Building Rich Front-End Applications with React and ES6

Module 1 Building Rich Front-End Applications with React and ES6

Package/Method | **Description** | **Code Example**

let and const

let allows you to restrict the scope of variables within the block where they are declared. const allows you to declare constants whose values cannot be changed.

```
{
    let a = 10
    console.log(a)
    a = 15
    console.log(a)
}
console.log(a)
const num = 5
console.log(num)
num = 8
console.log(num)
```

Copied!

Arrow function

Arrow functions allow you to write shorter function syntax.

```
hello = () =>
{
    return "Hello World!";
}
```

Copied!

Promises

The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value.

```
let promiseArgument = (resolve, reject) =>
    setTimeout (() => {
    let currTime = new Date().getTime();
    if(currTime % 2 === 0){
    resolve(“Success”)
    }else{
    reject(“Failed!!!”)}
    }
    ,2000)
    }
    let myPromise = new Promise(promiseArgument);

```

Copied!

class

Class is a template or blueprint for creating object.

```
function car(name,year)
{
    this.name = name
    this.year = year
    return this;
}
let car = car("Ford", 2014)
console.log(car)
console.log(car.name)
console.log(car.year)
```

Copied!

Inheritence

A class created with a class inheritance, inherits all the methods from another class.

```
class Square extends Rectangle
{
    constructor(height,width)
    {
        if(height === width)
        {
            super(height,width)
        }
        else
        {
            super(width,width)
        }
    }
}
let mySquare = new Square(5,5)
```

Copied!

React components

Components are reusable segments of code that come under the class and functional component types.

```
import React from 'react';
import {Text} from 'react-native’;
const Helloworld= ()=>
{
    return
    (Hello, World!);
}
export default Helloworld;
```

Copied!

React class Component

React class component contains-
Props: set from outside the class
State: internal to the class

```
import React from "react";
class App extends React.Component {
constructor(props) {
super(props);
this.state={change: true };
}
render() {
return(
<button Click={()=>{this.setState({change: !this.state.change});}}>Click Here!</button>
{this.state.change?(Hello!!):(Welcome to the React Course)}
);}}
export default App;
```

Copied!

onClick

When an event fires, event handlers decide what should happen next. This could involve pressing a button or altering a text entry.

```
function changeColor() {
const change = () => {
    alert("Color Changed!");
}
return (
<button onClick={change}>Change the Color! </button>
);
}
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<changeColor />);
```

Copied!

Skills Network