I want to display Indian languages Unicode scripts.

I was experimenting with changing the script on the fly with selection buttons.

Most of Indian language scripts are multi tier. Generally 3 some times 4.

"upp-indian-fonts.png" file shows the strings as seen in TheIDE and in Notepad++. The text was created in notepad++ and pasted in TheIDE.

Win 7 It is working fine with proper font selection. Button text displayed properly. If I choose wrong font for the button text display then button text is not displayed correctly but "Title" is always displayed correctly.

"Windows-Screen.png" shows windows 7 screens.

Same code compiled in Ubuntu 10.04 is not showing correct font rendering. But title text is rendered correctly.

"Linux-Screen.png" file shows output.

To check my font installation I pasted the code from TheIDE to emacs (Ubuntu). All font rendering is correct.

"Linux-emacs.png" font rendering in emacs on ubuntu.

There are two Indian scripts in the text.

Request for some hints. How to get correct rendering.

Thanks.

Deepak.

File Attachments
1) upp-indian-fonts.png, downloaded 585 times
2) Windows-Screen.png, downloaded 552 times
3) Linux-Screen.png, downloaded 552 times
4) linux-emacs.png, downloaded 542 times
5) multilang.zip, downloaded 473 times

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Mon, 31 Oct 2011 13:36:19 GMT

Now I can get translation to work. In LangInfo.cpp there are defs of Indian scripts. I added these in "t.h" file.

mrIN,knIN,saIN ... about 10

with this change I can now use .t file. My modified t.h file enclosed.

I modified Honza's example file hello.cpp. When I set explicitly the default font with Draw::SetStdFont function the button text rendering is OK. But without this setting button text is not rendered properly.

Both cases Title is correctly rendered. I am using TheIDE-4085 on windows 7.

Any suggestion for getting it right.

Thanks.

File Attachments
1) hello1.zip, downloaded 700 times
2) t.zip, downloaded 426 times
3) MarathiRendering.jpg, downloaded 508 times

Subject: Re: Help for Indian Language Unicode display
Posted by dolik.rce on Mon, 31 Oct 2011 14:43:00 GMT

Hi Deepak,

The reason why the title bar is correct while the button labels are not is because the title is rendered by OS but the content of the window is rendered by U++. I don't understand the code doing this much, but I think it might not be really ready for the multi tier scripts. Hopefully Mirek will be able to give you more detailed info and possibly also fix it.

Best regards,
Honza

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Mon, 31 Oct 2011 19:28:23 GMT

Hi Mirek,

One more observation.

On Ubuntu 11.04 ide-3211 Title text rendered properly. Other text is not rendered correctly (characters appear one after another) even if font is selected with Font::FindFaceNameIndex.

The Lohit fonts are displayed with reference/Display prog. The fonts are avalable for selection in OpenOffice.

On windows DrawText is rendering properly. But on Ubuntu DrawText is selecting fontface but not rendering it properly.

Best Regards.

Deepak

Subject: Re: Help for Indian Language Unicode display
Posted by cbpporter on Wed, 02 Nov 2011 09:28:01 GMT
View Forum Message <> Reply to Message

Wow, forum is back. It dropped as I was submitting my comment and it did not appear. So here is a rephrased and considerably shorter answer.

Unfortunately you can not render a few scripts (including Indian and Arabic) without the rendering engine understanding the specifics of these scripts. It does not work on the principle of just trowing characters out there and they will work. For Indian the text drawing mechanism must know about composition. And for Arabic Unicode stipulates that only the basic character must be encoded, and use of stand alone/beginning/middle/end form must be handled by the rendered.

U++ support basic Latin character substitution, but no substitution, composition and ligatures for other languages. You can not implement these aspects without having a good working knowledge of these scripts. And you will have a hard time getting you patches accepted. U++ has a lot of great features, but last time I checked it was really lagging behind on the Unicode front for non Latin scripts.

In your case you could fix your problem either by creating a custom Display or some custom controls that use the underlying API of your OS to render the text. Windows API can do this easily (and better as the version of Windows increases), X11 can't do anything advanced, but Gtk/Qt are again very good.

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Wed, 02 Nov 2011 14:40:09 GMT
View Forum Message <> Reply to Message

Hi cbpporter,

Thanks for response

I am not looking for correct rendering of Indian scripts in TheIDE. I want it right in my running Application.

In my application display I want to have Indian fonts. So if I change correct default font I am getting correct rendering on windows. It is working with .t file also with on the fly display changing. I want to test it with .tr file.

When I compile the same code on Ubuntu with Draw::SetStdFont it is not rendering properly.

I want some hints on what needs to be done. DrawText renders correctly on windows but not on Ubuntu. But same strings display properly in Emacs and gedit in ubuntu without font settings.

Subject: Re: Help for Indian Language Unicode display
Posted by cbpporter on Wed, 02 Nov 2011 14:52:25 GMT
View Forum Message <> Reply to Message

In your first post, in the "upp-indian-fonts.png" file you have wrong rendering in TheIDE. Can you make your compiled application display the same text correctly? If yes, how do you do that?

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Wed, 02 Nov 2011 18:39:49 GMT
View Forum Message <> Reply to Message

I used Draw::SetStdFont and Button.SetFont functions used. This is working fine in Windows. Not rendering OK in Ubuntu.


```
void multilang::Click(int Lang)
{
  switch ( Lang )
 {
  case 0 :
   Draw::SetStdFont(fntEnglish);
   Title(" English ");
   break;
  case 1 :
   Draw::SetStdFont(fntMarathi);

   break ;
  case 2 :
   Draw::SetStdFont(fntKannada);
```

```
   break ;
  default :
   Draw::SetStdFont(fntEnglish);
   Title(" English ");
   Lang = 0 ;
   break;
 }

 btnMenu1.SetLabel(btnLables[Lang][0]);
 btnMenu2.SetLabel(btnLables[Lang][1]);
 btnMenu3.SetLabel(btnLables[Lang][2]);
};

multilang::multilang()
{
 int i1 ;

 i1 = Font::FindFaceNameIndex("Lohit Kannada");
 fntKannada = Font(i1,20);
 btnKannada.SetFont(fntKannada);

 i1 = Font::FindFaceNameIndex("Lohit Marathi");
 fntMarathi = Font(i1,20);
 btnMarathi.SetFont(fntMarathi);

 i1 = Font::FindFaceNameIndex("Arial");
 fntEnglish = Font(i1,20);
 btnEnglish.SetFont(fntEnglish);

 InitDisplay();
 Click(0);
};
```

Interestingly if I set Setup->Environment->Fonts->Normal to "Lohit Marathi" or "Lohit Kannada"
IDE is showing correct rendering in windows of Marathi or Kannada fonts.

IDE with MINGW compiles and renders properly.
IDE with MSC10 gives following warning and runs with wrong font rendering. Warning is for every
non ASCII char.

C:\MyApps3991\multilang\main.cpp(58) : warning C4566: character represented by
universal-character-name '\u0CBE' cannot be represented in the current code page (1252)
C:\MyApps3991\multilang\main.cpp(58) : warning C4566: character represented by
universal-character-name '\u0C9F' cannot be represented in the current code page (1252)
C:\MyApps3991\multilang\main.cpp(58) : warning C4566: character represented by
universal-character-name '\u0C95' cannot be represented in the current code page (1252)

Subject: Re: Help for Indian Language Unicode display
Posted by mr_ped on Thu, 03 Nov 2011 08:08:19 GMT
View Forum Message <> Reply to Message

In windows the U++ application call OS API to render the text, thus it works.

In linux much of the text rendering is done either in U++ or in X11, so it can't render the text correctly. Window title is rendered by OS (probably GTK code), it's different code path.

To get correct font rendering in application you have to either call some better text rendering (GTK/Qt), or fix the U++ font renderer (by fix I mean to add all the needed code to render composed characters better = lot of work).

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Sun, 06 Nov 2011 04:57:03 GMT
View Forum Message <> Reply to Message

In my application I want to have Indian Language support on Windows and Linux.

I understand it will be lot of work on Linux and I want to attempt it.

First step probably will be to start showing labels in Indian scripts. Then extend it to other components.

I want some guidance / hints on how to start.

In upp makefile I see libpango and libpangocairo is linked. is pango used for font rendering in UPP?.

Which area of the uppsrc code I should start with?

If libpango is to be used for font rendering then any example available ?

With lib pango probably we can use other pango libs

pango-arabic-fc.so
pango-arabic-lang.so
pango-basic-fc.so
pango-basic-x.so
pango-hangul-fc.so
pango-hebrew-fc.so
pango-indic-fc.so
pango-indic-lang.so
pango-khmer-fc.so
pango-syriac-fc.so
pango-thai-fc.so
pango-tibetan-fc.so

Subject: Re: Help for Indian Language Unicode display
Posted by mirek on Mon, 07 Nov 2011 08:38:36 GMT
View Forum Message <> Reply to Message

First of all, sorry for delay.

U++ generally supports left-to-right scripts.

Now I am not much informed about indian scripts, but if they are left-to-right, there is a good chance we can get it working.

So if they are, the most likely cause they are not displaying correctly in Linux is missing substitution fonts.

When U++ does not find required glyph in requested font, it goes through substitution fonts and tries to locate it there.

The substitution fonts are in Draw/FontCR.cpp in sFontReplacements table. So far we mostly cared about CJK fonts there. So perhaps adding some line there might fix your problem. As for those constants on lines, for now you can just put there 0xffffffff, these are only used to speedup the process, we can do that later.

Mirek

Subject: Re: Help for Indian Language Unicode display
Posted by cbpporter on Mon, 07 Nov 2011 08:58:12 GMT
View Forum Message <> Reply to Message

Wouldn't it be a good idea to provide a code path that uses Gtk under Linux for font rendering. Like the one for Windows, with the difference being that WinAPI does some font substitution while the X one does not? I'm not sure what Gtk uses? Pango maybe. ?Or was it Cairo? It won't work for NOGTK, but it would provide much better results. There is no way internationalization support under U++ is going to get as good as even Gtk (which is not the best) in a reasonable amount of time with the current dev composition.

Subject: Re: Help for Indian Language Unicode display
Posted by mirek on Mon, 07 Nov 2011 13:05:17 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Mon, 07 November 2011 03:58Wouldn't it be a good idea to provide a code path that uses Gtk under Linux for font rendering. Like the one for Windows, with the difference being that WinAPI does some font substitution while the X one does not? I'm not sure what Gtk uses? Pango maybe. ?Or was it Cairo? It won't work for NOGTK, but it would provide much better results. There is no way internationalization support under U++ is going to get as good as even Gtk (which is not the best) in a reasonable amount of time with the current dev composition.

I am afraid Pango is too highlevel for us and there is nothing lower-level...

Subject: Re: Help for Indian Language Unicode display
Posted by chickenk on Tue, 08 Nov 2011 08:37:24 GMT
View Forum Message <> Reply to Message

mirek wrote on Mon, 07 November 2011 14:05I am afraid Pango is too highlevel for us and there is nothing lower-level...

I'm not quite sure, but maybe Harfbuzz?

Here's a very interesting reading: http://behdad.org/text/

EDIT: I suggest this because I know that the stack used in the Enlightenment project, for example, is freetype/fontconfig/fribidi/harfbuzz.

Cheers
Lionel

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Tue, 08 Nov 2011 15:42:04 GMT
View Forum Message <> Reply to Message

Hi Mirek

I will try first font replacement.

Indic scripts are Left-to-Right. General info about Indic fonts available @ http://en.wikipedia.org/wiki/Devanagari .

The main difference from other scripts is Conjuncts (combination of multiple glyphs) Representation of conjuncts is quite complex.
Also Combination of Vowels and Consonants also creates kind of different glyph.

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Mon, 21 Nov 2011 11:45:33 GMT

Hi,

I was traveling last 2 weeks. Hence could not work on this.

I have added entries in Draw/FontCR.cpp , entries with 0xffffffff are mine.

```
{"Microsoft YaHei", 0xfd800000, 0x9ffff00f }, //MS Ya Hei
{"gargi",0xffffffff,0xffffffff}, // Gargi
{"Chandas",0xffffffff,0xffffffff}, // Chandas
// {"Kedage",0xffffffff,0xffffffff}, // Gargi
// {"Mallige",0xffffffff,0xffffffff}, // Gargi
{"Lohit Hindi",0xffffffff,0xffffffff}, // Lohit Hindi
// {"\351\273\221\344\275\223", 0xfd800000, 0x09ffff00 },  // Hei Ti
```

There is no difference with or without these lines.

But what I noticed is it is using correct font face. These fonts have Ascii and Devanagari Glyphs.
Font face selection for rendering is ok but rendering of Devanagari is not correct in linux.

Example of conjunct Devanagari.

Linux rendering. Using RichTextView

Windows rendering

File Attachments

1) Devanagari.png, downloaded 1023 times
2) Linux-Devanagari.jpg, downloaded 1059 times
3) Windows-Devanagari.png, downloaded 1078 times

Subject: Re: Help for Indian Language Unicode display
Posted by deep on Thu, 24 Nov 2011 14:23:56 GMT

Referring to Lance thread about CJK fonts. I have done some tests.
Results from log file for DDUMP

Results are with following change in struct sRFace


```
sFontReplacements[] = {
 { "sans-serif", 0xffee0008, 0xdc000801 },
 { "Arial", 0xfffe0000, 0x9c000801 },
 {"\346\226\260\345\256\213\344\275\223", 0xfd800000, 0x9ffff00d },//SimSun (or New Song Ti)
 {"SimSun", 0xfd800000, 0x9ffff00d },//SimSun (or New Song Ti)
 {"\345\256\213\344\275\223", 0xfd800000, 0x9ffff00d }, // Song Ti
 {"\345\276\256\350\275\257\351\233\205\351\273\221", 0xfd800000, 0x9ffff00f }, //MS Ya Hei
 {"Microsoft YaHei", 0xfd800000, 0x9ffff00f }, //MS Ya Hei
 {"gargi",0xffffffff,0xffffffff}, // Gargi
 {"Chandas",0xffffffff,0xffffffff}, // Chandas
 {"Kedage",0xffffffff,0xffffffff}, // Gargi
 {"Mallige",0xffffffff,0xffffffff}, // Gargi
 {"Lohit Hindi",0xffffffff,0xffffffff}, // Lohit Hindi
```


```
 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();
```


```
FormatIntHex(chr) = 00000930
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000
FormatIntHex(chr) = 0000093e
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
```

FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000
FormatIntHex(chr) = 00000937
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000
FormatIntHex(chr) = 0000094d
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000
FormatIntHex(chr) = 0000091f
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000
FormatIntHex(chr) = 0000093f
FormatIntHex(th) = 40000000
Font(rface[i], 10) = <sans-serif:10>
FormatIntHex(h[i]) = dc000801
FormatIntHex(h[i] & th) = 40000000
Font(rface[i], 10) = <gargi:10>
FormatIntHex(h[i]) = ffffffff
FormatIntHex(h[i] & th) = 40000000

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

Results for font list.

STDFONT

serif
sans-serif
monospace
UnDotum
LMMonoLt10
Samyak Devanagari
Century Schoolbook L
OpenSymbol
Khmer OS System
Nakula
Chandas
LMSansQuot8
Lohit Nepali
LMMathSymbols10
LMRomanSlant9
LMRomanSlant8
LMSans9
LMSans8
Mukti Narrow
Meera
Kalimati
Vemana2000
Lohit Maithili
LMMonoSlant10
Umpush
Purisa
Pothana2000
DejaVu Sans Mono
Norasi
Loma
URW Palladio L
Phetsarath OT
Sawasdee
Sahadeva
Tlwg Typist
URW Gothic L
Dingbats
URW Chancery L
FreeSerif
ori1Uni
WenQuanYi Micro Hei Mono
Kedage
DejaVu Sans
Kinnari
LMSans17
LMSans12
LMSans10
Lohit Punjabi

LMRoman17
LMRoman12
LMRoman10
TlwgMono
Symbol
LMRomanDunh10
LMRoman7
LMRoman6
LMRoman5
LMRoman9
LMRoman8
Bitstream Charter
KacstOne
Lohit Kashmiri
Khmer OS
Liberation Mono
Courier 10 Pitch
Nimbus Sans L
TlwgTypewriter
TakaoPGothic
LMRomanDemi10
Rachana
WenQuanYi Micro Hei
LMMonoCaps10
Samanata
LMMonoLtCond10
Standard Symbols L
Lohit Marathi
Lohit Gujarati
Nimbus Mono L
Nimbus Mono L
Liberation Serif
Lohit Sindhi
Mallige
LMMathItalic10
Nimbus Roman No9 L
LMMathItalic12
LMRomanUnsl10
Lohit Konkani
Liberation Sans
LMMono10
LMMono12
LMMathItalic7
LMMathItalic6
LMMathItalic5
LMMathItalic9
LMMathItalic8
Mukti Narrow

LMMathSymbols6
LMMathSymbols7
LMMathSymbols5
FreeSans
LMMathSymbols8
LMMathSymbols9
Sarai
LMMono8
LMMono9
LMMathExtension10
Lohit Tamil
Tlwg Typo
LMRomanCaps10
UnBatang
Lohit Bengali
LMSansDemiCond10
LMRomanSlant10
LMRomanSlant12
LMRomanSlant17
Waree
gargi
Lohit Hindi
DejaVu Serif
Saab
LMMonoProp10
Garuda
Rekha
URW Bookman L
LMMonoPropLt10
FreeMono