Why does the documentation for ReadFile say that the lpNumberOfBytesRead parameter is optional when it is sometimes mandatory?

devblogs.microsoft.com/oldnewthing/20161026-00

October 26, 2016

Raymond Chen

The documentation for the `ReadFile` function says

```
BOOL WINAPI ReadFile(
  _In_ HANDLE hFile,
  _Out_ LPVOID lpBuffer,
  _In_ DWORD nNumberOfBytesToRead,
  _Out_opt_ LPDWORD lpNumberOfBytesRead,
  _Inout_opt_ LPOVERLAPPED lpOverlapped
);
```

The `lpNumberOfBytesRead` parameter is declared as `_Out_opt_`. The `_Out_` part means that the function writes to the pointed-to value. The `_opt_` part means that the parameter is optional (may be null). And yet, if you call `ReadFile` and pass `nullptr` for the fourth parameter, it crashes. What's going on here?

What's going on is in the fine print:

> ***lpNumberOfBytesRead*** **[out, optional]**
> A pointer to the variable that receives the number of bytes read when using a synchronous *hFile* parameter. **ReadFile** sets this value to zero before doing any work or error checking. Use **NULL** for this parameter if this is an asynchronous operation to avoid potentially erroneous results.
>
> This parameter can be **NULL** only when the *lpOverlapped* parameter is not **NULL**.
>
> For more information, see the Remarks section.

Note that second paragraph.

The deal is that the fourth parameter is sometimes optional and sometimes mandatory. The simplified annotation language used by this function prototype cannot express this sort of conditional mandatory state, so it takes the most generous position of declaring the

parameter as optional, so as to avoid raising false positives in static analysis tools.

A more precise annotation for that parameter would be something like `_When_(lpOverlapped == NULL, _Out_opt_) _When_(lpOverlapped != NULL, _Out_)`.

Raymond Chen

Follow