- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Kubernetes
- **Objective C**
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML

Objective C static code analysis

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

| All rules **315** | 🔒 Vulnerability **10** | 🐛 Bug **75** | Security Hotspot **18** | Code Smell **212** | Quick Fix **13** |

Tags ⌄   Search by name...

"memset" should not be used to delete sensitive data

🔒 Vulnerability

POSIX functions should not be called with arguments that trigger buffer overflows

🔒 Vulnerability

Function-like macros should not be invoked without all of their arguments

🐛 Bug

The address of an automatic object should not be assigned to another object that may persist after the first object has ceased to exist

🐛 Bug

"pthread_mutex_t" should be unlocked in the reverse order they were locked

🐛 Bug

"pthread_mutex_t" should be properly initialized and destroyed

🐛 Bug

"pthread_mutex_t" should not be consecutively locked or unlocked twice

🐛 Bug

Functions with "noreturn" attribute should not return

🐛 Bug

"memcmp" should only be called with pointers to trivially copyable types with no padding

🐛 Bug

Stack allocated memory and non-owned memory should not be freed

🐛 Bug

Closed resources should not be accessed

🐛 Bug

Dynamically allocated memory should be released

🐛 Bug

"default" clauses should be first or last

Analyze your code

🔧 Code Smell   🔺 Critical ?   🏷 based-on-misra misra-c2004 misra-c2012

switch can contain a default clause for various reasons: to handle unexpected values, to show that all the cases were properly considered.

For readability purpose, to help a developer to quickly find the default behavior of a switch statement, it is recommended to put the default clause at the end of the switch statement. This rule raises an issue if the default clause is not the first or the last one of the switch's cases.

Noncompliant Code Example

```
switch (param) {
  case 0:
    doSomething();
    break;
  default: // default clause should be the first or last one
    error();
    break;
  case 1:
    doSomethingElse();
    break;
}
```

Compliant Solution

```
switch (param) {
  case 0:
    doSomething();
    break;
  case 1:
    doSomethingElse();
    break;
  default:
    error();
    break;
}
```

See

- MISRA C:2004, 15.3 - The final clause of a switch statement shall be the default clause
- MISRA C++:2008, 6-4-6 - The final clause of a switch statement shall be the default-clause
- MISRA C:2012, 16.4 - Every switch statement shall have a default label
- MISRA C:2012, 16.5 - A default label shall appear as either the first or the last switch label of a switch statement

Available In:

sonarcloud | sonarqube Developer Edition

Freed memory should not be used

🐞 Bug

Memory locations should not be released more than once

🐞 Bug

Memory access should be explicitly bounded to prevent buffer overflows

🐞 Bug

Printf-style format strings should not lead to unexpected behavior at runtime

🐞 Bug

Recursion should not be infinite

🐞 Bug

Resources should be closed

🐞 Bug

Hard-coded credentials are security-sensitive

🛡 Security Hotspot

"goto" should jump to labels declared later in the same function

☢ Code Smell

Only standard forms of the "defined" directive should be used

☢ Code Smell

Switch labels should not be nested inside non-switch blocks

☢ Code Smell