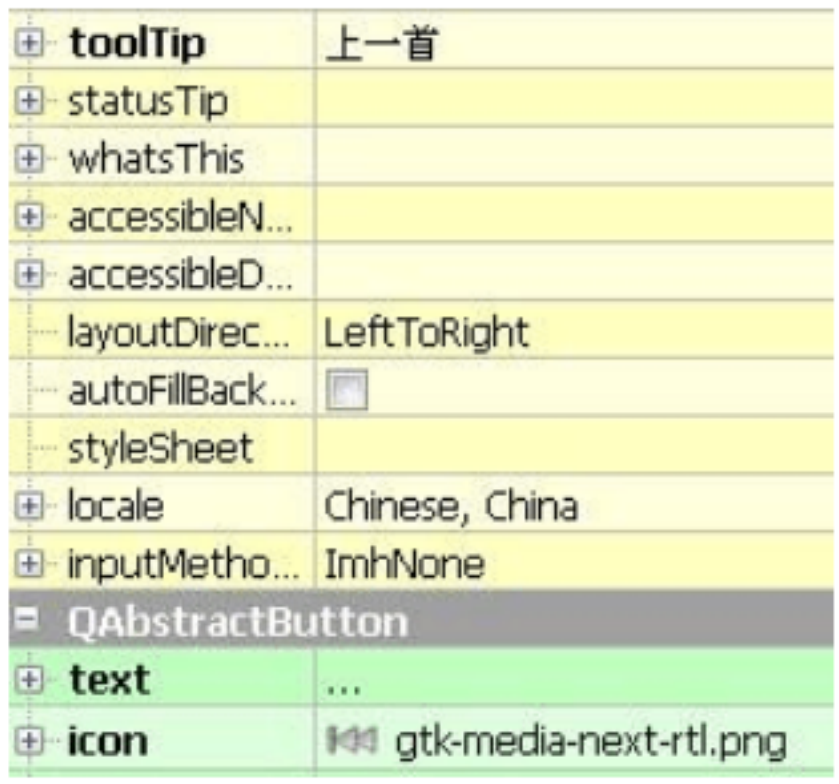
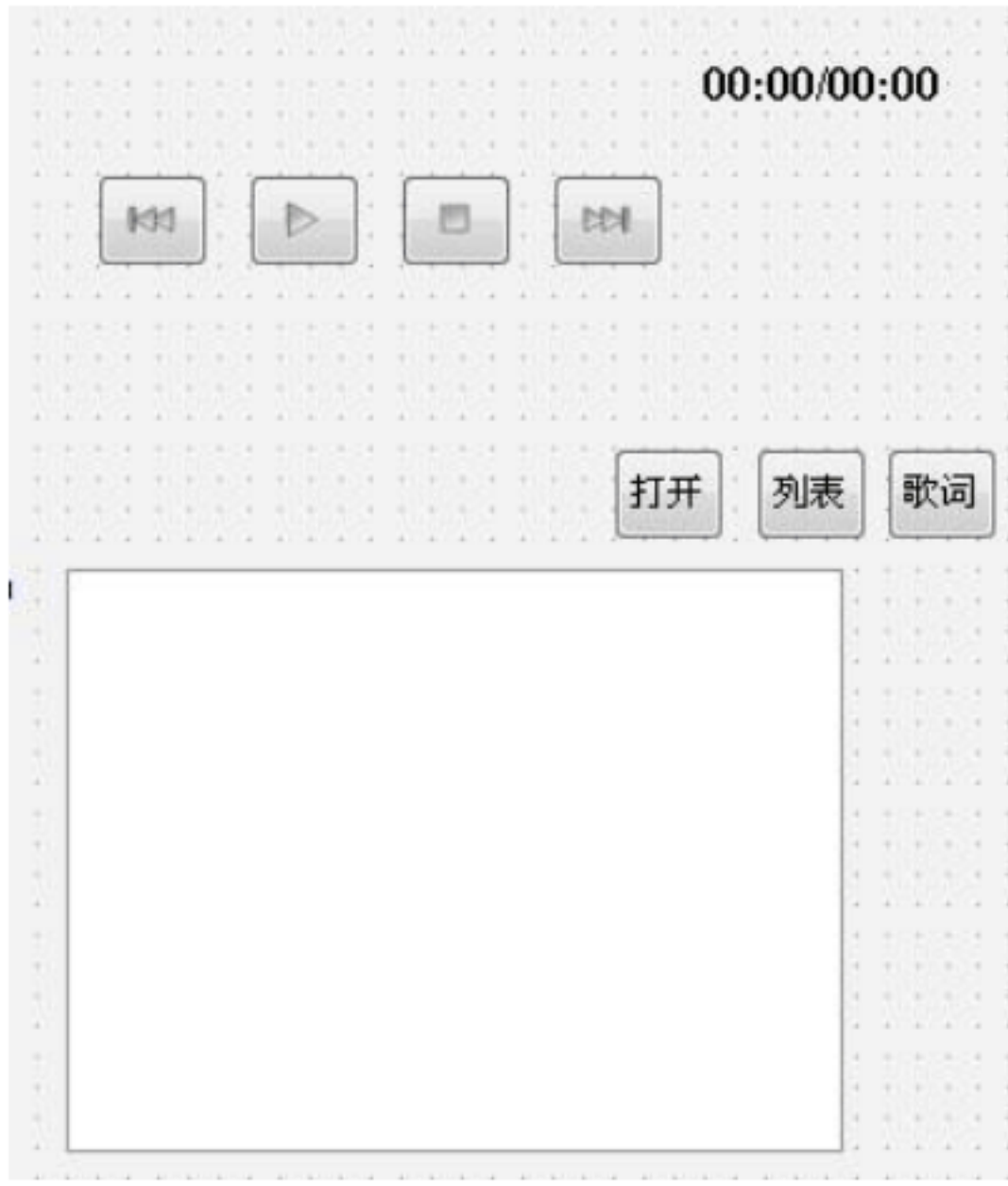
1. 新建工程，基类选择 Qwidget。双击打开界面文件，在界面文件中创建 label 显示时间、若干个 toolbutton 按钮和一个 listWidget 列表显示歌曲列表。点击“文件”->“新建”创建 QT资源文件，在工程目录下建文件夹“ images”， 双击 QT资源文件添加前缀“ / ”，再添加图标文件。

2. 单击上一曲按钮，在属性栏设置标题 tooltip 和图标 icon ：



界面如下：



3.右键几个按钮，转入槽函数，列表和歌词的信号选择 clicked （bool）。

4.工程文件中添加语句

QT += phonon

对音乐播放器进行设计，主要用到模块中的 MediaObject （管理媒体源） AudioOutput （连接物理设备） SeekSlider （ 实现进度条） QList （实现播放

列表 ）

大概地说， Phonon 的工作机制是使用 MediaObject 来管理 MediaSource 即源文件，通过 Path 连接到 AudioOutput ，最后是由 AudioOutput 将数据发送到相关物理设备。

5.修改 widget.h 头文件

在 widget.h 头文件添加语句：

```
#include <phonon>
```

在头文件声明变量：

```
private :
    Phonon:: MediaObject *audio; // 管理媒体源
    Phonon:: MediaObject *musicInformationMediaObject ; //
    Phonon:: VideoWidget *videoWidget ; //
    Phonon:: AudioOutput *audioOutput ; // 连接物理设备
    Phonon:: SeekSlider *seekSlider ; // 实现进度条
    Phonon:: VolumeSlider *volumeSlider ; // 音量调节
    QList <Phonon:: MediaSource> sourceList ; // 播放列表
    QTimer *timer ;
    QIcon *iconplay ;
    QIcon *iconpause;
    QAction *play;
    QAction *stop;
    QAction *open;
    QAction *sound;
    QAction *exit ;
    QAction *remove;
```

头文件声明函数：

```
public :
        void creatActions();      //创建动作
```

6.修改 widget.cpp 文件

构造函数添加：

```
this ->setWindowTitle(tr("音乐播放器")); // 设置标题
    /** 初始化媒体 **/
    audio =new Phonon:: MediaObject (); // 媒体对象
    audio ->setTickInterval(1);
    audioOutput = new Phonon:: AudioOutput ( Phonon:: VideoCategory ); // 音频输出
    Phonon:createPath( audio , audioOutput ); // 连接媒体对象与音频输出
    musicInformationMediaObject = new Phonon:: MediaObject ( this ); // 音乐信息对象
    volumeSlider = new Phonon:: VolumeSlider ( audioOutput , this ); // 音量滑动条
```

```
        volumeSlider  ->move(190, 100);
        volumeSlider  ->resize(   50, 20);

volumeSlider  ->setStyleSheet(    "background-color:rgb(255,255,255,100)"        );
        volumeSlider  ->setFixedWidth(   100); // 固定音量条大小
        seekSlider    = new Phonon:  SeekSlider  ( audio , this  );   // 进度滑动条
        seekSlider  ->move(10, 35);
        seekSlider  ->resize(   170, 20);

seekSlider  ->setStyleSheet(    "background-color:rgb(255,255,255,100)"        );
                creatActions();


        函数定义：
// 播放/暂停
void   Widget::on_toolButton_playpause_clicked()
{
        if  ( sourceList  .isEmpty())
                {
                        //QMessageBox::information(this,          tr("no     music  files"),
tr("no     files     to   play"));
                        return    ;
                        }
                        audio ->setQueue( sourceList   ); // 列表循环
        if  ( audio ->state()      == Phonon:  PlayingState   )
                        audio ->pause();
        else
        {
                        audio ->play();
        }
}
// 停止播放
void   Widget::on_toolButton_stop_clicked()
{
        audio ->stop();
}

void   Widget::on_toolButton_open_clicked()
{
        QStringList     files     = QFileDialog  ::getOpenFileNames(   this  ,   tr(  "Selec
Files    to   play"  ));
        //   使用  QFileDialog    的  getOpenFileNames 方法获取若干个音乐文件,
        QString   file;
```

```
        foreach (file, files) // 使用 Qt 中的 foreach 遍历每个选中的文件, 将
其添加到播放列表中。
        {
            ui -> listWidget ->addItem(file);
            sourceList .append(file);
        }
}

void Widget::creatActions()
{
        QIcon iconremove( ":/images/remove.png" );
        QIcon iconstop( ":/images/gtk-media-stop.png" );
        QIcon iconopen( ":/images/gtk-open.png" );
        QIcon iconsound( ":/images/sound.png" );
        QIcon iconexit( ":/images/exit.png" );
        iconpause = new QIcon( ":/images/gtk-media-pause.png" );
        iconplay = new QIcon( ":/images/gtk-media-play-ltr.png" );
        remove = new QAction (iconremove,tr( "清空播放列表 "), this );
        connect( remove, SIGNAL(triggered()), this , SLOT(removeSlot()));
        play = new QAction (* iconplay ,tr( "播放"), this );

connect( play , SIGNAL(triggered()), this , SLOT(on_toolButton_playpause_cl
icked()));
        stop = new QAction (iconstop,tr( "停止"), this );
        connect( stop , SIGNAL(triggered()), audio , SLOT(stop ()));
        open = new QAction (iconopen,tr( "打开文件"), this );

connect( open, SIGNAL(triggered()), this , SLOT(on_toolButton_open_clicked
()));
        sound = new QAction (iconsound,tr( "静音"), this );
        sound->setCheckable( true );

connect( sound, SIGNAL(triggered( bool )), audioOutput , SLOT(setMuted( bool )
));
        exit = new QAction (iconexit,tr( "退出"), this );
        connect( exit , SIGNAL(triggered()), this , SLOT(exitSlot()));
}
```

7. 在 main.cpp 文件中添加中文支持：

```
#include <QTextCodec>
```

QTextCodec:setCodecForTr(QTextCodec:codecForLocale());

8.运行程序，效果如下：

9. 到此为止，程序能实现“ 打开 ”、“ 播放 ”、“ 暂停 ”、“ 停止 ”、进度条拉动，声音调节，静音功能。下一步工作是完善其他功能及歌词显示。