8253jang@daum.net
장희정

React의 State란?

컴포넌트 내부에서 관리되는 동적인 데이터이다.
컴포넌트의 상태를 나타내며, 사용자 입력 등에 따라 변경될 수 있다.
state가 변경되면 리액트는 자동으로 해당 컴포넌트와 자식 컴포넌트를 다시 렌더링 한다.
useState() 훅를 통해 state 상태를 관리할 수 있다.

리액트 훅(Hooks)
함수형 컴포넌트에서 상태와 생명주기 기능을 사용할 수 있도록 해주는 기능으로 다양한 종류의 훅이 있다. 대표적으로 useState, useEffect, useContext, useReducer, useCallback, useMemo, useRef 등이 있다.

State 선언

state 변수와, state 값을 변경할 함수 이름을 함께 선언하며, 우변에는 초기값을 useState() 의 괄호 안에 작성한다. - useState 사용을 위해 useState 모듈 추가

```
importReact, { useState } from'react';
```

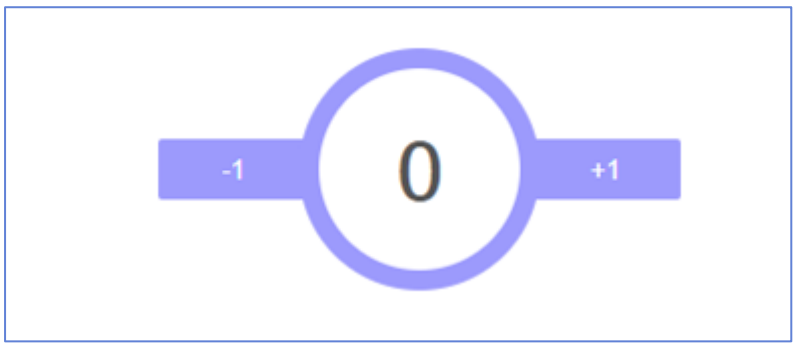
[사용 법]

```
const [상태 값 저장 변수, 상태 값 갱신함수] = useState(상태 초기 값);
```

[예제]

```
const [no, setNo] = useState(0);
```

useState 실습



8253jang@daum.net
장희정

```jsx
import './EX01_Count.css';
const Ex01_Count = () => {
    const [no, setNo] = useState(0); //0은 no초기값 /   setNo는 값변경할때 사용하는 함수
    let i=0;
    const minusFn = ()=>{
      setNo(no-1);
      i--;
    }

    const plusFn = ()=>{
        setNo(no+1);
        i--;
    }

    return (
        <div className='countStyle'>
            <button onClick={minusFn}>빼기</button>
            <span>{no}/{i}</span>
            <button onClick={plusFn}>더하기</button>
        </div>
    );
};

export default Ex01_Count;
```

State 사용하기 – Form01

Form1.jsx 생성 및 input 태그 작성

```jsx
const EX02_Form1 = () => {
    const inputName =(e)=>{
        console.log(e.target.value)
    }
    return (
        <div>
            <h2>폼 입력값 state Test</h2>
            이름 :<input type="text" onChange={inputName}/>
        </div>
    );
};

export default EX02_Form1;
```

name이라는 state를 정의하여 입력값을 담기

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

const EX02_Form1 = () => {
    const [name,setName] = useState("");

    const inputName =(e)=>{
        console.log(e.target.value)
        setName(e.target.value);
    }
    return (
        <div>
            <h2>폼 입력값 state Test</h2>
            이름 :<input type="text" value={name} onChange={inputName}/><br/>

        </div>
    );
};
```

나이 입력 폼 추가해보자.

```jsx
const EX02_Form1 = () => {
    const [name,setName] = useState("");
    const [age,setAge] = useState(0);

    const inputName =(e)=>{
        console.log(e.target.value)
        setName(e.target.value);
    }
    return (
        <div>
            <h2>폼 입력값 state Test</h2>
            이름 :<input type="text" value={name} onChange={inputName}/><br/>
            나이 :<input type="text" value={age} onChange={(e)=>setAge(e.target.value)}/>
        </div>
    );
};
```

State 사용하기 – Form02

회원정보를 입력 받는 폼을 정의하고 입력된 정보를 profile이라는 하나의 object로 state 지정

Form2.jsx 생성 및 input 태그 작성 이름, 나이, 이메일에 각각 입력을 하면 inputUpdate 함수 호출 inputUpdate 함수는 입력한 값의 name, value를 콘솔에 출력한다.

8253jang@daum.net
장희정

객체 비구조화 할당 (Object Destructuring Assignment)

const { name, value } = e.target;

e.target에 저장된 name, value 속성에 담긴 값을 비구조화 할당 방식을 통해 추출

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

const Ex03_Form2 = () => {
    const [profile, setProfile]= useState({
        name:"",
        age:"",
        email:"",
    })

    const inputUpdate =(e)=>{
        const {name, value} = e.target;
        console.log(name, value);
    }
    return (
        <div>
            <h2>폼 입력값 state Test - Profile</h2>
            이름: <input type="text" name="name" value={profile.name} onChange={inputUpdate} />
            <br></br>
            나이: <input type="text" name="age" value={profile.age} onChange={inputUpdate} />
            <br></br>
            이메일: <input type="text" name="email" value={profile.email} onChange={inputUpdate} />
        </div>
    );
};

export default Ex03_Form2;
```

state에 Object 저장

구조분해할당

state 값을 쓰지 않았기 때문에 input에 값을 입력해도 화면에 입력 값이 보이지는 않는다.

State에 값 담기

inputUpdate 함수에 profile state 값을 담기 위한 코드 추가한다.

전개연산자(spread operator)를 사용하여 값을 담는다.

전개 연산자(spread operator)

전개 연산자는 배열이나 객체의 요소를 개별요소로 분리할 때 사용하는 연산자이다.

객체 값을 복사할 때 활용한다.

```
setProfile({
    ...profile,
    [name]: value,
});
```

8253jang@daum.net
장희정

...profile : 기존 profile에 담긴값을 그대로 가져온다.

[name] : value, : 새롭게 추가된 값은 바꿔준다.

[ ]를 사용하지 않으면 문자열 리터럴 'name'으로 취급되어 name이라는 속성이 추가된다.

[ ]사용하면 name변수의 값이 속성이름이 된다. 즉 동적으로 속성이름을 결정할 수 있다.

```
11      const inputUpdate =(e)=>{
12        const {name, value} = e.target;
13        console.log(name, value);
14
15        setProfile({
16         ...profile,
17         [name] : value,
18        })
19      }
20      return (
21          <div>
22            <h2>폼 입력값 state Test - Profile</h2>
23             이름: <input type="text" name="name" value={profile.name} onChange={inputUpdate} />
24             <br></br>
25             나이: <input type="text" name="age" value={profile.age} onChange={inputUpdate} />
26             <br></br>
27             이메일: <input type="text" name="email" value={profile.email} onChange={inputUpdate} />
28          </div>
29      );
30  };
```

조건부 랜더링

state 값에 따라 화면에 보여줄 요소를 다르게 하는 것

삼항 연산자를 활용하는 것이 일반적이다.

삼항연산자

삼항연산자는 조건에 따라 실행할 내용을 구분할 수 있도록 하는 연산자로서 if, else 와 비슷한 기능을 한다. 표현이 간결하기 때문에 조건부 렌더링에 많이 활용된다.

```
[조건] ? 실행블록1(참인경우) : 실행블록2(거짓인경우)
```

- condition 변수 값에 따라 참 또는 거짓 출력

```
function ConditionalRendering1() {
  const condition = true;
  return (
    <>
      <h2>ConditionalRendering1.jsx</h2>
      {condition ? <h2>참</h2> : <h2>거짓</h2>}
    </>
  );
}
export default ConditionalRendering1;
```

조건부 렌더링 예제

조건을 따질 변수를 state로 지정하여 값이 바뀜에 따라 화면에 보여지는 내용을 바꾼다.

버튼을 클릭할 때마다 isLogin의 반대 값을 isLogin 값으로 저장한다.

로그인상태(isLogin=true ⇒ Logout), 로그아웃상태(isLogin=false ⇒ Login) 출력한다.

```
2  import { useState } from 'react';
3
4  const Ex04_ConditionRendering = () => {
5      const [isLogin, setIsLogin] = useState(false);
6      return (
7          <div>
8              <h3>ConditionRendering Test</h3>
9              <button onClick={()=>setIsLogin(!isLogin)} >
10                 {isLogin? "Logout" : "Login"}
11             </button>
12         </div>
13     );
14 };
15
16 export default Ex04_ConditionRendering;
```