Java static code analysis

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

Nested "enum"s should not be declared static
Code Smell

"catch" clauses should do more than rethrow
Code Smell

Mutable fields should not be "public static"
Code Smell

The diamond operator ("<>") should be used
Code Smell

"finalize" should not set fields to "null"
Code Smell

Subclasses that add fields should override "equals"
Code Smell

Catches should be combined
Code Smell

Methods of "Random" that return floating point values should not be used in random integer generation
Code Smell

Parsing should be used to convert "Strings" to primitives
Code Smell

Classes should not be empty
Code Smell

Fields in non-serializable classes should not be "transient"
Code Smell

Boolean checks should not be inverted

Enabling file access for WebViews is security-sensitive

Analyze your code

Security Hotspot Major cwe owasp android

WebViews can be used to display web content as part of a mobile application. A browser engine is used to render and display the content. Like a web application a mobile application that uses WebViews can be vulnerable to Cross-Site Scripting if untrusted code is rendered.

If malicious JavaScript code in a WebView is executed this can leak the contents of sensitive files when access to local files is enabled.

Ask Yourself Whether

- No local files have to be accessed by the Webview.
- The WebView contains untrusted data that could cause harm when rendered.

There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

It's recommended to disable access to local files for WebViews unless it is necessary. In the case of a successful attack through a Cross-Site Scripting vulnerability the attackers attack surface decreases drastically if no files can be read out.

Sensitive Code Example

```
import android.webkit.WebView;

WebView webView = (WebView) findViewById(R.id.webview);
webView.getSettings().setAllowFileAccess(true); // Sensitive
webView.getSettings().setAllowContentAccess(true); // Sensit
```

Compliant Solution

```
import android.webkit.WebView;

WebView webView = (WebView) findViewById(R.id.webview);
webView.getSettings().setAllowFileAccess(false);
webView.getSettings().setAllowContentAccess(false);
```

See

- OWASP Top 10 2021 Category A3 - Injection
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- OWASP Top 10 2017 Category A7 - Cross-Site Scripting (XSS)
- MITRE, CWE-79 - Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Available In:

sonarcloud sonarqube

Code Smell

Redundant casts should not be used

Code Smell

"@Deprecated" code should not be used

Code Smell

"toString()" should never be called on a String object

Code Smell

Annotation repetitions should not be wrapped