



error-boundaries

Validates usage of Error Boundaries instead of try/catch for errors in child components.

Rule Details

Try/catch blocks can't catch errors that happen during React's rendering process. Errors thrown in rendering methods or hooks bubble up through the component tree. Only [Error Boundaries](#) can catch these errors.

Invalid

Examples of incorrect code for this rule:

```
// ❌ Try/catch won't catch render errors
function Parent() {
  try {
    return <ChildComponent />; // If this throws, catch won't help
  } catch (error) {
    return <div>Error occurred</div>;
  }
}
```

Valid

Examples of correct code for this rule:

```
// ✅ Using error boundary
function Parent() {
  return (
```

```

    <ErrorBoundary>
      <ChildComponent />
    </ErrorBoundary>
  );
}

```

Troubleshooting

Why is the linter telling me not to wrap use in try / catch?

The `use` hook doesn't throw errors in the traditional sense, it suspends component execution. When `use` encounters a pending promise, it suspends the component and lets React show a fallback. Only Suspense and Error Boundaries can handle these cases. The linter warns against `try / catch` around `use` to prevent confusion as the `catch` block would never run.

```

// ❌ Try/catch around `use` hook
function Component({promise}) {
  try {
    const data = use(promise); // Won't catch - `use` suspends, not throws
    return <div>{data}</div>;
  } catch (error) {
    return <div>Failed to load</div>; // Unreachable
  }
}

```

```

// ✅ Error boundary catches `use` errors
function App() {
  return (
    <ErrorBoundary fallback={<div>Failed to load</div>}>
      <Suspense fallback={<div>Loading...</div>}>
        <DataComponent promise={fetchData()} />
      </Suspense>
    </ErrorBoundary>
  );
}

```

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

