**컴포넌트 파해치기**

-파일에서 JSX를 사용하려면 꼭 React를 import 해주어야함

import React, {Component} from ’react’;

-이 import는 webpack을 사용하기에 가능한 것.

-컴포넌트를 만드는 두가지 방법

-클래스로 만드는 방법

class App extends Component {

render() { // 클래스로 만들시 render 함수가 꼭 있어야한다.

return (…); // 안에는 return을 적고, html 요소를 넣는다.

};

}

-함수로 만드는 방법

function App() {

return (

… //JSX 내용

);

}

-마지막엔 export default App; 으로 내보내고 있음.

-index.js를 보면, .App.js 파일을 import하여 내보내고 있음

ReactDOM.render(<APP />, document.getElementById(‘root’));

//만든 컴포넌트를 내보낼 땐 이렇게 ReactDOM.render() 함수를 쓴다.

//첫번째는 렌더링 할 결과물이며, 두번째는 어떤 DOM에 넣을지 정하는 것.

**JSX**

-html 같지만 자바스크립트.

-HTML과 비슷한 문법을 작성하면 이를 React.createElement를 사용하는 자바스크립트로 변환 시켜줌

-뭔가 return () 사이에 html 처럼 쓰이는 것

**JSX 규칙**

-태그는 꼭 닫혀야 한다. html하곤 달리 모든 태그는 반드시 닫아주자.

-두 개 이상의 엘리먼트는 무조건 하나의 엘리먼트로 감싸져야 한다.

<div> </div> <div> </div> -> 오류

<div> <div></div><div></div> </div> -> 정상

//똑같이 div를 쓰면 안될 땐 Fragment를 사용하면 됨.

import React, {Component, Fragment } from ‘react’;

<Fragment> <div></div><div></div> </Fragment>

-ES6에선 var을 쓰지 않고, let과 const로만 구현해주자.

-조건부 렌더링 : JSX에선 삼항 연산자, AND 연산자를 주로 사용함. if문을 쓰고 싶으면 IIFE로 사용해야함

ex) 삼항연산자 사용

<div>

{

1+ 1 === 2

? (<div>맞아요</div>)

: (<div>틀려요</div>)

}

</div>

ex) AND 연산자 사용. 앞이 true일 때 뒤를 검사하면서 실행.

<div>

{

1 + 1 === 2 && (<div>맞아요!</div>)

}

</div>

ex) IIFE (즉시 실행 함수 표현) 웬만하면 밖에서 로직을 만들고 꼭 해야할때만 쓰자. switch문도

<div>

{

(() => {

if(value === 1) return (<div>하나</div>)

if(value === 2) return (<div>하나</div>)

})()

}

</div>

-style과 className : style은 같은 파일내 객체로 선언한 css 쓸때,

className은 다른 css파일 import해서 쓸 때

// 같은 파일 내

const style = {

backgroundColor : ‘black’

};

return (

<div style={style}>

hi

</div>

);

//다른 파일

//./App.css 파일 내

.App {

background : black;

color : aqua;

}

//./App.js

import ‘./App.css’ //css Import할 땐 이름이랑 ; 안붙여도 됨?

function App() {

return (

<div className=”App”>

hi

</div>

);

}

-주석 : {/\* \*/} 로 아무데나, 또는 태그 안에 //로

<div>

{/\* 주석 \*/}

<h1

//태그 사이

>hello world!</h1>

</div>