

# 一文詳解共享內存-附帶QT共享內存Demo實現

原創 周旋 周旋機器視覺 2022-03-14 18:30

收錄於話題

#熟悉Cpp

2個

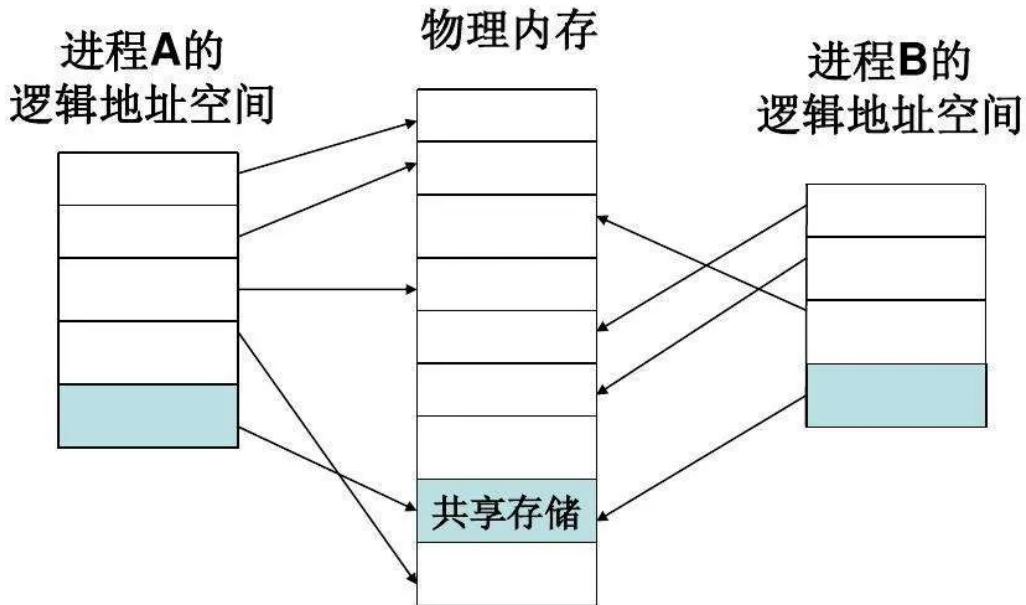
## 1、共享內存的概念

共享內存是供多個進程（可以是多CPU，也可以是多個程序之間）通信的一種方式，它的百度解釋：

共享內存 (shared memory)指在多處理器的計算機系統中，可以**被不同中央處理器（CPU）訪問的大容量內存**。

由於多個CPU需要快速訪問存儲器，這樣就要對存儲器進行緩存（Cache）。任何一個緩存的數據被更新後，由於其他處理器也可能要存取，共享內存就需要立即更新，否則不同的處理器可能用到不同的數據。共享內存是Unix下的多進程之間的通信方法,這種方法通常用於一個程序的多進程間通信，實際上多個程序間也可以通過共享內存來傳遞信息。

# 共享内存示意图

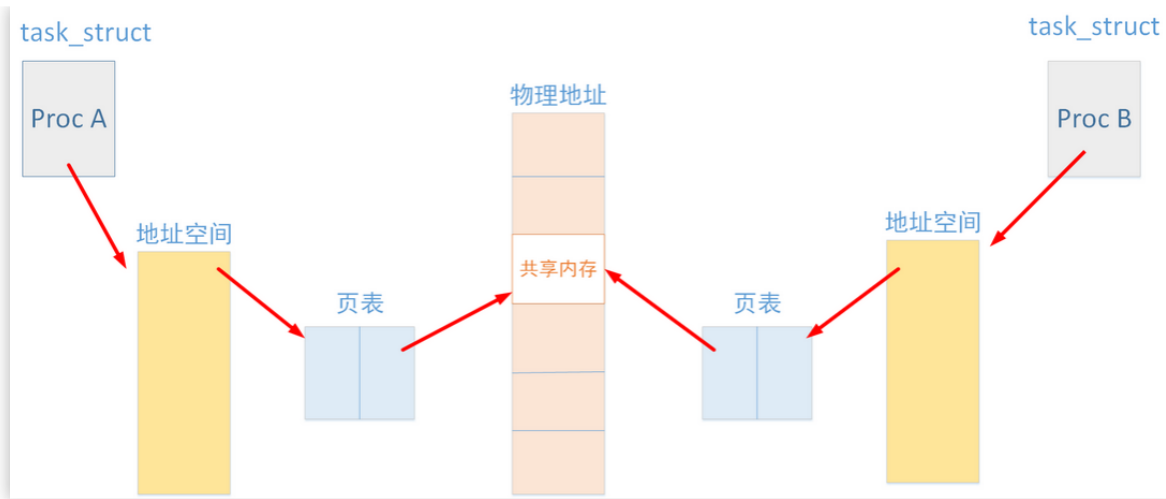


共享内存允許兩個不相關的進程訪問同一個邏輯內存，共享内存是兩個正在運行的進程之間共享和傳遞數據的一種非常有效的方式。但從上解釋可以看出，共享内存需要做到進程間的信息同步，但共享内存本身是沒有這種功能的，所以**共享内存實現進程間通信常常搭配信號量來使用**。

## 2、共享内存通信的工作原理

在Linux中，每个进程都有属于自己的进程控制块（PCB）和地址空间（Addr Space），并且都有一个与之对应的页表，负责将进程的虚拟地址与物理地址进行映射，通过内存管理单元（MMU）进行管理。**两个不同的虚拟地址通过页表映射到物理空间的同一区域，它们所指向的这块区域即共享内存。**

共享内存的通信原理示意图：



当两个进程通过页表将虚拟地址映射到物理地址时，在物理地址中有一块共同的内存区，即共享内存，这块内存可以被两个进程同时看到。这样当一个进程进行写操作，另一个进程读操作就可以实现进程间通信。但是，我们要确保一个进程在写的时候不能被读，因此我们使用信号量来实现同步与互斥操作。

### 3、共享内存的使用流程

看完上面的概念肯定是无法理解的。下面我们举一个例子的流程。

有两个进程（可以等效为两个程序），一个叫Read，一个叫Write，

Write进程想将一张图片写到一块共享内存，怎么办呢？首先系统本来是没有共享内存的，所以Write进程需要先用Key（一个标识，可以是随意长度的数字或者字母等等，如123）来Create（创建）一个共享内存。

Key是一个标识，作用是为该共享内存起一个名字，可以方便其它进程根据这个Key来访问这块共享内存。

Create函数的作用是创建一个某一大小（大小Size做为参数）的内存块，这个Size大小的内存块就是开辟出来的共享内存。

Create函数将自动把当前进程与该内存块attach（链接）在一起。这个attach的过程其实就相当于我们前面概念所说的，将进程的虚拟内存与实际逻辑内存地址相对应的过程。

这时Write进程就可以向共享内存中写图片数据了。

其它进程如何读取图片呢？当Read进程读取图片时，首先应该告诉Read共享内存的地址，也就是通过SetKey(Key)来传递标识。

现在Read光知道共享内存的名字了，但它还不知道共享内存的逻辑地址，所以需要同attach（链接）将Read与共享内存链接在一起。

这时Read就可以读取共享内存了。当读取完共享内存后，需要detach将该进程与共享内存分离。否则当其它进程想要访问这块共享内存时，就会attach失败。

## 4、QT共享内存QsharedMemory的使用

建议直接阅读QT的QsharedMemory类的帮助文档。此小节也是简要概括的帮助文档。

### Contents

[Public Types](#)


[Public Functions](#)

[Detailed Description](#)

## QSharedMemory Class

The QSharedMemory class provides access to a shared memory segment. [More...](#)

Header: `#include <QSharedMemory>`

make: `QT += core`

Since: Qt 4.4

Inherits: [QObject](#)

This class was introduced in Qt 4.4.

- [List of all members, including inherited members](#)

QSharedMemory提供了多个线程和进程对共享内存段的访问。它还为单个线程或进程提供了一种锁定内存以进行独占访问的方法，也就是通过信号量对共享内存实现了同步操作。

QsharedMemory类有如下几个常用函数：

```
void QSharedMemory::setKey(const QString &key)
```

为这个共享内存对象设置键值key。如果key与当前的key相同，函数将不做任何操作返回。

如果当前的共享内存对象已经链接到底层共享内存段（isAttached），它将在设置新键之前与它分离（detach）。这个函数不执行attach链接操作。

```
bool QSharedMemory::create(int size, QSharedMemory::AccessMode mode = ReadWrite)
```

该函数根据共享内存的Key值来创建一个size大小的共享内存段，可以根据mode设为ReadWrite可读可写或者ReadOnly只读模式。

```
bool QSharedMemory::attach(QSharedMemory::AccessMode mode = ReadWrite)
```

该函数将会依据共享内存对象的Key值，来将共享内存对象与实际的共享内存逻辑地址相链接，这样进程就可以通过QSharedMemory对象来访问实际的共享内存了。

```
bool QSharedMemory::detach()
```

将进程与共享内存段分离。如果这是链接到共享内存段的最后一个进程，那么这个共享内存段将被系统释放，也就是说，共享内存中的内容将被销毁。

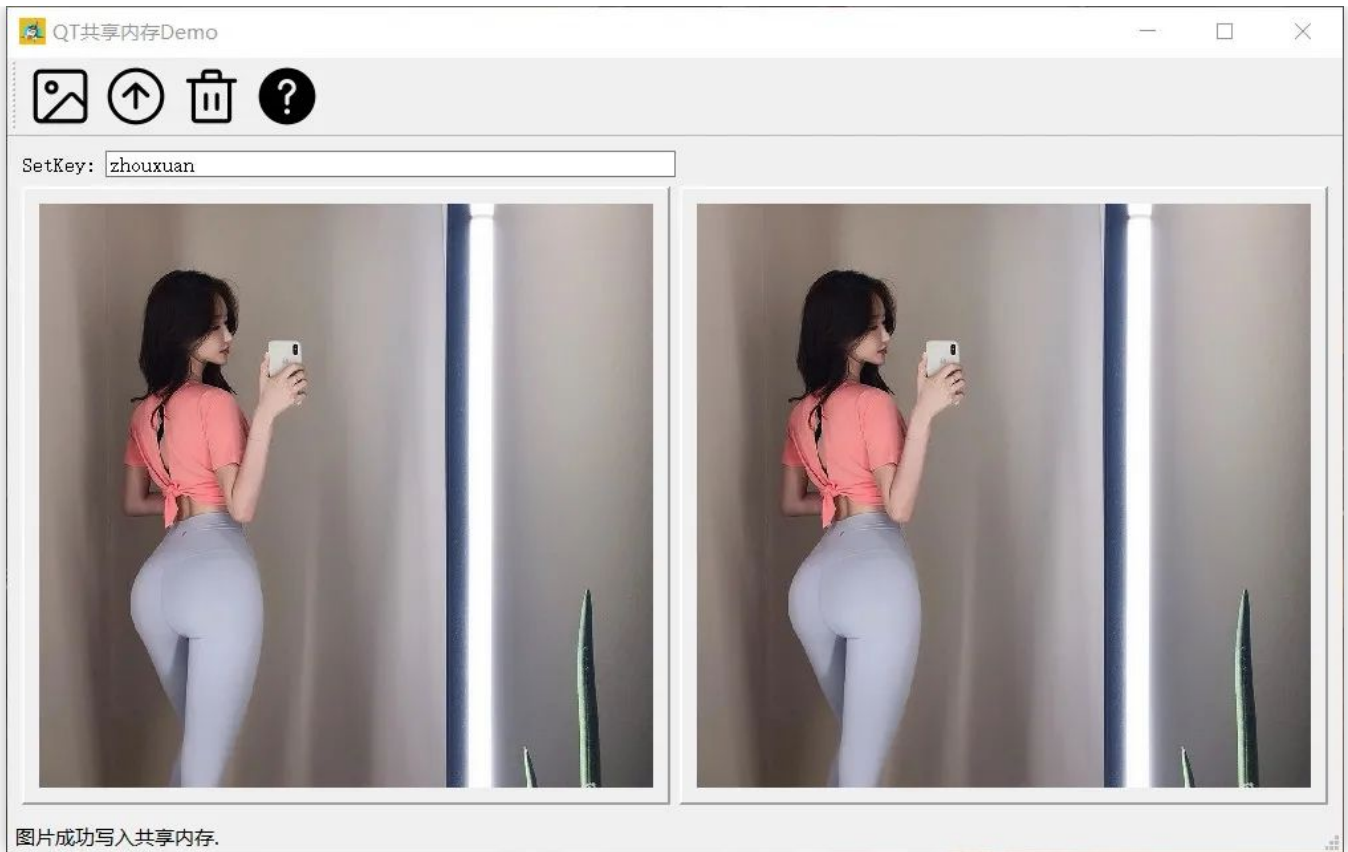
```
bool QSharedMemory::lock()
```

这是一个可以锁住共享内存段以供该进程访问的信号量。当进程对共享内存进程操作时，为防止其它进程也对该内存进行改动从而操作数据不同步的情况，需要将内存进行锁定，也就是lock。

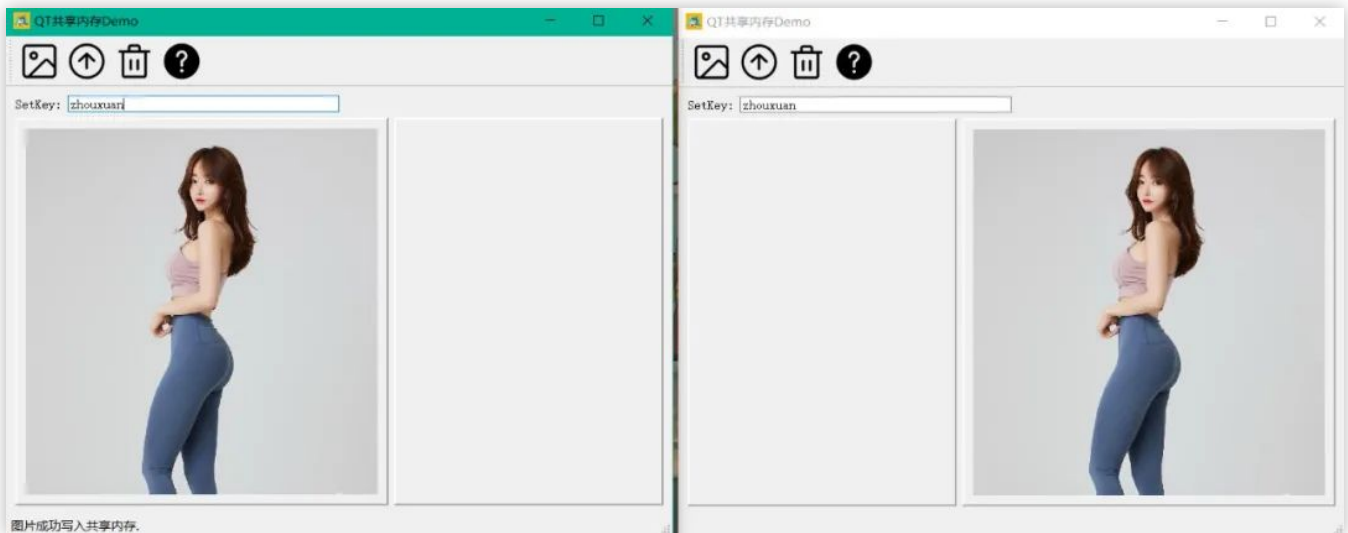
```
bool QSharedMemory::unlock()
```

当对共享内存操作完成后，需要释放共享内存段上的锁。否则其它进程无法对共享内存进行任何操作。

## 5、实用Demo演示



上图是我自己实现的一个基于QT共享内存实现图片读写的小Demo，左边为写，右边为读。可以在一个进程内通信，也可以再开一个进程两者通过共享内存通信。如下。



程序源码公众号【周旋机器视觉】回复【QT1】即可下载。

以下两段程序是网上最为常见，读和写，也贴在这里叭。

QT写数据进程：

```
1 #include "widget.h"
2 #include "ui_widget.h"
3 #include <QDebug>
```

```
4  #include <QtGlobal>
5
6  Widget::Widget(QWidget *parent)
7      : QWidget(parent)
8      , ui(new Ui::Widget)
9  {
10     ui->setupUi(this);
11
12     this->setWindowTitle("Write");
13     sharememory = new QSharedMemory(this);
14 }
15
16 Widget::~Widget()
17 {
18     delete ui;
19 }
20
21
22 void Widget::on_btnWrite_clicked()
23 {
24     // 设置访问标识
25     sharememory->setKey(ui->lineEditKey->text());
26
27     if(sharememory->isAttached()){
28         sharememory->detach();
29     }
30
31     if(!sharememory->create(100, QSharedMemory::ReadWrite)){
32         qDebug() << "创建共享内存失败：" << sharememory->errorString();
33         return;
34     }
35
36     sharememory->lock();
37
38     char* sm = static_cast<char*>(sharememory->data());
39     QByteArray ba = ui->lineEditValue->text().toUtf8();
40     memcpy(sm, ba.data(), static_cast<size_t>(qMin(sharememory->size(), ui->li
41
42     sharememory->unlock();
43 }
```



## QT读数据进程：

```
1  #include "widget.h"
2  #include "ui_widget.h"
3  #include <QDebug>
4  #include <QtGlobal>
5
6  Widget::Widget(QWidget *parent)
7      : QWidget(parent)
8      , ui(new Ui::Widget)
9  {
10     ui->setupUi(this);
11     sharememory = new QSharedMemory(this);
12     setWindowTitle("Read");
13 }
14
15 Widget::~Widget()
16 {
17     delete ui;
18 }
19
20
21 void Widget::on_btnRead_clicked()
22 {
23     //设置访问标识
24     sharememory->setKey(ui->lineEditKey->text());
25
26     if(!sharememory->attach()){
27         qDebug() << "attach failed.";
28     }
29
30     sharememory->lock();
31
32     char* sm = static_cast<char*>(sharememory->data());
33     char* out = new char[static_cast<unsigned int>(sharememory->size())];
34     memcpy(out, sm, static_cast<size_t>(sharememory->size()));
35     ui->lineEditValue->setText(QString(out));
36 }
```



```
37     sharememory->unlock();  
38  
39     sharememory->detach();  
40  
41 }
```

## THE END

點擊下方“[閱讀原文](#)”可以查看我的個人博客。想加交流群的小伙伴公眾號後台回復【加群】。

今天就到這兒啦。



周旋機器視覺

機器視覺研發工程師，算法與軟件開發

64篇原創內容

---

公眾號

收錄於話題#熟悉Cpp 2

下一篇 · [【秋招】1：機械轉碼（如何從一個坑，跳到另一個）](#)

[閱讀原文](#)

喜歡此內容的人還喜歡

一個神奇的開源項目：讓照片快速3D 化！

小白學視覺

---

Appium自動化測試之安卓模擬器安裝及配置

性能測試之道

---

Python計算渦度、散度、渦度平流和溫度平流(附1980~2020年中國區域30米和1000米土地利用數據)

氣象水文科研貓

