**배운 내용 요약**

Component 문법

텍스트이(가) 표시된 사진

자동 생성된 설명

// 관습 : 대문자로 시작, <>태그 여러개 나열 불가, 할거면 크게 하나로 묶으셈

/\*component 만들면 좋은 거

    1. 반복출현하는 html 덩어리

    2. 자주 변경되는 html ui

    3. 다른 페이지 만들 때\*/

/\*component 만들면 안좋은 거

    1. state 쓸 때 복잡함(props 문법 사용해야함)\*/

    return (

        <div className="App">

            <div className="black-nav">

                <div>개발 Blog

                </div>

            </div>

            <button onClick={ 제목변경 }>버튼</button>

            <div className='list'>

                <h3> {글제목[0]} <span onClick={()=>{좋아요변경(좋아요++)}}> 👍 </span> {좋아요} </h3>

                    <p>2월 17일 발행</p>

                    <hr/>

            </div>

            <div className = 'list'>

                <h3> {글제목[1]} <span onClick={ ()=>{console.log(1)}}> 👍 </span> 0  </h3>

                    <p>2월 17일 발행 </p>

                    <hr/>

            </div>

            <div className = 'list' >

                <h3> {글제목[2]}</h3>

                <p>2 월 17 일 발행</p>

                <hr/>

            </div>

            <Modal></Modal>

            <Modal/>

        </div>

    );

}

function Modal(){

    return(

        <div className="modal">

            <h2>제목</h2>

            <p>날씨</p>

            <p>상세내용 </p>

        </div>

        // 관습 : 대문자로 시작, <>태그 여러개 나열 불가, 할거면 크게 하나로 묶으셈

                                                                // <></>이거로 묶으셈

    )

}

/\*component 만들면 좋은 거

    1. 반복출현하는 html 덩어리

    2. 자주 변경되는 html ui

    3. 다른 페이지 만들 때\*/

/\*component 만들면 안좋은 거

    1. state 쓸 때 복잡함(props 문법 사용해야함)\*/

오늘은 모달창처럼 뜨는 상세페이지를 한번 만들어보도록 합시다.

근데 모달창 하나에 필요한 <div>가 좀 많을 것으로 예상합니다.

그래서 많은 HTML을 한 단어로 깔끔하게 축약할 수 있는 Component 문법을 이용해서 만들겁니다.

**상세페이지 겸 모달창 UI를 하나 만들어봅시다**

일단 HTML CSS부터 잘 짜서 여러분의 메인페이지 안의 원하는 곳에 넣어보십시오.

<div className="modal">

<h2>제목</h2>

<p>날짜</p>

<p>상세내용</p>

</div>

.modal{

margin-top : 20px;

padding : 20px;

background : #eee;

}

각각 App.js, App.css 아무데나 집어넣어보십시오.

근데 HTML짤 때 유의할 점은, return () 안에 HTML을 넣으실 때

return(

<div></div>

<div></div>

)

이런거 안됩니다. 태그 2개를 평행하게 적어넣을 수 없습니다.

굳이 쓰고 싶다면 이걸 <div>같은 걸로 한번 더 감싸셔야합니다.

return(

<div>

<div></div>

<div></div>

</div>

)

이렇게 하셔야합니다. return() 안엔 태그 하나만 들어갈 수 있습니다.

**복잡한 HTML들을 한 단어로 치환할 수 있는 Component 문법**

리액트는 긴 HTML을 한 단어로 깔끔하게 치환해서 넣을 수 있는 문법을 제공합니다.

Component라고 합니다.

이걸 이용하시면 함수 만들듯, 변수 만들듯 HTML을 깔끔하게 한 단어로 치환해서 원하는 곳에 꽂아넣을 수 있습니다.

우리 방금 만든 모달창도 깔끔하게 한 단어로 치환해봅시다.

이렇게 따라합니다.

function App (){

return (

<div>

HTML 잔뜩있는 곳

...

<Modal></Modal>

</div>

)

}

function Modal(){

return (

<div className="modal">

<h2>제목</h2>

<p>날짜</p>

<p>상세내용</p>

</div>

)

}

▲ 이렇게 하시면 원하는 HTML을 한 단어로 줄일 수 있습니다.

줄이는 법은

1. function을 이용해서 함수를 하나 만들어주시면 되고

2. 그 함수 안에 return () 안에 원하는 HTML을 담으시면 됩니다.

3. 그럼 원하는 곳에서 **<Modal></Modal>** 이라고 사용했을 때 아까 축약한 HTML이 등장합니다.

이렇게 축약한 HTML 덩어리를 **Component** 라고 칭합니다.

앞으로 HTML 깔끔하게 축약해서 쓰고싶으면 Component 이런 식으로 많이 만들어 쓰십시오.

Component 문법을 사용하면 <div>지옥보다는 깔끔하게 <Modal> 이렇게 되어있으니

뭐가 뭔지 바로 파악이 가능하고 나중에 관리하기도 좋겠죠?

**Component의 특징**

1. Component 이름지으실 땐 영어대문자로 보통 시작합니다.

2. return () 안엔 태그들이 평행하게 여러개 들어갈 수 없습니다.

평행하게 여러개의 태그를 쓰고 싶으면 <div>로 묶으셔야합니다.

아니면 의미없는 div를 쓰기 싫으시면 <> </> 이걸로 묶으셔도 됩니다.

function 예시컴포넌트(){

return (

<>

<div>안녕</div>

<div>안녕</div>

<div>안녕</div>

</>

)

}

▲ fragments라는 문법입니다. 의미없는 <div>쓰기 싫을 때 사용가능합니다.

3. component 위치는.. function App(){} 이것과 보통 나란히 만들어줍니다.

왜냐면 function App(){} 이것도 다시보니 컴포넌트 생성문법이죠?

보통 컴포넌트용 function 안에다가 컴포넌트용 function을 만들진 않습니다.

4. 강의에서 언급은 안했는데, component 안에 미리 만들어둔 component 집어넣기도 가능합니다.

function Modal(){

return (

<div className="modal">

<h2>제목</h2>

<p>날짜</p>

<p>상세내용</p>

<다른컴포넌트></다른컴포넌트>

</div>

)

}

function 다른컴포넌트(){

return ( <div> 안녕 </div> )

}

▲ 그니까 앞서 만든 Modal 컴포넌트의 <div> 안에 다른 컴포넌트를 만들어서 집어넣을 수도 있다는 소리입니다.

그냥 만들어서 집어넣으시면 됩니다.

컴포넌트 이름은 한글로 작성시 작동하지 않을 수 있습니다.

**어떤 HTML들을 Component 만드는게 좋을까**

기준은 없습니다만 관습적으로 어떤 부분을 주로 Component화 하냐면

- 사이트에 반복해서 출현하는 HTML 덩어리들은 Component로 만들면 좋습니다.

- 내용이 매우 자주 변경될 것 같은 HTML 부분을 잘라서 Component로 만들면 좋습니다.

- 다른 페이지를 만들고 싶다면 그 페이지의 HTML 내용을 하나의 컴포넌트로 만드시는게 좋습니다. (그냥 좋을 뿐 필수는 아님)

- 또는 다른 팀원과 협업할 때 웹페이지를 컴포넌트 단위로 나눠서 작업을 분배하기도 합니다.

님들 함수문법 언제씁니까 긴 코드 축약할 때, 다른 곳에서 코드 재사용할 때, 기능별로 나눌 때 쓰지 않습니까

그냥 함수 문법이랑 똑같이 여겨주면 됩니다.

**Component의 단점**

일단 HTML 깔끔하게 쓰려고 함수 자체를 많이 만들면 그것 만으로도 관리가 힘듭니다.

하지만 가장 큰 단점은 <Modal>이라는 컴포넌트가 App(){} 안에 있는 state를 사용하고 싶을 때 그냥 바로 쓸 수 없다는 것입니다.

props라는 문법을 이용해 state를 <Modal>까지 전해줘야 비로소 사용가능합니다.

props를 배우진 않았지만 듣기만 해도 귀찮죠?

그러니까 리액트 갓 배운 초보처럼 막 온갖 잡다한걸 Component로 쪼개지 말고 꼭 필요한 곳만 Component로 쪼개시길 바랍니다.