**毕 业 设 计 [论 文]**

题 目： 基于iOS的音乐播放器

学 院： 计算机科学与工程

专 业： 计算机科学与技术

姓 名： 郭晓甜

学 号： 081412220

指导老师： 张芳芳

完成时间： 2016年05月18日

摘要

随着时代的发展，社会的进步，人们的生活逐步变得越来越舒适，同时对生活水平和质量的要求也越来越高。伴随着智能手机的发展，人们越来越开始享受并依赖智能手机。现代化的智能手机几乎成为现代人生活中不可获取的一部分，它的出现彻底改变了现代人的生活方式，从传统的无论是购物，餐饮还是其他娱乐等都要到实体店的方式改变成了如今的只有手机在手，完全足不出户就可以实现一切的方式。

在人们日常生活方式悄然改变的同时，对生活质量的要求，对生活的舒适化的追求也越来越高，网上购物，网上订餐，亦或是网上叫出租车等等，手机的娱乐化也越来越丰富多彩。

本文主要是基于iOS系统，采用Object-c语言在Mac OS 上利用XCode软件编写的，同时给出了需求分析，设计目的，和详细设计等过程以及部分图形界面和主要功能的流程图。音乐播放器的功能涉及到，上一曲下一曲，音量调节，歌词显示，歌词开闭等友好功能。

关键词：Object-c；iOS；MVC；音乐播放器；

# Abstract

With the development of the times, the progress of the society, people's life gradually become more and more comfortable, at the same time, the quality of life and the quality of the requirements are also getting higher and higher. With the development of smart phones, more and more people began to enjoy and rely on smart phones. Modern smartphones become almost can not obtain a part of modern life, it completely changed the way of life of modern people, from traditional whether it is shopping, restaurants and other entertainment to store to change into the now is only the phone in hand, completely can stay at home all the way to achieve.

In people's daily life style quietly changed at the same time, the requirements on the quality of life, for life comfortable in the pursuit of increasingly high, online shopping, online ordering, or online call a taxi and so on, mobile entertainment has become increasingly rich and colorful.

This paper is mainly based on IOS system using Object-C language in Mac OS using Xcode software written, and give the requirement analysis, design, and detailed design process and flow charts of graphic interface and main function. Music player function involves, the next song, volume control, lyrics display, lyrics, open and close, and other friendly functions.

key word：Object-c；iOS；MVC，music player；

目录

[摘要 I](#_Toc12463)

[Abstract II](#_Toc250)

[1 序言 2](#_Toc20037)

[1.1设计背景 2](#_Toc26893)

[1.2国内外研究现状 2](#_Toc6446)

[1.3设计思想 3](#_Toc11788)

[1.4设计目的及意义 3](#_Toc19357)

[2 应用可行性研究 5](#_Toc15007)

[2.1可行性分析 5](#_Toc16877)

[2.2需求分析 5](#_Toc27831)

[2.3系统开发工具及环境介绍 6](#_Toc12586)

[2.3.1开发环境 6](#_Toc16526)

[2.3.2相关工具介绍 6](#_Toc15003)

[3 应用总体设计 8](#_Toc22647)

[3.1音乐播放器的功能概述 8](#_Toc3273)

[3.2各功能模块介绍 9](#_Toc28757)

[4 应用功能详细设计 12](#_Toc1434)

[4.1界面设计 12](#_Toc442)

[4.2应用控件组件介绍 22](#_Toc9855)

[4.3整体架构设计 23](#_Toc17908)

[5 问题与解决方案 25](#_Toc29374)

[5.1歌曲播放和歌词同步问题 25](#_Toc28053)

[5.2数据持久化问题 25](#_Toc20831)

[6 软件性能测试 27](#_Toc24896)

[6.1软件测试目的 27](#_Toc5969)

[6.2软件测试方法 27](#_Toc15798)

[7 总结 29](#_Toc17326)

[8 致谢 30](#_Toc3453)

[9 参考文献 31](#_Toc1882)

1 序言

1.1设计背景

进入二十一世纪后，我们国家也逐步步入移动互联网的时代，同时也得到社会各界的大力支持，一直处于稳步并且有效的发展状态。移动互联网的发展给我们的生活带来了巨大的改变，我们的生活方式也在不知不觉中变得越来越多元化，现代化。

近几年来，智能手机的发展，使得移动互联网的发展得到了很大程度的深化。智能手机也越来越大众化，普通化，已经不仅仅只是我们日常用来打电话，发短信的简单工具，其应用功能的发展，也越来越被大众所看重。在传统功能的基础上，人们对手机的应用娱乐功能的要求也越来越高。在众多的娱乐应用功能中，听音乐也是人们日常娱乐不可或缺的一项重要功能。好的音乐播放器可以更好的体现手机功能的强大和用户的舒适体验。音乐播放器经过近几年的发展，无论是规模还是市场都已经具备相当规模了，而且都很有竞争力。苹果手机进入中国市场之后，一直深受大众的追捧，与安卓手机形成了强大的竞争力，所以基于iOS的手机应用娱乐软件需求量迅速上升，而市场上的播放器也是纷繁复杂，种类各异，基本上满足了大众的需求，比如酷狗音乐，QQ音乐，百度音乐等等也纷纷出现了支持iPhone版本的软件，为广大苹果手机用户带来了福音。

苹果手机的引进，在国内市场掀起了一股疯狂的苹果风，很快的iOS技术也逐渐开始流行起来，支持iOS的Object-c语言也迅速成为程序员追捧的开发语言，并且每年都有大量程序员涌入Object-c的学习工作中。成为了继C语言，java之后又一强大开发语言。基于此，本人决定采用Object-c来开发iPhone手机应用。

1.2国内外研究现状

iOS系统最早是由苹果公司设计研发的，并支持提供苹果公司所推出的一系列产品所使用。如同苹果电脑是基于Mac OS X 一样，也是以Darwin为基础。在2010年的WWDC大会上，苹果正式对外界宣布更名为iOS，并且系统架构分为4个层次，分别是：核心操作系统层（the Core OS layer），核心操作服务层（the Core Services layer），媒体层（the Media layer），可轻触层（the Cocoa Touch layer），其整体所占内存相对较小。

相对于安卓来说，iPhone是不开放源代码的封闭系统，我们无法去了解具体系统内部的构造，但是系统相对稳定，而且移植性非常好。iPhone所面向的客户群相对高端，主打个性化路线，对于苹果公司所出的系列产品，开发人员开发的应用均可适用。

良好的平台基础和易上手的开发语言，为开发音乐播放器奠定了基础。国外如今也出现很多形态各异的音乐播放器，这一方面技术也比较成熟，产品相对都很受用户喜欢。所以在iOS系统下开发一款小巧精美的音乐播放器是完全可行的。

1.3设计思想

本人设计的音乐播放器是按照一般的应用程序开发过程进行的，通过调查了解大众人群对音乐播放器的需求，总结出了用户心目中所希望的播放器的简洁，方便，轻巧等特点。所以决定自己开发一款轻便小巧的音乐播放器。

设计初期，本人参阅并借鉴大量相关成熟产品的框架结构等，虽然由于iOS的代码非开源不能够直接参阅，但是本人在网上相关社区以及查阅相关书籍来学习借鉴比较好的想法和思路。总结了前辈们的经验之后本人决定采用比较热门的MVC设计模式来构建基本框架。但是由于本人对iOS的学习时间和周期都比较短，很难做到对软件设计的综合考虑，于是决定去除掉与播放无关的其他功能，只留下播放，歌词同步显示等功能。虽说比较基础，但是满足基本的音乐功能还是可以的。

1.4设计目的及意义

移动互联给人们的生活带来了巨大的便捷，也越来越成为人们生活中不可缺少的一部分，移动端的开发也是时下很流行很热门的行业，越来越多的人群投入到移动端得研发中导致很多软件的外观虽然华丽但是无关的功能却过于强大，浪费资源占用CPU耗费内存，导致用户在多任务操作的时候总会受到得不少的影响。对于多数普通用户来说，软件中很多功能都是毫无意义形同虚设的，针对目前很多软件的此类弊病，本人选择了设计可以满足用户最根本需求的音乐播放器。

2 应用可行性研究

2.1可行性分析

可行性分析，也叫可行性研究报告，是对一款软件应用或者系统的可行性做出分析总结。软件的设计基础不能是盲目的，必须要从经济，技术，实用性等方面做可行性的分析，从而能够尽量减少资金，时间的浪费，保证软件系统的开发完成，并最终达到实用性良好的效果。

1. 经济可行性：众所周知，苹果系统是区别于现在市面上的安卓系统和window系统的，在传统的window系统下无法完成开发，首要因素是需要有一台基于iOS系统的苹果电脑，现阶段虽多种原因暂无法购买苹果设备，但可以在window电脑下安装苹果系统（我们促俗称黑苹果），从而解决了硬件限制的因素。所以从经济上来说是完全可行的。
2. 技术可行性：基于iOS的音乐播放器，所用到的主要技术有----开发语言Object-c，UI设计，第三方类库的应用等等。就开发语言而言，自己已经学习了iOS软件开发有三四个月的时间。太过复杂的技术因水平有限暂无法完成，但就简单一点的软件，自己应该还是可以完成的。所以从技术角度来讲也是可行的。
3. 操作可行性：设计的软件是在苹果系统下完成的，需要基于iOS系统，同时XCode本身自带虚拟机可以做各类调试，同时也可以借用他人的苹果手机做真机调试，没有太过繁琐的流程等。所以在操作层面也是可行的。

综上所述 ，该音乐播放器的开发可行性还是很高的，可以进行接下来的工作。

2.2需求分析

所谓需求分析，一般是指在开发新的应用产品的时候，就其目的，作用，意义等方面做出综合分析。一般在应用产品的可行性研究之后，需要做需求分析。不同的应用产品所针对的用户可能各不相同，作为开发者，在新产品研发之前必须要详细的了解客户的需求，不能简单盲目的以偏概全，更不能依照个人喜好去随意开发，需要多与客户交流讨论，去了解确定客户的真实需求后，才能着手去开发应用。

就目前针对苹果这一款产品而言，因其十分重视用户体验，所以经过近几年的发展已经拥有相当数量的用户群，但就音乐播放器而言，已经拥有了不小的规模。不过市场上的音乐播放器种类相对比较繁多，没有确切的定位，虽然说基本上满足了大部分用户的需求，但是总体的市场情况并不是很乐观。相对于功能强大，外形华美的音乐播放器而言，新颖，小巧的外形则更受现在年轻人的喜爱和欢迎。所以这一方面市场发展前景还是比较广阔的，也正是在此前提下，设计一款更简洁，更美观的产品也是具备较大的需求的。

2.3系统开发工具及环境介绍

2.3.1开发环境

电脑硬件：iMac，Mac book

处理器 2.7GHZ intel Core i5

操作系统：OS X 10.0以上

开发工具：Xcode 7.3

运行环境：iphone5 及以上均可

运行系统：ios8.0及以上均可

2.3.2相关工具介绍

1. Object-C

对于整个软件来说，开发所用的语言是Object-c。作为计算机中流砥柱的语言----C语言，属于面向过程的语言，而Object-c则是C语言的升华版，其最大的特点就是面向对象。主要应用于Mac OS X 和GNUstep这类采用OpenStep标准的系统，而且Object-c也是NeXTSTEP和OpenStep中的基本语言。同时Object-c也支持Gcc运作的系统，因为Gcc编译器中同样包含了Object- c编译器。近些年来，由于苹果手机的盛行，以及用户体验第一位的宗旨，促使Object-c语言相对于C语言，java等开发语言，排名有明显的大幅提升，毕竟Object-c一直以来都用于编写iPhone应用程序。Object-c是实用性很强的软件开发语言，它是由C语言写成的比较小的运行库，让应用程序的大小降低了很多，节省了很多内存，与大部分OO系统所使用大量的VM执行时间取代了整个系统的运作恰好相反。采用Object-c所写出的程序往往与其原始代码相差无几，而且其函数库也和Smalltalk系统需要使用很大的内存去打开窗口的情况相反。

1. XCode开发工具

XCode是苹果公司为iOS软件开发人员提供的集成开发环境，开发人员所建立的基于iOS的应用的最快捷的方式。运行环境需要是苹果公司推出的Mac操作系统下。

作为一个软件开发人员，不论是采用C语言，C++，Object-C或者是Java语言来开发程序，不论是在AppleScript中编写脚本，还是试图从其他工具中转移代码，你都会发现XCode的编译速度十分迅速，每次操作都很便捷轻松，这也是很多iOS开发人员很喜欢做iPhone软件开发的一个重要因素。

苹果公司同时也为软件开发人员提供了全套免费的Cocoa程序开发工具，与Mac OS X一起发行，在苹果公司的官方网站便可下载使用。

3 应用总体设计

在经过前面可行性分析与需求分析之后，本章主要介绍软件的总体设计，对于一款应用，其所实现的功能的好坏是衡量一个软件质量的重要标准，通过对音乐播放器的全面分析并且结合调研情况，可以得出音乐播放器的功能模块与结构图，从而更好地去了解系统的结构与功能。

3.1音乐播放器的功能概述

总体来讲，我的音乐播放器分为以下几个模块：

1. 本地音乐：本地音乐主要包括已经提前预存好的歌曲，可以实现任意点选轮流播放功能。
2. 网络音乐：因网络数据的获取存在较大的问题，所以本软件采取的方式是抓取了某一服务器上构建的音乐列表来呈现。
3. 个人中心：此模块主要包含个人昵称，个性签名，问题反馈以及版本信息等部分。

框架如图3.1所示：

音乐播放器

个人中心

本地歌曲

网络歌曲

图3.1音乐播放器功能模块

3.2各功能模块介绍

音乐播放器功能模块详情如下：

1. 本地歌曲

用户可以通过点选此模块来进入本地音乐播放列表，音乐播放功能包括歌曲的播放，歌词的显示或者隐藏，和歌词的同步跟进，歌曲的暂停，播放，以及播放上一曲，下一曲，播放歌曲音量的调节（包括增加音量减小音量）等功能，如图3.2.1：

本地歌曲

点击播放

跳转至播放界面

歌词隐藏

歌曲播放

音量调节

打开同步歌词

图3.2.1本地歌曲模块

1. 网络歌曲

用户可以根据此模块来选择网络上的歌曲，因所抓服务器中包含的文件有限，所以暂无法提供更加全面的歌曲列表，不过就目前来看，相信还是可以满足用户的基本听歌需求。

1. 个人中心

用户可以在此界面下对音乐播放器进行一些简单的设置，比如修改昵称，改变个性签名，在反馈信息一栏，用户可以点击进入提供一些个人的看法和意见，以及软件当前版本信息等操作，让播放器更能适合用户各自需求，如图3.2.3所示：

个人中心

反馈意见

查看版本

个性设置

图3.2.2 个人中心模块

4 应用功能详细设计

4.1界面设计

软件整体涉及到用户可见界面一共有五个：

1. 首页

首页以风格画面外加三个可触碰按钮构成，分别可以跳转到下面不同的三个操作界面。这一界面主要在storyBoard中拖拽完成所以并没有涉及代码问题，相对比较简单。如图4.1.1所示：



图4.1.1首页

1. 本地歌曲界面

此界面主要提前存放好的歌曲列表为主，可以播放本地任意歌曲，点选任意歌曲之后可以跳转到歌曲播放界面，在读取本地文件过程中首先需要定义一个可变数组，通过重写get方法（懒加载）获取plist文件中的数据，代码如下：  
-(NSMutableArray \*)songListArray  
{  
    if (\_songListArray == nil) {  
        NSString \*path = [[NSBundle/\*懒加载过程\*/ mainBundle] pathForResource:@"SongsInfos" ofType:@"plist"];  
        NSArray \*array = [NSArray arrayWithContentsOfFile:path];  
        NSMutableArray \*models = [NSMutableArray array];  
        for (NSDictionary \*dictionarys in array) {  
            MusicModels \*cellModel = [MusicModels musicWithDictionary:dictionarys];  
            [models/\*逐个添加至可变数组中\*/ addObject:cellModel];  
        }  
        \_songListArray = models;  
    }  
    return \_songListArray;  
}

如图4.1.2所示：



图4.1.2 本地歌曲

1. 网络歌曲界面

在多次尝试网络抓取失败后，本人在论坛上获取某服务器端的歌曲来当网络歌曲播放，因为是从网络端获取的文件一般以JSON数据为主，需要先解析具体的数据内容，之后自定义各类属性来接收相对应的数据内容，同时在界面的搭建上，为了能够更好地给用户提供视觉效果，所以本人决定采取手动自定义的方式来布局整个播放界面，同时用到了第三方框架的支持。所用到的一些重点代码有：1.歌词解析过程：lrc格式的歌词都是以时间+歌词的形式排列的，无法直接使用，需要写一个方法来将此种歌词切割成单独的歌词形式来显示出来：

- (void)creatLrc:(NSString \*)str{

NSArray \*array = [str componentsSeparatedByString:@"\n"];

//遍历数组。获取每一行具体内容

for (int i = 0; i < [array count]; i++) {

NSString \*lineString = [array objectAtIndex:i];

NSArray \*lineArray= [lineString componentsSeparatedByString:@"]"];

if ([lineArray[0] /\*长度不过的作忽略处理\*/length] > 8) {

/\*获取逗号和点，从而为接下来分割歌词做准备\*/ NSString\*str1=[lineStringsubstringWithRange:NSMakeRange(3, 1)];

NSString \*str2 = [lineString substringWithRange:NSMakeRange(6, 1)];

if ([str1 isEqualToString:@":"] && [str2 isEqualToString:@"."]) {

for (int i = 0; i < lineArray.count - 1; i++) {

NSString \*lrcString = [lineArray objectAtIndex:lineArray.count - 1];

//分割区间求歌词时间

NSString \*timeString = [[lineArray objectAtIndex:i] substringWithRange:NSMakeRange(1, 5)];

//把时间 和 歌词 加入已经定义好的两个数组中

[self.lrcTime addObject:timeString];

[self.lrcAll addObject:lrcString];

}

}

}

}

}

1. 分别自定义三个点击按钮（上一曲，下一曲，暂停/播放）和一个进度条，需要将它们分别添加到self.view上，同时每个都需要添加点击事件：//    给按钮添加事件  
   //    播放  
       [self.btn3 addTarget:self action:@selector(go) forControlEvents:UIControlEventTouchUpInside];

/\*下一曲\*/  
    [btn4 addTarget:self action:@selector(next) forControlEvents:UIControlEventTouchUpInside];

/\*上一曲\*/  
    [btn2 addTarget:self action:@selector(back) forControlEvents:UIControlEventTouchUpInside];  
    [btn1 addTarget:self action:@selector(too) forControlEvents:UIControlEventTouchUpInside];  
    /\*滑块移动\*/  
    self.slider = [[UISlider alloc]initWithFrame:CGRectMake(120, 90, 280, 20)];  
      
    self.slider.thumbTintColor = [UIColor blackColor];  
      
    self.slider.maximumValue = self.player.duration;

[self.slider addTarget:self action:@selector(slider:) forControlEvents:UIControlEventTouchUpInside];

/\*分别添加到父视图中\*/

[view addSubview:btn1];

[view addSubview:btn2];

[view addSubview:self.btn3];

[view addSubview:btn4];

[view addSubview:self.slider];

如图4.1.3所示：



图4.1.3 网络歌曲界面

1. 歌曲播放界面

歌曲播放界面主要涉及到具体的有音量调节，每次可以通过点击增加或者减小音量按钮，并且伴随着音量调节提示，歌词控制按钮，则可以让用户自己去控制歌曲播放界面是否显示歌词，头像旋转是通过一定的方法和布局来共同实现的。紧接着下边是一个tableView来跟踪同步显示歌词进度，最下边则是我们熟悉的一些控制按钮，用户可以通过点选来自主选择。如图4.1.4所示



图4.1.4 歌曲播放界面

1. 用户个人中心界面

这个界面主要用来展示用户的头像，个性昵称和个性签名，作者寄语，则是希望用户每次在这个界面看到“美好的一天，从聆听开始”之后能够心情愉悦，听歌很多时候的确是一种陶冶情操，放松心情的娱乐活动。紧接着是用户意见反馈的入口，下边的版本信息则向用户直观地展示了此软件为发布第一个版本的新产品。如图4.1.5所示：



图4.1.5 个人中心界面

1. 修改个人信息界面

此界面主要提供用户去自定义自己喜欢的个性昵称和个性签名，通过数据持久化技术，可以实现用户什么时候进入个人中心都能看到自己的昵称和签名，不过暂不支持自定义头像，如图4.1.5所示



图4.1.6 用户自定义界面

1. 反馈意见界面

此界面是用户在点击意见反馈这一行的时候，屏幕会以压栈的方式弹出填写反馈信息的界面，本人十分希望用户在使用了本软件之后，能够提出宝贵的意见和建议，这也是本人在以后的版本升级，甚至未来从事其他软件开发过程中的动力与鼓舞。具体到代码层面的话，这一界面相对也比较简单，在控制器中添加了需要用到的控件之后，编写了少量的提示代码，以及意见框是否空白的用户提示：

- (IBAction)btnClick:(UIButton \*)sender {  
    if (\_textView.text.length == 0) {  
        [self showAlert:@"您没有填写意见哦^\_^"];  
    }  
        \_textView.text = nil;  
        [self showAlert:@"感谢您的意见！"];  
      
}



图4.1.7（1） 意见未写反馈界面



图4.2.7（2）用户提出意见界面

4.2应用控件组件介绍

在整个软件开发过程中需要用到大量的控件来完成各功能的基本实现和协调作用首先需要介绍的是很基础也很重要的视图控制器：ViewController，顾名思义，能够控制并且管理每个视图，在所有的应用中，我们所能看到的，都是视图（view）或者它的子视图，作为控制管理的view的控制器，它有着很重要的一个作用：可以在一个视图控制器中容纳多个视图，而在一个应用中，往往会有很多viewController类，且会包含大量的代码，在创建view过程中，可以有三种方式，分别是：storyBoard，xib，代码方式，相比于代码方式来说，前两种控件方式会快捷很多，但是后者却因为是自定义的，所以功能很强大，在上边添加子视图和布局姿势图的时候更加自由。

另外一个需要介绍的便是可以让用户点击的按钮：Button。相对于整个软件中所用到的控件，Button的使用频率十分高。从软件首页选择开始，到音乐播放过程中任意切换歌曲，控制歌词显示与否，再到个人中心界面修改个性签名等，足以说明这个控件的重要性。同样的，Button按钮可以直接拖拽使用，也可以自定义使用，在用户使用软件的人机交互过程中，Button起着举足轻重的作用， 通过点击Button按钮，可以很轻松很直观的达到我们想要的结果与界面位置，这一点是众多控件所无法替代的。

4.3整体架构设计

本软件整体采用MVC设计模式来开发。MVC是一种很常规也很实用的软件设计模式。模型M（Model），视图V（View），控制器C（Controller），三者结合起来并称MVC。Model通常继承于NSObject类，用来存放需要用到的具体事物模型，View则比较灵活，通常用于存放视图类，一些自定义的xib文件和系统默认的main.storyBoard等，Controller则主要用来存放软件开发过程中需要使用到的重要代码，比如逻辑的实现等。

MVC模式优越之处就在于可以将对象与事物分开来处理，避免了代码的大量冗余与耦合。模型能够给视图传输各类数据，视图则是各种类的对象，而所搭界面的显示则是由控制类来完成的，相对来讲各个模块相互独立互不影响。如图4.3所示：

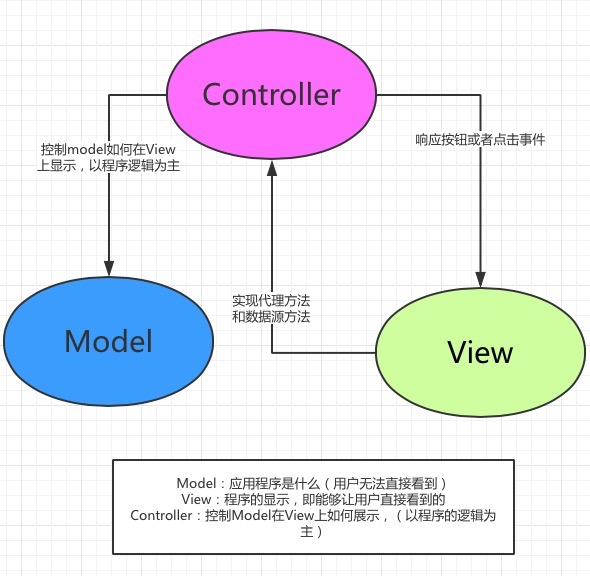


图4.3 MVC模式图

5 问题与解决方案

5.1歌曲播放和歌词同步问题

由于自己刚学习UI时间不长，对很多地方的了解和掌握还很肤浅不够深入，项目开始初期，在确立了整个框架之后，首先需要实施的是本地歌曲播放功能，在此基础上需要让歌词同步显示，此时问题来了，在我看来并没有问题的时候，歌词却如何都显示不出来。

一开始我的想法是plist文件中是否存在问题，仔细翻阅之后发现，无论是字典转模型还是属性传值都是没有问题的，当再次翻阅plist文件的时候，我发现，在传值的过程中，plist文件中存放的只是具体文件的名称，而文件的具体位置还需要在去赋值接收才可以。想到这里，我又翻阅了以前的一些笔记，同时和同学进行交流讨论后发现， 在调用歌词时需要将文件所在路径提取出来，然后通过方法 ：

//获取播放的mode

SongMode\*mode=[SongListHandle shareHandel].songArr[currentIndex];

//歌词解析

\_lrcMode=[ParLrcHandel initWithPath:[[NSBundle mainBundle] URLForResource:mode.kName withExtension:@"lrc"]];

从而处理了问题。

5.2数据持久化问题

所谓数据持久化，简单一点来讲就是将内存中的数据模型转化成存储模型，在将来需要的时候再将存储模型转化成数据模型。通俗地讲就是将数据保存在不易丢失的设备中，并在需要的时候将数据还原。在音乐播放器中主要应用于个人界面中个性签名和个性昵称的使用，这一点也困扰了我很长时间。

一开始才去的是Block块传值回调的方式来处理。通过用户选择进入编辑界面，修改完成之后点击保存，在segue回调的过程中，会将用户修改后的值传递过来显示在个人中心界面，然而在系统退出再次重新登录的时候却并没有保存用户上一次更改后的数据，而是系统默认数据。在网上查阅了很多资料了解到数据持久化可以有很多种方式，比如plist文件，比如数据库方式等都可以。在进行一番修改和尝试之后，最终实现了这一功能。从而解决了问题。

6 软件性能测试

6.1软件测试目的

软件测试，是软件开发过程中的一个重要环节，是贯穿整个软件生命周期，对软件产品（包括阶段性产品）进行验证过和确认的过程。其主要目的是尽可能全面的去发现软件产品中所存在的各种问题，比如软件存在的BUG或者与用户需求，预先想法不一致等，从而交由软件开发人员去及时纠正错误。项目测试的目的不仅仅是为了发现软件存在的缺陷和问题，也是为整个软件的整体质量水平进行度量和评估，软件开发的过程可能需要面对很多纷繁复杂的问题，同时每个卡发阶段都会不可避免的产生一些错误和问题。所以开发人员需要在每个阶段完成之前进行严格的技术审查并尽可能早的发现并改纠正错误，以期达到比较良好的效果。

6.2软件测试方法

软件测试中最常用的方法就是白盒测试和黑盒测试。所谓白盒测试即对软件的相关代码进行比较详细的测试，比如代码的正确性，书写规则等是否正确。而黑盒测试则是根据软件的规格对软件进行的测试，与白盒测试相反，给和测试不需要考虑软件内部的组成结构，只需要对软件整体测试即可，黑盒测试更多的是需要软件测试人员从用户的角度出发进行各类功能的测试，同时需要实时记录软件中暴露出来的各类错误和缺陷，黑盒测试最关心的便是软件是否能够满足用户的需求和各种需求是如何实现的，并不关心软件设计过程。

对音乐播放器的测试主要以下几个方面进行：

1. 代码测试：以Object-C为基础语言所编写的音乐播放器，可以分别从正常数据，异常数据，正常与错误交叉使用的方式来进行测试。
2. 功能测试：可以对音乐播放器进行大量重复相同的操作，也可以进行一些基本的操作，来测试软件是否能够完成基本功能，比如在播放音乐的过程中点击别的控件是否会影响到当前音乐的播放，比如不断地点击暂停/开始按钮，查看是否会出现顿卡甚至崩溃现象等等。

3.部分与整体测试：一般需要贯穿整个测试周期中，从一开始的功能模块，到最后的软件整体功能模块，在这个过程中尝试对每个模块进行适度的测试和修改，最后再将各种功能集中在整个软件上进行综合测试。

7 总结

为期两个月的毕业设计马上要告一段落了，作为自选的“基于iOS的音乐播放器”的基本功能也都实现了，在此期间，自己也深切感受到了一款应用的开发是一项分厂复杂的工程，平时自己也经常使用手机中的各类软件，但从来没有想象过每一款成型的软件背后都是经历了怎样的设计过程。现在回想，软件开发初期整体框架应该是最重要的，合理的框架是逐步实现各项功能的基础。同时作为一名合格的软件开发者，头脑中要有清晰的逻辑科学的方法合理的思路，这一点也是很重要的。

经过此次毕业设计让我收获很多，不仅是在平时所学的知识上，还有在做项目的时候从导师那里学到的优秀的品质和严谨的态度，都为以后的工作和学习打下了深深地基础。在编程语言上，我学习了全新的语言Object-C，虽说与以前学习过的C语言，C++有许多相似之处，但是自己也学到了很多新的东西，明白了委托（代理），属性，协议等等全新的概念，同时也了解了内存的分配与管理，以及对内存的手动释放和自动释放（ARC），这些东西在软件的开发过程中都有很重要的作用。在基础语言的学习之后，又了解并学习了iOS如何去构建基础框架，主要以Foundation和UIKit框架中一些类的原理和作用，比如UIView，UITableView，UIViewController等，以及程序是如何开始运行的，运行的具体流程是怎样的，程序中所指用到的委托代理方法是何时使用，以及属性，block块的使用等，还有就是在storyBoard中如何去拖控件，并设置属性来合理使用控件等等。在软件设计过程中，本人在音乐播放，歌词同步，网络歌曲播放等方面花费了较大的时间和精力，甚至在设计过程中推翻自己很多次，并最终在导师的指导下，确定了如今的方案。

由于自己知识有限，开发经验不足，在代码的编写过程中出现了很多的代码冗余现象，设计过程中也遇见很多错误。通过此次设计，我深切地认识到，在软件开发过程中一定要有很清晰的思路，尽可能去简化设计，减少代码量，同时也明确了自己在知识学习中的不足，更加明确了自己接下来努力的方向。

8 致谢

光阴似箭，时光飞逝，转眼间，为期两个月的毕业设计要告一段落了，猛然间想到自己的大学生涯也即将结束了，回首望向那已经流失于指尖的校园生活，不禁心中万分感叹。

大学以前，从来没有感受到一个学期可以过得这么快，仿佛昨天才拎着行李踏进校园的大门，今天就要挥手告别了。在这转瞬即逝的大学生活里，首先要感谢在毕业设计中给予我很大帮助的张芳芳老师，在做毕业设计的过程中，张老师仔细而又有耐心地对我的设计以及任务进行合理的指导，并且在后边的设计说明书中做了很全面合理的修改批注，竭尽全力的帮助我完成此次毕业设计任务。在这里，中心的表示对张老师耐心指导和悉心关怀的无限感谢和感恩。更重要的是，在做毕业设计期间，张老师所具备的严谨认真的务实精神，也成为我以后学习和工作中的榜样。同样需要感谢的还有这四年的时间里，传授我知识，引导我思考，激发我的思维，教会我人生智慧的众多老师。他们在这短暂的时间里用他们广博的知识与智慧，用狂光的胸襟与丰富的阅历，教导我去学习知识掌握技能。他们不仅是我的老师，更是我人生的朋友，我相信他们会为我的成功而感到自豪。他们一次次用温暖的力量带领我去成长，去释放青春的活力与激情。大学期间，有这么一群人，他们与我朝夕相处，嬉笑怒骂----我的同学，犹记得大家的音容笑貌，还记得初次见面时那一张张青涩的面孔，我们一起走过了青春，走向了成熟，走向了明天。这不仅是我人生最美好的时候更是我人生最宝贵的财富。最初的而我们来自五湖四海，不同的人生经历，不一样的性格特点，与那广泛的兴趣爱好使我们彼此吸引，相互交织融入在一起，我们怀揣着各自的人生梦想汇聚这里，一起欢笑一起忧伤，我们会在球场上尽情挥洒青春的函数，也会在深夜熄灯之后轻声讨论。这些美好的往事如烟云般不断浮现脑海，也随着大学生涯的结束而埋藏在心底。我们即将各自踏上离开的列车，在此我也衷心的祝福我的好同学，好朋友们：一路顺风。

9 参考文献

[1] [美] Dave Mark，Jack Nutting，Jeff LaMarche iOS 5基础教程.人民邮电出版社.2012

[2] [美] Stephen G.Kochan. Objective-C 程序设计[M].电子工业出版社.2012

[3] [美] Dave Mark,Jack Nutting,Jeff LaMarche. iOS基础教程[M].人民邮电出版社.2012

[4] 金福生，李朴之. iOS应用程序开发方法与实践[M]．北京: 人民邮电出版社，2012

[5] Vandad Nahavandipoor [iOS 6 Programming Cookbook](http://detail.tmall.com/venus/spu_detail.htm?spu_id=219556886&no_switch=1&default_item_id=18530370919" \t "_blank)

[6] Klehr．Objective-C 2 Mac and iOS development of practice guidelines[M]．Wiley Publishing，2012

[7] Mcleod.Printice Hall．Management Information System[M]．2008

[8] Brian X. Chen．Apple mania：iPhone open the always-on age Wiley Publishing 2012

[9] Carlo Chung．Objective-C programming way [M]．Wiley Publishing，2011

[10] Jiva DeVoe．Good knowledge of Objective-C[M]．Wiley Publishing，2012