Subject: Re: More Unicode questions
Posted by cbpporter on Mon, 25 May 2009 10:49:55 GMT
View Forum Message <> Reply to Message

I started my hunting down of invalid uses of ANSI API on Unicode Windows.

I started with the part that gets the environment variables, namely AppInitEnvironment__. This uses ANSI versions, so even through my environment variables are kept by windows in Unicode, any non Latin characters would be lost in U++ applications. Following the normal procedure, I would test with IsWinNT and duplicate the code, but this time using wchar and W version of the API. Simple enough in theory.

Unfortunately I have bitten of more than I can chew. It seems that even GetEnvironmentStringsW isn't capable of returning Unicode strings. Even worse, result vary a lot depending on system locale, but even a string entered with characters from that locale won't be returned correctly, with '?' characters replacing non Latin ones, but not all of them. I have tried all other Win API functions to retrieve the variables, and the result is the same.

There is only one place where the values are correct: the Windows registry. I've written a test version that in my tests retrieves correctly environment variables with all possible of characters used.

If anybody has time and patience for a short experiment, then please try to create a environment variable with some national characters, or any Unicode character and see if the following code retrieves the correct values:

```
WString GetWinRegStringW(const wchar *value, const wchar *path, HKEY base_key) {
 HKEY key = 0;
 if(RegOpenKeyExW(base_key, path, 0, KEY_READ, &key) != ERROR_SUCCESS)
  return WString::GetVoid();
 dword type, data;
 if(RegQueryValueExW(key, value, 0, &type, NULL, &data) != ERROR_SUCCESS)
 {
  RegCloseKey(key);
  return WString::GetVoid();
 }
 WStringBuffer raw_data(data);
 if(RegQueryValueExW(key, value, 0, 0, (byte *)~raw_data, &data) != ERROR_SUCCESS)
 {
  RegCloseKey(key);
  return WString::GetVoid();
 }
 if(data > 0 && (type == REG_SZ || type == REG_EXPAND_SZ))
  data -= 2;
 raw_data.SetLength(data / 2);
 RegCloseKey(key);
 return raw_data;
}
```

```
void Win32GetEnvVars()
{
#ifndef PLATFORM_WINCE
 //FileOut f("c:\\test.log");
 if (IsWin2K()) {
  wchar *env = GetEnvironmentStringsW();
  for(wchar *ptr = env; *ptr; ptr++)
  {
   const wchar *b = ptr;
   if(*ptr)
    ptr++;
   while(*ptr && *ptr != '=')
    ptr++;
   WString varname(b, ptr);

   if(*ptr)
    ptr++;
   b = ptr;
   while(*ptr)
    ptr++;
   WString value = WString(b, ptr);

   WString newval = GetWinRegStringW(varname,
L"SYSTEM\\CurrentControlSet\\Control\\Session Manager\\Environment",
HKEY_LOCAL_MACHINE);
   if (!newval.IsVoid())
    value = newval;

   newval = GetWinRegStringW(varname, L"Environment", HKEY_CURRENT_USER);
   if (!newval.IsVoid())
    value = newval;

   /*WString temp = (varname + "=" + value);
   WString nl = L"\r\n";
   f.PutW(temp, temp.GetCount());
   f.PutW(nl, 2);*/

   coreEnvPtr__().GetAdd(ToUpper(varname).ToString()) = value.ToString();
  }
  FreeEnvironmentStringsW(env);
 }
 else {
  char *env = GetEnvironmentStringsA();
  for(char *ptr = env; *ptr; ptr++)
  {
   const char *b = ptr;
   if(*ptr)
```

```
    ptr++;
   while(*ptr && *ptr != '=')
    ptr++;
   String varname(b, ptr);

   if(*ptr)
    ptr++;
   b = ptr;
   while(*ptr)
    ptr++;
   coreEnvPtr__().GetAdd(ToUpper(varname)) = FromSystemCharset(String(b, ptr));
  }
  FreeEnvironmentStringsA(env);
 }
 //f.Close();
#endif
}

void AppInitEnvironment__() {
 Win32GetEnvVars();
 CommonInit();
}
```

The code replaces AppInitEnvironment__ in Core/App.cpp.

Thank you!

---