**배운 내용 요약**

npm start

jsx 문법1 : class=""를 넣고 싶으면 className=""

jsx 문법2 : 데이터바인딩은 중괄호안에 {데이터바인딩하고 싶은 변수명}만 담으시면

jsx 문법3 : 스타일용 오브젝트 자료형은 - { 속성명 : '속성값' } 이렇게 넣으시면

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**저번시간 셋팅이 완료되었으면 App.js로 들어갑시다**

App.js가 여러분의 메인페이지 역할을 하는 파일입니다.

App.js를 잘 보시면 div가 여러개 등장하는 HTML 부분이 있습니다.

필요없는 HTML 내용들은 다 지우고 div 하나만 남기고 깔끔한 마음으로 시작해보도록 합시다.

(App.js의 function App()부분)

function App(){

return (

<div className="App">

*//다지움 ㅅㄱ*

</div>

)

}

다 지웠으면 이제 미리보기를 한번 띄워봅시다.

터미널을 여시고 npm run start 를 입력해주시면 됩니다.

혹시 yarn이라는 이상한게 깔려있으면 yarn start 입력해주시면 됩니다.

**본격적으로 블로그 상단 nav를 제작해봅시다.**

리액트 환경에서 HTML 레이아웃 어떻게 만드냐고요?

리액트라고 뭐 다르게 코드짠다고 오해하는 분들이 많은데

그냥 옛 방식 그대로 똑같이 div 떡칠해서 짜시면 됩니다.

한번 상단 navbar 역할을 할 div를 하나 만들고 CSS를 넣어보도록 합시다.

CSS는 App.css 파일에 넣으시면 됩니다.

(App.js)

function App(){

return (

<div className="App">

<div className="black-nav">

<div>개발 blog</div>

</div>

</div>

)

}

(App.css)

.black-nav {

background : black;

width : 100%;

display : flex;

color : white;

padding : 20px;

font-weight : 600;

font-size : 20px;

}

저장까지하시면 미리보기 화면에서 예쁜 검은 navbar 하나가 생성됩니다. 끝!

근데 HTML을 잘 보시면 className="" 이라는 이상한 속성이 하나 있죠?

이게 오늘 배울 첫째 문법입니다.

리액트에서 class=""를 넣고 싶으면 className="" 이라고 사용하셔야합니다.

실은 여러분 HTML을 작성하는 것 같지만 HTML이 아니라 JSX라는 문법입니다.

자바스크립트 파일 안에서 HTML을 직관적으로 작성하기 위해 도와주는 리액트 기본 내장 문법이라고 생각하시면 됩니다.

JSX도 일종의 자바스크립트기 때문에

자바스크립트에서 사용하는 예약어인 class라는 키워드를 막 사용하시면 안됩니다.

className이라고 쓰셔야 CSS 파일의 class명을 박아넣을 수 있습니다.

별거아니니까 빠르게 외우십시오.

**JSX에서 데이터바인딩하기**

데이터바인딩이라는 전문용어는 별거아니고

자바스크립트 데이터를 HTML에 꽂아넣는 작업을 뜻합니다.

특히 요즘 프론트엔드 웹앱 개발시 데이터바인딩할 작업이 매우 많은데

리액트는 이걸 매우 쉽게 구현할 수 있습니다.

function App(){

var data = '안녕하세요';

return (

<div className="App">

<div className="black-nav">

<div>개발 blog</div>

<div>여기에 꽂고 싶으면?</div>

</div>

</div>

)

}

위의 예제를 보시면 data라는 변수를 하나 만들었습니다. (함수 안에 변수만드는건 자유입니다.)

근데 이 data라는 변수를 <div>안에 꽂아 보여주고 싶습니다.

이걸 어떻게 할까요?

옛날 자바스크립트 문법을 쓴다면 document.getElementById().innerHTML = ?? 이런 식이었겠죠.

근데 리액트는 더 쉽게 데이터를 꽂아넣을 수 있습니다.

function App(){

var data = '안녕하세요';

return (

<div className="App">

<div className="black-nav">

<div>개발 blog</div>

<div>{ data }</div>

</div>

</div>

)

}

중괄호안에 데이터바인딩하고 싶은 변수명만 담으시면 됩니다.

그럼 실제 미리보기 해보셨을 때 **<div>안녕하세요</div>**가 출력됩니다.

변수명 뿐만 아니라 미리 만들어둔 함수명이든 뭐든 별걸 다 집어넣을 수 있습니다.

그리고 href, id, className, src 등 여러가지 HTML 속성들에도 데이터바인딩이 가능합니다.

function App(){

var data = 'nav';

return (

<div className="App">

<div className="black-nav">

<div>개발 blog</div>

<div className={data}>안녕하세요</div>

</div>

</div>

)

}

이런 식으로도 온갖 곳에 {} 중괄호를 열어서 변수들을 집어넣을 수 있습니다.

위의 예제는 **<div className="nav">안녕하세요</div>** 라고 데이터바인딩하는 중입니다

**HTML에 스타일을 직접 넣고 싶으면**

style="color : blue" 이런걸 넣고 싶으면 어떻게 하는지 알려드리겠습니다.

JSX 상에서는 무조건 {} 오브젝트로 바꿔서 넣으셔야합니다.

<div style={ 스타일용 오브젝트 }> 글씨 </div>

이렇게 하셔야합니다. (중괄호 있는거 유의하세요)

스타일용 오브젝트 자료형은 어떻게 만드냐면

<div style={ {color : 'blue', fontSize : '30px'} }> 글씨 </div>

이렇게 하셔야합니다.

- { 속성명 : '속성값' } 이렇게 넣으시면 됩니다.

- 근데 속성명에 - (대쉬)기호를 쓸 수 없습니다. 대쉬기호 대신 모든 단어를 붙여써야합니다. 붙여쓸 땐 앞글자를 대문자로 치환해야합니다.

세상에 style 넣으려면 저런 짓을 해야한다니.. 복잡해보이고 보기싫죠?

그러니 이걸 변수로 따로 저장해놓고 style={변수명} 이렇게 넣으시거나

아니면 그냥 얌전히 CSS파일에 class를 만들어 사용하도록 합시다.

      <div className="black-nav">

        <div style={{color:'blue', fontSize: '30px' }}>개발 Blog</div>

      </div>

      <h4>{posts}</h4>

      <h4>{함수()}</h4>

      <img src={logo}/>

      <div className={posts}></div>

body {

    font-family: 'nanumsquare ';

}

.black-nav {

    background: black;

    width: 100%;

    display: flex;

    color: white;

    padding: 20px;

    font-weight: 600;

    font-size: 20px;

}