

Chapter 10. 리스트와 키

교재: 처음만난 리액트 (저자: 이인제, 한빛출판사)



Contents

- CHAPTER 10: 리스트와 키

10.1 리스트와 키란 무엇인가?

10.2 여러 개의 컴포넌트 렌더링하기

10.3 기본적인 리스트 컴포넌트

10.4 리스트의 키에 대해 알아보기

10.5 (실습) 블로그 출력하기

SECTION 10.1 리스트와 키란 무엇인가?

◦ 리스트(List)

- 같은 아이템을 순서대로 모아놓은 것
- 리스트를 위해 사용하는 구조 – 배열
 - 배열 – 자바스크립트의 변수나 객체를 하나의 변수로 묶어놓은 것

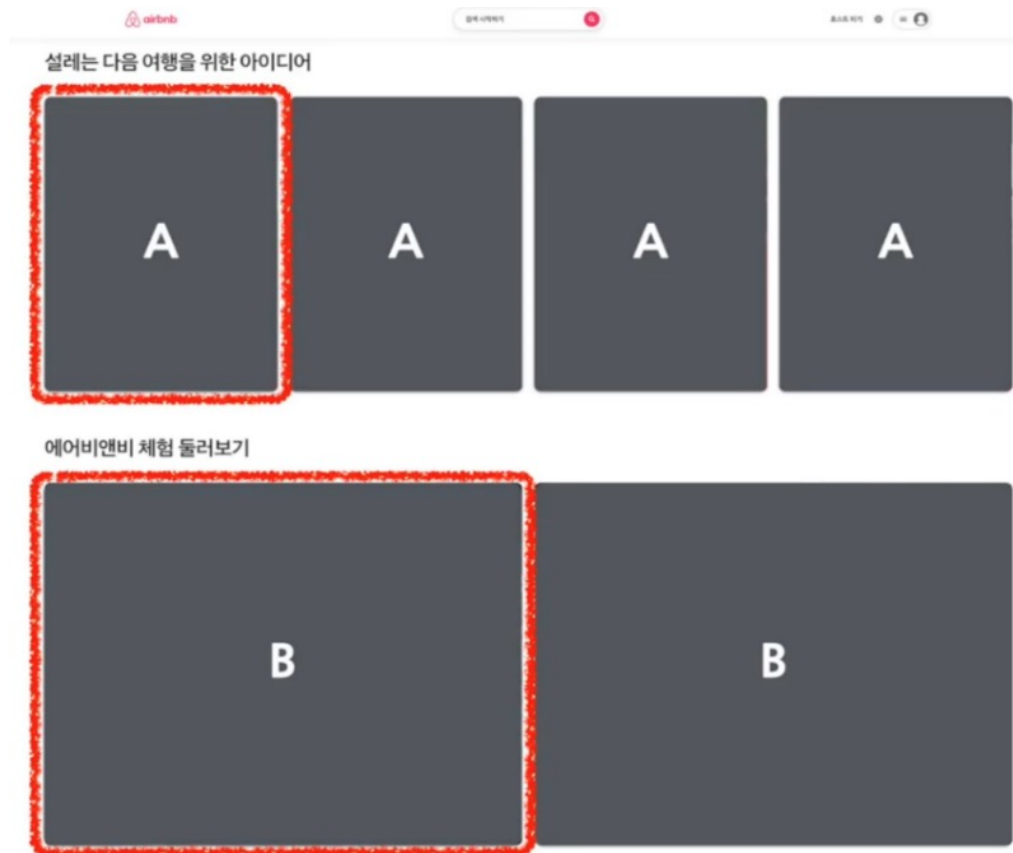
```
const numbers = [1, 2, 3, 4, 5];
```

◦ 키(Key)

- 각 객체나 아이템을 구분할 수 있는 고유한 값
- ✓ 리액트에서는 배열과 키를 사용하여 반복되는 다수의 엘리먼트를 쉽게 렌더링할 수 있음

SECTION 10.2 여러 개의 컴포넌트 렌더링하기

- 컴포넌트 반복 구조



SECTION 10.2 여러 개의 컴포넌트 렌더링하기

◦ Array.map()

- 배열 내의 모든 요소 각각에 대하여 주어진 함수를 호출한 결과를 모아 새로운 배열을 반환

```
arr.map(callback(currentValue, index, array)[, thisArg])
```

currentValue : 처리할 현재 요소

index : 처리할 현재 요소의 인덱스

array : map()을 호출한 배열

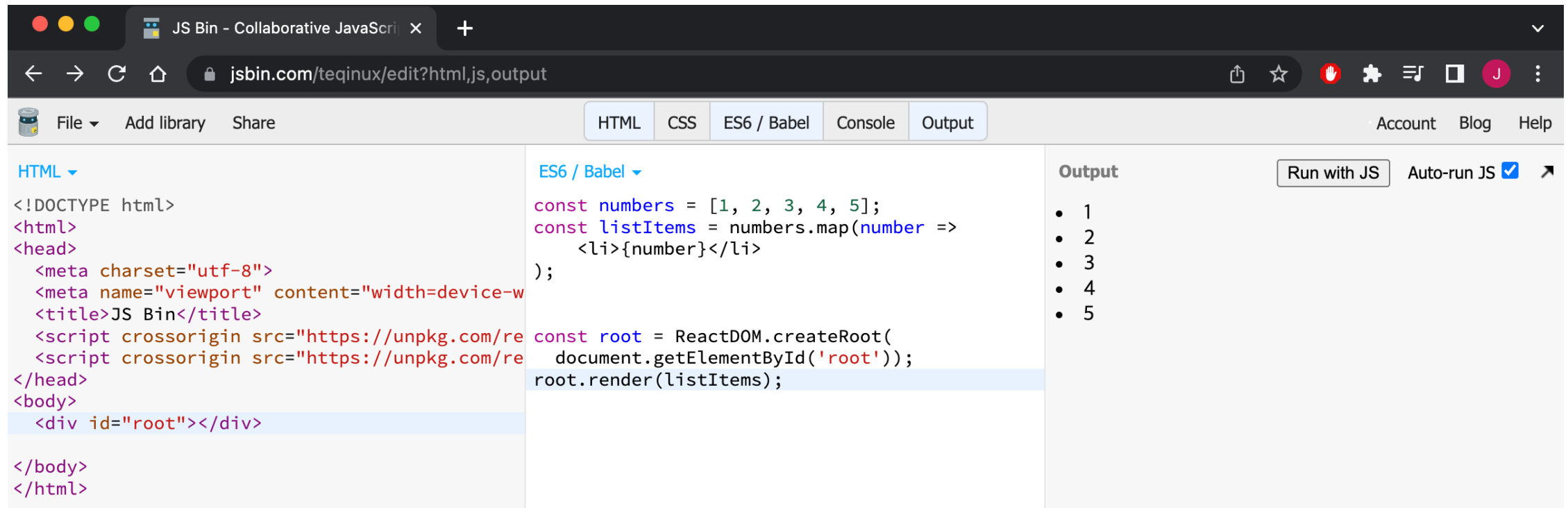
```
const array1 = [1, 4, 9, 16];  
const map1 = array1.map(x => x * 2);  
console.log(map1);
```

```
const map2 = array1.map(Math.sqrt);  
console.log(map2);
```

SECTION 10.2 여러 개의 컴포넌트 렌더링하기

◦ Array.map()

```
const numbers = [1, 2, 3, 4, 5];  
const listItems = numbers.map(number =>  
  <li>{number}</li>  
);
```



The screenshot shows the JS Bin editor interface. The top bar includes navigation icons and the URL `jsbin.com/teqinux/edit?html,js,output`. Below the bar are tabs for `HTML`, `CSS`, `ES6 / Babel`, `Console`, and `Output`. The `HTML` panel on the left contains a basic HTML structure with a `div id="root"`. The `ES6 / Babel` panel in the middle shows the JavaScript code for creating a list of numbers and rendering it with React. The `Output` panel on the right displays the rendered list of numbers from 1 to 5. Buttons for `Run with JS` and `Auto-run JS` are visible in the top right of the output panel.

```
HTML  
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="utf-8">  
  <meta name="viewport" content="width=device-width">  
  <title>JS Bin</title>  
  <script crossorigin src="https://unpkg.com/react@16.8.6">  
  <script crossorigin src="https://unpkg.com/react-dom@16.8.6">  
</head>  
<body>  
  <div id="root"></div>  
</body>  
</html>
```

```
ES6 / Babel  
const numbers = [1, 2, 3, 4, 5];  
const listItems = numbers.map(number =>  
  <li>{number}</li>  
);  
  
const root = ReactDOM.createRoot(  
  document.getElementById('root')  
>);  
root.render(listItems);
```

Output

- 1
- 2
- 3
- 4
- 5

SECTION 10.3 기본적인 리스트 컴포넌트

리스트를 출력하는 컴포넌트



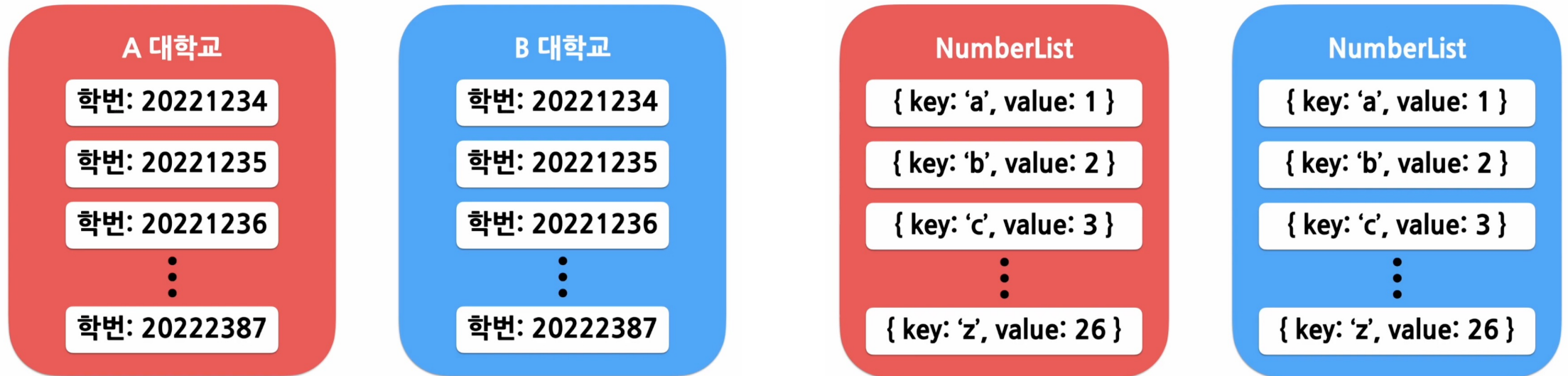
```
function NumberList(props) {  
  const { numbers } = props;  
  
  const listItems = numbers.map(number =>  
    <li>{number}</li>  
  );  
  
  return (  
    <ul>{listItems}</ul>  
  );  
}  
  
const numbers = [1, 2, 3, 4, 5];  
const root = ReactDOM.createRoot(  
  document.getElementById('root'));  
  
root.render(  
  <NumberList numbers={numbers} />  
);
```

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

SECTION 10.4 리스트의 키에 대해 알아보기

리스트의 키

- 리스트에서 아이템을 구분하기 위한 고유한 값
- 리스트에서 어떤 아이템이 변경, 추가 또는 제거되었는지 구분하기 위해 사용
- 키의 값은 같은 List에 있는 Elements 사이에서만 고유한 값이면 됨



SECTION 10.4 리스트의 키에 대해 알아보기

- 키값을 만드는 방법

- 키값으로 숫자의 값을 사용

```
const numbers = [1, 2, 3, 4, 5];
const listItems = numbers.map(number =>
  <li key={number.toString()}>{number}</li>
);
```

-> 배열 요소가 중복된 값이 있을 수 있음

- 키값으로 id를 사용하는 방식 (아이템의 고유한 id가 있을 경우 사용)

```
const todos = [
  { id: 1, text: '첫 번째 할일' },
  { id: 2, text: '두 번째 할일' }
]
const todoItems = todos.map(todo =>
  <li key={todo.id}>{todo.text}</li>
);
```

SECTION 10.4 리스트의 키에 대해 알아보기

- 키값을 만드는 방법

- 키값으로 인덱스(index)를 사용하는 방법 (아이템의 고유한 id가 없을 경우 사용)

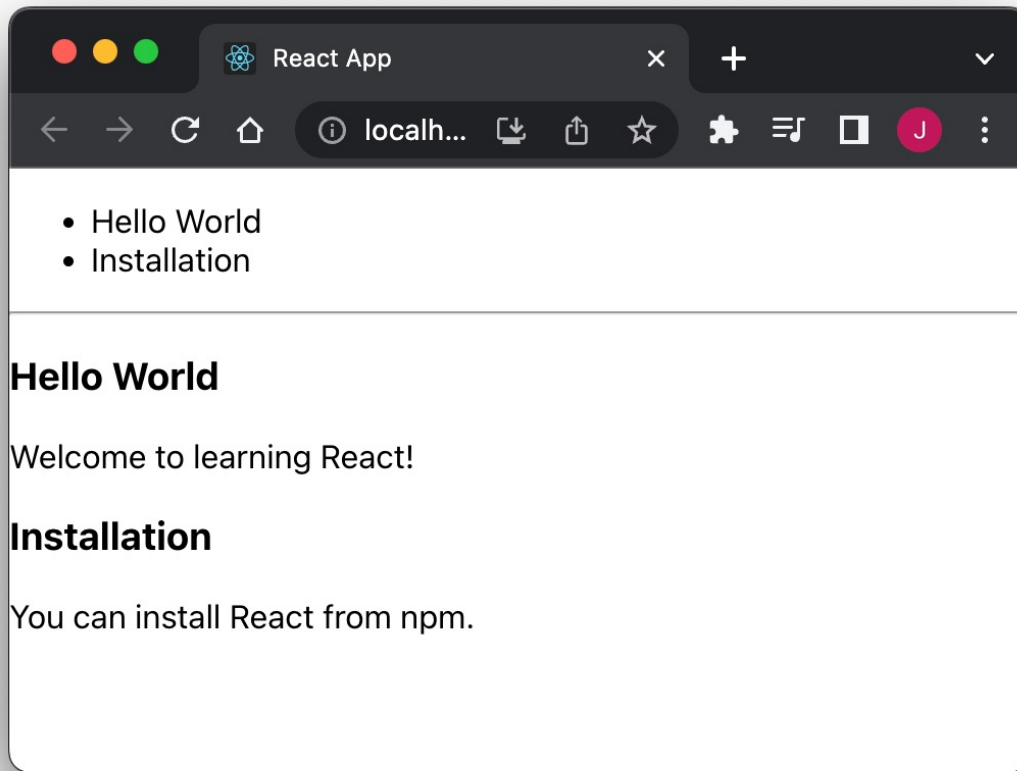
```
const todos = [  
  { text: '첫 번째 할일' },  
  { text: '두 번째 할일' },  
]  
const todoItems = todos.map((todo, index) =>  
  <li key={index}> {todo.text} </li>  
);
```

- map() 메서드 안에 있는 엘리먼트는 key가 필수

이 코드를 보면
key 값이 없으면
오류가 발생한다

PRACTICE 10.5 블로그 출력하기

- 블로그 사이드바와 콘텐츠 영역을 분리하여 표시하는 컴포넌트
 - 사이드바에는 제목, 콘텐츠 영역에는 제목과 콘텐츠 내용 표시



PRACTICE 10.5 블로그 출력하기

◦ Blog.jsx

```
const posts = [
  { id: 1, title: 'Hello World', content: 'Welcome to learning React!' },
  { id: 2, title: 'Installation', content: 'You can install React from npm.' }
];

function Blog(props) {
  const sidebar = (
    <ul>
      {posts.map((post) =>
        <li key={post.id}>
          {post.title}
        </li>
      )}
    </ul>
  );

  const content = posts.map((post) =>
    <div key={post.id}>
      <h3>{post.title}</h3>
      <p>{post.content}</p>
    </div>
  );

  return (
    <div>
      {sidebar}
      <hr />
      {content}
    </div>
  );
}

export default Blog;
```

[요약]

- 리스트
 - 같은 아이템을 순서대로 모아 놓은 것
- 키
 - 각 객체나 아이템을 구분할수 있는 고유한 값
- 여러 개의 컴포넌트 렌더링
 - 자바스크립트 배열의 `map()` 메서드 사용
 - 배열에 들어있는 각 변수에 어떤 처리를 한 뒤 결과(엘리먼트)를 배열로 만들어서 리턴
 - `map()` 메서드 안에 있는 엘리먼트는 꼭 키가 필요함
- 리스트의 키
 - 리스트에서 아이템을 구분하기 위한 고유한 문자열
 - 리스트에서 어떤 아이템이 변경, 추가 또는 제거되었는지 구분하기 위해 사용
 - 리액트에서는 키의 값은 같은 리스트에 있는 엘리먼트 사이에서만 고유한 값이면 됨

[요약]

- 다양한 키값의 사용법
 - 숫자값을 사용
 - 배열에 중복된 숫자가 들어있다면 키값도 중복되기 때문에 고유해야 한다는 키값의 조건이 충족되지 않음
 - id를 사용
 - id의 의미 자체가 고유한 값이므로 키값으로 사용하기 적합
 - id가 있는 경우에는 보통 id값을 키값으로 사용
 - 인덱스를 사용
 - 배열에서 아이템의 순서가 바뀔수 있는 경우에는 키값으로 인덱스를 사용하는 것을 권장하지 않음
 - 리액트에서는 키를 명시적으로 넣어 주지 않으면 기본적으로 이 인덱스 값을 키값으로 사용