Ep3: Laying the foundation

npx parcel index.html

→ npm is executing parcel and index.html is passed as an input.

We can write script to directly launch the dev build or prod build.

And this is present in package.json

```
"scripts": {
    "start": "parcel index.html",
    "build": "parcel build index.html",
    "test": "jest"
  }
```

Now we can run by

```
npm start
npm run start
npm run build
```

```
// this how we can run script.
//npm start work because start is a keyword reserve by NPM
```

//React Element

React Elements are equivalent to DOM element.

React Elements are object when we render onto the DOM then it becomes an HTML elements.

Creating more complex structure becomes very clumsy.

Facebook Dev created JSX.

JSX and react is separate.

JSX is a convention where is kind of merge html and js together.

JSX is not html in JS. JSX is HTML-like or XML syntax.

Is JSX valid JS?

```
const jsxHeading = <h1 id="heading">React using JSX</h1>;
```

Both, the answer is its not pure JS. JS engine will not be able to understand JSX. Then how its working in our case. Its because of Parcel and specifically because of Babel.

This code is transpiled(convert it into code that React or JS engine can understand) before going to JS engine.

React Element conversion

React.createElement ⇒ ReactELement- JS Object ⇒ HTML element(render)

JSX Conversion

JSX ⇒ React.createElement ⇒ ReactELement- JS Object ⇒ HTML element(render)

Babel convert JSX to React Code.

Babel can also transpiled ES6 JS code to older JS code.

CamelCasing is used to add attributes.

In jsx if your code is only of one line then its fine but as soon has you want to move to next line then whole code must be include in circular bracket ( )

```
// React Element
//created using react
const heading = React.createElement("h1",{id:"heading"},"This is H1");
console.log(heading);
//created using JSX
// JSX
const jsxHeading = (<h1 id="heading" className="head">
    React using JSX
    </h1>);
```

React Component

There are two types of components in React:

1. Class Based Component - OLD way of writing code. It uses JS Classes
2. Functional Component - NEW way . It uses Function.

There are still class based component are asked in interviews. We will learn it later:

What is React Functional Component? It just a normal javascript function.

Whenever we create a component name it in Capitalize form so that it can understand its a component.

A function which return some JSX code is functional component.

```
//React Functional  Component
const HeadingComponent = () => {
    return <h1>Hello World React!</h1>
}
//way 2 of writing
const HeadingComponent = () => <h1>Hello World React!</h1>
// way 3 of writing
const HeadingComponent = () => (
     <h1 className="heading1">Hello World Functional Component React!</h1>
)
//normal function also valid
const Head = function(){
return (<h1 className="heading1">Hello World Functional Component React!</h1>)
}
```

JSX is a react element. A JS function which return react element then its a functional component.

How to render components in react?

```
root.render(<HeadingComponent />;
```

How to add component inside component?

```
const Title = ()=> (
    <h1 className="head" tabIndex="5">
        Hi Everyone this is Kaif
    </h1>
);

//React Functional  Component
//Component Composition
const HeadingComponent = () => (
    <div id="container">
    <Title/>
    <h1 className="heading1">Hello World Functional Component React!</h1>
    </div>
)
```

This is also called as Component Composition (Writing one component in another)

How to put an element within a component?

```
const title = (
    <h1 className="head" tabIndex="5">
        Hi Everyone this is Kaif
    </h1>
);
//React Functional  Component
const HeadingComponent = () => (
    <div id="container">

    <h1 className="heading1">Hello World Functional Component React!</h1>
    </div>
)
```

{ } we can write javascript code in this braces.

We can also call the functional component

```
{Title()}
<Title></Title>
<Title />
```

We can add element inside component and vice versa also.

JSX prevents cross site scripting attack also. JSX is powerful

Homework

▼ What is JSX?

JSX is a javascript extension that allows creation of DOM trees using an XML/HTML-like syntax.

▼ Superpowers of JSX?

1. It converts the HTML-like code to React-element which is then converted to object and then with the use of render we display as an HTML on website.
2. JSX prevents cross site scripting attacks.
3. Makes code more readable.
4. JSX is faster compare to Normal JS code
5. JSX also allows React to show more useful error and warning messgaes.

▼ Role of type attribute in script tag?

type attribute indicates the type of script represented.

type=”importmap” → This value indicates that the body of the element contains an import map. The import map is a JSON object that developers can use to control how the browser resolves module specifiers when importing JS modules.

module→ this value causes the code to be treated as a JS module.

speculationrules→ This value indicates that the body of the element contains speculation rules. Speculation rules take form of a JSON object that determine what resources should be prefetched or prerendered by the browser.