Subject: How to convert utf-8 to (strcmp type)?
Posted by JoseB on Fri, 28 Jan 2011 01:24:57 GMT
View Forum Message <> Reply to Message

Hello
How can i convert utf-8 String variable in order to pass it to the strcmp() C++ function?

Thank you

JoseB

Subject: Re: How to convert utf-8 to (strcmp type)?
Posted by koldo on Fri, 28 Jan 2011 07:35:16 GMT
View Forum Message <> Reply to Message

JoseB wrote on Fri, 28 January 2011 02:24Hello
How can i convert utf-8 String variable in order to pass it to the strcmp() C++ function?

Thank you

JoseB
Hello JoseB

Could you indicate more details?.

Subject: Re: How to convert utf-8 to (strcmp type)?
Posted by JoseB on Fri, 28 Jan 2011 09:16:50 GMT
View Forum Message <> Reply to Message

Basically I have a upp GUI app and a command line program which is called from that upp application. I want to pass it an argument with is a string or better a char * array. Internally this char * array will be used by strcmp.

JoseB

Subject: Re: How to convert utf-8 to (strcmp type)?
Posted by cbpporter on Fri, 28 Jan 2011 09:22:35 GMT
View Forum Message <> Reply to Message

Do you want to convert it to a legacy encoding? strcmp will work for equality test with utf-8, but not with comparison.

Subject: Re: How to convert utf-8 to (strcmp type)?
Posted by koldo on Fri, 28 Jan 2011 22:41:59 GMT
View Forum Message <> Reply to Message

Hello JoseB

I do not understand well the question but, if you require to pass a String to an API function you would have to do a ToSystemCharset().

For example, in Functions4U/FUnctios4U.cpp ReadOnly() function:

- In Windows, GetFileAttributesW() and SetFileAttributesW() require a ToSystemCharsetW().

- In POSIX, stat() and chmod() require a ToSystemCharset().

Subject: Re: How to convert utf-8 to (strcmp type)?
Posted by mirek on Sun, 30 Jan 2011 11:49:20 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Fri, 28 January 2011 04:22Do you want to convert it to a legacy encoding? strcmp will work for equality test with utf-8, but not with comparison.

Actually, it will work for comparison as well, only the results might not be what you have expected...

Anyway, the same is true for 'legacy encoding'. The correct comparison depends on language rules; you ought to use LanguageInfo::Compare to get correct results, but it is now implemented only for Czech language and even that is not a full implementation of rules...

(That said, there is some default implementation in int DefaultLanguageCompare that would probably work reasonably well for most Latin based languages).

Mirek