Subject: Re: Choosing the best way to go full UNICODE
Posted by cbpporter on Wed, 31 May 2017 09:30:33 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 31 May 2017 12:00
If you can iterate linearly, Unicode is indexable...

Sorry, you don't get it. I'm using a very specific meaning of indexable, probably that is the problem.

A vector is indexable. You can reach v[7] without going though 0 to 6 and 0 though 6 can't do anything to change the "offset" of 7. A list is not. You need to traverse it to get list[7].

You need to traverse a String s to get to s[7]. The contents up to 7 can change the offset of 7. Unicode strings are not indexable.

Indexable data structures are predictable contant-cost no need to iterate over them to get some "groundind". Non-indexable does not mean that you can't reach them though an index. It means that you can get only obtain that index with a linear traversal O(index) traversal and you need to reach than index at least once.

mirek wrote on Wed, 31 May 2017 12:00
I am now really thinking that "multibyte" String is the solution. The one that returns a variable sequence of bytes for each position. I am now even thinking this does not need to be bound to graphemes only.

The longterm point with that is to replace WString as processing facility in editors.

That sound to me like you trying to make String indexable. Which can't be done with Unicode. How are you going to determine the sequence of bytes that you must return for a position without traversing it linearly?

If you give it a position as input, this still needs to be a deterministic constant cost memory jump. That should be the bread and butter operation. In the few cases where you need your multibyte behavior, there you abandon indexing randomly into the string (string[7] is forbidden) and iterate over it, going left to right, returning multiple bytes for each logical "position".

Let me sum up the Unicode fallacy:
1. Code points are only indexable in Utf32. Which is not always helpful because
2. Glyps/Graphemes are never indexable.

The common misconception is that 1 and/or 2 are false.

mirek wrote on Wed, 31 May 2017 12:00
I am now leaning against it. Vector<int> is good enough for utf32 - what we eventually need to do

with it.

I agree, that is enough. And in some context even C vectors will work.

We do need a standardized functions that convert from code units to a single code point to fill up these structures.

mirek wrote on Wed, 31 May 2017 12:00
Not so sure about this - not that important IMO at this point. So I will not get correct ToUpper for many characters - that has little impact on most applications.

Half measures again...
