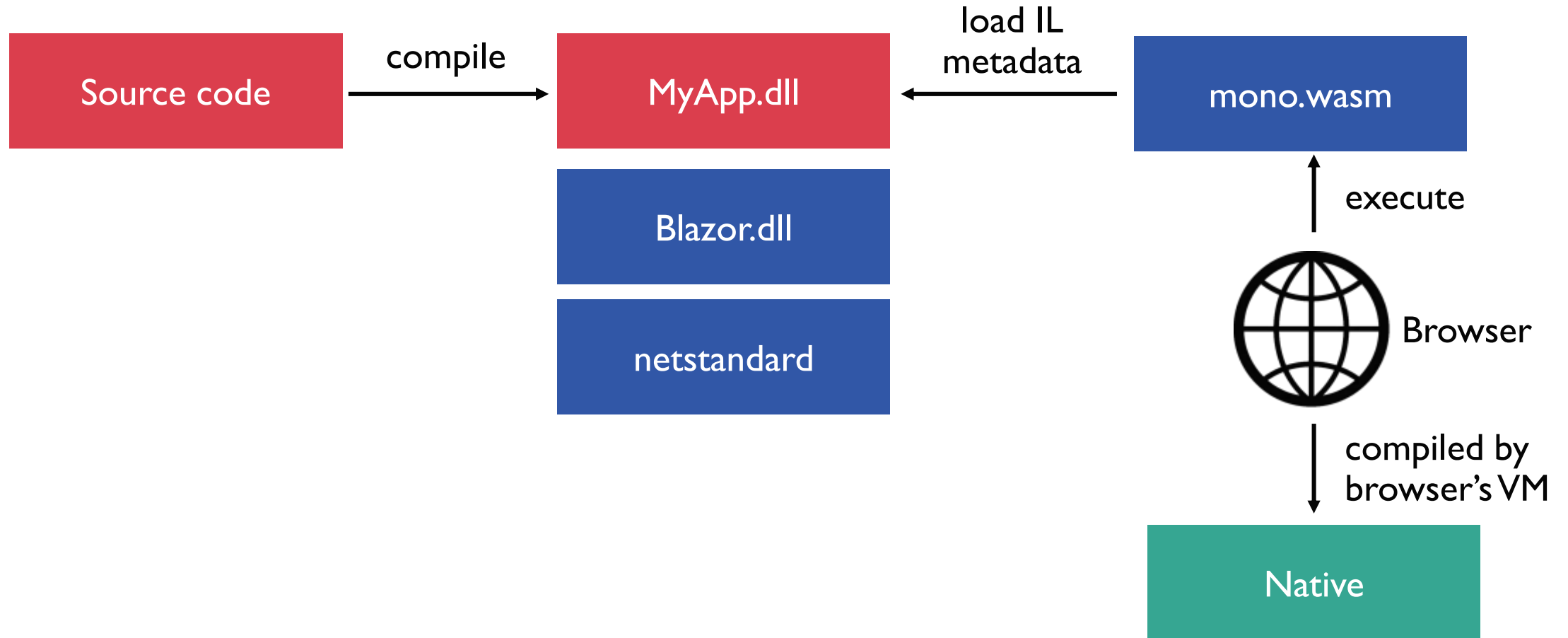
# .NET Core architecture



# Mono is an open source implementation of Microsoft's .NET Framework



# Blazor architecture





# React vs. Blazor

This demo shows how React apps can live together with Blazor apps, which are basically C# apps running in the browser with the help of WebAssembly.

## React chat Blazor chat

Send

It's coming

B

22 seconds ago

 Not loading fast yet...

28 seconds ago

Hi! I am cool, a?!

B


54 seconds ago

Send

It's coming

B

22 seconds ago


 Not loading fast yet...

28 seconds ago

Hi! I am cool, a?!

B

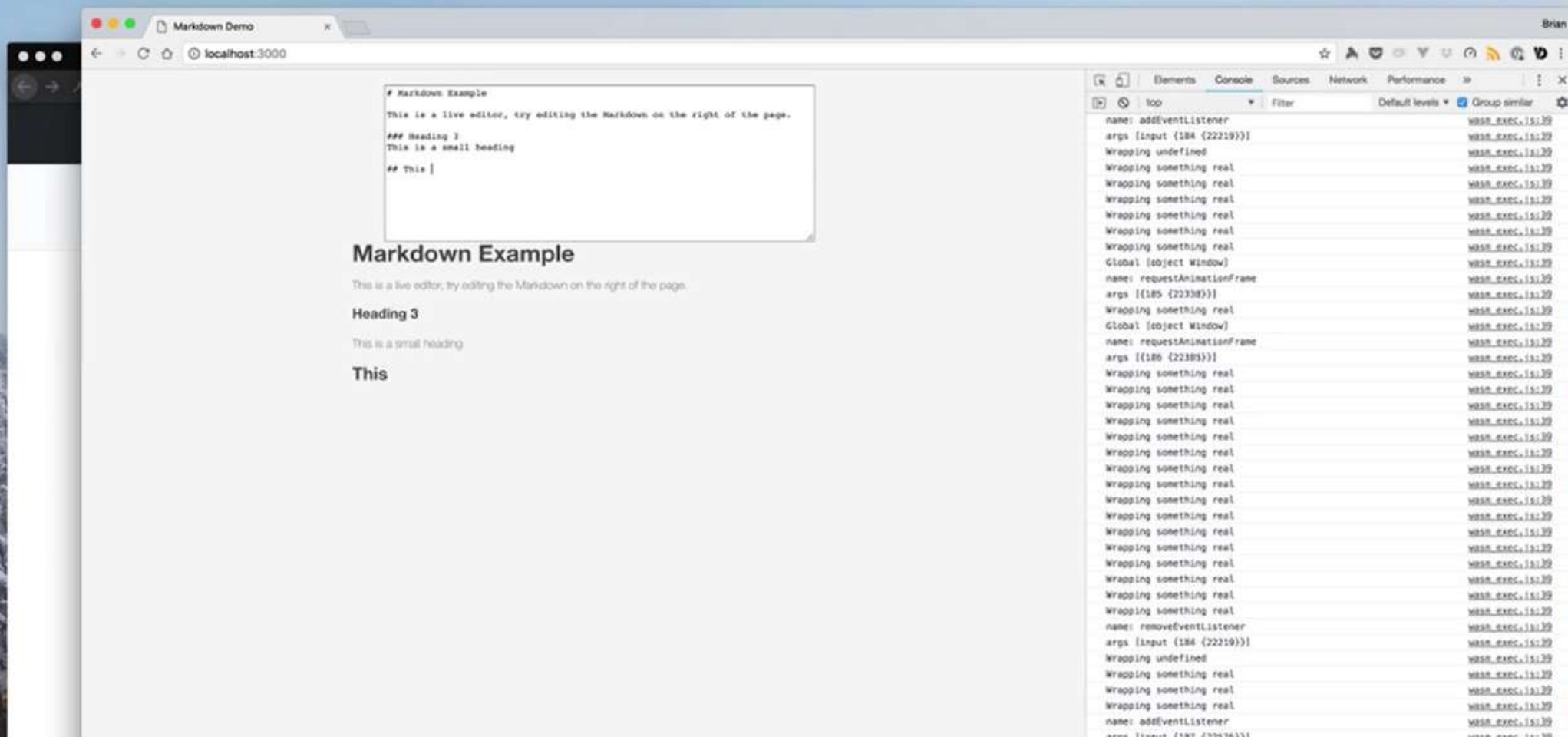
54 seconds ago

 Hello!

1 minute ago

# React vs. Blazor

<https://boyan.io/react-blazor/>



# Blazor-inspired Web framework in Go

<https://github.com/bketelsen/wasmplay>

WebAssembly  
enables different  
languages to work  
together on the Web

<https://boyan.io/wasm-wheel/>



The future of Web  
belongs to those, who compile

Boyan Mihaylov / @boyanio / boyan.io