How can I call freopen but open the file with shared access instead of exclusive access?

devblogs.microsoft.com/oldnewthing/20180221-00

February 21, 2018

Raymond Chen

A customer wants to redirect their program's `stdout` to a file. They followed the sample code which basically boils down to the line

```
FILE* stream = freopen("output.txt", "w", stdout);
```

or its security-enhanced alternate version:

```
errno_t err = freopen_s(&stream, "output.txt", "w", stdout);
```

The customer reported that this worked exactly as exepcted, but the output file is opened for exclusive access. They want another process to be able to read from the output file while the original process is writing to it, but the exclusive access prevents that.

The Microsoft-specific function `_fsopen` lets you specify a custom sharing mode, but there is no corresponding `_fsreopen` function that augments the `freopen` function with a sharing mode.

Is there anything the customer can do?

The C/C++ runtime library folks suggested using `_dup2` to remap the file descriptor. Something like this:

```
#include <fcntl.h>
#include <io.h>
#include <stdio.h>
#include <windows.h>

// All error checking omitted for clarity
int main()
{
  // Open with desired sharing mode
  HANDLE h = CreateFileW(L"output.txt", GENERIC_WRITE, FILE_SHARE_READ,
                         nullptr, CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL,
                         nullptr);

  // Convert to a file descriptor
  int fd = _open_osfhandle(reinterpret_cast<intptr_t>(h), _O_WRONLY);


  // Remap stdout's file descriptor to be a copy of the one we just created
  _dup2(fd, _fileno(stdout));

  // Don't need our file descriptor any more
  _close(fd);


  printf("Hello, world!\n");

  return 0;
}
```

The customer confirmed that this does exactly what they needed.

Raymond Chen

Follow