



# useDebugValue

useDebugValue 는 [React DevTools](#)에서 커스텀 훅에 라벨을 추가할 수 있게 해주는 React Hook입니다.

```
useDebugValue(value, format?)
```

- [레퍼런스](#)

- [useDebugValue\(value, format?\)](#)

- [사용법](#)

- [커스텀 Hook에 라벨 추가하기](#)
- [디버그 값의 포매팅 지연하기](#)

---

## 레퍼런스

### useDebugValue(value, format?)

읽을 수 있는 디버그 값을 표시하기 위해 [커스텀 Hook](#)의 최상위 레벨에서 useDebugValue 를 호출하세요.

```
import { useDebugValue } from 'react';

function useOnlineStatus() {
  // ...
  useDebugValue(isOnline ? 'Online' : 'Offline');
  // ...
}
```

아래에서 더 많은 예시를 확인해 보세요.

## 매개변수

- `value`: React DevTools에 표시하고 싶은 값입니다. 어떤 타입이든 될 수 있습니다.
- **선택사항** `format`: 포매팅 함수입니다. 컴포넌트가 검사될 때, React DevTools는 `value` 를 인자로 포매팅 함수를 호출하고, 포매팅 함수가 반환한 포매팅 된 값을 표시합니다. (포매팅 된 값은 어떤 타입이든 될 수 있습니다.) 포매팅 함수를 지정하지 않으면, 원래의 `value` 가 표시됩니다.

## 반환값

`useDebugValue` 는 아무것도 반환하지 않습니다.

## 사용법

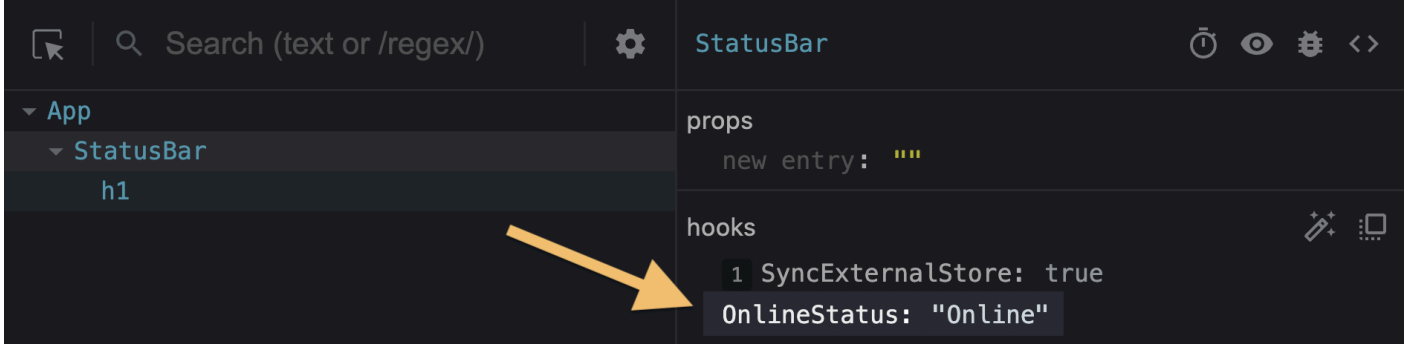
### 커스텀 Hook에 라벨 추가하기

읽을 수 있는 디버그 값 을 표시하기 위해 **커스텀 Hook**의 최상위 레벨에서 `useDebugValue` 를 호출하세요.

```
import { useDebugValue } from 'react';

function useOnlineStatus() {
  // ...
  useDebugValue(isOnline ? 'Online' : 'Offline');
  // ...
}
```

이렇게 하면 `useOnlineStatus` 를 호출하는 컴포넌트를 검사할 때, `OnlineStatus:` "Online" 와 같은 라벨이 붙습니다.



useDebugValue 를 호출하지 않으면, 기본 데이터(이 예시에서는 true)만 표시됩니다.

App.js useOnlineStatus.js

새로고침 Clear 포크

```
import { useSyncExternalStore, useDebugValue } from 'react';

export function useOnlineStatus() {
  const isOnline = useSyncExternalStore(subscribe, () => navigator.onLine, () => true);
  useDebugValue(isOnline ? 'Online' : 'Offline');
  return isOnline;
}

function subscribe(callback) {
  window.addEventListener('online', callback);
  window.addEventListener('offline', callback);
  return () => {
```

자세히 보기

## ☑ 중요합니다!

모든 커스텀 Hook에 디버그 값을 추가하지 마세요. 이는 공유 라이브러리의 일부이고 내부 구조가 복잡하여 검사하기 어려운 커스텀 Hook에 가장 유용합니다.

## 디버그 값의 포매팅 지연하기

`useDebugValue` 의 두 번째 인자로 포매팅 함수를 전달할 수 있습니다.

```
useDebugValue(date, date => date.toString() );
```

포매팅 함수는 디버그 값 을 인자로 받고, 포매팅된 표시 값 을 반환해야 합니다. 컴포넌트가 검사될 때, React DevTools는 이 함수를 호출하고 그 결과를 표시합니다.

이는 컴포넌트가 실제로 검사될 때까지 높은 비용이 들 수 있는 포매팅 로직을 실행하지 않도록 해 줍니다. 예를 들어, `date` 가 `Date` 값이라면, 이는 렌더링마다 `toString()` 을 호출하는 것을 피할 수 있습니다.

이전  
< [useContext](#)

다음  
[useDeferredValue](#) >

uwu?

빠르게 시작하기

설치하기

UI 표현하기

상호작용성 더하기

**State** 관리하기

탈출구

React APIs

React DOM APIs

## 커뮤니티

행동 강령

팀 소개

문서 기여자

감사의 말

## 더 보기

블로그

React Native

개인 정보 보호

약관

