Subject: Re: Adding a socket to a GUI application
Posted by Giorgio on Wed, 26 Sep 2018 16:51:35 GMT
View Forum Message <> Reply to Message

Hi there,
I put this question aside for a while, few days ago I resumed it.

I had a look to threads and tcpsocket as suggested. For a starter I decided to focus on thread. So, I forked before the .Run() and put in the thread the code to manage the socket. I ended up with this:

```
void tagidSocket()
{
 //Socket
 RLOG("Socket's thread started");
 for(;;) {
  if(Thread::IsShutdownThreads())
   return;
  RLOG("I'm still alive");
  Sleep(1000);
 }
}

GUI_APP_MAIN{
 //Reading a .ini file, connection to a db
 [...]

 Thread t;
 t.Run(callback(tagidSocket));
 app.Run();

 RLOG("Exiting, terminating socket");
 Thread::ShutdownThreads();
}
```

This works as expected: I can use my application normally, I can see in the log the sentence "I'm still alive" several times, and when I close the applications it exits nicely / does not hung up.

After that I put the socket management in the equation and here came the problems. I began using the very same code used in the example "SocketServer". I copied everything in my tagidSocket() function. This is the result (GUI_APP_MAIN does not change):

```
void tagidSocket()
{
 TcpSocket server;
```

```
 if(!server.Listen(23456, 5)) {
  RLOG("Unable to initialize server socket");
  return;
 }
 RLOG("Socket started, waiting for requests...");
 for(;;) {
  if(Thread::IsShutdownThreads())
   return;
  TcpSocket s;
  if(s.Accept(server)) {
   String w = s.GetLine();
   //Cout() << "Request: " << w << " from: " << s.GetPeerAddr() << '\n';
   RLOG("Request: " + w + " from: " + s.GetPeerAddr() + '\n');
   if(w == "time")
    s.Put(AsString(GetSysTime()));
   else
    s.Put(AsString(3 * atoi(~w)));
   s.Put("\n");

  }
 }
}
```

With this change, when I launch my application everything is ok, I can connect to the socket and exchange data, but when I close my application it hangs and I have to kill it manually.

I try to debug the problem and I found out that the code responsible for the problem is the following: if(s.Accept(server)) { [...] }.

If I comment out everything in the if above (and also the if itself), the application can be closed normally (but of course a socket without the "listening" part makes no sense).

Why is this happening?

Regards,
gio
