

# Introduction to React JS

# Agenda

- Prerequisites
- What is React?
- Why React?
- Single Page Applications
- Local Setup
- Create React App
- Next Generation JavaScript
- Basics of React

# Prerequisites

- **HTML**
  - Basic Tags
  - Ex: <div>, <span>, etc
- **CSS**
  - Basic Styling
  - Selectors
- **JavaScript**
  - Objects
  - Functions
  - Control Flow
  - DOM Manipulation
    - **Select and Manipulate, Events etc**

Proprietary Content © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.  
This file is meant for personal use by [konepala.indu@gmail.com](mailto:konepala.indu@gmail.com) only.

Sharing or publishing the contents in part or full is liable for legal action.

# What is React?

- JavaScript library for building user interfaces.
- Library is a pre-written code that is developed already and is readily available.
- Ex: math.js, day.js etc.

# Why React?

- Alternatives of React :- Angular.js.
- More control over the flow of application.
- Reusing the components(optimization).
- Easy to learn.
- Trusted by some leading companies - Facebook, Netflix, PayPal, Tesla etc.
- More job opportunities.
- Mainly used for single page application.

# Single Page Application

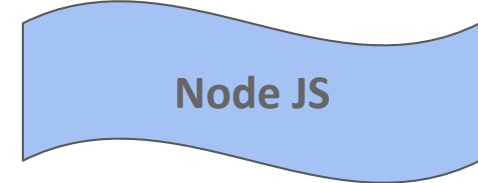
- Single request to server.
- HTML, CSS and JS as response from the server.
- Virtual DOM is created then any further requests are handled by React.
- React only updates that object in actual DOM
- Saves a lot of time, bandwidth, etc.
- Improves performance.
- Some SPA are Gmail, Facebook, Twitter etc.

# Local Setup

## 1. Download and Install NodeJS

Npm or Yarn (Helps to manage JS packages)

Development Server



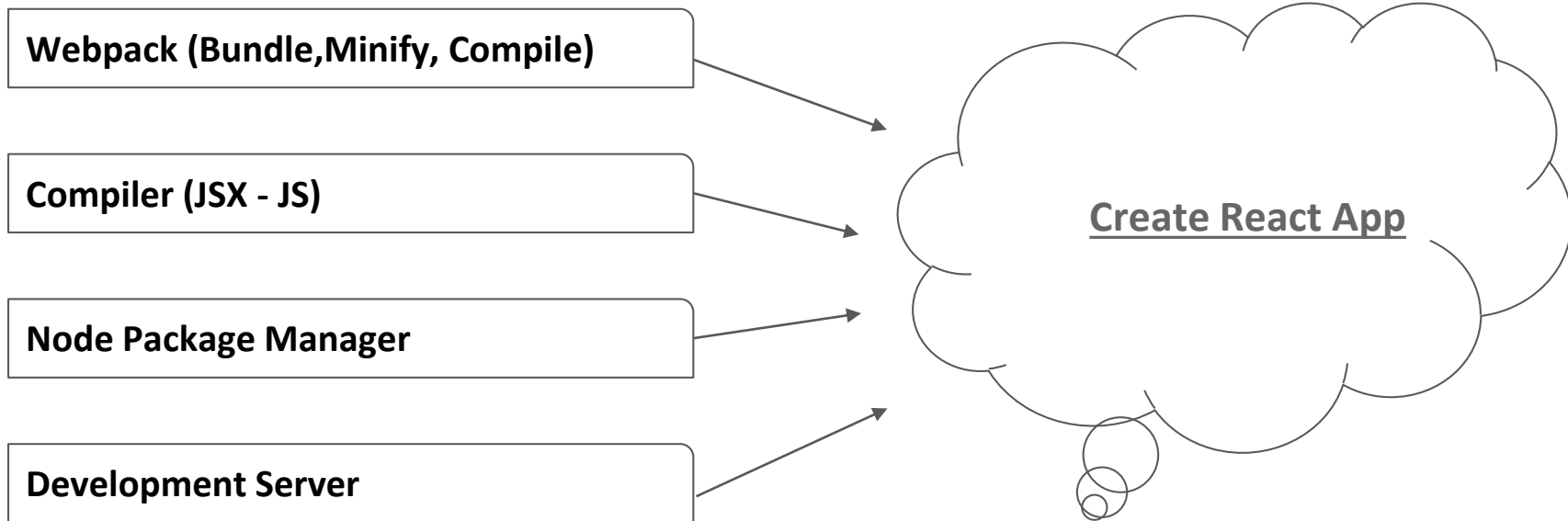
## 2. Download and Install Any Editor

# Create React App

- Create React App is the best practise to create single-page React applications.
- Don't need install or configure tools.
- We don't require Webpack or Babel.
- Built by developers at Facebook.
- Giving a head start thus saving a lot of time.



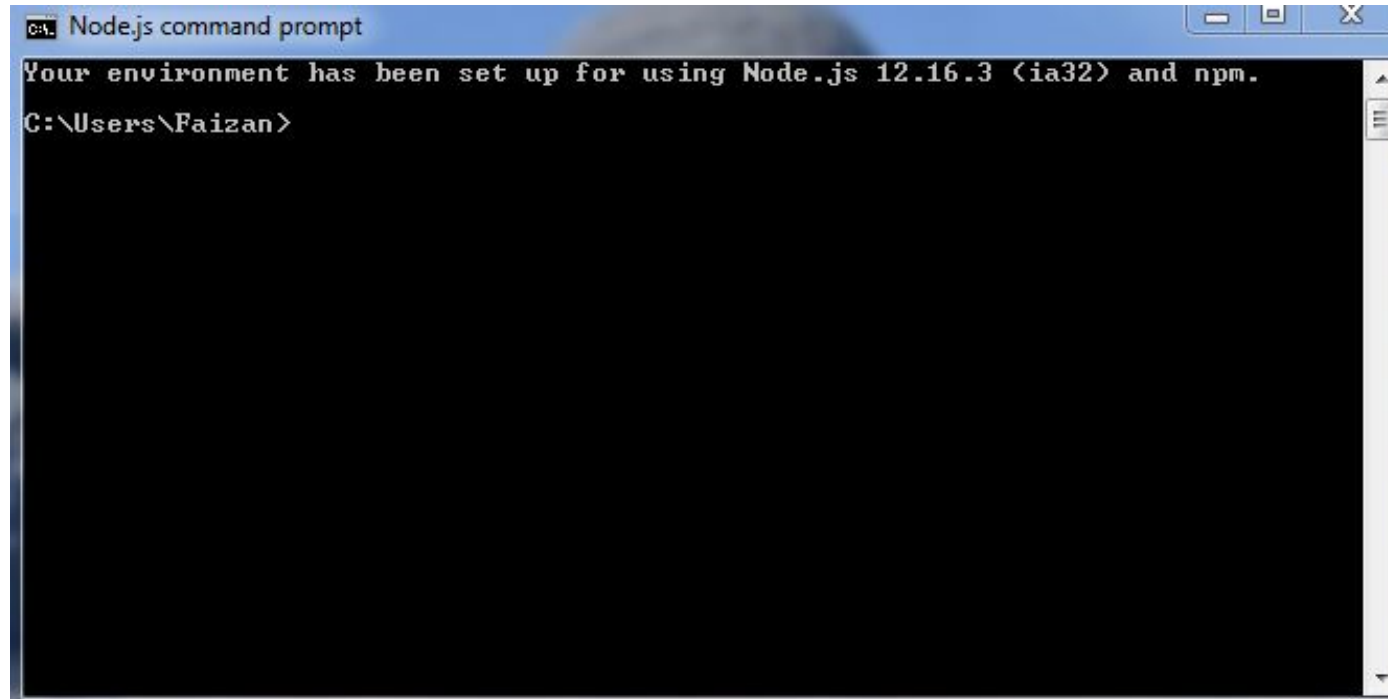
# What if we didn't use this tool?



# Getting Started With Create React App

# Step #1

Open Node.js command prompt.

A screenshot of a Windows command prompt window titled "Node.js command prompt". The window has a blue title bar with standard minimize, maximize, and close buttons. The main area is black with white text. The first line of text reads: "Your environment has been set up for using Node.js 12.16.3 (ia32) and npm." The second line shows the current directory: "C:\Users\Faizan>". There is a vertical scrollbar on the right side of the window.

```
C:\Users\Faizan> Your environment has been set up for using Node.js 12.16.3 (ia32) and npm.
C:\Users\Faizan>
```

Proprietary content ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

This file is meant for personal use by [korepara.indu@gmail.com](mailto:korepara.indu@gmail.com) only.

Sharing or publishing the contents in part or full is liable for legal action.

# Step #2

Run this command on the terminal to install create react app globally.

```
C:\> Node.js command prompt

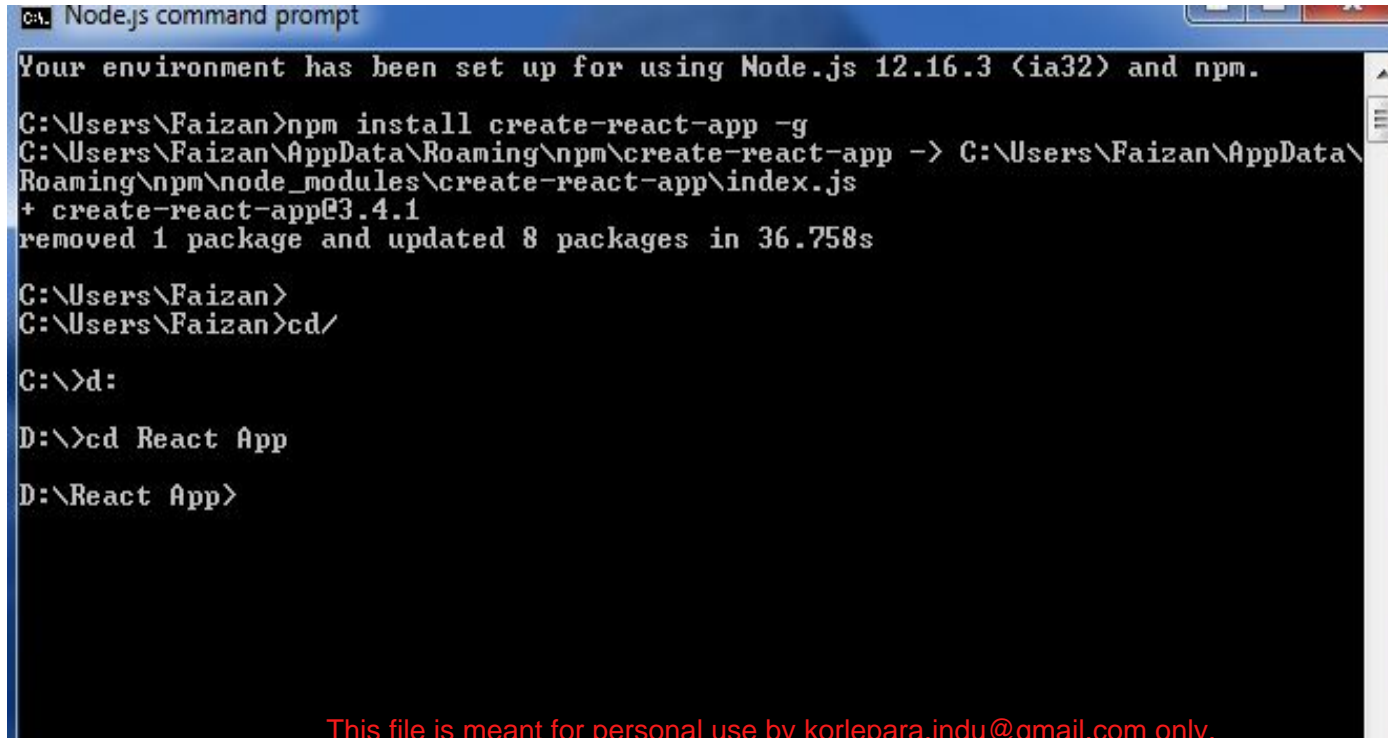
Your environment has been set up for using Node.js 12.16.3 (ia32) and npm.

C:\Users\Faizan>npm install create-react-app -g
C:\Users\Faizan\AppData\Roaming\npm\create-react-app -> C:\Users\Faizan\AppData\Roaming\npm\node_modules\create-react-app\index.js
+ create-react-app@3.4.1
removed 1 package and updated 8 packages in 36.758s

C:\Users\Faizan>
C:\Users\Faizan>
```

# Step #3

Navigate to path where we want to create our app.



```
Node.js command prompt
Your environment has been set up for using Node.js 12.16.3 (ia32) and npm.

C:\Users\Faizan>npm install create-react-app -g
C:\Users\Faizan\AppData\Roaming\npm\create-react-app -> C:\Users\Faizan\AppData\Roaming\npm\node_modules\create-react-app\index.js
+ create-react-app@3.4.1
removed 1 package and updated 8 packages in 36.758s

C:\Users\Faizan>
C:\Users\Faizan>cd/

C:\>d:

D:\>cd React App

D:\React App>
```

# Step #4

Run the create-react-app command along with project name.

```
D:\React App>create-react-app reactproject

Creating a new React app in D:\React App\reactproject.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

> core-js@2.6.11 postinstall D:\React App\reactproject\node_modules\babel-runtime\node_modules\core-js
> node -e "try{require('./postinstall')}catch(e){}"

> core-js@3.6.5 postinstall D:\React App\reactproject\node_modules\core-js
> node -e "try{require('./postinstall')}catch(e){}"

> core-js-pure@3.6.5 postinstall D:\React App\reactproject\node_modules\core-js-pure
> node -e "try{require('./postinstall')}catch(e){}"

+ react-dom@16.13.1
+ react@16.13.1
+ cra-template@1.0.3
+ react-scripts@3.4.1
added 1613 packages from 750 contributors and audited 921730 packages in 477.819s
```

```
58 packages are looking for funding
  run `npm fund` for details

found 1 low severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details

Created git commit.

Success? Created reactproject at D:\React App\reactproject
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd reactproject
  npm start

Happy hacking!
```

# Step #5

Navigate to the project and run npm start command.

```
ca npm
D:\React App>cd reactproject
D:\React App\reactproject>npm start

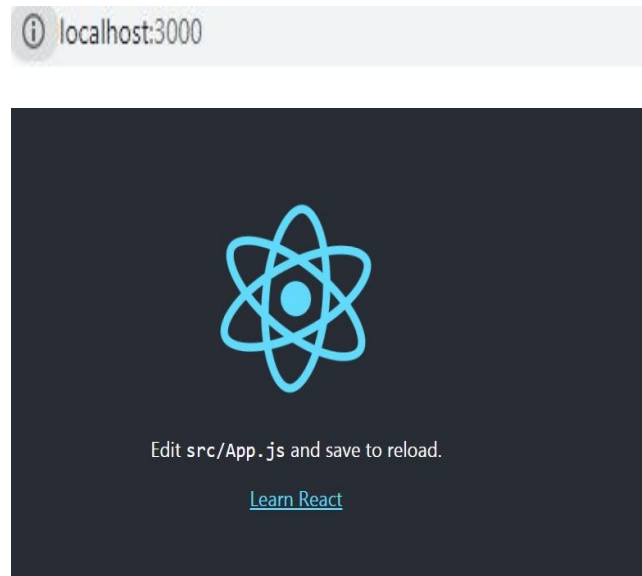
> reactproject@0.1.0 start D:\React App\reactproject
> react-scripts start

Compiled successfully!

You can now view reactproject in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.1.3:3000

Note that the development build is not optimized.
To create a production build, use npm run build.
```



# Next Gen. JavaScript



# Let Keyword

# Block scope of Let

```
console.log("Global Scope");
for(var g_scope=1;g_scope<5;g_scope++)
{
    console.log(g_scope);
}
console.log(g_scope); //Global Scope

console.log("Block Scope");
for(let b_scope=1;b_scope<5;b_scope++)
{
    console.log(b_scope);
}
console.log(b_scope); // Error (Block Scope)
```

Global Scope	<a href="#">practise.js:2</a>
1	<a href="#">practise.js:5</a>
2	<a href="#">practise.js:5</a>
3	<a href="#">practise.js:5</a>
4	<a href="#">practise.js:5</a>
5	<a href="#">practise.js:7</a>
Block Scope	<a href="#">practise.js:10</a>
1	<a href="#">practise.js:13</a>
2	<a href="#">practise.js:13</a>
3	<a href="#">practise.js:13</a>
4	<a href="#">practise.js:13</a>
✖ ▶ Uncaught ReferenceError: b_scope is not defined at <a href="#">practise.js:15</a>	<a href="#">practise.js:15</a>

# Let Declaration

```
for(let declare_again=1;declare_again<5;declare_again++)  
{  
  console.log(declare_again);  
}  
var declare_again=1;  
  console.log(declare_again); // Different block No ERROR.  
  
var declare = 5;  
  console.log(declare);  
  
let declare = 5;                // Same block ERROR.  
  console.log(declare);
```

1	<a href="#">practise.js:5</a>
2	<a href="#">practise.js:5</a>
3	<a href="#">practise.js:5</a>
4	<a href="#">practise.js:5</a>
1	<a href="#">practise.js:8</a>
5	<a href="#">practise.js:11</a>

✖ Uncaught SyntaxError: Identifier 'declare' has already been declared [practise.js:13](#)

# Constant Keyword

# Const Keyword

Const Reference and value though property can be changed.

```
const object={name:"Jimmy"};
```

```
undefined
```

```
object.age=23;
```

```
23
```

```
object.name="Jim";
```

```
"Jim"
```

```
object
```

```
▶ {name: "Jim", age: 23}
```

```
object = "Jimmy";
```

```
▶ Uncaught TypeError: Assignment to constant variable.  
at <anonymous>:1:8
```

```
> const do_not_change = 10;
```

```
< undefined
```

```
> do_not_change = 100;
```

```
▶ Uncaught TypeError: Assignment to constant variable.  
at <anonymous>:1:16
```

# Functions

# Function-Declaration & Expression

**Loads at Compile Time**

```
> function execute(){  
  console.log("Hello");  
}
```

```
< undefined
```

```
> execute()
```

```
Hello
```

```
< undefined
```

**Loads at Runtime**

```
var execute = function name(){  
  console.log("Bye");  
}
```

```
undefined
```

```
execute()
```

```
Bye
```

VM858:2

# Arrow Function

```
> var execute = () => {  
  console.log("Arrow");  
}
```

```
< undefined
```

```
> execute();
```

```
Arrow
```

```
VM1084:2
```



# Export

Export keyword helps us to use one JavaScript file in another JavaScript file.

## With Default:

```
const academy = {  
  name: "GL";  
}  
  
export default academy;
```

## Without Default:

```
export const subject = "React";  
  
export const status = () => {  
  console.log("Completed");  
}
```

# Import

To include the exported file we use “Import” keyword in the destination JS file.

## With Default:

```
import academy from './file_name.js';  
or  
import acad from './file_name.js';
```

## Without Default:

```
import {subject} from './file_name.js';  
import {status} from './file_name.js';
```

# Class

**CLASS(Methods & Properties)**

```
Class c_name{  
  constructor()  
}
```

**Object(Instance of Class)**

```
Const x = new  
c_name();
```

**Inheritance(Inheriting Methods & Properties)**

```
super()
```

# Spread

- Spread operator is used with arrays or objects .
- Spread operator is used to retrieve all values or properties from one array/object into other.
- Represented by three dots(...).

```
const arr =[1,2,3];  
undefined  
const new_arr=[...arr,4];  
undefined  
new_arr  
▶ (4) [1, 2, 3, 4]  
arr  
▶ (3) [1, 2, 3]
```

# Rest

- Rest operator is used with functions.
- Rest operator forms an array of arguments passed to the function.
- Represented as (...).

```
function find_sum(...numbers)
{
    var sum = 0;
    for(let n of numbers){
        sum=sum+n;
    }
    return sum;
}
undefined
find_sum(10,20,30,40);
100
```

# Destructuring

Destructuring is a way with which we can extract few elements from the array or few properties from the object.

```
> var values =[10,20,30];  
< undefined  
> [first,,last]=values;  
< ▶ (3) [10, 20, 30]  
> first  
< 10  
> last  
< 30  
>
```

# Basics of React

# Package.json Folder

Dependencies created by Create-React-App

```
"dependencies": {  
  "@testing-library/jest-dom": "^4.2.4",  
  "@testing-library/react": "^9.5.0",  
  "@testing-library/user-event": "^7.2.1",  
  "react": "^16.13.1",  
  "react-dom": "^16.13.1",  
  "react-scripts": "3.4.1"  
},
```









# Public Folder

- In this folder we will always have single html file (index.html) which runs on the server and we will edit this file.
- All the components are rendered on index file
- In the div with id root and displayed on the browser.

# Source Folder

This is our react application.

All the script files are added in the source folder.

 App.css	10/26/1985 1:45 PM	Cascading Style S...	1 KB
 App.js	10/26/1985 1:45 PM	JScript Script File	1 KB
 App.test.js	10/26/1985 1:45 PM	JScript Script File	1 KB
 index.css	10/26/1985 1:45 PM	Cascading Style S...	1 KB
 index.js	10/26/1985 1:45 PM	JScript Script File	1 KB
 logo.svg	10/26/1985 1:45 PM	SVG Document	3 KB

# Files After Removal

Remove the logo, css and make changes to App.js and index.js file.

```
import React from 'react';
import './App.css';

function App() {
  return (
    <div className="App">
      <h1>Your First React App</h1>
    </div>
  );
}

export default App;
```

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

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

# Functional Component

- To render anything to DOM, we require React file from react library.
- Functional Component is created.

```
import React from 'react';
import './App.css';

function App() {
  return (
    <div className="App">
      <h1>Your First React App</h1>
    </div>
  );
}

export default App;
```

# Class Component

In class based component render method is used to return html element to DOM which can be rendered to the browser .

```
import React from 'react';
import {Component} from 'react';

class App extends Component
{
  render()
  {
    return (
      <div className="App">
        <h1>Your First React App</h1>
      </div>
    );
  }
}

export default App;
```

# JSX

- Everything written lowercase is treated as JSX which looks like HTML but is JavaScript with syntax extension.
- Alternatively you can use `React.createElement(...)` with at least 3 arguments (element,config,text).

# Summary

- Introduction to React
- Single Page Applications like Facebook, Gmail, etc
- Local Setup and steps involved in Create React App
- Next Generation JavaScript ( Let, const, etc)
- Basics of React

# Thank You