Subject: Re: A terminal emulator widget for U++
Posted by Oblivion on Sat, 13 Jul 2019 15:35:03 GMT
View Forum Message <> Reply to Message

Hello,

It's been a while since I posted any news about the Terminal package. I had to focus on my other works fo a while.

The good news is last week I implemented the final missing pieces for the 0.1 release:

- 256 colors support added.
- ANSI colors support added.
- XTerm dynamic colors suppport added.
- VT 4xx rectangular area operations are added (copy, invert, move, fill, in both selective and normal modes)
- UDK (DEC's user-defined function keys support) added.
- Lazy resize option is added (to reduce flickers on network terminals uch as SSH-based ones)
- Size hint added.
- A .usc file is added to the Terminal package. (The most common options (font, ink, paper, cursor, sizehint etc.) can be set using TheIDE's layout editor. Also it shows a size hint (in calculated cell size) to simplfy positiong the widget in the layout ediyor.
- It is also tested on Windows, and it works well. :) (currently as SSH terminal, in the near future as a frontend for Windows power shell too)

Two notes on the upcoming initial release:

1) Terminal package currently does not contain any external code/libaray. It uses U++, and it's plugins. :)
2) Although a virtual terminal requires a pty device, and Terminal pacjage contains one, they are completely decoupled.
   Terminal ctrl can be used and compiled without PtyProcess. This gives it a huge flexibility
   In this regard I will provide 4 basic examples with the package:
   - TerminalExample | Uses ptyprocess (currently ptyprocess requires POSIX-compliant operationg systems (or possibly cygwin on Windows.)
   - TerminalExampleWithLayout | The same as above.

   - SShTerminalExample | Does not use PtyProcess. IT uses Core/SSH package isntead
   - SshTerminalExampleWithLayout | The same as above.

Here is the actual code of TerminalExample (36 LOCs total):

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

```
using namespace Upp;

const char *nixshell = "/bin/bash";

struct TerminalExample : TopWindow {
 Terminal term;
 PtyProcess pty;

 TerminalExample()
 {
  term.WhenBell =   [=]()  { BeepExclamation(); };
  term.WhenTitle =   [=](String s) { Title(s); };
  term.WhenResize =   [=]()  { pty.SetSize(term.GetPageSize()); };
  term.WhenOutput =   [=](String s) { PutGet(s); };
  SetRect(term.GetStdSize()); // 80 x 24 cells (scaled).
  Sizeable().Zoomable().CenterScreen().Add(term.SizePos());
  SetTimeCallback(-1, [=]()  { PutGet(); });
  pty.Start(nixshell, Environment(), GetHomeDirectory());
 }

 void PutGet(String out = Null)
 {
  term.Write(pty.Get());
  pty.Write(out);
  if(!pty.IsRunning())
   Break();
 }
};

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

Below was a sort of "final boss" for the first release. It shows the mapscii, an OpenStreetMap implementation for terminal devices, running on the above code and on Gnome-terminal.
On the left is TerminalExample, running mapscii.
On the right is gnome terminal running mapscii
Both are running on 256 colors mode + mouse tracking support. :)

As a final note: Terminal package will be availabe within this weeek.

Best regards,
Oblivion

File Attachments

1) Terminal.png, downloaded 1750 times