Tooling For React

npm

- Node Package Manager
  - even though they say it isn't an acronym
- Each project/library is described by a **`package.json`** file
  - lists all dependencies (runtime and development)
  - can define scripts to be run using "**`npm run`**" command
- To generate **`package.json`**
  - **`npm init`**
  - answer questions
- To install a package globally
  - **`npm install -g`** ***name***
- To install a package locally and add dependency to **`package.json`**
  - for runtime dependencies, **`npm install --save`** ***name*** — or **`npm i -S`** ***name***
  - for development dependencies, **`npm install --save-dev`** ***name*** — or **`npm i -D`** ***name***

To find outdated dependencies, **`npm outdated`**

`package.json` Scripts

- Defined by **`scripts`** property object value
  - keys are script names
  - values are strings of shell commands to run

with some care, it's possible to write scripts that are compatible with both *nix and Windows

binaries of locally installed modules (in **`node_modules/.bin`**) are available

- Manually add script tasks
  - to do things like
    start a server,
    run a linter,
    run tests, or
    delete generated files
- To run a script, **`npm run`** ***`name`***
  - can omit **`run`** keyword for special script names

    `npm test` can be shorted to **`npm t`**
- To see a list of available scripts,
  `npm run`
- See example ahead

Special Script Names

- **`prepare`**, **`publish`**, **`postpublish`**
- **`preinstall`**, **`install`**, **`postinstall`**
- **`preuninstall`**, **`uninstall`**, **`postuninstall`**
- **`preversion`**, **`version`**, **`postversion`**
- **`pretest`**, **`test`**, **`posttest`**
- **`prestart`**, **`start`**, **`poststart`**
- **`prestop`**, **`stop`**, **`poststop`**
- **`prerestart`**, **`restart`**, **`postrestart`**

React

- Install React with `npm install --save react react-dom`
  - `react-dom` is used when render target is web browsers
- Can use `browser.js` to compile React code in browser at runtime, but not intended for production use
- Let’s start serious and use **webpack**

webpack

- Module bundler
  - combines all JavaScript files starting from “entry” by following imports
  - can also bundle CSS files referenced through imports
- Tool automation
  - through loaders
  - ex. ESLint, Babel, Sass, ...
- **`npm install --save-dev webpack`**
- https://webpack.github.io

webpack-dev-server

- HTTP server for development environment
- Provides watch and hot reloading
- Bundles are generated in memory
  and served from memory for performance
- **`npm install --save-dev webpack-dev-server`**
  - see command to start in **`package.json`** ahead

don't need to install globally because it
will be started using "**`npm start`**" which
searches **`.bin`** directories below **`node_modules`**

- If another server must be used
  - for example, when REST services are
    implemented in Node.js and served from Express or
    implemented in Java and served from Tomcat
  - use **`webpack --watch`** and **webpack-livereload-plugin**
  - start from an npm script with
    `"start": "webpack --watch"`
  - see https://github.com/statianzo/webpack-livereload-plugin

Babel

- "Transforms your JavaScript"
  - transforms ES6 code to ES5
  - "can convert JSX syntax and strip out Flow type annotations"
- To use from command line
  - **`npm install -g babel-cli`**
- To use from webpack
  - **`npm install --save-dev babel-core`**
    - transpiles ES6 code to ES5
  - **`npm install --save-dev babel-loader`**
    - allows webpack to run Babel on JavaScript files
- https://babeljs.io/

ESLint

- "Pluggable linting utility for JavaScript and JSX"
  - configure via a `.eslintrc` file
- To use from command line
  - **`npm install -g eslint`**
- To use from webpack
  - **`npm install --save-dev eslint eslint-plugin-react`**
    - lints JavaScript files that use React
    - requires configuration in `.eslintrc`
  - **`npm install --save-dev eslint-loader`**
    - allows webpack to lint JavaScript files using ESLint
    - requires configuration in `.eslintrc`
  - **`npm install --save-dev babel-eslint`**
    - runs ESLint using Babel as the JavaScript parser
- http://eslint.org/

see React and JSX rules described at
https://github.com/yannickcr/eslint-plugin-react

because it understands
newer JavaScript features

Snippet from .eslintrc

```
  ...
  "parser": "babel-eslint",

  "parserOptions": {
    "ecmaVersion": 6,
    "sourceType": "module",
    "ecmaFeatures": {
      "jsx": true
    }
  },

  "plugins": [
    "react"
  ],
  ...
```

webpack.config.js

- Create **`webpack.config.js`**
  - **`entry`** is main JavaScript file that imports others
  - use eslint-loader to check for issues in JavaScript files
  - use babel-loader to transpile ES6 code to ES5
  - use css-loader to resolve URL references in CSS files
  - use style-loader to
    "add CSS to the DOM by injecting a <style> tag"
- To generate **`bundle.js`** file
  - run **`webpack`** for non-minimized
  - run **`webpack -p`** for minimized (production)

gift-redux example adds use of **Bootstrap** and **Sass** to its **`webpack.config.js`**

```
module.exports = {
  entry: './src/main.js',
  output: {
    path: __dirname,
    filename: 'build/bundle.js'
  },
  module: {
    loaders: [
      {
        test: /\.js$/,
        exclude: /node_modules/,
        loader: 'babel!eslint'
      },
      {
        test: /\.css$/,
        exclude: /node_modules/,
        loader: 'style!css'
      },
    ]
  }
};
```

loaders are run in the reverse order in which they are listed

webpack.config.js

"Loading CSS requires the css-loader and the style-loader. They have two different jobs. The css-loader will go through the CSS file and find url() expressions and resolve them. The style-loader will insert the raw css into a style tag on your page."

package.json

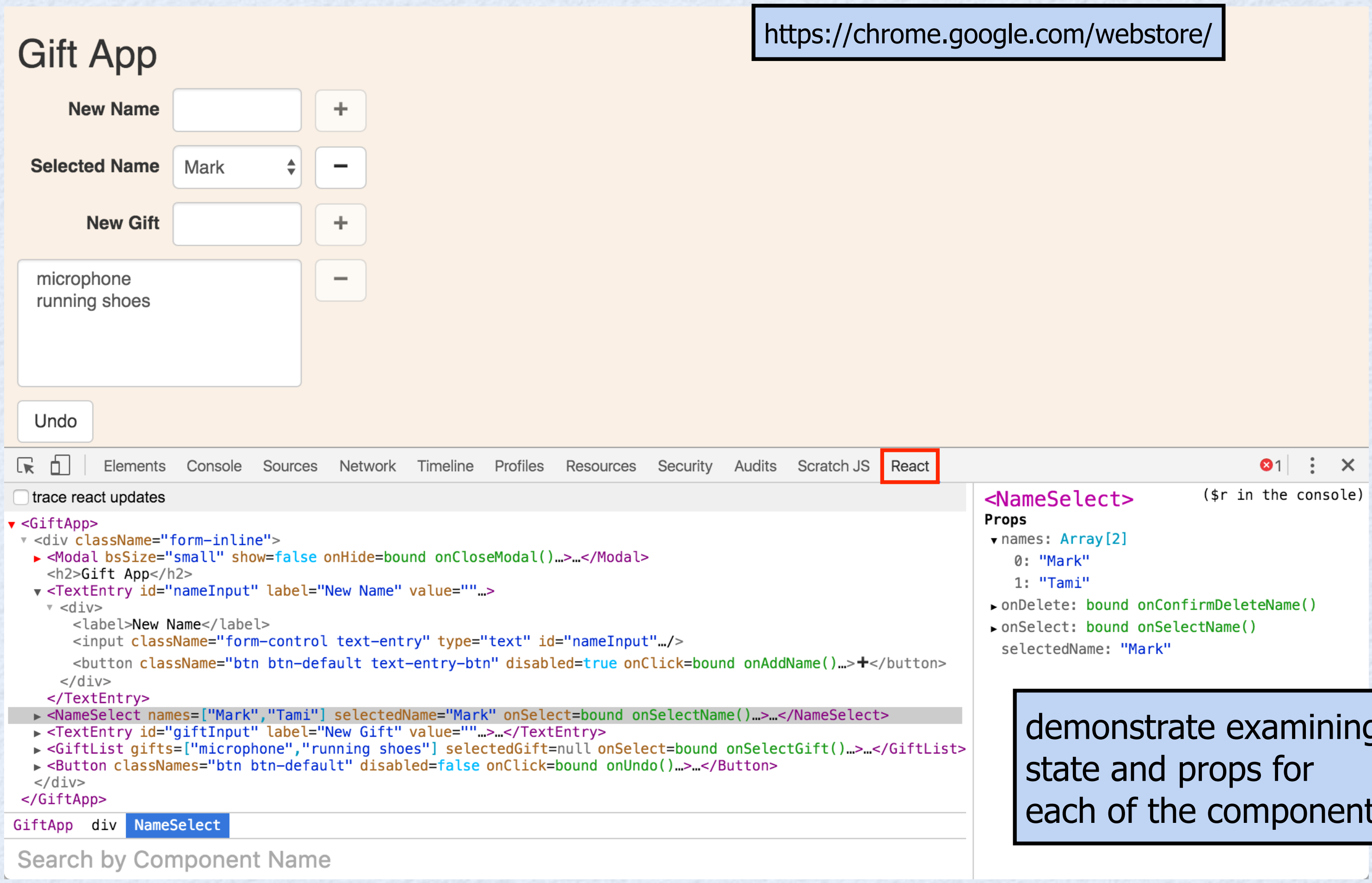
```
{
  "name": "my-project-name",
  "version": "1.0.0",
  "description": "my project description",
  "scripts": {
    "start": "webpack-dev-server --content-base . --inline"
  },
  "author": "my name",
  "license": "my license",
  "devDependencies": {
    "babel-core": "^6",
    "babel-eslint": "^5",
    "babel-loader": "^6",
    "babel-preset-es2015": "^6",
    "babel-preset-react": "^6",
    "css-loader": "^0",
    "eslint": "^2",
    "eslint-loader": "^1",
    "eslint-plugin-react": "^5",
    "style-loader": "^0",
    "webpack": "^1",
    "webpack-dev-server": "^1"
  },
  "dependencies": {
    "react": "^15",
    "react-dom": "^15"
  }
}
```

to start server and watch process, enter "**`npm start`**"

gift-redux example adds use of **Bootstrap**, **Sass** , **Immutable**, and **Expect** to its **`package.json`**

React Developer Tools ...

- See http://facebook.github.io/react/blog/2015/09/02/new-react-developer-tools.html
- Browser extension for Chrome and Firefox (can install from Chrome Web Store)



https://chrome.google.com/webstore/

```
▾ <GiftApp>
  ▾ <div className="form-inline">
    ▸ <Modal bsSize="small" show=false onHide=bound onCloseModal()…>…</Modal>
      <h2>Gift App</h2>
    ▾ <TextEntry id="nameInput" label="New Name" value=""…>
      ▾ <div>
          <label>New Name</label>
          <input className="form-control text-entry" type="text" id="nameInput"…/>
          <button className="btn btn-default text-entry-btn" disabled=true onClick=bound onAddName()…>+</button>
        </div>
      </TextEntry>
    ▸ <NameSelect names=["Mark","Tami"] selectedName="Mark" onSelect=bound onSelectName()…>…</NameSelect>
    ▸ <TextEntry id="giftInput" label="New Gift" value=""…>…</TextEntry>
    ▸ <GiftList gifts=["microphone","running shoes"] selectedGift=null onSelect=bound onSelectGift()…>…</GiftList>
    ▸ <Button classNames="btn btn-default" disabled=false onClick=bound onUndo()…>…</Button>
    </div>
  </GiftApp>
GiftApp  div  NameSelect
```

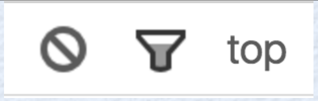
```
<NameSelect>                ($r in the console)
Props
▾ names: Array[2]
    0: "Mark"
    1: "Tami"
▸ onDelete: bound onConfirmDeleteName()
▸ onSelect: bound onSelectName()
  selectedName: "Mark"
```

demonstrate examining state and props for each of the components

... React Developer Tools

- Features
  - adds "React" tab to browser dev tools
  - displays JSX of component tree
    - shows current prop values inline in JSX
    - can expand and collapse components
  - displays props and state of selected component on right side
  - displays ancestors of selected component at bottom
  - can search for a component by name at bottom
  - hover over a component in JSX to highlight in UI
    - if scrolled out of view, right click in JSX and select "Scroll to Node"
  - selected component is available in console as **`$r`**
    - can run **`$r.setState({`*`key`*`: `*`value`*`})`** to update state and UI
  - right-click a prop or state value on right and
    select "Store as global variable" to make available in console as **`$tmp`**
  - and more

top of console must have "top" selected like this:



Redux Developer Tools

- “DevTools for **Redux** with **actions history, undo, and replay**”
- Code change required to use
  - pass additional parameters to `createStore` function
- https://github.com/zalmoxisus/redux-devtools-extension
- https://egghead.io/lessons/javascript-getting-started-with-redux-dev-tools

create-react-app

- Tool that creates a great starting point for new React apps
- Installs and configures many tools and libraries
  - Babel, ESLint, Immutable.js, lodash, React, react-dom,
    webpack (including webpack-dev-server, html-webpack-plugin, css-loader, and style-loader),
    whatwg-fetch, and more
- Provides watch and live reload
- Steps to use
  - **`npm init react-app`** ***my-app-name*** creates and populates directory; installs all dependencies
  - **`cd`** ***my-app-name***
  - **`npm start`** starts local server and loads app in default browser
- Configuration is in node_modules/react-scripts
  - see "scripts" property near bottom of **`package.json`**
- For more information, see https://github.com/facebook/create-react-app