Here Be Dragons
JS Debugging

Rami Sayar - @ramisayar

Senior Technical Evangelist

Microsoft Canada

Here be dragons!

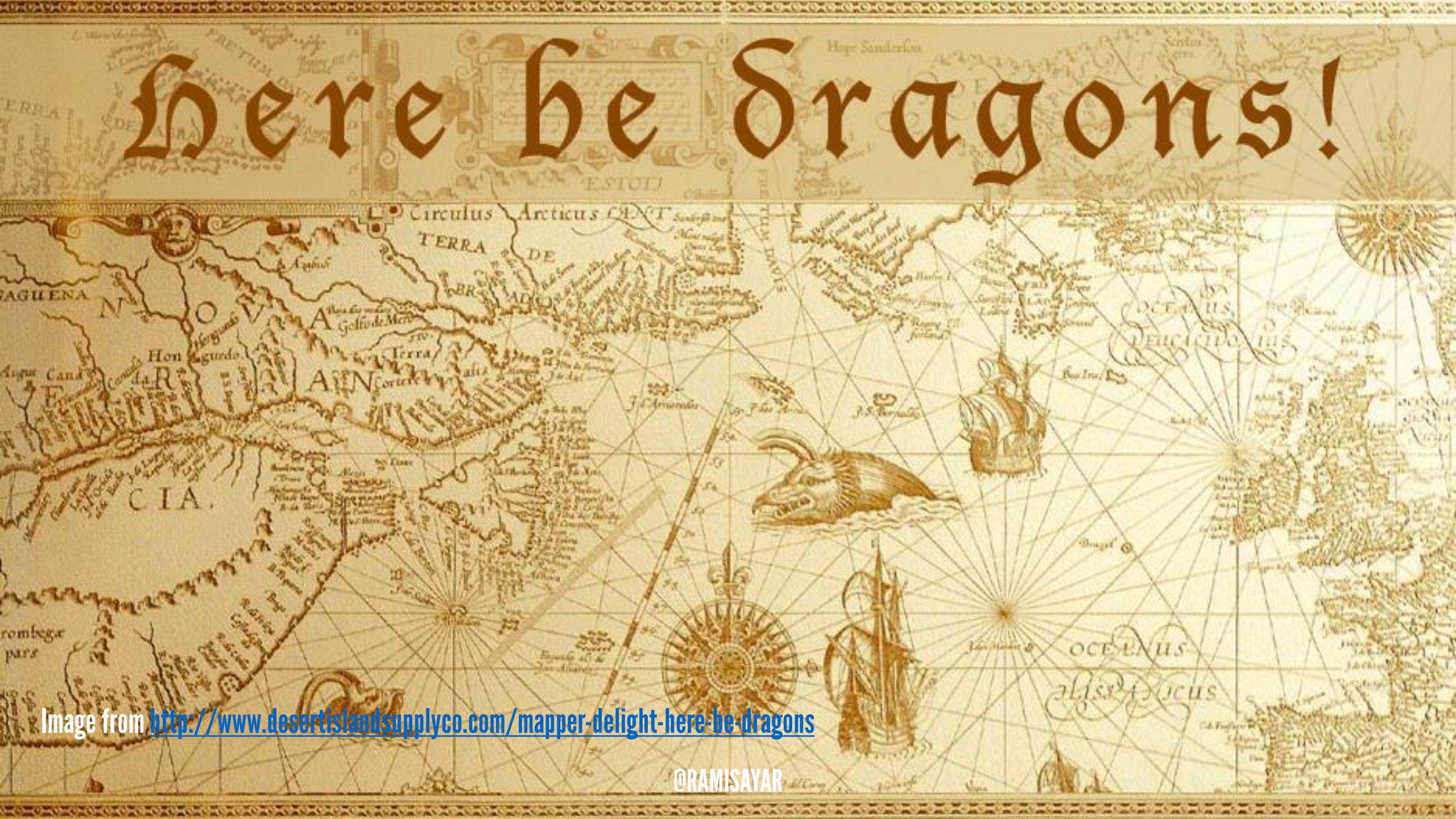
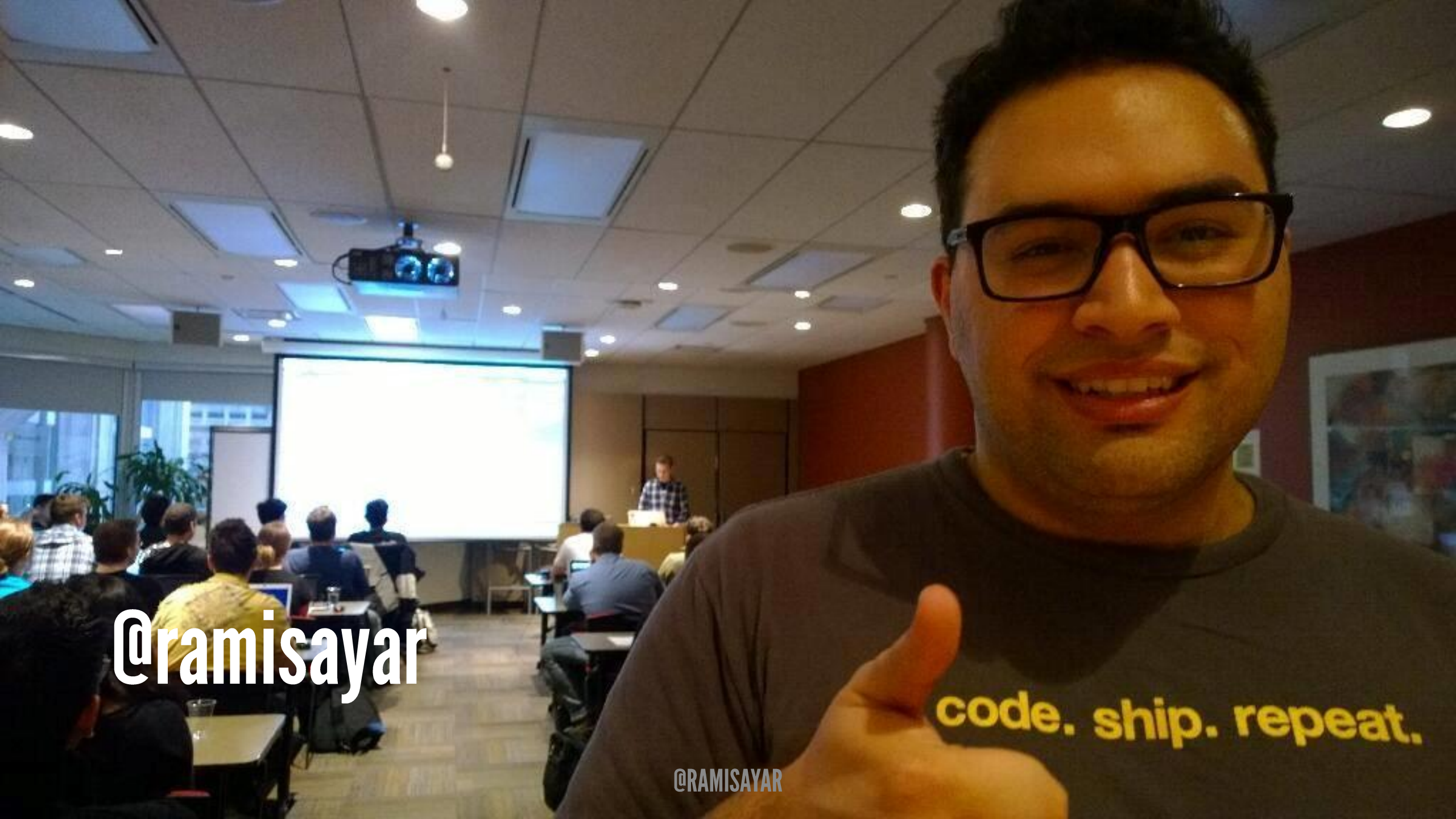


Image from http://www.desertislandsupplyco.com/mapper-delight-here-be-dragons

@RAMISAYAR

Agenda

- Types of Errors in JavaScript
- Frequent Locations of Errors in JavaScript
- Debugging and Introspection Tools
- Remote Debugging Node Processes
- Remote Debugging Front-End JavaScript


@ramisayar
code. ship. repeat.
@RAMISAYAR

99 little bugs in the code.
99 little bugs in the code.
Take one down, patch it around.

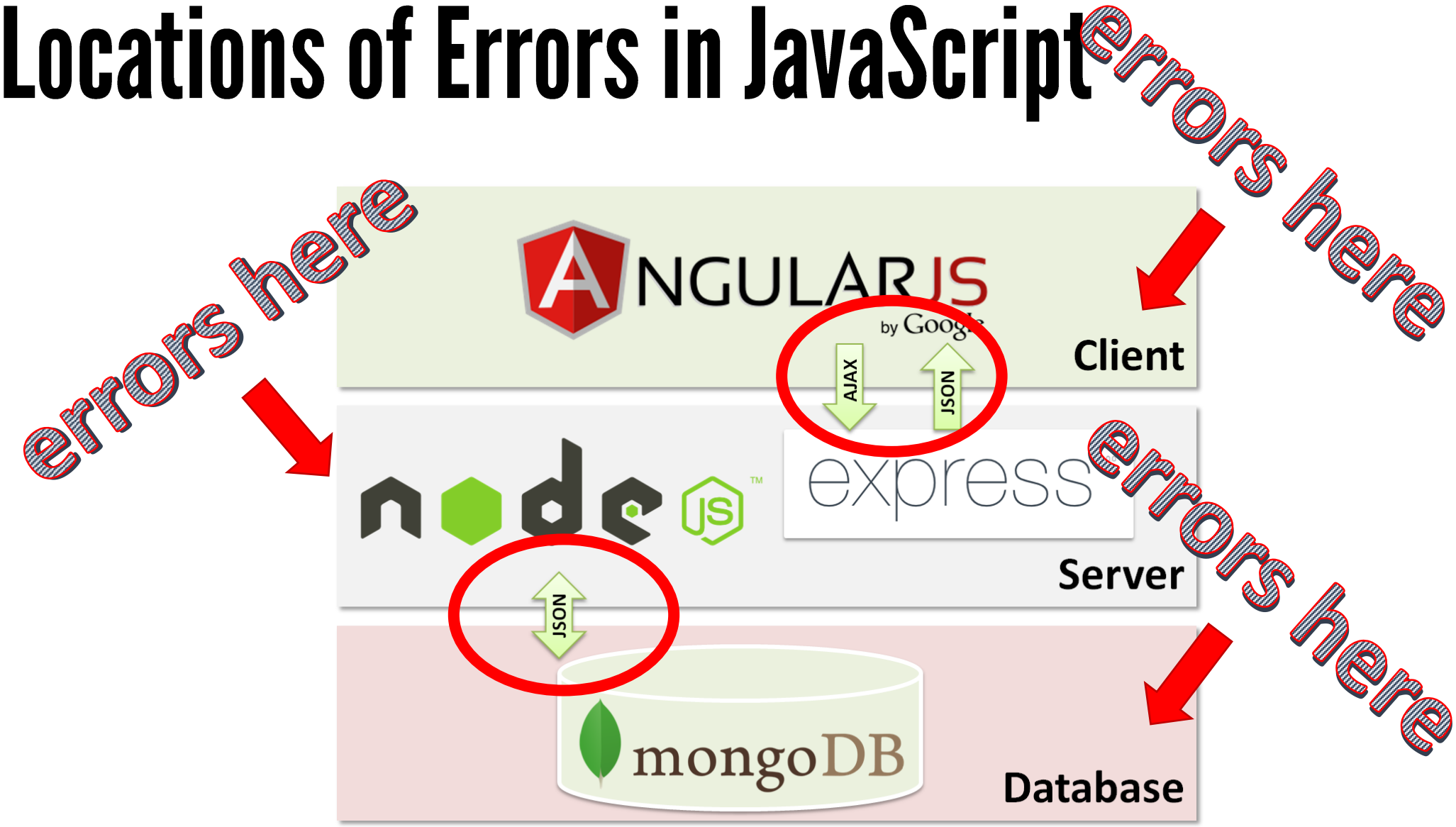
127 little bugs in the code...

Image from developermeme.com

Types of Errors in JavaScript

- Loading Errors
  - E.g. Incorrect syntax, minification errors, network-related errors, missing files, etc.
- Runtime Errors
  - E.g. Syntax errors, misspelled variables, illegal assignment, variables/classes don't exist, etc.
- Logic Errors
  - E.g. BUGS! But also errors due to format and parameters (JSON vs XML), etc...

Locations of Errors in JavaScript



JAVASCRIPT ERRORS

@RAMISAYAR

memegenerator.net


But what really happens...
@RAMISAYAR

Image from
https://www.reddit.com/r/aww/comments/2303gv/fly_you_fools/

Debugging & Introspection Tools

DEMOS!

Typical Logic Errors in JavaScript

- Using this wrong an scoping errors
- Accidentally creating memory leaks
  - Dangling references to unused objects
  - Circular references
- Incorrect coercion, comparisons and equality
- Incorrect references to instance objects & prototypical inheritance errors
- So much more…

Debugging & Introspection Tools

- Allow you to control execution and walk through line-by-line
- Debug with breakpoints to conditionally stop execution
- Examine the call stack
- Pause on exceptions
- Stack trace exceptions

Debugging Node Apps

node --inspect

- Since node >v6.3.0, Node has built-in debuggability support with the Chrome DevTools.
  - Complete breakpoint debugging, stepping w/ blackboxing
  - Source maps for transpiled code
  - LiveEdit: JavaScript hot-swap evaluation w/ V8
  - Console evaluation with ES6 feature/object support and custom object formatting
  - Sampling JavaScript profiler w/ flamechart
  - Heap snapshot inspection, heap allocation timeline, allocation profiling
  - Asynchronous stacks for native promises

Demo – node --inspect

node --inspect index.js

node --inspect --debug-brk index.js

node-inspector

- For Node <v6.3.0, Node Inspector supports almost all of the debugging features of the Blink DevTools, including extra features:
  - Set breakpoints in files that are not loaded into V8 yet - useful for debugging module loading/initialization.
  - Embeddable in other applications

Demo: Node-Inspector

```
npm install -g node-inspector
node-debug app.js
```

Debugging Node Apps – Visual Studio Code

- Code supports all the typical debugging features including:
  - Break on exceptions
  - Remote debugging

- With **Node-ChakraCore** and Code -> **Time Travel Debugging**

Back to the Future!

Image from Flickr

Demo: Time Travel Debugging with Code

node-v8.0.0-nightly20170330a35216f4b2-win-x64\node --record .\bin\www

Node-ChakraCore is on GitHub.
Demo Instructions on GitHub. Great Tutorial.

“I hate using the Edge, Chrome & Electron DevTools” or “I need a better TypeScript IDE for Angular 4 than Chrome DevTools”

Using the Chrome Debugging Protocol with Code

- Debugging multiple instances of Chrome / Chromium / Electron / Edge can be painful.
- Debugging Angular4 / TypeScript, CoffeeScript & SASS is best with a fully-featured IDE.

- Using the Chrome Debugging Protocol, you can remote debug or attach to a Chrome browser with a VS Code Extension.
- Equivalent for Edge using the Chrome Debugging Protocol is coming soon.

Demo: Debugging with CDP (Chrome & Edge)

DON'T FORGET TO KILL ALL CHROME PROCESSES FIRST!

Demo: Compound Debugging with Code

Not All Debugging Happens During Development...

Occasionally...

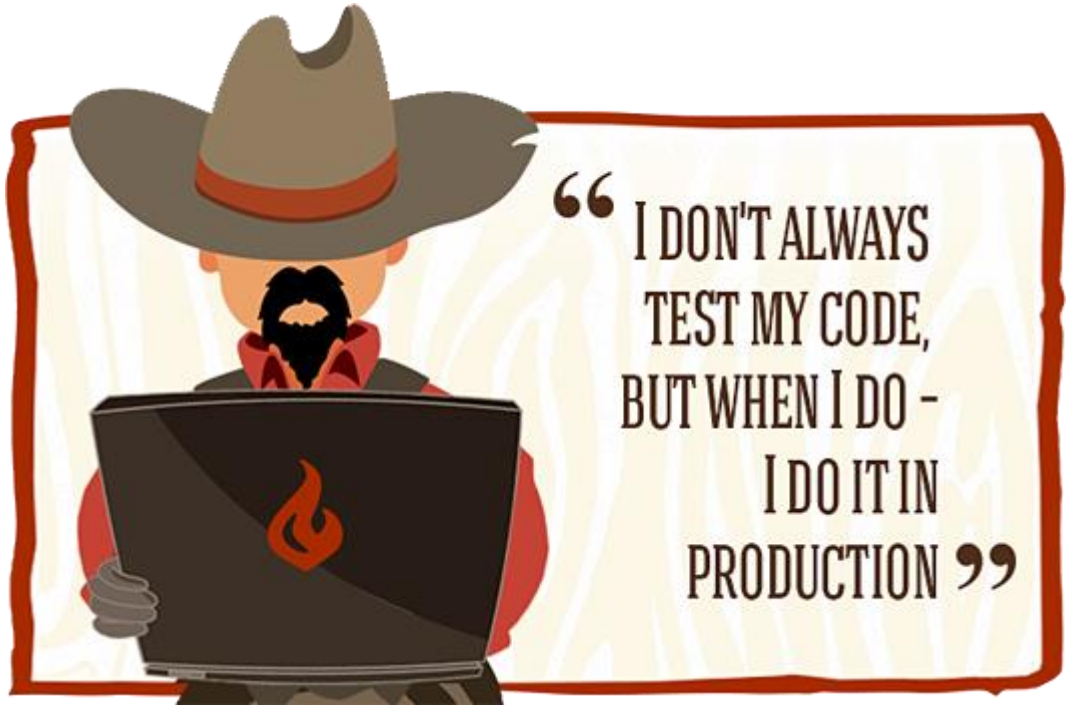


Image from https://www.siteground.com/blog/siteground-staging/

Remote Debugging Node Applications

Enabling Debugging on a Node Process

Pass --debug flag to node process

```
$ node --debug app.js
```

Or on OSX/Linux...

```
$ pgrep -l node
    2345 node your/node/server.js
$ kill -s USR1 <PID>
```

Connecting to a Remote Node Process

- With built-in node debugger:

```
$ node debug localhost:5858
```

- With node-inspector:

```
$ node-inspector
```

- With Visual Studio Code
  - Use Attach launch setting, debug… ☺

Demo: Debugging Remote Node Process

“your website doesn’t work on my pc”

Who hasn’t heard that one or a variant before?

Debugging JavaScript in Remote Browsers

Wait... what! You can do that?

@RAMISAYAR


VORLON.JS
@RAMISAYAR

Vorlon.JS

- An open source, extensible, platform-agnostic tool for remotely debugging and testing your JavaScript.
- Plugins:
  - Console: View logs and errors for your application.
  - Modernizr: View a list of supported and unsupported features.
  - DOM Explorer: Inspect the DOM tree and its corresponding styles.
  - Object Explorer: Display the living JavaScript variables tree.
  - XHR Panel: View XHR calls information sent by your devices.
  - ngInspector: Inspect your Angular.js scopes
  - Network Monitor: View network activities (XHR & resources loading).
  - Resources Explorer: Inspect local resources such as localStorage or cookoes.

Getting Started with Vorlon.JS

```
$ npm install -g vorlon
$ vorlon
```

Add to your code.

```
<script src="http://localhost:1337/vorlon.js"></script>
```

Open http://localhost:1337

Using Vorlon.JS to Remote Debug

- Deploy Vorlon.JS to a public server/PaaS/wtv.
  - As simple as a git push

- Add this to your public beta website:

```
<script src="http://mywebsite.com/vorlon.js"></script>
```

Demo: Vorlon.js

Note: Don’t do this in production with SSL and authentication setup. Better yet, only do this in staging/beta.


FIXED BUG IN CODE
WITHOUT CREATING NEW BUGS

What did we learn?

- Types of Errors & Locations in JavaScript
- Debugging and Introspection Tools
  - node-inspector
  - VS Code
- Remote Debugging Node Processes
- Remote Debugging Front-End JavaScript
  - Vorlon.JS

@RAMISAYAR

Thank You! Questions?

tw: @ramisayar | gh: @sayar

slideshare.net/ramisayar

github.com/sayar/jsdebuggingtalk

Resources, References, Links

- http://www.toptal.com/javascript/10-most-common-javascript-mistakes
- https://developers.google.com/web/tools/javascript/breakpoints/
- http://www.w3schools.com/js/js_debugging.asp
- http://www.webreference.com/programming/javascript/rg31/index.html
- http://eloquentjavascript.net/08_error.html
- http://www.pluralsight.com/courses/fixing-common-javascript-bugs
- https://trackjs.com/
- http://www.standardista.com/javascript/15-common-javascript-gotchas/

Microsoft