



2회차 Challenge Mission - 노진현

1. localStorage에 todoData 값을 담아서 페이지를 refresh 해도 todoData가 계속 남아 있을 수 있게 해 주세요.

▼ localStorage에 대해서

localStorage란?

- localStorage는 영구적으로 정보를 브라우저에 보존하고자 할 때 사용한다.
- localStorage는 물리적으로 저장되며, 사용자가 직접 데이터를 삭제하기 전까지 영구적으로 보관된다.
- 애플리케이션 전역에 접근 가능하다.
- storage에 저장된 데이터는 모두 문자열만 사용 가능하기 때문에 다른 타입 데이터 사용시에는 JSON 형태로 읽고 써야 함.
- 가급적 1MB 이상의 큰 데이터를 쓰는 것은 피하는 것이 좋고, 표준 스펙상으로는 5MB를 최대 용량으로 권장한다.

Chrome localStorage 경로

`~/Library/Application Support/Google/Chrome/<Profile>/Local Storage/`

localStorage 읽고 쓰고 삭제하기


- 읽기 `getItem(key)`
- 쓰기 `setItem(key, value)`

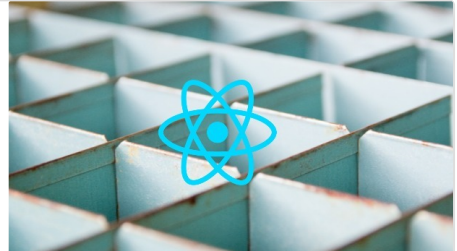
- 삭제하기 `removeItem(key)`

참고자료

How to use localStorage with React | JSON World


For a long time, cookies were the main way to store information at the browser(client side). They were used to record stateful elements like shopping cart items or options

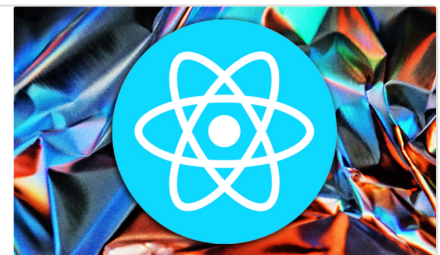
 <https://jsonworld.com/demo/how-to-use-localStorage-with-reactjs>



Using localStorage with React Hooks - LogRocket Blog

localStorage is one of the two mechanisms of a browser's web storage. It allows users to save data as key-value pairs in the browser for later use. Unlike the sessionStorage mechanism,

 <https://blog.logrocket.com/using-localstorage-react-hooks/>



문제 해결하기

1. `useEffect` 를 사용해서 `todoData`가 업데이트 될 때마다 `localStorage`에 저장하기

▼ `useEffect`란?

`useEffect`란?

`useEffect` 가 하는 일은 무엇일까요? `useEffect` Hook을 이용하여 우리는 React에게 컴포넌트가 렌더링 이후에 어떤 일을 수행해야하는 지를 말합니다. React는 우리가 넘긴 함수를 기억했다가(이 함수를 'effect'라고 부릅니다) DOM 업데이트를 수행한 이후에 불러낼 것입니다. 위의 경우에는 effect를 통해 문서 타이틀을 지정하지만, 이 외에도 데이터를 가져오거나 다른 명령형(imperative) API를 불러내는 일도 할 수 있습니다.

`useEffect` 를 컴포넌트 안에서 불러내는 이유는 무엇일까요? `useEffect` 를 컴포넌트 내부에 둬으로써 effect를 통해 `count` state 변수(또는 그 어떤 prop에도)에 접근할 수 있게 됩니다. 함수 범위 안에 존재하기 때문에 특별한 API 없이도 값을 얻을 수 있는 것입

니다. Hook은 자바스크립트의 클로저를 이용하여 React에 한정된 API를 고안하는 것보다 자바스크립트가 이미 가지고 있는 방법을 이용하여 문제를 해결합니다.

useEffect는 렌더링 이후에 매번 수행되는 걸까요? 네, 기본적으로 첫번째 렌더링과 이후의 모든 업데이트에서 수행됩니다.(나중에 effect를 필요에 맞게 수정하는 방법에 대해 다룰 것입니다.) 마운팅과 업데이트라는 방식으로 생각하는 대신 effect를 렌더링 이후에 발생하는 것으로 생각하는 것이 더 쉬울 것입니다. React는 effect가 수행되는 시점에 이미 DOM이 업데이트되었음을 보장합니다.

```
useEffect(() => {
  localStorage.setItem("todoData", JSON.stringify(todoData));
}, [todoData]);
```

useEffect는 페이지가 처음 렌더링 되고 난 후 무조건 한 번 실행됩니다. 또한 **useEffect** 두 번째 인자로 넘겨주는 배열인 **useState**의 값이 업데이트 되면 실행됩니다.

즉, **todoData**의 값이 변화할 때마다 함께 실행될 수 있기 때문에, **localStorage**에 값을 저장하는 **setItem** 함수를 이곳에 작성하였습니다.

localStorage는 JSON 문자열만 취급할 수 있기 때문에 배열형태의 **todoData** 값을 **JSON.stringify** 메서드를 활용하여 JSON 문자열로 변환합니다.

2. todoData 초기값을 localStorage 값으로 수정하기

수정 전 **todoData** 초기값 = 빈 배열

```
const [todoData, setTodoData] = useState([]);
```

수정 후 **todoData** 초기값

```
const [todoData, setTodoData] = useState(()=>{
  const localData = localStorage.getItem("todoData");
  const initialData = JSON.parse(localData);
  return initialData ? initialData : [];
});
```

useState의 인자로 함수를 생성하고 반환 값으로 **localStorage**에 저장된 값을 전달합니다.

