rules.sonarsource.com

C++ static code analysis: Lambdas that capture "this" should capture everything explicitly

2-3 minutes

A lambda can only capture local variables. When a lambda is defined within a member function, you may believe that you are capturing a member variable of the current class, but in fact, what you are capturing is `this`. This may be very surprising, and lead to bugs if the lambda is then used after the current object has been destroyed.

Therefore, it's better to be explicit about exactly what is captured as soon as `this` is captured.

If the lambda is used immediately (for instance, called or passed as an argument to `std::sort`), there is no such risk and no issue is raised.

In C++20, capturing `this` via [=] has been deprecated. An issue is raised in that case, even if the lambda is used immediately.

Note: This rule does not apply if the capture list of the lambda contains `*this` (possible since C++17). In that situation, what is captured is not the pointer `this`, but a local copy of the object pointed-to by `this` and any reference to `this` (explicit or implicit) in the lambda body then refers to this local copy (see {rule:cpp:S6016}).

Noncompliant Code Example

```
void useLambda(std::function<int,int> lambda);

class A {
  int i;
  void f(int j) {
    auto l = [=](int k) { return i+j+k;}; // Noncompliant, someone reading the code might believe that i is captured by copy
```

```
    useLambda(l);
  }
};
```

Compliant Solution

```
void useLambda(std::function<int,int> lambda);

class A {
  int i;
  void f(int j) {
    auto l = [this, j](int k) { return i+j+k;}; // It is now clearer that i is
not directly captured
    useLambda(l);
    // auto l = [i, j](int k) { return i+j+k;}; // Would not compile

    auto l2 = [=, *this](int k) { return i+j+k;}; // Compliant, i refers to
the member i of the captured copy
    useLambda(l2);

    auto l3 = [=](int k) { return i+j+k;}; // Compliant because l3 is only
used immediately
    int ijk = l3(i,j,k);
  }
};
```

See

- C++ Core Guidelines F.54 - If you capture `this`, capture all variables explicitly (no default capture)