Subject: Re: Use same variable in different threads
Posted by tojocky on Mon, 13 Dec 2010 10:35:39 GMT
View Forum Message <> Reply to Message

koldo wrote on Mon, 13 December 2010 00:16Hello all

Nice answers !

For now if I understand well I will use:

- AtomicVar class. Nice!

- Mutex as here:

```
struct mySharedData {
 int a, b;
 String c;
 Mutex m;
};

mySharedData data;

void ThreadFunction(){
 for(int i = 0; i < 10; i++){
  Thread::Sleep(Random(10)); // Pretend some work...
  data.m.Enter();     // Enter the section that accesses the shared data
  data.a = i;
  data.c << data.a << "\n";
  Thread::Sleep(20); //Let's pretend that some slow operation happens here (for example file
access)
  data.b = data.a;
  data.m.Leave();    // we don't need the exclusive access to data anymore, release the mutex
 }
}
```

But what about read control?
Example:
read data.a
write data.a and data.b
read data.b (b will be modified)

In this case I think:
1. the write look need to wait until the all reads was finished. and every read locks need to wait until the write lock was finished.
2. Or reads and writes will use only one lock (data.m). But in this case will not possible multiple parallel reads.

Later I will try to provide a simple example.