ReaderWriterTest说明文档

**程序设计思路**

首先采用窗体的交互形式，来展示形象地展示多线程的运行状况。设计三个窗体，主窗体、读者窗体、写者窗体。

主窗体的任务，输入文件路径、打开读者或者写者窗体。

读者窗体，文本框不可编辑，只能读文件。

写者窗体，申请到write锁之后，才可以读到文件现有内容并且可以编辑，保存到文件。

**函数说明**

ReaderWriterMain.cs:

public static void setPath(String filePath)

设置文件路径字符串

public static String getPath()

获取文件路径字符串

MainWindow.cs:

private void confirmBtn\_Click(object sender, EventArgs e)

“确认路径”按钮单击触发函数，检查路径合法性，并打开文件

private void ReaderAddBtn\_Click(object sender, EventArgs e)

“增加读者”按钮单击触发函数，增加并启动一个读者进程

private void WriterAddBtn\_Click(object sender, EventArgs e)

“增加写者”按钮单击触发函数，增加并启动一个写者进程

ReaderThread.cs:

public static void read()

申请读者锁，申请不到，则等待，成功后打开一个读者窗体

ReaderWindow.cs:

public ReaderWindow(String path)

窗体构造函数，构造的同时，从文件中读取现有内容

private void readCloseBtn\_Click(object sender, EventArgs e)

“结束线程”按钮单击触发函数，调用窗体关闭函数

private void ReaderWindow\_FormClosing(object sender, FormClosingEventArgs e)

窗体关闭函数，释放读者锁，释放窗体资源，结束线程

WriterThread.cs:

public static void write()

申请写着锁，申请不到，则等待，成功后打开一个写者窗体

WriterWindow.cs:

public WriterWindow(String path)

窗体构造函数，构造的同时，从文件中读取现有内容

private void endThread\_Click(object sender, EventArgs e)

“结束线程”按钮单击触发函数，调用窗体关闭函数

private void WriterWindow\_FormClosing(object sender, FormClosingEventArgs e)

窗体关闭函数，释放写者锁，释放窗体资源，结束线程

private void write\_Click(object sender, EventArgs e)

“确认写入”按钮单击触发函数，将文本框内字符，写入文件

**程序使用说明**

1. 运行后，会显示一个Form1的窗体
2. 在文件路径的文本框内输入测试文件的路径，若输入”test.txt”，则默认文件在ReadWriteTest\bin\Debug下。
3. 点击“确认路径”按钮：

若输入的文件不存在则提示找不到文件，并且清空文本框。

若文件存在，正常打开则提示打开成功，并且文本框不可再编辑，确认路径按钮隐藏。

1. 点击下方的“增加读者线程”和“增加写者线程”按钮，每点击一次，如果当前能申请到read锁或者write锁，将会弹出相应的ReaderWindow和WriterWindow；若当前申请不到，则会等待申请到锁，自动弹出响应窗体。因此，如果点击增加写者锁两次，则首先会弹出一个窗体，等该窗体关闭后，会立刻弹出下一个窗体。
2. ReaderWindow 显示文件内容。
3. WriterWindow 显示文件内容，并且可以写入新内容，点击“确认写入”后会写入文件。
4. 点击“结束线程”按钮或者关闭窗体，若当前线程拥有锁，则释放相应的锁，并关闭窗体，否则直接关闭窗体。