How can I get the Windows Runtime HttpClient to display a basic authentication prompt?

devblogs.microsoft.com/oldnewthing/20240212-00

February 12, 2024

Raymond Chen

A customer was trying to get the Windows Runtime `Windows.Web.Http.HttpClient` to display a basic authentication prompt when used from a classic Win32 application. Unfortunately, no authentication dialog appeared even though they set `AllowUI = true`; the request reported status 401 (Unauthorized).

```
namespace winrt
{
    using namespace winrt::Windows::Foundation;
    using namespace winrt::Windows::Web::Http;
    using namespace winrt::Windows::Web::Http::Filters;
}

winrt::IAsyncOperation<winrt::HttpStatusCode> Sample(HWND hwnd)
{
    auto uri = winrt::Uri(L"http://httpbin.org/basic-auth/user/password");
    auto filter = winrt::HttpBaseProtocolFilter();
    filter.AllowUI(true);

    auto client = winrt::HttpClient(filter);
    auto result = co_await client.GetAsync(uri);

    co_return result.StatusCode();
}
```

Originally, the objects in the Windows Runtime namespace were intended for use in UWP apps (Universal Windows Platform). In UWP apps, each thread can have at most one CoreWindow object, and it was common for Windows Runtime objects to infer their UI context by asking the thread for its CoreWindow, and then using that CoreWindow for any further user interface operations.[1]

But classic Win32 apps don't follow the "one CoreWindow per thread" model. They can create multiple windows on each thread, and none of them are required to be a CoreWindow. Under these conditions, Windows Runtime objects that were designed for UWP apps run into a problem: What window should they use for user interface operations?

When used in a classic Win32 app, the `HttpBaseProtocolFilter` tries to guess which window to use: Sometimes it guesses correctly, sometimes it guesses wrong, and sometimes its guess comes up empty.

To avoid guessing, use the `IInitializeWithWindow` interface (which we learned about earlier) to give the `HttpBaseProtocolFilter` an explicit window to use.

```
namespace winrt
{
    using namespace winrt::Windows::Foundation;
    using namespace winrt::Windows::Web::Http;
    using namespace winrt::Windows::Web::Http::Filters;
}

winrt::IAsyncOperation<winrt::HttpStatusCode> Sample(HWND hwnd)
{
    auto uri = winrt::Uri(L"http://httpbin.org/basic-auth/user/password");
    auto filter = winrt::HttpBaseProtocolFilter();
    filter.as<IInitializeWithWindow>()->Initialize(hwnd);
    filter.AllowUI(true);

    auto client = winrt::HttpClient(filter);
    auto result = co_await client.GetAsync(uri);

    co_return result.StatusCode();
}
```

With this extra nudge, the `HttpBaseProtocolFilter` will be able to display the basic authentication dialog.

[1] This model stopped working even for UWP apps with the introduction of AppWindow objects, which let you create multiple windows on a single thread.