Subject: Re: send/receive arguments to console(cmd/terminal) in GUI app
Posted by dolik.rce on Tue, 05 Jul 2011 21:39:08 GMT
View Forum Message <> Reply to Message

dave wrote on Tue, 05 July 2011 22:28like you said, the string sys function worked just fine. it was heartning to see calc.exe run, but things were short lived. when i tried to give other commands, it failed to execute and GUI stopped responding, even though its process was running (as seen in task manager).

when calc.exe did run, the GUI app was left unresponsive till calc.exe was closed.
It is because Sys() waits till the executed program ends, so it can return its output. If you wish to use your app in between, use LocalProcess:

```
#include <CtrlLib/CtrlLib.h>
using namespace Upp;

class App:public TopWindow {
 typedef App CLASSNAME;
 Array<LocalProcess> p;
 Button b;
public:
 App(){
  Add(b.HCenterPos().VCenterPos());
  b.SetLabel("Push me!");
  b<<=THISBACK(Launch);
 }
 void Launch(){
  // we add a new process to the array and start a command
  p.Add().Start("calc.exe");
 }
 ~App(){
  // if you wish the processes run after closing the app, detach them like this:
  for(int i=0;i<p.GetCount();i++){
   p[i].Detach();
  }
 }
};

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

dave wrote on Tue, 05 July 2011 22:28also the cmd window remains hidden. can i show the cmd window?IIRC it is Windows itself who decides whether to open the cmd or not. If a GUI app is executed the cmd prompt is not displayed and the apps output/input is not bound to it. (Try executing calc.exe from cmd, if I'm correct it will start the calculator and immediately let you work with the shell). I haven't use windows in a while though, so I might be slightly misinterpreting some things.

You might also try to execute whatever you want as "cmd /c your_command", that should open the cmd.

Honza