C++ static code analysis

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

"switch" statements should cover all cases

Analyze your code

Code Smell   Major   suspicious

For completeness, a `switch` over the values of an `enum` must either address each value in the `enum` or contain a `default` case. `switch` statements that are not over `enum` must end with a `default` case.

This rule is a more nuanced version of {rule:cpp:S131}. Use {rule:cpp:S131} if you want to require a `default` case for every `switch` even if it already handles all enumerators of an `enum`. Otherwise, use this rule.

Noncompliant Code Example

```
typedef enum {APPLE, GRAPE, KIWI} fruit;

void example(fruit f, int i) {
  switch (f) {  // Noncompliant; no case for KIWI
    case APPLE:
      //...
    case GRAPE:
      //...
    case 3: // Noncompliant; case value not in enum
      // ...
  }

  switch (i) { // Noncompliant; no default
    case 0:
      // ...
    case 1:
      // ...
  }
}
```

Compliant Solution

```
typedef enum {APPLE, GRAPE, KIWI} fruit;

void example(fruit f) {
  switch (f) {
    case APPLE:
      //...
    case GRAPE:
      //...
    default:
      // ...
  }

  switch (i) {
    case 0:
      // ...
    case 1:
      // ...
    default:
      // ...
  }
}
```

or

```
typedef enum {APPLE, GRAPE, KIWI} fruit;
```

```
void example(fruit f) {
  switch (f) {
    case APPLE:
      //...
    case GRAPE:
      //...
    case KIWI:
      //...
  }

  switch (i) {
    case 0:
    case 1:
      // ...
    default:
      // ...
  }
}
```

See

- C++ Core Guidelines - Enum.2 - Use enumerations to represent sets of related named constants

See Also

- {rule:cpp:S131}

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