Is std::exception_ptr nothrow constructible and assignable?

devblogs.microsoft.com/oldnewthing/20210506-00

May 6, 2021

Raymond Chen

When I was writing my series on awaitable signals and coroutine promises, I had to investigate a few details of the language specification, like why `coroutine_handle::resume()` is potentially-throwing. Another detail I had to investigate was whether `std::exception_ptr` was potentially-throwing in its constructors and assignment operators.

The answer is *No*, but you have to chase through the language specification a bit.

The `exception_ptr` is introduced in the C++ language specification under **[propagation]**, and it says that `exception_ptr` is a *NullablePointer*[1] but otherwise says nothing about exceptions.

Chase through to **[nullablepointer.requirements]**, which spells out the various operations that must be supported. It says, for example, that the object must satisfy *Cpp17CopyConstructible* (or *CopyConstructible*, as cppreference.com calls it), but if you follow that definition, there is still no mention of exceptions.

But the magic sentence is right there. You just overlooked it.

Back in the definition of *NullablePointer*, it says,

> 4 No operation which is part of the *NullablePointer* requirements shall exit via an exception.

So there it is. The `exception_ptr`, or more generally, anything that satisfies *NullablePointer*, is nothrow comparable, constructible, assignable, destructible, swappable, and testable.

[1] Starting in C++20, these requirements have been renamed *Cpp17NullablePointer*, *Cpp17CopyConstructible, etc.*

Raymond Chen

Follow