Subject: Re: [BUG] CommandLine() returns non-unicode arguments
Posted by mirek on Fri, 11 Apr 2008 12:24:42 GMT
View Forum Message <> Reply to Message

exhu wrote on Fri, 11 April 2008 04:05Tested with 2007.1 win, MSVC 2005, WinXP SP2.

UPDATE: use windows console with Lucida Console font (that which is unicode-aware).

run as:

result:
"shu?bandig"

(the string is to contain umlauts or other symbols not in current ANSI encoding, try russian letters etc. if your current code page in WinXP contains umlauts).

Program source:

```
#include <iostream>

CONSOLE_APP_MAIN
{
 const Vector<String> & cmdline = CommandLine();

 for(Vector<String>::ConstIterator i = cmdline.Begin();i != cmdline.End();++i) {
  std::cout << (*i).Begin() << std::endl;
 }
}
```

NOTE: cout actually is not a proper class to use here, but if the U++ uses wide char win32 api then the produced string would be utf-8 or so and not the word with question mark and simply "a" instead of "a umlaut"...

U++ uses wide char win32 api when available and mostly for filenames only... Internally, it can work in several encodings, utf8 being the recommended one.

Anyway, this is interesting problem, something not considered before... Commandline actually is used "as is", in ANSI version, without any conversions. The same is true for Cout.

Still, something is wrong with console output - input (commandline) seems to be OK, but output is wrong. But maybe you know more about these issues...

INteresting question is also what is the correct solution. Should Cout perform on-the-fly encoding coversion?