How do I enumerate drives the same way that the NET USE command does?

devblogs.microsoft.com/oldnewthing/20141117-00

November 17, 2014

Raymond Chen

If you use the Remote Desktop Connection client to connect to another computer, you have the option of making your local drives available to the remote computer.

A customer wanted to know how to enumerate all the drives on the local machine. The were able to get the volumes mapped to drive letters, but they also wanted to get the redirected drives injected by Terminal Services. (Mind you, these aren't volumes that are assigned drive letters, so it's not clear why they are interested in them, but whatever.)

With the `NET USE` command, they see the Terminal Services volumes in Explorer, and they can be browsed via `\\tsclient\d` :

```
Status       Local     Remote                    Network
-------------------------------------------------------------------------------
                       \\TSCLIENT\D              Microsoft Terminal Services
The command completed successfully.
```

The customer wanted to enumerate these Terminal Services client volumes. "How does the `NET USE` command enumerate these volumes?"

Let's write that program. Remember, Little Programs do little to no error checking.

```
#define UNICODE
#define _UNICODE
#include <windows.h>
#include <winnetwk.h>
#include <stdio.h>
void report(PCWSTR pszLabel, PCWSTR pszValue)
{
 printf("%ls = %ls\n", pszLabel, pszValue ? pszValue : L"(null)");
}
int __cdecl main(int, char **)
{
 HANDLE hEnum;
 WNetOpenEnum(RESOURCE_CONNECTED,
              RESOURCETYPE_DISK,
              0,
              NULL,
              &hEnum);
 DWORD cbBuffer = 65536;
 void *buffer = LocalAlloc(LMEM_FIXED, cbBuffer);
 LPNETRESOURCE pnr = (LPNETRESOURCE)buffer;
 DWORD err;
 do {
  DWORD cEntries = INFINITE;
  DWORD cb = cbBuffer;
  err = WNetEnumResource(hEnum, &cEntries, buffer, &cb);
  if (err == NO_ERROR || err == ERROR_MORE_DATA) {
   for (DWORD i = 0; i < cEntries; i++) {
    report(L"localName", pnr[i].lpLocalName);
    report(L"remoteName", pnr[i].lpRemoteName);
    report(L"provider", pnr[i].lpProvider);
    printf("\n");
   }
  }
 } while (err == ERROR_MORE_DATA);
 LocalFree(buffer);
 WNetCloseEnum(hEnum);
 return 0;
}
```

We open an enumeration for connected disks and then start enumerating out of it. The usage pattern for `WNetEnumResources` is kind of messy, with a bunch of in/out parameters that need to get reset each time. Each time, we say “Enumerate as much as you can into this 64KB buffer” and then print what we got. If we were told, “There’s still more,” then we go back and ask for more.

That’s all. Nothing particularly fancy.

Raymond Chen

Follow