Subject: Deadlock - what LeaveGMutexAll does?
Posted by pete82 on Sun, 01 Jan 2012 11:35:28 GMT
View Forum Message <> Reply to Message

Hi!
I'm creating simple network monitoring application. I've two threads - one thread serves network and the other is main. I've also some lua bindings, so i can do some fast prototyping and modify code on the fly (which works). Lua is single threaded, so i've mutex, call it lM.
Now because i've overriden Paint which is called with GuiLock held (as i understand), i have to acquire GuiLock and then lM to prevent deadlocks (the Paint is served in my lua code) for every lua access (i'm painting in lua often anyway).
I've also some timers which are timing GUI refreshes from network thread, and the refreshes are called from main thread (this was not intentional, but seems good to me).
I've still some deadlocks, which i'm unable to solve. So i fired up good old ollydbg (theide was not showing correct info) and it seems to me, that the problem is LeaveGMutexAll in Ctrl::ICall.
The network thread is holding lM and GuiLock, the timer fires or smt else and the GuiLock is released. So the first thread can acquire it, but can't acquire lM.
Or am i wrong? How can i solve this?
The call sequence(or stack) is smt like this

```
main->timer->GuiLock->waiting on lM to call timer function
thread->GuiLock->lM->Lua->refresh->waiting in UPP
```

I can provide more precise stack traces, but it takes some time to reproduce...
Thnx for answers.