Posted by Giorgio on Thu, 22 Mar 2018 12:32:44 GMT
View Forum Message <> Reply to Message

Hi there,
I have a function that executes a process (an external program that works in background, so the user is not aware about what is happening). This process can take sometime to elaborate data, so I would like to add a progress for the user to know that the process is still running. I do not need to show the real progress, just showing the running bar. I would like also the ability for the user to interrupt the process in case it takes too much time (may happen that the external program has problems...). If possible, I would prefer to avoid multithreading. So far, I did this:

```
void Capture::Process()
{
 p("Processing data..."); //Progress p; is a global variable for the class
 p.Create();
 p.Canceled();
 DoProcess();
 p.Close();


}

void Capture::DoProcess()
{
 std::ofstream lockfile (imgjlock); //Create a lockfile, when the external program is done it deletes
the lockfile
 lockfile.close();
  ShellExecute(NULL, NULL, batfile, NULL, NULL, SW_HIDE);
  while(true)
  {
   p.Step();
   if(!FileExists(imgjlock)) {
    p.Close();

  //Here some code to show the results

    break;
   }
  }
 } else {
  remove(imgjlock);
 }
}
```

...but it does not work.

It would be very useful if someone could point me to some example.

Regards,
gio