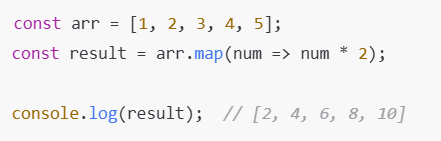
**JavaScript의 map() 함수**

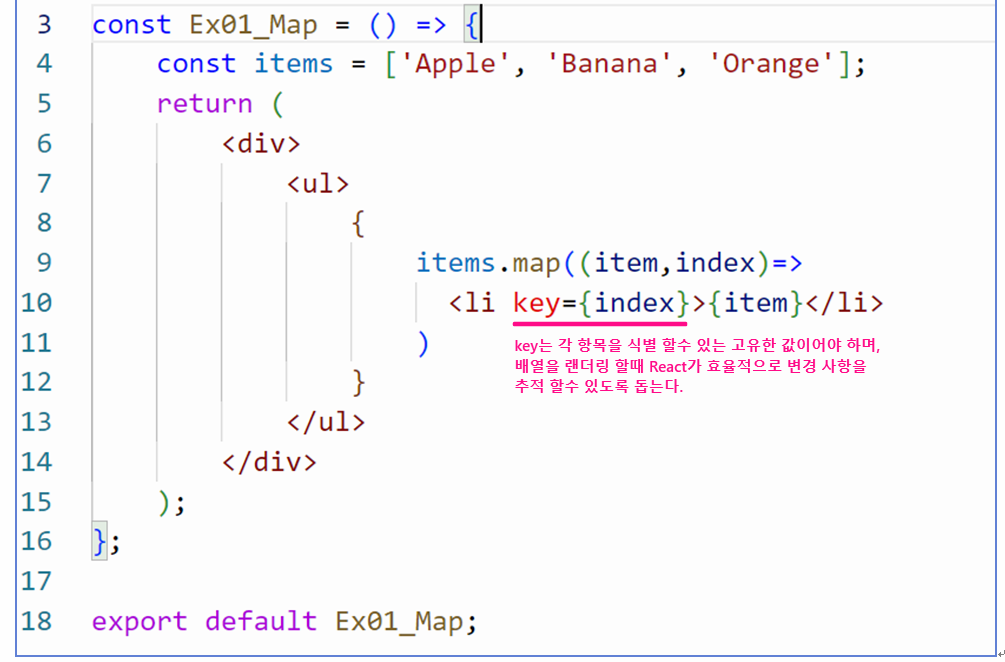
|  |
| --- |
| **JavaScript 배열 메서드** 중 하나로, 배열의 각 요소에 대해 주어진 함수를 실행하고 그 결과로 새로운 배열을 반환한다. React에서 map() 함수는 주로 **동적으로 리스트를 렌더링**할 때 사용된다.  예를 들어, 배열에 있는 데이터를 기반으로 여러 개의 컴포넌트를 생성할 때 유용하다.  map()은 원본 배열을 변경하지 않고, **변환된 새로운 배열을 반환한다.** |

**기본적인 map()함수**

****

**React에서 map() 함수의 사용**

React에서 map() 함수는 **리스트 렌더링** 시에 많이 사용된다. 예를 들어 배열에 있는 데이터를 기반으로 여러 개의 React 컴포넌트를 동적으로 생성하려면 map()을 사용하여 각 항목을 반복하면서 JSX를 반환다.



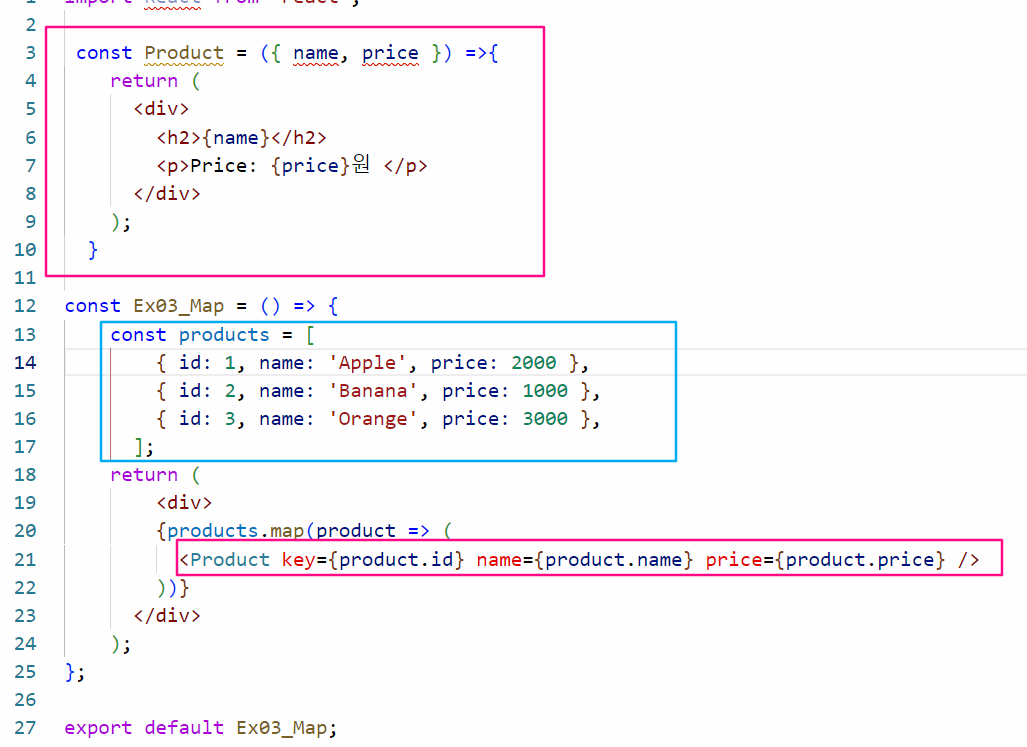
**객체 배열로 동적 리스트 렌더링**

배열의 각 항목이 객체일 경우, 객체 속성을 사용하여 렌더링할 수 있다.

****

**JSX를 사용하여 복잡한 컴포넌트 렌더링**

map() 함수는 JSX 내부에서 컴포넌트를 렌더링하는 데에도 사용된다.

****

|  |
| --- |
| **주의**  React에서 리스트를 렌더링할 때 key 속성은 매우 중요하다. key는 각 항목을 고유하게 식별할 수 있도록 돕고, React가 변경된 항목을 효율적으로 업데이트할 수 있게 해준다.  key 값은 고유해야 하며, 일반적으로 항목의 id 값을 사용하는 것이 좋다.  **인덱스를 key로 사용하는 것은 바람직하지 않다.** 특히, 배열이 동적으로 변할 때 문제가 발생할 수 있다.  . |

**댓글 컴포넌트 만들기**

src > comments > Comment.jsx | CommentList.jsx

**전체구조**

**텍스트, 도표, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명**

**Comment.jsx에 추가**

|  |
| --- |
| const styles = {  wrapper: {  margin: 8,  padding: 8,  display: "flex",  flexDirection: "row",  border: "1px solid grey",  borderRadius: 16,  },  imageContainer: {},  image: {  width: 50,  height: 50,  borderRadius: 25,  },  contentContainer: {  marginLeft: 8,  display: "flex",  flexDirection: "column",  justifyContent: "center",  },  nameText: {  color: "black",  fontSize: 16,  fontWeight: "bold",  },  commentText: {  color: "black",  fontSize: 16,  },  }; |

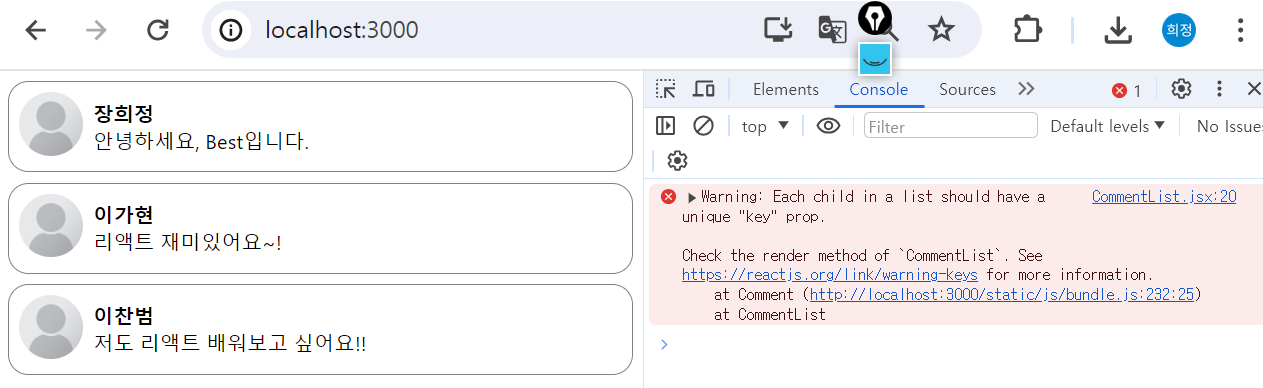
**commentList.jsx에 추가**

|  |
| --- |
| const comments = [  {  name: "장희정",  comment: "안녕하세요, Best Programmer입니다.",  },  {  name: "이가현",  comment: "리액트 재미있어요~!",  },  {  name: "이찬범",  comment: "저도 리액트 배워보고 싶어요!!",  },  ]; |

**이미지 경로**

<img src="./assert/profile.png" alt="" style={styles.image} />

**실행결과**



**위 그림의 오른쪽에 오류가 나는 이유?**

만약, map함수를 사용했을 때 key를 설정하지 않으면 위의 오른쪽 같은 오류가 발생한다.

리액트에서 **key는 컴포넌트 배열을 렌더링했을 때 어떤 원소에 변동이 있었는지 알아내려고 사용한다.** 유동적인 데이터를 다룰 때는 key가 없다면 Virtual DOM을 비교하는 과정에서 리스트를 순차적으로 비교하면서 변화를 감지한다. 하지만 key가 있다면 이 값을 이용하여 어떤 변화가 일어났는지 더욱 빠르게 알아낼 수 있다.

**key값은 언제나 유일 해야 한다.** 따라서 데이터가 가진 고유 값을 key값으로 설정해야 한다.

**단, map의 index로 key를 설정하면 안된다!**

아래의 코드를 실행해 보고 index를 key로 설정했을 때의 문제점을 확인하고 수정해보자.

Ex04\_MapKeyTest.jsx 파일 참고

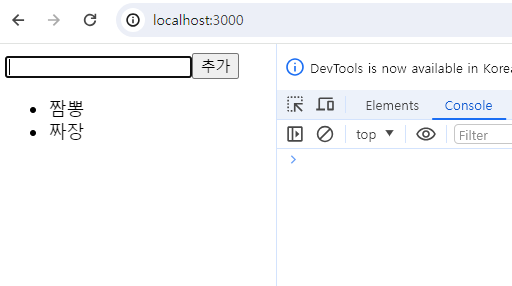
텍스트, 스크린샷, 디스플레이, 번호이(가) 표시된 사진

자동 생성된 설명

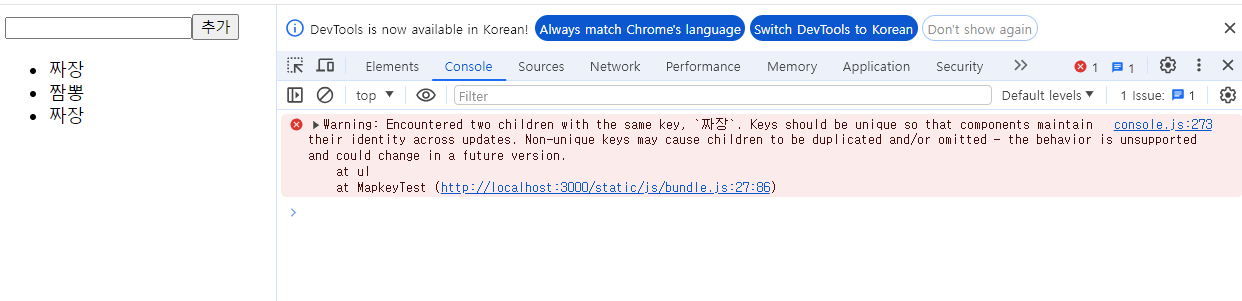
**key가 중복 되었을 때의 문제점을 확인해보자!**

**텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명**



**동일한 메뉴를 추가해보자!**



**동일한 key를 넣으면 경고메시지가 나온다.**

**오류를 해결 해 보자.( Ex05\_MapKeyTest.jsx 파일참고 )**

텍스트, 스크린샷, 번호, 평행이(가) 표시된 사진

자동 생성된 설명