MATLAB® News & Notes

Programming Patterns

Calling Generic DLL Functions From MATLAB

by Peter Webb

Access to external data and functions has always been important to MATLAB users. The earliest versions of MATLAB could read external data. The ability to call external functions followed soon after.

External Functions and MATLAB

Since MEX-files were introduced, in the late 1980s, MATLAB has supported the ability to call functions written in other languages, such as C or Fortran. With MATLAB 5.2 it became possible to call Java functions directly, and now users of MATLAB 6.5 can call functions within Windows DLLs without writing a MEX interface function. This means that more functions are accessible to MATLAB users, making it that much easier to develop robust solutions.

Calling Generic DLL Functions

On machines running Microsoft®Windows, MATLAB MEX-files are 32-bit WindowsDLLs that export a C function named `mexFunction`. This function provides MATLAB with a standard way to exchange data with the functions in that DLL. By contrast, generic 32-bit Windows DLLs do not export `mexFunction`. Until now, if you wanted to call functions in a generic DLL, you had to write a MEX-file that called the generic C functions. The DLL Interface Library removes that requirement, enabling you to call generic DLL functions directly from MATLAB.

The ability to call generic functions is available only in MATLAB 6.5 on Windows XP/2000/NT/98 for DLLs that export functions written in C. The DLL Interface Library contains seven functions:

- `loadlibrary`:Load a DLL into memory so that MATLAB can call it.
- `unloadlibrary`:Unload a DLL and free the associated memory.
- `calllib`:Call a function in a loaded DLL.
- `libfunctions`:List the functions available in a DLL.
- `libpointer`:Create a reference to MATLAB data passed to a DLL function.

- `libstruct`:Construct an externally defined structure in MATLAB.
- `libisloaded`:Test to see if a given DLL has been loaded into MATLAB.

For complete details on the library interface, see the file `shared_library_doc.pdf` in the downloadable DLL Interface Library package (described in the next section).

To call generic Windows DLL functions from MATLAB, you must have both the DLL containing the function and a C header file that defines both the function prototypes and any nonprimitive types used by the DLL's exported functions. Programs that use the DLL Interface Library call external functions using a four-step pattern:

1. Load the DLL Interface Library with `loadlibrary`.
2. Prepare data for calls to functions in the library with `libpointer` and `libstruct`.
3. Call library functions via `calllib`.
4. Unload the DLL library by calling `unloadlibrary`.

Many externally compiled functions do not use the same data types as MATLAB, but most types of MATLAB data have corresponding C data types, and the library performs the appropriate data conversion automatically.

Three kinds of C data types require special treatment: enumerated types, structures, and references (also called pointers). Enumerated types are the simplest:When a C function requires an enumerated type, you may pass it either the name of the enumeration member (as a string) or the corresponding numeric value. You may pass MATLAB structures directly to C functions via `calllib`. MATLAB will attempt to automatically convert the MATLAB structure to a suitable C structure. However, automatic conversion allocates memory for a new structure on each call, and might be too expensive when the structures are large or the application makes large numbers of function calls. `libstruct` and `libpointer` help you manually manage data conversion for structures and references.

`libstruct` lets you create a structure suitable for passing to C functions. In the simplest case, you supply `libstruct` the string name of a C structure defined in the DLL that you're calling and a MATLAB structure. If the MATLAB structure matches the C structure definition, `libstruct` returns a MATLAB corresponding to the appropriate C structure.

`libpointer` lets you create a C pointer type that references MATLAB data. For any given C data type, say `MyCStruct`, you construct the corresponding pointer type name by adding the suffix `Ptr`, e.g., `MyCStructPtr`. This works for MATLAB primitive types as well—a pointer to an `int16` is an `int16Ptr`, for example.

Getting the Software

To call DLL functions from MATLAB 6.5, you need to download and install the

DLL Interface Library. The library is available from our Web site: *ftp://ftp.mathworks.com/pub/tech-support/solutions/ s33513/GenericDll_1p1.exe*

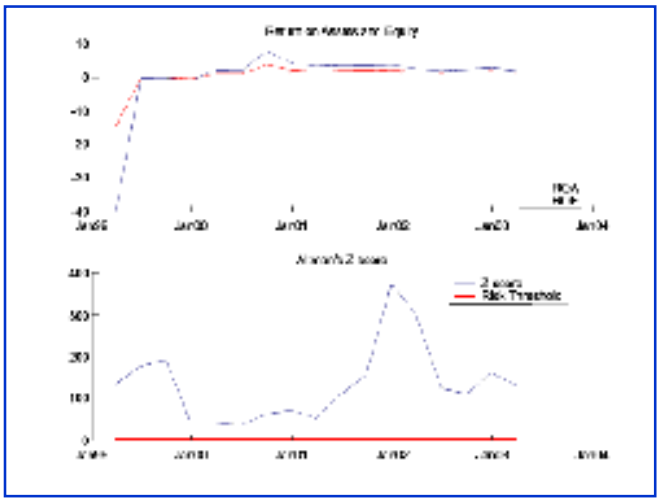

Figure 1: Quarterly Perfomance of PeopleSoft, 1999-2003
Click on image to see enlarged view.

To install the library, quit any MATLAB sessions that you have running, and then double-click on the downloaded executable. For more detailed instructions, see *www.mathworks.com/nn_ppsol*

Once you have installed the library, you can download the example code from MATLAB Central at *www.mathworks.com/nn_dll* Please read the installation instructions in the readme.txt file enclosed in that ZIP file.

Example: Corporate Financial Performance

This sample code consists of a single MATLAB M-file, `finperf.m`, which uses two external C functions to help analyze the financial performance of an organization: `dupont( )` and `zscore( )`. `dupont( )` calculates two measures of profitability: return on assets (ROA) and return on equity (ROE). `zscore()` computes the *Altman Z-Score*, a widely used measure of the risk of bankruptcy. The sample code follows the four-step pattern described above. All the code samples below are from `finperf.m`.

First, `finperf` loads the library. DLL libraries behave a little like MEX-files, in that MATLAB loads only a single global instance. You must verify that a library is not already loaded before loading it. The two arguments to `loadlibrary` are the path to the DLL and the path to a header file describing the functions in the DLL (in that order). If you omit the extensions, as this example does, loadlibrary assumes that the DLL ends with `.dll` and the header file ends with `.h`.

```
libname = 'BizLib';
if ~libisloaded(libname)
loadlibrary(libname, libname);
end
```

Next, the example uses `xlsread` to load the financial data from a Microsoft

Excel spreadsheet (see the example's `loaddata` function). `finperf` loops over the data, calling the external functions. The loop is necessary because the external functions do not take array parameters. `dupont( )` has 4 inputs and 2 outputs.

The second output is passed to dupont( ) as a pointer to a double:

```
reteq = -1;
[roa(i) roe(i)] = calllib(libname, 'dupont', ...
reteq, profit(i), ...
assets(i), revenue(i), ...
equity(i));
```

`calllib` converts the data automatically, implicitly calling `libpointer` to create the reference value from the variable `reteq` (implicitly creating a pointer variable `ptr2reteq`) and then implicitly calling `get(ptr2reteq,'Value')` to extract the returned value.

The `zscore( )` function takes a pointer to a structure and returns a double. Again, `calllib` converts the data automatically, implicitly calling `libstruct` and `libpointer` to create a pointer to a C structure from the MATLAB structure.

```
zdata.WorkingCapital = wc(i);
zdata.RetainedEarnings = re(i);
zdata.EBIT = ebit(i);
zdata.BookValueOfDebt = debt(i);
zdata.MarketValueOfStock = mktval(i);
zdata.TotalAssets = assets(i);
zdata.SalesRevenue = revenue(i);
zscore(i) = calllib(libname, 'zscore', zdata);
```

Finally, `finperf` plots the data in a two-axis figure, as shown in Figure 1.

Building Robust Solutions Quickly

If you've been writing MEX-files in MATLAB 6.5 to call DLL functions in Windows, download the DLL Interface Library today. You'll see an immediate increase in your productivity. By enabling MATLAB to call external functions defined in standard C DLLs, the DLL Interface Library enables you to easily incorporate legacy code or third-party function libraries into your applications.

For more information visit:

- Example Code in MATLAB Central www.mathworks.com/nn_dll
- MathWorks DLL Interface Library www.mathworks.com/nn_ppsol
- The Du Pont ROI Model http://home.att.net/~nickols/dupont.htm
- Definition of Altman Z-Score www.investopedia.com/terms/a/altman.asp
- Definition of Return on Assets www.investopedia.com/terms/r/returnonassets.asp

- Definition of Return on Equity
  www.investopedia.com/terms/r/returnonequity.asp

Next Article

related topics:

Using MathWorks Products For... | Training | MATLAB Based Books | Third-Party Products