Wednesday, January 11, 2023 11:41 AM

React = Powerful CLIENTSIDE JAVASCRIPT LIBRARY !!

Prerequisites = NODE should be installed. Check node -v

Create a React project Template =
Step1 = Open CMD
Step 2 = cd to the folder where you want to create all your react projects
Step3 = run the command with internet on !!
npx create-react-app firstapp

React is a **component based** library !!
Multiple components are created ,
Final output = Integration of all the components
Advantages of component based architecture ---
1. Component is an independent entity , can be separately by a team
2. Components can be easily replaced by other versions
3. Time to develop is reduced
4. We can download readymade components
5. Unit testing of components is easy

Challenges of component based architecture --- FOR Integration
1. Communication between the components MUST be properly defined
2. Component MUST follow some specification

The React project Template - Firstapp
-open the Firstapp folder in VSCode editor
- Observe the folder structure-
  - node_modules = folder indicates that this is a node type of project
    - This has libraries
  - public folder = holds HTML (index.html)
    `<div id="root"></div>`
    All the efforts are made to add content to this div using DOM manipulation
  - src folder = folder containing JS files and also css and images
    - Index.js = this is the starting point of the REACT application
      We RENDER the **react component named App** in the tag in index.html
    - App.js = First component that is rendered .
  - package.json = this file indicated that the current folder is a node project
    - It holds the list of dependencies of the project
    - It holds the npm commands that can be executed for the project
      - Start = used to start the development server
      - Build = used to create the production build
  - Package-lock.json = For VERSION control of the dependent libraries

Run the Development Server and see the output on the browser ----------
Step 1 = open the terminal on VSCODE
Step 2 = cd to the firsapp project (or any other project we want to test)
Step 3 = npm start

It starts the development server at localhost : 3000
It will deploy the firstapp project by default
We keep the server running as we make modifications in the project
--- it hot deploys and shows the output after changes

Modify the App Component ---

Step 1 - remove all the imports
- remove all the code in return
Step 2 - write h1 tag in the return , save and observe in the browser

React has two types of components ----

1. Function based component = the component is a function
the function MUST **return** renderable output (html)

2. Class based component = the component is a class
the class must have a render() method that returns some output that can be rendered in html

Ex1 - Let us add a component(AddComponent) that displays a text field and a button

CREATE THE COMPONENT ------------

Step1 - add a JS file in src folder
Step2 - add a function
Step3 - export the function component
Step4 - return the div component holding textfied and button
**** use () after return to include multiline code
**** we can return only one element
**** if not div use empty tag fragment to enclose textfield and Button

INTEGRATE THE COMPONENT with index.html…….

Step1 - add the <AddComponent /> tag in the return of App.js
Step2 - import the component in the App.js
Step3 - save and check in browser

import , export , aliasing -------

export = tell node that the current component can be used outside the file
import = tell node that use the implementation that is outside the current file

Aliasing = give some other name to the imported component

Export can be done in two ways -

export default	import AddComponent from "location"	Dont use curly brackets while importing

			One JS file can have at the max 1 export default
1.	export	import {AddComponent } from "location"	Use curly brackets while importing One JS file can have many exports
	We can write export and export default on separate lines at the end of the file OR We can write export or export default on the same lines where function is declared		

EX - Add Two Components SayHi and SayBye in the AddComponent.js
we will export default AddComponent
Simple export SayHi and SayBye
Integrate in App.js and see output on browser

Modify the import such that SayHi is aliased as Hi and
AddComponent is aliased as AC

Aliasing	
Default exported component	import ALIASNAME from "location"
Simple exported component	import {SayHI **as** Hi } from "location"

Class Component in React ----------

Ex --- Write a class component to Show a name and the address in a div

Step1 ---- create a Show.js
Step2 --- write a class inherit it from Component class from "react " library
Step3 --- export the class
Step4 ---- write a render() in the class that returns the values to be rendered
Step5 --- integrate with App() and see in the browser

JSX -------

JSX = Java Script Extension
= Reacts understanding of html
= The code looks similar to html but it isn't html
= JSX compiler is used to translate JSX to HTML which will be rendered by browser
the JSX compiler is called as Babel (see it in node_modules)

Ex1 - Write a function component TestJSX
Observations
1. If we forget to write return in the component = no error

2. Type of v = object = JSX code is an **object** in React
3. We see the JSX object key value pairs
4. The console output is printed twice ---to print once (remove <StrictMode> from index.js)

PRINT variables in JSX ----------

{var} = interpolation symbol
{expression } = the expression or variable or function call can be written within {}
{func() }

The content in {} should evaluate to **a single value**

Ex --------- add variables in JSX of TestJSX2 component . Integrate it with TestJSX

PROPS in React ----------

PROPS = Properties , custom attributes that can be added to the React component tags
= they are a way to communicate between parent component(ENCLOSING component tag) and the child component (ENCLOSED component tag)

Ex - Write a function component Welcome that gets the name of the person to be welcomed from the props !!! Integrate it with App.js and see on browser

HW
1. TRY OUT EVERYTHING DONE IN CLASS
2. Write a function component MathsComponent
It gets 3 props num1 , num2 and operator
Print the calculation as per the props values
Integrate with App.js