

[API 참고서 >](#)

내장 React API

[Hook](#)과 [컴포넌트](#) 외에도 react 패키지는 컴포넌트를 정의하는데 유용한 몇 가지 API를 가지고 있습니다. 이 페이지는 최신 React API를 모두 나열합니다.

- [act](#) 를 통해 테스트에서 렌더링이나 상호작용을 감싸서 관련된 업데이트가 모두 처리된 뒤에 검증합니다.
- [cache](#) 를 통해 가져온 데이터나 연산의 결과를 캐싱합니다.
- [captureOwnerStack](#) 을 통해 개발 환경에서 현재 Owner Stack을 읽고, 사용 가능한 문자열을 반환합니다.
- [createContext](#) 를 통해 자식 컴포넌트들에게 전달할 수 있는 컨텍스트를 정의하고 제공합니다. 보통 [useContext](#) 와 함께 사용합니다.
- [lazy](#) 를 통해 컴포넌트가 처음 렌더링될 때까지 해당 컴포넌트의 코드를 로딩하는 것을 지연합니다.
- [memo](#) 를 통해 동일한 Props일 경우 컴포넌트가 다시 렌더링되지 않도록 최적화합니다. 주로 [useMemo](#), [useCallback](#) 과 함께 사용합니다.
- [startTransition](#) 을 통해 상태 업데이트를 “덜 긴급한 작업”으로 표시하여 UI의 반응성을 유지합니다. [useTransition](#) 과 유사합니다.
- [use](#) 는 Promise나 Context와 같은 데이터를 참조하는 React Hook입니다.
- [taintObjectReference](#) 를 통해 user 객체와 같은 특정한 객체 인스턴스를 클라이언트 컴포넌트로 전송하는 것을 방지합니다.
- [taintUniqueValue](#) 를 통해 패스워드, 키 또는 토큰과 같은 고유 값을 클라이언트 컴포넌트로 전송하는 것을 방지합니다.
- [addTransitionType](#) 를 통해, 트랜지션의 발생한 원인을 상세히 나타냅니다.

Resource APIs

Resource를 State의 일부로 포함하지 않고도 컴포넌트에서 Resource에 액세스할 수 있습니다. 예를 들어, 컴포넌트는 Promise에서 메시지를 읽거나 Context에서 스타일 정보를 읽을 수 있습

니다.

Resource에서 값을 읽으려면 다음 API를 사용하세요.

- `use` 를 사용하면 `Promise`나 `Context`와 같은 Resource의 값을 읽을 수 있습니다.

```
function MessageComponent({ messagePromise }) {  
  const message = use(messagePromise);  
  const theme = use(ThemeContext);  
  // ...  
}
```

이전

<ViewTransition>

다음

act

 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

더 보기

블로그

React Native

개인 정보 보호

