Subject: It's suspected to be an issue with Font.
Posted by Lance on Fri, 18 Mar 2011 22:50:28 GMT
View Forum Message <> Reply to Message

The programs I used to compare are UWord from the UPP, and MS Word. The platform is Windows 7.

The text I used to test is:

The problem with U++ drawed text is that some characters are notably larger than others and some have incorrect horizontal displacement.

Please see attached picture for a visual effect.

I also encountered issue where chinese characters are displayed correctly displayed on Windows but are blank on Ubuntu. And when I copies the same text that was displayed as blank to, say gedit, the text displayed correctly as in Windows. That part I will attach picture in future.

File Attachments
1) font problem.png, downloaded 859 times

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sun, 10 Apr 2011 12:42:52 GMT
View Forum Message <> Reply to Message

Lance wrote on Fri, 18 March 2011 18:50The programs I used to compare are UWord from the UPP, and MS Word. The platform is Windows 7.

The text I used to test is:

The problem with U++ drawed text is that some characters are notably larger than others and some have incorrect horizontal displacement.

It works fine on my Windows 7. However, I believe that the problem is caused by font substitution mechanism and perhaps on your system, you have some font that takes precendence for some glyphs, but does not contain other characters.

However, I have noticed that OpenOffice or Wordpad are preferring SimSun font, so I have given

it higher priority now. Please check.

The core table for font subsitutions is in Draw/FontCR, table sFontReplacements. If U++ does not find required glyph in the font, it tries to get the glyph by going through fonts in this table and uses first glyph available... Maybe you could play with it a bit to get better results.

Quote:
I also encountered issue where chinese characters are displayed correctly displayed on Windows but are blank on Ubuntu. And when I copies the same text that was displayed as blank to, say gedit, the text displayed correctly as in Windows. That part I will attach picture in future.

I believe this is basically the same issue - gedit has better knowledge about fonts, so is able to find a better replacement.

Maybe, if openoffice behaves similary in Ubuntu as in Windows, you can try to paste characters into Openoffice and then check what font it has actually used for it. Then we can add this font to replacement table...

Alternatively please post some problematic text here so that I can do this myself.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Tue, 12 Apr 2011 12:03:33 GMT
View Forum Message <> Reply to Message

Great Mirek.

Thank you very much for your effort. I didn't know you were working on it. Thanks a million.

I will download the most recent version and give it a try.

Lance

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Tue, 12 Apr 2011 13:19:17 GMT
View Forum Message <> Reply to Message

Sorry but the problem doesn't seemed to be fixed.

I tested only on Windows 7 with version 3336, didn't notice any improvement. Even font substitution is the old way (SimHei I believe instead of SimSun).

Has be fix been committed yet?

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sat, 16 Apr 2011 18:54:48 GMT
View Forum Message <> Reply to Message

Lance wrote on Tue, 12 April 2011 09:19Sorry but the problem doesn't seemed to be fixed.

I tested only on Windows 7 with version 3336, didn't notice any improvement. Even font substitution is the old way (SimHei I believe instead of SimSun).

Has be fix been committed yet?

Well, it was.

Maybe you could try yourself, I believe it is easy to do so.

Maybe create just simple app that only draws the text in window, then put a DUMP to Replace(Font fnt, int chr, Font& rfnt) in CRFont.cpp to find out what is going on...

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sat, 16 Apr 2011 19:02:48 GMT
View Forum Message <> Reply to Message

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

using namespace Upp;

struct MyApp : TopWindow {
 virtual void Paint(Draw& w) {
  w.DrawRect(GetSize(), White);

 }
};

GUI_APP_MAIN
{
 MyApp().Run();
}
```

And this DDUMP

bool Replace(Font fnt, int chr, Font& rfnt)

```
{
 static Vector<int> rface;
 static Vector<dword> l, h;
 ONCELOCK {
  for(int i = 0; i < __countof(sFontReplacements) && rface.GetCount() < 20; i++) {
   int q = Font::FindFaceNameIndex(sFontReplacements[i].name);
   if(q > 0) {
    rface.Add(q);
    l.Add(sFontReplacements[i].l);
    h.Add(sFontReplacements[i].h);
   }
  }
 }

 Font f = fnt;
 dword tl = chr < 4096 ? 0x80000000 >> (chr >> 7) : 0;
 dword th = 0x8000000 >> ((dword)chr >> 11);
// DDUMP(FormatIntHex(chr));
// DDUMP(FormatIntHex(th));
 for(int i = 0; i < rface.GetCount(); i++) {
//  DDUMP(Font(rface[i], 10));
//  DDUMP(FormatIntHex(h[i]));
//  DDUMP(FormatIntHex(h[i] & th));
  if(((l[i] & tl) || (h[i] & th)) && IsNormal(f.Face(rface[i]), chr)) {
   int a = fnt.GetAscent();
   int d = fnt.GetDescent();
   if(f.GetAscent() > a || f.GetDescent() > d) {
    static sFontMetricsReplacement cache[256];
    int q = CombineHash(fnt, f) & 255;
    if(cache[q].src != fnt || cache[q].dst != f) {
     cache[q].src = fnt;
     cache[q].dst = f;
     while((f.GetAscent() > a || f.GetDescent() > d) && f.GetHeight() > 1) {
      f.Height(max(1, min(f.GetHeight() - 1, f.GetHeight() * 9 / 10)));
     }
     cache[q].mdst = f;
    }
    else
     f = cache[q].mdst;
   }
   rfnt = f;
   DDUMP(rfnt.GetFaceName()); // <<- HERE
   return true;
  }
 }
 return false;
}
```

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sat, 16 Apr 2011 19:21:27 GMT
View Forum Message <> Reply to Message

BTW, I have tried with debian, default installation, and it does not display the text even in the browser

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Wed, 20 Apr 2011 20:07:21 GMT
View Forum Message <> Reply to Message

Just saw it. Thank for the effort! Font is too complicated a topic for me but I'll definitely give it a try.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Wed, 20 Apr 2011 20:30:16 GMT
View Forum Message <> Reply to Message

Here is the output. The test was done on a Windows machine.

rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sat, 30 Apr 2011 08:37:17 GMT
View Forum Message <> Reply to Message

Lance wrote on Wed, 20 April 2011 16:30Here is the output. The test was done on a Windows machine.

rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS

Well, for me, it is SimSun all the way...

Could you please try to paste the text into WordPad and tell me what font it reports?

Mirek

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sun, 01 May 2011 15:23:00 GMT
View Forum Message <> Reply to Message

Hi Mirek:

Thank you for your continued attention to this issue.

I don't have immediate access to a Windows U++ development environment. I did the same test on Ubuntu and here is a screen shot:

See how different characters are missing in TheIDE editors and in the running test program? That is the problem I mentioned earlier regarding missing characters on Linux. BTW: it's Ubuntu 11.04.

And the LOG file:

rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum

I will report the result as soon as I can put my hands on a Windows version of TheIDE.

File Attachments
1) Screenshot2.png, downloaded 1253 times

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sun, 01 May 2011 15:35:00 GMT
View Forum Message <> Reply to Message

As a comparison, in Ubuntu gedit, no matter what font I use, the Chinese characters either displayed as default font if these Characters are not in the Font's character set, or displayed with the font.

File Attachments
1) Screenshot3.png, downloaded 1277 times

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sun, 01 May 2011 17:41:04 GMT
View Forum Message <> Reply to Message

On Windows XP, here is the LOG file content:

rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Mon, 02 May 2011 05:07:19 GMT
View Forum Message <> Reply to Message

Well, but I am still not getting information about what font should be used on your systems.

I need to you to paste the text into editor that would display replacement font. I know for sure that

OpenOffice is capable of doing that. WordPad in Win7 as well.

Simply paste text there, move cursor over CJK characters, it would show what font is used.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Mon, 02 May 2011 13:58:44 GMT
View Forum Message <> Reply to Message

It seems OpenOffice hide the font substitution from end user. Here is the result of my experiment in Ubuntu + Open Office. See the hilighted character is related with nominal font "FreeSerif"

Let me know if I can somehow find the actual font used.

Regardless what font (serif, sans, etc), gedit/open office will render Chinese characters at uniform size and space, so the Upp way of interpret non-western font may have flaw. Interesting enough, on Windows, the characters almost always get displayed albeit with size/placement issues, while on Ubuntu, some characters cannot be displayed and become blanks.

File Attachments
1) Screenshot4.jpg, downloaded 1323 times

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Mon, 02 May 2011 14:11:46 GMT
View Forum Message <> Reply to Message

By the way, SimSun should be a safe bet. Any Chinese characters should be representable in SimSun. Say there is a fancy font FancyFont. It only implemented 2000 most common chinese characters, than a character that's not in this set should be displayed in SimSun

---

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Mon, 02 May 2011 18:07:11 GMT
View Forum Message <> Reply to Message

Lance wrote on Mon, 02 May 2011 10:11By the way, SimSun should be a safe bet. Any Chinese characters should be representable in SimSun. Say there is a fancy font FancyFont. It only implemented 2000 most common chinese characters, than a character that's not in this set should be displayed in SimSun

Well, but problem is that SimSun is already tested and required...

It is all really weird. In Windows, it works for me just fine. In your machine, it seems to choose "Arial Unicode MS" instead.

Hm, one possible explanation: Do not you have the same Arial Unicode MS installed on ubuntu?

It actually seems there is something wrong with that font, the replacement algorithm seems to have detected required glyphs in it...

OK, one more try: try to move SimSun in the list BEFORE Arial Unicode MS.

```
struct sRFace {
 const char *name;
 dword l, h;
} sFontReplacements[] = {
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x09c00080 },
 { "SimSun", 0xfd800000, 0x09ffff00 },
 { "Arial Unicode MS", 0xfffc3fef, 0xfa7ff7e7 },
 { "MS UI Gothic", 0xffc01008, 0x0fffff00 },
 { "MS Mincho", 0xffc01008, 0x0fffff00 },
.....
```

Let us see, maybe it could help in windows.

Another thing to consider is to list all fonts to find out whether there is something bad with names:

```
GUI_APP_MAIN
{
 for(int i = 0; i < Font::GetFaceCount(); i++)
  LOG(Font::GetFaceName(i));
}
```

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Tue, 03 May 2011 12:48:51 GMT
View Forum Message <> Reply to Message

DDUMP font result.

```
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
rfnt.GetFaceName() = UnDotum
```

List Fonts Result:

STDFONT
serif
sans-serif
monospace

UnDotum
LMMonoLt10
Century Schoolbook L
OpenSymbol
Khmer OS System
LMSansQuot8
LMMathSymbols10
LMRomanSlant9
LMRomanSlant8
LMSans9
LMSans8
Mukti Narrow
Meera
Vemana2000
LMMonoSlant10
Umpush
Purisa
Pothana2000
DejaVu Sans Mono
Norasi
Loma
AR PL UKai TW MBE
URW Palladio L
Phetsarath OT
Sawasdee
Tlwg Typist
Lucida Bright
URW Gothic L
Dingbats
URW Chancery L
Ubuntu
FreeSerif
ori1Uni

Kedage
DejaVu Sans
Kinnari
LMSans17

LMSans12
LMSans10
Lohit Punjabi
LMRoman17
LMRoman12
LMRoman10
TlwgMono
Symbol

LMRomanDunh10
LMRoman7
LMRoman6
LMRoman5
LMRoman9
LMRoman8
Bitstream Charter
KacstOne
AR PL UKai TW
Khmer OS
AR PL UKai HK
Liberation Mono
Courier 10 Pitch
AR PL UKai CN

Nimbus Sans L
TlwgTypewriter
TakaoPGothic
AR PL UMing TW MBE
LMRomanDemi10
Rachana

AR PL UMing HK
LMMonoCaps10
AR PL UMing CN
LMMonoLtCond10
AR PL UMing TW
Standard Symbols L
Lohit Gujarati
Nimbus Mono L
Nimbus Mono L
Lucida Sans
Liberation Serif
Mallige
LMMathItalic10
Nimbus Roman No9 L
LMMathItalic12
LMRomanUnsl10
Ubuntu

Lucida Sans Typewriter
Liberation Sans
LMMono10
Lucida Sans Typewriter
LMMono12
LMMathItalic7
LMMathItalic6
LMMathItalic5
LMMathItalic9
LMMathItalic8
Mukti Narrow
LMMathSymbols6
LMMathSymbols7

LMMathSymbols5
FreeSans
LMMathSymbols8
LMMathSymbols9
LMMono8
LMMono9
LMMathExtension10
Lohit Tamil
Tlwg Typo
LMRomanCaps10
UnBatang
Lohit Bengali
LMSansDemiCond10
LMRomanSlant10
LMRomanSlant12
LMRomanSlant17
Waree
gargi
Lohit Hindi
DejaVu Serif
Saab
LMMonoProp10
SimSun-ExtB
Garuda
Rekha
WenQuanYi Bitmap Song
URW Bookman L
LMMonoPropLt10
FreeMono

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Tue, 03 May 2011 12:50:34 GMT
View Forum Message <> Reply to Message

Above test was done on Ubuntu 11.04.

Will do a similar test on Windows as soon as I can. Thanks!

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Tue, 03 May 2011 12:52:27 GMT
View Forum Message <> Reply to Message

STDFONT
serif
sans-serif
monospace

<< was skipped and the font below it was chosen.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Thu, 05 May 2011 16:46:07 GMT
View Forum Message <> Reply to Message

Results for test done on Windows XP

STDFONT
Times New Roman
Arial
Courier New
Symbol
Wingdings
Tahoma
System
Terminal
Fixedsys
Roman
Script
Modern
Small Fonts
MS Serif
WST_Czec
WST_Engl
WST_Fren

WST_Germ
WST_Ital
WST_Span
WST_Swed
Courier
MS Sans Serif
Marlett
Lucida Console
Lucida Sans Unicode
Verdana
Arial Black
Comic Sans MS
Impact
Georgia
Franklin Gothic Medium
Palatino Linotype
Trebuchet MS
Webdings
Estrangelo Edessa
Gautami
Latha
Mangal
MV Boli
Raavi
Shruti
Tunga
Sylfaen
Microsoft Sans Serif
Arial Unicode MS
Book Antiqua
Bookman Old Style
Century
Century Gothic
Garamond
MS Outlook
Wingdings 2
Wingdings 3
MS Reference Sans Serif
MS Reference Specialty

Haettenschweiler
Bookshelf Symbol 7
Bitstream Vera Sans
Bitstream Vera Serif
Bitstream Vera Sans Mono
Myriad Web Pro
Myriad Web Pro Condensed
Arial Narrow
Kartika
Vrinda
Lucida Sans
Free 3 of 9 Extended
Free 3 of 9
DejaVu Sans Condensed
DejaVu Serif
DejaVu Serif Condensed
DejaVu Sans Mono
DejaVu Sans
DejaVu Sans Light
OpenSymbol
MS Mincho
MS PMincho
MS Gothic
MS PGothic
MS UI Gothic
Gulim
GulimChe
Dotum
DotumChe
Batang
BatangChe
Gungsuh
GungsuhChe

MingLiU
PMingLiU

And

rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS

rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = MS UI Gothic
rfnt.GetFaceName() = Arial Unicode MS
rfnt.GetFaceName() = Arial Unicode MS

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Fri, 06 May 2011 08:05:07 GMT
View Forum Message <> Reply to Message

Lance wrote on Tue, 03 May 2011 08:52
STDFONT
serif
sans-serif
monospace

<< was skipped and the font below it was chosen.

Ahaaa!

So the reason is that the name of font itself is translated to CJK. No wonder we cannot find it.

OK, the one simple step to prove this theory: Could you switch your windows to English and retest? (I believe that in english, the name would be SimSun, so it could be found).

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Fri, 06 May 2011 08:36:56 GMT
View Forum Message <> Reply to Message

http://msdn.microsoft.com/en-us/library/dd162620%28v=vs.85%2 9.aspx

"The fonts for many East Asian languages have two typeface names: an English name and a localized name. EnumFonts, EnumFontFamilies, and EnumFontFamiliesEx return the English typeface name if the system locale does not match the language of the font."

Well, well, well, always some surprise waiting to bite us...

I guess the simple fix now is to simply add CJK names to FontCR.cpp, something like:

```
struct sRFace {
 const char *name;
```

```
 dword l, h;
} sFontReplacements[] = {
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x09c00080 },
 { "Arial Unicode MS", 0xfffc3fef, 0xfa7ff7e7 },
 { "SimSun", 0xfd800000, 0x09ffff00 },

 { "MS UI Gothic", 0xffc01008, 0x0fffff00 },
 { "MS Mincho", 0xffc01008, 0x0fffff00 },
```

Above CJK glyphs are just example, please replace with CJK name (I cannot read CJK . Perhaps, if you can, there are more fonts with possibly alternate CHJ name in the table, they would need the same treatment.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Fri, 06 May 2011 12:59:14 GMT
View Forum Message <> Reply to Message

I see. Is it that the linux world also follows MS principle on this regard?

I will do some test and report the result.

Thanks!

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Fri, 06 May 2011 15:51:23 GMT
View Forum Message <> Reply to Message

Hi Mirek:

It works, even though not in the intended way.

I couldn't do it on Windows as the UTF-8 Chinese characters representation will fail MSVC and I don't have MinGW installed yet. But the test on Linux works beautifully. All previously invisible characters now display properly. Placement issue also disappears.

Only strange thing is that I use StdFont, and edit the name in sFontReplacements[] for SimSun only, but the Replacement font actually used is WenQuanYi Zen Hei, which is sans serif while SimSum is serif.

So it seemed edit the name of SimSun (that's the only entry I edited) makes WenQuanYi Zen Hei take precedence over UnDotum which would otherwise be selected.

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Fri, 06 May 2011 17:18:18 GMT
View Forum Message <> Reply to Message

Lance wrote on Fri, 06 May 2011 11:51Hi Mirek:

It works, even though not in the intended way.

I couldn't do it on Windows as the UTF-8 Chinese characters representation will fail MSVC and I don't have MinGW installed yet.

Ah, we have met this MSVC issue before...

The thing to do is that you will have to convert <32 >128 characters in string to escapes (octal or hexadecimal).

One way is to use this 'script'

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

using namespace Upp;

CONSOLE_APP_MAIN
{

}
```

and then paste text from log. Of course, you need to do that in Linux

Quote:
Only strange thing is that I use StdFont, and edit the name in sFontReplacements[] for SimSun only, but the Replacement font actually used is WenQuanYi Zen Hei, which is sans serif while SimSum is serif.

Sounds weird...

Well, whatever. If I may ask you, please fix that table by adding 'cjk' names and post here, so that it can be fixed in svn...

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 01:42:56 GMT
View Forum Message <> Reply to Message

Mirek:

Regarding MSVC character set issue, the suggested way doesn't seem to work all the time. For example, the T file for GridCtrl used to cause trouble with MSVC for zhTW; then recent version of the T file apparently change to the the octel escaped version suggested by you, but it still fails MSVC on my computer, while similarly escaped Russian etc translations are just fine. I don't know exactly why.

As for the sFontReplacements array translation part, I will do in the way you suggested. I guess I will need to put 4 fonts entries: two for the typical serif/sans serif fonts on Windows, and 2 for the counterpart fonts on Linux.

Update: I might be wrong. GridCtrl's T file is no longer causing compilation trouble on Windows.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 02:00:42 GMT
View Forum Message <> Reply to Message

OK, here is what I am going to do:

1. On Windows(Linux), run our test program to enumerate fonts. Filter out non-chinese fonts.

2. Transalate to octet escaped version for font names when necessary

3. Create an entry for each of the font name, with the most favorate 4 fonts (serif/sans serfi on win/lin) listed on top of all chinese font entries.

4. Post the result to this thread.

If you have a better plan, please let me know. Thanks!

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 03:19:07 GMT
View Forum Message <> Reply to Message

{"\346\226\260\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 },//SimSun (or New Song Ti)
{"\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 }, // Song Ti
{"\345\276\256\350\275\257\351\233\205\351\273\221", 0xfd800000, 0x09ffff00 }, //MS Ya Hei
{"\351\273\221\344\275\223", 0xfd800000, 0x09ffff00 }, // Hei Ti
{"\346\226\207\346\263\211\351\251\277\346\255\243\351\273\221", 0xfd800000, 0x09ffff00 }, //WenQuanYi Zheng Hi
{"\346\226\207\346\263\211\351\251\277\347\255\211\345\256\275\345\276\256\347\261\263\351\273\221", 0xfd800000, 0x09ffff00 },//WenQuanYi Wei Hei
{"\344\273\277\345\256\213", 0xfd800000, 0x09ffff00 }, //Fang Song

{"\346\245\267\344\275\223", 0xfd800000, 0x09ffff00 }, // Kai Ti

Above entries should covered the most common and acceptible fonts on both MS and Linux(ubuntu, free fonts) platforms. I am not sure if it will work as well for HongKong/Taiwan/Korean/Japanese users. But now we know where to go to fix similar issues.

Thanks for your effort. U++ becomes more frienldy to CJK users because of it!

I will build the TheIDE with the above entries applied and see how well it works on Linux(Ubuntu) and Windows, and will report the results hopefully within the next 24 hours.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 04:01:04 GMT
View Forum Message <> Reply to Message

Works beautifully on Linux.

On windows, I noticed the font replacement has some issue. See attached picture:

The second Chinese character is notably larger. I believe a different font is applied to it, possibly because it's not represented in the first font which other characters uses.

I will twink the ordering of the entries a little bit for best results.

File Attachments
1) fontOnWin.jpg, downloaded 1325 times

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Sat, 07 May 2011 06:30:37 GMT
View Forum Message <> Reply to Message

OK, I will wait for your reordering attempt, then apply to svn.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 12:59:46 GMT
View Forum Message <> Reply to Message

Hi Mirek:

Sorry for keeping you waiting.

Here is the adjusted sFontReplacements array

```
struct sRFace {
 const char *name;
 dword l, h;
} sFontReplacements[] = {
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x09c00080 },
 {"\346\226\260\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 },//SimSun (or New Song Ti)
 {"\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 }, // Song Ti
 {"\345\276\256\350\275\257\351\233\205\351\273\221", 0xfd800000, 0x09ffff00 }, //MS Ya Hei
 {"\351\273\221\344\275\223", 0xfd800000, 0x09ffff00 },  // Hei Ti
 {"\346\226\207\346\263\211\351\251\277\346\255\243\351\273\221", 0xfd800000, 0x09ffff00 },
//WenQuanYi Zheng Hi
 {"\346\226\207\346\263\211\351\251\277\347\255\211\345\256\275\345\276\256\347\261\263\35
1\273\221", 0xfd800000, 0x09ffff00 },//WenQuanYi Wei Hei
 {"\346\245\267\344\275\223", 0xfd800000, 0x09ffff00 }, // Kai Ti
 {"\344\273\277\345\256\213", 0xfd800000, 0x09ffff00 }, //Fang Song
 { "Arial Unicode MS", 0xfffc3fef, 0xfa7ff7e7 },
 { "MS UI Gothic", 0xffc01008, 0x0fffff00 },
 { "MS Mincho", 0xffc01008, 0x0fffff00 },
 { "VL Gothic", 0xfd800000, 0x09a7ff80 },
 { "VL PGothic", 0xffe00008, 0x0de7ff80 },
 { "UnDotum", 0xe5800000, 0x0aa7ff7e },
 { "UnBatang", 0xe5800000, 0x0aa7ff7e },
 { "DejaVu Sans Mono", 0xffec0004, 0x0fc00080 },
 { "DejaVu Sans", 0xfffd000c, 0x0fc40080 },
 { "AlArabiyaFreeSerif", 0xffdc0008, 0xd8000007 },
 { "Kochi Mincho", 0xffdc0008, 0xd8000007 },
 { "Kochi Gothic", 0xffdc0008, 0xd8000007 },
 { "Sazanami Mincho", 0xffdc0008, 0xd8000007 },
 { "Sazanami Gothic", 0xffdc0008, 0xd8000007 },
 { "Gulim", 0xf7c00000, 0x0ba7ff7e },
 { "PMingLiU", 0xff800000, 0x09ffff00 },
 { "FreeSans", 0xfff23d00, 0x0fc00000 },
 { "FreeSerif", 0xfffd3938, 0x0fc00080 },
 { "Symbol", 0xe4000000, 0x88000002 },
};
```

Turns out "Arial Unicode MS" is the culprit. Some Chinese characters will be intercepted by it.

Not all the entries are strictly necessary. The first two Chinese fonts, Song Ti and SimSun(New Song Ti) are generally available on Windows and Linux platform. They are serif fonts. SongTi(or

SimSun) is the most popular/common font. Most Chinese Characters should be implemented in this(these two) font. In the past, I noticed on Linux platform that some supposedly Hei Ti font were actually rendered using Song Ti because those characters are not implemented in Hei Ti. Sorry for my expression but you know what I mean.

So if Upp doesn't actually differentiate between Serif/Sans Serif in font replacement logic, we should be able to keep the SongTi and SimSun entries only and eliminate other Chinese Font entries.

Thank you again for your attention to this issue! It's very important to me.

Edited by Lance, Reason: SimSun is the way to go. Tried Fang Song, looks great, but apparently it has much smaller implemented character set.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 13:29:05 GMT
View Forum Message <> Reply to Message

Sorry, one more issue. Refer to the following picture:

Notice when I draw it in the plain and simple way in MyApp, the text are compact and nice looking while in TheIDE code editor the same characters looks worse because of the exessive space in between.

If you can make theide code editor display Chinese character as compact as in MyApp or as it displays StdFont, that would be great. If too complicated then forget it.

File Attachments
1) FontOnWin2.jpg, downloaded 1261 times

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 15:39:12 GMT
View Forum Message <> Reply to Message

Sorry but it's getting more complicated than we had expected.

I did test on another Windows XP machine. Here is the font replacement table:

```
struct sRFace {
 const char *name;
 dword l, h;
} sFontReplacements[] = {
```

```
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x09c00080 },
 {"\346\226\260\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 },//SimSun (or New Song Ti)
 {"\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 }, // Song Ti
 {"\345\276\256\350\275\257\351\233\205\351\273\221", 0xfd800000, 0x09ffff00 }, //MS Ya Hei
 {"\351\273\221\344\275\223", 0xfd800000, 0x09ffff00 },  // Hei Ti

 { "Arial Unicode MS", 0xfffc3fef, 0xfa7ff7e7 },
 { "SimSun", 0xfd800000, 0x09ffff00 },
 { "MS UI Gothic", 0xffc01008, 0x0fffff00 },
 { "MS Mincho", 0xffc01008, 0x0fffff00 },
 { "WenQuanYi Zen Hei Mono", 0xfd800000, 0x0ae7ff7e },
 { "WenQuanYi Zen Hei", 0xfd800000, 0x0ae7ff7e },
 { "VL Gothic", 0xfd800000, 0x09a7ff80 },
 { "VL PGothic", 0xffe00008, 0x0de7ff80 },
 { "UnDotum", 0xe5800000, 0x0aa7ff7e },
 { "UnBatang", 0xe5800000, 0x0aa7ff7e },
 { "DejaVu Sans Mono", 0xffec0004, 0x0fc00080 },
 { "DejaVu Sans", 0xfffd000c, 0x0fc40080 },
 { "AlArabiyaFreeSerif", 0xffdc0008, 0xd8000007 },
 { "Kochi Mincho", 0xffdc0008, 0xd8000007 },
 { "Kochi Gothic", 0xffdc0008, 0xd8000007 },
 { "Sazanami Mincho", 0xffdc0008, 0xd8000007 },
 { "Sazanami Gothic", 0xffdc0008, 0xd8000007 },
 { "Gulim", 0xf7c00000, 0x0ba7ff7e },
 { "PMingLiU", 0xff800000, 0x09ffff00 },
 { "FreeSans", 0xfff23d00, 0x0fc00000 },
 { "FreeSerif", 0xfffd3938, 0x0fc00080 },
 { "Symbol", 0xe4000000, 0x88000002 },
};
```

Here is the result of font enumeration on the machine:

```
STDFONT
Times New Roman
Arial
Courier New
Symbol
Wingdings
Tahoma
System
Terminal
Fixedsys
Roman
Script
Modern
Small Fonts
```

MS Serif
WST_Czec
WST_Engl
WST_Fren
WST_Germ
WST_Ital
WST_Span
WST_Swed
Courier
MS Sans Serif
Marlett
Lucida Console
Lucida Sans Unicode
Verdana
Arial Black
Comic Sans MS
Impact
Georgia
Franklin Gothic Medium
Palatino Linotype
Trebuchet MS
Webdings
Estrangelo Edessa
Gautami
Latha
Mangal
MV Boli
Raavi
Shruti
Tunga
Sylfaen
Microsoft Sans Serif
Arial Unicode MS
Book Antiqua
Bookman Old Style
Century
Century Gothic
Garamond
MS Outlook
Wingdings 2
Wingdings 3
MS Reference Sans Serif
MS Reference Specialty

Haettenschweiler
Bookshelf Symbol 7
Bitstream Vera Sans
Bitstream Vera Serif
Bitstream Vera Sans Mono
Myriad Web Pro
Myriad Web Pro Condensed
Arial Narrow
Kartika
Vrinda
Lucida Sans
Free 3 of 9 Extended
Free 3 of 9
DejaVu Sans Condensed
DejaVu Serif
DejaVu Serif Condensed
DejaVu Sans Mono
DejaVu Sans
DejaVu Sans Light
OpenSymbol
MS Mincho
MS PMincho
MS Gothic
MS PGothic
MS UI Gothic
Gulim
GulimChe
Dotum
DotumChe
Batang
BatangChe
Gungsuh
GungsuhChe

MingLiU
PMingLiU

f.GetFaceName() = Arial Unicode MS
f.GetFaceName() = Arial Unicode MS
f.GetFaceName() = Arial Unicode MS

f.GetFaceName() = Arial Unicode MS

And here is the font substitution report:

f.GetFaceName() = Arial Unicode MS
f.GetFaceName() = MS UI Gothic
f.GetFaceName() = Arial Unicode MS
f.GetFaceName() = Arial Unicode MS

SimSun and Song Ti are skipped even though the font are present in the system and they are supposed to take precedence over Arial Unicode MS and MS UI Gothic. The more weired thing is it works just fine on Windows Vista and Ubuntu. Will do a test on a Windows 7 Machine.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 17:47:25 GMT
View Forum Message <> Reply to Message

In a newly installed machine with Windows 7 Home Premium English version and VC10 Chinese installed, I discovered that chinese font name are in English or Pinyin. And neither Song Ti or SimSun is present, while Microsoft YaHei is the default. I can confirm this is yet another SongTi.

looks decent even in small font size. So the following revised entries and ordering should work on most machine.

```
struct sRFace {
 const char *name;
 dword l, h;
} sFontReplacements[] = {
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x09c00080 },
 {"\346\226\260\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 },//SimSun (or New Song Ti)
 {"SimSun", 0xfd800000, 0x09ffff00 },//SimSun (or New Song Ti)
 {"\345\256\213\344\275\223", 0xfd800000, 0x09ffff00 }, // Song Ti
 {"\345\276\256\350\275\257\351\233\205\351\273\221", 0xfd800000, 0x09ffff00 }, //MS Ya Hei
 {"Microsoft YaHei", 0xfd800000, 0x09ffff00 }, //MS Ya Hei
// {"\351\273\221\344\275\223", 0xfd800000, 0x09ffff00 },  // Hei Ti
// {"\346\226\207\346\263\211\351\251\277\346\255\243\351\273\221", 0xfd800000, 0x09ffff00 },
//WenQuanYi Zheng Hi
// {"\346\226\207\346\263\211\351\251\277\347\255\211\345\256\275\345\276\256\347\261\263\3
51\273\221", 0xfd800000, 0x09ffff00 },//WenQuanYi Wei Hei
```

```
// {"\344\273\277\345\256\213", 0xfd800000, 0x09ffff00 }, //Fang Song
// {"\346\245\267\344\275\223", 0xfd800000, 0x09ffff00 }, // Kai Ti
 { "Arial Unicode MS", 0xfffc3fef, 0xfa7ff7e7 },
 { "MS UI Gothic", 0xffc01008, 0x0fffff00 },
 { "MS Mincho", 0xffc01008, 0x0fffff00 },
 { "VL Gothic", 0xfd800000, 0x09a7ff80 },
 { "VL PGothic", 0xffe00008, 0x0de7ff80 },
 { "UnDotum", 0xe5800000, 0x0aa7ff7e },
 { "UnBatang", 0xe5800000, 0x0aa7ff7e },
 { "DejaVu Sans Mono", 0xffec0004, 0x0fc00080 },
 { "DejaVu Sans", 0xfffd000c, 0x0fc40080 },
 { "AlArabiyaFreeSerif", 0xffdc0008, 0xd8000007 },
 { "Kochi Mincho", 0xffdc0008, 0xd8000007 },
 { "Kochi Gothic", 0xffdc0008, 0xd8000007 },
 { "Sazanami Mincho", 0xffdc0008, 0xd8000007 },
 { "Sazanami Gothic", 0xffdc0008, 0xd8000007 },
 { "Gulim", 0xf7c00000, 0x0ba7ff7e },
 { "PMingLiU", 0xff800000, 0x09ffff00 }, // <--- SHOULD MOVE UP
 { "FreeSans", 0xfff23d00, 0x0fc00000 },
 { "FreeSerif", 0xfffd3938, 0x0fc00080 },
 { "Symbol", 0xe4000000, 0x88000002 },
};
```

I still cannot figure out why it would not work on my Windows XP machine. (it's Win XP Professional English version, but many Chinese software has been installed/uninstalled, so its precise condition cannot be determined and reproduced. One thing is for sure, Chinese character in MS Office or OpenOffice are just fine). I will do further investigation. If you can give me some ideas on how to figure out the exact trouble point, I would appreciate that.

Edit: Promote entry for PMingLiu to above that for Arial Unicode MS solve the problem on the WinXP machine. The reason is still unknown. Even though SimSun, SongTi, MS YaHei all are present and work just fine in MS Office and probably many other programs, and enumeration in U++ also shows them, they will somehow report false information to UPP font substitution logic so that they are eliminate as viable candidates.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 18:41:04 GMT
View Forum Message <> Reply to Message

Sorry for throwing too much at you. Here I discovered another issue which I believe is related to Upp way of interpreting UTF-8 characters.

will cause otherwise displayable Chinese characters following it disappear.

Here is a test program:

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

using namespace Upp;

struct MyApp : TopWindow {
 virtual void Paint(Draw& w) {
  const char * texts[]={
   "\346\234\213", //PENG
   "\345\217\213",  //YOU
   "\346\234\213\357\274\214\345\217\213",//PENG CHINESECOMMA YOU
   "\346\234\213\345\217\213\357\274\214\346\234\213\345\217\213" // PENG YOU
CHINESECOMMA PENG YOU
  };

  w.DrawRect(GetSize(), White);
  for(int i=0; i<4; ++i)
   w.DrawText(10, 10+i*30, texts[i]);
 }
};
GUI_APP_MAIN
{
 MyApp().Run();
}
```

Output is something like:

A second issue: On Ubuntu, I applied the above changes to font substitution table and recompiled theide, the Chinese font displays perfect, but this time the input method won't work. Chinese characters entered in the code editor are displayed as narrow blanks, when copy&pasting the blanks to gedit, gedit also display blanks; copy good text from web page or gedit to the code editor works fine.

File Attachments

1) WideComma.png, downloaded 1165 times

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 21:10:36 GMT
View Forum Message <> Reply to Message

Quote:

A second issue: On Ubuntu, I applied the above changes to font substitution table and recompiled theide, the Chinese font displays perfect, but this time the input method won't work. Chinese characters entered in the code editor are displayed as narrow blanks, when copy paster the blanks to gedit, gedit also display blanks; copy good text from website or gedit to the code editor works fine.

Regarding this issue, here is some results of my further experiments. It has been re-confirmed with the most recent version 3407. The test was done on a Ubuntu. G++ is 64bit. The patch to font replacement table has been applied.

When I compile theide using GCC Debug mode, the ide works beautifully: font is pretty, input method works fine; all is good, well almost, as the chinese wide comma issue remains.

When I compile theide using GCC Optimal mode, the ide display existing chinese fonts as well, but Chinese input method doesn't work. Strange, non-displayable characters are inserted, who are invisible to me but will fail the compiler.

BTW, the ide "version.h" file reads:

#define IDE_VERSION "3274-lucid-i386-nogtk"

As this is a pretty old version, the actually nightly release version may already have this issue fixed. That I wouldn't know.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 07 May 2011 21:25:45 GMT
View Forum Message <> Reply to Message

Mirek, if you want a quick status regarding where we are, here you go:

1. Apply the changes to Font Substitution Table as in message #32309 will fix 90% of the problem on both Linux and windows platforms.

2. There is a Wide Comma issue needs further investigation. See message #32311

3. There is a possible [Chinese]Input Method problem. We don't know if it actually exists in the current version of theide; if it does, I am not sure if it's caused by our change to the Font Substitution Table. For some symptom or for double check, please refer to latter part of message #32311 and message #32314.

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Mon, 09 May 2011 03:55:21 GMT

Hi Mirek:

I found out that the WideComma issue is independent of Font Substitution Table. It's even independent of Chinese Characters. The symptom can be further generalized as the following:

1. a string, say "abc" will paint correctly.
2. the WideComma(and WideQuestionMark, maybe more) follows it ("abc") will not display;
3. otherwise good text, say "abc", following the WideComma, will not show
4. If select the WideComma only, or the text following it, the text become visible.

I believe the problem most likely happens in the process of translating the WideComma (or WideQuestionMark) from utf-8 to WString(wchar_t?)

Subject: Re: It's suspected to be an issue with Font.
Posted by mirek on Fri, 13 May 2011 10:30:05 GMT

Lance wrote on Sat, 07 May 2011 17:25Mirek, if you want a quick status regarding where we are, here you go:

1. Apply the changes to Font Substitution Table as in message #32309 will fix 90% of the problem on both Linux and windows platforms.

Applied.

Quote:
2. There is a Wide Comma issue needs further investigation. See message #32311

Fixed.

Quote:
3. There is a possible [Chinese]Input Method problem. We don't know if it actually exists in the current version of theide; if it does, I am not sure if it's caused by our change to the Font Substitution Table. For some symptom or for double check, please refer to latter part of message #32311 and message #32314.

Going to look into it now... Difficult, as I do not know how to install or use input methods

Mirek

Subject: Re: It's suspected to be an issue with Font.
Posted by Lance on Sat, 14 May 2011 21:54:47 GMT
View Forum Message <> Reply to Message

Thanks a lot!

I am going to do another test. Chance is the input method issue on ubuntu may have magically disappeared.

Update[@ Sun, 15 May 2011 00:03]:
Version 3418 works beautifully on Windows. All known issue fixed. I will test on Ubuntu later today. Thanks, thanks, thanks!

2nd Update: Works beautifully on Ubuntu. No input method issue detected!