REACT

K-Digital Class 4

REACT 란?

- React는 Web Application에서 User Interface(UI)를 개발하기 위한 Open Source, JavaScript Library이다.
- React is developed and released by Facebook.
- Facebook은 지속적으로 React Library를 작업하고 있으며 Bug를 수정하고 새로운 기능을 도입하여 개선하고 있다.

REACT 란?

Features.

- React Library의 두드러진 특징은 다음과 같다.
  - 견고한 기본 아키텍처.
  - 확장 가능한 아키텍처.
  - 구성 요소 기반 라이브러리.
  - JSX 기반 설계 아키텍처.
  - 선언적 UI 라이브러리.
  - Single Page Applications을 Build하는 데 사용된다.
  - 재사용 가능한 UI 구성 요소를 만들 수 있다.

REACT 란?

Benefits

- React Library를 사용하면 다음과 같은 몇 가지 이점(Benefit)이 있다.
  - Easy to learn.
  - Easy to adapt in modern as well as legacy application.
  - Faster way to code a functionality.
  - Availability of large number of ready-made component.
  - Large and active community.

REACT 란?

Application

- Facebook, popular social media application.
- Instagram, popular photo sharing application.
- Netflix, popular media streaming application.
- Code Academy, popular online training application.
- Reddit, popular content sharing application.

REACT 설치

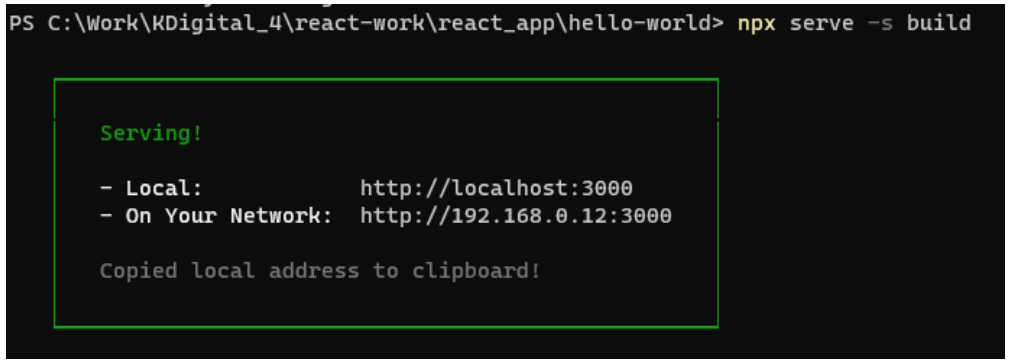
- npm install -g create-react-app
- npm install –g yarn
- npx create-react-app hello
- cd hello
- npm start
  - npm run build : 명령어를 입력하고 나면 build라는 디렉토리가 새로 생긴 것을 볼 수 있다.

```
import React from "react";
import ReactDOM from "react-dom";

function Hello(props) {
  return <h1>Hello World!</h1>;
}

ReactDOM.render(<Hello />, document.getElementById("root"));
```

Yarn을 사용하는 이유는 더 나은 속도, 더 나은 캐싱 시스템을 사용하기 위함이다.

REACT 설치

- npm insatll -g serve
- npx serve -s build
  - npx의 경우 한 번만 실행시킬 서버를 다운로드 하는 것이고, serve라는 웹서버를 다운받아서 실행 시킬 때, build directory를 root로 하겠다는 의미이다.



REACT 설치

Updating the toolchain

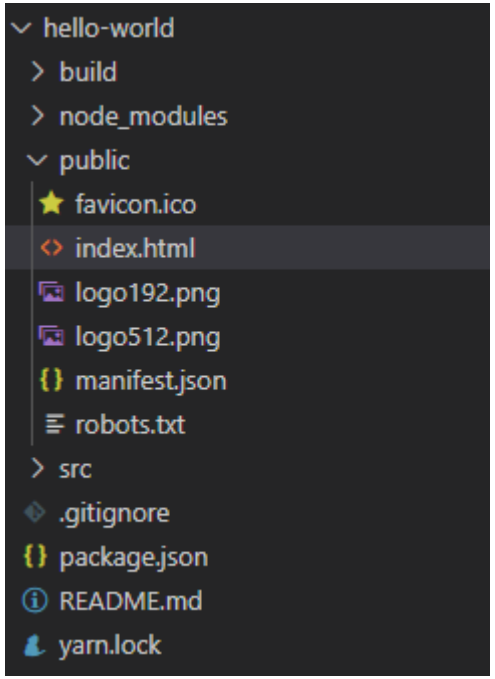
- react-create-app 툴체인은 react-scripts package를 사용하여 Application을 build하고 실행한다.
- Application이 작업을 시작하면 npm package manager를 사용하여 'react-script'를 언제든지 최신 버전으로 업데이트할 수 있다.

```
npm install react-scripts@latest
```

REACT SOURCE 구조

- React의 directory는 public과 src 2개로 나뉘어 있다. Public에는 index.htmI이 들어있다.

- 즉 웹의 메인 페이지가 public 이라는 directory에 포함되어 있다는 의미이다.



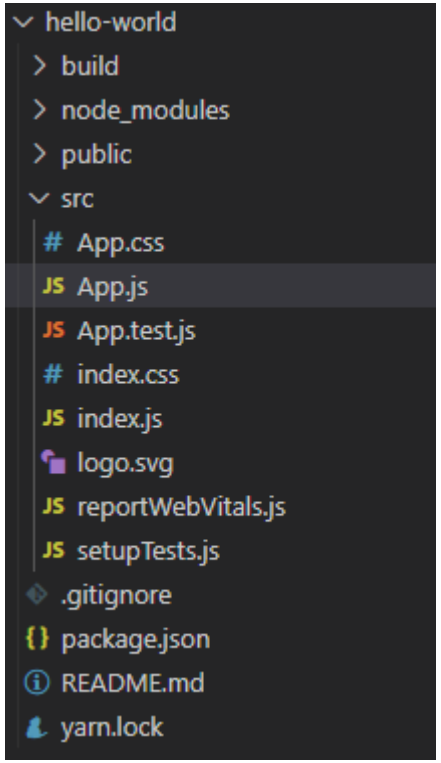
REACT SOURCE 구조

- react를 하면서 여러 컴포넌트들을 만들어 나가게 되는데, create-react-app은 div id ="root" 안에 들어가도록 약속되어 있다.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

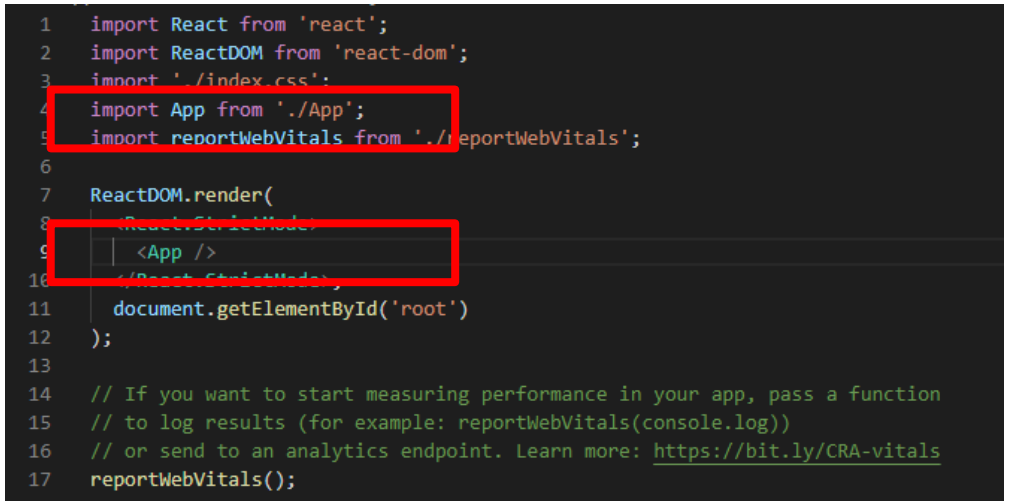
REACT SOURCE 구조

- VSCode로 보면 root라는 아이디 아래에 class App과 App-header 등등이 들어가 있는 것을 볼 수 있다.
- 그렇다면 이 아래에 있는 class들은 어떤 파일들을 수정해야 할까?
  - src 디렉토리에 있는 파일들을 이용해서 수정할 수 있다.
  - 그중 진입 파일은 index.js에 포함되어 있다.



REACT SOURCE 구조

- index.js 파일을 살펴보면 document.getElementById('root')라고 되어있는 부분을 볼 수 있다.
- 이 부분은 어디에 해당하느냐? 좀 전에 살펴본 index.html에서의 root 부분에 해당한다.
- <App />은 react를 통해 만든 사용자 정의 태그, 즉 컴포넌트이다.
- 이 컴포넌트의 실제 구현은 가장 상단에 import App from './App';이라고 되어있다. /App이라는 파일을 import해서 불러온 것이다.



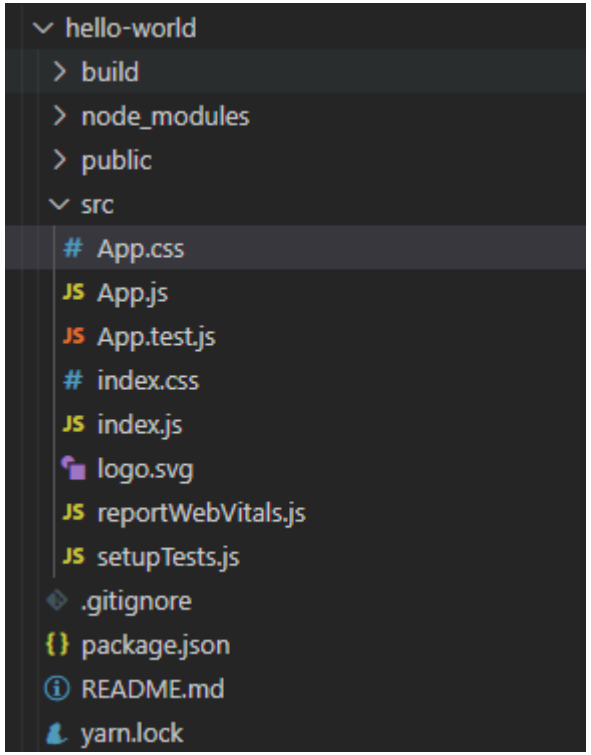
REACT SOURCE 구조

- src 디렉토리를 보면 App.js 라고 되어있는데, 이것 App이다. 즉, 이 App이라는 것을 불러와서 메인 화면을 구성한 것다. idnex에 출력되는 실제 내용은 부터 시작해서 마지막 까지가 내용이다.

```
function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}
```

CSS 코딩

- react에서 CSS를 수정하는 방법.
  - index.js 파일을 보면 idnex.css라는 파일이 있다. index.css 라는 파일을 수정 해보도록 한다.

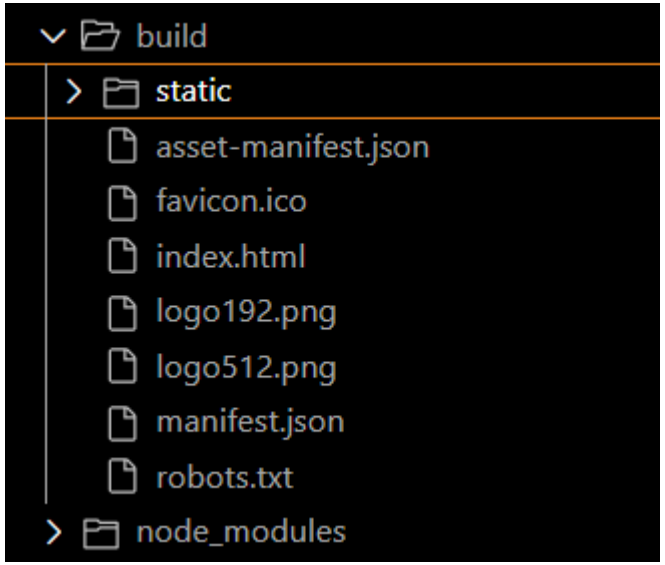


REACT 배포

build > index.html

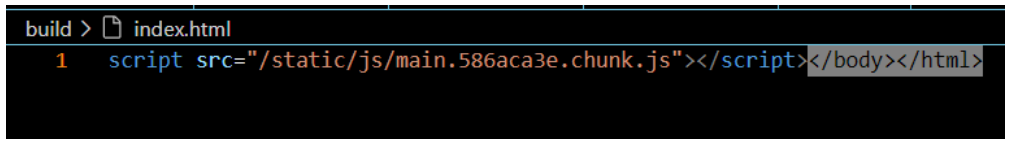
```
1   script src="/static/js/main.586aca3e.chunk.js"></script></body></html>
```

- React를 새로 고침 하면 그냥 hello react라는 문자밖에 없는데, 1.8MB나 된다.
- 현재 React는 개발의 편의성을 위해서 여러 가지 기능들을 추가해 놓은 상태이기 때문에, 상당히 무거운 편이다. 개발자만 사용하게 된다면 모르겠지만 이 파일을 그대로 유저가 이용하게 된다면 상당히 느려질 것이다.
- 이 부분을 어떻게 해결할지 알아보고자 한다. 이전에는 npm run start or npm start 라는 명령어로 웹 페이지를 실행시켰지만 실제 배포할 때는 'npm run build'를 실행 한다
- 명령어를 입력하고 나면 build라는 디렉토리가 새로 생긴 것을 볼 수 있다.



REACT 배포

- index.html을 한번 열어보면 이상하게 나오는 것을 볼 수 있다.
- 이렇게 나오는 이유는 원래 사용했던 index.html과 같이 불필요한 공백들과 같이 용량을 차지하는 것들을 전부 다 제거하기 위해서이다. 즉 실제 서비스할 때는 build 안에 있는 파일들을 사용하면 된다.



REACT 배포

- build를 할 때 다음과 같은 명령문이 뜨게 되는데 serve라고 하는 npm을 통해 설치할 수 있는 간단한 서버이다.
- 여기서 다시 용량을 살펴보면 1.8MB였던 Web Application이 엄청나게 줄어든 것을 볼 수 있다.
- 이런 식으로 React 전체를 다 Server로 돌리는 것이 아니라 build를 통해 용량을 줄여서 배포하면 유저 입장에서는 더 가벼운 Web Application을 이용할 수 있다.

```
npm insatll -g serve
npx serve -s build // 한번만 실행
```

HOW DOES REACT WORK?

- React는 Memory에 VIRTUAL DOM을 생성한다.
- Browser의 DOM을 직접 조작하는 대신 React는 Browser DOM을 변경하기 전에 필요한 모든 조작을 수행하는 Memory에 Virtual DOM을 생성한다.
- React는 변경해야 할 사항만 변경한다!
  - React는 어떤 변경 사항이 있는지 찾아내고 변경해야 할 부분만 변경한다.

REACT 시작하기

- React가 무엇인지에 대한 개요를 보려면 HTML로 직접 React 코드를 작성할 수 있다.
- 프로덕션에서 React를 사용하려면 npm과 Node.js가 설치되어 있어야 한다.

REACT 시작하기

React Directly in HTML

- React 학습을 시작하는 가장 빠른 방법은 HTML 파일에 React를 직접 작성하는 것이다.
  - Include three CDN's in your HTML file.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="https://unpkg.com/react@17/umd/react.development.js" crossorigin></script>
    <script src="https://unpkg.com/react-dom@17/umd/react-dom.development.js" crossorigin></script>
    <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
</head>
<body>

    <div id="mydiv"></div>

    <script type="text/babel">
      function Hello() {
        return <h1>Hello World!</h1>;
      }

      ReactDOM.render(<Hello />, document.getElementById('mydiv'))
    </script>

</body>
</html>
```

REACT 시작하기

CDN 이란?

- CDN (Content Delivery Network)은 곳곳에 분산되어 있는 서버 그룹을 말하며 이를 작동 시켜 인터넷 콘텐츠를 빠르게 전달 할 수 있는 서비스를 말한다. CDN을 사용하면 HTML 페이지, 자바 스크립트 파일, 스타일 시트, 이미지 및 비디오를 비롯한 인터넷 콘텐츠로드에 필요한 자산을 신속하게 전송할 수 있어 CDN 서비스의 인기는 계속해서 높아지고 있다.
- 또한 올바르게 구성된 CDN은 DDOS (Distributed Denial of Service) 공격과 같은 일부 악의적인 공격으로부터 웹 사이트를 보호하는 데 도움이 된다.

CDN 사용의 이점

- Web-Site Loading 시간 개선
  - 다른 최적화 중에서도 가까운 CDN 서버를 사용하여 웹 사이트 방문자에게 더 가까운 컨텐츠를 배포함으로써 방문자는 페이지 로드 시간이 더 빨라진다. 방문자가 느린 로딩 사이트를 클릭 할 확률이 높을수록 CDN은 이탈률을 낮추고 사람들이 사이트에 머문 시간을 늘릴 수 있다. 즉, 웹 사이트가 빠를수록 더 많은 방문자가 머물러 오래 있을 수 있다.
- 대역폭 비용 절감
  - 웹 사이트 호스팅을 위한 대역폭 비용은 웹 사이트의 주요 소비 비용이다. 캐싱 및 기타 최적화를 통해 CDN은 원본 서버를 제공해야하는 데이터 양을 줄여 웹 사이트 소유자의 호스팅 비용을 줄인다.
- 콘텐츠 가용성 및 이중화 증가
  - 대량의 트래픽 또는 하드웨어 오류로 인해 정상적인 웹 사이트 기능이 중단 될 수 있다. CDN은 분산 된 특성으로 인해 많은 트래픽을 처리 할 수 있으며 많은 Origin Server보다 하드웨어 오류를 견딜 수 있다.

MODIFY THE REACT APPLICATION

- src 폴더 안에 App.js라는 파일이 있습니다. 파일을 열면 다음과 같다.

```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

MODIFY THE REACT APPLICATION

- HTML 내용을 변경하고 파일을 저장한다.
- 파일을 저장한 직후 변경 사항이 표시되므로 브라우저를 다시 Load할 필요가 없다.

```
function App() {
  return (
    <div className="App">
      <h1>Hello World!</h1>
    </div>
  );
}

export default App;
```

REACT - ARCHITECTURE

- React Library는 견고한 기반 위에 구축되었다. 간단하고 유연하며 확장 가능하다.
- 앞에서 배웠듯이 React는 Web Application에서 User Interface를 생성하기 위한 Library이다. React의 주요 목적은 개발자가 순수한 JavaScript를 사용하여 User Interface를 만들 수 있도록 하는 것이다.
- 새로운 Template Language를 도입하는 대신 React는 다음과 같은 세 가지 간단한 개념을 도입한다.

- React elements
  - HTML DOM의 JavaScript 표현.
  - React는 React Element를 생성하기 위한 API React.createElement를 제공한다.
- JSX
  - User Interface를 디자인하기 위한 JavaScript 확장.
  - JSX는 수정이 거의 없는 HTML 구문을 지원하는 XML 기반의 확장 가능한 언어.
  - JSX는 React Elements로 컴파일하고 User Interface를 만드는 데 사용할 수 있다.
- React component
  - React 컴포넌트는 React 애플리케이션의 기본 빌딩 블록이다.React 요소와 JSX를 사용하여 User Interface를 design.
  - React 컴포넌트는 기본적으로 JavaScript 클래스 (React.component 클래스 확장) 또는 순수 JavaScript 함수.
  - React 구성 요소에는 속성, 상태 관리, 수명 주기 및 이벤트 핸들러가 있다.React 컴포넌트는 고급 로직 뿐만 아니라 단순하게도 할 수 있다.

REACT - WORKFLOW

- 간단한 React 애플리케이션을 만들고 분석하여 React 애플리케이션의 워크플로를 이해하도록 합시다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Reacat Application</title>
</head>
<body>
    <div id="react-app"></div>
      <script src="https://unpkg.com/react@17/umd/react.development.js" crossorigin></script>
      <script src="https://unpkg.com/react-dom@17/umd/react-dom.development.js" crossorigin></script>
      <script language="JavaScript">
        element = React.createElement('h1', {}, 'Hello React!')
        ReactDOM.render(element, document.getElementById('react-app'));
      </script>
</body>
</html>
```

REACT - WORKFLOW

React.createElement

- create React elements는 세 가지 parameter가 필요하다.
  - Element tag
  - Object로서의 element attributes.
  - Element content - It can contain nested React element as well(중첩된 React 요소도 포함할 수 있다).

ReactDOM.render

- Element를 container에 rendering하는 데 사용되며 두 개의 parameter가 필요하다.
  - React Element OR JSX.
  - Root element of the webpage.

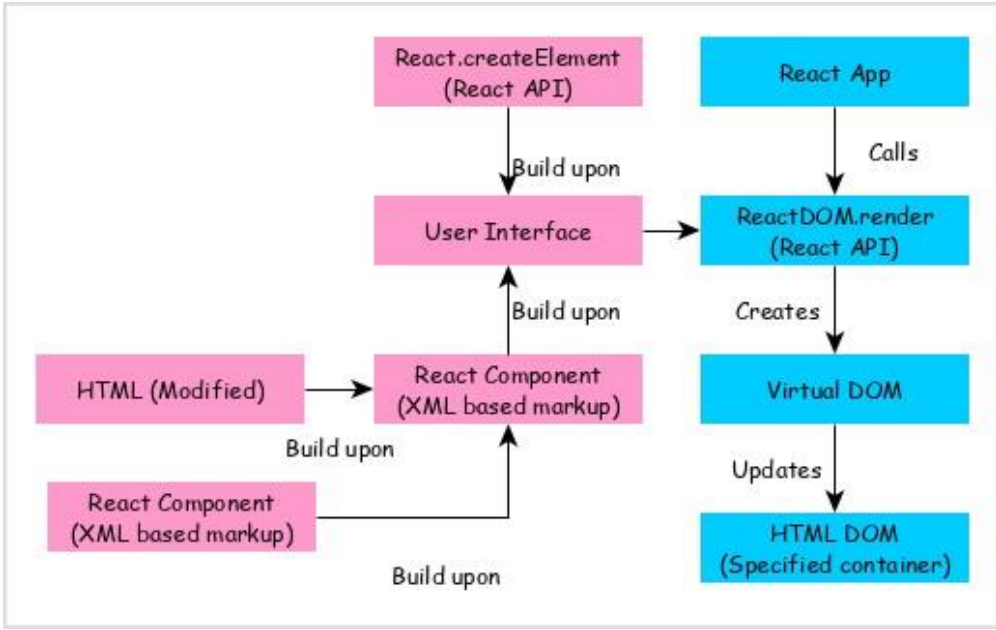
REACT - WORKFLOW

```
<script language="JavaScript">
    element = React.createElement('h1', {}, 'Hello React!')
    ReactDOM.render(element, document.getElementById('react-app'));
</script>
```

```
<div><h1> Hello React!</h1></div>
```

REACT - WORKFLOW

- 응용 프로그램을 분석하여 아래 다이어그램과 같이 React 응용 프로그램의 워크플로를 시각화 할 수 있다.
  - React 앱은 React 구성 요소(JSX 또는 React 요소 형식으로 코딩 됨)를 사용하여 만든 사용자 인터페이스와 사용자 인터페이스를 렌더링할 컨테이너를 전달하여 ReactDOM.render 메서드를 호출한다.
  - ReactDOM.render는 JSX 또는 React 요소를 처리하고 Virtual DOM을 내보낸다.
  - Virtual DOM이 병합되어 컨테이너에 렌더링된다.



ARCHITECTURE OF THE REACT APPLICATION

- React 라이브러리는 UI 라이브러리일 뿐이며 복잡한 애플리케이션을 작성하기 위해 특정 패턴을 강요하지 않는다.
- 개발자는 원하는 디자인 패턴을 자유롭게 선택할 수 있다. React 커뮤니티는 특정 디자인 패턴을 옹호한다. 패턴 중 하나는 Flux 패턴이다.

ARCHITECTURE OF THE REACT APPLICATION

- React 앱은 단일 root component(구성요소)로 시작한다.
- 각 component는 모든 수준에서 다른 component와 중첩될 수 있다.
- 구성은 React 라이브러리의 핵심 개념 중 하나이다.따라서 각 component는 다른 component에서 하나의 component를 상속하는 대신 더 작은 component를 구성하여 build된다.
- 대부분의 component는 User Interface component이다.
- React 앱은 routing, animation, 상태 관리 등과 같은 특정 목적을 위한 타사 구성 요소를 포함할 수 있다.