rules.sonarsource.com

C++ static code analysis: Handlers in a single try-catch or function-try-block for a derived class and some or all of its bases should be ordered most-derived-first

2 minutes

Exceptions handlers (`catch ()`) are evaluated in the order they are written. Once a match is found, the evaluation stops. If there is a handler for a base class followed by a handler for class derived from that base class, the second handler will never trigger: The handler for the base class will match the derived class, and will be the only executed handler. The derived class handler is dead code.

Noncompliant Code Example

```
class BaseException { };
class DerivedException: public BaseException { };

try
{
  // ...
}
catch ( BaseException &b ) // Will catch DerivedException as well
{
  // ...
}
catch ( DerivedException &d ) // Noncompliant, the previous handled effectively hides this one
{
  // Any code here will be unreachable,
}
```

Compliant Solution

```
class BaseException { };
class DerivedException: public BaseException { };

try
{
  // ...
}
catch ( DerivedException &d ) // Compliant
{
  // ...
}
catch ( BaseException  &b ) // Compliant, will be triggered for
BaseException that are not DerivedException
{
  // ...
}
```

See

- MISRA C++:2008, 15-3-6
- CERT, ERR54-CPP. - Catch handlers should order their parameter types from most derived to least derived