

experimental_taintObjectReference

🧪 실험적 기능

이 API는 실험적이며 React 안정 버전에서는 아직 사용할 수 없습니다.

이 API를 사용하려면 React 패키지를 가장 최신의 실험적인 버전으로 업그레이드해야 합니다.

- `react@experimental`
- `react-dom@experimental`
- `eslint-plugin-react-hooks@experimental`

실험적인 버전의 React에는 버그가 있을 수 있습니다. 프로덕션에서는 사용하지 마세요.

이 API는 [React 서버 컴포넌트](#)에서만 사용할 수 있습니다.

`taintObjectReference`를 사용하면 `user` 객체와 같은 특정한 객체 인스턴스를 클라이언트 컴포넌트로 전송하는 것을 방지할 수 있습니다.

```
experimental_taintObjectReference(message, object);
```

키, 해시 또는 토큰이 전달되는 것을 방지하는 방법은 [taintUniqueValue](#)를 참고하세요.

• 레퍼런스

- [taintObjectReference\(message, object\)](#)

- **사용법**
 - 사용자 데이터가 의도치 않게 클라이언트로 전달되는 것을 방지하기

레퍼런스

`taintObjectReference(message, object)`

클라이언트로 전달되지 않아야 할 객체를 `taintObjectReference` 와 함께 호출하여 React에 등록합니다.

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

experimental_taintObjectReference(
  '환경 변수는 클라이언트로 전달하지 마세요.',
  process.env
);
```

아래 예시를 참고하세요.

매개변수

- `message`: 객체가 클라이언트 컴포넌트로 전달될 때 표시할 메시지. 객체가 클라이언트 컴포넌트로 전달될 때 발생하는 오류 객체에 포함되어 나타나는 메시지입니다.
- `object`: 오염(`taint`)될 객체. 함수와 클래스 인스턴스도 `object` 로서 `taintObjectReference` 에 전달될 수 있습니다. 함수와 클래스는 클라이언트 컴포넌트로 전달되지 않도록 이미 막혀있지만 React의 기본 오류 메시지 대신 `message` 에 설정한 메시지를 보여줄 수 있습니다. 타입 배열 `Typed Array`의 인스턴스를 `object` 로서 `taintObjectReference` 에 전달하면 같은 타입 배열의 다른 인스턴스가 오염되지 않습니다.

반환값

`experimental_taintObjectReference` 는 `undefined` 를 반환합니다.

주의 사항

- 오염된 객체를 다시 작성하거나 복제하면 오염되지 않은 객체가 새로 만들어집니다. 새로 만들어진 객체는 민감한 데이터를 포함할 수 있습니다. 예를 들어, 오염된 `user` 객체가 있다고 할 때, `const userInfo = {name: user.name, ssn: user.ssn}` 혹은 `{...user}` 를 실행하면 오염되지 않은 새로운 객체를 작성합니다. `taintObjectReference` 는 객체가 변경되지 않은 상태에서 클라이언트 컴포넌트로 그대로 전달되는 것만 방지합니다.

⚠ 주의하세요!

보안을 오염Tainting에만 의존하지 마세요. 객체를 오염시켰다고 해서 모든 누출 가능성을 막을 수는 없습니다. 예를 들어 오염된 객체를 복제하면 오염되지 않은 새로운 객체가 만들어집니다. 오염된 객체에서 가져온 데이터를 사용하여(예: `{secret: taintedObj.secret}`) 작성된 새 값이나 객체는 오염되지 않습니다. 오염은 한 겹의 보호 장치일 뿐입니다. 보안성이 높은 애플리케이션은 여러 겹의 보호 장치와 잘 설계된 API를 마련해 두고 격리 패턴을 따릅니다.

사용법

사용자 데이터가 의도치 않게 클라이언트로 전달되는 것을 방지하기

클라이언트 컴포넌트에는 민감한 데이터를 담은 객체가 전달되어서는 안 됩니다. 이상적으로, 데이터 가져오기 `Data Fetching` 함수는 현재 사용자가 접근할 수 없는 데이터를 노출하면 안 됩니다. 하지만 리팩토링 도중 가끔 실수가 발생하기도 합니다. 데이터 API에서 사용자 객체를 “오염Taint”시켜서 이러한 실수를 방지할 수 있습니다.

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

export async function getUser(id) {
  const user = await db`SELECT * FROM users WHERE id = ${id}`;
  experimental_taintObjectReference(
    'user 객체 전체를 클라이언트로 전달하지 마세요.' +
    '필요하다면 일부 특정한 프로퍼티만 뽑아서 사용하는 것이 좋습니다.',
```

```
    user,  
  );  
  return user;  
}
```

이제 누군가 이 객체를 클라이언트 컴포넌트로 전달하려고 하면 전달된 오류 메시지와 함께 오류가 발생합니다.

 자세히 살펴보기

데이터 가져오기에서 누출 방지하기

[자세히 보기](#)

< [이전](#)
[use](#)

[experimental_taintUniqueValue](#) > [다음](#)

 Meta Open Source

Copyright © Meta Platforms, Inc

uwu?

React 학습하기

[빠르게 시작하기](#)

[설치하기](#)

[UI 표현하기](#)

[상호작용성 더하기](#)

[State 관리하기](#)

[탈출구](#)

API 참고서

[React APIs](#)

[React DOM APIs](#)

커뮤니티

[행동 강령](#)

[팀 소개](#)

[문서 기여자](#)

[감사의 말](#)

더 보기

[블로그](#)

[React Native](#)

[개인 정보 보호](#)

[약관](#)

