C++ static code analysis

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

All rules 578 | Vulnerability 13 | Bug 111 | Security Hotspot 18 | Code Smell 436 | Quick Fix 68

Tags

Search by name...

Reserved identifiers and functions in the C standard library should not be defined or declared

Analyze your code

Code Smell | Blocker | based-on-misra bad-practice cert

Defining or declaring identifiers with reserved names may lead to undefined behavior. Similarly, defining macros, variables or functions/methods with the same names as functions from the C standard library is likely to lead to unexpected results.

Additionally, such identifiers have the potential to thoroughly confuse people who are unfamiliar with the code base, possibly leading them to introduce additional errors. Therefore reserved words and the names of C standard library functions should not be used as identifiers.

This rule applies to:

- `defined`
- C standard library function names
- identifiers that contain two consecutive underscores
- identifiers that begin with an underscore, followed by an uppercase letter
- identifiers in the global namespace that start with an underscore

Noncompliant Code Example

```
#ifndef _MY_FILE
#define _MY_FILE   // Noncompliant: starts with '_'

#define FIELD__VAL(field) ##field // Noncompliant: contains "

int free(void *pArg, int len) {  // Noncompliant: free is a s
  int __i; // Noncompliant: starts with "__"
  //...
}
#endif
```

Compliant Solution

```
#ifndef MY_FILE
#define MY_FILE

#define FIELD_VAL(field) ##field

int clean(void *pArg, int len) {
  int i;
  //...
}
#endif
```

See

- MISRA C:2004, 20.1 - Reserved identifiers, macros and functions in the standard library, shall not be defined redefined or undefined.
- MISRA C++:2008, 17-0-1 - Reserved identifiers, macros and functions in the standard library shall not be defined, redefined, or undefined.
- MISRA C:2012, 21.2 - A reserved identifier or macro name shall not be declared
- CERT, DCL37-C. - Do not declare or define a reserved identifier
- CERT, DCL51-CPP. - Do not declare or define a reserved identifier

Available In:

sonarlint sonarcloud sonarqube Developer Edition

Bug

"std::move" and "std::forward" should not be confused

Bug

A call to "wait()" on a "std::condition_variable" should have a condition

Bug

A pointer to a virtual base class shall only be cast to a pointer to a derived class by means of dynamic_cast

Bug

Functions with "noreturn" attribute should not return

Bug

RAII objects should not be temporary

Bug

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

Bug

"memcpy", "memmove", and "memset" should only be called with pointers to trivially copyable types

Bug

"std::auto_ptr" should not be used

Bug

Destructors should be "noexcept"

Bug