unsupported-syntax

Validates against syntax that React Compiler does not support. If you need to, you can still use this syntax outside of React, such as in a standalone utility function.

Rule Details

React Compiler needs to statically analyze your code to apply optimizations. Features like `eval` and `with` make it impossible to statically understand what the code does at compile time, so the compiler can't optimize components that use them.

Invalid

Examples of incorrect code for this rule:

```
// ❌ Using eval in component
function Component({ code }) {
  const result = eval(code); // Can't be analyzed
  return <div>{result}</div>;
}

// ❌ Using with statement
function Component() {
  with (Math) { // Changes scope dynamically
    return <div>{sin(PI / 2)}</div>;
  }
}

// ❌ Dynamic property access with eval
function Component({propName}) {
  const value = eval(`props.${propName}`);
  return <div>{value}</div>;
```

```
}
```

Valid

Examples of correct code for this rule:

```
// ✅ Use normal property access
function Component({propName, props}) {
  const value = props[propName]; // Analyzable
  return <div>{value}</div>;
}

// ✅ Use standard Math methods
function Component() {
  return <div>{Math.sin(Math.PI / 2)}</div>;
}
```

Troubleshooting

I need to evaluate dynamic code

You might need to evaluate user-provided code:

```
// ❌ Wrong: eval in component
function Calculator({expression}) {
  const result = eval(expression); // Unsafe and unoptimizable
  return <div>Result: {result}</div>;
}
```

Use a safe expression parser instead:

```
// ✅ Better: Use a safe parser
import {evaluate} from 'mathjs'; // or similar library

function Calculator({expression}) {
```

```
  const [result, setResult] = useState(null);

  const calculate = () => {
    try {
      // Safe mathematical expression evaluation
      setResult(evaluate(expression));
    } catch (error) {
      setResult('Invalid expression');
    }
  };

  return (
    <div>
      <button onClick={calculate}>Calculate</button>
      {result && <div>Result: {result}</div>}
    </div>
  );
}
```

중요합니다!

Never use `eval` with user input - it's a security risk. Use dedicated parsing libraries for specific use cases like mathematical expressions, JSON parsing, or template evaluation.

이전
static-components

다음
use-memo

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