Blazor
C# running in the browser via WebAssembly

Scott Sauber
Audience

• Mostly targeted for .NET developers
• JS Developers interested in WebAssembly
Agenda

• What is WebAssembly?
• What is Blazor?
• How does Blazor work?
• Demos
• Questions
Purpose

• Differentiate what is Blazor vs WebAssembly
• Get excited for the future
Who am I?

• Lead Software Developer at Iowa Bankers
• Primarily .NET Developer
• React fanboy
• Actually enjoys JavaScript
• Blog primarily on ASP.NET Core on [scottsauber.com](http://scottsauber.com)
Current State of the SPA Front End

Pick a Language:
- ES6
- TS
- flow
- RE
- cljs

Pick a Framework:
- React
- Angular
- Vue
- Ember
- Next

Pick your tools:
- Gulp
- Babel
- ESLint
- Common
- Airbnb
- standard
- Google
## So. Many. Decisions.

<table>
<thead>
<tr>
<th>Editor</th>
<th>Bundler</th>
<th>Project structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which one?</td>
<td>Webpack, Browserify, Rollup...</td>
<td>By file type or feature?</td>
</tr>
<tr>
<td>Which plugins?</td>
<td>Linting</td>
<td>Centralize API?</td>
</tr>
<tr>
<td>Use built in terminal?</td>
<td>Which linter?</td>
<td>Allow Inline JS?</td>
</tr>
<tr>
<td>Editor config</td>
<td>Enable which rules?</td>
<td>Extract to POJOs?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Module format</th>
<th>HTML generation</th>
<th>HTTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES6 Modules, CommonJS...</td>
<td>Minify?</td>
<td>Library</td>
</tr>
<tr>
<td>HTML generation</td>
<td>Use plugin?</td>
<td>Mock schema format</td>
</tr>
<tr>
<td>Minify?</td>
<td>Inject prod only concerns?</td>
<td>Mock data generation</td>
</tr>
<tr>
<td>Use plugin?</td>
<td>Templating language?</td>
<td>Mock server</td>
</tr>
<tr>
<td>Inject prod only concerns?</td>
<td>Transpiling</td>
<td>Production build</td>
</tr>
<tr>
<td>Templating language?</td>
<td>Native ES or diff language?</td>
<td>Minification</td>
</tr>
<tr>
<td>Transpiling</td>
<td>Use experimental features?</td>
<td>Sourcemaps</td>
</tr>
<tr>
<td>Native ES or diff language?</td>
<td>Which plugins?</td>
<td>Bundle splitting</td>
</tr>
<tr>
<td>Use experimental features?</td>
<td>Production vs dev config</td>
<td>Cache busting</td>
</tr>
<tr>
<td>Which plugins?</td>
<td>Production vs dev config</td>
<td>Error logging</td>
</tr>
<tr>
<td>Production vs dev config</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: Cory House’s [Building a JavaScript Development Environment Pluralsight course](https://www.pluralsight.com/courses/building-javascript-development-environment)
At the end of the day....
Problems

• Whole host of people don’t like JS
  • Dynamically typed
  • Less integration, more stitching
  • Browser support
  • Moves too fast, lots of choice, intimidating
  • node_modules

• SPA’s are more expensive to maintain
  • Front-end + Back-end team
  • Training up full stack to be great at both (very difficult)

• When using a different language than JS on the backend...
  • Duplicate Business Logic (like validation)
    • Or just have server
    • Plz don’t just have client...plz
  • No IDE/compiler help between backend models + front end making AJAX calls
    • Unless bringing in yet another tool
What is Web Assembly (WASM)?

- WebAssembly (WASM) is a low-level binary format language that can be run in modern web browsers that runs at near-native speeds.
- Compilation Target for other languages
- Browser standard
- No more JS monopoly
Is WASM Ready?

And you can polyfill WASM with asm.js!
What is Blazor?

• Blazor is an experimental .NET SPA framework maintained by Microsoft using C# and HTML that runs in the browser via WebAssembly....
Wait a second....

It’s a Standard, not a plugin!
What is Blazor?

• Blazor is an **experimental** .NET SPA framework maintained by Microsoft using C# and HTML that runs *in the browser* via WebAssembly....

• Uses Razor syntax
  • Browser + L + Razor = Blazor

• Uses component-based architecture

• Runs on top of Mono
  • Blazor == UI Framework == MVC or Web Forms
  • Mono == Runtime == .NET Framework or .NET Core

• Development led by Steve Sanderson, of KnockoutJS fame
So I can write C# in the Browser!?!?

• Blazor is .NET Standard 2 compliant
• However, not all .NET Standard 2 API’s are implemented running in browser make sense
  • Examples
    • System.Net.Mail
    • System.IO
  • These throw Platform Not Supported exceptions
• But a lot do make sense
  • HttpClient => AJAX
Blazor Provides Calling C# from JS + vice versa

• C# Wrappers on top of JS API’s
  •LocalStorage
  •PaymentRequest
  •Or any npm library
• C# maps to JS pretty well
  •async/await
  •Task => Promise
• Future: automatically read TSD’s and generate C# bindings
Why would you be interested in this?

• C# is a fantastic language
  • ...not that JavaScript isn’t
  • ...but statically typed languages are winning (see: TS, Flow, Reason, etc.)
    • Airbnb React Native blog
• Share logic with existing .NET backend
  • Validation logic
  • Models from Server when retrieve from the Client
• Get off the JS churnwagon
• Consolidate frontend and backend teams under one language
Demo #1

• **Install VS Extension**
  • Templates, Razor Tooling
• Hello World on Blazor
• Component Architecture
• Dependency Injection
• Sharing models
• Sharing validation logic
Rapid Fire Questions

• How big is it?
  • 1.8MB in Dev, 1.3MB in Prod
  • Very little work done thus far to optimize

• Do WASM files cache like JS and CSS files?
  • Yes

• How does it work under the hood?
How does Blazor work?

**Today**

- **Dev time**
  - C# / Razor source files
  - `mono.wasm`
    - WebAssembly binary executed natively
  - `YourApp.dll`
    - netstandard .NET assembly files

- **Runtime (in browser)**
  - Browser APIs
    - visible DOM, HTTP requests, etc.

**Future**

- **Dev time**
  - C# / Razor source files
  - `YourApp.dll`

- **Runtime (in browser)**
  - Browser APIs
    - visible DOM, HTTP requests, etc.
  - `mono.wasm`
    - WebAssembly binary executed natively
  - `YourApp.wasm`
    - WebAssembly binary executed natively
Why Mono? Why not .NET Core?

- Already Client-side-focused
  - Xamarin, Unity, etc.
  - .NET Core is Server-side-focused
- Already developed for unique platforms (iOS, watchOS, PS4, etc.)
- Already had linker (DLL trimmer/tree shaker) for Xamarin
- They got it working first
- Long term they want to consolidate on .NET Core
Demo #2

• LocalStorage C# Wrapper
• Code: [https://github.com/scottsauber/BlazorToDoMVC](https://github.com/scottsauber/BlazorToDoMVC)
What else can we do?

- Blazor’s component model is de-coupled from the Browser
- Blazor on...
- The Server
  - Possible Replacement for MVC/Razor Pages if you prefer component-based over MV*?
  - Changes streamed via WebSocket
- Electron
  - Cross-platform desktop framework. Write once, run anywhere.
  - Proof of Concept Running on .NET Core
- Why?
  - Faster Code Execution
  - Full Debugger in VS
  - .NET Core instead of Mono
  - Access to Desktop API’s
Demo #3

• Blazor on Electron
  • Electron.App

• Code: https://github.com/SteveSandersonMS/BlazorElectronExperiment.Sample
What’s the hold up?

• Currently 0.4 – won’t ship this year per Damian Edwards

• What’s there
  • Component Model
  • Routing
  • Layouts
  • Dependency Injection
  • JS interop
  • Share Components between projects

• What’s still coming
  • Better tooling
  • Forms and Validation
  • Debugging
  • Hot reloading
  • AOT
  • Better Linker Assembly Trimming
# So. Many. Decisions.

<table>
<thead>
<tr>
<th>Editor</th>
<th>Bundler</th>
<th>Project structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which one?</td>
<td>Webpack, Browserify, Rollup...</td>
<td>By file type or feature?</td>
</tr>
<tr>
<td>Which plugins?</td>
<td>Linting</td>
<td>Centralize API?</td>
</tr>
<tr>
<td>Use built in terminal?</td>
<td>Which linter?</td>
<td>Allow Inline JS?</td>
</tr>
<tr>
<td>Editor config</td>
<td>Enable which rules?</td>
<td>Extract to POJOs?</td>
</tr>
<tr>
<td>Module format</td>
<td>Warning or error?</td>
<td></td>
</tr>
<tr>
<td>ES6 Modules, CommonJS...</td>
<td>Which plugins?</td>
<td></td>
</tr>
<tr>
<td>HTML generation</td>
<td>Use a preset?</td>
<td></td>
</tr>
<tr>
<td>Minify?</td>
<td>Testing</td>
<td></td>
</tr>
<tr>
<td>Use plugin?</td>
<td>Framework?</td>
<td></td>
</tr>
<tr>
<td>Inject prod only concerns?</td>
<td>Assertion Library?</td>
<td></td>
</tr>
<tr>
<td>Templating language?</td>
<td>Helpers?</td>
<td></td>
</tr>
<tr>
<td>Transpiling</td>
<td>Test file location?</td>
<td></td>
</tr>
<tr>
<td>Native ES or diff language?</td>
<td>File naming?</td>
<td></td>
</tr>
<tr>
<td>Use experimental features?</td>
<td>What environment?</td>
<td></td>
</tr>
<tr>
<td>Which plugins?</td>
<td>Mocking?</td>
<td></td>
</tr>
<tr>
<td>Production vs dev config</td>
<td>Code Coverage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Continuous Integration</td>
<td></td>
</tr>
</tbody>
</table>

**HTTP**
- Library
- Mock schema format
- Mock data generation
- Mock server

**Production build**
- Minification
- Sourcemaps
- Bundle splitting
- Cache busting
- Error logging
The remainder of these you’ve likely already decided on the backend!
Current State of the SPA Front End

Pick a Language:
- ES6
- TS
- flow
- RE
- cljs

Pick a Framework:
- React
- Angular
- Vue
- Ember
- Next

Pick your tools:
- Gulp
- Babel
- ESLint
- StandardJS
- Google
Future State of the Front End?

Pick a Language: C#  Java  Python  R  C++  COBOL

Pick a Framework: Blazor

Pick your tools: Visual Studio  IntelliJ IDEA  Xcode
Future

• Blazor is still experimental...
• However... I would be pretty surprised if they don’t ship this.
• Start thinking about “would this code run ok in the browser?”
  • Separate domain + input validation
• But still do NOT commit to Blazor yet for anything remotely real
• I repeat
Takeaways

• WASM is AWSM
• Potential of Blazor
• WASM has potential to radically disrupt WebDev
Resources

• [https://blazor.net](https://blazor.net)
  • Microsoft Documentation

• [https://learn-blazor.net](https://learn-blazor.net)
  • Community-led Documentation

• [https://github.com/aspnet/blazor](https://github.com/aspnet/blazor)
  • Blazor Source Code

• [https://github.com/scottsauber/BlazorToDoMVC](https://github.com/scottsauber/BlazorToDoMVC)
  • ToDoMVC Blazor example

• [https://github.com/SteveSandersonMS/BlazorElectronExperiment.Sample](https://github.com/SteveSandersonMS/BlazorElectronExperiment.Sample)
  • Blazor on Electron
Questions?
Thanks!