Creating a semaphore with a maximum count from WaitOnAddress

devblogs.microsoft.com/oldnewthing/20170613-00

June 13, 2017

Raymond Chen

Last time, we created a simple semaphore from `WaitOnAddress`. That sempahore did not have a maximum token count. Let's add that.

```
struct ALT_MAXSEMAPHORE
{
  LONG TokenCount;
  LONG MaxTokenCount;
};

void InitializeMaxAltSemaphore(ALT_MAXSEMAPHORE* Semaphore,
                               LONG InitialCount,
                               LONG MaxTokenCount)
{
  Semaphore->TokenCount = InitialCount;
  Semaphore->MaxTokenCount = MaxTokenCount;
}

bool ReleaseMaxAltSemaphore(ALT_MAXSEMAPHORE* Semaphore,
                            LONG ReleaseCount)
{
  while (true) {
    LONG OriginalTokenCount = Semaphore->TokenCount;
    LONG NewTokenCount = OriginalTokenCount + ReleaseCount;
    if (NewTokenCount > Semaphore->MaxTokenCount) {
     return false; // would exceed maximum
    }
    if (InterlockedCompareExchange(&Sempahore->TokenCount,
      NewTokenCount,
      OriginalTokenCount) == OriginalTokenCount) {
      if (ReleaseCount == 1) {
        WakeByAddressSingle(&Sempahore->TokenCount);
      } else {
        WakeByAddressAll(&Sempahore->TokenCount);
      }
      return true;
    }
  }
}
```

Releasing the tokens is a little trickier because we have to verify that we aren't going to exceed the maximum token count. Our attempt to release may fail if other threads snuck in and either claimed or released tokens, in which case we loop back and try again.

The code to wait for and claim a token is unchanged. We merely had to tweak how we release tokens.

Okay, next time, we'll specialize to the case of an event, which is like a semaphore with a maximum count of 1.

Raymond Chen

Follow