湖南工业大学

课程设计任务书

**2015 —2016 学年第 1 学期**

计算机与通信 学院（系、部） 计算机科学与技术 专业 1403 班级

课程名称： 面向对象程序设计

设计题目： 基于mplayer开发的视频播放器

完成期限：自 2015 年 12 月 28 日至 2016 年 1 月 3 日共 1 周

|  |  |  |
| --- | --- | --- |
| 内  容  及  任  务 | 一、设计的主要技术参数  1.面向对象程序语言的构造函数与析构函数  2.窗口类及各种窗口类子类的使用  3.利用图片类来更换按钮  4.按钮样式的设置  5.文件的保存与打开  二、设计任务  1.能流畅的播放视频  2.播放视频时进度条能自动刷新  3.截图完成后能自己选择图片保存格式  三、设计工作量  1. UI界面的设计  2. 程序部分的设计  3. 系统的调试 | |
| 进  度  安  排 | 起止日期 | 工作内容 |
| 2015.12-28-2015.12.29 | 系统框架的大致搭建 |
| 2015.12.29-2015.12.31 | 能进行视频的播放并且能实现播放暂停等功能 |
| 2015.12.31-2016.1.3 | 细节的修改与程序的调试 |
| 主  要  参  考  资  料 | 1.Qt高级编程 电子工业出版社 2011.04  2.CSDN论坛Qt社区 <http://bbs.csdn.net/forums/Qt>  3.谭浩强.C++面向对象程序设计[M].北京：清华大学出版社，2006. | |

指导教师（签字）： 年 月 日

系（教研室）主任（签字）： 年 月 日

面向对象程序设计课程设计

设计说明书

|  |
| --- |
| 基于mplayer开发的视频播放器 |

起止日期： 2015 年 12 月 28 日 至 2016 年 1 月 3 日

|  |  |
| --- | --- |
| 学生姓名 | 谭鹏 |
| 班级 | 计算机1403 |
| 学号 | 14408100322 |
| 成绩 |  |
| 指导教师(签字) |  |

**计算机与通信学院**

2016 年 1 月 3 日

**目录**

**一、课题的介绍和课题的任务**

**----------------2**

**二、设计的要求**

**----------------2**

**三、系统的分析和系统中类的设计**

**----------------2**

**四、系统的数据设计**

**----------------2**

**五、系统的实现及调试**

**----------------2**

**六、系统的使用说明**

**----------------16**

**七、总结**

**----------------16**

**视频播放器**

**一、课题的介绍和课题的任务**

**1.课题的介绍**

**本次课程设计我选的题目是视频播放器。目前市场上各种应用商店上有各种播放器，而且功能都很强大。一直对里面的具体实现有兴趣，所以这次就尝试着写了个视频播放器。并且加进去了一些很实用的功能，比如截屏和录屏。**

**2.课题的任务**

**包括视频的播放暂停、音量的加减、播放速度的设置、以及视频文件的添加删除、屏幕的截屏以及录屏。画面不卡顿。播放时进度条能刷新。**

**二、设计的要求**

1. **能进行视频的播放**
2. **能暂停正在播放的视频**
3. **能播放正在暂停的视频**
4. **能调节视频播放的视频**
5. **能进行视频播放进度的调节**
6. **能停止正在播放的视频**
7. **能添加新的视频文件到播放列表**
8. **能删除播放列表中选中的视频**
9. **能改变视频播放的速度**
10. **能对正在播放的视频进行截屏**
11. **能录制正在播放的视频**

**三、系统的分析和系统中类的设计**

**1.窗口类QMainWidndow**

**2.按钮类QPushButton**

**3.标签类QLabel**

**4.列表类QList**

**5.滑动条类QSilider**

**6.显示类QWidget**

**四、系统的数据设计**

**无**

**五、系统的实现及调试**

**5.1系统模块框架图**

视频播放器

截屏

与

录屏

视频播放速度调节

播放列表的

增删

音量的

调节

视频进度条的调节

快进与

回退

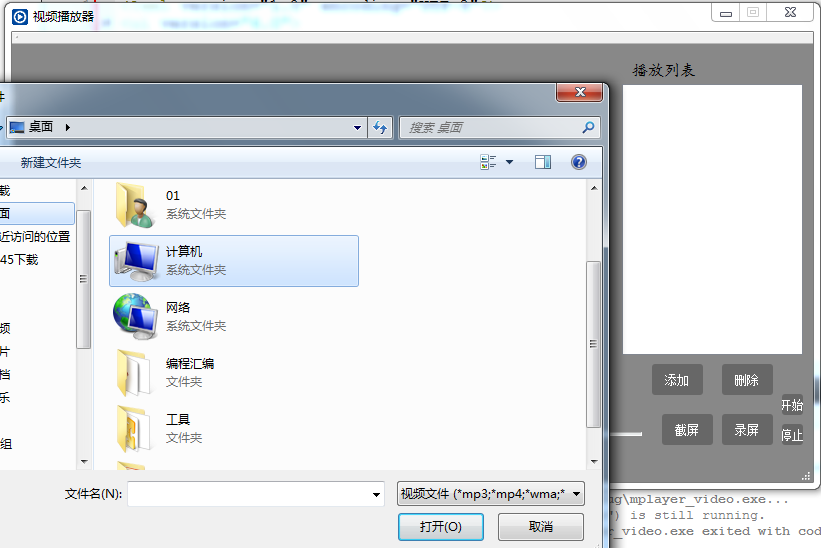
播放

与

暂停

**5.2 程序的具体实现：**

1. **添加文件进视频播放列表**



**图1 播放文件的添加**

**代码实现：**

void mplayer\_video::on\_open\_clicked()

{

QStringList filenames = QFileDialog::getOpenFileNames(this,tr("选择文件"),"/",tr("视频 文件(\*mp3 \*mp4 \*wma \*3gp \*wav \*avi \*flv \*rmvb \*mkv \*ts \*wmv)")); // \*号和前面的要隔开至少一个空格，不能连起来。

if(filenames.count()!=0)

ui->list->addItems(filenames);

}

**2.双击列表选项进行播放**

**图2 双击进行视频的播放**

**代码实现 ：**

void mplayer\_video::on\_list\_itemDoubleClicked(QListWidgetItem \*item)

{

speed=1;

ui->lupin\_lable->*setVisible*(false);

play(item->text());

}

void mplayer\_video::play(QString fileName)

{

QStringList arg1;

process->kill();

process=new QProcess(this);

//connect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

arg1 << fileName;

arg1 << "-slave";//默认情况下，mplayer接受键盘的命令，而"-slave"使其不再接受键盘事件，而是作为后台程序运行，

//接受以“\n”结束的命令控制，这样我们可以在进程中给他发送命令，而不需要操作键盘了.

arg1 << "-quiet"; //尽可能的不打印播放信息

arg1 << "-zoom"; //视频居中，四周黑条，全屏播放

arg1 << "-wid" << QString::number((unsigned int)(ui->widget->winId()));

// "-wid <窗口标识>" 是指让MPlayer依附于那个窗口，

//ui->widget->winId() 这个指令就是用来获取widget的标识码 ,

//这样视频播放的时候，就在这个部件里播放，相当于给他固定起来。

connect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

process->start(arg,arg1); //开始播放进程

ui->zanting->setText("暂停");

}

void mplayer\_video::dataRecieve()

{

process->write("get\_time\_length\n");

process->write("get\_time\_pos\n");

process->write("get\_percent\_pos\n");

while(process->*canReadLine*())

{

QByteArray b=process->readLine();

if(b.startsWith("ANS\_TIME\_POSITION"))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

currentStr=s.mid(18);

ui->horizontalSlider->setValue(s.mid(18).toFloat()); //更新进度条

}

else if((b.startsWith("ANS\_LENGTH")))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

totalTime=s.mid(11);

ui->label\_time->setText(currentStr+"秒/"+totalTime+"秒 "); //显示时间进度

ui->horizontalSlider->setRange(0,s.mid(11).toFloat());

}

else if((b.startsWith("ANS\_PERCENT\_POSITION")))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

currentPercent=s.mid(21);

ui->percentLabel->setText(currentPercent+"%");

}

}

}

 **3.播放与暂停的实现**

**图3 播放与暂停**

**代码实现：**

void mplayer\_video::on\_zanting\_clicked()

{

ui->lupin\_lable->*setVisible*(false);

process->write("pause\n");

//ui->list->currentRow()!=-1 判断是否有选中某条数据

//play(ui->list->item(ui->list->currentRow())->text()); 播放选中的某条数据

if(ui->zanting->text()=="播放")

{

if(ui->list->count()==0)

QMessageBox::warning(this,"提示","播放列表为空",QMessageBox::Yes);

else{

connect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

process->write("get\_time\_length\n");

process->write("get\_time\_pos\n");

process->write("get\_percent\_pos\n");

ui->zanting->setText("暂停");

}

}

else

{

disconnect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

ui->zanting->setText("播放");

}

}

**4.快进与回退**

**代码实现：**

void mplayer\_video::on\_back\_clicked()

{

process->write("seek -10\n");

}

void mplayer\_video::on\_quick\_clicked()

{

process->write("seek +10\n");

}

**5.加速与减速**

**代码实现：**

void mplayer\_video::on\_back\_clicked()

{

process->write("seek -10\n");

}

void mplayer\_video::on\_quick\_clicked()

{

process->write("seek +10\n");

}

**6.停止播放**

**代码实现：**

void mplayer\_video::on\_stop\_clicked()

{

process->write("quit\n");

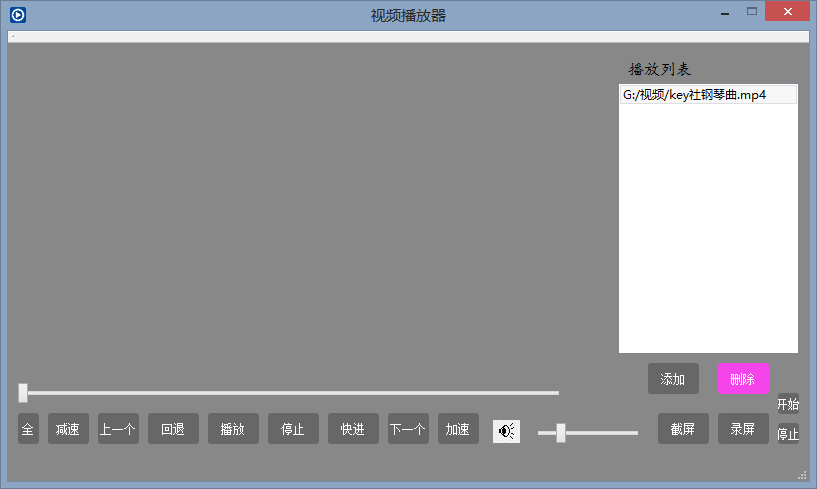
ui->horizontalSlider->setSliderPosition(0);

ui->label\_time->clear();

ui->percentLabel->clear();

ui->zanting->setText("播放");

}

**7.从播放列表中删除中选中的视频**

**图4 删除播放列表中的视频**

**代码实现：**

void mplayer\_video::on\_del\_clicked()

{

if(ui->list->currentRow()==-1)

QMessageBox::warning(this,"提示","未选中项目或列表为空",QMessageBox::Yes);

else

{

ui->list->takeItem(ui->list->currentRow());

process->*close*();

ui->horizontalSlider->setSliderPosition(0); //更新进度条

ui->label\_time->clear();

ui->percentLabel->clear();

}

}

**8.加速播放与减速播放：**

**图****5 加速播放与减速播放**

**定义一个全局静态浮点型变量 speed并初始化为1，每次点击加速，speed的值乘以二，每次点击减速，speed的值除以2**

**代码实现：**

void mplayer\_video::on\_jiansu\_clicked()

{

speed=speed/2;

process->write(QString("speed\_set "+QString::number(speed)+" 2\n").toUtf8());

}

void mplayer\_video::on\_jiasu\_clicked()

{

speed=speed\*2;

process->write(QString("speed\_set "+QString::number(speed)+" 2\n").toUtf8());

}

**9.视频播放进度条的刷新**

**这是本程序里最复杂的部分之一。**

**主要实现是用connect函数链接进程与DataRecieve函数**

**主要代码：**

void mplayer\_video::dataRecieve()

{

process->write("get\_time\_length\n");

process->write("get\_time\_pos\n");

process->write("get\_percent\_pos\n");

while(process->*canReadLine*())

{

QByteArray b=process->readLine();

if(b.startsWith("ANS\_TIME\_POSITION"))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

currentStr=s.mid(18);

ui->horizontalSlider->setValue(s.mid(18).toFloat()); //更新进度条

}

else if((b.startsWith("ANS\_LENGTH")))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

totalTime=s.mid(11);

ui->label\_time->setText(currentStr+"秒/"+totalTime+"秒 "); //显示时间进度

ui->horizontalSlider->setRange(0,s.mid(11).toFloat());

}

else if((b.startsWith("ANS\_PERCENT\_POSITION")))

{

b.replace(QByteArray("\n"),QByteArray(""));

QString s(b);

currentPercent=s.mid(21);

ui->percentLabel->setText(currentPercent+"%");

}

}

}

**10.声音滑动条对声音的调节**

**图6 音量的调节**

**代码实现：**

void mplayer\_video::on\_voice\_slider\_sliderMoved(int v)

{

if(ui->zanting->text()=="播放")

{

process->write(QString("volume "+QString::number(v)+" 2\n").toUtf8());

process->write("pause\n");

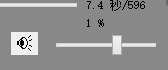
}

else{

process->write(QString("volume "+QString::number(v)+" 2\n").toUtf8());

}

}

**11.设置图片按钮**

**图7 设置图片为按钮**

**代码实现：**

QPixmap pixmap;

pixmap.load("G:/mplayer\_video/voice.jpg");

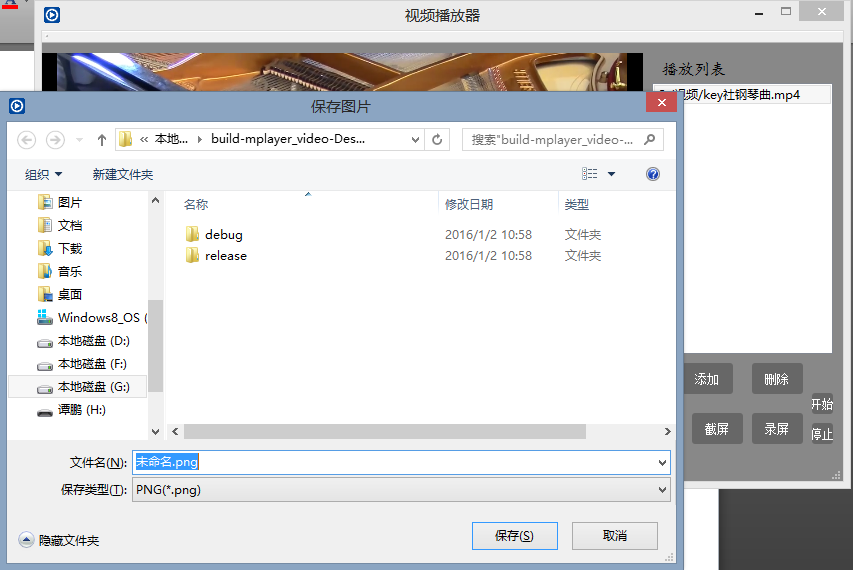
ui->voice->setFixedSize(pixmap.width(),pixmap.height());

ui->voice->setIcon(pixmap);

ui->voice->setIconSize(QSize(pixmap.width(),pixmap.height()));

ui->voice->setToolTip("音量");

ui->voice->show();

1. **屏幕截图**

**图8 截图保存**

**代码实现：**

void mplayer\_video::on\_jietu\_clicked()

{

cout<<"ok"<<endl;

//QString slcStr=".jpg";

//QString filename = QFileDialog::getSaveFileName(this,tr("保存图片"),QDir::currentPath(),tr("PNG(\*.png);;JPG(\*.jpg);;BMP(\*.bmp)"),&slcStr);

// snapImage.save(filename);//snapName是文件名字

process->write("pause\n");

if(ui->zanting->text()=="播放")

{

cout<<"1"<<endl;

QPixmap snapImage = QPixmap::grabWindow(QApplication::desktop()->winId());

//QString filename = QFileDialog::getSaveFileName(this, tr("Save File"), "./未命名.jpg", "Image Files(\*.jpg \*.png)");

//snapImage.save(filename);

QString filename;

QString slcStr;

filename = QFileDialog::getSaveFileName(this,tr("保存图片"),"./未命名",tr("PNG(\*.png);;JPG(\*.jpg);;BMP(\*.bmp)"),&slcStr);//弹出保存图片的文件窗口

if(slcStr.left(3)=="JPG")

{

if(filename.right(3)!="jpg")

{

filename+=".jpg";

}

}

if(slcStr.left(3)=="PNG")

{

if(filename.right(3)!="png")

{

filename+=".png";

}

}

if(slcStr.left(3)=="BMP")

{

if(filename.right(3)!="bmp")

{

filename+=".bmp";

}

}

if(filename!=NULL)

{

snapImage.save(filename);

}

}

else

{

cout<<"2"<<endl;

disconnect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

ui->zanting->setText("播放");

QPixmap snapImage = QPixmap::grabWindow(QApplication::desktop()->winId());

//QString filename = QFileDialog::getSaveFileName(this, tr("Save File"), "./未命名.jpg", "Image Files(\*.jpg \*.png)");

//snapImage.save(filename);

QString filename;

QString slcStr;

filename = QFileDialog::getSaveFileName(this,tr("保存图片"),"./未命名",tr("PNG(\*.png);;JPG(\*.jpg);;BMP(\*.bmp)"),&slcStr);//弹出保存图片的文件窗口

if(slcStr.left(3)=="JPG")

{

if(filename.right(3)!="jpg")

{

filename+=".jpg";

}

}

if(slcStr.left(3)=="PNG")

{

if(filename.right(3)!="png")

{

filename+=".png";

}

}

if(slcStr.left(3)=="BMP")

{

if(filename.right(3)!="bmp")

{

filename+=".bmp";

}

}

if(filename!=NULL)

{

snapImage.save(filename);

}

connect(process,SIGNAL(readyReadStandardOutput()),this,SLOT(dataRecieve()));

process->write("get\_time\_length\n");

process->write("get\_time\_pos\n");

process->write("get\_percent\_pos\n");

ui->zanting->setText("暂停");

}

}

1. **屏幕录制**

**代码实现：**

void mplayer\_video::on\_lupin\_clicked()

{

if(ui->lupin->text()=="录屏")

{

photoStart();

timer->start(50);

ui->lupin->setText("停止");

}

else

{

timer->stop();

ui->lupin->setText("录屏");

}

}

void mplayer\_video::photoStart()

{

QDir \*d = new QDir();

QString dirName("D:\\video");

if(!d->exists(dirName))

{

d->mkdir(dirName);

}

}

void mplayer\_video::timerUpdate()

{

QPixmap p;

QString dirNamee("D:\\video\\");

dirNamee=dirNamee+QString::number(number)+".jpg";

p = QPixmap::grabWindow(QApplication::desktop()->winId());

p.save(dirNamee);

number+=1;

}

1. **屏幕录屏的播放**

**代码实现：**

void mplayer\_video::on\_lupinstart\_clicked()

{

process->write("quit\n");

QString dirName("D:\\video");

qdir = new QDir(dirName);

//qdir->setSorting(QDir::Name|QDir::Reversed);

qdir->setSorting(QDir::Reversed|QDir::Time);

list = qdir->entryInfoList();

number2=0;

timer2->start(100);

}

void mplayer\_video::on\_stop\_bofang\_clicked()

{

timer2->stop();

ui->lupin\_lable->setVisible(false);

}

**六、系统的使用说明**

**打开程序，点击添加文件进行视频播放列表视频的添加。双击列表中某条记录进行视频的播放，播放按钮自动转换成暂停按钮。点击播放将会播放视频，点击暂停视频会暂停。除了一些视频播放器的基本功能外，本程序还加入了屏幕截图与屏幕录制的功能。具体操作用按钮可以实现。**

**七、总结**

**这次C++课程设计我选的课题是视频播放器。程序的编写过程中遇到了各种困难，但最后还是实现了绝大部分的功能，这让我觉得很开心而且收获很多进步很多。**

**最开始接触C++课设的时候，我决定用Qt来写，但之前从来没用过Qt，因为自己的方向在Java和Android上所以用Eclipse和MyEclipse用的比较多，Qt只是听过但从来没用过。所以这次课设对于编译器的学习使用完全是从零开始。各种看书上网搜资料，终于熟悉了编译器的使用。在程序的编写过程中也遇到了各种问题，比如播放视频时进度条怎么都刷新不了，这个问题我找了好多资料，最后终于解决了。感觉很有成就感。**

**这个课设我从很久以前就开始写并且在课设正式开始的时候基本上已经搞定了，给老师看，老师一直让我加进去新的功能。这看上去是在为难我其实我知道是为了我好，最后还是把老师说的截屏和录屏给加进去了，这让我觉得收获很多，很感谢我的C++老师。虽然最后那个关于网页文件流抓取的问题并没有解决，但老师还是给了我优秀，我觉得很意外。接下来的时间里我会好好努力，并且争取早日解决C++老师留给我的这个难题。**

**最后，我相信我会越变越优秀。**

**附1：完成该系统的所有成员名单及其工作**

谭鹏；负责整个程序的设计与实现。