Javascript 라이브러리2

React

react 앱 만들기

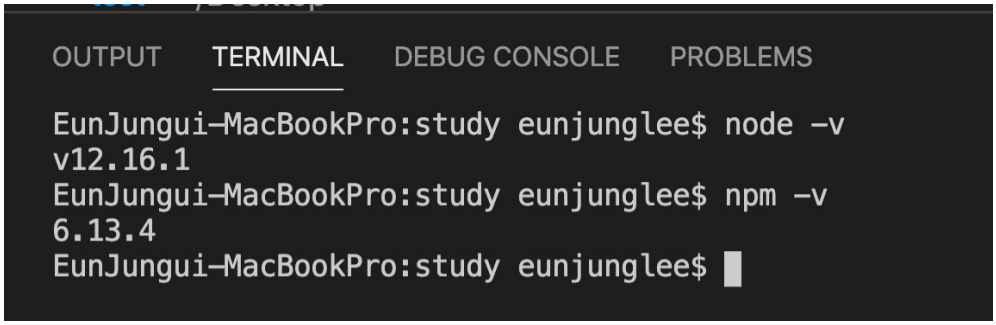
1. npm 을 사용하기

Nodes.org에서 다운로드 하기

LTS 버전: 안정적버전 -> 대부분 이것을 사용한다.
현재 버전: 최신버전-> 이것을 사용하기에는 아직 불안정한 부분이 있기에 사용하지 않는 것이 좋다.

다운로드 후 터이널에서 Node js 버전 확인하기

$node –v (Node 버전 확인)
$npm –v (Node 매니저 버전 확인)



2. React 패키지 설치하기

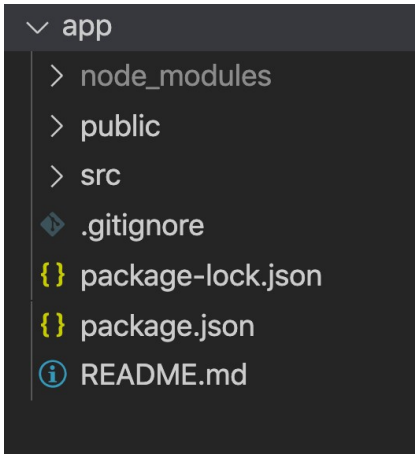
create-react-app 설치하기

create-react-app 란?

react를 쉽게 사용할 수 있게 셋팅 해 준다.
주의: 생성하는 폴더명을 대문자로 시작하면 안된다. 한글 폴더명 안 된다.

npx create-react-app <생성해야 하는 폴더명>

만약에 선택한 폴더에 사용할 경우에는 '.'을 입력한다.



node_modules : react modules이 이곳에 모여있다.

Public : 브라우저에서 로딩이 되는 기본 되는 화일들이 있는 곳이다.

Src:소스 화일이 자리를 잡고 있다.

react 실행 기본 명령어

npm start

현재 작업을 로컬 호스트를 통해 브라우저에 실행 시킨다.

npm run build

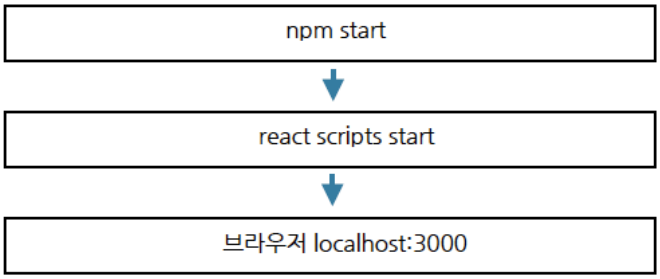
App 완성시 업로드 되는 압축 화일을 만들어 준다.

npm test

작업을 테스트 작업시 사용한다.

react실행하기

1. cd 폴더명
2. npm start



*화일 수집이 있을 경우 자동으로 loading 된다. 이런 경우를 hot reloading이라 한다.

*화일 수집이 있을 경우 자동으로 loading 된다. 이런 경우를 hot reloading이라 한다.

1. Import React form 'react';
2. Import App from './App' ;
3. ReactDOM.render(<App />, document.getElementById('root'));
4. index.html의 id="root"에 로딩된다.

5. 브라우저에 자동 loading 된다.

2. 기본 셋팅하기

필요소스 가져오기

```
import React, { 내장 소스 가져오기 } from 'react';
import ReactDOM from 'react-dom';
```

3. 컴포넌트 만들기

1. Class만들기

```
class 컴포넌트이름 extends React. Component {
  render() {
        return (
              JSX 코드
        );
    }
}
```

2 함수로 만들기

```
function 컴포넌트이름(){
        return (
              JSX코드
              )
}
```

출력하기

```
ReactDOM.render(
  <컴포넌트 />,
  document.getElementById('containe아이디명')
);
```

4. JSX사용하기

JSX는 객체 생성과 호출을 위한 문법적 편의를 제공하기 위해 만들어진 확장 기능이다. react 안에서는 코드를 JSX로 작성해야 한다.

1. 하나의 부모 container 박스안에 코드가 작성되어야 한다.

1. 부모 box 요소를 이용한다.
2. <fragment></fragment>를 이용한다.

3. <></> 를 이용한다.

```
1.
<div>
                <p>welcome</p>
                <p>React</p>
</div>
2.
<Fragment>
                <p>welcome</p>
                <p>React</p>
</Fragment>
** import React, { Fragment } from 'react';
3.
<>
    welcome;
</>
```

2. 기본코드 작성하기

A. 태그가 비어있다면 XML처럼 < /> 를 이용해 바로 닫아주어야 합니다.

B. {} 사용하기

자바스크립트변수 또는 식은 {} 로 감싼다.

숫자 속성값은 {} 로 감싼다.

C. 속성 사용시 주의사항

class-> className

tabindex -> tabIndex

3. 조건문 사용하기

if문은 즉시실행함수로 작성해야 한다.

삼항 연산자로 작성한다.

4. and , or 연산자 사용하기

and(&&) 연산자를 조건이 참일때만 작용이 된다.

or(||) 연산자는 조건이 거짓일 경우에만 작용이 된다.

5. null값과 underfined

null 값은 허용이 되지만 undifined는 작용이 되지 않는다.

6. 주석사용하기

JSX 안에서는 주석은 {/* */} 로 사용된다.

```
<ul>
  {/*  */}
  { title.map((str,index)
    =>(<li key={index}>{str}</li>))
```

7. JSX 안에서 스타일 사용하기

속성으로 사용하기

```
style = {
            {
            color:'red',
            backgroundColor:'yellow',
                        fontSize:'30px'
            }
      }
```

8. props 사용하기

props란 properties의 줄인 표현으로 부모 컴포넌느로 부처 전달된 속성값 혹은 상속받는 속성값을 말한다.

부모 컴포넌트가 자식 컴포넌트의 props를 설정하면 자식 컴포넌트에서는 해당 props를 사용할 수 있지만 변경하는 것은 불가능하다. props의 변경이 필요한 경우에는 props를 설정 및 전달한 부모 컴포넌트에서 변경해야 한다.

```
function Hello(props){
return (
  <div>
            <h1>Welcome {props.title}</h1>
            <p>Room {props.room}</p>
  </div>
);
}
ReactDOM.render(
<Hello title="react" list="classroom" room={5} />,
document.getElementById('root').    );
```

9. state 사용하기

props는 부모 컴포넌트에서 생성하고 이를 받아서 사용하는 사용하는 컴포넌트는 변경이 불가능하다. 컴포넌트 내부에서 값을 생성하고 변경하여 컴터넌트 상태를 관리할 수 있는 것이 state이다.

state는 상태라는 의미로 컴포넌트에서 변화할 수 있는 값을 나타내며 상태가 변하면 컴포넌트는 리렌더링(re-rendering)이 될 수 있다.

이전 버전에서는 네이티브로 작업 할 경우 함수형 컴포넌트에서는 state를 관리할 수 없었지만 현재로는 Hooks를 사용하여 관리할 수 있다.

리액트 Hooks 중 useState 사용하기

```
//import 하기
import React ,{useState} from 'react';

//state 설정하기
let [btn,setBtn] = useState('No');

//설정 값 변경하기
<button onClick={()=>setBtn('Yes')}>yes</button>
```

10 이벤트 사용하기

on이벤트 = 함수 발생 <- {함수의 이름}, {() =>}

```
<button onClick={()=>setBtn('Yes')}>yes</button>
```

11. form 사용하기

value -> defaultValue

for -> htmlFor

```
<button onClick={()=>setBtn('Yes')}>yes</button>
```

12. 반복문 -> 리스트와 key 사용하기

javascript

```
const numbers = [1, 2, 3, 4, 5];
const doubled = numbers.map((number) => number * 2);
console.log(doubled);
```

react

```
const numbers = [1, 2, 3, 4, 5];
const listItems = numbers.map((number) =>
  <li>{number}</li>
```

key 값이 생성 되어야 함.

Key는 React가 어떤 항목을 변경, 추가 또는 삭제할지 식별하는 것을 돕습니다. key는 엘리먼트에 안정적인 고유성을 부여하기 위해 배열 내부의 엘리먼트에 지정해야 합니다.

```
<ul>
    {
     title.map((str,index)=>(<li key={index}>{str}</li>))
    }
</ul>
```

CDN 사용하기

1. 개발자용

<script crossorigin src="https://unpkg.com/react@17/umd/react.development.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-dom.development.js"></script>

2. 사용자용

<script crossorigin src="https://unpkg.com/react@17/umd/react.production.min.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-dom.production.min.js"></script>

3. babel 사용하기

<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
이제 어떤 <script> 태그에서든 type="text/babel"성질을 추가하면 JSX를 사용할 수 있습니다.

4. 나의 script 사용하기

<script src="경로 연결" type="text/babel"></script>

```
<script crossorigin src="https://unpkg.com/react@17/umd/
react.production.min.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-
dom.production.min.js"></script>
<script crossorigin src="https://unpkg.com/babel-standalone@6/babel.min.js"></
script>
<script src="data/data.js"></script>
<script src="app_3.js" type="text/babel"></script>
```