# In C++/WinRT, what happens when I treat an IInspectable as or convert one to a bool

devblogs.microsoft.com/oldnewthing/20191206-00

December 6, 2019

Raymond Chen

Last time, we looked at weirdness in how C++/CX treats hat pointers in a `bool` context. Fortunately, C++/WinRT is much less weird.

The `IInspectable` type supports a conversion to `bool` which tests whether the underlying pointer is null. It also supports comparison against `nullptr` which tests the same thing. And, unlike C++/CX, C++/WinRT uses this conversion for both explicit and contextual conversions.

```
IInspectable p = winrt::box_value(false);
IInspectable q = winrt::box_value(false);

if (p)                    std::cout << 1;
if ((bool)p)              std::cout << 2;
if (static_cast<bool>(p)) std::cout << 3;
if (p == q)               std::cout << 4;
if (p == false)           std::cout << 5;
if (!p)                   std::cout << 6;
if ((bool)p == (bool)q)   std::cout << 7;
```

| Condition | What's happening | Result |
|---|---|---|
| `if (p)` | Tests `p` against `nullptr`. | prints `1` |
| `if ((bool)p)` | Tests `p` against `nullptr`. | prints `2` |
| `if (static_cast<bool>(p))` | Tests `p` against `nullptr`. | prints `3` |
| `if (p == q)` | Compares two objects for identity. | does not print |
| `if (p == false)` | Not allowed (compiler error). | |
| `if (!p)` | Tests `p` against `nullptr`. | does not print |
| `if ((bool)p == (bool)q)` | Tests `p` and `q` against `nullptr`. | prints 7 |

Note that the last case prints `7` but not for the reason you think. It's not doing any unboxing at all. It's just checking whether both variables are non-null.

```
IInspectable t = winrt::box_value(true);
if ((bool)p == (bool)t)   std::cout << 8; // prints 8!
```

**Bonus chatter**: There is a little quirk in the `p == false` case. My understanding is that prior to C++11, `false` was a legal *null pointer constant*, but the rules in C++11 were tightened so that `false` is no longer a null pointer context.

Microsoft's Visual Studio C++ compiler, however, continues to accept `false` as a null pointer constant, even in non-permissive mode. This means that if you're using Microsoft's Visual Studio C++ compiler, the fifth row of the table is slightly different:

| Condition | What's happening | Result |
|---|---|---|
| `if (p == false)` | `false` converted to `IInspectable{ nullptr }` and compared with `p` | does not print |

Raymond Chen

**Follow**