# ProcMonDebugOutput C++ Code Review

By Giovanni Dicanio <[giovanni.dicanio@gmail.com](mailto:giovanni.dicanio@gmail.com)>

<https://github.com/Wintellect/ProcMonDebugOutput>

1. Updated from “old” SAL to **new SAL**. E.g. \_\_success 🡪 \_Success\_, \_\_in 🡪 \_In\_. Moreover, since the lpOutputString parameter is a NUL-terminated string, used the more precise \_In\_z\_ SAL annotation.
2. Parameter lpOutputString renamed as pszOutputString, since **psz** is the specific Hungarian for pointer to NUL-terminated strings.
3. Used modern C++11 **nullptr** instead of C/C++98/03 NULL.
4. Removed the *C-ism* of using f(void) if a function has no parameter: the modern C++ way is just f(). Applied to OpenProcessMonitorLogger() and CloseProcessMonitorLogger().
5. Since both OpenProcessMonitorLogger() and CloseProcessMonitorLogger() are private helper functions, they can be wrapped inside the **anonymous namespace** in C++.
6. \_tcslen() is for code that follows the *TCHAR* model, and can be compiled in both ANSI/MBCS and Unicode. However, modern Win32 C++ code should be Unicode-only (and, in fact, some modern Vista+ APIs just use PCWSTR – i.e. const WCHAR \* – instead of LPCTSTR, and also the pszOutputString parameter **o**f the ProcMonDebugOutput function is Unicode, i.e. LPCWSTR). So, \_tcslen() was replaced with the Unicode version **wcslen()**.  
   (The <tchar.h> header was also removed from the "stdsfx.h" precompiled header.)
7. The C-style cast to DWORD applied to the return value of \_tcslen() was replaced with the C++-style cast static\_cast<DWORD>(). Removed also the (VOID\*) cast applied to the string parameter passed to DeviceIoControl() (the cast was rewritten in terms of modern C++-style casts).
8. Used raw string literals to simplify string literals in C++ source code (e.g. LR"(\\.\Global\ProcmonDebugLogger)" instead of duplicated backslashes L"\\\\.\\Global\\ProcmonDebugLogger" ).
9. Updated some code in NativeTest, e.g. using the *wchar\_t*-forms like swprintf\_s() instead of the *TCHAR*-forms like \_stprintf\_s(); moreover, since this is C++ code and not C code, the destination buffer size can be deduced by the compiler using template techniques, so the explicit destination buffer size was removed from source code, making the code simpler.