How can I check if I'm on a DispatcherQueue's thread if I can't call HasThreadAccess?

devblogs.microsoft.com/oldnewthing/20231018-00

October 18, 2023

Raymond Chen

A customer had code that used the `DispatcherQueue.HasThreadAccess` property, but found that the code crashed when running on Windows Server 2019 systems,[1] because the `HasThreadAccess` property wasn't added until Windows Server Version 1903.

They wondered if there was a way to find out whether you are running on a `DispatcherQueue`'s thread without using the `HasThreadAccess` property?

Well, let's browse around the members of `DispatcherQueue`:

Member	Available in Server 2019?
GetForCurrentThread	Yes
CreateTimer	Yes
TryEnqueue	Yes
ShutdownStaring	Yes
ShutdownCompleted	Yes
HasThreadAccess	No

It occurred to me that you can see if a particular `DispatcherQueue` belongs to the current thread by simply reversing the question: Ask the current thread for its `DispatcherQueue` and see if it's the one you were given. This relies on the fact that each thread can have at most one `DispatcherQueue`.

```
// Alternate version that simulates HasThreadAccess
// on Windows Server 2019.

// C++/WinRT
bool DispatcherQueueHasThreadAccess(DispatcherQueue const& q)
{
    ASSERT(q != nullptr); // caller should have checked this first
    return q == DispatcherQueue::GetForCurrentThread();
}
```

The customer reported back that it worked great.

A lot of computer programming is just looking at the tools you have available in your toolbox and seeing whether you can combine them in an interesting way to accomplish your goal.

[1] Still in extended support until 2029!