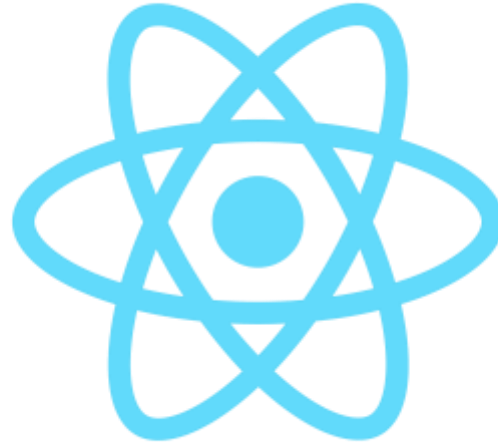


16IT056-React JS

Unit1-Session2-Part1

P RAMPRAKASH,
ASSISTANT PROFESSOR/IT,



React JS –Environmental setup

- **C01** : Understand the basic concepts in React JS and its advantages
- **L01.3: Explain the procedure of environment setup of ReactJS**

Pre-Requisite for ReactJS

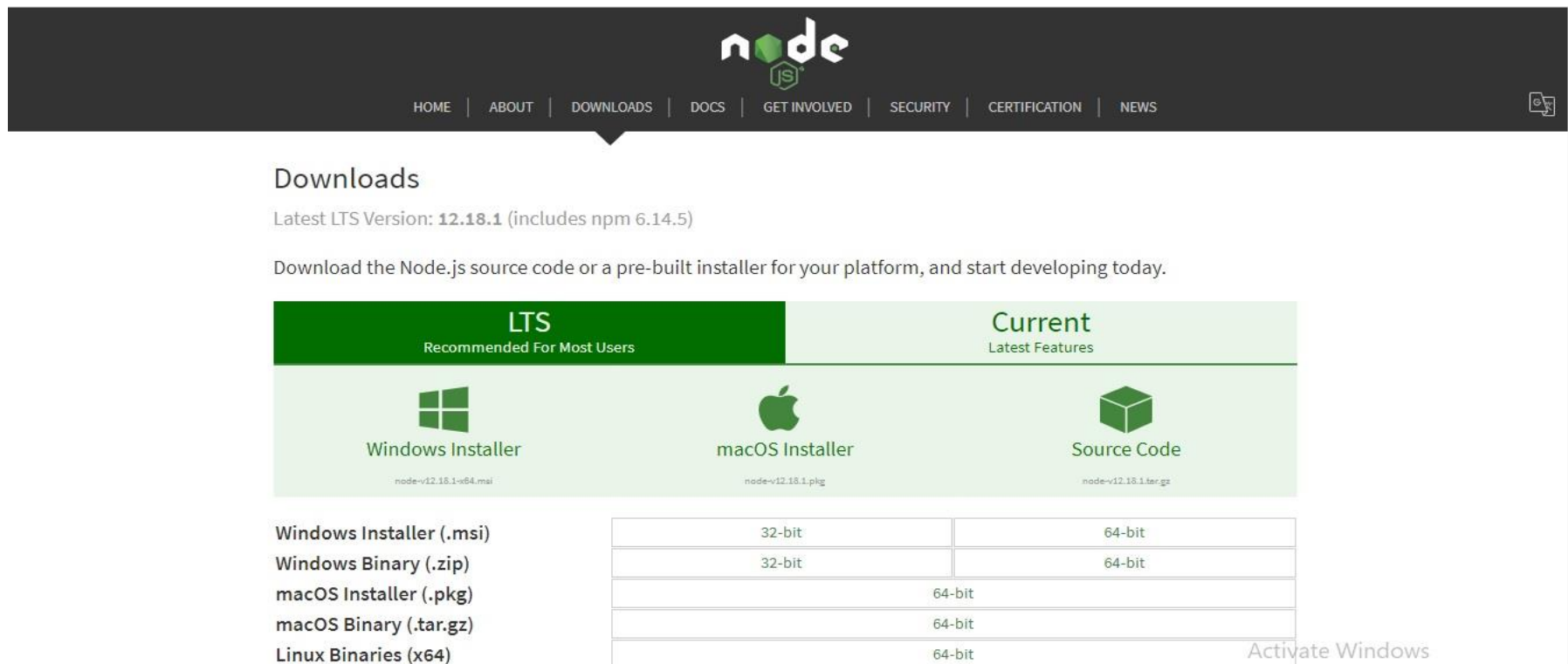
- NodeJS and NPM
- React and React DOM
- Webpack
- Babel

Two ways of installing the ReactJS:

- **Using npm command**
- **By using create-react-app**

By using npm command

- **Step 1:** Install NodeJS. Go to <https://nodejs.org>
- **Step 2:** Click on downloads.



The screenshot shows the Node.js website's 'Downloads' section. At the top, the Node.js logo is centered, with a navigation bar containing links: HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, CERTIFICATION, and NEWS. Below the navigation bar, the 'Downloads' heading is followed by the text 'Latest LTS Version: 12.18.1 (includes npm 6.14.5)'. A paragraph states: 'Download the Node.js source code or a pre-built installer for your platform, and start developing today.' Below this, there are two main tabs: 'LTS Recommended For Most Users' (highlighted in green) and 'Current Latest Features'. Under the 'LTS' tab, there are three options: 'Windows Installer' (with a Windows logo and file name 'node-v12.18.1-x64.msi'), 'macOS Installer' (with an Apple logo and file name 'node-v12.18.1.pkg'), and 'Source Code' (with a cube icon and file name 'node-v12.18.1.tar.gz'). Below the 'Windows Installer' option, a list of download links is provided: 'Windows Installer (.msi)', 'Windows Binary (.zip)', 'macOS Installer (.pkg)', 'macOS Binary (.tar.gz)', and 'Linux Binaries (x64)'. To the right of this list is a table with two columns: '32-bit' and '64-bit'. The table has four rows, with the first two rows containing '32-bit' and '64-bit' respectively, and the next two rows containing '64-bit' and '64-bit' respectively. At the bottom right of the table, there is a link 'Activate Windows'.




node

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Downloads

Latest LTS Version: **12.18.1** (includes npm 6.14.5)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS Recommended For Most Users	Current Latest Features
 Windows Installer <small>node-v12.18.1-x64.msi</small>	 macOS Installer <small>node-v12.18.1.pkg</small>
	 Source Code <small>node-v12.18.1.tar.gz</small>

Windows Installer (.msi)
 Windows Binary (.zip)
 macOS Installer (.pkg)
 macOS Binary (.tar.gz)
 Linux Binaries (x64)

32-bit	64-bit
32-bit	64-bit
	64-bit
	64-bit
	64-bit

Activate Windows

By using npm command (contd..)

- **Step 3:** Install This Node.js
- **Step 4:** Check the version by using the following command.

```
C:\Users\Gigabit>node -v  
v11.13.0
```

- **Step 5:** Now, create a root folder in your desired directory with your desired name.

```
F:\>mkdir reactnpm  
F:\>cd reactnpm  
F:\reactnpm>_
```

By using npm command (contd..)

- **Step 6:** Create a package.json file. This file is required for generating a module. Create this file by using the command:
- ***npm init -y***

```
F:\reactnpm>npm init -y
Wrote to F:\reactnpm\package.json:

{
  "name": "reactnpm",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

F:\reactnpm>
```


By using npm command (contd..)

- **Step 7: Install React and its DOM Packages** by using the npm command:
- ***npm install react react-dom --save***

```
F:\reactnpm>npm install react react-dom --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN reactnpm@1.0.0 No description
npm WARN reactnpm@1.0.0 No repository field.

+ react@16.10.2
+ react-dom@16.10.2
added 8 packages from 3 contributors and audited 22 packages in 4.559s
found 0 vulnerabilities

F:\reactnpm>
```

By using npm command (contd..)

- **Step 8: Install Webpack**
- Webpack is mainly used for module packaging, development, and production pipeline automation.
- For installing Webpack, we have to use the following command:
- ***npm install webpack webpack-dev-server webpack-cli --save***
 - ***webpack-dev-server*** ---> development
 - ***webpack*** - - -> create production builds
 - ***webpack-cli*** - - -> commands.

By using npm command (contd..)

```
F:\reactnpm>npm install webpack webpack-dev-server webpack-cli --save
npm WARN reactnpm@1.0.0 No description
npm WARN reactnpm@1.0.0 No repository field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":
"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ webpack-cli@3.3.9
+ webpack-dev-server@3.8.2
+ webpack@4.41.1
added 559 packages from 358 contributors and audited 8829 packages in 265.179s
found 0 vulnerabilities
```

By using npm command (contd..)

- **Step 9:** Install Babel.
- Babel is simply a JavaScript compiler that is used for creating one source code to others. For installing Babel, use the following command:
- ***npm install babel-core babel-loader babel-preset-env babel-preset-react babel-webpack-plugin --save-dev***

By using npm command (contd..)

```
F:\reactnpm>npm install babel-core babel-loader babel-preset-env babel-preset-react babel-webpack-plugin --save-dev

> core-js@2.6.9 postinstall F:\reactnpm\node_modules\core-js
> node scripts/postinstall || echo "ignore"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)

npm WARN babel-loader@8.0.6 requires a peer of @babel/core@^7.0.0 but none is installed. You must install peer dependencies yourself.
npm WARN reactnpm@1.0.0 No description
npm WARN reactnpm@1.0.0 No repository field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ babel-core@6.26.3
+ babel-loader@8.0.6
+ babel-preset-react@6.24.1
+ babel-webpack-plugin@0.1.1
+ babel-preset-env@1.7.0
added 174 packages from 176 contributors and audited 14008 packages in 75.791s
found 0 vulnerabilities

F:\reactnpm>
```

By using npm command (contd..)

- **Step 9:** Create files. To complete the installation process, we need to add some files to the project folder. The files are:
- **index.html**
- **App.js**
- **main.js**
- **webpack.config.js**
- **.babelrc**
- We can manually create these files, or by using the command prompt. For creating files, use the command prompt, and type the following command:
- **type nul > *filename***

By using npm command (contd..)

- **Step 10: Now, Set Compiler, loader, and server for React Application**
- **Configure Webpack**
- Add the code given below in the **webpack.config.json** to configure the Webpack. It sets the development server to 8080 port. It is responsible for defining the loaders for processing the file types that uses within your app and wrap up by adding plugins needed during our development.

By using npm command (contd..)

```
webpack.config.json
const path = require('path');
const HtmlWebpackPlugin = require('html-
webpack-plugin');
module.exports = {
  entry: './main.js',
  output: {
    path: path.join(__dirname, '/bundle'),
    filename: 'index_bundle.js'
  },
  devServer: {
    inline: true,
    port: 8080
  },
}
```

```
module: {
  rules: [
    {
      test: /\.jsx?$/,
      exclude: /node_modules/,
      use: {
        loader: "babel-loader",
      }
    }
  ],
  plugins: [
    new HtmlWebpackPlugin({
      template: './index.html'
    })
  ]
}
```


By using npm command (contd..)

```
index.js x App.js — reactnpm x webpack.config.js x serviceWorker.js x App.js — react/reactfirst/src x App.test.js x
1  const path = require('path');
2  const HtmlWebpackPlugin = require('html-webpack-plugin');
3
4  module.exports = {
5    entry: './main.js',
6    output: {
7      path: path.join(__dirname, '/bundle'),
8      filename: 'index_bundle.js'
9    },
10   devServer: {
11     inline: true,
12     port: 8080
13   },
14   module: {
15     rules: [
16       {
17         test: /\.jsx?$/,
18         exclude: /node_modules/,
19         use: {
20           loader: "babel-loader", |
21         }
22       }
23     ]
24   },
25   plugins: [
26     new HtmlWebpackPlugin({
27       template: './index.html'
28     })
29   ]
30 }
```

By using npm command (contd..)

- Now, open **package.json** file and delete ***“test”*** ***“echo \” Error: no test specified\” && exit 1”***, which is inside ***“scripts”*** object, and add the commands ***start*** and ***build***.

```
{
  "name": "reactnpm",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"/>

```

Delete this

By using npm command (contd..)

- Delete which is shown in pervious image and add the following two:
- **“start”: “webpack-dev-server –mode development –open –hot”,**
- **“build”: “webpack –mode production”**
- It is described in the following image:

By using npm command (contd..)

```
{
  "name": "reactnpm",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "webpack-dev-server --mode development --open --hot",
    "build": "webpack --mode production"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "react": "^16.10.2",
    "react-dom": "^16.10.2",
    "webpack": "^4.41.1",
    "webpack-cli": "^3.3.9",
    "webpack-dev-server": "^3.8.2"
  },
  "devDependencies": {
    "babel-core": "^6.26.3",
    "babel-loader": "^8.0.6",
    "babel-preset-env": "^1.7.0",
    "babel-preset-react": "^6.24.1",
    "babel-webpack-plugin": "^0.1.1"
  }
}
```

By using npm command (contd..)

- **HTML webpack template for index.html**
- We have to add a custom template to generate index.html using the **HtmlWeb-packPlugin** plugin. It enables us to add a viewport tag for supporting mobile scaling of our app.
- Add the following code to your index.html file.

By using npm command (contd..)

```
<!DOCTYPE html>
<html lang = "en">
  <head>
    <meta charset = "UTF-8">
    <title>React App</title>
  </head>
  <body>
    <div id = "app"></div>
    <script src = 'index_bundle.js'></script>
  </body>
</html>
```

By using npm command (contd..)

- **App.jsx** and **main.js**
- This is the app entry point i.e., the first react component. It will render Hello World.
- Add the following code to your **App.js** file.

```
1  import React, { Component } from 'react';
2  class App extends Component{
3      render(){
4          return(
5              <div>
6                  <h1>Hello World</h1>
7              </div>
8          );
9      }
10 }
11 export default App;
```

By using npm command (contd..)

- **main.js**

```
import React from 'react';  
import ReactDOM from 'react-dom';  
import App from './App.js';  
  
ReactDOM.render(<App />, document.getElementById('app'));
```

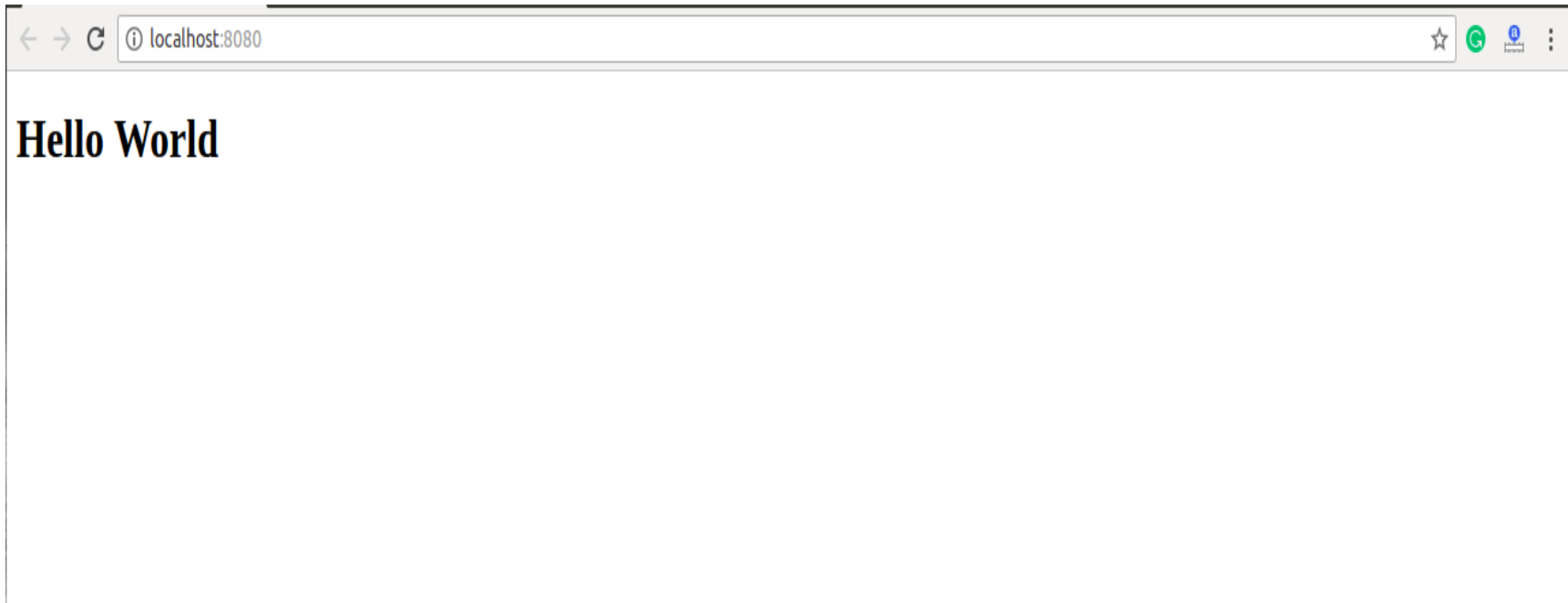

By using npm command (contd..)

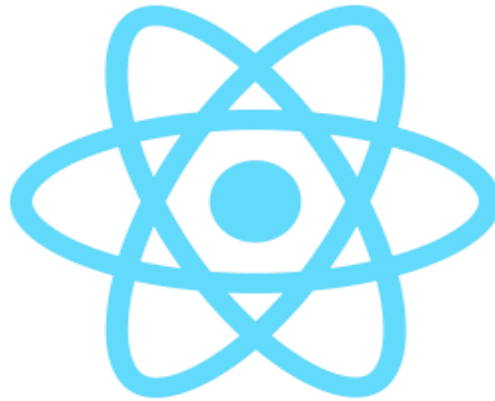
- **Create .babelrc file**
- **Add the following code to your .babelrc file.**

```
2 .babelrc
3 {
4   "presets":[
5     "@babel/preset-env","@babel/preset-
6     react"]
7 }
```

By using npm command (contd..)

- Now, run the server. For running the server, apply the following command in your command prompt within the same folder in which all react files exist.
- ***npm start***





16IT056-React JS

Unit1-Session2-Part2

P RAMPRAKASH,
ASSISTANT PROFESSOR/IT,

using the create-react-app command

- **Step 1:** Install NodeJS. Go to <https://nodejs.org>
- **Step 2:** Click on downloads.

Downloads

Latest LTS Version: **12.18.1** (includes npm 6.14.5)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS Recommended For Most Users	Current Latest Features	
 Windows Installer <small>node-v12.18.1-x64.msi</small>	 macOS Installer <small>node-v12.18.1.pkg</small>	 Source Code <small>node-v12.18.1.tar.gz</small>

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit	
macOS Binary (.tar.gz)	64-bit	
Linux Binaries (x64)	64-bit	

Activate Windows

using the create-react-app command

- **Step 3:** Install This Node.js
- **Step 4:** Check the version by using the following command.

```
C:\Users\Gigabit>node -v  
v11.13.0
```

- **Step 5:** Now **Install NPM** in your selected drive, by using the command: **npm install -g create-react-app**

```
F:\react>npm install -g create-react-app  
C:\Users\Gigabit\AppData\Roaming\npm\create-react-app -> C:\Users\Gigabit\AppData\Roaming\npm\node_modules\create-react-app\index.js  
+ create-react-app@3.2.0  
updated 1 package in 9.901s  
  
F:\react>
```

using the create-react-app command

- **Step 6:** Now create your own React project by using the command:
- **create-react-app** *your project name*
- It will take some time.

```
F:\react>create-react-app reactfirst

Creating a new React app in F:\react\reactfirst.

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

yarn add v1.13.0
[1/4] Resolving packages...
[2/4] Fetching packages...
info There appears to be trouble with your network connection. Retrying...
info fsevents@1.2.9: The platform "win32" is incompatible with this module.
info "fsevents@1.2.9" is an optional dependency and failed compatibility check. Excluding it from installation.
info fsevents@2.0.7: The platform "win32" is incompatible with this module.
info "fsevents@2.0.7" is an optional dependency and failed compatibility check. Excluding it from installation.
[3/4] Linking dependencies...
warning "react-scripts > @types/eslint-plugin > tsutils@3.17.1" has unmet peer dependency "typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta".
[4/4] Building fresh packages...
success Saved lockfile.
warning Your current version of Yarn is out of date. The latest version is "1.19.1", while you're on "1.13.0".
info To upgrade, run the following command:
$ curl --compressed -o- -L https://yarnpkg.com/install.sh | bash
success Saved 7 new dependencies.
info Direct dependencies
├─ react-dom@16.10.2
├─ react-scripts@3.2.0
└─ react@16.10.2
```

using the create-react-app command

```
info All dependencies
├─ react-app-polyfill@1.0.4
├─ react-dev-utils@9.1.0
├─ react-dom@16.10.2
├─ react-error-overlay@6.0.3
├─ react-scripts@3.2.0
├─ react@16.10.2
└─ scheduler@0.16.2
Done in 292.31s.

Success! Created reactfirst at F:\react\reactfirst
Inside that directory, you can run several commands:

  yarn start
    Starts the development server.

  yarn build
    Bundles the app into static files for production.

  yarn test
    Starts the test runner.

  yarn 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 reactfirst
  yarn start

Happy hacking!

F:\react>_
```

using the create-react-app command

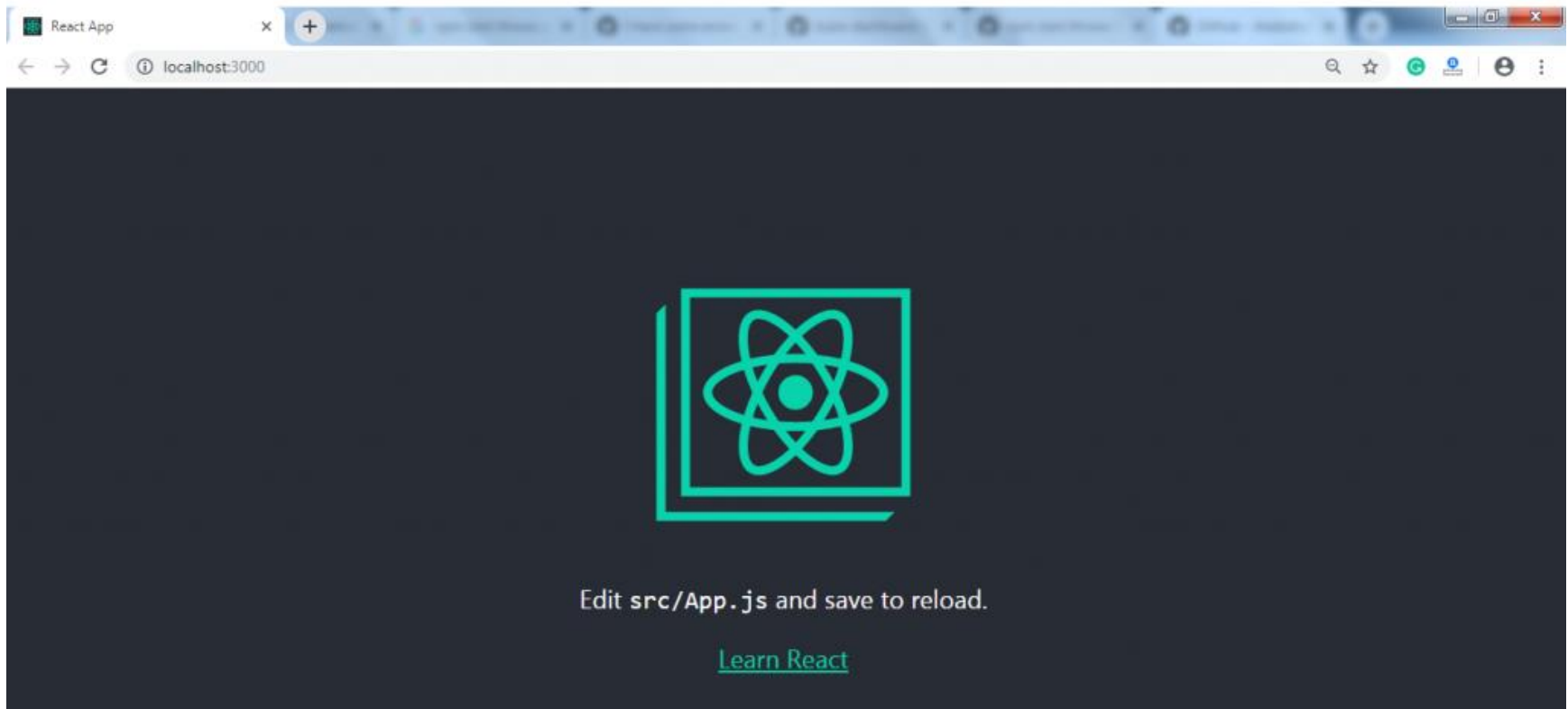
- **Step 7:** Now move to your project folder by using: `cd 'your folder name.'`

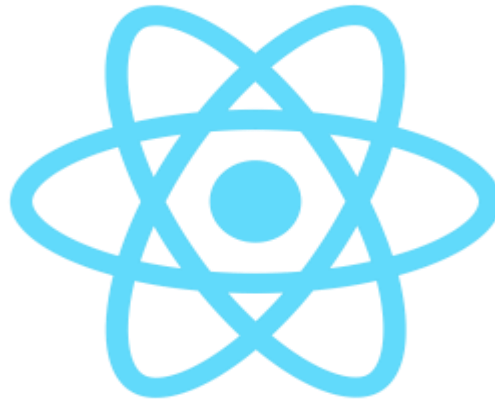
```
F:\react>cd reactfirst
```

- **Step 8:** Now write the command `npm start`

```
F:\react\reactfirst>npm start  
  
> reactfirst@0.1.0 start F:\react\reactfirst  
> react-scripts start  
  
Starting the development server...
```


using the create-react-app command





create-react-app Hands-on Demo

Thank You