Subject: Re: String improvements
Posted by mirek on Sun, 26 Feb 2006 11:36:00 GMT
View Forum Message <> Reply to Message

hojtsy wrote on Sun, 26 February 2006 06:10OK, if negative offsets will not be supported, would it be possible to include these other minor modifications:
- ReverseFind should ASSERT for invalid values of "from"
- the loop in ReverseFind should be fixed to avoid reading into the memory before the first char. See the fix in my post above.
- the type of chr parameter should be T and not int.

Definitely!

Quote:
Also I raised another problem with ReverseFind: ReverseFind(c, x) starts to search at position x-1. This is unintuitive and different from both std::string and QString, in which the reverese find method starts from the position given as parameter. Would it be possible to change this? (backward incompatible change)

Well, I think we should be able to afford that change. I will have to debate it this with Tom a little though...

Quote:
As for the reasons to support negative offsets:
- brevity: If I would like to search in the last x chars of a String calculated with a complex exression, it would be shorter to write:
longExpression->something().Format(somemore).Find(something, -3)
instead of creating separate local variable to store the String, and doing s.Find(something, s.GetLength()-3).

The tradeoff there is how often you need something like that vs. bug catching. All I can say at the moment is that I never needed to search through last n characters (by checking my codebase).

Generally, I am quite reluctant adding features that just "could be useful sometimes". Means, I need real-world example before considering it further.

Quote:
Maybe an other parameter could solve the problem of brevity:
Find(int chr, int from = 0, OffsetDirection dir = FromStart)

So instead of negative offset, you can use the third param to specify if the offset is calculated relative to the start or the end.

Well, considering usage scenarios you have gave me so far (I mean, the above example), I would

prefer separate method like
"FindInLast" or something like that (less typing, simple inliner,, bug catching ability retained).

In any case, this has to wait after 602... (after AttrText disaster, I would like to keep the code-base as stable as possible during next week...).

Quote:
Here is another suggestion for a new method in Stringtemplate <class T, class S>
bool AString<T, S>::ContainsAt(const S &s, int pos) const
{
 ASSERT(pos >= 0);
 if(pos < 0 || pos + s.GetCount() > GetCount()) return false;
 return memcmp(ptr + pos, s.Begin(), s.GetCount()) == 0;
}
[/quote]

Well, there is a couple of methods (or external functions) I am missing in String and something like this would certainly be useful.

Mirek