

---

Subject: Re: Building & using U++ without TheIDE  
Posted by [mirek](#) on Wed, 19 Sep 2007 07:34:05 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

sergei wrote on Tue, 18 September 2007 18:43Update:

I'm done implementing unicode. I don't exactly like the way I did it, but it works. I prefer some global strings manager like the TSTR I suggested, but since Mirek said he doesn't want TCHARs I've updated all API calls manually. I added PLATFORM\_UNICODE define, and replaced many `#ifdef PLATFORM_WINCE` with it. Conversion mostly implemented through `ToSystemCharset` and `FromSystemCharset`.

In the process I've also found and eliminated a "security vulnerability" (that's what memory bugs are called nowadays ) in Log.h:

```
printf(h, "%s %02d.%02d.%04d %02d:%02d:%02d, user: %s\n",  
      (const char*)FromSystemCharset(exe),  
      t.day, t.month, t.year, t.hour, t.minute, t.second, (const char*)FromSystemCharset(user));
```

I added the `(const char*)`, otherwise I got segmentation fault in debug. Probably without the explicit cast `printf` thought that `String` is a char array.

I received another segmentation fault earlier (but maybe it was this one...), and that one was solved by replacing all `To/FromSysChrSet` with `To/FromSystemCharset`. Not a big deal IMHO, they were all used in Win32-specific code anyway...

Uh oh, that is no good. There definitely should be `FromSysCharset`, which returns a pointer to static char \* array.

The reason is that Log is supposed to work independently from the rest of U++, so that it can be used in situation when the rest failed or is not available. By using vanilla `FromSystemCharset`, you make it dependent on `String` and in turn on memory allocator. Therefor it will not work when heap crashes or cannot be used to debug memory allocator...

Quote:

I've tested unicode filenames - it did open a multilingually-named file and read its content successfully. I've also tested registry - successfully wrote that filename to a `REG_SZ` key, it's fine. Didn't try to create unicode-named keys (don't wanna kill my windows).

Sort of redundant work. The final solution will use dynamic .dll loading to choose between W and A variants.

Quote:

P.S.2 since you know about the MSVC bug, did anyone report to MS? There might be a chance

that they fix it...

No.

Quote:

Edit: I think UTFBOM class I posted above, or something else implementing that functionality, should be added to some place in U++. That way unicode support will be complete - unicode filenames + unicode text.

Well, the question is the definition. Thinking about it, I just fail to see what is the exact description. If detection of utf-8 files is the purpose, that can be easily done without specific code. Also, what is going to happen if there is no UTFBOM?

Should it be available as

```
bool SkipUTFBOM(Stream& in);  
const char *SkipUTFBOM(const char *s);
```

?

Mirek

---