Why am I getting a RPC_E_WRONG_THREAD exception when I'm on the right thread?

devblogs.microsoft.com/oldnewthing/20221104-00

November 4, 2022

Raymond Chen

A customer was having trouble diagnosing a `RPC_E_WRONG_THREAD` exception in their code. The customer's code is written in C++/CX and PPL, but the same principles apply to C++/WinRT and C#, so I'll provide equivalents along the way.

In C++/CX, the `RPC_E_WRONG_THREAD` exception is projected as a `Platform::Wrong-ThreadException`, and they were getting it from an attempt to create a custom class that derives from `Xaml::DependencyObject`:

```
public ref class MyViewModel :
    Windows::UI::Xaml::DependencyObject,
    Windows::UI::Xaml::Data::INotifyPropertyChanged
{
public:
    virtual event Windows::UI::Xaml::Data::
        PropertyChangedEventHandler^ PropertyChanged;

    ...
};

void MyFlyout::OnLoaded()
{
    create_task(ApplicationData::Current->LocalFolder->
                GetFileAsync(L"cache.json"))
    .then([](StorageFile^ file)
    {
        return FileIO::ReadTextAsync(file);
    }).then([this](String^ contents)
    {
        auto json = JsonObject::Parse(contents);
        m_viewModel = ref new MyViewModel(json);
    }).then([this](task<void> t)
    {
        try {
            t.get();
        } catch (Platform::COMException^) {
            // Something went wrong. Start from scratch.
            m_viewModel = ref new MyViewModel();
        }
    });
}

// C++/WinRT equivalent
winrt::fire_and_forget MyFlyout::OnLoaded()
{
    auto lifetime = get_strong();

    try {
        auto file = co_await ApplicationData::Current().LocalFolder().
                        GetFileAsync(L"cache.json");
        auto contents = co_await FileIO::ReadTextAsync(file);
        auto json = JsonObject::Parse(contents);
        m_viewModel = MyViewModel(json);
    } catch (...) {
        // Something went wrong. Start from scratch.
        m_viewModel = MyViewModel();
    }
}

// C# equivalent
async void OnLoaded()
```

```
{
    try {
        var file = await ApplicationData.Current.LocalFolder.
                        GetFileAsync(L"cache.json");
        var contents = await FileIO.ReadTextAsync(file);
        var json = JsonObject.Parse(contents);
        m_viewModel = new MyViewModel(json);
    } catch (...) {
        // Something went wrong. Start from scratch.
        m_viewModel = new MyViewModel();
    }
}
```

The `RPC_E_WRONG_THREAD` exception was coming from the catch block.

The customer concluded that somehow the PPL library (or the C++/WinRT library, or the C# await operator) was catching the exception on the wrong thread, but a closer look at the crash stack showed a different story:

```
KERNELBASE!RaiseFailFastException+0x152
combase!RoFailFastWithErrorContextInternal2+0x4d9
Windows_UI_Xaml!ErrorHelper::ProcessUnhandledError+0xf4
Windows_UI_Xaml!FinalUnhandledErrorDetectedRegistration::
                               OnFinalUnhandledErrorDetected+0xbc
Windows_UI_Xaml!<lambda_...>::operator()+0x1d
...
twinapi_appcore!UnhandledErrorInvokeHelper::Invoke+0x24
twinapi_appcore!CoreApplication::ForwardLocalError+0x79
twinapi_appcore!CoreApplicationFactory::ForwardLocalError+0x77
combase!CallErrorForwarder+0x150
contoso!`_ExceptionHolder::ReportUnhandledError'::catch$1+0x39
ucrtbase!_CallSettingFrame_LookupContinuationIndex+0x20
ucrtbase!__FrameHandler4::CxxCallCatchBlock+0x115
ntdll!RcFrameConsolidation+0x6
contoso!_ExceptionHolder::ReportUnhandledError+0x27
contoso!_ExceptionHolder::~_ExceptionHolder+0x2b
contoso!std::_Ref_count_base::_Decref+0x2c
contoso!std::_Ptr_base<_Task_completion_event>::_Decref+0x12
contoso!std::shared_ptr<_ExceptionHolder>::{dtor}+0x5
contoso!_Task_completion_event::~_Task_completion_event+0x53
contoso!std::_Ref_count_base::_Decref+0x2c
contoso!std::_Ptr_base<_Task_completion_event>::_Decref+0x2f
contoso!std::shared_ptr<_Task_completion_event>::{dtor}+0x2f
...
contoso!__abi_FunctorCapture<<lambda_...>, ...>::Invoke+0x5d
contoso!<lambda_...>::operator()+0x74
contoso!_PPLTaskContextCallbackBridge+0xd
combase!CRemoteUnknown::DoCallback+0x4e
rpcrt4!Invoke+0x73
rpcrt4!Ndr64StubWorker+0xb7c
rpcrt4!NdrStubCall3+0xd2
combase!CStdStubBuffer_Invoke+0x65
combase!<lambda_...>::operator()+0x22
combase!ObjectMethodExceptionHandlingAction<<lambda_...> >+0x4d
combase!InvokeStubWithExceptionPolicyAndTracing+0xda
combase!DefaultStubInvoke+0x257
combase!SyncServerCall::StubInvoke+0x38
combase!StubInvoke+0x2d8
combase!ServerCall::ContextInvoke+0x46b
combase!CServerChannel::ContextInvoke+0x84
combase!DefaultInvokeInApartment+0xc1
combase!ASTAInvokeInApartment+0x85
combase!ComInvokeWithLockAndIPID+0xa13
combase!ThreadDispatch+0x3d5
combase!ModernSTAState::HandleMessage+0x55
combase!ModernSTAWaitContext::Wait+0x632
combase!ModernSTAWaitInNewContext+0xd4
combase!ModernSTAThreadWaitForHandles+0xa9
combase!CoWaitForMultipleHandles+0x80
combase!ASTAState::WaitForPendingGitRegistrations+0x3e
combase!CComApartment::WaitForPendingGitRegistrations+0x49
```

```
combase!ApartmentUninitialize+0x10c640
combase!wCoUninitialize+0x13a
combase!CoUninitialize+0xeb
twinapi_appcore!Windows::Foundation::Uninitialize+0x7
twinapi_appcore!<lambda_...>::operator()+0xe9
SHCore!_WrapperThreadProc+0x10f
kernel32!BaseThreadInitThunk+0x10
ntdll!RtlUserThreadStart+0x2b
```

The thread is in the process of uninitializing: Notice that at the bottom of the stack are `ApartmentUninitializing` and `CoUninitialize`. The problem isn't that we're resuming on the wrong thread. The problem is that XAML has already shut down on this thread, so you can't create any new XAML objects.

And the exception that XAML throws when you try to create a XAML object on a thread that is not initialized for XAML is `RPC_E_WRONG_THREAD` because "You're on the wrong thread. Go find a XAML thread."

The customer came to the conclusion that while the `OnLoaded` function was loading the content asynchronously, the user dismissed the flyout by clicking on something else, and that triggered the tear-down of the thread. When `OnLoaded` resumed execution, it tried to create a `MyViewModel` object, but failed because XAML had already left the thread.

In this case, the proper course of action is just to abandon the initialization of the `MyFlyout` object, since the user has already dismissed it.

```
// C++/CX with PPL
void MyFlyout::OnLoaded()
{
    create_task(ApplicationData::Current->LocalFolder->
                    GetFileAsync(L"cache.json"))
    .then([](StorageFile^ file)
    {
        return FileIO::ReadTextAsync(file);
    }).then([](String^ contents)
    {
        auto json = JsonObject::Parse(contents);
        m_viewModel = ref new MyViewModel(json);
    }).then([this](task<void> t)
    {
        try {
            t.get();
        } catch (COMException^) {
            // Something went wrong. Start from scratch.
            try {
                m_viewModel = ref new MyViewModel();
            } catch (Platform::WrongThreadException^) {
                // We were already dismissed - abandon.
            }
        }
    });
}

// C++/WinRT equivalent
winrt::fire_and_forget MyFlyout::OnLoaded()
{
    auto lifetime = get_strong();

    try {
        auto file = co_await ApplicationData::Current().LocalFolder().
                        GetFileAsync(L"cache.json");
        auto contents = co_await FileIO::ReadTextAsync(file);
        auto json = JsonObject::Parse(contents);
        m_viewModel = MyViewModel(json);
    } catch (...) {
        // Something went wrong. Start from scratch.
        try {
            m_viewModel = MyViewModel();
        } catch (winrt::hresult_wrong_thread const&) {
            // We were already dismissed - abandon.
        }
    }
}

// or with function try
winrt::fire_and_forget MyFlyout::OnLoaded() try
{
    auto lifetime = get_strong();
    try {
        auto file = co_await ApplicationData::Current().LocalFolder().
                        GetFileAsync(L"cache.json");
        auto contents = co_await FileIO::ReadTextAsync(file);
        auto json = JsonObject::Parse(contents);
        m_viewModel = MyViewModel(json);
    } catch (...) {
        // Something went wrong. Start from scratch.
        m_viewModel = MyViewModel();
    }
} catch (winrt::hresult_wrong_thread const&) {
    // We were already dismissed - abandon.
}


// C# equivalent
async void OnLoaded()
{
    try {
        var file = await ApplicationData.Current.LocalFolder.
                        GetFileAsync(L"cache.json");
        var contents = await FileIO.ReadTextAsync(file);
        var json = JsonObject.Parse(contents);
        m_viewModel = new MyViewModel(json);
    } catch (Exception) {
        // Something went wrong. Start from scratch.
        try {
            m_viewModel = new MyViewModel();
        } catch (Exception e)
            when ((uint)e.HResult == 0x8001010E) {
            // We were already dismissed - abandon.
        }
    }
}
```

Raymond Chen

Follow