Subject: Re: Sharing and Locking
Posted by gridem on Sun, 21 Mar 2010 10:39:22 GMT
View Forum Message <> Reply to Message

luzr wrote on Sun, 21 March 2010 09:37
You miss the point: When the file is closed?

(I know when, of course, but the point is the shared ownership makes this very uncertain).
OK, usage sample:

```
void SetterThread()
{
   for (int i = 0; i < cycles; ++ i)
   {
      // create file object
      FileObject file;
      // assign reference to global variable
      *DataAccess::Access() = file;
      // create file itself
      file.Open("file.txt");
      // write some text, file will be opened because accesser doesn't use close
      // (try ... catch is not needed)
      file.Write(String().Cat() << "[" << i << "] setter");
      // close the file, accesser now cannot write into file
      file.Close();
   }
}

void AccesserThread()
{
   for (int i = 0; i < cycles; ++ i)
   {
      try
      {
         // try to get the real object from global reference
         FileObject file = DataAccess::Access()->Get();
         for (int j = 0; j < internalCycles; ++ j)
         {
            // try to write into file
            file.Write(String().Cat() << "[" << i << "," << j << "] accesser");
         }
      }
      catch(Exc& e)
      {
         Out(String().Cat() << "[" << i << "] Accesser error: " << e);
      }
   }
}
```

In the considered implementation the File lifetime is always predictable while lifetime of FileObject can be longer.

See attached file for detailed information.

Regards,
Grigory.

File Attachments

1) TestPtrMT.zip, downloaded 315 times