* create a folder, and put public folder and img under public
* set up Webpack and Webpack-web-server

"devDependencies": {

"babel-core": "^6.26.0",

"babel-loader": "^6.2.4",

"babel-polyfill": "^6.9.1",

"babel-preset-es2015": "^6.9.0",

"babel-preset-react": "^6.11.1",

"webpack": "^1.13.1",

"webpack-dev-server": "^1.14.1"

}

* create webpack.config.js

module.exports = {

devServer: {

// inline: true,

contentBase: './public',

port:3000

}

}

* modify the start script of package.json

"scripts": {

"start": "webpack-dev-server"

}

* npm start
* Download bootstrap css and create a css folder and place it inside

<link rel="stylesheet" href="/css/bootstrap.css">

<div class="jumbotron">

<h1 class="display-3">React Training</h1>

<p class="lead">Learning React is lots of fun</p>

<hr class="my-4">

<p>In this training you will learn about dev development with React</p>

<p class="lead">

<a class="btn btn-primary btn-lg" href="#" role="button">Learn more</a>

</p>

</div>

* create src folder and create client.js and put a console.log(“Hello”)
* Update the config file before the devServer

entry: './src/client.js',

output:{

path: './',

filename:'bundle.js'

},

* Open http://localhost:3000/bundle.js
* Update index.html. last before line add <script src="bundle.js"></script>
* Uncomment inline: true so that the changes are watched
* npm install --save-dev babel-core babel-loader babel-preset-es2015 babel-preset-react babel-polyfill
* update package.js

"babel": {

"presets": ["es2015","react"]

}

* update webpack.config.js

module: {

loaders: [{

test: /\.js$/,

excludes:/node\_modules/,

loader: 'babel'

}]

* Update the entry

entry: ['babel-polyfill','./src/client.js']

* npm start
* install react and react-dom. npm install react react-dom –save
* Create a simple component and add it in the client.js

import React from 'react';

import ReactDOM from 'react-dom';

import App from './app';

const component = <div className="jumbotron">

<h1 className="display-3">Hello, world!</h1>

<p className="lead">I am learning React training in 2 days</p>

<hr className="my-4"></hr>

<p>I will learn the basics of ES6, DevOps, Flux, Redux , test and build and kick start my React Skills</p>

<p className="lead">

<a className="btn btn-primary btn-lg" href="#" role="button">Learn more</a>

</p>

</div>

ReactDOM.render(component, document.getElementById('root'));

Place only the component and show the <http://localhost:3000/bundle.js> in the browser. Then add it in the render method of the ReactDOM. Update the root in the index.html

* Move to App component

import React from 'react';

**export default** class App extends React.Component{

render(){

return <div className="jumbotron">

<h1 className="display-3">Hello, world!</h1>

<p className="lead">I am learning React training in 2 days</p>

<hr className="my-4"></hr>

<p>I will learn the basics of ES6, DevOps, Flux, Redux , test and build and kick start my React Skills</p>

<p className="lead">

<a className="btn btn-primary btn-lg" href="#" role="button">Learn more</a>

</p>

</div>

;

}

}

ReactDOM.render(<App />, document.getElementById('root'));

Change the render method to include <App /> and talk about capital letter

Talk about Stateless and Stateful component. What we saw above is a stateful component. The whole lot of lifecycle method is available.

* Define a stateless or a function component

const Componen = ()=>{

return(<div><ul><li>Test</li></ul></div>);

}

ReactDOM.render(<Componen />, document.getElementById('root'));

* Add var name above the class , and show in the tag with { }

var name="React JS"

{name}

* Use this when the variable is an instance variable

constructor(){

super();

this.desc = "Description";

}

Use {this.desc} in the jsx template

* Define methods as well in the class

display(){

return "This is a cool text"

}

{this.display()}

* Go to slide for JSX
* Show single root.
* Display blue and green button based on condition
* Add this.rank=15 in the constructor
* Type in the render() function

var dynaStyle="btn btn-primary btn-lg"

if (this.rank>10)

dynaStyle = "btn btn-success btn-lg"

change the <a className to {dynaStyle}

Button is green. Change the rank to 10 to see the blue button

**Loops**

Add the lines in constructor

this.names = ["John Smith","Micheal Jordan"];

Show without key={e} and see the error

{this.names.map((e)=><p **key={e}**>{e}</p>)}

\_\_\_\_\_\_\_\_ **Break** \_\_\_\_\_\_\_\_\_\_\_\_

Move name, desc, rank as props

ReactDOM.render(<App desc="Description" rank="15" names={["John Smith","Micheal Jordan"]} />

Props is available in the parameter in functional components

const Component = (props)=>{

return(<div><ul><li>{props.name}Test</li></ul></div>);

}

<Component name="MyComponent"></Component>

* State

this.state={"text":"Hello"};

{this.state.text}

EVENTS (to update learn events )

* Simple click handler in the same example
* Write a method above render function

handleClick(){

this.setState({

text: "Hey hi there"

});

}

<a className={dynaStyle} href="#" role="button" onClick={this.handleClick.bind(this)}>