Subject: Re: String w/high characters but not UTF?
Posted by koldo on Tue, 10 Aug 2010 14:01:38 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Tue, 10 August 2010 15:23There is nothing wrong with that program, related to UTF8 or otherwise. I behaves as it should. The problem is that you are inserting a large value like 182 in a signed char and the result gets interpreted as a negative number.

Yes.

For example compiling with MSC I got three warnings like this:

warning C4309: 'initializing' : truncation of constant value

for the 181, 182 and 183.

In addition String d does not know the length of char data[] as it is not ended with '\0'. This easily can produce an error.

Check this:

```
#include <Core/Core.h>

using namespace Upp;

CONSOLE_APP_MAIN
{
 {
  puts("Original");
  char data[] = { 65, 65, 65, 181, 65, 65, 65, 182, 65, 65, 183 };
  String d(data);
  for (int i=0; i < d.GetCount(); i++)
   puts(FormatInt(i) + "=" + FormatInt(d[i]));
 }
 {
  puts("Changed");
  byte data[] = { 65, 65, 65, 181, 65, 65, 65, 182, 65, 65, 183 };
  String d(data, 11);
  for (int i=0; i < d.GetCount(); i++)
   puts(FormatInt(i) + "=" + FormatInt(byte(d[i])));
 }
 getchar();
}
```

The output is this:

```
Original
0=65
```

1=65
2=65
3=-75
4=65
5=65
6=65
7=-74
8=65
9=65
10=-73
Changed
0=65
1=65
2=65
3=181
4=65
5=65
6=65
7=182
8=65
9=65
10=183

byte type is a natural way in U++ to handle binary data.

If you need a classic C array with undefined length in compiling time you can also use:

Buffer<byte> data;

data.Alloc(dataLen);

instead of the usual and more dangerous malloc/free/new/delete.
