React

<1>

페이스북.com에서 ui(사용자 인터페이스)를 더 잘 만들기 위해 페이스북에서 만든 자바스크립트 UI이다.

웹사이트는 매우 빠른 속도로 복잡해진다. 정보가 많아질수록 정보를 표현하는 html태그들은 점점 복잡해진다.

탑을 이루는 수많은 태그들을 따로 빼서 top.html이라는 파일을 만들고, 메인 html에 <top></top>이라고 기술을 하고, 또 사이드를 이루는 수많은 태그들을 따로 빼서 side.html이라는 파일을 만들어 <side></side>라고 기술을 함

즉 사용자 정의 태그를 만들어 활용

메인 html에서 로딩한 다음에 (include의 개념인 듯)

이렇게 사용하는 사용자 정의 태그를 리액트에서는 Component라고 부름 🡪 가독성, 재사용성, 유지보수

재사용성 : 여러 파일에서 같은 코드를 재사용할 수 있음

유지보수 : 한번에 여러 컴포넌트를 업데이트할 수 있음

<2>

1. 개발환경, coding

2. 실행해서 결과 분석, run

3. deploy, 소비자에게 전달

이렇게 먼저 배워볼 것이다.

<3>

어떻게 하면 저 선생으로부터 독립할까!!!

get started에 들어가서?

online playground : 온라인 상으로 react를 이용

부분적으로 리액트를 이용하고 싶을 때 🡪 add react

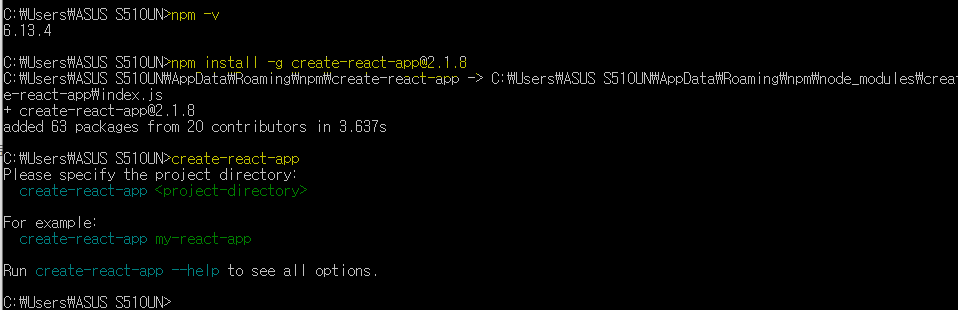
create a new react app 🡪 toolchain, 우리가 필요한 도구들의 모음, 추천 toolchain이 있음, create react app이라는 것을 사용할 것임

npm이 필요하다? node.js으로 만들어진 프로그램들을 쉽게 설치하기 위해 필요 (구글플레이 앱스토어의 역할을 한다)

<4>

node.js사이트에서 원하는 버전의 node.js를 다운받은 후에 명령 프롬프트를 통해  
npm -v  
를 입력한 후, 숫자가 나오는지 확인한다. 숫자가 나오면 잘 설치된 것이다.

그 다음 npm을 통해 create react app을 설치할 것인데…  
npm install -g create-react-app@2.1.8  
라고 입력한다. 위 명령어의 의미는 ‘npm에게 create-react-app의 2.1.8버전을 global로 설치해줘’라는 의미이며, global로 설치를 할 경우, 컴퓨터 어디에서나 실행할 수 있기에 아주 편리  
최신 버전을 설치하고 싶다면 2.1.8을 제외한 뒤에 설치하면 된다.  
마지막으로  
create-react-app이라고 입력한 후에, 잘 출력되는지 확인하면 된다.



<5> 개발환경구축

바탕화면에 react-app이라는 디렉토리를 만들고, 설치한 앱에게 react개발환경으로 만들어줘!라고 부탁할 것이다.

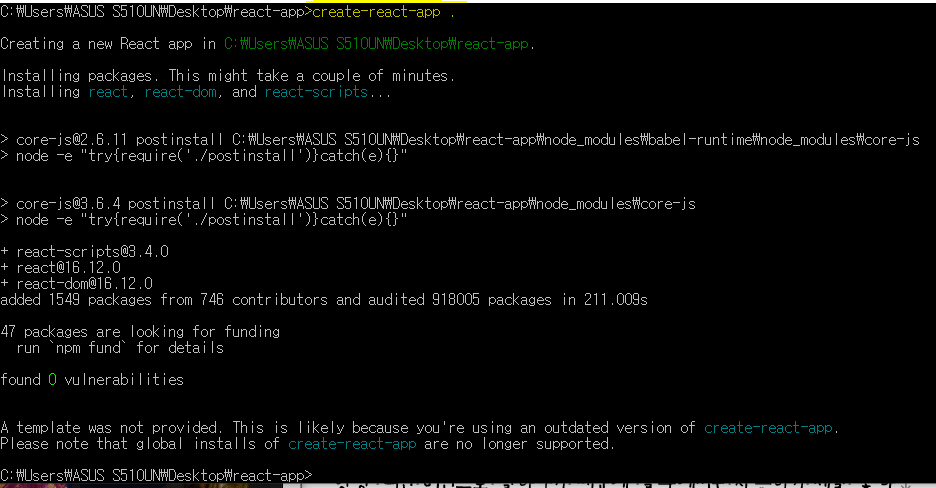
1. 바탕화면에 react-app이라는 폴더를 만든다.

2. cmd를 통해 react-app폴더 안으로 들어간다. (폴더를 당겨 cmd에 넣으면 경로가 자동으로 입력된다.)

3. create-react-app . (.은 현재 디렉토리라는 의미)

4. 작업이 끝나면 폴더 안에 파일이 생길 것 🡪 create-react-app이 우리가 작업하는 데에 필요한 것들을 다 알아서 장만해 놓는 것

공식 페이지를 보면 npx create-react-app? npx는 무엇인가?  
npm을 깔면 자동으로 깔리는데, npx는 일회용으로 사용할 create-react-app을 깔아주고, 작업이 끝나면 자동으로 삭제까지 해주는 친구이다.  
항상 최신 버전으로 깔아주기 때문에, 실무에서는 사용하기 제일 좋다.  
현재는 수업을 진행하기 위해 버전을 픽스한 것이다.



<6>

editor를 깔아준다. 강의영상에서는 아무거나 사용하라고 하였지만, 강사님과 똑 같은 editor를 사용하기 위해 visual studio code를 설치하였다.

view🡪appearance🡪toggle panel🡪TERMINAL을 들어가면 커맨드라인이 내부적으로 설치되어있다.

여기서  
npm run start  
를 입력하면 나오는 주소의 브라우저 창에서 create-react-app에서 react를 사용하여 미리 만들어 놓은 기본적인 app이 열린다.  
종료할 때는 ctral+c를 눌러 할 수 있고, 다시 시작하려면 npm run start를 입력해주면 된다.

<7>

샘플 app을 수정해보자!  
(어떤 디렉토리를 가지며, 어디를 수정해야하는지 알아보자)

디렉토리 구조를 보면 src, public이 있는데, public은 index.html이 있는 곳이다.  
현재 보이는 페이지가 index.html의 페이지이다.

index.html을 보면 중요한 부분이 id=’root’~~라고 있는데,  
리액트를 통해 컴포넌트들을 만들어 갈 때, id가 root인 곳인 태그 안에 넣을 수 있도록 약속한다.  
코드를 확인하면 리액트를 통해 만든 태그들이 root 안에 들어간 것을 확인할 수 있다.  
물론 id가 root인 것으로 약속한 것은 변경가능하다.

그렇다면 컴포넌트들은 어디서 수정할 수 있는가? 그것이 바로 src이다.

앞으로 리액트로 개발하게 되면 src라는 디렉토리 안에 넣을 것이다.  
진입 파일은 index.js이고, 이 파일에서 주목해야 할 부분은  
document.getElementById9(‘root’)인데, 이것을 통해 id를 정할 수 있다.  
App이 바로 컴포넌트 명이며 .App이 파일이름이다. .js가 생략되어 있다.

수업에서 class type을 사용할 것이기에 바꾸길 바란다.

create-react-app이 샘플로 만든 App이라는 사용자 정의 태그는 컴포넌트이며, App.js파일 안에 정의되어 있다.  
실제로 삽입된 코드를 보면 동일한 것을 볼 수 있다.  
실행한 결과는 페이지 소스를 통해 볼 수 있다.

내용을 Hello,React!!로 바꾸어 저장하면,  
자동으로 reload되어 업데이트된 내용을 보여준다.  
반드시 리액트는 태그 안에 코드를 적어야만 한다.

<8> create-react-app에서 css 코딩하는 법

index.css부분을 수정해보자

사용자 청의 태그인 컴포넌트 App은 위에 적혀있는  
import App from ‘./App’으로부터 온 컴포넌트이다. 작은 따옴표 안에 적힌 것은 파일 명이며 .js가 생략되어 있다.  
from앞의 App은 컴포넌트 이름이므로 밑의 코드에서 사용되는 이름과 동일해야 한다.

App.js라고 파일 안의 App 컴포넌트가 로드될 때 같이 App.css도 로드가 되면서, 디자인을 같이 된다.

먼저 css를 다 지우고 시작하자

<9>deploy하는 법

reload를 할 때 새로고침버튼에서 Empty Cache and Hard Reload를 하면 캐시를 모두 없애 버린다. 아무것도 없는데 1.7MB인 이유는 개발자를 편하게 하기 위한 기능을 다 넣어두었기 때문이다.  
하지만 사용자에게 너무 무거운 파일을 배포할 수 없기 때문에,  
원래 npm run start였지만, npm run build라고 명령어를 치면  
새로운 build라는 디렉토리가 추가된다.

거기엔 우리가 알고 있는 파일들이 있지만, 코드를 확인해보면 공백이 하나도 없다.  
이미 가지고 있는 불필요한 정보들을 모두 살균시켰기에 그러한 상태이며, 그래서 용량이 작다.  
보안적으로, 또한 심미적으로 좋지 않은 부분을 없앤 후에 index.html을 알아서 처리해주는 것이다.

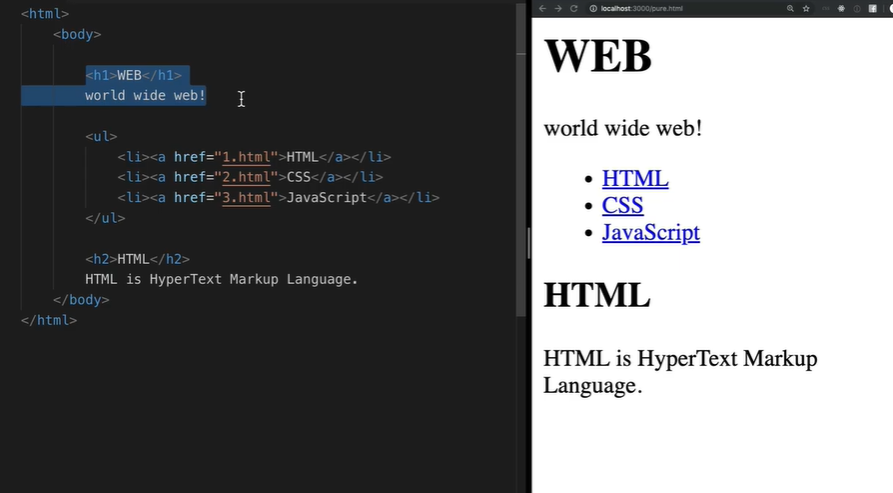
실제로 서비스할 때는 build안에 있는 파일들을 사용하면 되고,  
웹서버가 문서를 찾는 root디렉토리에 build 디렉토리에 있는 파일들을 위치시키면 실서버 환경을 완성시킬 수 있다.

npm install -g serve를 입력하면  
간단한 웹서버를 실행시킬 수 있다. (npx serve -s build) 를 입력하면 build를 document root로 하겠다는 의미이다.  
파일의 용량을 확인하면 확연히 줄어든 것을 볼 수 있다.  
개발환경의 앱은 무거운 녀석이지만, 배포된 것은 매우 가볍다.

<10> 리액트가 없다면?

대단원의 막이 시작된다 두둥 🡪 컴포넌트를 만드는 방법

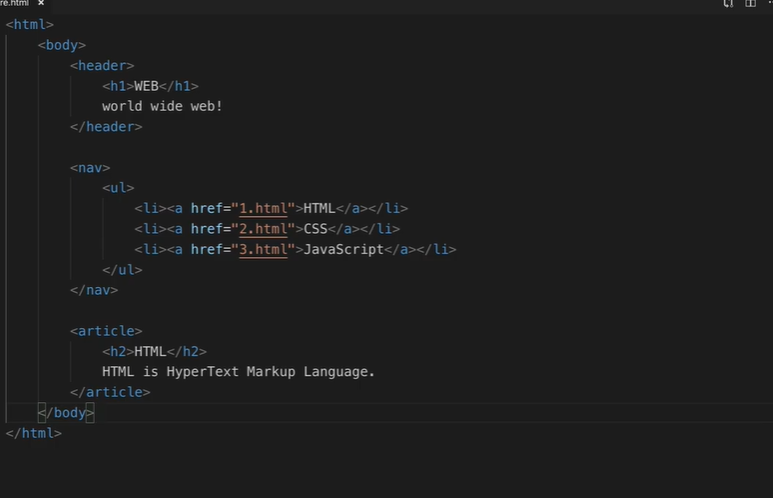
public이라는 디렉토리는 npm run start를 했을 때, 파일을 찾는 document root이다.

react가 아닌, 쓰기 전의 html코드를 짜보자  
더 복잡하게 보이기 위해 시멘틱 태그로 감싸보자 ^^  
기능상의 변화는 없지만, 구분을 위해 의미론적으로 명시하는 태그

하지만 만약 코드가 천만줄, 일억줄이라면 한눈에 코드가 들어올까?  
인지능력을 벗어나기에, 보기에 어렵다.  
그렇다면 그 복잡한 태그를 다른 곳에 숨겨두고, 이름만 적어두면 얼마나 좋을까?라는 생각을 하게 된다.

🡪이것을 바로 react가 해준다.

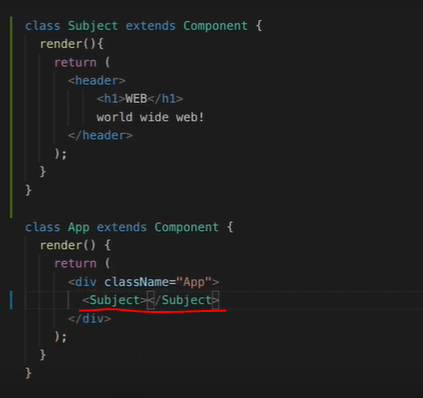
<11.1> 리액트  
순수한 pure.html을 리액트로 잘 정리정돈해보는 작업을 해볼 것이다.



한 쪽엔 src에 잇는 App.js를 한 쪽에 옮기자  
App.js에 있는 log뭐시기는 필요없으니 지우고 시작하자.

드디오 컴포넌트를 만들 것인데,  
객체지향을 이해하는 사람은 쉬울 것이다.  
Component라는 클래스를 상속받아 새로운 App이라는 클래스를 만들것이고,  
해당 클래스는 render()라는 함수를 가진다.

카피해서 똑 같은 틀은 만들어 보겠다.  
Subject라는 틀을 만들고 싶다!

반드시 render()라는 함수가 있어야 한다.  
다음에 render()안에 return()을 넣고, 그 안에 원래의 코드를 cut하여 붙여넣는다. (function은 생략한다)  
해당 컴포넌트는 최소 하나의 최상위 태그가 있어야하며, 지금의 경우 header가 해당된다.  
App class의 경우엔 div가 최상위태그라고 할 수 있다.

그 후에 Subject태그를 넣으면?

페이지 소스를 확인해보면 <header>태그가 추가된 것일 수도 있다.

웹브라우저는 리액트의 존재를 모르지만, 리액트가 최종적인 html코드를 전달해준다.

App.js는 자바스크립트 코드는 유사 자바스크립트이다.  
자바스크립트 문법이 아니다. 페이스북에서 만든 유사 자바스크립트 언어를 사용한다.  
알아서 리액트가 우리 대신에 자바스크립트 문법으로 바꾸어 주는 것이다.

<11.2> 컴포넌트 만들기 2

저번에 li태그였던 것들을 컴포넌트로 만들어 보자 (저번 최상위 태그가 <nav>였던 것)

1) TOC(Table of Contents)라는 컴포넌트로 만들어 보자

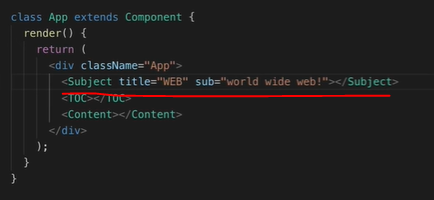
2) 그 밑의 <article>도 Content라는 컴포넌트로 만들어보자

만약 코드들이 일억줄이면? 정리정돈해서 보여주는 것이 훨씬 좋을 것이다.  
복잡도를 획기적으로 낮추고, 다른 것에 더 복잡도를 높일 수 있을 것이다.

<12> props

지난 시간까지는 컴포넌트들을 만들어 보았다.

li태그를 보면 a라는 것을 통해 태그라는 것을 알려준다. 동시에 어떤 주소랑 연결하는 가를 표현하기 위해 attribute라는 속성을 통해 연결되는 주소를 적는다.

그렇다면 언제나 똑같이 나오는 Subject라는 태그도 새로운 속성인 title을 넣을 수 있지 않을까?  
줄 친 것과 같이 말이다.

어떻게 하면 되는지는 react사이트 Main Concepts에 들어가면 된다.

즉 알아서 WEB, world wide web이라는 텍스트를 주고 싶다는 의미이다.

<h1>{this.props.title}</h1>  
{this.props.sub}  
리액트에서 attribute를 props라고 한다.  
이러한 방식으로 하면 전에는 정적이었던 것들을 이제 좀 더 자유롭게 사용할 수 있다.

<13> React Developer Tools

1) 설명서를 잘 보자  
2) 하지만 설명서는 가끔 빈약하고 복잡하다는 단점이 있다. 그러니 여러가지 도구를 이용하여 직접 공부를 하여 알아내야 한다.  
3) 다른 사람에게 질문하고, 검색해보자

현재 상태를 알 수 있는 tools이 있다.

React 공식 사이트 -> Community -> Tools -> Debugging -> 현재 리액트로 만들어진 앱의 상태를 확인 가능 -> Chrome을 클릭 -> 설치 후 브라우저 닫고 다시 열면? -> 주황색 아이콘 확인 가능

맨 마지막 React탭을 누르면? 컴포넌트 단위로 확인할 수 있다.  
그리고 오른쪽에서 props에서 속성들을 확인하고 변경 가능하다.

<14> Component 파일로 분리하기

현재 4개의 component들이 있지만, 천개가 있다고 생각해보자

그리고 App.js 안에 있다보니까 다른 파일에서는 사용할 수 없다.  
각 다른 파일로 분리해보자

1) src에 components라는 디렉토리를 만든다.

2) 디렉토리에 새로운 파일을 만들어 TOC.js라고 만든다.

3) TOC부분을 카피하여 붙여넣으면 빨간줄이 생긴다.  
위에 import React, {Component} from ‘react’;를 붙여넣는다.  
이것의 의미는 react라는 라이브러리에서 컴포넌트라는 클래스를 로딩하여 사용한다는 의미이다.  
앞에 React, 이것은 리액트로 코딩할 때, 꼭 필요한 것이다.

현재 TOC.js파일 안에는 TOC밖에 없지만 다른 여러 함수가 잇을 수 있다.  
그러므로 무엇을 밖에서 사용하게 허용할 것인지 정해야 하는데 이때,

export default TOC;

를 코드 마지막에 써주면 TOC클래스를 외부에서 가져다 쓸 수 있게된다.

그렇다면 index.js에서는  
원래 TOC클래스를 지워버리고 파일 위에  
import TOC from “./components/TOC”;를 넣어준다.

다른 클래스도 이와 같이 반복하면 외부로 컴포넌트를 파일로 분리할 수 있다.

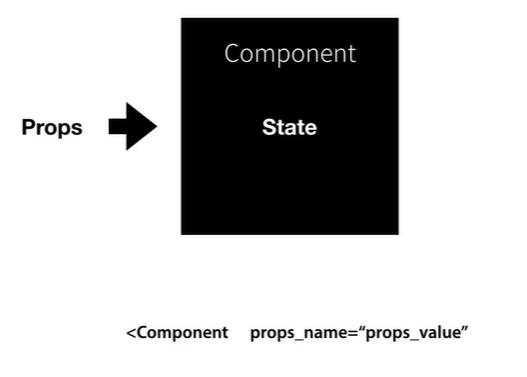
훨씬 더 코드가 간결해지며, 필요한 컴포넌트를 빨리 찾을 수 있으며, App.js말고 다른 파일에서도 사용가능하다.

<15.1> State소개

state는 props와 다른 점?

어떤 제품이 있을 때, 사용자의 UI들이 리액트의 관점에서는 props이다.  
그렇다면 개발자는? 내부적인 구현을 위해 다양한 상태와 조작장치들을 가지고 있는데, 이것을 state라고 할 수 있다.

props는 사용자 입장, state는 props의 값에 따라 내부 구현에 필요한 것이라고 얘기할 수 있다.

컴포넌트가 생긴대로 사용하면 재미없을 것이다. 사용자는 props를 통해 컴포넌트를 마음대로 조작할 수 있다.  
하지만 사용자에겐 필요없고, 알아서는 안되는 것은 state라고 한다.

내부 state는 철저히 분리되어있다.

<15.2> state 사용

App.js를 보면, App컴포넌트 안에 Subject라는 하위 컴포넌트가 존재한다.

Subject 안에 있는 props의 값이 하드코딩되어있으니 보기 싫다. state로 만들고, 그 state값을 subject에 props로 전달하자.

효과는 나중에 보여줄 것이다.  
(이래서 state가 중요하구나)

이해하지 말고, 그러려니 해라

state값을 초기화 하기 위함, 그 초기값으로 subject안에 들어갈 값을 세팅하기 위해

constructor를 저렇게 짜고, 안에 다시 적어준다.

컴포넌트가 실행될 때, constructor가 함수가 있다면, 제일 먼저 실행되어 초기화를 담당.  
초기화가 끝나면 this.state={subject:{title:’WEB’, sub:’world wide web!’}

리액트에서 따옴표 안에 넣어버리면 텍스트로 인식, 자바스크립트 코드로 실행시키고 싶으면 중괄호인 {}를 사용해야 한다.

title={this.state.subject.title}

이전과 똑같지만 데이터를 constructor에서 가져온다는 것으로 바뀌었다.

현재 index.js에서 App.js의 App컴포넌트를 실행하는데, state값에 subject가 있는지 없는지 알 수 없다. 철저하게 은닉한다 -> 좋은 사용성

내부적으로 사용할 데이터는 state를 사용

그렇게 만들어진 state값을 Subject의 props로 준 것인데,  
이것의 의미 ? 상위 컴포넌트인 App의 상태를 하위 컴포넌트로 전달하고 싶을 때, state를 사용해서  
state -> 하위의 props로 전달하는 것 가능

<15.3>Key

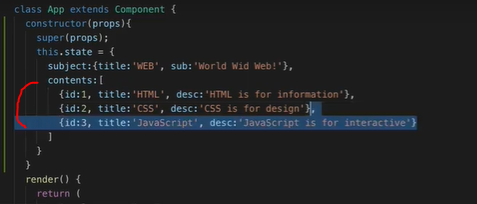
TOC는 li를 통해 이루어져있는데, 수정할 때마다 데이터를 추가할 수는 없으니,

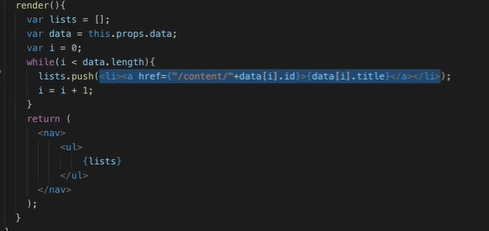
contents: [ (배열)

{id:1, title:’HTML’, desc : ‘HTML is for information’},

…

이것을 TOC에 주입하고 싶다?  
<TOC data={this.state.contents}></TOC>

TOC는 내부적으로 어떤 값을 가지는가?



헌데 이렇게 하면 오류가 나기에, key를 주어야 한다.

react에서 내부적으로 필요하기에 요청하는 것이기에 넣어주기만 하면 된다.

TOC의 data라고하는 props에서는 어떤 유형의 데이터를 전달하는지만 알려주면 된다.