Subject: Re: StaticMutex/ONCELOCK question
Posted by mirek on Tue, 03 Feb 2009 06:41:36 GMT
View Forum Message <> Reply to Message

Novo wrote on Tue, 03 February 2009 00:28I couldn't understand completely several things with StaticMutex and ONCELOCK.

StaticMutex will never call destructor of a contained Mutex object. Is this meant to be?

Yes. OS will clean that up when program exits.

Quote:

```
#define ONCELOCK \
for(static volatile bool o_b_; !ReadWithBarrier(o_b_);) \
 for(static StaticMutex o_ss_; !o_b_;) \
  for(Mutex::Lock o_ss_lock__(o_ss_); !o_b_; BarrierWrite(o_b_, true))
```

How the above code actually works?

TIA

Do not get fooled by 3 'for' loops - these are just syntactic sugar to make ONCELOCK work on C statements and blocks - they in fact simulate the outer block

```
{
   static volatile bool o_b_;
   if(!ReadWithBarrier(o_b_)) {
      static StaticMutex mutex;
      mutex.Enter();
      {
         do_the_initialization - the statement 'body'
         BarrierWrite(o_b_);
      }
   }
}
```

The purpose is to avoid locking mutex in subsequent passes of ONCELOCK - you need the barrier code to do that.

Note that both compilers we use optimize the for loops away.

Mirek