**Mp3播放器 QT Music**

**小组成员：刘宗熹 袁煦校 王正阳**

**1** 程序功能介绍（mp3播放器）

**2.1 开始界面**

点击“开始”按钮，可查看当前歌曲基本信息（歌曲名、演唱者、所属专辑、来源 即所属播放列表）

**2.2 歌单管理**

点击“管理”按钮，进行歌单管理操作（新建歌单、删除歌单）

1. 新建歌单：点击“新建歌单”，输入歌单名称，完成新歌单的创建（注意：新歌单的名称不能为空，也不能与已有歌单重复）
2. 删除歌单：点击“删除歌单”，选择欲删除的歌单（可多选），完成歌单的删除

（3）加入歌单：通过右键菜单将歌曲加入对应歌单

**2.3 播放控制**

1. 播放/暂停控制键
2. 播放模式：顺序播放（进入时默认）/单曲循环/随机播放
3. 切歌：上一首/下一首按钮；双击列表中歌曲，也可以切歌
4. 歌曲进度条实时反馈播放进度，且可拖动改变播放进度

**2.4 搜索歌曲**

点击搜索框，输入欲搜索歌曲的名称即可（搜索框有自动补全功能）

**2.5 mp3文件导入与读取**

用户可点击歌曲文件，将装有mp3直接放入弹出的文件夹中。程序讲寻找文件夹中所有的mp3文件，并从mp3文件中读取并显示歌曲信息（歌曲名、演唱者、所属专辑、发行年份、歌曲风格）

**2**  项目各版块与类设计细节

**2.1 界面设计**

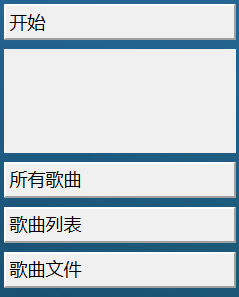
注：为了防止界面发生混乱，我们保证每次显示的窗口只有一个

**3.1 主界面widget（类型为qWidget）：**

  
包含控件

**4.1** LabelQTMusic 显示项目名称QT Music 图片略

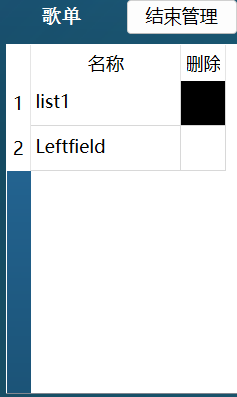
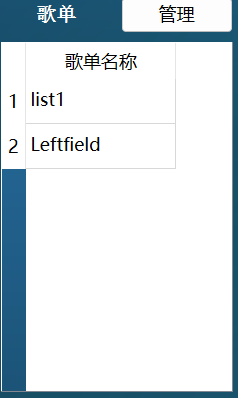
**4.2** toolbox用于切换开始、所有歌曲、歌曲列表、歌曲文件四个界面



**4.3 歌单栏及管理按钮**

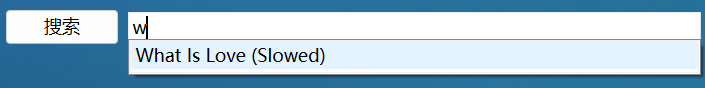
* 1. widgetSongList显示歌单信息
  2. pushButtonManage管理按钮

非管理模式（一般） 管理模式

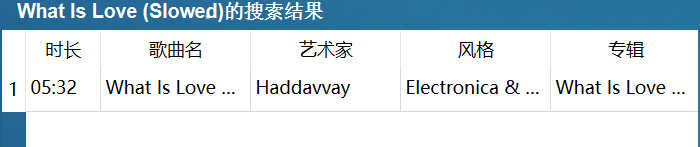


**4.4 lineEditSearch搜索框及pushButtonSearch搜索按钮**

搜索框有自动补全功能



选择选项后，tableWidget显示搜索结果，



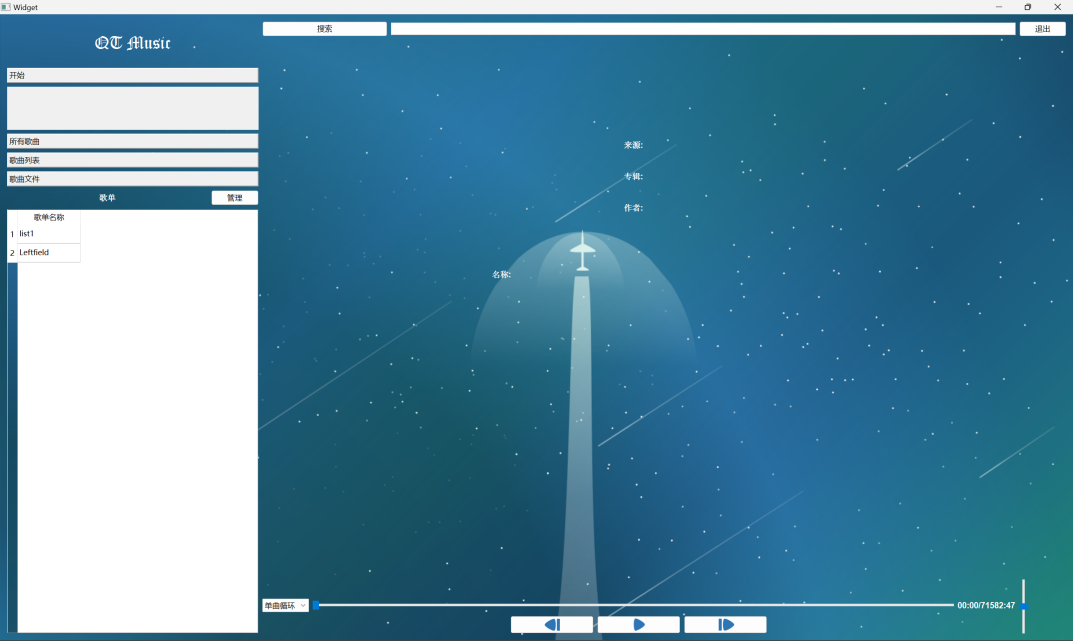
**4.5 Stackedwidget 包含四个界面**

开始界面 用Label显示歌曲名称、来源、专辑、作者信息（专辑图片还是没读出来呜呜呜）

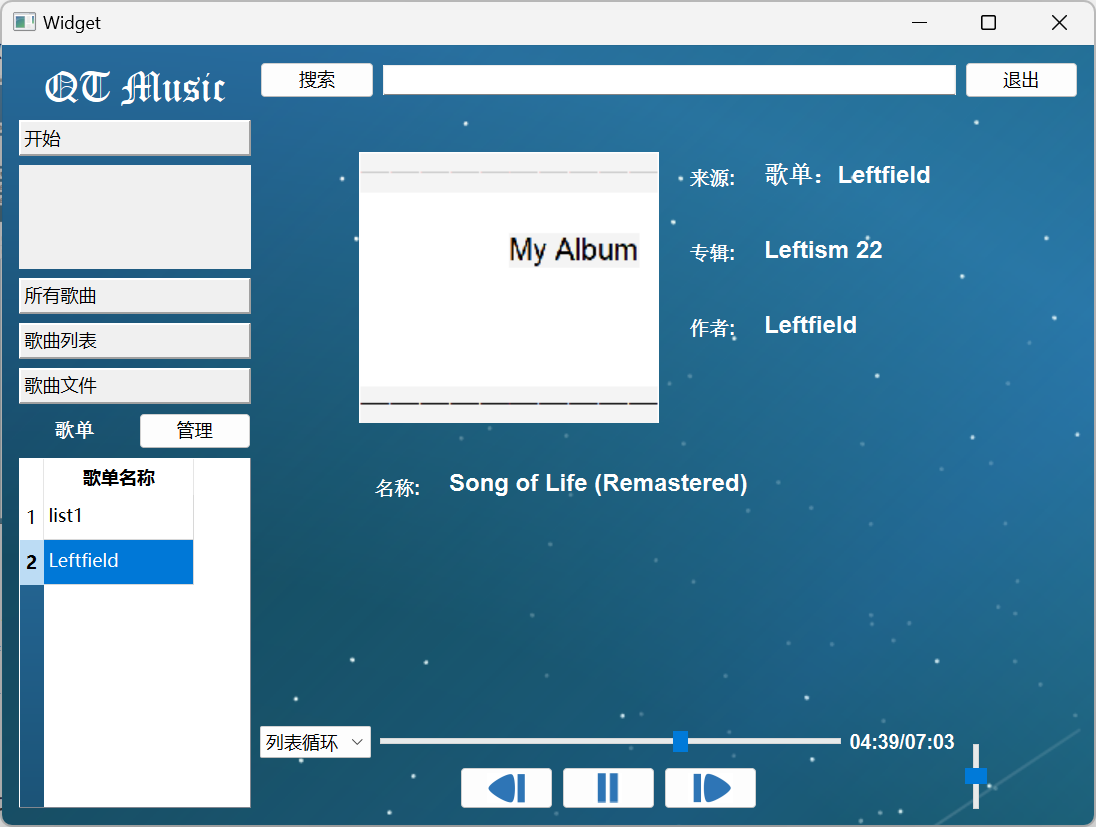
如果用户缩放窗口，各个Label会移动到合适的位置上

刚进入还未播放：

（1）开始界面



播放中：



（2）所有歌曲界面

Tablewidget以表格形式显示歌曲各种信息，同时labelListTitle显示“所有歌曲”

pushButtonLoadSong更新曲库，例如：打开“歌曲文件”，导入歌曲后，点一下，即可更新曲库

pushButtonSetAudio更新音频设置，例如：插入耳机之后，点一下，即可继续正常播放歌曲



（3）播放列表

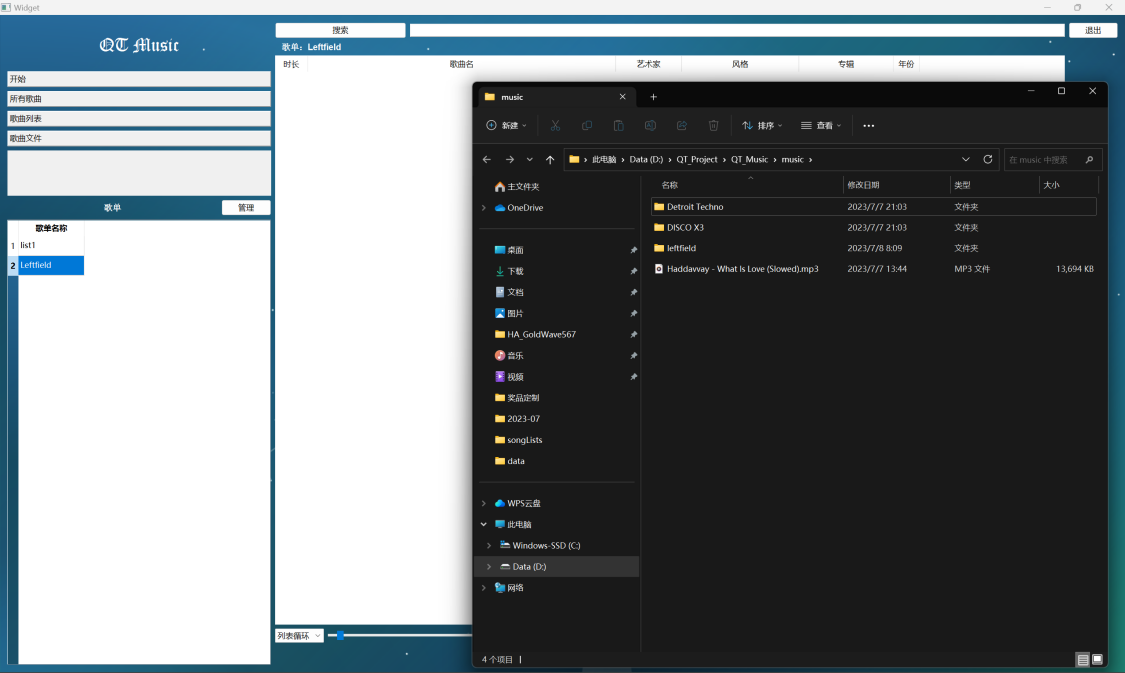
Tablewidget显示正在播放的歌曲列表（包括所有歌曲、歌单、搜索结果）的信息，同时labelListTitle显示对应的歌曲来源

播放列表与所有歌曲共用控件，后面的部分会详解怎么实现



（4）歌曲文件

弹出歌曲所在的文件夹，可以方便地添加歌曲，删除歌曲



**4.6 播放控制**

comboBoxPlayMode控制播放顺序

pushButtonPlayorStop播放暂停 pushButtonNextSong下一首 pushButtonLastSong上一首

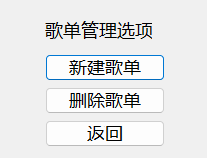
labelTime实时显示播放时间

horizontalSliderTime时间进度条 verticalSliderVolume音量调节（由于接口原因，只能实现滑块拨到底部是静音，不在底部是正常音量）

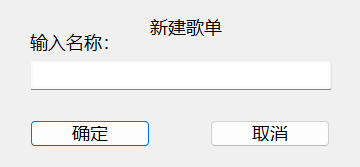
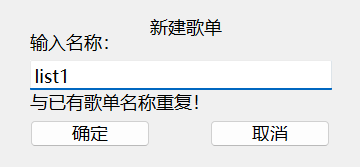
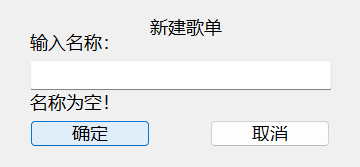


**4.7 退出按钮** 点击图片，弹出确认窗口，确定后推出程序 图片略

**3.2 歌单操作界面DialogSongListOp**



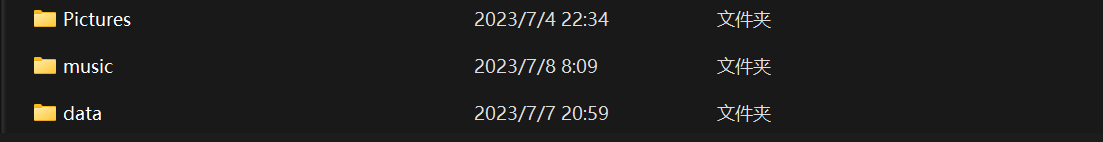
（1）点击新建歌单，打开新建歌单DialogNewSongList



（2）点击删除歌单widgetSongList进入管理模式

1. 点击返回，回到widget

**2.2 文件的设计：**



Pictures包含所有的贴图

Music包含用户导入的mp3文件

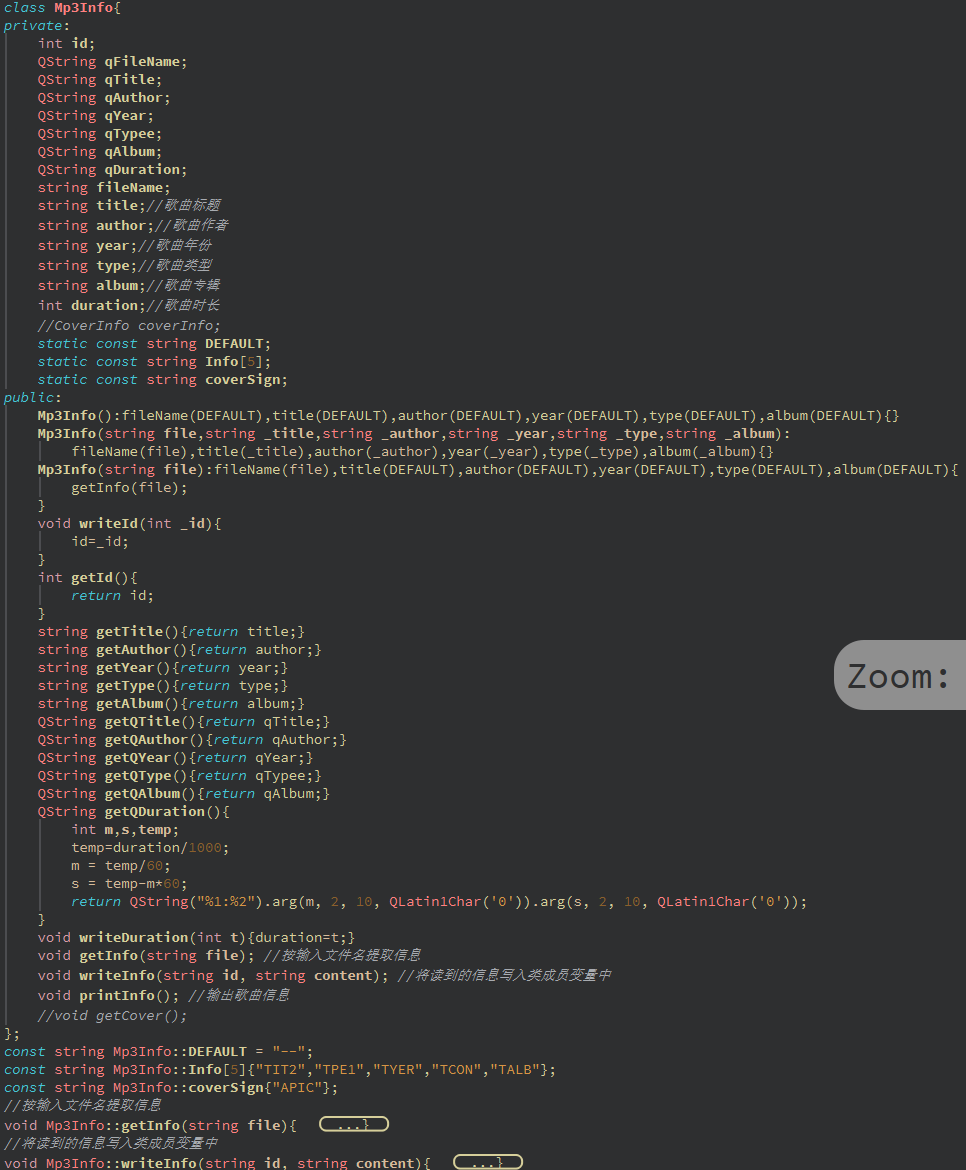
Data包含歌单相关的数据

**2.3 类的设计：**

**3.1 Mp3Info类（需要特别指出的是，qt自带的mp3接口无法顺理读取mp3信息，我们自行逐字节读取了这些信息）**

读取Mp3信息：

程序启动时，从../QT\_Music/music读取所有mp3文件地址，导入字符串数组fileNames。同时，将读出这些mp3文件的信息，并赋予每个文件一个整数编号（以下称为“绝对编号”），这样，所有歌曲都能用整数存储。其他成员如图：



**3.2 Widget类**

主窗口，除了ui控件外还包含如图的成员

**4.1**  **右键菜单功能**

将歌单视为一个个的QAction,用vector储存起来，逐个加入QMenu

当右键选中tablewidget歌曲时，发出信号，做：

一是打开右键菜单；二是取得选中的歌曲，可把歌曲加入对应歌单

**4.2** **songOrder歌曲顺序类**

（1）实现单曲循环、顺序播放、随机播放三个功能，封装成类后，可通过直接改变一个整型变量PlayMode改变播放顺序

（2）需要储存的歌曲顺序有两种，一是用户正在查看的窗口的歌曲顺序，用vTemp(vector<int>)显示；二是正在播放的歌曲的顺序，用v储存。用户一旦点击类tablewidget，vTemp中的数据就储存到v中。

（3）单曲循环：略 随机播放：略

列表循环：建立0到num和列表中绝对序号的映射

其他窗口类不再赘述



**3.3 SongList歌单类**

存储了所有歌单的信息和操作

**4.1 文件读写：**

（文件读写的难点在于，切换窗口时，数据必须写入文件中，在窗口生成时再次读出）

（1）"..\data\allSongList.txt"包含所有的歌单的名称

如果出现下列情况：  
1.Widget类调用构造函数；

2.歌单信息发生改变前（新建或删除）

就从文件中读出歌单信息

如果出现下列情况：  
1.Widget类消亡；

2.歌单信息发生改变后（新建或删除）

就向文件中写入歌单信息

1. 在tableWidgetSongList处于管理模式时：

"..\data\deleted.txt"包含待删除歌曲的名称，在切换窗口时，对应地进行读写

**4.2** IsManage表示歌单是否处于管理状态

**3.4 OneList单个歌单类**

**SongList**类中包含vector<OneList> l

OneList类存储单个歌单的歌曲信息

**4.1 文件**

"..\data\songLists\（歌单名字）.txt"

储存了歌曲的序号信息

将歌曲加入歌单时，文件的读取机制基本相同

由于歌单新建的需求频繁，构造函数中，文件能够自动新建初始化；删除歌单是，文件也会自动被删除

**4.2**  由于歌曲需要频繁地添加、删除，采用list来储存歌曲信息。在显示歌单中歌曲时，需要将list中歌曲转换成vector传递到SongOrder类中处理

**3.5 其他细节**

（1）歌曲的播放功能通过qt库<qmediaplayer>实现

（2）小组选择跳过中文编码问题，故整个程序读入的歌曲信息中不能含有中文，其他部分可以使用中文

（3）变量、对象名全部采用小驼峰命名法；类名全部采用大驼峰命名法

（4）对于qt控件，采用种类在前，功能在后的方法，如搜索框是lineEditSearch，虽然有些长，但能有效避免歧义。

**3 小组成员分工情况**

刘宗熹：

构思环节：搭建了播放器的架构与类，并且定出了每个环节的DDL，为负责前端的同学提供了模块化的任务

实操环节：主要负责后端：窗口界面设计与逻辑关系，将前端完成的类合并到工程中；同时负责部分前端内容，比如歌曲顺序类，主要撰写了报告

袁煦校：

构思环节：从前端的角度，促进播放器架构的模块化，整理出了一些重要的类

实操环节：主要负责前端：解决了MP3文件类，歌单类等模块化的问题

王正阳：

构思环节：参与讨论，结合平时的使用经验提出创意

实操环节：完成了部分函数，绘制了背景图片，撰写了部分报告

时间线：5.29组员初步聚会，定下所有的日程——6.4组员固定正式聚会，讨论进度——6.7路演，在这之前完成初版——经协商，从路演结束至期末考试结束，暂时搁置项目，小组成员投入期末复习——7.8上交项目（其中，非正式聚会越多越好，每有进展，及时向群里汇报情况）

**4 项目总结与反思**

本次项目没有选择游戏，而是选择了功能不甚有新意的音乐播放器，但是这样的选择符合小组的兴趣，很大程度上激发了成员们的热情。

在开发项目的过程中，几乎算是摸爬滚打，从分工、设计到具体实现都经历了一些改动。

分工上，一开始想象的是三个人分别负责程序搭建、mp3读取、美术设计，但是实践的过程中发现程序搭建和mp3读取难解难分，行进颇有困难。项目进行到一半，又从别的小组了解到前端和后端的分工方法，最后形成了报告中所写的分工。

具体设计、实现的过程中，很多环节非常具有挑战性，超出了预期，经常在寥寥数行代码上耗费一个下午乃至一天多的时间。比如说使用<QMediaPlayer>库时，一直调库失败，最后花了一天半的时间才发现是配环境的时侯少下载了一些组件。令人影响深刻的是，袁煦校在不了解前端和后端分工方法的时侯就写出了风格优良、封装良好的代码。刘宗熹坦言，在仔细阅读其代码之后，自己对程序设计课上讲授内容的理解更加深入了。

某次Debug到绝望的时候，刘宗熹偶然发现其舍友的小组会定期保存稳定的版本，防止改代码的时侯改到崩溃前功尽弃，遂高兴地取用之。到项目结束的时侯，已经保存到1.10。和舍友交流的时侯，常常可以发现很多小妙招，可以方便快捷地学到手，经此一役，着实体会到了和他人交流的重要性。

最后项目结束时，由于个人能力的原因，很多预定的功能没有完成，也没有一个非常有经验的人来告诉我们目前的代码风格是否优良，有没有向错误的思路上行进。虽然有不少遗憾，但小组收获颇丰，代码能力有了较大提升，上网debug也更加自如，对常打交道的mp3文件有了更深的理解。下次再做大作业时，应该努力增强团队的交流，尽量调动起每个组员的积极性，大家都努力看看别的小组的开发情况，可以大幅增加在开发过程中的收获，同时少走很多弯路。另外，由于没有想清楚实现目标而盲目浪费的时间也有很多，今后开发时，要努力保持头脑的清醒，不空耗精力。