**동적인 값 관리**

-리액트 16.8 버전부터 함수형 컴포넌트에서도 Hooks를 통해 상태를 관리할 수 있게 됨

**버튼을 누르면 숫자가 바뀌는 counter 컴포넌트 생성**

//Counter.js

import React, {useState} from ‘react’; //useState를 쓰기위한 import

function Counter() {

const [number, setNumber] = useState(0); //0을 인자로 넘겨받고, 변수와 함수를 배열로 반환

const onIncrease = () => {

setNumber(number +1); //set함수 사용

}

const onDecrease = () => {

setNumber(number-1);

}

return (

<div>

<h1>{number}</h1>

<button onClick={onIncrease}>+1<button> //이벤트={함수}

<button onClick={onDecrease>-1</button>

</div>

);

}

export default Counter;

//App.js

return (

<Counter />

);

**설명**

**-**useState를 사용하려면 밑처럼, import 해줘야함

import React, {useState} from ‘react’;

-useState()는 들어간 인자가 기본 값으로 들어가고, 그 값과 설정형 함수를 배열 형태로 반환함.

//배열 비구조화 할당

const [number, setNumber] = useState(0);

//그냥 배열로 받을 때

const numverState = useState(0);

const number = numberState[0];

const setNumber = numberState[1];

-마우스 클릭 이벤트 발생 시 적용할 함수를 만듬

const onIncrease = () => {

setNumber(number + 1);

}

-마우스 클릭 이벤트 적용

<button onClick={onIncrease}>+1</button> //onClick={onIncrease()} 는 안됨. () 빼자

**함수형 업데이트**

**-**setter 함수를 사용할 때, 다음처럼 기존 값을 어떻게 업데이트 할지에 대한 함수를 등록할 수 있음

const [number, setNumber] = useState(0);

const onIncrease = () => {

setNumber(prevNumber => prevNumber+1); //변수 이름이 중요한 게 아니고

} //그냥 이전값 => 이후값으로 표현하면 됨

-그리고 이게 더 최적화적임.(이유는 나중에 설명)