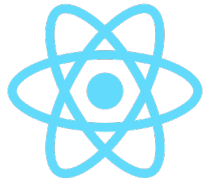
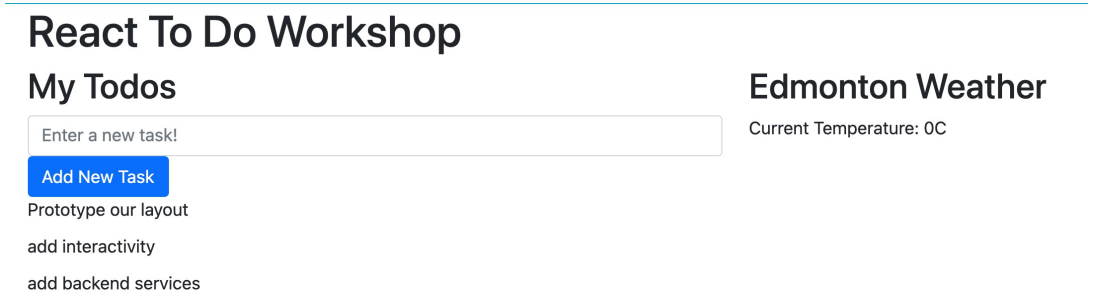
Web Applications with React

HackED 2022 | Intuit | Daniel Zhou

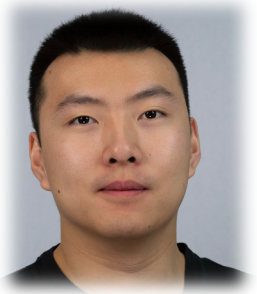
Takeaways

- React?
  - What is it
  - Why should I use it?
- Setting Up A Web Dev Environment
- A To-Do App w/ Modern React Programming
  - Functional React Components
  - State Management via React Hooks
  - Backend Integration via React Hooks
  - Functional Programming





Overview / About Me



UAlberta Alumni 2017 & 2021

Work @ intuit

QuickBooks Canada (Full Stack Dev.)

Intuit Globalization Platform (Backend Dev.)

Javascript, Java, Python, React, GraphQL, MongoDB, SQL, AWS, Splunk

Tech. Interests

Information Systems, Data Analytics, Data Visualization, Static Analysis Tooling

Overview / React Companies

intuit

Overview / About React

What Is React

- Front End Web App
- Declarative
- Component Based
- FOSS

Why React? [1]

- Most popular frontend web library – for good reasons.
- Many Employers Use It

Repository
github.com/facebook/react

Homepage
reactjs.org/

Weekly Downloads
11,907,703

Version	License
17.0.2	MIT

Setup

Install the development environment and the git repo.

Setup / The React Environment

Web Browsers

- Downloads JavaScript, HTML, CSS and renders interactive pages.
- React Is Just A Big JavaScript File
- Your web app. is a bundle of the three.

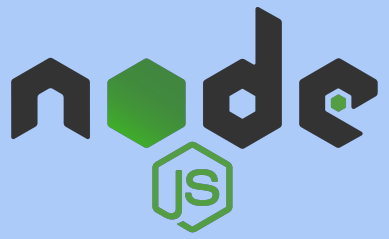
Building The Bundle

- NodeJS – Running JavaScript outside the browser.
- `create-react-app` – A quick start package for building your web. app bundle.

Detail: Node & `create-react-app` handle a lot of configurations and development management for us. This'll generate a lot of helpful scripts and services that'll let us host our web app locally for the time being.

Setup / Install Node

Installers For Each OS:
https://nodejs.org/en/



Commands we'll use:

- `node` – Like `python` or `python3`
- `yarn` – A package manager like pip

Npm & yarn will help us install packages from the internet automatically. They do the same thing in slightly different ways. I prefer yarn so we'll use one to install the other.

```
// Install Node - Linux
sudo apt install nodejs

// Install Node - Mac w/ Brew
brew update
brew install node

// Verify Node Has Been Installed
node --version

// Add Yarn
npm install --global yarn
```

Setup / Clone The Repo

```
// Clone the repo
git clone https://github.com/shouyang/HackED2022-React-Workshop.git react-workshop

// Enter the cloned repo
cd react-workshop

// Install dependencies
yarn

// Run the app!
yarn start
```

The repo was made via `create-react-app`.

Into To React!

Building a Single-Page Application With Functional Components

Project / Overview

The Project:

Build A React Component Based Single Page App

Track A List Of To-do Tasks

View Existing Tasks

Add / Delete Tasks

Fetch The Weather, And Display It ~ If we have time ~

Project / The Plan

Prototype your App Out.

- Iteration 1 - Make placeholder components to layout the UI.
- Iteration 2 – Makeover: style with Bootstrap-React components.

Introduce Interactivity

- Iteration 1 - Add business logic for event handling with hooks.
- Iteration 2 - Integrate with backend RESTFUL API to fetch the weather. (If there's time)

React / Components

Functional components are just functions that follow a convention….

They take a props parameter following a convention:

- props.* - Any parameters passed into the component.
- props.children – A specific parameter denoting nested components.

And they return JSX (HTML + React Components)

- Basically, they return recursively templated HTML.

React / Components – The User View

```
// Basics of React Components
// React Components In Use

<Component PropA= ... Prop= ... >
  Children Props
</Component>

// Example
<List color="blue" title={myListInJavaScript.title}>
  <Item> Hello </Item>
  <Item> World </Item>
</List>

// curly brackets denote a javascript section in the
template
```

React / Components – The Implementation

```
// Basics of React Components
// React Components Implementation

// Take Props, Return JSX in round-brackets
function List(props) {
  //  ... Regular Javascript Code Here  ...
  return (
    <div>
        <title color={props.color}>{props.title}</title>
        {props.children}
    </div>
  )
}

function Item(props) {
  return (<div>{props.children}</div>)
}
```

React / Components – The Rendered View

```
// Basics of React Components
// The rendered HTML

<div>
    <title color="blue">*some title*</title>
    <div>Hello</div>
    <div>World</div>
</div>
```

Live / Prototype Our App Layout

The Folder Structure

```
workshop/
  // Files
  package.json // Node's configuration for this repo.
  yarn.lock // Yarn autogenerates this from package.json for reasons.

  // Directories
  node_modules/ # Node stores dependencies here

  public/       # Our static content, the non-React HTML,CSS,JS being hosted.
  src/          # Our source code.
    index.js    # A really simple script to load our react app.
    App.js      # The root component for our App.

    finished/   # A finished version of the todo App.
    components/ # A folder to store sub-App components, we will reuse some of this.
```

React / The Entire process

```
// ***************************
// My React Code

import SubComponentA from "components/A";
import SubComponentB from "components/B";

function myApp() {
  return (
    <Container>
      <SubComponentA></SubComponentA>
      <SubComponentB></SubComponentB>
      // ....
    </Container>
  );
}
```

```
// ***************************
// index.js

import ReactDOM from 'react-dom'

ReactDOM.render(<myApp></myApp>, "root")
```

```
// ***************************
// index.html

<html>
  <head>
    // Load React
    <script src=" ... react ... " />
    // Load your index.js and app bundle
    <script src="...." />
  </head>
  <body>
    <div id="root">
      Placeholder!
      React will replace this with our app.
    </div>
    ...
  </body>
</html>
```

Library / Component Libraries

React has many popular UI Component libraries!

These'll provide us high quality UI elements like Buttons, Forms, Modals, etc …

And lets us focus on building the App logic components.

The Bootstrap UI Framework

A Popular HTML, CSS, JS Framework …

… React-Bootstrap – Someone made Bootstrap into React Components.

Live / Refine Our App Layout

React / Hooks

React Hooks provide stateful behaviors to functions.

They use a lot of functional programming concepts under the hood.

We're going to touch on two!

`useState`

Let's us use a stateful variable inside our functional component.

`useEffect`

Let's us call functions that'll update this component and re-render it.

… where and when to fetch data and to avoid recursively rendering updates.

React also has class-based components but functional ones are the modern style.
Functional components are simpler to write and test .

React / Hooks - Code

```
import React, {useState} from 'react'

// useState
function MyComponent() {

    // Initialize useState
    const initialCount = 0;
    const [count, setCount] = useState(initialState)
    // Even though this is ran each time ...
    //  ... React will manage the state for us behind the scenes

    // Nested Helper Functions - Wrap setState in helper functions to introduce interactivity
    function incrementCount() {setCount(count + 1)}
        // This uses a JS feature: closures ...
        //  ... these nested functions can reference stuff in MyComponent above it.

    return (<div>Count: {count}</div>)
}
```

React / Hooks - Code

```
import React, {useState} from 'react'

// useEffect
function MyComponent() {

    // Initialize useState to store our data
    const initialData = 'Loading';
    const [data, setData] = useState(initialData)

    // Nested Network Request Function
    async function getData() {
      const data = await fetch('...')
      setData(data);
    }

    // Case 1: Triggers once on load
    useEffect(getData, [])
    // or Case 2: Triggers everytime A changes.
    useEffect(get, [A])

    return (... do something with {data})
}
```

Live / Add Interactions & Integrations

Resources

Learning React

React Office Tutorial

W3 Schools

- JavaScript
- HTML
- CSS

46b6ed9e2209248c56327b99510e53c9