I tried to subscribe to a C++/WinRT event, but my callback was never called

devblogs.microsoft.com/oldnewthing/20250305-00

Raymond Chen March 5, 2025

A customer wanted to subscribe to an event, but they reported that the callback never occurred. They were kind enough to provide a minimal example.

```
#ifndef UNICODE
#define UNICODE
#endif
#include <windows.h>
#include <winrt/Windows.UI.ViewManagement.h>
#include <iostream>

winrt::Windows::UI::ViewManagement::UISettings uiSettings;

winrt::fire_and_forget OnTextScaleFactorChangedAsync(
    [[maybe_unused]] const winrt::UISettings& sender,
    [[maybe_unused]] const winrt::IInspectable& args)
{
    std::cout <> "New scale factor: "
              << uiSettings.TextScaleFactor()
              << std::endl;
}

int main()
{
    winrt::init_apartment();

    uiSettings.TextScaleFactorChanged(
        winrt::auto_revoke,
        OnTextScaleFactorChangedAsync);

    Sleep(INFINITE);

    return 0;
}
```

There are multiple things wrong here.

The first is that the `uiSettings` object is being created as a global object, and global objects are constructed *before* the `main` function begins. This means that we try to create the `uiSettings` before COM is initialized, which is not a great idea.

(But you get away with it because C++/WinRT has special code that detects that you're trying to create Windows Runtime objects without initializing COM, so it secretly initializes COM for you. I do not recommend that you rely on this feature.)

The other problem is the way the event handler is registered:

```
uiSettings.TextScaleFactorChanged(
    winrt::auto_revoke,
    OnTextScaleFactorChangedAsync);
```

The `auto_revoke` parameter selects the overload that returns a revoker object (in this case, a `TextScaleFactorChanged_revoker`), which is an RAII object that revokes the event handler upon destruction. Therefore, this code registers an event handler, and since the revoker was never saved anywhere, it destructed immediately, thereby unregistering the handler.

Now, the `auto_revoke` version of event registration is marked as `[[nodiscard]]` which is a recommendation to the compiler that it produce a diagnostic if the value is not saved anywhere.

We asked the customer to turn on warnings, and how about that, they got a warning that the return value of `TextScaleFactorChanged(winrt::auto_revoke, ...)` was discarded. Once they saved the revoker into a variable, they started receiving the events.

Author

Raymond has been involved in the evolution of Windows for more than 30 years. In 2003, he began a Web site known as The Old New Thing which has grown in popularity far beyond his wildest imagination, a development which still gives him the heebie-jeebies. The Web site spawned a book, coincidentally also titled The Old New Thing (Addison Wesley 2007). He occasionally appears on the Windows Dev Docs Twitter account to tell stories which convey no useful information.