

REACT #5

2023년 1학기

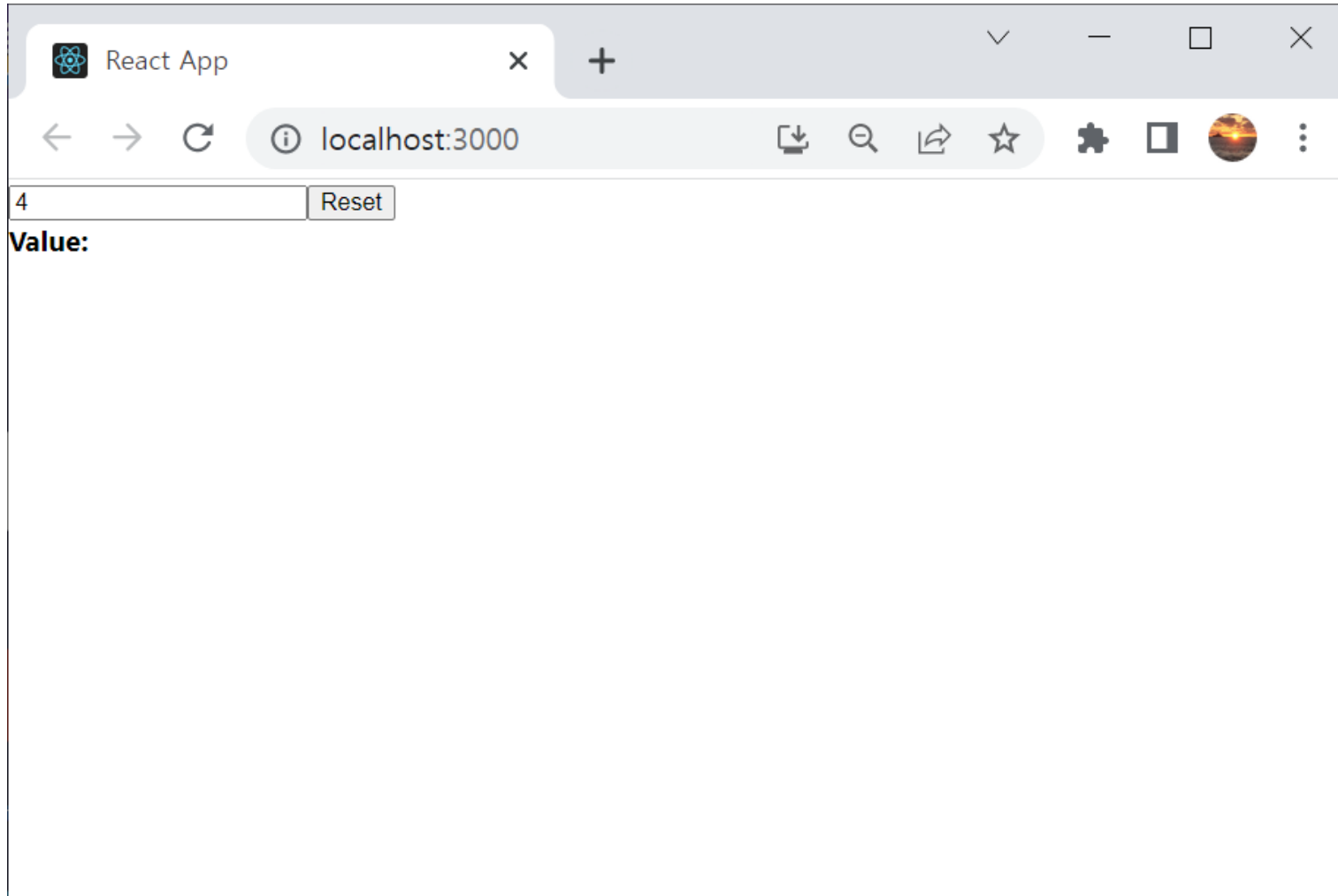
소프트웨어융합대학
고영웅 교수

목차

- input 관리하기
- useRef 이해하기
- map 사용하기
- key 사용하기
- useEffect 사용하기

출처 : <https://react.vlpt.us/>

입력화면을 만들자

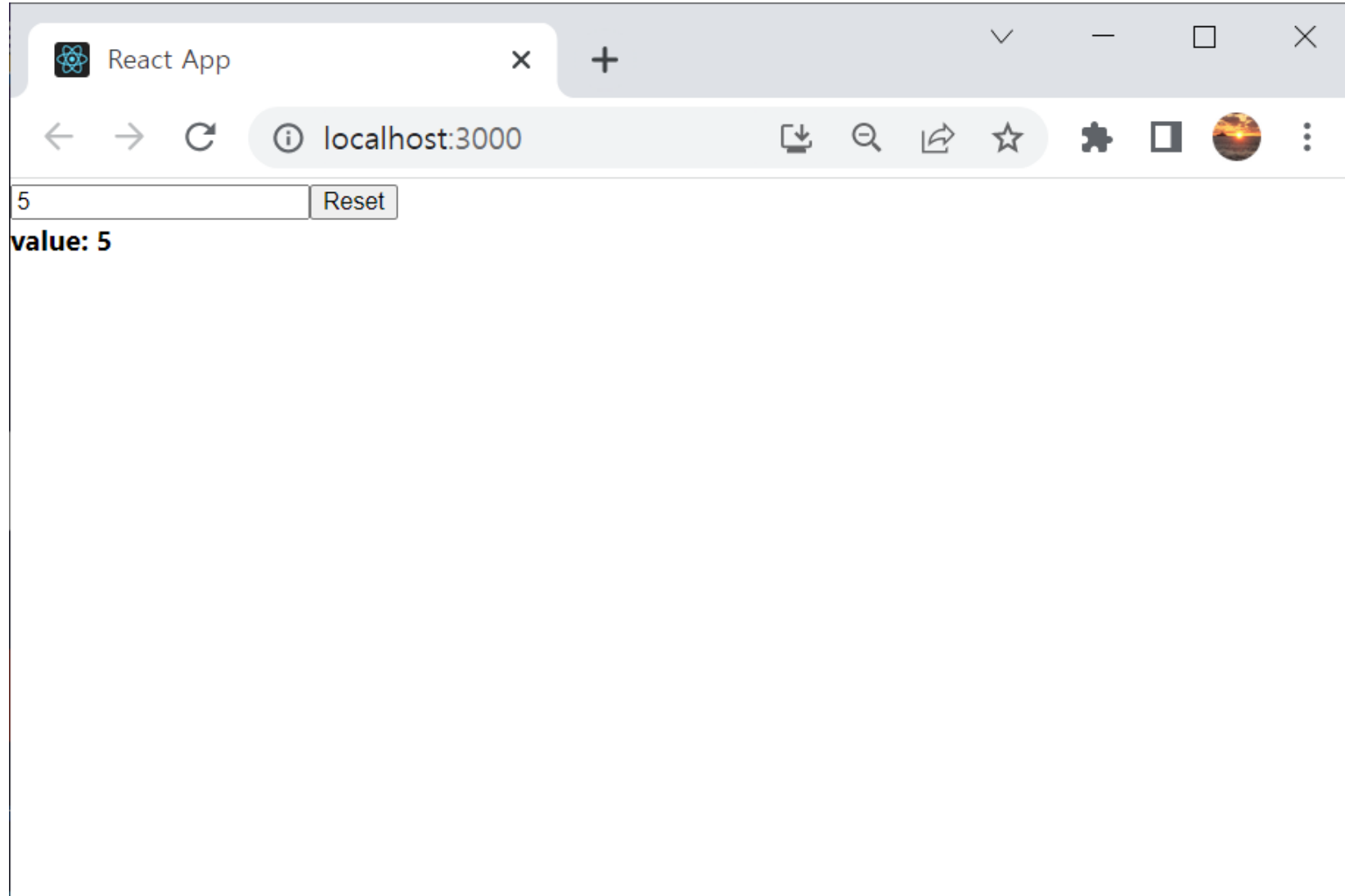


src > JS InputSample.js > [🔗] default

```
1  import React from 'react';
2
3  function InputSample() {
4    return (
5      <div>
6        <input />
7        <button>Reset</button>
8        <div>
9          <b>Value: </b>
10       </div>
11     </div>
12   );
13 }
14
15 export default InputSample;
```

src > JS App.js > [⌘] default

```
1  import React from 'react';
2  import InputSample from './InputSample';
3
4  function App() {
5    return (
6      <InputSample />
7    );
8  }
9
10 export default App;
```

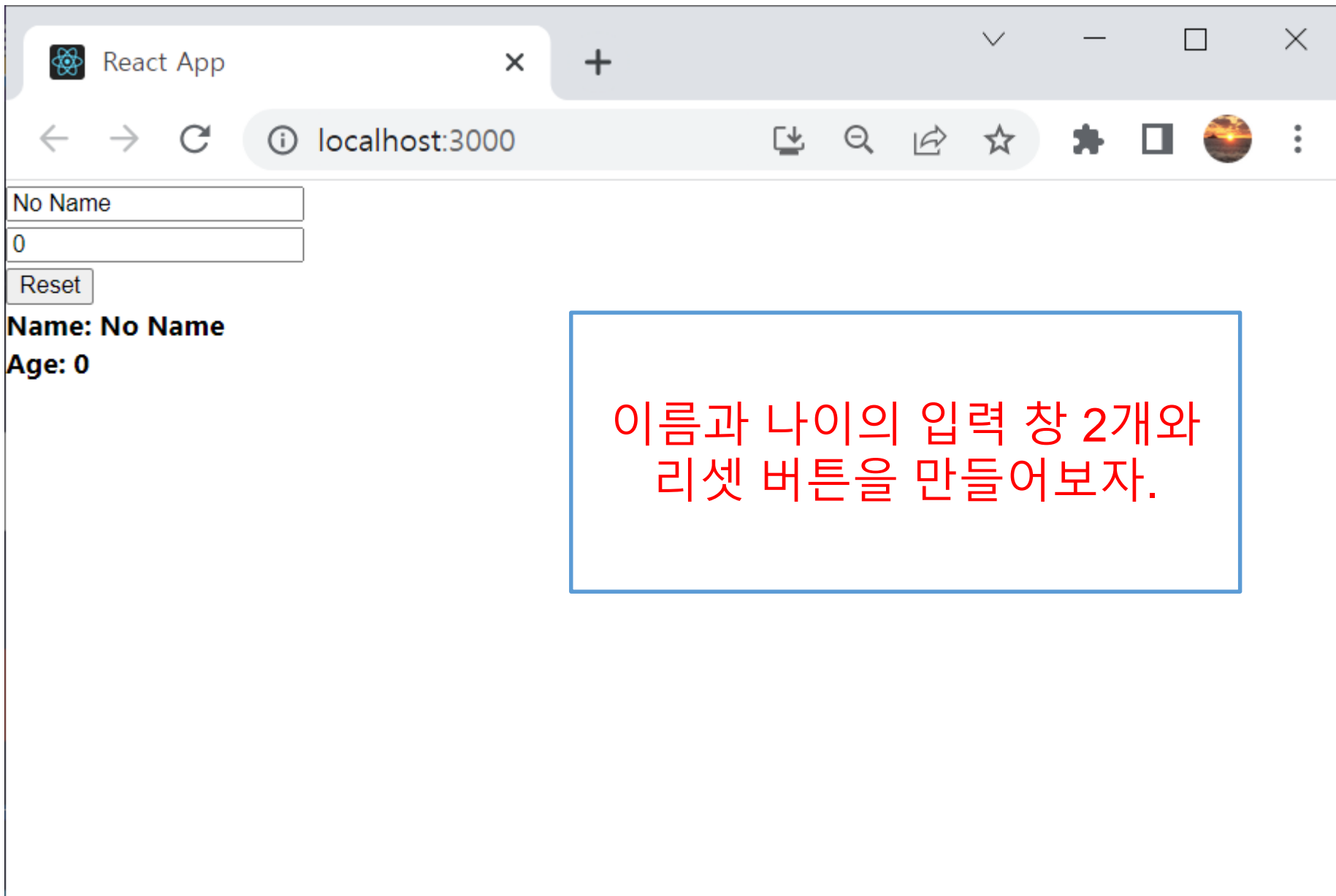


src > JS InputSample.js > ...

```
1  import React, { useState } from 'react';
2
3  function InputSample() {
4    const [text, setText] = useState('');
5
6    const onChange = (e) => {
7      setText(e.target.value);
8    };
9
10   const onReset = () => {
11     setText('');
12   };
13
```

```
    return (
      <div>
        <input onChange={onChange} value={text} />
        <button onClick={onReset}>Reset</button>
        <div>
          <b>value: {text}</b>
        </div>
      </div>
    );
  }

  export default InputSample;
```



JS InputSample.js > InputSample > onChangeName

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

```
function InputSample() {  
  const [name, setName] = useState('No Name');  
  const [age, setAge] = useState(0);
```

```
  const onChangeName = (e) => {  
    setName(e.target.value);  
  };
```


```
  const onChangeAge = (e) => {  
    setAge(e.target.value);  
  };
```

```
  const onReset = () => {  
    setName('No Name');  
    setAge(0);  
  };
```

```
  return (  
    <div>  
      <input onChange={onChangeName} value={name} /> <br/>  
      <input onChange={onChangeAge} value={age} /> <br/>  
      <button onClick={onReset}>Reset</button>  
      <div>  
        <b>Name: {name}</b> <br/>  
        <b>Age: {age}</b> <br/>  
      </div>  
    </div>  
  );  
}
```

```
export default InputSample;
```

State를 줄여보자

JS InputSample.js >  InputSample

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

```
function InputSample() {
```

```
  const [info, setInfo] = useState({name: '', age:0});
```

```
  const {name, age} = info;
```

```
  const onChange = (e) => {
```

```
    const {value, name} = e.target;
```

```
    setInfo({
```

```
      ...info,
```

```
      [name]: value
```

```
    });
```

```
  };
```

```
  const onReset = () => {
```

```
    setInfo({
```

```
      name: '',
```

```
      age: 0,
```

```
    })
```

```
  };
```

```
    return (  
      <div>  
        <input name="name" onChange={onChange} value={name} /> <br/>  
        <input name="age" onChange={onChange} value={age} /> <br/>  
        <button onClick={onReset}>Reset</button>  
        <div>  
          <b>Name: {name}</b> <br/>  
          <b>Age: {age}</b> <br/>  
        </div>  
      </div>  
    );  
  }  
  
  export default InputSample;
```

spread 문법

객체의 내용을 모두 "펼쳐서" 기존 객체를 복사

```
const animals = ['개', '고양이', '참새'];  
const anotherAnimals = [...animals, '비둘기'];  
console.log(animals);  
console.log(anotherAnimals);
```

```
▶ ["개", "고양이", "참새"]
```

```
▶ ["개", "고양이", "참새", "비둘기"]
```

```
const numbersOne = [1, 2, 3];  
const numbersTwo = [4, 5, 6];  
const numbersCombined = [...numbersOne, ...numbersTwo];
```

1,2,3,4,5,6

```
> const a = {  
  one: 1,  
  two: 2,  
}
```

```
< undefined
```

```
> const b = {  
  ...a,  
  three: 3  
}
```

```
< undefined
```

```
> b
```

```
< ► {one: 1, two: 2, three: 3}
```

```
> const c = {  
  ...a,  
  two: 4,  
}
```

```
< undefined
```

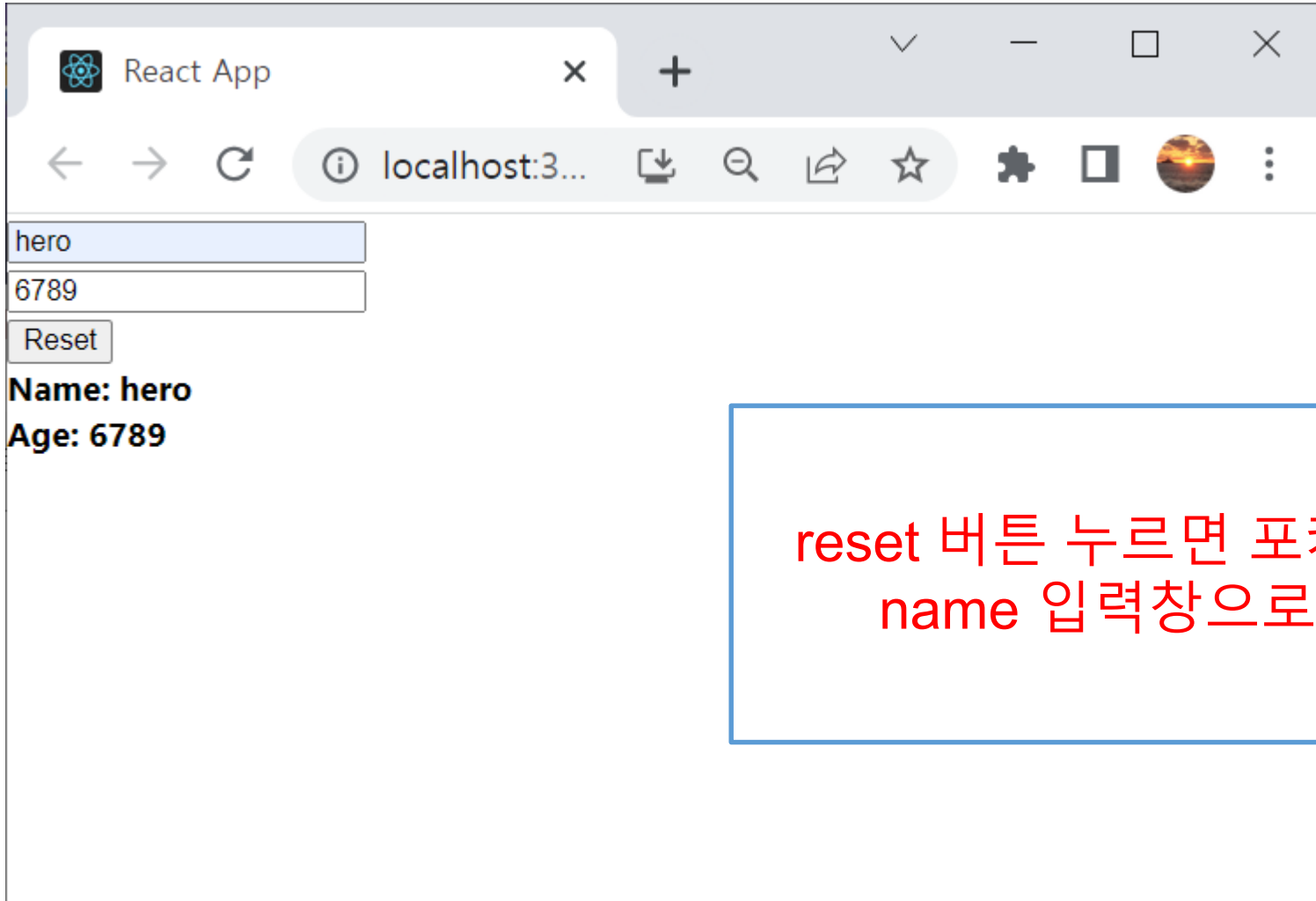
```
> c
```

```
< ► {one: 1, two: 4}
```

```
>
```

useRef 로 특정 DOM 선택

리액트를 사용하는 프로젝트에서 DOM 을 직접 선택해야 하는 상황이 발생
ex) 특정 엘리먼트의 크기를 가져오거나, 스크롤바 위치를 가져오거나 설정
또는 포커스를 설정 등 다양한 상황이 존재함




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

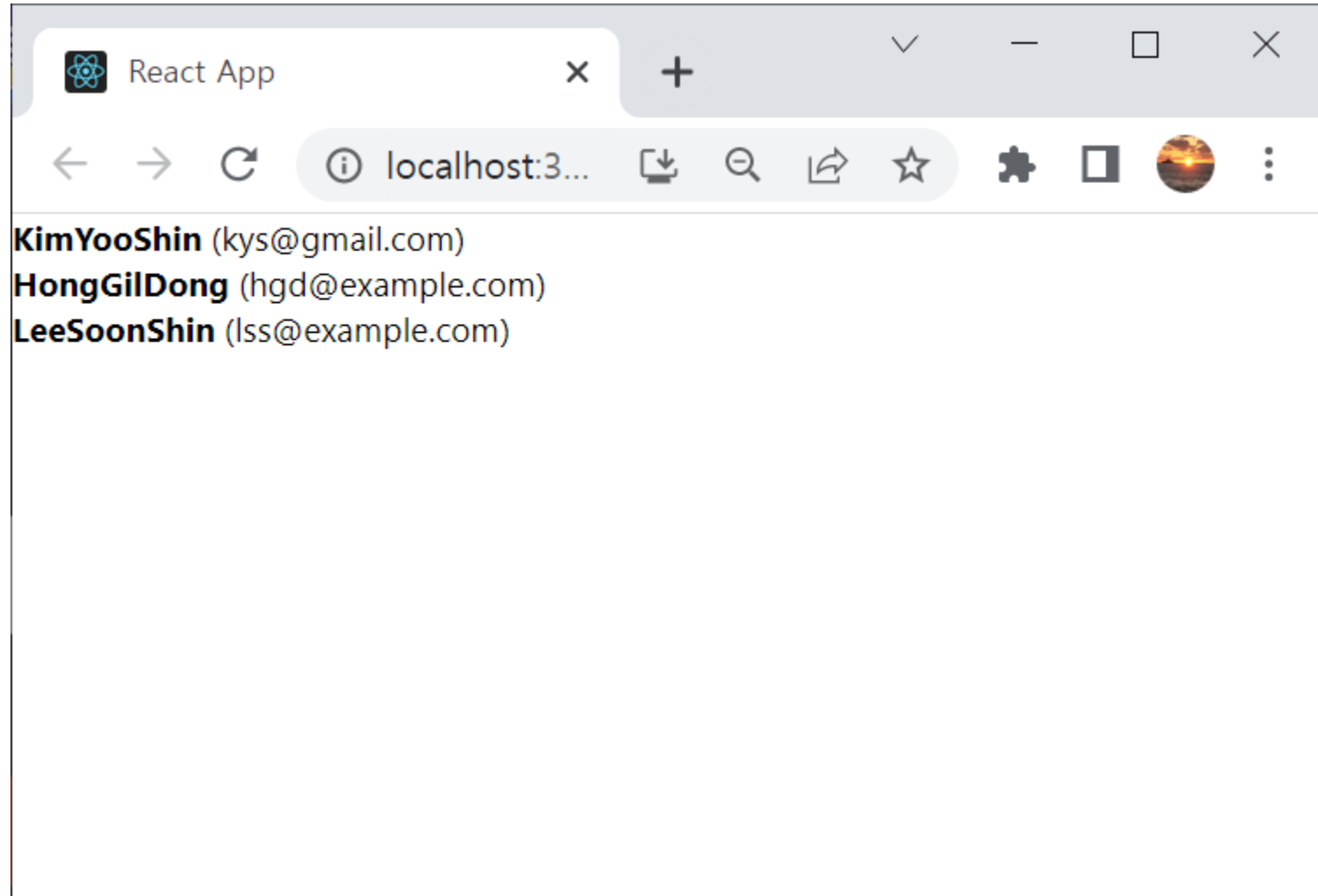
function InputSample() {
  const [info, setInfo] = useState({name: '', age:0});
  const {name, age} = info;
  const nameInput = useRef();




  const onChange = (e) => {
    const {value, name} = e.target;
    setInfo({
      ...info,
      [name]: value
    });
  };

  const onReset = () => {
    setInfo({
      name: '',
      age: 0,
    });
    nameInput.current.focus();
  };
}
```

```
return (  
  <div>  
    <input name="name" onChange={onChange} value={name} ref={nameInput} /> <br />  
    <input name="age" onChange={onChange} value={age} /> <br />  
    <button onClick={onReset}>Reset</button>  
    <div>  
      <b>Name: {name}</b> <br />  
      <b>Age: {age}</b> <br />  
    </div>  
  </div>  
)  
}  
  
export default InputSample;
```

map 사용법 및 key에 대해서 알아보자



JS UserList.js >  UserList >  users >  email

```
import React from 'react';
```

```
function UserList() {
```

```
  const users = [
```

```
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
```

```
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
```

```
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
```

```
  ];
```

```
  return (
```

```
    <div>
```

```
      <div>
```

```
        <b>{users[0].username}</b> <span>({users[0].email})</span>
```

```
      </div>
```

```
      <div>
```

```
        <b>{users[1].username}</b> <span>({users[1].email})</span>
```

```
      </div>
```

```
      <div>
```

```
        <b>{users[2].username}</b> <span>({users[2].email})</span>
```

```
      </div>
```

```
    </div>
```

```
  );
```

```
}
```

```
export default UserList;
```

```
import React from 'react';

function User({ user }) {
  return (
    <div>
      <b>{user.username}</b> <span>({user.email})</span>
    </div>
  );
}

function UserList() {
  const users = [
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
  ];
  return (
    <div>
      <User user={users[0]} />
      <User user={users[1]} />
      <User user={users[2]} />
    </div>
  );
}

export default UserList;
```

map() 함수는 배열안에 있는 각 원소를 변환하여 새로운 배열을 만들어준다.
리액트에서 동적인 배열을 렌더링해야 할 때는 이 함수를 사용하여 일반
데이터 배열을 리액트 엘리먼트로 이루어진 배열로 변환

```
function UserList() {  
  const users = [  
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },  
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },  
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }  
  ];  
  return (  
    <div>  
      {users.map(aa => (  
        <User user={aa} />  
      ))}  
    </div>  
  );  
}  
export default UserList;
```

React App

localhost:3000

KimYooShin (kys@gmail.com)
HongGilDong (hgd@example.com)
LeeSoonShin (lss@example.com)

리액트에서 배열을 렌더링 할 때에는 key 라는 props 를 설정.
key 값은 각 원소들마다 가지고 있는 고유값으로 설정

ElementsConsoleSourcesNetworkPerformanceMemory

topFilter

Warning: Each child in a list should have a unique "key" prop.

react_devtools_backend_compact.js:2367

Check the render method of `UserList`. See <https://reactjs.org/link/warning-keys> for more information.

at User (<http://localhost:3000/main.e2b14b7....hot-update.js:25:5>)

at UserList

at App

React App

localhost:3000

← → ↻ ⓘ

📄 🔍 🔗 ☆ ⚙️ 🖱️ 🌅 ⋮

KimYooShin (kys@gmail.com)

HongGilDong (hgd@example.com)

LeeSoonShin (lss@example.com)

```
return (  
  <div>  
    {users.map(user => (  
      <User user={user} key={user.id} />  
    ))}  
  </div>  
);
```

🖱️ 📄 | Elements Console

🎬 🚫 | top ▼ | 👁️ | Filter

Default levels ▼ | 1 Issue: 🗨️ 1 | ⚙️

⚠️ DevTools failed to load source map: Could not load content for chrome-extension://gighmmp_iobklfepjocnamgkbbiglidom/browser-polyfill.js.map: System error: net::ERR_FILE_NOT_FOUND

> |

key가 없다면....

```
const array = ['a', 'b', 'c', 'd'];
```

```
array.map(item => <div>{item}</div>);
```

위 배열의 b 와 c 사이에 z 를 삽입하게 된다면, 리렌더링을 하게 될 때 <div>b</div> 와 <div>c</div> 사이에 새 div 태그를 삽입을 하게 되는 것이 아니라, 기존의 c 가 z 로 바뀌고, d 는 c 로 바뀌고, 맨 마지막에 d 가 새로 삽입됩니다.

Map을 이용한 렌더링

- `arr.map(i =>)` 의 형태로 하위 컴포넌트에게 값을 전달해준다.

Map에서 Key가 필요한 이유

- Map에 key 값이없다면 중간의 값이 바뀌었을때 그 하위 값들이 전부 변하기 때문이다. key값을 사용한다면 key를 이용해 중간의 값을 추가하게 된다.

useRef 로 컴포넌트 안의 변수 만들기

useRef Hook 은 DOM 을 선택하는 용도 외에 추가 용도 제공.


컴포넌트 안에서 조회 및 수정 할 수 있는 변수를 관리하는 것입니다.

```
App.js > [🔗] default
import React from 'react';
import UserList from './UserList';

function App() {
  const users = [
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
  ];

  return <UserList users={users} />;
}

export default App;
```

src > JS UserList.js >  UserList

```
1  import React from 'react';
2
3  function User({ user }) {
4    return (
5      <div>
6        <b>{user.username}</b> <span>({user.email})</span>
7      </div>
8    );
9  }
10
11 function UserList({users}) {
12
13   return (
14     <div>
15       {users.map(user => (
16         <User user={user} key={user.id} />
17       ))}
18     </div>
19   );
20 }
21 export default UserList;
```

App.js > [🔍] default

```
import React, { useRef } from 'react';  
import UserList from './UserList';
```

```
function App() {  
  const users = [  
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },  
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },  
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }  
  ];  
  
  const nextId = useRef(4);  
  const onCreate = () => {  
    // ...  
  
    nextId.current += 1;  
  };  
  return <UserList users={users} />;  
}
```

```
export default App;
```

배열에 항목 추가하기

- CreateUser 컴포넌트 작성

```
import React from 'react';

function CreateUser({ username, email, onChange, onCreate }) {
  return (
    <div>
      <input
        name="username"
        placeholder="Account Name"
        onChange={onChange}
        value={username}
      />
      <input
        name="email"
        placeholder="email"
        onChange={onChange}
        value={email}
      />
      <button onClick={onCreate}>Registration</button>
    </div>
  );
}

export default CreateUser;
```

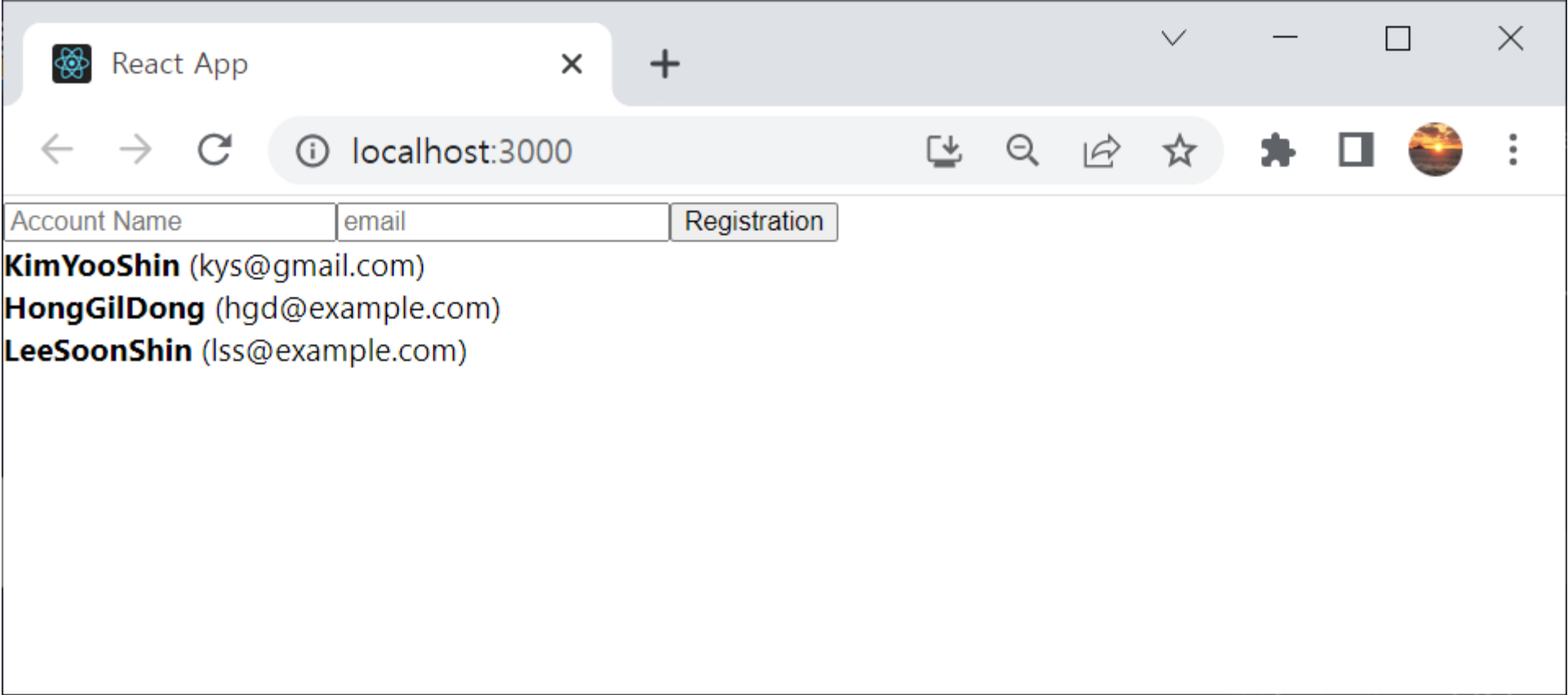
```
import React, { useRef } from 'react';
import UserList from './UserList';
import CreateUser from './CreateUser';

function App() {
  const users = [
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
  ];

  const nextId = useRef(4);
  const onCreate = () => {
    // ...

    nextId.current += 1;
  };
  return (
    <>
      <CreateUser />
      <UserList users={users} />
    </>
  );
}

export default App;
```

```

import React, { useRef, useState } from 'react';
import UserList from './UserList';
import CreateUser from './CreateUser';

function App() {
  const [inputs, setInputs] = useState({
    username: '',
    email: ''
  });
  const { username, email } = inputs;
  const onChange = e => {
    const { name, value } = e.target;
    setInputs({
      ...inputs,
      [name]: value
    });
  };

  const users = [
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
  ];

```

```

const nextId = useRef(4);
const onCreate = () => {
  // ...

  setInputs({
    username: '',
    email: ''
  });
  nextId.current += 1;
};

return (
  <>
    <CreateUser
      username={username}
      email={email}
      onChange={onChange}
      onCreate={onCreate}
    />
    <UserList users={users} />
  </>
);
}

export default App;

```

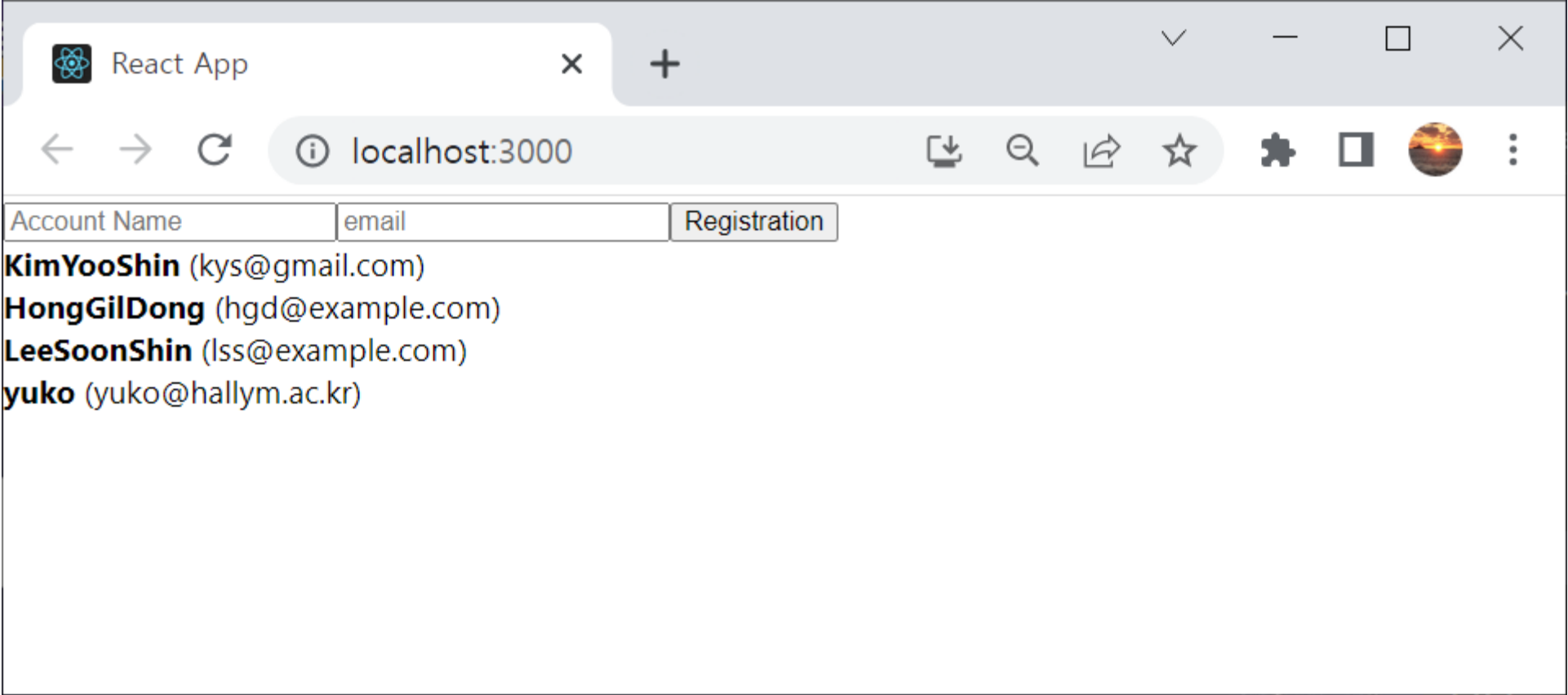
```
import React, { useRef, useState } from 'react';
import UserList from './UserList';
import CreateUser from './CreateUser';
```

```
function App() {
  const [inputs, setInputs] = useState({
    username: '',
    email: ''
  });
  const { username, email } = inputs;
  const onChange = e => {
    const { name, value } = e.target;
    setInputs({
      ...inputs,
      [name]: value
    });
  };
  const [users, setUsers] = useState([
    { id: 1, username: 'KimYooShin', email: 'kys@gmail.com' },
    { id: 2, username: 'HongGilDong', email: 'hgd@example.com' },
    { id: 3, username: 'LeeSoonShin', email: 'lss@example.com' }
  ]);
```

```
const nextId = useRef(4);
const onCreate = () => {
  const user = {
    id: nextId.current,
    username,
    email
  };
  setUsers([...users, user]);

  setInputs({
    username: '',
    email: ''
  });
  nextId.current += 1;
};
return (
  <>
    <CreateUser
      username={username}
      email={email}
      onChange={onChange}
      onCreate={onCreate}
    />
    <UserList users={users} />
  </>
);
}
```

```
export default App;
```



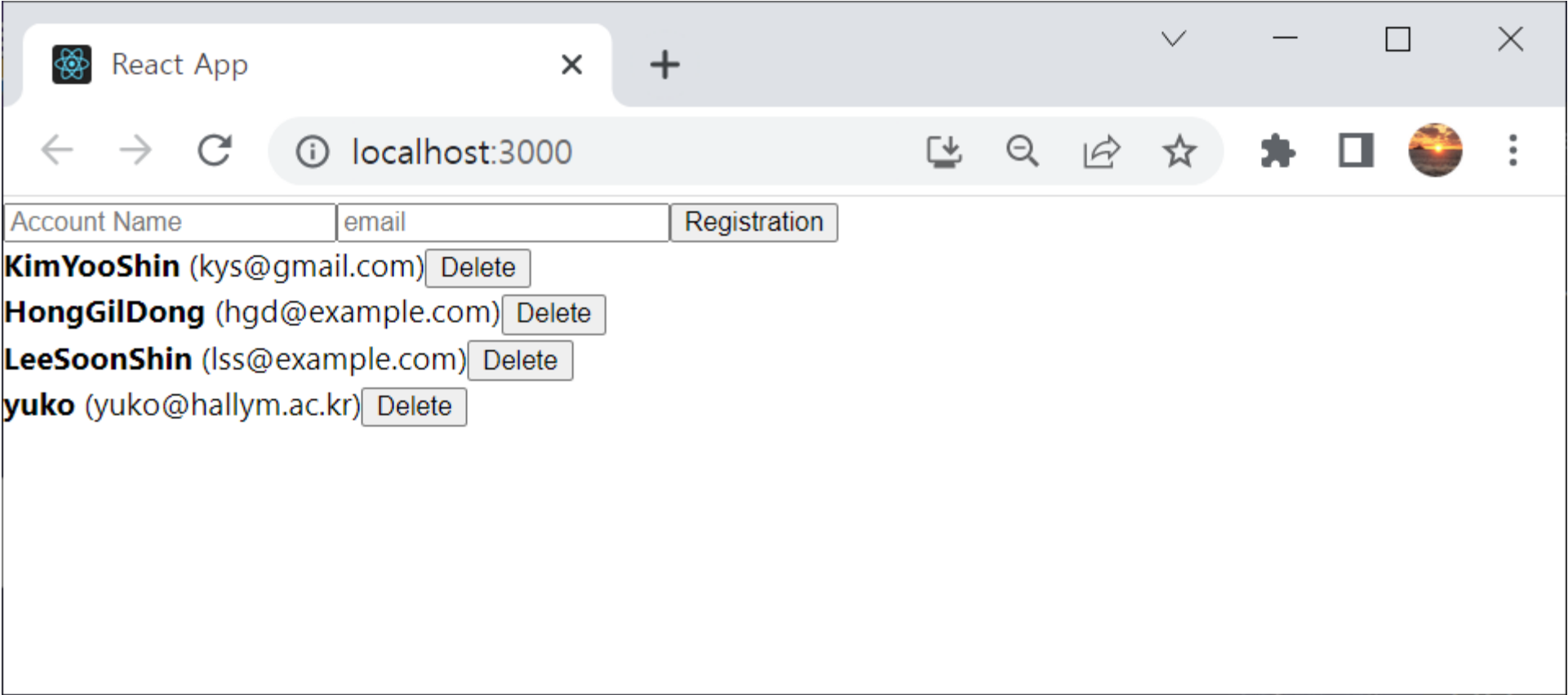
배열에 항목 제거하기

```
import React from 'react';

function User({ user, onRemove }) {
  return (
    <div>
      <b>{user.username}</b> <span>({user.email})</span>
      <button onClick={() => onRemove(user.id)}>Delete</button>
    </div>
  );
}

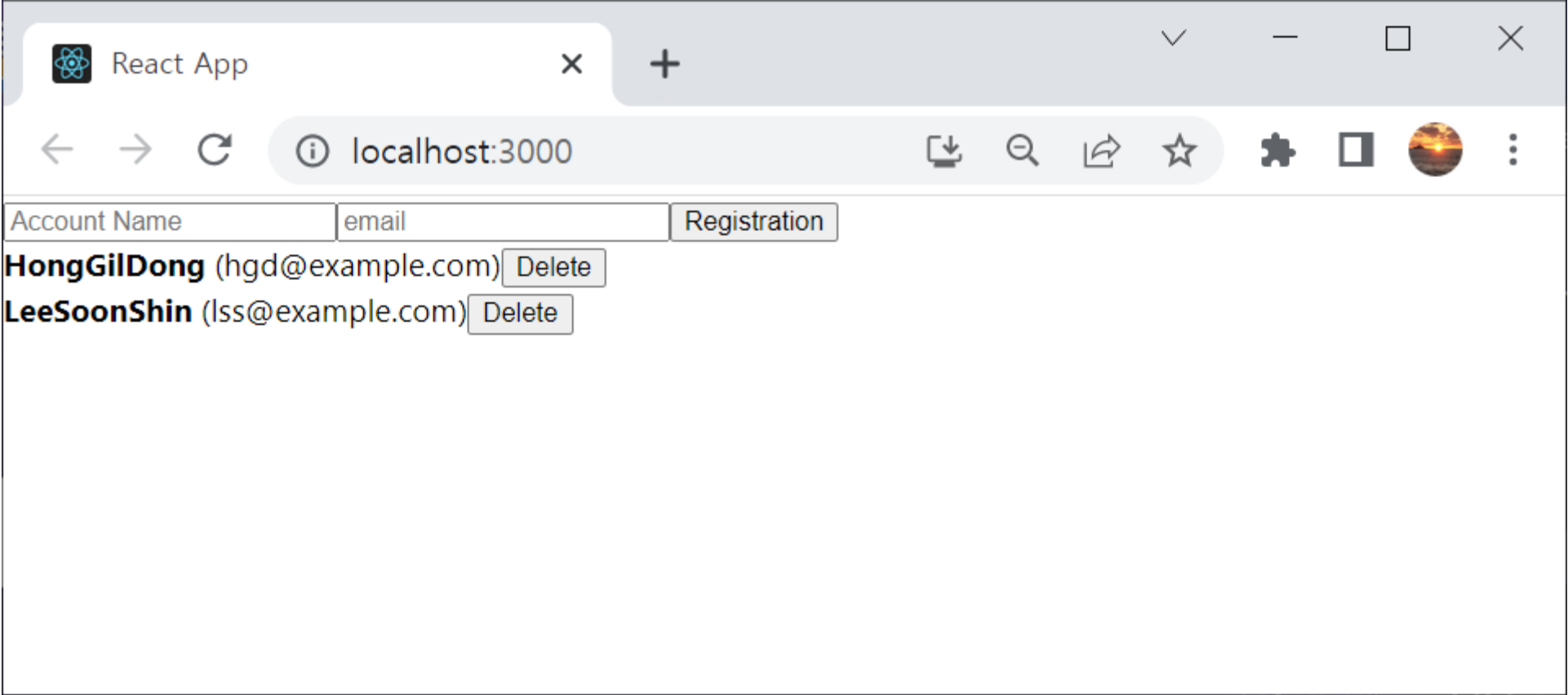
function UserList({ users, onRemove }) {
  return (
    <div>
      {users.map(user => (
        <User user={user} key={user.id} onRemove={onRemove} />
      ))}
    </div>
  );
}

export default UserList;
```



```
const onRemove = id => {
  |   setUsers(users.filter(user => user.id !== id));
};

return (
  <>
    <CreateUser
      |   username={username}
      |   email={email}
      |   onChange={onChange}
      |   onCreate={onCreate}
    />
    <UserList users={users} onRemove={onRemove} />
  </>
);
}
export default App;
```



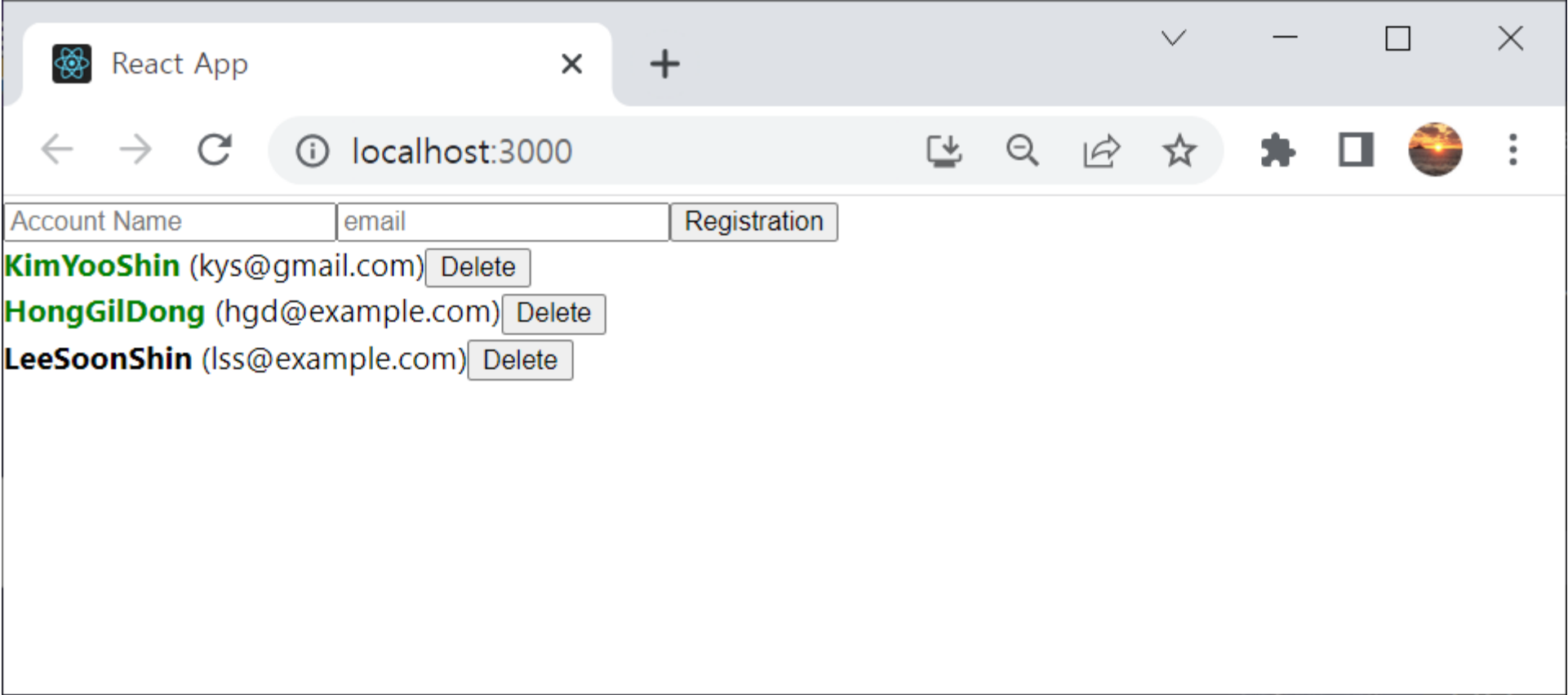
배열 항목 수정하기

```
import React from 'react';

function User({ user, onRemove, onToggle }) {
  return (
    <div>
      <b
        style={{
          cursor: 'pointer',
          color: user.active ? 'green' : 'black'
        }}
        onClick={() => onToggle(user.id)}
      >
        {user.username}
      </b>
      &nbsp;
      <span>{user.email}</span>
      <button onClick={() => onRemove(user.id)}>Delete</button>
    </div>
  );
}
```

```
function UserList({ users, onRemove, onToggle }) {  
  return (  
    <div>  
      {users.map(user => (  
        <User  
          user={user}  
          key={user.id}  
          onRemove={onRemove}  
          onToggle={onToggle}  
        />  
      ))}  
    </div>  
  );  
}  
  
export default UserList;
```

```
const onRemove = id => {
  |   setUsers(users.filter(user => user.id !== id));
};
const onToggle = id => {
  |   setUsers(
  |     users.map(user =>
  |       |   user.id === id ? { ...user, active: !user.active } : user
  |     )
  |   );
};
return (
  <>
  |   <CreateUser
  |     |   username={username}
  |     |   email={email}
  |     |   onChange={onChange}
  |     |   onCreate={onCreate}
  |   />
  |   <UserList users={users} onRemove={onRemove} onToggle={onToggle} />
  | </>
  | );
}
export default App;
```



useEffect를 사용하기

- useEffect 라는 Hook 을 사용하여 컴포넌트가 마운트 됐을 때 (처음 나타났을 때), 언마운트 됐을 때 (사라질 때), 그리고 업데이트 될 때 (특정 props가 바뀔 때) 특정 작업을 처리하는 방법

```

import React, { useEffect } from 'react';

function User({ user, onRemove, onToggle }) {
  useEffect(() => {
    console.log('컴포넌트가 화면에 나타남');
    return () => {
      console.log('컴포넌트가 화면에서 사라짐');
    };
  }, []);

  return (
    <div>
      <b
        style={{
          cursor: 'pointer',
          color: user.active ? 'green' : 'black'
        }}
        onClick={() => onToggle(user.id)}
      >
        {user.username}
      </b>
      &nbsp;
      <span>{user.email}</span>
      <button onClick={() => onRemove(user.id)}>Delete</button>
    </div>
  );
}

```

```

function UserList({ users, onRemove, onToggle }) {
  return (
    <div>
      {users.map(user => (
        <User
          user={user}
          key={user.id}
          onRemove={onRemove}
          onToggle={onToggle}
        />
      ))}
    </div>
  );
}

export default UserList;

```

React App

localhost:3000

Account Name

email

Registration

KimYooShin (kys@gmail.com)

Delete

HongGilDong (hgd@example.com)

Delete

LeeSoonShin (lss@example.com)

Delete

Elements

Console

Sources

Network

Performance

Memory

Application

3

Filter

Default levels

3 Issues

⚠ DevTools failed to load source map: Could not load content for chrome-extension://gighmmpiobklfepjocnamgkkbiglidom/browser-polyfill.js.map: System error: net::ERR_FILE_NOT_FOUND

3

컴포넌트가 화면에 나타남

UserList.js:5

3

컴포넌트가 화면에서 사라짐

UserList.js:7

3

컴포넌트가 화면에 나타남

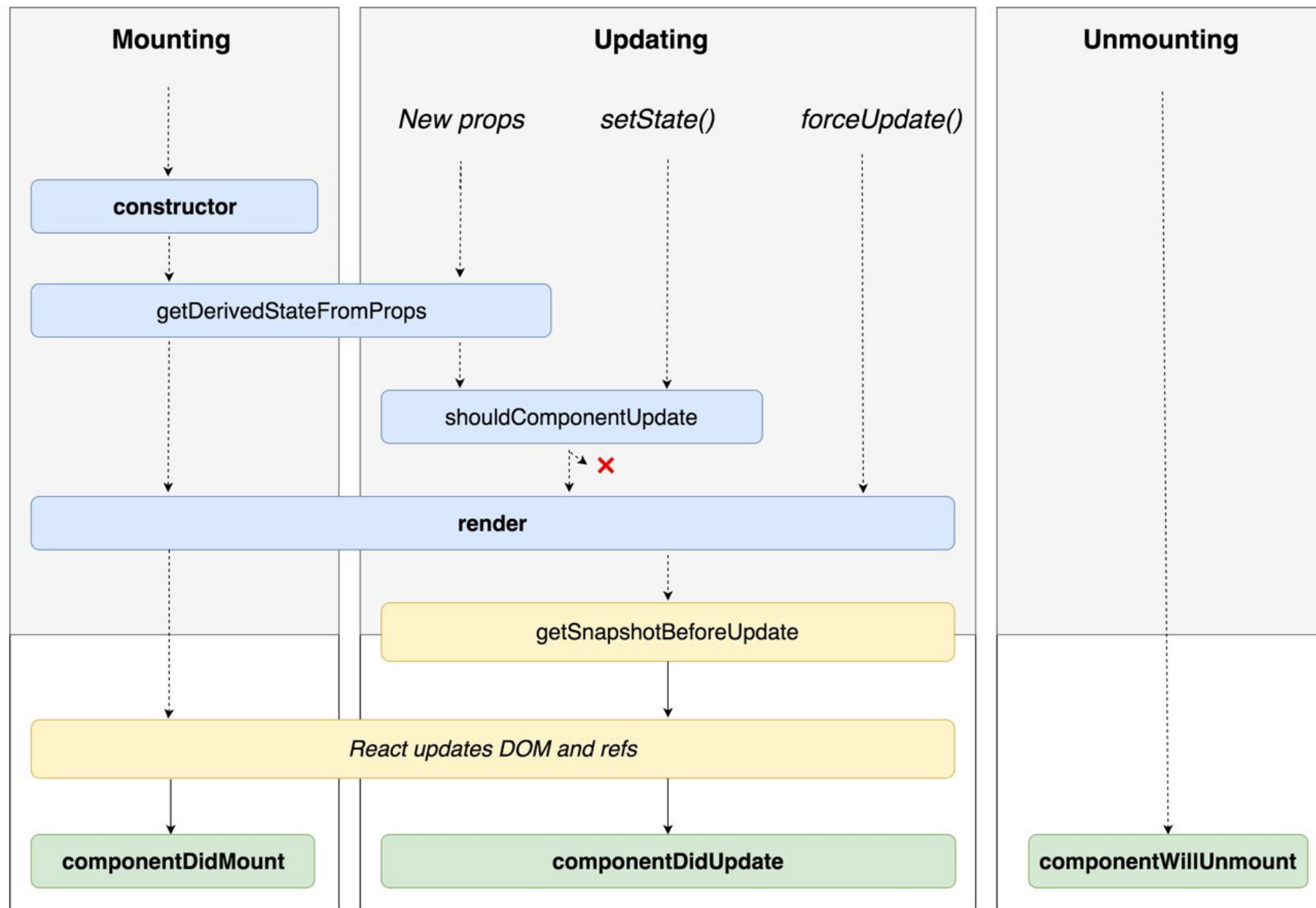
UserList.js:5

> |

“Render Phase”
Pure and has no side effects.
May be paused, aborted or
restarted by React.

“Pre-Commit Phase”
Can read the DOM.

“Commit Phase”
Can work with DOM,
run side effects,
schedule updates.



기본 형태

```
useEffect(function, deps)
```

- **function** : 실행하고자 하는 함수
- **deps** : 배열 형태. function을 실행시킬 조건.

deps에 특정값을 넣게 되면 컴포넌트가 mount 될 때, 지정한 값이 업데이트될 때 useEffect를 실행합니다.

useEffect 함수 불러오기

```
import React, { useEffect } from "react";
```

1. Component가 Mount 되었을 때(나타날 때)

```
useEffect(() => {  
  console.log("렌더링 될때마다 실행");  
});
```

deps부분을 생략한다면 해당 컴포넌트가 렌더링 될 때마다 useEffect가 실행되게 됩니다.
넣어줍니다.

```
useEffect(() => {  
  console.log("맨 처음 렌더링될 때 한 번만 실행");  
}, []);
```

한 번만 실행하고 싶다면 deps위치에 빈 배열:

2. Component가 Update 되었을 때(props, state 변경)

```
useEffect(() => {  
  console.log(name);  
  console.log("name이라는 값이 업데이트 될 때만 실행");  
},[name]);
```

특정값이 업데이트될 때만 실행하고 싶을 때는 deps위치의 배열 안에 실행 조건을 넣어줍니다.

```

import React, { useEffect } from 'react';

function User({ user, onRemove, onToggle }) {
  useEffect(() => {
    console.log('user 값이 설정됨');
    console.log(user);
    return () => {
      console.log('user 가 바뀌기 전..');
      console.log(user);
    };
  }, [user]);

  return (
    <div>
      <b
        style={{
          cursor: 'pointer',
          color: user.active ? 'green' : 'black'
        }}
        onClick={() => onToggle(user.id)}
      >
        {user.username}
      </b>
      &nbsp;
      <span>{user.email}</span>
      <button onClick={() => onRemove(user.id)}>Delete</button>
    </div>
  );
}

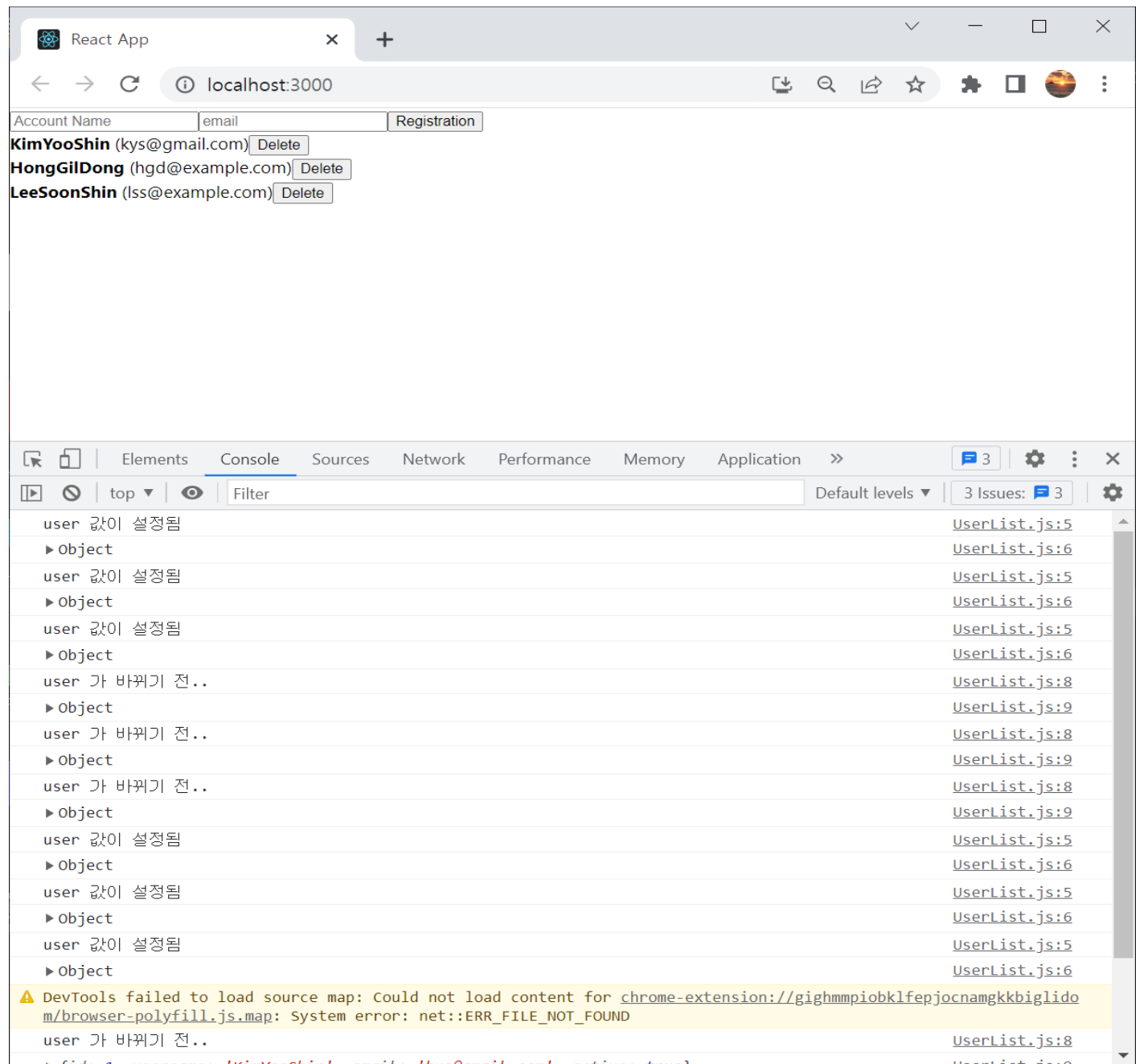
```

```

function UserList({ users, onRemove, onToggle }) {
  return (
    <div>
      {users.map(user => (
        <User
          user={user}
          key={user.id}
          onRemove={onRemove}
          onToggle={onToggle}
        />
      ))}
    </div>
  );
}

export default UserList;

```



React App

localhost:3000

Account Name

email

Registration

KimYooShin (kys@gmail.com)

Delete

HongGilDong (hgd@example.com)

Delete

LeeSoonShin (lss@example.com)

Delete

Elements

Console

Sources

Network

Performance

Memory

Application

top

Filter

Default levels

3 Issues: 3

user 가 바뀌기 전..

UserList.js:8

▶ {id: 1, username: 'KimYooShin', email: 'kys@gmail.com', active: true}

UserList.js:9

user 값이 설정됨

UserList.js:5

▶ {id: 1, username: 'KimYooShin', email: 'kys@gmail.com', active: false}

UserList.js:6

user 가 바뀌기 전..

UserList.js:8

▶ {id: 1, username: 'KimYooShin', email: 'kys@gmail.com', active: false}

UserList.js:9

user 값이 설정됨

UserList.js:5

▶ {id: 1, username: 'KimYooShin', email: 'kys@gmail.com', active: true}

UserList.js:6

>

Q&A