Win32 Programming

Jim Fawcett
CSE775 – Distributed Objects
Spring 2012
References

- Unicode, More Unicode
- Win32 Code Tips
- www.Catch22.net
  - Tips #1, Tips #2, Tutorials
- Windows SDK Issues
Topics

- Important header files
- Errors
- Win32 Types
- Unicode – coming later
- Win32 API
Important Windows Header Files

Important header files in PlatformSDK folder
- Windows.h Includes other headers
  - Commdlg.h Common controls (COM based)
  - Dlgs.h Dialog definitions
  - OLE2.h COM definitions
  - ShellAPI.h Shell API
  - ShFolder.h Folder definitions
  - Winbase.h Managing processes
  - Windef.h Windows type definitions
  - WinUser Windows messages
  - Many other headers
- WinError.h Error codes and strings
- Winsock2.h sockets
- MSHTML.h Browser Control definitions and IE DOM handling

Important header files in VC folder
- Process.h Threading
## Win32 API Error Protocols

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOID</td>
<td>Function cannot fail</td>
</tr>
<tr>
<td>BOOL</td>
<td>If function fails value is zero; otherwise nonzero.</td>
</tr>
<tr>
<td>HANDLE</td>
<td>If the function fails usually value is NULL; otherwise valid handle to kernel object. Some functions return INVALID_HANDLE on failure.</td>
</tr>
<tr>
<td>PVOID</td>
<td>If the function fails, value is NULL; otherwise PVOID points to a memory address to be manipulated.</td>
</tr>
<tr>
<td>LONG/DWORD</td>
<td>Usually indicates a count is returned. If counting fails, function usually returns -1 or 0. Check the platform SDK documentation.</td>
</tr>
</tbody>
</table>
# Data Types

<table>
<thead>
<tr>
<th>Windows Type</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPVOID</td>
<td>void*</td>
</tr>
<tr>
<td>LPSTR</td>
<td>wchar_t*</td>
</tr>
<tr>
<td>TCHAR</td>
<td>char or wchar_t</td>
</tr>
<tr>
<td>LPTSTR</td>
<td>char* or wchar_t*</td>
</tr>
<tr>
<td>DWORD</td>
<td>unsigned 32 bit integer</td>
</tr>
<tr>
<td>WORD</td>
<td>unsigned 16 bit integer</td>
</tr>
<tr>
<td>HANDLE</td>
<td>Windows handle</td>
</tr>
<tr>
<td>HINSTANCE</td>
<td>Handle to program instance</td>
</tr>
<tr>
<td>HMODULE</td>
<td>Handle to loaded DLL</td>
</tr>
<tr>
<td>HMENU</td>
<td>Menu handle</td>
</tr>
<tr>
<td>WPARAM</td>
<td>16 bit parameter</td>
</tr>
<tr>
<td>LPARAM</td>
<td>32 bit parameter</td>
</tr>
</tbody>
</table>
Win32 API

Path to documentation

- **Windows API Reference:**
  MSDN Library > Development Tools and Languages > Windows API

- **Windows Data Types:**
  MSDN Library > Windows Development > Getting Started > Using Windows Headers > Windows Data Types

- **System Services** (DLLs and Processes, File Services, ...):
  MSDN Library > Windows Development > System Services

- **User Interface** (Windows Controls, Windows Shell, ...)
  MSDN Library > Windows Development > Windows Application UI Development

C-Style functions often with extended versions
Win32 API – Extended Functions

Example – extended function

- **ReadFile**
  ```c
  BOOL ReadFile(
    HANDLE hFile, LPVOID lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPDWORD lpNumberOfBytesRead,
    LPOVERLAPPED lpOverlapped
  );
  ```

- **ReadFileEx**
  ```c
  BOOL ReadFileEx(
    HANDLE hFile, LPVOID lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPOVERLAPPED lpOverlapped,
    LPOVERLAPPED_COMPLETION_ROUTINE lpCompletionRoutine
  );
  ```
Win32 API by Function

- **File services**
  - Disk, Volume, Directory, and File management

- **Windows System Information**
  - Handles, Objects, Registry, System Info, Time

- **DLLs, Processes, and Threads**
  - DLLs, Processes and Threads, Services, Synchronization

- **Memory Management**
  - Reserving and committing memory, File Mapping

- **Interprocess Communication**
  - Mailslots, Network DDE (deprecated), Pipes
Files

**File Management**
- CreateFile, OpenFile, CloseHandle, ReplaceFile
- ReadFile(Ex), WriteFile(Ex), CopyFile(Ex), DeleteFile, MoveFile(Ex)
- FindFirstFile(Ex), FindNextFile, FindClose, SearchPath
- GetFileAttributes(Ex), SetFileAttributes, GetFileSize(Ex)
- GetFullPathName
- LockFile(Ex), UnlockFile(Ex)
Directories

**Directory Management Functions**

- `CreateDirectory(Ex), RemoveDirectory`
- `GetCurrentDirectory, SetCurrentDirectory`
- `FindFirstChangeNotification, FindNextChangeNotification, FindCloseChangeNotification`
Registry

Registry Functions

- RegCreateKeyEx, RegOpenKeyEx, RegCloseKey, RegDeleteKey(Ex), RegDeleteKeyValue, RegReplaceKey, RegRestoreKey, RegSaveKey(Ex), RegQueryValueEx
- RegGetValue, RegSetValueEx, RegDeleteValue, RegSetKeyValue
- RegLoadKey, RegUnloadKey
- RegEnumKeyEx, RegEnumValue
- RegCopyTree, RegDeleteTree
Dynamic-Link Libraries

Dynamic-Link Library Functions

- DllMain
- LoadLibrary(Ex), FreeLibrary, GetProcAddress
- GetDllDirectory, SetDllDirectory
- GetModuleFileName(Ex), GetModuleHandle(Ex)
Processes

- **Process Functions**
  - EnumProcesses, Process32First, Process32Next
  - GetCurrentProcess, GetCurrentProcessId
  - GetProcessId, GetProcessHandleCount
  - GetCommandLine, GetEnvironmentStrings, GetEnvironmentVariable, SetEnvironmentVariable, GetGuiResources
**Threads**

---

**Thread Functions**

- CreateThread, OpenThread, ThreadProc
  - don’t use if you call C/C++ libraries
  - Use `_beginthread(Ex)`
- CreateRemoteThread, ExitThread, ResumeThread, TerminateThread, GetExitCodeThread
- `Sleep(Ex)`, `WaitForInputIdle`
- `GetCurrentThread`, `GetCurrentThreadId`
- `GetThreadId`
- `GetThreadPriority`, `SetThreadPriority`
- `TlsAlloc`, `TlsFree`, `TlsGetValue`, `TlsSetValue`
C Run-Time Library Thread Functions

- **Thread functions**
  - `_beginthread(Ex), _endthread(ex)`
Virtual to Physical Mapping

Process 1 Virtual Memory:
- System addressable memory
- Application addressable memory

Process 2 Virtual Memory:
- System addressable memory
- Application addressable memory

Physical Memory:
- Free page 1
- System, page 3
- Process 1, page 5
- Process 2, page 1
- Process 1, page 2
- System, page 2
- Process 2, page 4
- Process 2, page 3
- Free page 2
- System, page 4
- Process 1, page 1
- Process 2, page 2
- Free page 4
- Process 1, page 4
- System, page 1
- Free page 3
- Process 1, page 3
Memory Management

- **Memory Management Functions**
  - **Used by COM — Allocates on heap**
    GlobalAlloc, GlobalReAlloc, GlobalLock, GlobalUnlock, GlobalFree, GlobalHandle
  - **Newer Heap Management** — not movable
    HeapAlloc, HeapCompact, HeapCreate, HeapDestroy, GetProcessHeap(s)
    - CopyMemory, FillMemory, MoveMemory
  - **Memory Mapped Files**
    CreateFileMapping, OpenFileMapping, MapViewOfFile(Ex), FlushViewOfFile, UnmapViewOfFile
  - **Virtual Memory**
    VirtualAlloc(Ex), VirtualFree(Ex)