

Javascript/React Crash Course

Roadmap

- Javascript Basic
- NodeJS
- Webpack
- Transpilation
- React
- React-Redux

Javascript Basic

- Interpreted
- Functions are values
- Weak-typing
- 'const'
- Globals, scope and closures
- 'let' vs. 'var'

NodeJS

- Combining javascript files together is dangerous because of overlapping function/variable names.
 - Even if you use 'const', you still have to worry about overlapping names.
- Module system for javascript files:
 - All functions/variables should be 'private'.
 - It should be possible to publish 'public' elements that can be used by other modules via name-spaces.
- Node Basic
 - Run javascript by itself (no browser).
 - A simple module.
- NPM
 - Repository management system for NodeJS.
 - Rating modules: <https://www.npmjs.com/>

NodeJS

- Module Magic

```
(function (exports, require, module, __filename, __dirname)
{
    // YOUR CODE INJECTED HERE!
});
```

NodeJS

- Combining javascript files together is dangerous because of overlapping function/variable names.
 - Even if you use 'const', you still have to worry about overlapping names.
- Module system for javascript files:
 - All functions/variables should be 'private'.
 - It should be possible to publish 'public' elements that can be used by other modules via name-spaces.
- Node Basic
 - Run javascript by itself (no browser).
 - A simple module.
- NPM
 - Repository management system for NodeJS.
 - Rating modules: <https://www.npmjs.com/>

webpack

- NodeJS allows you to create large javascript programs from multiple modules.
- Now, make it work for the browser:
 - Combine all modules into a single file.
 - Make it work without the NodeJS runtime.

webpack server

- Webpack knows how to create a bundle from a starting anchor.
- So, it can create an 'in-memory' bundle. It can also 'watch' the source files and re-create the 'in-memory' bundle.
- Proxy support

Using the latest javascript/css

- Lots of advancements being made to javascript, css.
- Problem – browser makers are slow to adopt changes.
- Solution
 - Convert 'newer' code in javascript/css files to 'older' versions that work in browsers.
 - Since webpack knows what files go into the bundle, perhaps it can be instructed to perform these 'conversions'.
- 'devtool' and source maps.

React Basics

- Javascript library
- Components mapping to HTML elements
- Sits between application code and DOM
 - Minimize DOM operations
- Lots of code for specifying elements.
 - JSX and transpilation.

React Components

- Top-level Application Component
- Component Basics
 - Package
 - Functional Components (simplest)
 - Properties
- React HTML components
 - Map to javascript DOM elements. Fixes some browser quirks. Also increases performance by minimizing DOM operations.
 - All the html attributes
 - style (javascript object vs string)
 - Direct access to DOM via 'refs' – try to avoid them.

React Components (2)

- Components are often 'wrapped'
- Classes vs Functional components
 - Lifecycle methods
 - state
 - Use Classes when necessary.

Managing State

- You could put state in components
 - Problem: components have lifecycles while state does not.
- What you want
 - A global model that acts like a cataloged repository.
 - State should exist as progressive data structures where $T3 = T2 + \text{minor change}$.
 - State changes should be made by Actions.
 - Reducers should modify the model in response to Actions.
 - The model should be immutable (except for reducers).
- Redux + ImmutableJS

Managing State

- Redux and React integration goals
 - Components to 'watch' a portion of the store and redraw when it changes.
 - Components should have an easy way to trigger actions.
- react-redux
 - Connecting a React application with a store.
 - Connecting a component with the store
 - Augmenting component properties
 - `mapStateToProps`
 - `mapDispatchToProps`
 - Functional Components can now have state
 - Use React classes and 'setState' when you have a 'private' setting that isn't part of your data model (i.e. whether or not a combo-list is displayed).
 - All other state should go through Redux
 - Gotchas
 - `mapStateToProps` and `toJS()`
- Redux Tools

More Component Concepts

- propTypes, defaultProps
- Dynamic list of sub-components
 - 'key' property

Style

- Basics
 - Style sheets
 - Using classes as concept anchors
 - Minimal specification
- Modular Styles
 - Similar goals to module javascript
 - CSS class are private to the component.
 - Exports used with name-spaces.
 - Production
 - Eliminate <style> tags by placing the contents in a single file.

Problems with CSS

- Browsers don't update often
- Lots of complaints
 - Common definitions
 - Macros
- Transpilation to the rescue
 - SCSS