81 QReadWriteLock

81.1 读写锁

利用QMutex表示的互斥体给代码区加锁，可以保证该区中的代码在任何时候只被获得锁的那个线程执行。这对于避免因多个线程同时对共享数据执行写操作而导致的并发冲突是有效的。但有时多个线程只是想同时读取共享数据，这其实并不会引发冲突。最好能在加锁时区分读写，只在必须同步时同步，降低对并发性的削弱，提高运行性能。

QReadWriteLock类所表示的读写锁，在某种程度上满足了上述要求：

- 已加读锁：不能加写锁，但可以加读锁
- 已加写锁：不能加写锁，也不能加读锁

这就叫“读共享，写独占”。

```
void QReadWriteLock::lockForRead();                // 加读锁，其它线程可以再加读锁，但不能加写锁（阻塞）
void QReadWriteLock::lockForWrite();               // 加写锁，其它线程既不能再加读锁（阻塞）也不能再加写锁（阻塞）
bool QReadWriteLock::tryLockForRead();             // lockForRead的非阻塞版本
bool QReadWriteLock::tryLockForWrite();            // lockForWrite的非阻塞版本
bool QReadWriteLock::tryLockForRead(int timeout);  // tryLockForRead的带超时版本
bool QReadWriteLock::tryLockForWrite(int timeout); // tryLockForWrite的带超时版本
void QReadWriteLock::unlock();                     // 解锁
```

81.2 案例

81.2.1 创建项目

通过QtCreator，在C:\Users\Minwei\Projects\Qt路径下，创建名为ReadWriteLock的控制台（Console）项目。

81.2.2 实现功能

C:\Users\Minwei\Projects\Qt\ReadWriteLock\main.cpp：

```
#include <iostream>
using namespace std;

#include <QCoreApplication>
#include <QThread>
#include <QMutex>
#include <QReadWriteLock>
#include <QTime>

unsigned int g_cn = 0;
QMutex g_mutex;
QReadWriteLock g_rwlock;

class WriterThread: public QThread
{
protected:
```

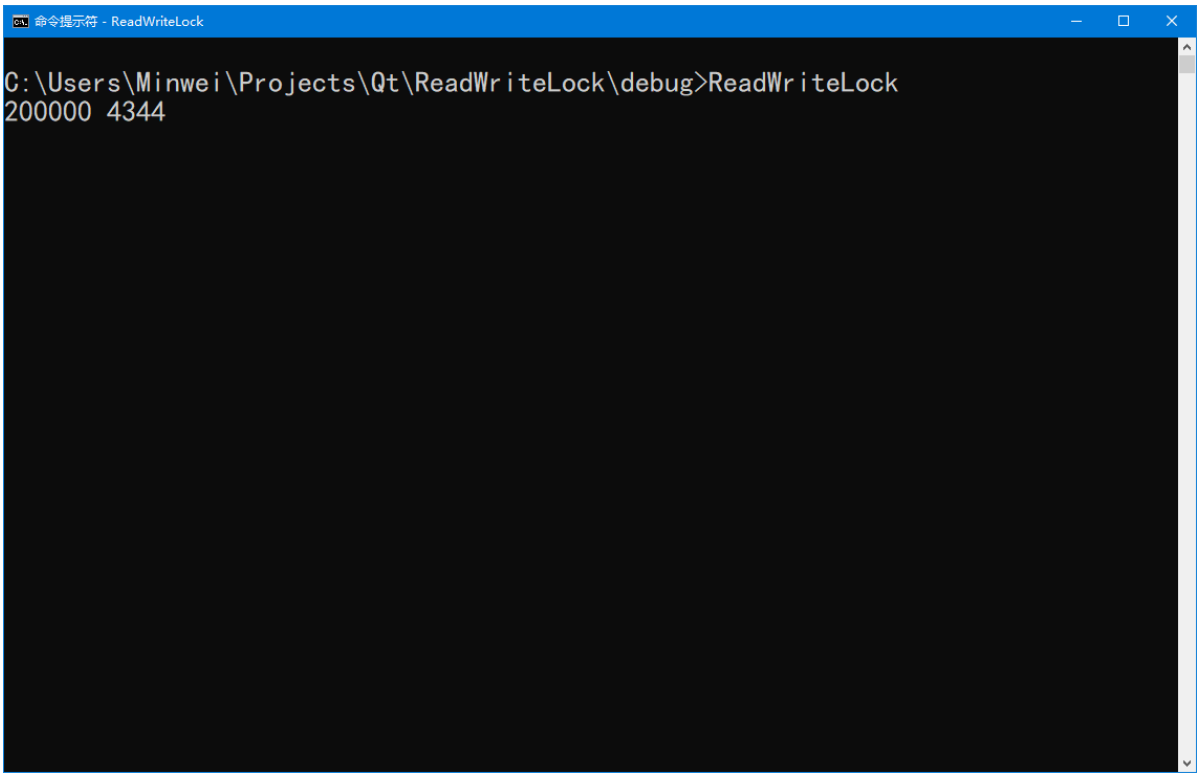
```
    /* QMutex
    void run()
    {
        for (unsigned int i = 0; i < 100000; i++) {
            g_mutex.lock();
            ++g_cn;
            g_mutex.unlock();
        }
    }
    */
    // QReadWriteLock
    void run()
    {
        for (unsigned int i = 0; i < 100000; i++) {
            g_rwlock.lockForWrite();
            ++g_cn;
            g_rwlock.unlock();
        }
    }
};

class ReaderThread: public QThread
{
protected:
    /* QMutex
    void run()
    {
        for (unsigned int i = 0; i < 100000; i++) {
            g_mutex.lock();
            unsigned int cn = g_cn;
            g_mutex.unlock();
        }
    }
    */
    // QReadWriteLock
    void run()
    {
        for (unsigned int i = 0; i < 100000; i++) {
            g_rwlock.lockForRead();
            unsigned int cn = g_cn;
            g_rwlock.unlock();
        }
    }
};

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    WriterThread writer1, writer2;
    ReaderThread reader1, reader2;

    QTime time;
    time.start();

    writer1.start();
```

```
73      writer2.start();
74      reader1.start();
75      reader2.start();
76
77      writer1.wait();
78      writer2.wait();
79      reader1.wait();
80      reader2.wait();
81
82      cout << g_cn << ' ' << time.elapsed() << endl;
83
84      return a.exec();
85  }
```

81.2.3 测试验证

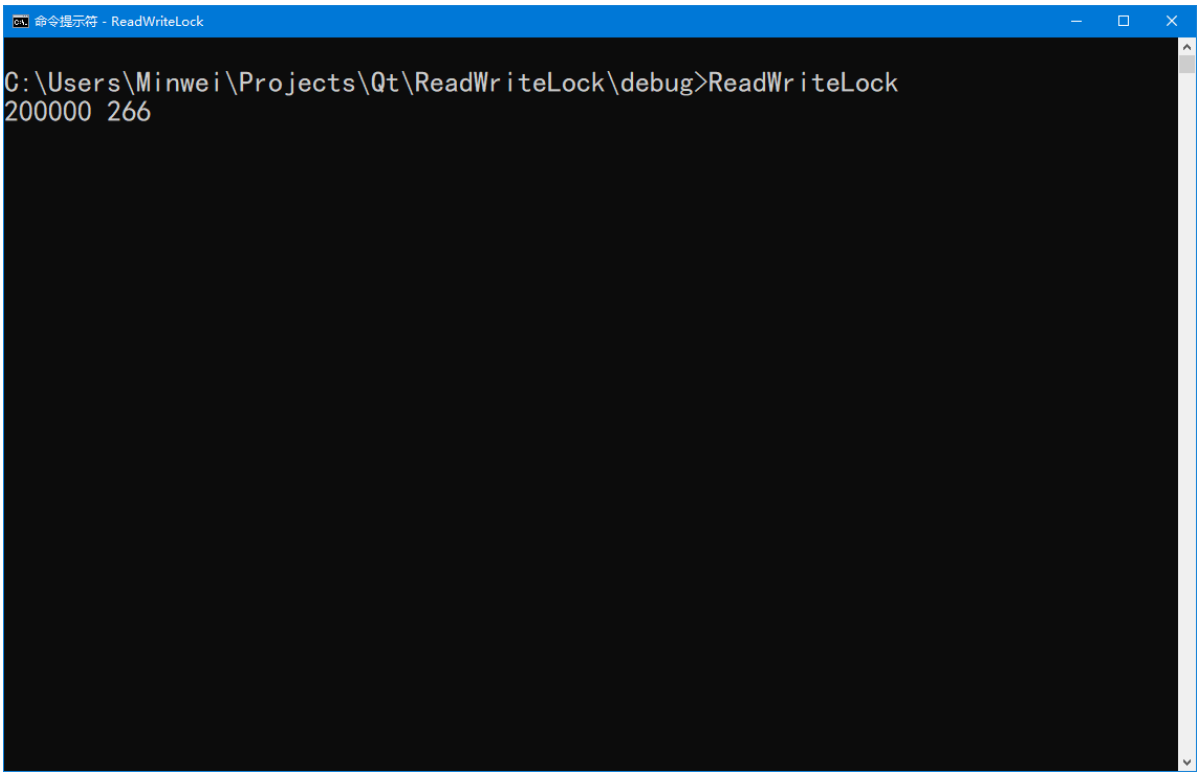
81.2.3.1 使用QMutex

运行效果如图所示：



81.2.3.2 使用QReadWriteLock

运行效果如图所示：


命令提示符 - ReadWriteLock
C:\Users\Minwei\Projects\Qt\ReadWriteLock\debug>ReadWriteLock
200000 266