**<About props>**

npm i(install) prop-types

🡪 father component로부터 받은 props에 대해서 알기 위해서 설치진행.

🡪 즉, 내가 전달받은 props가 내가 원하는 props인지를 확인해 준다.

🡪 package.json에서 확인할 수 있다.

*import* PropTypes *from* "prop-types";

텍스트, 검은색이(가) 표시된 사진

자동 생성된 설명

<prop types 예시>

<https://reactjs.org/docs/typechecking-with-proptypes.html#gatsby-focus-wrapper>

텍스트이(가) 표시된 사진

자동 생성된 설명

밑에 코드는 둘 다 같다.

const friends = ["dal", "mark", "lynn", "japan guy"]

friends.map(current => {

console.log(current);

return 0;

})

friends.map(function(current){

console.log(current);

return 0;

})

<”heart”를 추가해보기>

friends.map(current =>{

return current+”\_heart”;

})

print 🡪 ["da\_heart", "mark\_heart", "lynn\_heart", "japan guy\_heart"]

{foodILike.map(dish => <Food *name*={dish.name} *pic*={dish.image} />)}

위 코드처럼 dish => {}중괄호를 쓰지 않거나 ()소괄호를 쓰면 {return ~~~}과 같다.

**'Rendering' means calling the function that returns the component.**

**<오류>**

🡪 Warning: Each child in a list should have a unique "key" prop.

🡪 React에서 map함수를 쓰면 위와 같은 오류가 나타난다. (아래 링크 참조)

https://velog.io/@realryankim/Warning-Each-child-in-a-list-should-have-a-unique-key-prop

**<About State>**

🡪 state는 보통 우리가 동적 데이터와 함께 작업할 때 만들어진다.**(props가 해결할 수 없다.)**

동적 데이터 : 변하는 데이터, 존재하지않는 데이터, 생겨나고 사라지고 변경되는 데이터, 하나인 데이터, 두 개인 데이터 또는 그런 것들

**<arrow function>**

let sum = function(a, b){

return a + b;

}

sum = (a, b) => {

return a + b;

}

// 중괄호를 사용했다면, return 지시자로 결괏값을 반환해주어야 합니다.

sum = (a, b) => a + b; 🡪 {}가 없으면 한 줄로 쓰여진 code는 자동으로 return이 됨

아래와 같이 함수를 동적으로 만들 수 있다.

let age = prompt("나이를 알려주세요.", 18);

let welcome = (age < 18) ?

() => alert('안녕') :

() => alert("안녕하세요!");

welcome();

**<life cycle method>**

🡪 life cycle method는 react가 component를 생성하고 없애는 방법이다.

여러가지가 있는데 **mounting(“태어나는 것”) unmounting(“컴포넌트가 죽는 것”)**과 **updating**을 살펴보자

**■ mounting**

**1. constructor** : render()가 호출되기 전에 먼저 호출된다. render함수 위에console.log(“constructor”)와 render함수 안에 console.log(“render”)를 찍어보면 constructor가 먼저 호출됨을 알 수 있다.

**2. componentDidMount** : 1과 같이 componentDidMount함수에 console.log(“componentDidMount”)를 하면 **constructor -> render -> componentDidMount** 순서로 호출이 됨을 알 수 있다.

**3. componentWillUnmount** : component가 떠날 때 호출이 된다.(page를 refresh거나 page를 이동할 때)

**4. render** : view를 뿌려줄 때

**■ udpate**

🡪 내가 만든 함수 add, sub과 같은 것이 update에 관여한다.

**5. componentDidUpdate** : 이것은 add함수가 호출이 되면서 state의 변수를 바꿀 때마다 render -> componentDidUpdate함수가 호출이된다.

**<Movie App>**

// data를 fetch하는 것은 fetch()를 사용하면 되지만 axios로 진행한다.

// axios는 마치 fetch위에 있는 작은 layer와 같은 존재이다.

// axios를 사용하기 위해서는 npm install axios를 해야한다.

**<api를 사용함>**

YTS라는 불법적인 토렌트 영화를 업로드하는 사람들이 있는 토렌트에서 인터넷

**yts는 불법적으로 영화를 upload하고 있기 때문에 그들의 API는 항상 URL이 변경된다.**

<https://yts.mx/>

<https://yts.mx/api#list_movies>에서 <https://yts.mx/api/v2/list_movies.json>를 사용하는 것보다 nico가 만든(아래 url에서)

<https://github.com/serranoarevalo/yts-proxy>

<https://yts-proxy.now.sh/list_movies.json> --> 이것을 사용하자.

**<json viewer extension install at chrome browser>**

🡪 json파일을 가독성이 쉽게 도와준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

const {isLoading} = this.state;가 이해가 가지않아서 const {var} = this.state를 쳐봤는데

🡪 질문

텍스트이(가) 표시된 사진

자동 생성된 설명

🡪 해답

텍스트이(가) 표시된 사진

자동 생성된 설명

🡪 따라서 const {var} = this.state는 this.state.var를 const var에 넣어주는 것으로 간단히 이해하고 넘아가자. 또한 state object의 var와 const var의 var는 같아야 한다.

**<gh-pages>**

🡪 github에 업로드하여 webpage를 사용할 수 있게 해주는 모듈이다.

🡪 이것을 사용하기 위해서는 **package.json**을 수정해야 한다.



텍스트, 점수판이(가) 표시된 사진

자동 생성된 설명

**🡪 빨간 줄 추가해주어야 함!!!**

🡪 npm run build

**🡪 우리가 deploy를 호출할 때마다 predeploy(“npm run build”)가 먼저 호출된다.**

🡪 그러면 deploy에서는 build라는 폴더를 업로드해서 vscode내에 build라는 폴더를 볼 수 있다.

🡪 npm run build후에 npm run deploy실행!!!!

<npm install react-router-dom>

react-router-dom : navigator를 만드는데 도와주는 패키지이다.

**<vscode extension>**

🡪 eslint and prettier setting

<https://sunmon.github.io/vscode-eslint-prettier-setting/>