

Re: get wide character and multibyte character value

Re: get wide character and multibyte character value

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2008-01/msg00674.html>

- *From:* George <George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sun, 27 Jan 2008 00:26:00 -0800
-

Thanks Norbert,

And UTF16 says if a unicode codepoint does not fit into 16 bit, then split it up into more than one 16 bit number.

It is what UTF-16 says, not what Windows says...

How do you think the characters which requires more than 16-bit in UTF-16, could be represented by wide character on Windows `wchar_t` (which is defined to unsigned short, 16-bit). So, it is why I think Windows has limitations to represent all UTF-16 characters with `wchar_t` (unsigned short).

Any comments?

regards,
George

"Norbert Unterberg" wrote:

George schrieb:

Thanks Joe,

Czech characters). Unicode is designed to represent all characters that exist, where code pages are only designed to represent enough characters for you to work with on your computer. So, converting to your computer's code page, you are actually going from a

Re: get wide character and multibyte character value

more
versatile character set to a less versatile character set.

I like your above description so much! Better than that in wikipedia. I suggest you add them. :-)

Well, since you have a lot of experience, a question makes me confused is, in my understanding, Windows in always using 16-bit (WORD) to represent UTF-16, and never uses 32-bit. So, on Windows wide character is always 16-bit. Is that correct?

If yes, when 16-bit is not enough to represent all the characters, right? 32-bit should be enough. So Windows wide character has limitations?

No, George.
You should trad this nice article:
<http://www.joelonsoftware.com/articles/Unicode.html>

"Unicode" is a charcter set. It is a standard with the goal that every possible character on the earth (and even beyond) gets a specific number, which is called a codepoint.

UTF-8, UTF-16, UTF-32 are encodings. They are rules how to fit these numbers into a 8, 16 or 32 bit arrays. And UTF16 says if a unicode codepoint does not fit into 16 bit, then split it up into more than one 16 bit number.

But more about that can be found in the link I gave you above.

Norbert

regards,
George

"Zapanaz" wrote:

On Wed, 23 Jan 2008 23:13:01 -0800, George
<George@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

Hello everyone,

I need to know the wide character (unicode)
and multibyte (UTF-8) values of

Re: get wide character and multibyte character value

a character string of czech. I personally know nothing about czech. Is the following approach correct?

1. I use L on the character string and watch memory to get the wide character representation of the character string in little endian form;

2. I change the computer region/language to czech, and use function WideCharToMultiByte, and use CP_ACP as input code page and use the L character string as input to get the output multibyte character string output from parameter lpMultiByteStr. Is (1) and (2) correct? Any more efficient and smart ways?

thanks in advance,
George

The others already answered this better than I could, but I think I might be able to clarify a couple things (I have been working on converting an application to unicode for weeks, so this is all very fresh in my head)

With the precompiler option UNICODE defined and you have a string which is wchar_t (a string with L"" at the start will be wchar_t), then what you see viewing memory in Visual Studio is that string in 16-bit unicode.

If you use WideCharToMultiByte with CP_ACP as the CodePage argument, what you are doing is converting the 16-bit unicode to the current Windows code page on your computer.

Code Pages are pretty much obsolete. Code pages were a way of representing different characters not in the 0-127 ASCII range (like Czech characters). Unicode is designed to represent all characters

Re: get wide character and multibyte character value

that exist, where code pages are only designed to represent enough characters for you to work with on your computer. So, converting to your computer's code page, you are actually going from a more versatile character set to a less versatile character set.

To get the UTF-8 version of the string, use UTF-8 instead of CP_ACP.

Which version you want depends on what you're doing. Within Windows, generally the wide char version is what you want. UTF-8 is more generally used for unicode in general, if you are interacting with third-party software and they refer to unicode, they will frequently mean UTF-8. If the software is halfway-decently documented though, they should tell you explicitly which encoding they mean.

So there are three character encodings in what you're talking about, UTF-16 (not completely identical with wchar_t but basically the same in Windows) which is the Windows internal Unicode, UTF-8, and your local code page.

(I know I am not explaining everything perfectly ... like the difference between wchar_t and UTF-16 ... but I think as far as your question goes you should get the idea.)

It can be a little confusing, because when people say "unicode", it can mean a number of different things, and it isn't always clear which they mean.

--

Joe Cosby

<http://joecosby.com/>

My cat seems to produce enough extraneous hair to create another whole cat about twice a year. They actually seem to be trying to form under various tables and bookshelves, but only in uncarpeted areas. They

Re: get wide character and multibyte character value

never get much beyond a sort of misty transparent stage and
then they
get blown out into the middle of the room.
– nenslo

:: Currently listening to John Shirley Reads 1996 1997 1998,
by John Shirley, from "3-Fisted Tales of "Bob" Audio
Book"