Subject: Unicode support is now pretty seamless, right?
Posted by xrysf03 on Tue, 18 Dec 2018 21:54:40 GMT
View Forum Message <> Reply to Message

Hello everybody,
another day, another noob observation...

I live in Windows 7 at the moment, and using the Mingw tool chain.

While trying to set the window title text in an early "hello world" proggie, I quickly found the Title() method of the TopWindow class... and I've noticed that it supposedly takes a WString input. I went on to look for the header file to include, to be able to instantiate WString, I was searching old forum posts for functions to convert between char* and Unicode strings, I considered using #define UNICODE at the top of my source files etc. I also tried making a std::vector<std::string> and then use c_str() to get my hands on the internal char* ... And I kept getting stuck in various ways.

Then I realized that I can just create a Vector<String> (within namespace Upp) and I can use the String instances contained in the Vector directly as input to Title() - e.g. if I index into the Vector using operator[], even though the Assist does not mention such a version of the Title() prototype, i.e. taking Upp::String as input :) Does some automatic conversion from String to WString take place? An automatic intermediate WString object construction or some such?

Also, I can simply assign text string literals written in "eastern latin" to String variables, and I don't need to care about encoding... Does TheIDE use UTF-8 as a default encoding of the source code files? It would seem so, if I'm reading the hints from VIM and Notepad correctly...

It all feels pretty seamless - I feel like I can forget about encodings altogether and just "do my thing", in an ignorant autopilot mode... Is it really that simple? I mean - using Unicode in U++ :)
I would expect to meet some trouble as soon as I deviate from the U++ universe (stuff living in the Upp:: namespace) and start dealing with STL std:: stuff or syscalls and external DLL functions using plain old ASCII char*.

Any futher hints/ideas are welcome...

Frank

Subject: Re: Unicode support is now pretty seamless, right?
Posted by mirek on Wed, 19 Dec 2018 08:00:04 GMT
View Forum Message <> Reply to Message

xrysf03 wrote on Tue, 18 December 2018 22:54Hello everybody,
another day, another noob observation...

I live in Windows 7 at the moment, and using the Mingw tool chain.

While trying to set the window title text in an early "hello world" proggie, I quickly found the Title() method of the TopWindow class... and I've noticed that it supposedly takes a WString input. I went on to look for the header file to include, to be able to instantiate WString, I was searching old forum posts for functions to convert between char* and Unicode strings, I considered using #define UNICODE at the top of my source files etc. I also tried making a std::vector<std::string> and then use c_str() to get my hands on the internal char* ... And I kept getting stuck in various ways.

Then I realized that I can just create a Vector<String> (within namespace Upp) and I can use the String instances contained in the Vector directly as input to Title() - e.g. if I index into the Vector using operator[], even though the Assist does not mention such a version of the Title() prototype, i.e. taking Upp::String as input :) Does some automatic conversion from String to WString take place? An automatic intermediate WString object construction or some such?

Also, I can simply assign text string literals written in "eastern latin" to String variables, and I don't need to care about encoding... Does TheIDE use UTF-8 as a default encoding of the source code files? It would seem so, if I'm reading the hints from VIM and Notepad correctly...

It all feels pretty seamless - I feel like I can forget about encodings altogether and just "do my thing", in an ignorant autopilot mode... Is it really that simple? I mean - using Unicode in U++ :)
I would expect to meet some trouble as soon as I deviate from the U++ universe (stuff living in the Upp:: namespace) and start dealing with STL std:: stuff or syscalls and external DLL functions using plain old ASCII char*.

Any futher hints/ideas are welcome...

Frank

Yes, UTF-8 is the normal encoding of U++ apps (this can be changed to support some legacy code to any supported 8-bit encoding, but that is not a good idea). WString(const char *) constructor is using this default encodiding.

That said, long term, WString should probably be deprecated as full UNICODE is much more complicated. In reality it seems like any variant of String with more than 8-bits solves exactly nothing if full UNICODE is to be supported. Anyway, that is some distant future, for now WString solves most practical problems.

Mirek

Subject: Re: Unicode support is now pretty seamless, right?
Posted by xrysf03 on Wed, 19 Dec 2018 10:21:59 GMT
View Forum Message <> Reply to Message

Allright, thanks for the respose...
For a start, I'll try to get by with my "ignorant autopilot engaged" (just use the Upp::String) and I will ask futher questions if I face some stumbling blocks down the road (external Unicode API's to

work with).

Frank