***CREATE A NEW PROJECT (create-react-app)***

This is a project from Facebook, that helps to create ReactJS project from scratch.

***npx create-react-app <future-Project-name>***

***i.e: npx create-react-app portfolio***

***Start and test a project***

Use the following command, from the root of a ReactJS project

(where the package.json and other files are)

***npm run start***

***Debug React with Chrome***

This link describe how to install a plugin in VS Code and debug ReactJS application when using the Chrome browser (*Tested on Sept 28,2020 and worked*)

<https://code.visualstudio.com/docs/nodejs/reactjs-tutorial>

***ReactJS States***

Ways to have Global variables/methods, and have the application react to it.

***Note***: States are only possible in class components (not in function components)

***Steps:***

1 - Declare the variables in the class constructor as a JSON pair (state is part of the class object)

2 - (Important) Declare also getter/setter methods who will interface with global variable (in the constructor)

constructor(){

super();

// Declaring a state here named 'detailsVisible'

this.state = { detailsVisible: false };

// Declare a getter/setter (a toggler here) that will inteface with 'detailsVisible'

this.toggleDetails = this.toggleDetails.bind(this);

}

3 - User the ***setState*** method on the state

Should not reassign the value of a state directly (like a variable i.e: detailsVisible = false)

Use a getter/setting (declared in step #2)

// Exemple of a setter method

showDetails(){

this.setState( { detailsVisible: true } );

}

Check uDemy ReactJS , chap 12 & 13

Babel 🡪 Cross browser compiler

***Simple Component (function component)***

This simple one is not inheriting from the react libraries Component, and is lightweight

Example here is Person component, that is using the props for parameters and children

import React  from 'react';

const person = (props) => {

    return (

      <div>

         <p>This is a test, name is {props.name} and the age is {props.age}</p>

         <p>{props.children}</p>

       </div>

    )

};

export default person;

   <Person name='Daniel' age='49'/>

        <Person name='Jocelyne' age='58'> And this is at the end</Person>

***Class Component***

This type of component can have a state (main local memory)

import React  from 'react';

class TopMenuBanner extends React.Component {

  state = {

        currentSelectedUnit : FAHRENHEIT,

        count : 2

    };

    render(){

        return (

            <div className="TopMenuBanner">

          </div>

        );

    }

}

export default TopMenuBanner;

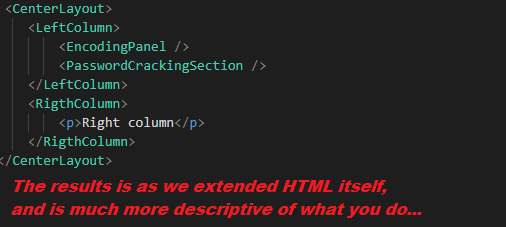
***Inline Component***

Instead of using <div> in ReactJS pages, create inline components makes code much more readable.

It uses the ***props.children*** to render the code



And the use of the inline Component will be like this:



Make sure that you define the right CSS classes for your inline Components.

***Package ReactJS***

Once done, you can run the following command to package the project:

npm run build

The script will create a ***‘build’*** folder that contains all the files to be uploaded to a web server.

***Events Handling***

You can pass an event handler to a children component. The Child component will access the function passed via the props

***Main Class (Mother Component)***

toggleErrorState = () => {

const newVal = !this.state.hasError;

this.setState( { hasError: newVal });

console.log( 'Error State changed to ' + newVal );

}

render(){

return (

<div className="App">

<MainActionButton title='Decode' clickHandler={ () => this.toggleEncodeMode() } />

</div>

}

***Children Class (Sub Component)***

function MainActionButton(props) {

  let className = 'MainActionButton';

  div className=’MainActionButton’ onClick={ props.clickHandler }>

   </div>

  );

}

***Props to Class Components***

Passing and using props from a function component is easy. But if you have 2 class components who needs to pass info (Mother and Son), then you need to use a special syntax.

The children component class can access its props, via the ***this*** operator

class App extends Component {

render() {

const greeting = 'Welcome to React';

return (<div> <Greeting *greeting*={greeting} /> </div>);

}

}

class Greeting extends Component {

render() {

return <h1>{this.props.greeting}</h1>;

}

}

Web site: <https://www.robinwieruch.de/react-pass-props-to-component>

}

***Visual Studio Code and ReactJS snippet***

VS Code as very good tools, for ReactJS, and one of them is ***ReactJS Code Snippets***

***JavaScript***

Template Litterals (the ` ` string)

Example: `string text ${expression} string text`

Use this ES6 synthax when dealing with string… A good page [here](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)