Introduction to Aurelia

MODERN WEB APPLICATION DEVELOPMENT

Modern Web Development

Cliff notes… for HTML

- **HTML 4.01 (December 1999)**
  - `<button>`, **not** `<input type="button"></input>`
- **HTML 5 (October 2014)**
  - New Semantic Elements
    - `<nav>, <article>, <aside>, <section>, <header>, <footer>…`
  - Global Attributes, ex: `data-custom`
    - NOT case sensitive
    - Do NOT use special characters (like parens or *)
    - If a period is used, the left portion is still parsed
    - ex. **`value.bind`**
- **Web Components** (Working Draft, Feb 2017)
  - Shadow Dom v1 (encapsulation)
  - HTML Imports, `<link rel="import" href="…" />`
  - HTML Templates, `<template></template>`

Modern Web Development

Cliff notes… for JavaScript

- **ES2015 (June 2015)**
  - Template Literals: `` `Welcome, ${name}` ``
  - Classes: `class Epic {}`
    - `function` keyword not needed inside class
  - New Variable declarations: `let, const`
  - `import` – so far, only implemented by node
- **ES2016 (June 2016)**
  - Decorators: @autoinject
- **ES2017** (Working Draft, Jan/Feb 2017)
  - `await`, `async`

aurelia

Features

- Modular Architecture
- **Dependency Injection**
- **Two-Way Data Binding**
- *Binding Behaviors*
- Value Converters
- **UI Composition** (templates)
- **Routing / Navigation**
- Task Queues
- Pub-Sub Messaging
- **HTML Templating**
- **Custom Elements**
- Logging

aurelia-cli

Scaffolding, Building, Testing

- Installed globally with npm
- `au new` // scaffold app!
  - supports multiple tools:
    - LESS, SASS, TypeScript, BABEL
  - Uses Gulp 4.0 under the hood
- `au build` // build app!
  - Uses RequireJS or SystemJS today
  - Webpack version in development
- `au generate` <item> // Generate common items
- `au test` // Test all the things!

aurelia-cli

Project Anatomy

- aurelia_project
  - Config, tasks, build, etc.
- index.html => host html

- /src
  - app[.ts | .html] => router, view host
  - main.ts => config loaded
  - resources => common stuff

- /scripts => output of build

Bootstrapping

index + main class

index.html

```
<html>
  <head></head>
  <body aurelia-app="main">
    <script
      src="scripts/vendor-bundle.js"
      data-main="aurelia-bootstrapper">
    </script>
  </body>
</html>
```

main class

```
import { Aurelia } from 'aurelia-framework';

export async function configure(aurelia: Aurelia) {
  aurelia.use
    .standardConfiguration();

  await aurelia.start().then(a => a.setRoot());
}
```

Bootstrapping & Routing

app class

```
import { Router, RouterConfiguration } from 'aurelia-router';

export class App {
  router: Router;

  configureRouter(config: RouterConfiguration, router : Router): void {
    config.options.pushState = true;
    config.map([
      { route: 'welcome', name: 'welcome', moduleId: 'welcome' }
    ]);
    this.router = router;
  }
}
```

Bootstrapping

app template with <router-view> container

```
<template>
  <nav>…</nav>
  <div>
    <router-view></router-view>
  </div>
</template>
```

Components, aka “Custom Elements”

Our Hello Page

Class

```
export class Welcome {
  name: string = 'Aurelia';
  hobbies: string[] = [
    'Travel', 'Fine Dining', 'Aurelia'
  ];

  greet(): void {
    alert(`Welcome to ${this.name}`);
  }
}
```

HTML

```
<template>
  <input id="name" type="text" value.bind="name" />
  <h3>Hobbies</h3>
  <ul>
    <li repeat.for='hobby of hobbies'>${hobby}</li>
  </ul>
  <button click.trigger='greet'>OK</button>
</template>
```

Binding

A Closer Look

Simple, One-Way and Two-Way

```
<!-- Simple Binding -->
${name}

<!-- One-time -->
<input type="text" value.one-time="name" />

<!-- One-way -->
<input type="text" value.one-way="name" />

<!-- Two-way -->
<input type="text" value.bind="name" />
```

Looping, events, conditionals

```
<!-- Looping -->
<li repeat.for='hobby of hobbies'>${hobby}</li>

<!-- Trigger events -->
<button click.delegate='greet'>OK</button>
<button click.trigger='greet'>OK</button>

<!-- Conditionals -->
<div if.bind='name.length > 0'>${name}</div>
<div show.bind='isWelcomed'>${name}</div>
```

Component Usage

Local & Global Requires

Local Component

```
<!-- Component with class -->
<require from="./resources/elements/grid">
</require>
<grid>…</grid>

<!-- Component with HTML only -->
<require from="./grid.html"></require>
<grid />
```

Global Component

```
// Add ref in main
aurelia.use.feature('resources');

// Uncomment line #4 in resources/index.ts
config.globalResources([ <component list> ]);

<!- Then use *anywhere* without require -->
<grid />
```

Services

Simple JavaScript Class

Reusable by default (aka Singleton)

```
import { autoinject } from 'aurelia-framework';
import { HttpClient }
  from 'aurelia-fetch-client';

@autoinject
export class Data {

  constructor(private http: HttpClient) {
  }

}
```

New Instance Per Request

```
import { transient }
  from 'aurelia-framework';

@transient()
export class UserModel {
  public name: string;

  constructor(data) {
    this.name = data.name;
  }

}
```

Documentation and Samples

- Aurelia Doc Hub
  - http://aurelia.io/hub
- This Presentation ☺
  - https://github.com/SteveHartzog/thinkful-intro-to-aurelia
- Other good samples
  - https://github.com/aurelia/skeleton-navigation

Aurelia Fundamentals
Brian Noyes, @BrianNoyes (Virginia)
https://www.pluralsight.com/courses/aurelia-fundamentals

Building Applications with Aurelia
Scott Allen, @OdeToCode (Maryland)
https://www.pluralsight.com/courses/building-applications-aurelia

Aurelia Blog
http://blog.aurelia.io/

Blue Spire Enterprise Support
http://aurelia.io/support.html, support@bluespire.com

Thank you.

shartzog@federalidentity.com
@SteveHartzog
#dctech / @SteveHartzog
https://github.com/SteveHartzog

VS Code Plugin – Team Essentials
https://marketplace.visualstudio.com/items?itemName=SteveHartzog.team-essentials