01. REACT Introduction.

React is a front-end library developed by Facebook. It is used for handling the view layer for web and mobile apps. ReactJS allows us to create reusable UI components. It is currently one of the most popular JavaScript libraries and has a strong foundation and large community behind it.

1.1. What is React?

ReactJS is JavaScript library used for building reusable UI components. According to React official documentation, following is the definition −

> *React is a library for building composable user interfaces. It encourages the creation of reusable UI components, which present data that changes over time. Lots of people use React as the V in MVC. React abstracts away the DOM from you, offering a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using React Native. React implements one-way reactive data flow, which reduces the boilerplate and is easier to reason about than traditional data binding.*

To sum up:

- React is a JavaScript library created by Facebook
- React is a User Interface (UI) library
- React is a tool for building UI components

React Features

- **JSX – JSX is JavaScript syntax extension.** It isn't necessary to use JSX in React development, but it is recommended.
- **Components – React is all about components.** You need to think of everything as a component. This will help you maintain the code when working on larger scale projects.
- **Unidirectional data flow and Flux** − React implements one-way data flow which makes it easy to reason about your app. Flux is a pattern that helps keeping your data unidirectional.
- **License** − React is licensed under the Facebook Inc. Documentation is licensed under CC BY 4.0.

React Advantages

- **Uses virtual DOM which is a JavaScript object.** This will improve apps performance, since JavaScript virtual DOM is faster than the regular DOM.
- Can be used on **client and server** side as well as with other frameworks.
- **Component and data patterns improve readability**, which helps to maintain larger apps.

React Limitations

- **Covers only the view layer of the app**, hence you still need to choose other technologies to get a complete tooling set for development.
- **Uses inline templating and JSX**, which might seem awkward to some developers.

1.2. React Quickstart Tutorial

This is a quickstart tutorial. If you want to work with ReactJS, you need to have solid knowledge of JavaScript, HTML5, and CSS. Even though ReactJS doesn't use HTML, the JSX is similar so your HTML knowledge will be very helpful. We will also use EcmaScript 2015 syntax so any knowledge in this area can be helpful.

Before you start, you shoud have a basic understanding of:

- What is HTML
- What is CSS
- What is DOM
- What is ES6
- What is Node.js
- What is npm

1.3. Adding React to an HTML Page

This quickstart tutorial will add React to a page like this:

Example:

```
<!DOCTYPE html>
<html lang="en">
  <head>
```

```
    <title>Test React</title>

    <!-- Load React API -->
    <script src= "https://unpkg.com/react@16/umd/react.production.min.js">
    </script>

  <!-- Load React DOM-->
  <script src= "https://unpkg.com/react-dom@16/umd/react-
  dom.production.min.js">
  </script>

  <!-- Load Babel Compiler -->
  <script src="https://unpkg.com/babel-standalone@6.15.0/babel.min.js">
  </script>

  </head>

  <body>

      <script type="text/babel">
      //  JSX Babel code goes here
      </script>

  </body>
  </html>
```

1.4. What is Babel?

Babel is a JavaScript compiler that can translate markup or programming languages into JavaScript.

With Babel, you can use the newest features of JavaScript (*ES6 - ECMAScript 2015*).

Babel is available for different conversions. **React uses Babel to convert JSX into JavaScript**.

NOTE: Please note that <*script type="text/babel"*> is needed for using Babel.

1.5. What is JSX?

JSX stands for JavaScript XML.

JSX is an XML/HTML like extension to JavaScript.

Example:

```
const element = <h1>Hello World!</h1>
```

As you can see above, JSX is not JavaScript nor HTML.

JSX is a XML syntax extension to JavaScript that also comes with the full power of ES6 (ECMAScript 2015).

Just like HTML, JSX tags can have a tag names, attributes, and children. If an attribute is wrapped in curly braces, the value is a JavaScript expression.

NOTE: Note that JSX does not use quotes around the HTML text string.

1.6. React DOM Render

The method ReactDom.render() is used to render (display) HTML elements:

Example:

```
<div id="id01">Hello World!</div>

<script type="text/babel">
ReactDOM.render(
    <h1>Hello React!</h1>,
    document.getElementById('id01'));
</script>
```

1.7. JSX Expressions

Expressions can be used in JSX by wrapping them in curly {} braces.

Example:

```
<div id="id01">Hello World!</div>

<script type="text/babel">
const name = 'John Doe';
ReactDOM.render(
    <h1>Hello {name}!</h1>,
    document.getElementById('id01'));
</script>
```

1.8. React Elements

React applications are usually built around a single HTML element.

React developers often call this the root node (root element):

Example:

```
<div id="root"></div>
```

React elements look like this:

Example:

```
const element = <h1>Hello React!</h1>
```

Elements are rendered (displayed) withe the *ReactDOM.render()* method:

Example:

```
ReactDOM.render(element, document.getElementById('root'));
```

React elements are immutable. They cannot be changed.

The only way to change a React element is to render a new element every time:

Example:

```
function tick() {
    const element = (<h1>{new Date().toLocaleTimeString()}</h1>);
    ReactDOM.render(element, document.getElementById('root'));
}
setInterval(tick, 1000);
```

1.9. React Components

React components are JavaScript functions.

This example creates a React component named "Welcome":

Example:

```
function Welcome() {
    return <h1>Hello React!</h1>;
}
ReactDOM.render(<Welcome />, document.getElementById('root'));
```

React can also use ES6 classes to create components.

This example creates a React component named Welcome with a render method:

Example:

```
class Welcome extends React.Component {
    render() { return(<h1>Hello React!</h1>); }
}
ReactDOM.render(<Welcome />, document.getElementById('root'));
```

1.10. React Component Properties

This example creates a React component named "Welcome" with property aguments:

Example:

```
function Welcome(props) {
    return <h1>Hello {props.name}!</h1>;
}
ReactDOM.render(<Welcome name="John Doe"/>,
document.getElementById('root'));
```

React can also use ES6 classes to create components.

This example also creates a React component named "Welcome" with property arguments:

Example:

```
class Welcome extends React.Component {
    render() { return(<h1>Hello {this.props.name}</h1>); }
}
ReactDOM.render(<Welcome name="John Doe"/>,
document.getElementById('root'));
```

1.11. Create React Application

Facebook has created a **Create React Application** with everything you need to build a React app.

It is a development server that uses **Webpack** to compile React, JSX, and ES6, auto-prefix CSS files.

The *Create React App* uses **ESLint** to test and warn about mistakes in the code.

To create a *Create React App* run the following code on your terminal:

Example:

```
npx create-react-app react-tutorial
```

Make sure you have Node.js 5.2 or higher. Otherwise you must install npx:

Example:

```
npm i npx
```

Start one folder up from where you want your application to stay:

Example:

```
C:\Users\myUser>npx create-react-app react-tutorial
```

Ouput:

```
npx: installed 63 in 10.359s
Creating a new React app in C:\Users\myUser\react-tutorial.
Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...
+ react-dom@16.5.2
+ react@16.5.2
+ react-scripts@2.0.4
added 1732 packages from 664 contributors and audited 31900
packages in 355.501s
found 0 vulnerabilities+ react@16.5.2

Success! Created react-tutorial at C:\Users\myUser\react-tutorial
```

Inside that directory, you can run several commands:

Example:

```
npm start
```

Starts the development server.

Example:

```
npm run build
```

Bundles the app into static files for production.

Example:

```
npm test
```

Starts the test runner.

Example:

```
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:

Example:

```
cd react-tutorial
npm start
```