Printing the contents of the clipboard as text to stdout

devblogs.microsoft.com/oldnewthing/20131007-00

October 7, 2013

Raymond Chen

The `clip.exe` takes its stdin and puts it on the clipboard. But how do you get it out? That's today's Little Program. (I guess we could call it `clop.exe`.)

```
#define UNICODE
#define _UNICODE
#define STRICT
#include <windows.h>
#include <stdio.h>
#include <tchar.h>
#include <strsafe.h>
void WriteToStdOut(const void *pvBuf, DWORD cbBuf)
{
 DWORD cbWritten;
 WriteFile(GetStdHandle(STD_OUTPUT_HANDLE), pvBuf, cbBuf,
           &cbWritten, nullptr);
}
int __cdecl _tmain(int argc, PTSTR *argv)
{
 if (OpenClipboard(nullptr)) {
  HANDLE h = GetClipboardData(CF_UNICODETEXT);
  if (h) {
   auto pwchText = static_cast<PCWSTR>(GlobalLock(h));
   if (pwchText) {
    SIZE_T cbMemory = GlobalSize(h);
    // arbitrary limit because I am lazy
    cbMemory = min(cbMemory, 0x10000000);
    size_t cbActual;
    if (SUCCEEDED(StringCbLengthW(pwchText, cbMemory,
                                  &cbActual))) {
     if (argc == 2 && _tcsicmp(argv[1], TEXT("/u")) == 0) {
      WriteToStdOut(pwchText, cbActual);
     } else {
      UINT cp = (argc == 2 &&
                 _tcsicmp(argv[1], TEXT("/a")) == 0) ?
                      CP_ACP : CP_OEMCP;
      int cch = WideCharToMultiByte(cp, 0, pwchText,
                cbActual / 2, nullptr, 0, nullptr, nullptr);
      if (cch > 0) {
       auto psz = new(std::nothrow) char[cch];
       if (psz) {
        WideCharToMultiByte(cp, 0, pwchText, cbActual / 2,
                            psz, cch, nullptr, nullptr);
        WriteToStdOut(psz, cch);
        delete[] psz;
       }
      }
     }
    }
    GlobalUnlock(h);
   }
  }
  CloseClipboard();
 }
 return 0;
}
```

Okay, what do we have here?

We open the clipboard and try to get the Unicode text on it. We then look for the null terminator within the first 0x10000000 bytes. Why do I stop at 256MB? Because I'm lazy and this lets me avoid worrying about integer overflow. This is a Little Program, remember.

If you pass the `/U` command line switch, then the output is printed to stdout as the Unicode string itself.

If you pass the `/A` command line switch, then the output is converted to ANSI.

Otherwise the output is converted to the OEM code page.

Bonus chatter: You can get most of the same program above (no Unicode output) in much less code if you're willing to use C#:

```
class Program {
  [System.STAThread]
  public static void Main(string[] args)
  {
    string text = System.Windows.Forms.Clipboard.GetText();
    if (args.Length == 1 && string.Compare(args[0], "/a", true) == 0) {
        System.Console.OutputEncoding = System.Text.Encoding.Default;
        System.Console.Write("changed encoding");
    }
   System.Console.Write(text);
  }
}
```

Or perl (ANSI output only):

```
use Win32::Clipboard;
print Win32::Clipboard()->GetText();
```

Raymond Chen

Follow