Subject: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 09:39:06 GMT
View Forum Message <> Reply to Message

does anyone know how to incorporate an additional core console window (additional task) somewhere in a gui app as a frame?

i'm looking howto place a python console window in an application..
any hints?

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Thu, 10 Feb 2011 11:08:11 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Thu, 10 February 2011 10:39does anyone know how to incorporate an additional core console window (additional task) somewhere in a gui app as a frame?

i'm looking howto place a python console window in an application..
any hints?
I think the easiest way around this would be to create new widget that has a python running in LocalProcess. It could probably be based on LineEdit to save some coding, if you overwrite some keys (e.g. to prevent moving cursor on previous lines) and send the current command to underlaying process on Enter... Never tried anything like that, but it might work

Btw: If it works, you could turn it into general ConsoleCtrl, so that one could interactively run any CLI app, maybe even bash or cmd.exe

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 12:33:21 GMT
View Forum Message <> Reply to Message

i already thought of sth like this. but it seems quite difficult. i'd prefer to have a native window space.. but it seems as if it is not easy to solve. otoh visual studio does sth similar as well, eclipse has a console window too..but i dont know if it's a native or simple widget with redirecting input / output as you stated.
every system interaction is them posted via system native exec calls, i think.

i thought of having the custom control (LineEdit, DocEdit) write its generated input to a Stream, and have another Stream as the receiver.

the flow control in this case is different, opposed to normal processes that read / write blocking to stdin/stdout. in the control i can't read the stream blocking. i could, using a separate thread that reads the Stream, which uses blocking read for its _Get(). but it's a whole different setup. short:

console apps are dataflow based, while gui apps are event based.

i already have a python environment working. it's quite easy. but having a test envirnment, and the ability to include such a prompt in own app would make a great deal in upp. redirecting I/O properly is as usual, the problem..

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Thu, 10 Feb 2011 12:57:21 GMT
View Forum Message <> Reply to Message

I know for sure that Gnome implements this as a widget (I use it in geany editor). On windows, I've seen how to do it using frameless console window positioned so that it looks like it is part of the app - which is ugly hack I am not an expert, but I think there is no simple way how to force windows console to render in defined window (e.g. pointing it to DHCtrl).

I don't think it would be much of a problem to python feed the python ( ) with the commands. Theide already does the same when debugging with gdb, it just doesn't have the console-like interface.

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by mdelfede on Thu, 10 Feb 2011 13:03:42 GMT
View Forum Message <> Reply to Message

I guess you shall spawn a terminal process and then use a pipe for communication, even in non-blocking mode or with multithreading.
It's not so easy, but still not overcomplex.

You can take a look at SysExec package for pipe usage in Linux, and on Uniq package for named pipes usage in both (Windows can be quite tricky....).

In non-blocking mode you could poll the input pipe for incoming chars and send to an edit control.

Ciao

Max

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 13:04:07 GMT
View Forum Message <> Reply to Message

i dont mind programming own control for that. i merly seek a good concept on how to redirect /

process std streams in a decent manner
thanks for support.

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by mdelfede on Thu, 10 Feb 2011 13:10:36 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Thu, 10 February 2011 14:04i dont mind programming own control for that. i merly seek a good concept on how to redirect / process std streams in a decent manner
thanks for support.

You shall redirect them to a pipe and use it in non-blocking mode on client side or use a separate thread (with PostCallback() to communicate with main one) if you want to stay with blocking mode.

First is trickier but more reliable on shutdown, latter is easier but you can have problems on shutdown.

Ciao

Max

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 13:45:10 GMT
View Forum Message <> Reply to Message

scenario:

a console process is running separately, there is no option to do it in same thread due to blocking read desire from stdin.

the console process (or any other)

its read end (stdin) has been wired up to read from a pipe end, the write end is fed by some control of upp, LineEdit key input or what ever. this is easy. this side can be blocking on writing to pipe, the process is blocking reading from pipe

its write (stdout) end must also be another pipe to which the process may write blocking. the read end would need to be polled by upp gui say every 100 ms. this can be done in a timer callback.. so the gui directly processes the read pipe data.

thus arbitrary processes can be hooked up in upp, even a common cmd.exe console.

i'll try that one.

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Thu, 10 Feb 2011 13:54:56 GMT
View Forum Message <> Reply to Message

I think you are taking it too seriously Here is a working example (change the paths if you are on windows): #include <CtrlLib/CtrlLib.h>
using namespace Upp;

```
class ConsoleCtrl:public LineEdit{
 typedef ConsoleCtrl CLASSNAME;
 LocalProcess p;
 int last;
public:
 virtual bool Key(dword key,int count){
  switch(key) {
  case K_ENTER:{
   String cmd=line.Top().text.Right(GetLength()-last)+'\n';
   Insert(GetLength(),"\n");
   p.Write(cmd);
   }
   return true;
  case K_UP:    //might implement history browsing...
  case K_DOWN:
  case K_PAGEUP:
  case K_PAGEDOWN:
  case K_CTRL_PAGEUP:
  case K_CTRL_HOME:
  case K_CTRL_PAGEDOWN:
  case K_CTRL_END:
   return true;
  }
  return LineEdit::Key(key,count);
 }
 void Read(){
  String s;
  p.Read(s);
  if(s.IsEmpty()) return;
  Insert(GetLength(),s);
  last=GetLength();
  SetCursor(last);
 }
 ConsoleCtrl(const String& cmd="/bin/bash"){
  p.Start(cmd);
  SetTimeCallback(-50,THISBACK(Read));
 }
 ~ConsoleCtrl(){
  p.Kill();
 }
};
```

```
class App :public TopWindow {
 typedef App CLASSNAME;
 ConsoleCtrl c;
public:
 App():c("/usr/bin/python -i"){  // -i is required to use it interactively
  Add(c.SizePos());
 }
};

GUI_APP_MAIN{
 App().Sizeable().Run();
}
```

It is just quickly hacked to gather, but the general idea seems to be working just fine. I tested it with bash and python and it feels almost like a terminal, except for missing features like history or tab completion... but those could be probably implemented too with a bit of work.

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 14:02:55 GMT
View Forum Message <> Reply to Message

genious ! works on windows as well, using "cmd".

thats what makes the difference knowing upp, i didnt know about LocalProcess...

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Thu, 10 Feb 2011 14:20:58 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Thu, 10 February 2011 15:02genious ! works on windows as well, using "cmd".

thats what makes the difference knowing upp, i didnt know about LocalProcess...
I mentioned it in the first post Also note that there is couple more: RemoteProcess and SlaveProcess, but I never used those, so I don't know much.

Actually after bit more thinking it would be OK to implement it to run just a shell (cmd or sh, platform dependently). And run any other command (e.g. python) from within...

Anyway, I'm glad that it helped you and also really amazed that it works on windows without much trouble

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 14:34:50 GMT
View Forum Message <> Reply to Message

i'll try to implement the basic stuff additionally, like cursor movements and command cache.
and make a package, maybe this can go to bazaar or example...

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Thu, 10 Feb 2011 15:14:31 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Thu, 10 February 2011 15:34i'll try to implement the basic stuff additionally, like cursor movements and command cache.
and make a package, maybe this can go to bazaar or example...
Cursor movements and other console commands (color, clear, ...) would be nice. Also I believe that if it works fine it could even get into CtrlLib. There is definitely a use for it, e.g. the "Run in console" mode in theide would definitely benefit from nice interactive ConsoleCtrl

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 19:52:19 GMT
View Forum Message <> Reply to Message

i think we have lost one point here.

normally, any console application expects direct keyboard input, ascii codes, and it eventually echoes them back. but in this case we do the preprocessing ourselves, trying to generate pseudo ascii stream.

any idea how to acess the ascii codes from down layer to feed the process directly?

EDIT: another problem: do you know why the 'cmd' is not echoing back what it receives through the pipe? this is quite a normal behaviour...
i.e. try 'nslookup' with no params, it also opens a interactive mode, but inputing stuff is not echoed. when it is started through 'Run' or with cmd, it does.

or is the echoeing buisness from the shell itself? since the programs usually read until '\n' and then start processing. and they themselves dont echo or do they?

seems as if the toooop most shell is echoing, subshels dont.

'python -i' works in windows as well...

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Thu, 10 Feb 2011 20:58:42 GMT
View Forum Message <> Reply to Message

here comes a simplified version with echo support. i use another LineEdit to process the keys..dont know a better faster solution for now. i'd be great to be able to make all the key strokes visible / accessible, just as in bash..

File Attachments

1) ConsoleCtrl.zip, downloaded 304 times

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by dolik.rce on Fri, 11 Feb 2011 08:29:38 GMT
View Forum Message <> Reply to Message

I'm afraid I can't help you much with the behavior of cmd - I don't have windows POSIX shells don't echo, it is business of the terminal. The idea is the same as if you feed the script to shell using a pipe: "cat script.sh | sh". This would print just the output of the script, not its content. We are doing basically the same thing. Everything that user types stays in the LineEdit, so it should be not necessary to have any echoing. The linux version of nslookup works fine with the simple code I posted before.

I tried your code, but it has one great fault. You are processing each character separately, feeding it directly to console. It is in principle possible, but than you have to take care about control sequences (cursor movement, delete, backspace), otherwise you're sending crap I agree that some programs might need it (if they expected single letter commands without trailing \n), but it is much more work to do.

Honza

Subject: Re: Q: howto incorporate a native console window in GUI
Posted by kohait00 on Fri, 11 Feb 2011 10:32:21 GMT
View Forum Message <> Reply to Message

thanks,

seems like in cmd it's the same, cmd.exe is only the command interpreter thar react on commands when '\n' arrives. echoing must be done by terminal.. @echo off will probably modify the global terminal setting in that case.

yes i agree with you in terms of sending a whole string at once..
but what about the arbitrary cases they are there.. i.e. starting telnet which would connect to another terminal, or even ssh..they expect char by char, and they echo them back.

i was thinking of an additional TerminalCtrl, which could do what you suggested. where one can

hook up an interpreter to a 'Callback1<const String&> WhenCommand;'
and has Stream to which generated output can be written. it directly prints where the cursor is moving it along..

cursor handling is another important topic..not quite trivial, as you see during linux boot sequence or anywhere in a build script run.. this comes from the composed echoing and printing..
ofcorse, the char commands are there to move the cursor..

we could consider implementing such things in a simple manner..

an important issue is the separation of print view and input buffer of the command though..
