# ReactJS

React is Open source, JavaScript library to build UI

### Framework vs. library

* Framework to do all domains thinks
* Library is to do one specific think in better way

## Component Based Architecture

Break down application into small parts, then composed to make more complex UIs

|  |  |
| --- | --- |
| HEADER | |
| SIDE NAV | MAIN CONTENT |
| FOOTER | |

## Advantages

1. Reusable code
   1. One componentX in react can be reused in Angular/ view.
2. React is declarative
   1. We need to tell what we want. React will do the work for us.
3. React handle efficiently
   1. Updating and Rendering Components
   2. DOM updates
4. Integrated with any applications
5. Fit to build part of a page or entire application
6. With react native mobile applications

### Imperative vs. declarative

Ask a child to draw a house. We need instruct every step to that child

Ask an Artist to draw a house he will draw

He;;p d

Dkjafkds

Djfakljddkjfl

## Start

Download and install nodeJS.org

create application

Cmd:

way1:

npx create-react-app projectName

npx : npm package runner

way2:

npm install create-react-app –g

Install the package “create-react-app” globally

create-react-app projectName

“create-react-app” package is responsible to create app

Way1 is good because we don’t need to update the “create-react-app” package

cd projectName

npm start

run the server in port 3000

## Folder structure

package.json

Configure dependency & script for the project

package-lock.json

Ensure consistence installation of dependency

node-modules

All dependencies are installed

Updates when create-react-app or npm install command

public/index.html

<body>

<div id=”root”></div> is responsible to display views

Known as DomElement/ root DomNote

Every think inside it will controlled by react

scr/

index.js

starting point

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

<App/> component going to be rendered inside domNot

import App from './App';

./ == src/

./App == src/App.js

So import src/App.js as App

App.js

So it’s going to be rendered inside domNote

import './App.css'; // src/App.css

function App() {

  return (

     <div className="App"> // css class App gonna be applied

<HTML ☺ >

</div>

  );

}

export default App;

App.test.js

This will automatically generates when App.js create

For unit test

App.css

This will automatically generates when App.js create

For styling App.js

.App {

  text-align: center;

}

.App-logo {

### Css selectors

P{ => <p>

#myId{ => <div id=”myId”>

.myClass{ => <h2 class=”myClass”>

p.center{ => <p class=”center”> will work

<h1 class=”center”> not work

<p class =”big center”> => .big{ and .center will work

## Running

npm start

index.html served in browser it has root node

next control enters to index.js

reactDom renders the App component in the rootDomNote

App component contains the HTML

## Components

|  |  |
| --- | --- |
| HEADER | |
| SIDE NAV | MAIN CONTENT |
| FOOTER | |

Describe part of UI

Root(App)

Component

Here App component contains every other component

Components are reusable

Same component can display different information with different properties

Example: Left Side Nav 🡺 Right Side Nav

Components can contain other Components

JS/ JSX file contains Component code

App.js

App

component

## Component types

### Stateless functional component

A JavaScript function

JS function

Properties(props) HTML(jsx)

It return html code (not exact html because it’s not static page)

function Greet(props) {

  return (

    <div className="Hello">

  <h2> Hello, {props.name}</h2>

  </div>);

}

Export default Greet

//export Greet component from this file to use some ware else

### Stateful class component

A class extending Component class

Class contains render() method, this returning HTML

class Welcome extends React.Component {

  render(){

    return <h2> Hello, {this.props.name}</h2>;

  }

}