Subject: Re: String improvements
Posted by hojtsy on Sun, 26 Feb 2006 11:10:40 GMT
View Forum Message <> Reply to Message

OK, 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.

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)

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).
- conformance to expectations: QString has it. People coming from Qt will expect this behaviour.

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.

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;
}
```