




 Secrets


 ABAP


 Apex


 C


 C++


 CloudFormation


 COBOL


 C#


 CSS


 Flex


 Go


 HTML


 Java


 **JavaScript**


 Kotlin


 Objective C


 PHP


 PL/I


 PL/SQL


 Python


 RPG


 Ruby


 Scala


 Swift


 Terraform


 Text


 TypeScript

 T-SQL

 VB.NET

 VB6

 XML



## JavaScript static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVASCRIPT code

All rules285

Vulnerability29

Bug62

Security Hotspot43

Code Smell151

Quick Fix41

Tags ▾

Search by name... 🔍

Boolean checks should not be inverted

Code Smell

Deprecated APIs should not be used

Code Smell

Wrapper objects should not be used for primitive types

Code Smell

Multiline string literals should not be used

Code Smell

Local variables should not be declared and then immediately returned or thrown

Code Smell

Unused local variables and functions should be removed

Code Smell

Function call arguments should not start on new lines

Code Smell

"switch" statements should have at least 3 "case" clauses

Code Smell

A "while" loop should be used instead of a "for" loop

Code Smell

Unnecessary imports should be removed

Code Smell

Return of boolean expressions should not be wrapped into an "if-then-else" statement

Code Smell

Using intrusive permissions is security-sensitive

Analyze your code

Security Hotspot

Major ?

cwe privacy owasp

**Powerful features** are browser features (geolocation, camera, microphone ...) that can be accessed with JavaScript API and may require a permission granted by the user. These features can have a high impact on privacy and user security thus they should only be used if they are really necessary to implement the critical parts of an application.

This rule highlights intrusive permissions when requested with **the future standard (but currently experimental) web browser query API** and specific APIs related to the permission. It is highly recommended to customize this rule with the permissions considered as intrusive in the context of the web application.

Ask Yourself Whether

- Some powerful features used by the application are not really necessary.
- Users are not clearly informed why and when powerful features are used by the application.

You are at risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

- In order to respect user privacy it is recommended to avoid using intrusive powerful features.

Sensitive Code Example

When using **geolocation API**, Firefox for example retrieves personal information like nearby wireless access points and IP address and sends it to the default geolocation service provider, **Google Location Services**:

```
navigator.permissions.query({name:"geolocation"}).then(function(p){ // Sensitive: geolocation is a powerful feature with high impact on privacy...});

navigator.geolocation.getCurrentPosition(function(position){ console.log("coordinates x="+position.coords.latitude+" and y="+position.coords.longitude); // Sensitive: geolocation is a powerful feature with high impact on privacy...});
```





Compliant Solution

If geolocation is required, always explain to the user why the application needs it and prefer requesting an approximate location when possible:

```
<html>
<head>
  <title>
    Retailer website example
  </title>
</head>
<body>
  Type a city, street or zip code where you want to retrieve location
  <form method=post>
```

https://rules.sonarsource.com/javascript/RSPEC-5604

1/2

Boolean literals should not be used in comparisons
 Code Smell
Extra semicolons should be removed
 Code Smell
Class names should comply with a naming convention
 Code Smell
Track uses of "TODO" tags
 Code Smell
Web SQL databases should not be used

```
<input type=text value="New York"> <!-- Compliant -->
</form>
</body>
</html>
```

See

- [OWASP Top 10 2021 Category A1](#) - Broken Access Control
- [OWASP Web Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [MITRE, CWE-250](#) - Execution with Unnecessary Privileges
- [MITRE, CWE-359](#) - Exposure of Private Information
- [W3C](#) - Permissions
- [Mozilla](#) - Does Firefox share my location with websites?

Available In:

