The move constructor that you have to declare, even though you don’t want anyone to actually call it

devblogs.microsoft.com/oldnewthing/20230612-00

June 12, 2023

Raymond Chen

Blah blah blah C++ return value optimization (RVO), named return value optimization (NRVO), and copy elision.

RVO support was optional in C++11 but became mandatory in C++17. NRVO support remains optional (but recommended).

To allow NRVO in C++17 (or RVO and NRVO in C++11), a move constructor must be available, even though the compiler will not call it if the optimization is employed.

You may have a class that you want to participate in RVO or NRVO, but you also don’t want it to be moved. For example, it may contain a `std::mutex`, which is not movable. But you nevertheless have to declare a move constructor. What can you do?

Declare the move constructor, but don’t implement it.

```
class MyClass
{
public:
  MyClass();

  // Not copyable.
  MyClass(const MyClass&) = delete;

  // Movable only for NRVO purposes (and RVO in C++11).
  // Never implemented.
  MyClass(MyClass&&);

  // Not assignable.
  void operator=(const MyClass&) = delete;
};

MyClass test1()
{
    return MyClass(); // RVO
}

MyClass test2()
{
    MyClass c;
    return c; // NRVO
}

MyClass test3()
{
    MyClass c, d;
    if (some_condition()) {
        return c; // failed NRVO
    } else {
        return d; // failed NRVO
    }
}
```

We declared a move constructor in order to permit RVO in C++11 and NRVO everywhere. The compiler demands to see a move constructor, even though RVO and NRVO ultimately optimize it out.

The first test shows that RVO works. The second test shows that NRVO works.

The `test3` version compiles but does not link: NRVO is not possible given the way that `test3` is written, and the compiler is forced to use the move constructor to move either `c` or `d` into the return value. We declared the move constructor but never implemented it, so we get an unresolved external, which tells us, “Sorry, this object doesn’t support move, even though the tin says that it does. The label on the tin is a lie and exists only to allow the compiler to use RVO and NRVO.”