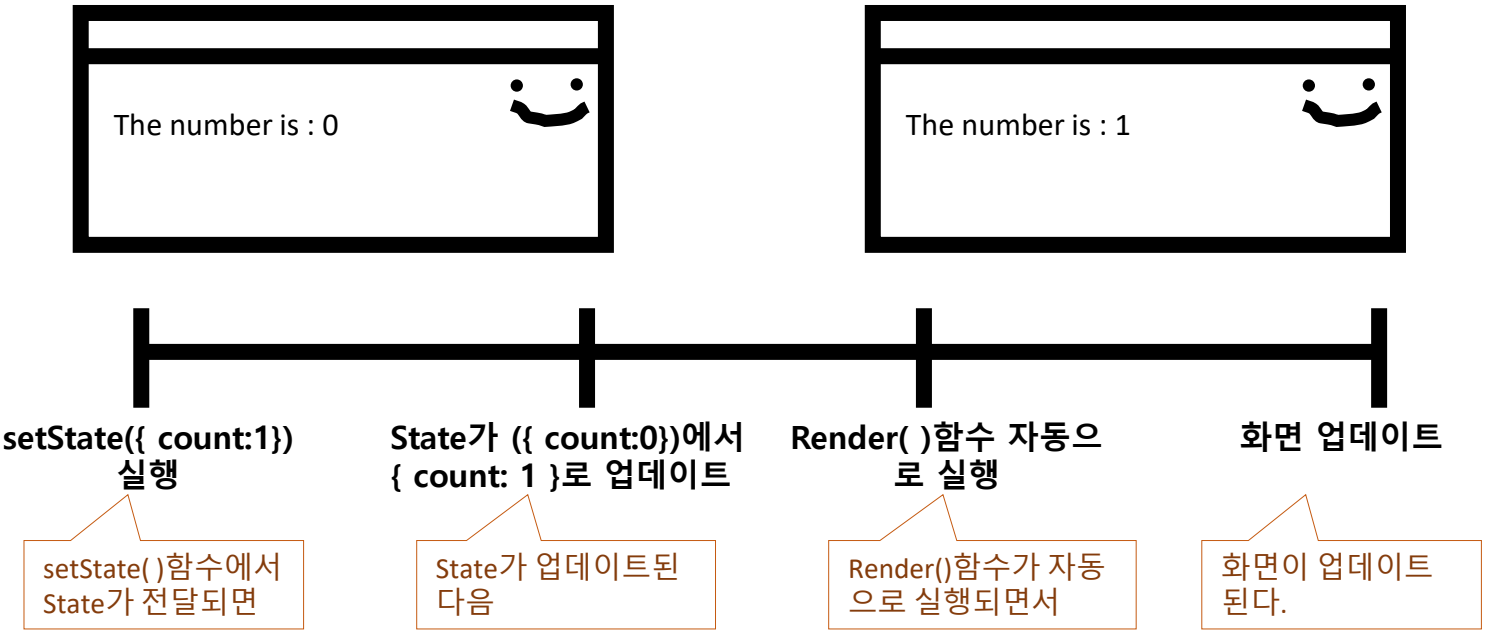
5강
useState() 생명 주기

React State

- prop(properties(속성)의 줄임 말)와 state는 일반 자바스크립트 객체다. 두 객체 모두 렌더링 결과물에 영향을 주는 정보를 갖고 있는데 한 가지 중요한 방식에서 차이가 있다. **props는 (함수 매개변수처럼)** 컴포넌트에 전달되는 반면 **state는 (함수 내 선언된 변수처럼) 컴포넌트 안에서 관리한다**. React에서 this.props와 this.state는 모두 렌더링 된 값을 나타낸다. 다시 말해 현재 화면에 보이는 걸 말한다.
- **State는 동적 데이터를 다룰 때 사용된다**. 동적이란 말 그대로 변경될 가능성이 있는 데이터를 말한다. 객체를 예로 들면 객체의 구성 요소 중 일부가 있다가 없을 수도 있고, 구성 요소가 하나였다가 둘이 될 수도 있고, props는 이런 데이터를 다룰 수 가 없기 때문에 state를 사용한다.

React setState( )함수로 count state 변경하기

- 리액트가 state( )함수의 호출을 감시하고, setState( )함수가 동작하면 state가 새로운 값으로 바뀌고, 이어서 render( ) 함수를 동작시켜 화면을 업데이트 시킨다.

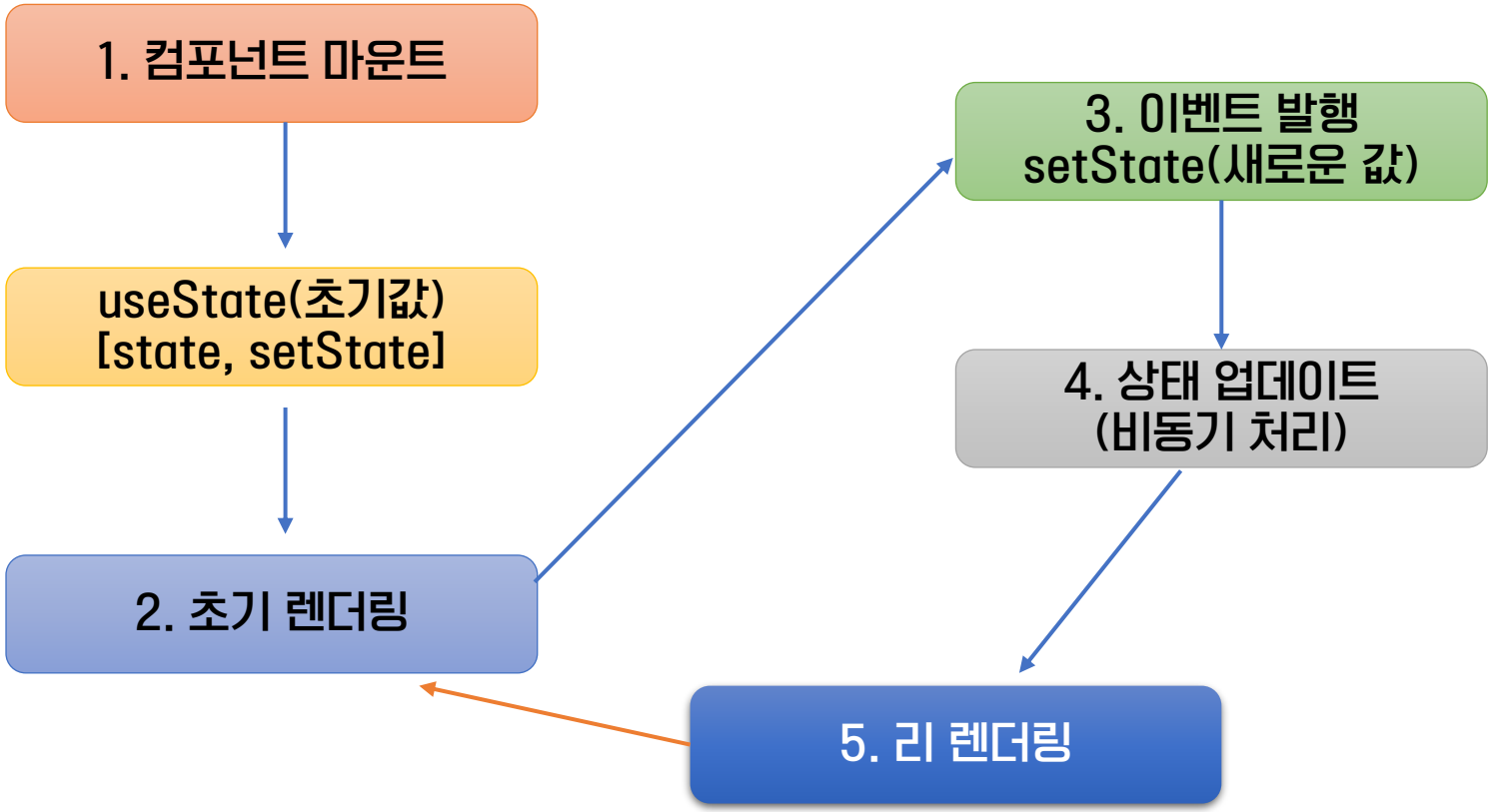


- 단, setState( )의 인자로 state를 전달하면 이전 state가 완전히 새로운 state로 교체되는 것이 아니라, 이전 state와 새로운 state를 비교하여 변경된 데이터만 업데이트됨을 주의 하자!!

React 컴포넌트 생명주기 함수

- 생명주기란 앱이 실행되고 종료되는 과정을 특정 시점 별로 나눠 둔 것을 말한다.
- React의 생명주기는 **컴포넌트가 이벤트를 다룰 수 있는 특정 시점**을 말하며 마운트, 업데이트, 언마운트 상태로 구성되어 있다.
  - 컴포넌트가 실제 DOM에 삽입되는 것을 **마운트**
  - 컴포넌트가 변하는 것을 **업데이트**
  - 컴포넌트가 DOM 상에서 제거되는 것을 **언마운트**

useState() 생명주기(Lifecycle)

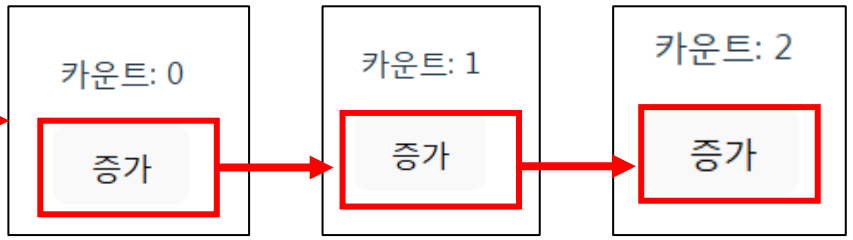


useState 간단한 예제

```jsx
import React,{useState} from 'react';


export default function Counter01() {
  const [count, setCount] = useState(0); // 초기값 0


  return (
    <div>
      <p>카운트: {count}</p>
      <button onClick={() => setCount(count + 1)}>증가</button>
    </div>
  );
}
```



반환값 설명

state : 현재 상태 값

setState : 상태를 바꾸는 함수. 호출하면 React가 그 컴포넌트를 다시 렌더링함.

setState에 넘길 수 있는 것:

새로운 값 직접: setCount(3)

함수형 업데이트 (이전 값을 기반으로 갱신할 때 권장): setCount(prev => prev + 1)

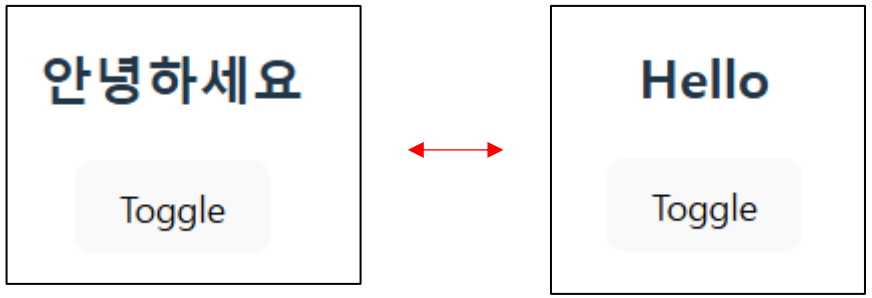
기초 연습 문제

문제1] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 클릭할 때마다 "안녕하세요" ↔ "Hello" 번갈아 표시하도록 작성하시오.
② 함수이름은 Exstate01()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex01.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex01.jsx를 import하여 실행 하시오.

<출력 결과>

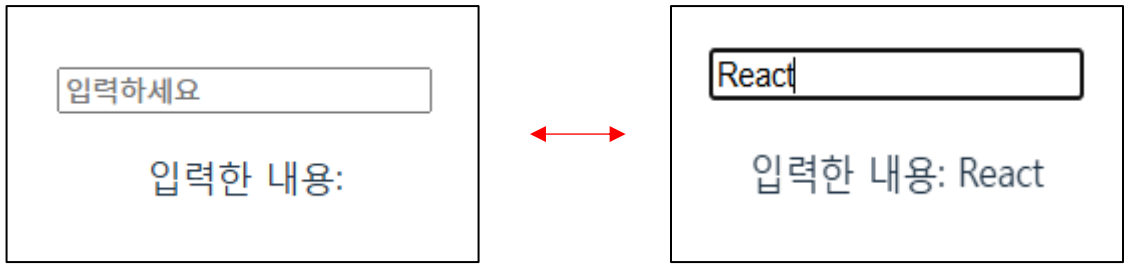


문제2] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 입력한 텍스트를 아래 문장에 실시간 표시하도록 작성하시오.
② 함수이름은 Exstate02()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex02.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex02.jsx를 import하여 실행 하시오.

<출력 결과>

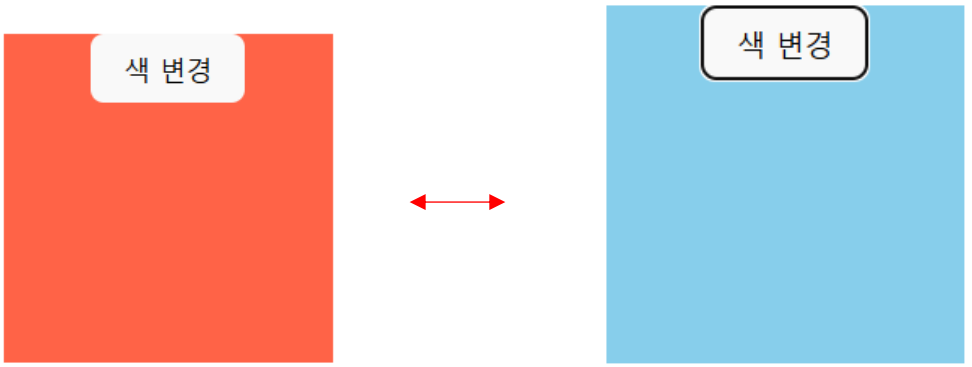


문제3] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 버튼 클릭 시 배경색이 바뀌는 박스 만들기 작성하시오.
② 함수이름은 Exstate03()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex03.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex03.jsx를 import하여 실행하시오.

<출력 결과>

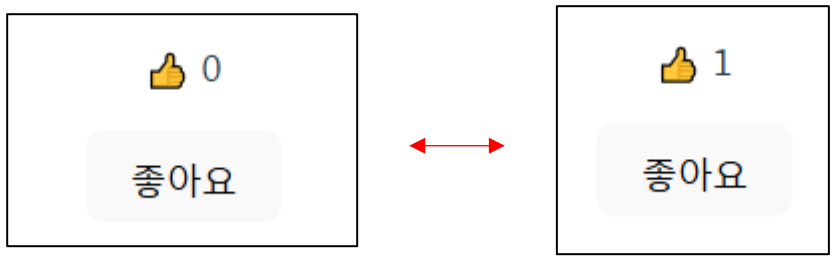


문제4] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 버튼 클릭 시 "좋아요 👍" 카운트 올리기를 작성하시오. (이모지 아이콘 이용)
② 함수이름은 Exstate04()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex04.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex04.jsx를 import하여 작성하시오

<출력 결과>

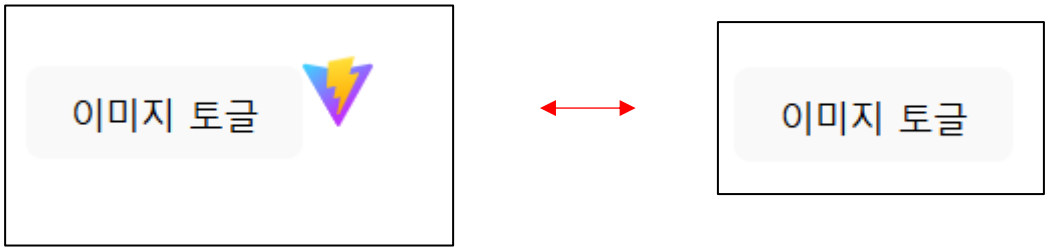


문제5] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 이미지 보이기 / 숨기기 토글 버튼 만들기 표시하도록 작성하시오.
② 함수이름은 Exstate05()로 작성하시오.
③ Public -> vite.svg 이미지 이용해 작성하시오.
④ useState() 를 이용해 작성하시오.
⑤ src 폴더안에 -> stateComponent 폴더 생성 -> Ex05.jsx 파일을 생성하여 작성하시오.
⑥ src 폴더안에 -> App.jsx에 작성한 Ex05.jsx를 import하여 작성하시오

<출력 결과>

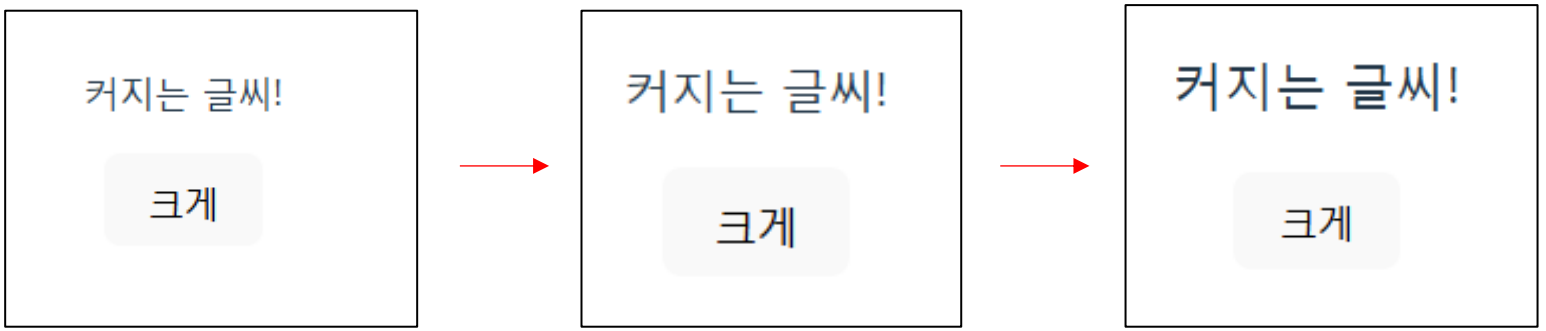


문제6] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 버튼 클릭 시 글자 크기가 커지는 텍스트 만들기작성하시오.
② 함수이름은 Exstate06()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex06.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex06.jsx를 import하여 작성하시오

<출력 결과>

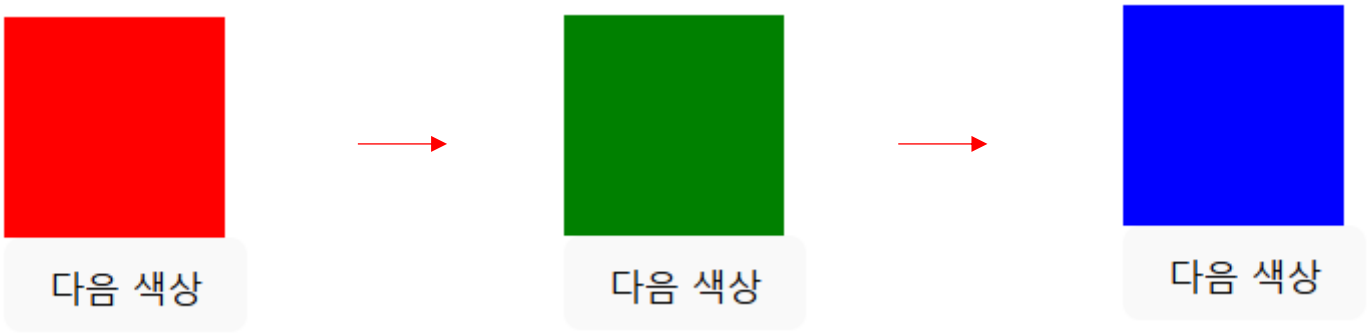


문제7] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 버튼 클릭 시 세 가지 색상이 순환하는 박스 (red → green → blue)를 작성하시오.
② 함수이름은 Exstate07()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex07.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex07.jsx를 import하여 작성하시오

<출력 결과>

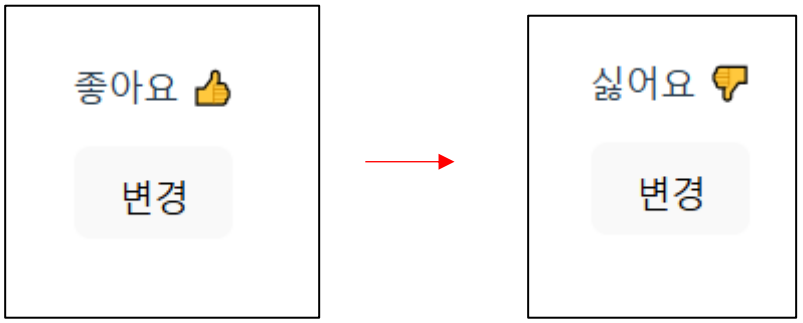


문제8] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 클릭 시 좋아요 / 싫어요(이모지 아이콘 사용) 상태 변경되도록 작성하시오.
② 함수이름은 Exstate08()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex08.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex08.jsx를 import하여 작성하시오

<출력 결과>

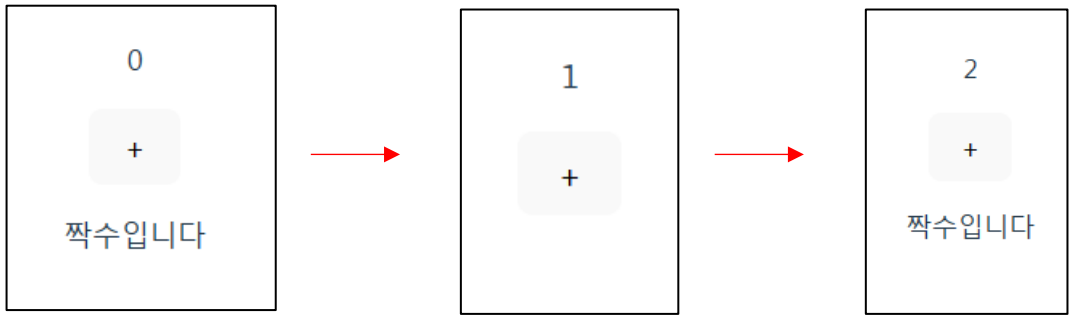


문제9] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 숫자가 짝수일 때만 "짝수입니다" 표시되도록 작성하시오.
② 함수이름은 Exstate09()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex09.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex09.jsx를 import하여 작성하시오

<출력 결과>

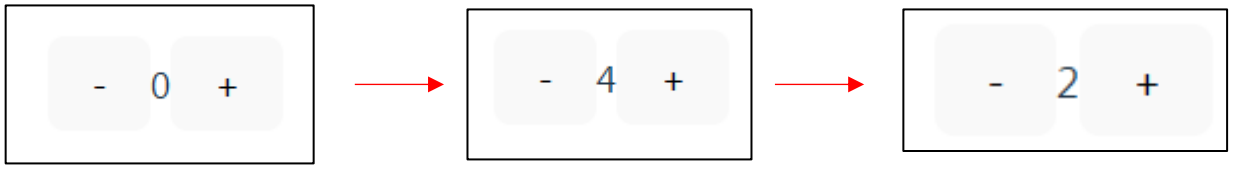


문제10] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 두 개의 버튼으로 증가(+) / 감소(-) 동작 구현되도록 작성하시오.
② 함수이름은 Exstate10()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex10.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex10.jsx를 import하여 작성하시오

<출력 결과>



문제11] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 색상 입력창에 입력한 색으로 배경 변경되도록 작성하시오.
② 함수이름은 Exstate11()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex11.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex11.jsx를 import하여 작성하시오

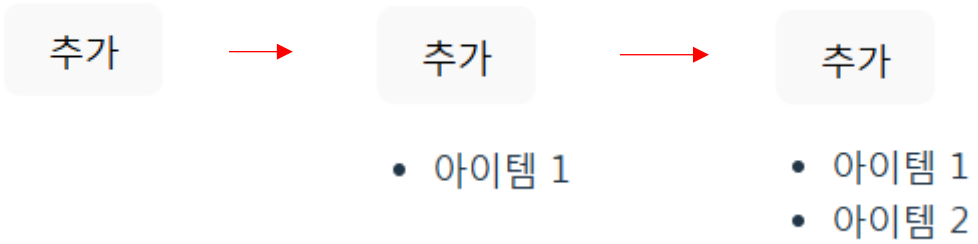
<출력 결과>

문제12] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 버튼 클릭 시 배열에 값 추가되도록 작성하시오.
② 함수이름은 Exstate12()로 작성하시오.
③ Map()함수 이용해 작성하기
④ 얕은 복사 이용해 작성하기
⑤ useState() 를 이용해 작성하시오.
⑥ src 폴더안에 -> stateComponent 폴더 생성 -> Ex12.jsx 파일을 생성하여 작성하시오.
⑦ src 폴더안에 -> App.jsx에 작성한 Ex12.jsx를 import하여 작성하시오

<출력 결과>



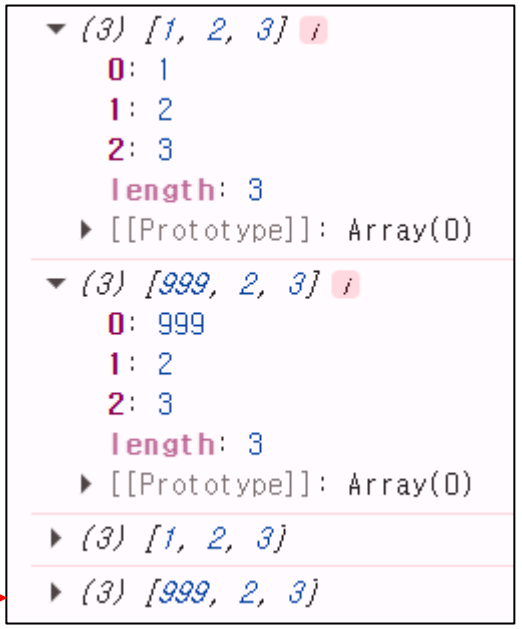
문제12] 얕은 복사 (Shallow Copy) 란?

- 얕은 복사란 객체나 배열을 겉부분만 복사하고, 내부의 중첩된 객체나 배열은 원본과 같은 참조(주소)를 공유하는 복사 방법입니다.

- 객체나 배열을 새로운 메모리 공간에 복사하는 것
- 참조(reference)를 공유하지 않고 최상위 속성만 복사

```
export default function Exstate13() {
  const original = [1, 2, 3];
  const shallow = [...original]; // ← 이게 얕은 복사!

  // 1차원 배열은 문제없음
  shallow[0] = 999;
  console.log(original); // [1, 2, 3] - 원본 안전!
  console.log(shallow); // [999, 2, 3]
}
```



➢ React에서 state를 직접 수정하지 않고 얕은 복사를 사용하는 이유

1. React는 상태 변화를 감지해야 재렌더링을 함

- React는 state나 props의 주소(참조값)가 바뀌었는지 비교해서 변경 여부를 판단한다.
- 만약 기존 객체/배열을 그대로 수정하면 참조가 바뀌지 않아서 React가 변화를 감지하지 못함.

2. 불변성(Immutable) 유지

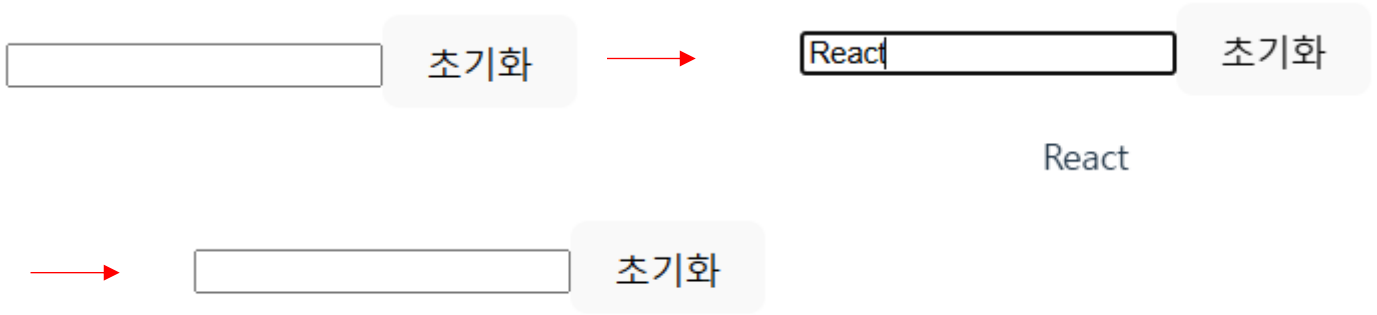
- 원본 state를 직접 수정하면 다른 컴포넌트나 이전 상태를 참조하는 로직이 예상치 못한 버그를 발생시킬 수 있음.
- 얕은 복사로 새 객체를 만들고 변경하면 안전하게 상태를 관리할 수 있음.

문제13] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 입력값을 초기화하는 Reset 버튼 만들기를 작성하시오.
② 함수이름은 Exstate13()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex13.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex13.jsx를 import하여 작성하시오

<출력 결과>

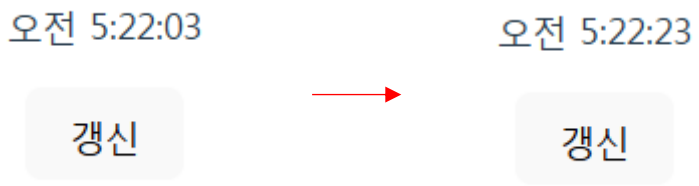


문제14] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 클릭할 때마다 "현재 시간"을 갱신하는 버튼을 작성하시오.
② 함수이름은 Exstate14()로 작성하시오.
③ 단, new Date().toLocaleTimeString() 함수를 이용해 작성하시오.
④ useState() 를 이용해 작성하시오.
⑤ src 폴더안에 -> stateComponent 폴더 생성 -> Ex14.jsx 파일을 생성하여 작성하시오.
⑥ src 폴더안에 -> App.jsx에 작성한 Ex14.jsx를 import하여 작성하시오

<출력 결과>



문제15] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 아래의 여러 개의 state를 동시에 변경하도록 작성하시오.

```
const [name, setName] = useState("홍길동");
const [age, setAge] = useState(20);
```

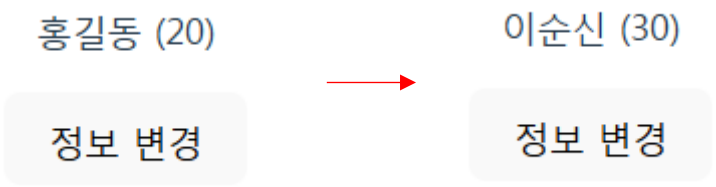
② 함수이름은 Exstate15()로 작성하시오.

③ useState() 를 이용해 작성하시오.

④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex15.jsx 파일을 생성하여 작성하시오.

⑤ src 폴더안에 -> App.jsx에 작성한 Ex15.jsx를 import하여 작성하시오

<출력 결과>



문제16] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① state에 아래의 객체를 저장하고 특정 속성만 업데이트되도록 작성하시오.

```
const [user, setUser] = useState({ name: "철수", age: 25 });
```

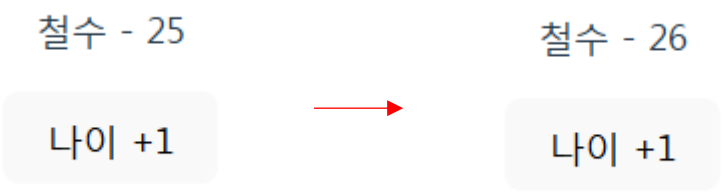
② 함수이름은 Exstate16()로 작성하시오.

③ useState() 를 이용해 작성하시오.

④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex16.jsx 파일을 생성하여 작성하시오.

⑤ src 폴더안에 -> App.jsx에 작성한 Ex16.jsx를 import하여 작성하시오

<출력 결과>

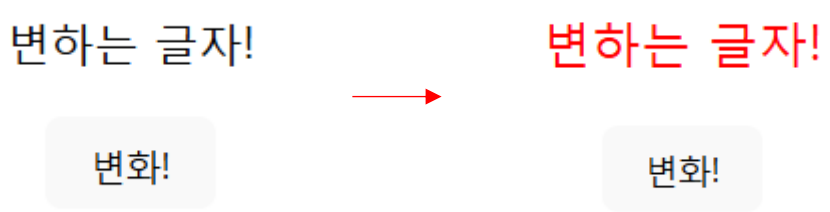


문제17] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 클릭할 때마다 글자 크기와 색이 모두 바뀌는 애니메이션 효과를 작성하시오.
② 함수이름은 Exstate17()로 작성하시오.
③ useState() 를 이용해 작성하시오.
④ src 폴더안에 -> stateComponent 폴더 생성 -> Ex17.jsx 파일을 생성하여 작성하시오.
⑤ src 폴더안에 -> App.jsx에 작성한 Ex17.jsx를 import하여 작성하시오

<출력 결과>



문제18] 다음의 조건에 만족하도록 React를 작성 하시오.

조건

① 자식에서 부모의 상태 변경 함수 호출하여 [+1]버튼 클릭 시 숫자 증가하도록 작성하시오.
② 함수이름은 Exstate18()로 작성하시오.
③ 자식 함수 이름은 Child18()로 작성하시오.
④ props를 이용해 작성하시오.
⑤ useState() 를 이용해 작성하시오.
⑥ src 폴더안에 -> stateComponent 폴더 생성 -> Ex18.jsx 파일을 생성하여 작성하시오.
⑦ src 폴더안에 -> App.jsx에 작성한 Ex18.jsx를 import하여 작성하시오

<출력 결과>