**배운 내용 요약**

function App() {

    let [글제목, 글제목변경] = useState(['남자 코트 추천', ' 강남 우동 맛집', '파이썬 독학']);

    let [좋아요, 좋아요변경] = useState([0,0,0]);

    let [modal, modal변경] = useState(false);

    let [입력값, 입력값변경] = useState('');

    // 변수

    let [누른제목, 누른제목변경] = useState(0);

    function 좋아요함수(idx){

        var newArray = [...좋아요];

        newArray[idx] = 좋아요[idx]+1;

        좋아요변경(newArray)

    }

    return (

        <div className="App">

            <div className="black-nav">

                <div>개발 Blog

                </div>

            </div>

            {/\* map으로 돌린 함수는 key={}를 안써주면 개발자도구에서 warning이 뜬다.

                파라미터의 정수값을 안에 넣어주기 \*/}

            {

                글제목.map(function(글, i){

                    return(

                        <div className = 'list' key={i}>

                        <h3  onClick={()=>{ 누른제목변경(i) }}> {글제목[i]}<span onClick={ ()=>{ 좋아요함수(i)}}> 👍 </span> {좋아요[i]}   </h3>

                        <p>2 월 17일 발행</p>

                        <hr/>

                    </div>

                    )

                })

            }

            {/\* input 태그 만들기

                onChange랑 onInput은 똑같음(리액트에서는)\*/}

            {/\* <input onChange={()=>{console.log('안녕')}}  /> \*/}

            {/\* 사용자가 입력한 값을 리액트에서 어떻게 가져오냐면은

                 e.target.value = 현재 이벤트가 동작한 곳.값 \*/}

            {/\* <input onChange={(e)=>{ console.log(e.target.value) }}  /> \*/}

            {/\* 이제진짜 사용자가 input에 입력한 값을 입력값 state로 저장해보자 \*/}

            <input onChange={(e)=>{ 입력값변경(e.target.value) }}  />

            {입력값}

            <button onClick={()=>{modal변경(!modal)}}>열고닫기</button>

            {

                modal===true

                // <Modal 작명 = {전송할state} ></Modal>

                ? <Modal 글제목 = {글제목} 누른제목={누른제목}></Modal>

                : null

            }

        </div>

    );

}

실제 사이트엔 <form> <input> 이런 것들이 매우 많습니다.

여기에 입력된 값을 저장하거나 어디로 보내거나 그래야할 경우가 많은데

리액트에서 input 데이터는 어떻게 처리하는지 알아봅시다.

실은 쌩 자바스크립트 문법이랑 똑같습니다.

그리고 map 반복문 쓰면 콘솔창에 warning 뜨는 것도 마지막 부분에서 해결해보도록 합시다.

**사용자가 input에 입력한 데이터를 state로 저장하는 법**

사용자가 input에 입력한 데이터는 역시나 중요한 데이터기 때문에 state 저장해서 쓰는게 일반적입니다.

저장부터하는 법을 알아봅시다.

저장하려면 일단 빈 state가 하나가 필요할테니 state를 하나 만들어봅시다.

function App (){

let [입력값, 입력값변경] = useState('');

return (

<div>

HTML 잔뜩있는 곳...

</div>

)

}

빈 state를 하나 만들어줍니다.

그리고 초기값은 따옴표 두개로 **빈 문자열**을 입력해주면 되겠습니다.

(state 초기값으로 뭐 담을지 모르겠으면 그냥 빈 문자나 빈 array 이런걸 담으시면 됩니다.)

**사용자가 input에 입력한 값 알아내는 법**

input에 입력한 값을 알고싶으면 input에 onChange 이벤트핸들러를 달고

그냥 쌩자바스크립트 문법을 쓰시면 됩니다.

function App (){

let [입력값, 입력값변경] = useState('');

return (

<div>

HTML 잔뜩있는 곳...

<input onChange={ (e)=>{ console.log(e.target.value) } } />

</div>

)

}

onChange라는건 input에 무언가 입력할 때마다 특정 함수를 동작시키고 싶을 때 사용합니다.

그래서 입력할 때마다 console.log(e.target.value)를 콘솔창에 출력하라고 써놓은 것입니다.

e.target이라는건 쌩자바스크립트 문법으로 '지금 이벤트가 동작하는 HTML요소'

.value라는건 input등에 입력한 값을 의미합니다.

그럼 이제 input에 뭔가 적을 때마다 콘솔창에 input에 입력한 값이 출력이되죠?

이게 리액트에서의 input에 입력한 값 구하는 법입니다.

<input>뿐만 아니라 <textarea>, <select> 이런 태그들에도 똑같이 사용가능합니다.

(참고)

<input>태그는 원래 HTML에선 그냥 쓸 수 있지만 리액트에선

<input /> 이렇게 쓰거나

<input></input> 이렇게 써야합니다.

안그럼 태그를 열었는데 안닫았다고 에러납니다.

**input에 뭔가 입력할 때마다 input에 입력된 값을 state에 저장해보자**

input에 뭔가 입력할 때마다 state에 그 값을 저장해봅시다.

그리고 일단 저장해놔야 나중에 활용할 수 있으니까요.

function App (){

let [입력값, 입력값변경] = useState('');

return (

<div>

HTML 잔뜩있는 곳...

<input onChange={ (e)=>{ 입력값변경(e.target.value) } />

</div>

)

}

이렇게 하면 되겠군요

입력값이라는 state를 변경할 땐 입력값변경이라는 함수를 쓸 수 있으니

거기 안에 e.target.value를 담아봤습니다.

그럼 이제 input에 뭔가 입력할 때마다 입력값이라는 state가 e.target.value로 변경됩니다.

끝입니다.

**e.target 이게 뭔소린데요**

쌩자바스크립트 문법인데 그냥 현재 onChange 이벤트가 동작하는 HTML을 뜻합니다. (input태그겠죠뭐)

그리고 .value는 그 HTML에 유저가 입력한 값을 출력해줍니다.

역시 기존 자바스크립트 문법을 잘 알아야 리액트도 잘합니다.

[collapse]

**map을 쓰면 콘솔창에 에러가 뜨는 이유는**

key={} 라는 속성을 안적어서 그렇습니다.

리액트는 반복문 돌린 HTML 요소엔 꼭 key={}를 적으라고 권장합니다.

key={} 안에는 반복문이 돌 때마다 0,1,2,3... 이렇게 하나씩 증가하는 변수같은걸 넣어주시면 됩니다.

그런 변수가 어딨냐고요? 모든 반복문엔 그런 변수가 항상 있습니다.

map 반복문에서 저번에 i라는 변수를 쓸 수 있다고 배워봤습니다.

function App (){

return (

<div>

...

{ 글제목.map(function(a, i){

return (

<div className="list" key={i}>

<h3 onClick={ ()=>{ 누른제목변경(i) } }>{ a } <span>?</span> </h3>

<p>2월 18일 발행</p>

<hr />

</div>

}) }

</div>

)

}

▲ i는 반복문이 돌 때마다 0,1,2,3.. 이렇게 증가하는 변수입니다. 그걸 이용해주시면 됩니다.

위처럼 key 속성을 적으시면 이제 반복문을 사용해도 에러가 나지 않습니다.