# 3. 기능 구현 준비하기

📄

**Table Of Contents**

UI를 완료했으니 이제 컴포넌트의 기능을 구현합니다.

먼저 컴포넌트별로 어떤 기능을 구현해야 하는지 다시 살펴보겠습니다.

* App 컴포넌트: 할 일 데이터 관리하기
* Header 컴포넌트: 오늘의 날짜 표시
* TodoEditor 컴포넌트: 새로운 할 일 아이템 생성
* TodoList 컴포넌트: 검색에 따라 필터링된 할 일 아이템 렌더링
* TodoItem 컴포넌트: 할 일 아이템의 수정 및 삭제

구현할 컴포넌트의 기능을 잘 살펴보면, 주로 데이터를 추가(생성)하고, 조회하고, 수정하고, 삭제하는 기능으로 이루어져 있음을 알 수 있습니다. 이렇듯 데이터를 다루는 4개의 기능, 즉 추가(Create), 조회(Read), 수정(Update), 삭제(Delete) 기능을 앞글자만 따서 CRUD라고 합니다.

CRUD는 데이터 처리의 기본 기능으로, 웹 서비스라면 기본적으로 갖추고 있어야 합니다.

* Create: 할 일 아이템 생성
* Read: 할 일 아이템 렌더링
* Update: 할 일 아이템 수정
* Delete: 할 일 아이템 삭제

따라서 이번 프로젝트의 기능 구현은 CRUD 순서에 따라 하나씩 진행하겠습니다.

**기초 데이터 설정하기**

먼저 CRUD의 대상인 할 일 아이템부터 생성해야 합니다.

다음과 같이 App.js를 수 정합니다.

**src/App.js**

**Copy**

import { useState } from "react";

(..)

function App() {

const [todo, setTodo] = useState([]); ①

return (

(...)

);

}

export default App;

**①** useState를 이용해 할 일 아이템의 상태를 관리할 State를 만듭니다. 함수 useState에서 인수로 빈 배열을 전달해 State 변수 todo의 기본값을 빈 배열로 초기화합니다.

5장에서 살펴보았듯이 함수 useState는 리액트 훅으로 react 라이브러리에서 불러옵니다. 리액트에서는 보통 리스트 형태의 데이터를 보관할 때 배열을 이용합니다.

State 변수 todo는 [할 일 관리] 앱에서 데이터를 저장하는 배열이면서 동시에 일종의 데이터베이스 역할을 수행합니다. 예를 들어 사용자가 새 할 일 아이템을 만들면, 빈 배열이었던 todo 값은 아이템이 추가된 배열로 업데이트됩니다. 이는 삭제, 수정 모두 동일합니다.

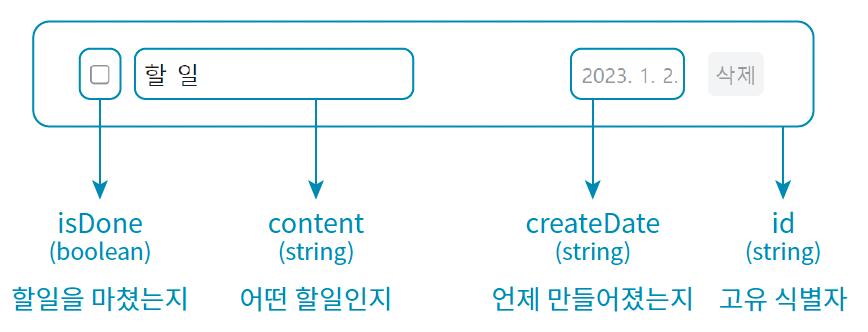
**데이터 모델링하기**

자바스크립트에서는 보통 현실의 사물이나 개념을 표현할 때 객체를 사용합니다. 현실의 사물은 일반적으로 여러 속성을 동시에 가지고 있기 때문입니다. 예를 들어 저자를 객체로 표현한다면 다음과 같습니다.

**Copy**

let author = { name : "이정환b", gender : "male"}

이렇게 현실의 사물이나 개념을 프로그래밍 언어의 객체와 같은 자료구조로 표현 하는 행위를 ‘데이터 모델링’이라고 합니다. 그렇다면 데이터 모델링은 왜 하는 걸 까요? 이유는 간단합니다. [할 일 관리] 앱의 ‘할 일’처럼 현실 세계의 사물이나 개념을 프로그래밍 언어로 표현하고 다뤄야 하기 때문입니다.

[할 일 관리] 앱의 할 일 아이템 데이터 모델링

위 그림에서는 [할 일 관리] 앱에서 만들 할 일 아이템의 요소를 세부적으로 나누어 표시하였습니다. 하나의 할 일 아이템에는 일의 완료 여부, 일의 종류, 생성 날짜 등 3가지 정보가 담겨 있습니다. 세 요소는 각각 isDone, content, createdDate 라는 별도의 이름으로 구분합니다.

또한 할 일 아이템에는 페이지에 렌더링하지는 않지만, id라는 고유 식별자가 있습니다. 이는 데이터베이스를 다뤄본 독자라면 금방 아실 겁니다. 모든 아이템에는 해당 아이템을 구별하기 위한 고유한 식별자가 필요하기 때문입니다. 고유 식별자가 없으면 특정 아이템을 삭제하거나 수정하는 등의 연산이 불가능합니다. 모델링한 정보를 토대로 할 일 아이템을 자바스크립트 객체로 만들면 다음과 같습니다.

**Copy**

{

id: 0, ①

isDone: false, ②

content: "React 공부하기", ③

createdDate: new Date().getTime(), ④

}

**①** id는 특정 아이템을 식별하는 고유한 값입니다. 간단하게는 0부터 시작해 아이템을 추가할 때마다 1씩 늘어나도록 id에 값을 부여할 수 있습니다.

**②** isDone은 불리언 자료형으로 현재 상황에서 할 일이 완료 되었는지 여부를 확인할 때 이용합니다.

**③** content는 할 일이 무엇인지 알려주는 문자열입니다.

**④** createDate는 할 일의 생성 시간입니다. new Date()로 Date 객체를 만들고 getTime 메서드를 이용해 이 객체를 타임 스탬프값으로 변환합니다. 타임 스탬프값으로 시간을 저장하면 보관할 데이터의 양이 대폭 줄어듭니다.

데이터를 모델링하는 이유는 데이터를 어떻게 관리할지 생각하기 위함입니다. 모델링 과정이 잘못되면 작업 과정에서 큰 문제가 생길 수 있습니다. 문제가 발생하면 데이터를 관리하는 모든 과정을 수정하게 되는데, 아예 프로젝트를 처음부터 다시 시작하는 상황도 발생합니다.

혼자 개발하는 프로젝트라면 시간을 더 쏟으면 괜찮겠지만, 여러 팀원과 협업하는 프로젝트라면 문제가 더 심각해집니다. 따라서 모델링은 반드시 데이터 관리 프로그램을 구현하기 전에 노트나 메모장 등에 적어 보면서 코드를 작성해야 합니다.

많은 시간을 들여서라도 데이터 모델링의 완성도를 높이는 게 프로젝트의 시간이나 비용을 줄이는 데 큰 도움이 됩니다.

**목 데이터 설정하기**

목(Mock) 데이터란 모조품 데이터라는 뜻입니다. 기능을 완벽히 구현하지 않은 상태에서 테스트를 목적으로 사용하는 데이터입니다. 임시 데이터라 표현하기도 합니다.

기능을 아직 개발하지 않아 데이터가 없는 상황일 때 목 데이터를 사용합니다. 임시 데이터 역할을 하는 목 데이터가 있으면, 데이터 관리 기능 개발이 한결 수월 해집니다. App.js에서 목 데이터를 다음과 같이 작성합니다.

**src/App.js**

**Copy**

(...)

const mockTodo = [ ①

{

id: 0,

isDone: false,

content: "React 공부하기",

createdDate: new Date().getTime(),

},

{

id: 1,

isDone: false,

content: "빨래 널기",

createdDate: new Date().getTime(),

},

{

id: 2,

isDone: false,

content: "노래 연습하기",

createdDate: new Date().getTime(),

},

];b

function App() {

const [todo, setTodo] = useState(mockTodo); ②

b

return (

(...)

);

}

export default App;

**①** 3개의 객체를 저장하는 배열 목 데이터를 만듭니다. 이 배열에 저장된 객체는 각각 다른 할 일 아이템입니다.

**②** State 변수 todo의 기본값으로 목 데이터를 전달합니다.

✋🏼

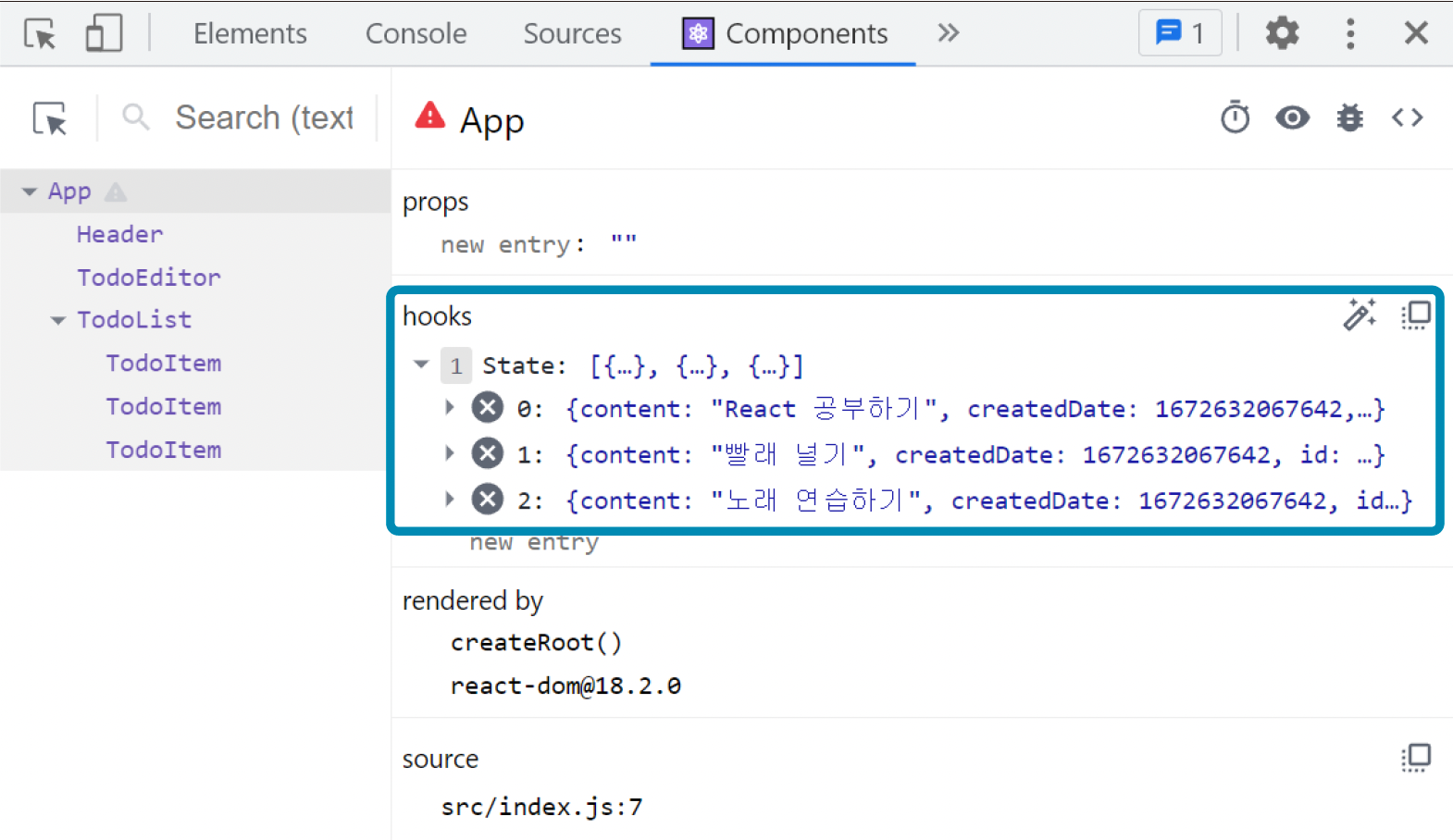
이 시점에서 비주얼 스튜디오 코드에 “(변수명) is assigned a value but never used” 라는 경고 메시지가 나옵니다. 선언된 변수를 어디에서도 사용하지 않을 때 발생하는 경고로 오류는 아닙니다. 앞으로도 실습 과정에서 이 메시지가 나오면 무시해도 괜찮습니다.

아직 TodoList 컴포넌트에 목 데이터를 전달하지 않았기 때문에 데이터를 페이지에 렌더링하지는 않습니다. 따라서 지금은 리액트 개발자 도구를 이용해 데이터가 잘 설정되는지 확인해야 합니다.

개발자 도구에서 [Components] 탭을 엽니다. App 컴포넌트의 hooks 항목에서 State의 값을 확인합니다.

Component 탭이 없으면 크롬 웹스토어에서 React Devloper Tools 추가



[Components] 탭에서 목 데이터 확인하기

위 그림과 같이 App를 클릭해 작성한 목 데이터가 제대로 나타나는지 확인합니다. 빈 배열만 표시된다면 페이지를 새로고침하여 App 컴포넌트를 다시 마운트한 다음 데이터를 확인합니다.

위 그림과 같이 3개의 목 데이터가 나온다면 정상적으로 동작하는 겁니다.

할 일 아이템 요소 가운데 createDate에 저장한 타임 스탬프값은 이 책을 집필하는 시점의 값이므로 여러분과 다르게 나온다는 점에 유의하기 바랍니다.