

基于 QT 的音乐播放器

舒博

四川大学锦城学院计算机与软件学院 四川 成都 610000

【摘要】在今天的这个高科技众多的时代，越来越多的人投入软件工程这个专业方向，有不少人都喜欢用音乐播放器目标练手，而利用 QT 的强大图形界面的功能，及其优良的跨平台性能，能够做出优美的界面和全面的功能，本播放器具有播放，暂停，曲目选择等最基本的功能外，还利用数据库实现对歌曲下载，上传等不同用户有专属的音乐列表。

【关键词】音乐播放器；QT

1 整体功能及架构设计

1.1 播放器基本功能

用户能够对歌曲暂停，播放进度调整，上/下曲播放，选择歌曲，列表的管理，及其对于播放器的皮肤，歌词的选择等操作。

1.2 播放器整体架构

1.2.1 播放器硬件架构

播放器由 QT 客户端，MYSQL 数据库及文件管理系统组成，所有音乐相关的文件保存在服务器文件管理系统中，客户通过对客户端的操作，进而对数据库的信息进行拉取及修改，进而对歌曲进行一系列操作，硬件整体结构如图 1 所示：

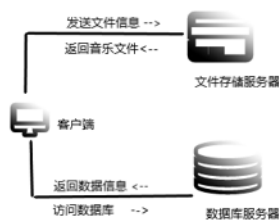


图1 硬件整体结构

1.2.2 播放器软件架构

QT 客户端采用 QT Creator 作为开发环境，利用它可以制作出具有强大功能及美观外部设计的播放器。由于我们不需要体积太大，成本太高的数据库服务器但要求速度快所以我采用被广泛地应用在 Internet 上的中小型网站中的 MySQL 数据库服务器。使用它最重要的原因是他开放源码，也是是开放源码这一特点，所以许多中小型网站为了降低网站总体拥有成本而选择了 MySQL 作为网站数据库，它对多种语言提供了 API，可以对数据进行增删改查等一系列操作^[1]。

1.3 数据库的设计

数据库设计有几类数据表，分别为用于存放用户信息、存放歌曲信息、存放用户列表信息。其中用户信息表中存放了包括用户名，用户密码，用户序列号，及其拥有的列表名这一类数据等，歌曲信息包括了歌名，专辑，时长，歌曲地址，歌曲 id，歌词地址，作者等等信息。列表信息表中包括了列表名，列表所属，列表序列号，及其包含歌曲的一系列信息。我这里用的是 Navicat Premium 对表格直接操作的，表格如图 2：数据库表格样式，图 2 中可以看到字符集为 utf8 格式，这个很重要必须要与客户端连接是设置的字符集格式对应，否则会显示乱码。设计表格具体步骤为：1、打开 Navicat Premium，点击左上角的连接，我这里选择 MYSQL，在输入连接名等一系列后连接。2、在打开连接右键选择新建数据表，界面会如图 2：数据库表格样式，名则为列名，类型为要设计的数据类型，比如时间可以用 date 类型，其中添加，插入，删除栏位是对列的操作。3、在‘选项’板块里可以设置字符集，我这里设置的是 utf8 字符集格式。

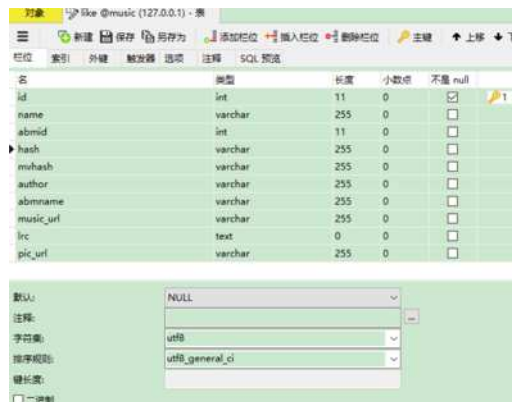


图 2 数据库表格样式

2 基于 QT 的音乐播放器设计

2.1 播放部分

QT 客户端利用 multimedia 模块中的 Qmediaplayer 类对多媒体文件进行播放, 所以要在 .pro 文件中加入 QT+=multimedia, 当然默认播放音频的格式是有限的, 我们可以安装 LAVFilters 来播放更多的格式的音频。LAVFilters 是一个过滤器可为大多数类型的音频或视频文件提供通用支持。

实现音乐播放 / 暂停的核心是利用 QMediaPlayer 类的 pause() 函数及 play() 函数进行暂停及其播放, 利用 QMediaPlayer::PlayingState 函数判定播放状态, 其中 QMediaPlayer::PlayingState 函数共有 QMediaPlayer::PlayingState=1 播放, QMediaPlayer::PausedState=2 暂停, 和 QMediaPlayer::StoppedState=0 停止 3 个枚举变量, 我们可以利用它们来实现播放暂停和停止。具体代码如下:

```
QMediaPlayer * playmusicbtn = new QMediaPlayer;
if(playmusicbtn->PlayingState()==0){ playmusicbtn
->play();// 播放 }
else if(playmusicbtn->PlayingState()==1) { playmusicbtn
->pause();// 暂停 }
```

else {playmusicbtn ->play();// 播放 }。而播放列表则可以用 QMediaPlaylist 类实现, 当然也可以用表格控件来做。其内存放的是歌曲的地址, 我们可以通过 QMediaPlayer 类中的 Setplaylist(Qmediaplayer) 函数来设置播放列表。

2.2 歌词部分

歌词可分为时间部分和歌词部分, 利用 map 来存储临时歌词, 再通过 QMediaPlayer 类的 positionChanged 消息的相应函数里面。对比时间, 现在播放的时间处于前面记录下来的时间的 list 中的哪两个之间, 就显示相应的歌词。非桌面歌词的显示可以直接在 QLabel 控件里显示, 当然如果要用歌词条进度需要用到鼠标事件, 我们不深究。而桌面歌词则可以新建一个带 UI 的类。歌词渐变效果我这里是利用 QGradient 类和 QPainter 联合实现, 当然也用到了 QTimer, 它们都属于 gui 模块所以我们需要在 pro 文件加入 gui 模块, 方法和上面一样, 具体思路为利用 QPainter 绘制出歌词, 再利用 QGradient 下的 QLinearGradient 类进行线性渐变设计, 线性渐变的注意点是 setStart(), setFinalStop(); 这两个函数要成对出现, 用法是利用它们参数 x 与 y 的值来实现垂直渐变还是水平渐变, 当 x 相同时及为垂直渐变, y 相同时及为水平渐变。要实现歌词渐变, 需要绘制两次歌词, 第一次为底色, 第二次为渐变色。垂直方向代码如下:

```
QLinearGradient lrcczjb;
lrcczjb.setStart(10,10);
lrcczjb.setFinalStop(10,30);
lrcczjb.setColorAt(0.2,QColor(14,200,255));
lrcczjb.setColorAt(0.6,QColor(14,50,255));
lrcczjb.setColorAt(0.9,QColor(14,200,255));
}
```

2.3 连接数据库部分

数据库的连接与操作采用了 QSqlQuery 及 QSqlDatabase 类来实现当然要用到他们需在 pro 文件加上 QT+=sql, 新建一个 QSqlDatabase 对象代码为 QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL"); // 数据库对象

```
db.setHostName( "" ); // 数据库服务器 IP
db.setUserName( "" ); // 数据库用户名
db.setPassword( "" ); // 数据库密码
db.setDatabaseName( "" ); // 使用哪个数据库
QTextCodec *textcodec =QTextCodec::codecForName( "UTF-8" ); // 设置字符集格式
```

```
QTextCodec::setCodecForLocale(textcodec);
```

这里设置的文本格式为 UTF_8, 那数据库的格式设置一定要一致否则会出现中文混乱码。对于获取数据库信息则直接使用 QSqlQuery 类创建的变量 query 的 query.exec(str) 函数对数据库进行增删改查的操作, 其中 str 为操作数据库的语句。对于查询到的内容用 query.next 进行遍历, 每个 query.value 代表一个表格的值, 我们其实可以把 query.next 看成 mysql 表格的行而 query.value 看做是列。

2.4 样式皮肤设置

Qt 是一个强大的工具, 它的强大之处在于它不仅能够创建一个成熟的应用程序应该具有的功能特性, 而且还能创建一个美丽的外观和感觉, 这将使应用程序更加完整, 对用户更具吸引力。其不仅有跨平台的 UI 开发框架, 还有非常强大而且更灵活的界面外观设计的机制。Qt 样式表能够自定义各个部件外观, Qt 样式表的概念、术语和语法类似于 CSS, 我们把他叫 QSS。它可以设计出非常精美的界面, 且设计方法多种多样。其一是最方便的右键点击控件中的‘改变样式表’设置样式, 这样设置很方便不仅可以快速添加资源, 颜色, 字体等, 还可以粗略的检查格式是否正确, 当某个地方格式错误, 左下角的有效样式表字样会淡化。当然这样设置缺点是程序运行开始便固定样式不能改变, 那么我们在程序中需要改变样式怎么办? 这就是我要讲的其二, 在代码中设置控件样式, 代码以按钮为例如下: setStyleSheet("QPushButton { color: blue } "); // 设置字体

颜色为蓝色，当运行到这段代码时，按钮中的字变成了蓝色，当然这样是设置每个按钮，那要设置某个控件呢？其实每个控件都有 `setStyleSheet()` 函数，例：`ui->pushButton1->setStyleSheet("background-color:red; color:blue");` 这样就是设置 `pushButton1` 的样式了，在 QT 中我们直接设置有效样式表时要注意，如果没特别写明是哪一种控件，在父控件设置样式时，子控件会继承父控件的样式，就如 `QFrame{background-color:blue}` 这个样式，我们在 `QFrame` 中设置背景颜色，那么 `QFrame` 类其下的子控件所有背景颜色都会跟随 `QFrame` 的样式，所以我们可以这样设置 `QFrame#frame{background-color:blue}` 指定控件 `frame` 设置样式。接下来是样式设置的规则，一个样式设置规则是由一个选择符和一个声明组成。选择符指定了受该规则作用的控件，声明指定了这个部件上要设置的属性，就如 `QLabel { color: blue}`，`QLabel` 是选择符，`{ }` 中的是声明，`color` 是属性，`blue` 是值，每一种属性和值都由 `' ; '` 分开。对于复杂的控件还有许多的子控件，就比如进度条 `QSlider` 中的 `add-page` (增加进度条)，和列表控件的 `border`，我们可以单独设置其样式如 `QSlider::add-page:Horizontal{...}` 这里就不详说了，最后是控件的伪状态，一般来说我们经常用到 `hover` 伪状态，其作用就是当鼠标悬停在控件上时执行样式，除了 `hover` 还有 `checked` (点击) `pressed` (按下) 等等。接下来是皮肤设置，我这皮肤并不是很复杂，只是利用按钮显示出对应皮肤的图片，点击按钮直接把页面背景图修改为对应图片，对于皮肤按钮的显示可以利用 `QWidget` 或者 `QFrame` 显示当点击皮肤图标识设置显示

上面的控件，再次点击时隐藏控件，这样就做到了皮肤选择框架的样式，背景设置具体代码如下：`ui->frame->setStyleSheet(QString(`

```

        "QFrame#frame" // 指定控件
        "{ "
        " border-radius:10px;"
        "border-image:url(/image/5910636204220842
1aaabcb9.jpg);" // 背景图片
        "background-color:rgb(170, 255,
255,100%);" // 背景颜色
        " border:0;" // 边框
        " margin:0;"
        "}" );

```

这里要注意在代码设置的样式每行都必须加上 `" "` 但不接受注释。

3 结束语

本播放器并没有运用到很复杂的代码，对于一些软件编程的萌新来说，用 QT 制作的音乐播放器可以用来练手，而且对熟悉 QT 的基本功能及其样式设置都会有很大的帮助。

【参考文献】

- [1] 何欣茹. 网易云音乐营销推广策划 [D]. 杭州: 浙江大学, 2014.
- [2] 陈振宇. 基于 Android 的手机音乐播放器的设计与开发 [D]. 西安: 西安电子科技大学, 2014.