rules.sonarsource.com

C++ static code analysis: "std::endl" should not be used

2 minutes

When injecting `std::endl` into an output stream, two things happen:

- An end of line character `'\n'` is added to the stream
- The stream is flushed

  In many situations, you don't need the stream to be flushed: It takes some time, and additionally, the stream is also flushed automatically in several circumstances:

- When the stream is closed
- In the case of `std::cout`, each time an input is read on `std::cin` or an output is written on `std::cerr`
- In the case of `std::cerr`, each output is immediately written, the is no need to flush

  Therefore, if your only goal is to add an end of line, `'\n'` is usually more efficient than `std::endl`. If you do want to flush, you can be explicit and inject `std::flush` into the stream, or call the `flush` member function on the stream.

Noncompliant Code Example

```
void f() {
  cout << "Hello world!" << endl << endl << "How are you?" << endl;
// Noncompliant, 3 useless flushes
  string s;
  cin >> s;
  cout << "Starting long operation now..." << endl; // Noncompliant,
flushing is useful, but not explicit enough
  longOperation();
  cout << "Long operation is done" << endl; // Noncompliant
}
```

Compliant Solution

```
void f() {
  cout << R"(Hello world!

How are you?
)" << endl;
  // Or
  cout << "Hello world!\n\nHow are you?\n";
  string s;
  cin >> s;
  cout << "Starting long operation now...\n" << flush;
  longOperation();
  cout << "Long operation is done\n";
}
```

See

- C++ Core Guidelines SL.50 - Avoid endl