DLL Parameter Types in OpenInsight

Introduction

One feature that puts the “Open” in OpenInsight is the ability to use functions that are stored within DLL files. Quite often these are Windows API functions but they can also be third-party utilities. The OpenInsight 7.0 Programmers Reference Manual has a lot of information on how to make this work (see Chapter 7: Calling DLL Functions from Basic+). However, the list of parameter types has not been updated with those that were introduced in OpenInsight 7.0. Here is a complete list:

Data Types

VOID	void, only valid as a return type
CHAR	ansi or utf8 character value
ACHAR	ansi character value
UCHAR	utf8 character value
WCHAR	unicode character value
BYTE	1 byte signed integer
UBYTE	1 byte unsigned integer
SHORT	2 byte signed integer
USHORT	2 byte unsigned integer
LONG	4 byte signed integer
ULONG	4 byte unsigned integer
INT	platform signed integer
UINT	platform unsigned integer
FLOAT	4 byte floating point number
DOUBLE	8 byte floating point number
HANDLE	generic windows handle value
LPVOID	generic pointer value
LPCHAR	pointer to ansi or utf8 characters
LPACHAR	pointer to ansi characters
LPUCHAR	pointer to utf8 characters
LPWCHAR	pointer to unicode characters
LPBYTE	pointer to 1 byte signed integer
LPUBYTE	pointer to 1 byte unsigned integer
LPSHORT	pointer to 2 byte signed integer
LPUSHORT	pointer to 2 byte unsigned integer
LPLONG	pointer to 4 byte signed integer
LPULONG	pointer to 4 byte unsigned integer
LPINT	pointer to platform signed integer
LPUINT	pointer to platform unsigned integer
LPFLOAT	pointer to 4 byte floating point number
LPDOUBLE	pointer to 8 byte floating point number
LPHANDLE	pointer to generic windows handle value
LPSTR	pointer to null terminated ansi or utf8 string

LPASTR	pointer to null terminated ansi string
LPUSTR	pointer to null terminated utf8 string
LPWSTR	pointer to null terminated unicode string
LPBINARY	pointer to binary data

Notes

1. New Data Types are indicated in BLUE.
2. The new Data Types distinguish between Ansi, Unicode, and UTF8 to allow for more accurate transfer of data.
3. LPCHAR used to be the only means for specifying strings. In reality, LPCHAR means “Pointer to an array of characters.” However, most DLL parameters want a null terminated string of characters, which cannot be guaranteed by LPCHAR. We recommend using LPCHAR, LPACHAR, LPUCHAR, and LPWCHAR only for passing pointers to single characters or for backward compatibility.
4. LPSTR, LPASTR, LPUSTR, and LPWSTR should now be used when passing strings. These types append a null character when passing BASIC+ strings to the DLL. When used as return values, the LPSTR types truncate returned strings at the first null character.
5. Always use the specific pointer type rather than the generic. LPCHAR and LPSTR are generic because they do not specify Ansi, Unicode or UTF8 translation. Using these types will cause strings to be passed “as is” without proper conversion. Use the Ansi, Unicode, and UTF8 counterparts to these types to ensure proper encoding of data.
6. LPBINARY is meant for passing an array of bytes. Use this when passing Structs to and from a DLL function.
7. As before, to specify the passing of an explicit pointer, declare the parameter as LPVOID rather than LPx. Otherwise, the LPx types will pass the pointer to the pointer.