|  |  |
| --- | --- |
| 231 | 123 |
| **PINGDINGSHAN UNIVERSITY** |

**毕业论文(设计)**

**题 目:** 基于Android与J2EE技术的

答题软件的设计与实现

**院(系):** 软件学院

**专业年级:** 软件工程 2014级

**姓 名:** 马子超

**学 号:** 141530436

**指导教师:**  张国平 副教授

**2018年1月4日**

原 创 性 声 明

本人郑重声明：本人所呈交的毕业论文，是在指导老师的指导下独立进行研究所取得的成果。毕业论文中凡引用他人已经发表或未发表的成果、数据、观点等，均已明确注明出处。除文中已经注明引用的内容外，不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究成果做出重要贡献的个人和集体，均已在文中以明确方式标明。

本声明的法律责任由本人承担。

论文作者签名： 日 期：

关于毕业论文使用授权的声明

本人在指导老师指导下所完成的论文及相关的资料（包括图纸、试验记录、原始数据、实物照片、图片、录音带、设计手稿等），知识产权归属平顶山学院。本人完全了解平顶山学院有关保存、使用毕业论文的规定，同意学校保存或向国家有关部门或机构送交论文的纸质版和电子版，允许论文被查阅和借阅；本人授权平顶山学院可以将本毕业论文的全部或部分内容编入有关数据库进行检索，可以采用任何复制手段保存和汇编本毕业论文。如果发表相关成果，一定征得指导教师同意，且第一署名单位为平顶山学院。本人离校后使用毕业论文或与该论文直接相关的学术论文或成果时，第一署名单位仍然为平顶山学院。

论文作者签名： 日 期：

指导老师签名： 日 期：

基于Android与J2EE技术的答题软件的设计与实现

摘 要

各个高校每逢学期末期末考试来临之际，学生们复习都是手忙脚乱并且存在以下问题：每个科目都有大量的复习资料，打印后不便于携带。也有些同学去书店买试题来复习，但是同一套试题作答一次会留下痕迹，不利于再次复习。并且同学们需要很早去图书馆或者教室抢占座位。针对以上问题，开发一款不受时间地点限制，方便高校学生期末复习的答题软件，具有重要的现实意义。

基于Android与J2EE技术的答题系统主要由两套系统构成，对于手机端使用AndroidDevTools开发的手机Android答题软件，对于题库的维护使用Eclipse集成J2EE开发技术的Web端后台管理。对于手机端的开发本项目使用了较为实用的Xutils3框架，可以更加轻便的管理应用。使用框架开发软件对于二次开发非常友好。Web后台管理端使用Maven项目管理，整体项目框架为SpringBoot集成，内部包含Spring框架，SpringData+JPA框架，SpringMVC框架。对于后台项目的搭建，采用目前最流行的Spring体系。使得开发者能更加快速的完成项目开发。

答题系统主要分为以下模块：答题模块，错题模块，公告模块，自动更新模块。管理系统主要分为以下模块：登录模块，题库扫描模块，题库更换模块，题目查看模块。管理员负责维护题库，用户使用手机软件进行答题复习。

如今几乎每个人都有自己的手机，而高效的复习手段更能获得青睐。普通的复习方式是将题目打印出来，使用纸质文档进行复习，题目做一次就会留下答案痕迹，不利于再次复习。而在手机上直接答题就特别方便。并且可以多次复习，加上软件可以自动更新，能大大延长软件的生命周期。

**关键词**：答题软件，Android，J2EE，SpringBoot

Design and Implementation of Answer Software Based on Android and J2EE Technology

Abstract

At the end of each semester, examinations are approaching at the end of each semester. The students' reviews are confused and have the following problems: Each subject has a large amount of reviewing materials and is not easily transportable after printing. Some students go to the bookstore to buy questions to review, but the same set of questions will leave a trace of the answer is not conducive to review again. And students need to go to the library or classroom occupy seats very early. In view of the above problems, it is of great practical significance to develop an answer software which is not limited by time and place and convenient for university students to review at the end of the period.

Answer system based on Android and J2EE technology mainly consists of two sets of systems, for the mobile terminal using AndroidDevTools developed mobile answering software for Android, for the maintenance of the exam using Eclipse integrated J2EE development technology Web-based background management. For the development of mobile terminal This project uses a more practical Xutils3 framework, you can more easily manage applications. Framework development software is very friendly to secondary development. The Web backend uses Maven project management. The overall project framework is SpringBoot integration, which includes Spring framework, SpringData + JPA framework and SpringMVC framework. For the background project to build, using the most popular Spring system. Enable developers to complete project development more quickly.

Answer system is divided into the following modules: answer module, the wrong module, bulletin module, automatic update module. Management system is divided into the following modules: login module, the exam bank scanning module, the exam bank replacement module, the title view module. The administrator is responsible for maintaining the exam bank, the user uses the mobile software to answer the questions.

Now almost everyone has their own cell phone, and effective review means more to gain favor. Ordinary review method is to print out the title, the use of paper documents for review, the subject will leave a trace of the answer is not conducive to review again. Direct answer on the phone is particularly convenient. And can review many times, together with the software can be automatically updated, can greatly extend the software life cycle.

**Keywords:** Answer Software, Android, J2EE, SpringBoot

目 录

[1 绪论 1](#_Toc502829278)

[1.1 项目开发背景及意义 1](#_Toc502829279)

[1.2 项目开发现状 1](#_Toc502829280)

[1.3 论文主要工作和结构安排 2](#_Toc502829281)

[1.4 本章小结 2](#_Toc502829282)

[2 相关技术及工具介绍 3](#_Toc502829283)

[2.1 Android端使用的技术点 3](#_Toc502829284)

[2.1.1 XUtils3 3](#_Toc502829285)

[2.1.2 Gson 3](#_Toc502829286)

[2.2 Web端使用的技术点 4](#_Toc502829287)

[2.2.1 SpringBoot 4](#_Toc502829288)

[2.2.2 Apache POI 4](#_Toc502829289)

[2.3 本章小结 4](#_Toc502829290)

[3 需求分析 5](#_Toc502829291)

[3.1 业务操作流程 5](#_Toc502829292)

[3.2 系统设计目标 6](#_Toc502829293)

[3.3 功能需求 6](#_Toc502829294)

[3.3.1 Web服务器端 6](#_Toc502829295)

[3.3.2 Android手机端 10](#_Toc502829296)

[3.4 性能需求 13](#_Toc502829297)

[3.5 本章小结 14](#_Toc502829298)

[4 总体设计 15](#_Toc502829299)

[4.1 系统结构设计 15](#_Toc502829300)

[4.2 系统功能设计 16](#_Toc502829301)

[4.2.1 Web端后台管理功能的设计 17](#_Toc502829302)

[4.2.2 Android手机端功能的设计 17](#_Toc502829303)

[4.3页面设计 18](#_Toc502829304)

[4.3.1 Web端后台管理的页面设计 18](#_Toc502829305)

[4.3.2 Android手机端的页面设计 19](#_Toc502829306)

[4.4 数据库设计 20](#_Toc502829307)

[4.4.1 数据库概念结构设计 20](#_Toc502829308)

[4.4.2 数据库逻辑结构设计 21](#_Toc502829309)

[4.4.3 数据库物理结构设计 21](#_Toc502829310)

[4.5 本章小结 22](#_Toc502829311)

[5 详细设计与实现 23](#_Toc502829312)

[5.1 后台Web端功能设计 23](#_Toc502829313)

[5.1.1 题库管理模块设计 23](#_Toc502829314)

[5.1.2 公告栏管理模块设计 24](#_Toc502829315)

[5.1.3 手机端版本管理模块设计 24](#_Toc502829316)

[5.2 手机Android端功能设计 25](#_Toc502829317)

[5.2.1 自动更新模块设计 25](#_Toc502829318)

[5.2.2 答题模块设计 26](#_Toc502829319)

[5.2.3 背题功能模块设计 27](#_Toc502829320)

[5.3 后台Web端系统类的设计 28](#_Toc502829321)

[5.4 手机Android端系统类的设计 29](#_Toc502829322)

[5.5 系统部分功能的实现 31](#_Toc502829323)

[5.5.1 题目的解析 31](#_Toc502829324)

[5.5.2 手机端题目的构造 32](#_Toc502829325)

[5.5.3 自动更新的处理 34](#_Toc502829326)

[5.5.4 手机端应用数据缓存 35](#_Toc502829327)

[5.5.5 答题功能 36](#_Toc502829328)

[5.5.6 题库管理 37](#_Toc502829329)

[5.6 关键技术使用说明 38](#_Toc502829330)

[5.7 本章小结 40](#_Toc502829331)

[6 测试 41](#_Toc502829332)

[6.1 测试的意义 41](#_Toc502829333)

[6.2 测试环境 41](#_Toc502829334)

[6.3 功能测试 41](#_Toc502829335)

[6.4 界面测试 43](#_Toc502829336)

[6.5 系统性能测试 44](#_Toc502829337)

[6.6 问题严重程度描述 44](#_Toc502829338)

[6.7 本章小结 44](#_Toc502829339)

[7 结束语 45](#_Toc502829340)

[7.1 总结 45](#_Toc502829341)

[7.2 展望 45](#_Toc502829342)

[附 录 47](#_Toc502829343)

[参考文献 51](#_Toc502829344)

[致 谢 52](#_Toc502829345)

1. 绪论

随着计算机技术的推进和发展，J2EE企业级框架的应用也越来越多。针对笔试考试中部分题目数量较多，但是有具体明确的知识点范围，所以开发出一款移动端的答题软件供同学们复习测试。在移动端的手机中，Android系统占据了半壁江山。所以采用Android系统作为开发使用的手机平台，后台使用J2EE企业级框架SpringBoot进行集成搭建起来的一款答题软件，能快速的适应市场和需求，让开发出软件的生命周期延长。本章主要介绍了项目开发的背景和意义，以及开发答题软件的主要研究工作，最后对论文的结构进行了介绍。

## 1.1 项目开发背景及意义

在当今信息时代，计算机技术与网络技术越来越广泛的应用于各个领域，改变人们的学习、工作、生活乃至思维方式，也引起了教育领域的重大变革。将计算机网络技术应用于现代高等教育中，是现代高等教育发展的需要，也是改革教育模式，提高学校教学效果和教学效率。高校中每学期的考试周的来临，如何去复习一些考试的题目成为学生们的主要目的，以往的复习方式是去书店购买各种学科复习题进行纸上答题，各个学科中都会有选择题和判断题，有些科目的选择题和判断题有规律可寻。比如说政治，历史等等的文科科目。对这些题目的进行大量做题加深印象，保证考试中能记住答案的思路。选择题和判断题的涉及的知识点多而杂，于是乎进行多次大量的重复记忆，是一种有效的复习手段。

手机等移动终端已经成为人们必不可少的随身携带品，特别是在4G网络问世后，手机上网更是受到了更多用户的喜爱和推崇，甚至于现在的多数人只要一离开手机就会显得无所适从。相较于电脑，手机等移动终端接入网络的方式更显得灵活，不受时空限制，随时随地的在手机上进行答题复习，对自己不会不懂的知识点查漏补缺，将复习时间碎片化，则可大大的提高复习效率，节省时间来复习剩余科目[1]。所以我准备开发出一款在手机上进行答题的软件。

## 1.2 项目开发现状

国内答题软件最出名的就类似于驾考宝典手机App那样，将驾校中科目一考试和科目四考试的题库进行整合，题库唯一。我们不能用驾考宝典手机App进行考试题目的复习。还有就是一些是网上在线考试系统，这些在线考试系统都需要一个教师机和一屋子学生机配合使用。虽然可以满足需求，但是我的目的是将碎片时间利用起来，如果每次复习都需要找到在线考试的教室，使用电脑进行答题，还不如去图书馆来的方便。

于是我知道了开发的方向。一是便捷，利用碎片时间，二是自定义的题库，可以测试自己想复习的题目。查询了网上对于此项目的研究方向，大致都是往在线考试方向研究，对试题的加密和考试中如何防止作弊，也满足不了我的需求。

## 1.3 论文主要工作和结构安排

本论文主要介绍了答题软件的开发背景、开发与实现的过程，项目实现中遇到的一些技术点和难点如何解决的思路和实现，数据传输和数据存储，系统测试。全文共七章，具体如下安排：

第一章 介绍项目开发背景、开发意义和现阶段开发现状，以及论文的结构安排等。

第二章 介绍该系统使用的相关技术和开发工具还有项目运行环境等。

第三章 对项目功能的需求分析，和项目的性能的需求分析。

第四章 项目的总体设计和各个模块的设计与实现，和数据传输和存储设计与实现。

第五章 详细介绍项目中具体的类如何实现该功能。

第六章 介绍本项目的各种测试及分析测试结果。

第七章 总结本次项目的不足之处和对项目未来的分析与展望。

## 1.4 本章小结

本章首先介绍了本项目开发的背景和意义接着分析了现阶段的开发现状，并且介绍了本次论文的整体结构，以便能按计划顺利完成本次毕业设计。

1. 相关技术及工具介绍

系统整体分两大块，Android端和后台数据Web端。使用MySQL数据库进行数据存储，使用Tomcat作为后台Web服务器。开发工具使用Eclipse，Web后台端使用Eclipse Java EE版本，Android端使用Android Developer Tools工具进行开发。

2.1 Android端使用的技术点

Android端中采用分层开发，使用全局的工具类进行整合，使用XUtils3框架代理整个Android应用，使用Gson作为数据解析工具，Toast为提示工具，SharedPreferences作为信息的存储和获取。

页面中使用ViewPager对题目进行轮循展示，使用ViewPagerAdapter对每个页面进行构造。使用ViewHolder对数据进行缓存加速。个别技术解释如下：

### 2.1.1 XUtils3

XUtils3一共有4大功能：注解模块，网络模块，图片加载模块，数据库模块。

注解模块可以说是手机端的ORM框架，在布局文件中为各种空间可以直接通过注解进行关联关系，不再使用大量的findViewById()方法进行控件的加载。

网络模块可以更加简便的使用HTTP协议，通常Android访问网络的代码需要在子线程中进行，然后使用Handler监听返回结果。使用起来特别麻烦。如果使用XUtils3的网络模块，那就只需要对访问请求的CallBack函数中进行处理就可以获取数据[2]。图片加载模块在本项目中未使用，所以不做过多的介绍。

数据库模块也是一个ORM框架，将SQLite数据库的表格与实体类进行关联。能更好的对数据进行维护。

### 2.1.2 Gson

Gson是由Google提供的一款数据解析工具，主要是对JSON格式的数据进行相互转换。使用过程中只需要编写与JSON格式对应的实体类，通过Gson进行数据转化后，实体类中的属性就会被填充，大大降低了对JSON数据格式的解析。

2.2 Web端使用的技术点

Web端使用SpringBoot快速搭建，使用Maven对项目中的Jar包进行管理控制。集成了J2EE开发框架：Spring + Spring MVC + Spring Data JPA。后台对数据的处理和提供相应的访问接口。

数据解析工具使用Apache POI对题库Word文档进行解析，使用正则表达式对文档中的每行数据进行分析，然后将数据进行规整存储。个别技术解释如下：

### 2.2.1 SpringBoot

SpringBoot是将Spring应用更加轻量化，所有的Spring的应用开箱即用，提供很多默认配置来简化项目配置[3]。有内嵌式Tomcat服务器，没有大量的XML的冗余配置。本项目使用SpringBoot集成了SSSP(Spring,SpringMVC,SpringData,

JPA)框架。项目中多使用注解开发。

### 2.2.2 Apache POI

Apache POI是对Microsoft Office格式的文档进行读写的框架。主要用于对题库的解析，配合正则表达式完成数据的规整化。

2.3 本章小结

本章主要介绍了此项目中所用到的相关技术，包括Android端采用的XUtils3框架和Gson解析数据工具，以及Web端使用的SpringBoot集成框架的使用，Apache POI文档解析的使用。介绍了开发工具Eclipse和Android Developer Tools的使用。

1. 需求分析

需求分析主要描述了用户对软件应用的期望和使用需求，了解用户的需求后才能使软件开发有一个明确的方向，根据用户需求设计应用的功能。

3.1 业务操作流程

用户使用手机端打开答题软件App，软件进入引导页面，用户可查看当前公告和检测版本是否需要升级。如果需要升级则自动下载最新安装包进行覆盖安装。再次打开。用户可选择进行答题或者复习错题，在答题过程中如果打错则系统自动将本题加入到错题集中，可在进入页面选择复习错题来打开错题库。在错题库中如果回答正确则将本题目移出错题库。在答题的过程中可选择模拟测试和查看所有答案的背题模式。如果开启背题模式，所有题目显示出答案，用户可以进行背题、学习。

后台是管理员使用的，需要将题库放到指定的文件夹中，系统启动的时候自动扫描改文件夹下的Word文档，将该文档中的所有题库存入到数据库中。以便手机端App访问，后台使用J2EE技术提供数据访问接口，将数据库中的数据通过转化传递到手机端。然后手机端进行数据解析就可以获取到所有的题目了。根据业务操作流程的描述，可以画出系统流程图。系统流程图如下图3-1所示。



图3-1 系统流程图

3.2 系统设计目标

基于Android与J2EE技术的答题软件从模块上来说，分为两大模块。一个是题库后台数据提供模块，也就是通过J2EE技术实现的Web端，另一块就是Android端使用模块。具体的功能期望实现如下目标：

* + - 1. 用户可以学习、复习指定的题库。
      2. 用户可以了解最新版本公告，和动态。
      3. 用户使用的手机端可以提示自动更新最新版本。
      4. 管理员可以发布新的公告和发布新的App版本。
      5. 管理员可以简便的切换题库供所有用户使用。

3.3 功能需求

本系统的使用者有两种，分别是用户和管理员。系统主要面向用户开发，管理员只负责维护一些数据。系统的总用例图如图3-2所示。



图3-2 系统总体用例图

通过以上的项目需求分析，拟开发的答题软件应该有Web端模块和Android端模块，每个模块具备的功能如下：

### 3.3.1 Web服务器端

1. 更换题库。管理员可以通过上传不同的Word文档进行更换用户查看的题库。
2. 发布公告。管理员可以修改公告文件，进行发布一些提示性的公告。
3. 发布新版本。当管理员需要用户App版本更新的时候，可以发布新版本提示用户更新。

管理员的Web端模块用例如图3-3所示。



图3-3 Web模块用例图

管理员Web端模块的用例描述如表3-1至3-5所示。

表3-1更换题库用例描述

|  |  |
| --- | --- |
| **用例名称** | 更换题库 |
| **简要描述** | 对题库的维护操作 |
| **参与者** | 管理员 |
| **涉众** | 用户 |
| **前置条件** | 后台Web服务器开启 |
| **后置条件** | 后台Web服务器开启，题库已更换 |
| **基本事件流**   1. 进入题库管理页面 | |
| 1. 选择需要启动的题库进行设置 2. 启用题库，题库更换完毕用例结束，更换失败转(A-1) | |
| **备选事件流**  A-\* 用户在更换题库时可以随时终止该用例  A-1 更换失败后可以继续点击更换转(3) | |
| **补充约束-非功能需求**  题库：同一时间只能有一个题库为启用状态 | |

表3-2发布公告用例描述

|  |  |
| --- | --- |
| **用例名称** | 发布公告 |
| **简要描述** | 管理员发布新的公告 |
| **参与者** | 管理员 |
| **涉众** | 用户 |
| **前置条件** | 管理员打开配置文件 |
| **后置条件** | 修改成功后即可 |
| **基本事件流**   1. 在配置文件中找到公告字段 2. 修改公告内容 3. 重新访问公告栏，用例结束，修改失败转(A-1) | |
| **备选事件流**  A-\* 用户在修改公告时可以随时终止该用例  A-1 刷新服务器缓存后重启后台程序再次修改转(2) | |
| **补充约束-非功能需求**  配置文件：整体的更新配置文件在后台项目中，直接修改即可更新 | |

表3-3发布新版本用例描述

|  |  |
| --- | --- |
| **用例名称** | 发布新版本 |
| **简要描述** | 管理员需要让用户更新App版本时发布新版本 |
| **参与者** | 管理员 |
| **涉众** | 用户 |
| **前置条件** | 管理员打开配置文件 |
| **后置条件** | 修改成功即可 |
| **基本事件流**   1. 将打包好的App文件放到指定位置 2. 在配置文件中找到版本字段 3. 修改指定的版本url和修改指定版本号 4. 保存信息，用例结束，修改失败转(A-1) | |
| **备选事件流**  A-\* 用户在发布新版本时可以随时终止该用例  A-1刷新服务器缓存后重启后台程序再次修改转(3) | |

续表3-3

|  |
| --- |
| **补充约束-非功能需求**  配置文件：整体的更新配置文件在后台项目中，直接修改即可更新 |

表3-4题库扫描用例描述

|  |  |
| --- | --- |
| **用例名称** | 题库扫描 |
| **简要描述** | 管理员上传题库后将题库添加到系统中 |
| **参与者** | 管理员 |
| **涉众** | 用户 |
| **前置条件** | 登陆后台管理页面，打开题库管理 |
| **后置条件** | 成功扫描文件夹中的题库信息 |
| **基本事件流**   1. 将需要扫描的题库文档放入指定文件夹中 2. 打开题库管理页面 3. 点击扫描题库按钮 4. 题库扫描后启用某一题库，用例结束 | |
| **备选事件流**  A-\* 用户在题库扫描时可以随时终止该用例 | |
| **补充约束-非功能需求**  指定文件夹：指定路径存放到配置文件中 | |

表3-5删除题库用例描述

|  |  |
| --- | --- |
| **用例名称** | 删除题库 |
| **简要描述** | 管理员管理题库，删除不需要的题库信息 |
| **参与者** | 管理员 |
| **涉众** | 用户 |
| **前置条件** | 登陆后台管理页面，打开题库管理 |
| **后置条件** | 成功删除指定题库 |
| **基本事件流**   1. 打开题库管理页面 2. 选择需要删除的题库，点击删除 3. 删除题库后更新题库列表，用例结束 | |
| **备选事件流**  A-\* 用户在删除题库时可以随时终止该用例 | |

续表3-5

|  |
| --- |
| **补充约束-非功能需求**  操作：删除操作尽量简化 |

### 3.3.2 Android手机端

1. App提示自动更新功能。用户每次打开应用时，系统会自动检测当前版本是否是最新版本，如果不是则提示自动更新。
2. 查看公告功能。登录后用户可查看当前系统的公告内容。
3. 背题功能。用户可以选择查看所有题目答案进行背题学习。
4. 答题功能。用户点击题库列表可选择进行答题或者错题复习。进入答题功能后显示的是每一道试题，可以滑动手机屏幕或者点击下一题进行查看题目。题目下方是选项区域，点击可以选择答案。选择答案后可以点击查案答案进行检查，如果回答错误则将此题添加到错题库中。
5. 错题复习功能。错题库中保存的是答题过程中所有回答错误的题目，如果再次作答的时候回答正确则此题移出错题库。

用户的Android模块用例如图3-4所示。



图3-4 Android模块用例图

用户Android模块的用例描述如表3-6至3-10所示。

表3-6答题功能用例描述

|  |  |
| --- | --- |
| **用例名称** | 答题功能 |
| **简要描述** | 用户登录手机端进行答题的用例描述 |
| **参与者** | 用户 |

续表3-6

|  |  |
| --- | --- |
| **涉众** | 用户 |
| **相关用例** | 题库答题、错题复习 |
| **前置条件** | 查看题库后进行答题 |
| **后置条件** | 答题完成，错题进入错题库 |
| **基本事件流**   1. 选择希望进行答题的题库 2. 选择开始答题，进入答题页面 3. 阅读题目，选择正确答案，提交提交本次答题结果 4. 系统给出答案，如果答题错误转(A-1) 5. 答题成功跳转下一题继续答题 | |
| **备选事件流**  A-\* 用户在答题时可以随时终止该用例  A-1 提示回答错误，将此题添加到错题库中 | |
| **补充约束-非功能需求**  题库：所有题库通过联网获取，第一次开启软件需要联网 | |

表3-7 App更新版本用例描述

|  |  |
| --- | --- |
| **用例名称** | 版本更新 |
| **简要描述** | 用户手机端版本更新 |
| **参与者** | 用户 |
| **涉众** | 用户 |
| **相关用例** | 题库答题 |
| **前置条件** | 打开手机答题软件 |
| **后置条件** | 成功更新版本，覆盖安装新版本 |
| **基本事件流**   1. 打开手机答题软件，系统联网查询是否是最新版本 2. 如果有新版本，则提示自动更新，取消更新后转(A-1) 3. 点击更新按钮，系统下载更新包 4. 下载完成后出发覆盖安装程序，系统覆盖安装答题软件，用例结束 | |
| **备选事件流**  A-\* 用户在版本更新时可以随时终止该用例  A-1 取消更新，直接进入主页面 | |

续表3-7

|  |
| --- |
| **补充约束-非功能需求**  更新信息：更新信息通过联网获取，如需更新版本需要开启网络 |

表3-8 题库查看、背题功能用例描述

|  |  |
| --- | --- |
| **用例名称** | 背题功能 |
| **简要描述** | 题库预习和背题功能的开启 |
| **参与者** | 用户 |
| **涉众** | 用户 |
| **相关用例** | 题库答题 |
| **前置条件** | 打开手机答题软件 |
| **后置条件** | 显示所有题目答案 |
| **基本事件流**   1. 打开手机答题软件 2. 进入答题页面，用户可以选择是否开启背题模式 3. 开启背题模式后显示所有题目答案，用例结束 | |
| **备选事件流**  A-\* 用户在背题时可以随时终止该用例 | |
| **补充约束-非功能需求**  操作：用户使用时操作可以滑动屏幕控制题目切换 | |

表3-9 错题复习用例描述

|  |  |
| --- | --- |
| **用例名称** | 错题复习 |
| **简要描述** | 进入错题复习功能 |
| **参与者** | 用户 |
| **涉众** | 用户 |
| **相关用例** | 题库答题 |
| **前置条件** | 在答题系统中回答错误 |
| **后置条件** | 复习错题 |
| **基本事件流**   1. 打开手机答题软件 2. 进入答题页面 3. 回答错误的题目进入错题库 4. 回到主页面后进入错题库，进行错题复习 | |

续表3-9

|  |
| --- |
| 1. 错题复习时进行判断转(A-1)，用例结束 |
| **备选事件流**  A-\* 用户在背题时可以随时终止该用例 |
| **补充约束-非功能需求**  操作：用户使用时操作可以滑动屏幕控制题目切换 |

表3-10 查看公告用例描述

|  |  |
| --- | --- |
| **用例名称** | 查看公告 |
| **简要描述** | 用户使用答题软件查看系统公告栏 |
| **参与者** | 用户 |
| **涉众** | 用户 |
| **相关用例** | 无 |
| **前置条件** | 打开手机答题软件 |
| **后置条件** | 无 |
| **基本事件流**   1. 打开手机答题软件，系统联网获取最新的更新信息 2. 如果没有网络则获取本地缓存数据转(A-1) 3. 系统进行解析数据，获取最近的公告信息 4. 将公告信息展示到公告栏中 | |
| **备选事件流**  A-\* 用户在版本更新时可以随时终止该用例  A-1 读取缓存数据，替换网络获取数据进行解析 | |
| **补充约束-非功能需求**  更新信息：更新信息通过联网获取，如需更新版本需要开启网络 | |

3.4 性能需求

系统能高效长久的运行，答题系统需要满足以下的性能需求：

系统响应时间

系统应该具备快速的响应时间，题库的解析和数据的获取的处理时间应统一低于5秒。

系统可靠性

系统应具备可靠的稳定性，无论是Android端的使用和后台数据的提供都应该保持较稳定的体验。需要在所有环节都能正常运行。即使后台数据提供出现问题，Android端也可以访问本地缓存进行临时使用。

系统易用性

用户使用Android端整体布局简洁，灵活。用户很容易上手使用。

系统技术的成熟性与先进性

后台系统使用SpringBoot集成框架开发，能更加快速的对系统进行扩展。集成的SSSP后台J2EE框架也是先进的开发框架[4]。

3.5 本章小结

本章对答题软件系统各功能进行了分析，明确了开发目标。对整体的需求进行了分析和提取，并且画出了系统的用例图和用例描述。

1. 总体设计

系统设计主要的目的是将项目的需求进行转化，转化为用软件系统表示的一个过程。通过对答题软件系统的分析和研究，做出了一个总体的规划，这是一个项目进行全面开发的重要基础。

4.1 系统结构设计

后台的题库管理系统的框架采用SSSP(Spring,SpringMVC,SpringData,JPA)技术进行框架搭建，Spring利用切面编程和控制反转控制业务，SpringMVC使用一个过滤器调度程序作为控制器，SpringData+JPA主要负责将数据持久化。后台试题管理系统中用Spring对数据源和事务进行配置，通过切面编程使用环绕事务的方式处理方法调用，通过自动注入利用发射将实体对象分发给各个模块。用SpringData将试卷试题和用户的相关信息持久化到数据库中，SpringMVC对页面请求进行处理[5]。如图4-1所示。



图4-1 SSSP架构图

客户使用前台答题软件的Android端系统架构采用多个Activity控制手机展示页面，通过Intent传递数值进行手机页面的跳转，所有的数据通过后台传递，后台将SpringData的持久化类转化为JSON格式的数据流[6][7]，手机端通过HTTP请求将JSON获取，然后在手机端进行数据解析，解析的数据成为标准格式的试题。如图4-2所示。



图4-2 Android获取数据架构图

4.2 系统功能设计

系统中主要功能角色分为管理员和使用用户。管理员负责对题库的管理，对公告栏的修改和发布新的Android端版本。用户则使用Android端有自动更新功能，答题功能，错题查看功能，背题功能。系统功能模块图如图4-3所示。



图4-3 系统功能模块图

### 4.2.1 Web端后台管理功能的设计

答题软件系统采用先进的J2EE架构实现了后台的业务逻辑处理，使用SpringBoot集成了SSSP框架作为后台数据处理功能。使用Apache POI对文档进行解析[8]。然后将数据存入数据库以便访问。分为三大模块：题库管理，公告栏管理，手机端版本管理。

题库管理模块：该模块主要实现管理员指定题库后所有用户可以立即查看最新的题库，可以实时更新题库，主要依靠上传Word文档到指定文件位置，然后系统会自动扫描到文件，然后通过正则表达式对文档中的题进行解析，最后解析后添加到数据库中。手机端获取数据只需要访问数据库即可，不用每次访问重复扫描文档。

公告栏管理模块：该模块主要是管理员发布新版本后给用户的提示，比如说该版本修复了什么bug和新增加了什么功能。用户一打开应用程序即可看到公告栏，以后也可以插入广告等。

手机版本管理模块：该模块主要控制手机端的自动更新。修改响应版本号和与之对应的Apk安装文件，手机客户端即可自动下载。

### 4.2.2 Android手机端功能的设计

手机Android端采用XUtils3框架能更便捷的使用某些功能，比如说联网更新和注解注入，使用Gson解析获取到的JSON格式数据，传递到ViewPagerAdapter中构造题目页面。手机端分为五大模块：自动更新，答题功能，错题库功能，背题功能，查看公告栏。

自动更新模块：该模块放到了Android应用启动的首个Activity页面中，每次打开应用，都检测当前版本与服务器上最新版本是否一致。如果有新版本，则会提示更新，自动更新的会将服务器中的Apk文件进行下载，然后调用系统的安装包命令自动覆盖安装程序。

答题功能模块：该模块是整个应用的核心模块，题目数据从后台获取，然后将所有题目以缓存的形式存储到手机本地，以便下次快速加载，答题功能主要通过Android的ViewPager来切换题目，对于单项选题选项使用RadioButton控件来实现选择项，一次只能选择一个。对于多项选择题来说使用Checkbox控件实现多选，选择后点击下一题或者查看答案则进行判断是否回答正确，然后给出响应提示。如果回答错误则将本题添加到错题库中。

错题库功能模块：该模块是关联答题模块的，在答题模块中回答的错题则被添加到错题库中，在主页中选择题库时如果选择错题库的话则可以进行复习，在回答错题的过程中回答正确则将本题移出错题库。

背题功能模块：该模块主要是在答题的时候选择是否开启背题功能，如果开启后则显示所有题目答案，用户可以进行学习背题。

公告栏模块：该模块是在主页中，每次打开应用会自动更新加载最新公告内容，以提示用户使用。

4.3页面设计

页面设计主要目的是描绘出应用大概的页面结构，下面主要介绍了手机Android端的页面布局设计。

### 4.3.1 Web端后台管理的页面设计

整体页面设计。对于管理者来说需要一个总览，刚进入后台页面的总览页面，能展示出所有题库和题库信息。在管理页面中可以搜索题库和搜索题目。有表格显示所有数据。页面整体布局如图4-4所示。



图4-4 后台布局

主页面是通过左侧的页签选项点击打开的新页面，使用一个iframe展示所有页面，在右上角的页签操作栏可以对打开的页签进行管理。在个人信息中可以退出登录和更换皮肤等操作。而题库管理页面则显示一个具体的数据表格，上面有搜索栏用来对数据进行搜索。其中题库管理页签布局如图4-5所示。



图4-5 题库管理布局

### 4.3.2 Android手机端的页面设计

主页布局设计。其中的主要功能是用于显示版本号和展示公告，并且提供答题错题的入口。公告栏中显示最新获取的数据，点击错题库按钮是进入错题复习的入口按钮，点击后进入复习模式，答题库是进入开始答题的入口按钮，点击后就可以进行答题测试。页面布局如图4-6所示。



图4-6 主页布局

答题页面布局设计。其中的主要功能是用户题目的展示和答题，页面左上角有开启背题模式的开关，如果开启背题模式则显示所有题目答案，中间题目是用ViewPager包裹，可以识别手势滑动，也可点击切换上一题和下一题。当点击下一题或者点击提交答案的时候进行判断此题是否回答正确，如果回答错图则将此题添加到错题库中，错题库的答题页面与此页面相同只是传递的题库不一样。在错题库中回答正确则移出错题库。页面布局如图4-7所示。



图4-7 答题页面布局

4.4 数据库设计

数据库设计是整个答题软件的一部分，用来存储扫描文档后的数据，如果用户每次请求访问题库的时候都需要重新扫描一次Word文档将大大浪费时间，所以，每次题库只需要扫描一次，然后存入数据库。一个合理的数据库设计能大大提高软件实用性，能很明显的提高应用的性能。数据库设计分为两个方面：数据库分析和数据库的设计与实现。

4.4.1 数据库概念结构设计

将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计。将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计。

有需求分析得出系统使用数据库作为题库存储的一种方式，所以需要明细题库的信息，和优化题库的存储方式。一个文档为一个题库，题库中包含许许多多的试题，每一个试题的类型不同，有单项选择题和多项选择题。有题干和选项，最后还需要保存试题的正确答案。根据需求分析画出系统E-R图如图4-8所示。



图4-8 系统E-R图

图中各实体属性如下：

题库包括唯一标识、创建时间、题库名。

试题包括唯一标识、题目、对应类型、答案、对应题库。

试题类型包括唯一标识、类型名。

试题选项包括唯一标识、选项、对应试题。

实体类之间的对应关系有：一个题库有多个试题，一个试题属于一个试题类型，一个试题包含多个选项。

4.4.2 数据库逻辑结构设计

数逻辑结构设计的任务就是把概念结构设计阶段设计好的基本E-R图转换为与选用DBMS产品所支持的数据模型相符合的逻辑结构。

在数据库概念设计阶段我们已经得出了需要设计数据库的E-R图，下面就是把设计好的E-R图转换为对应的关系模式（主键用下划线着重标出）。

* + - 1. 题库关系模型

题库（编号，创建时间，题库名）

* + - 1. 试题关系模型

试题（编号，题目，类型编号，答案，题库编号）

* + - 1. 试题类型关系模型

试题类型（编号，试题类型）

* + - 1. 试题选项关系模型

试题选项（编号，选项，试题编号）

4.4.3 数据库物理结构设计

若要完成实现响应数据存储需要，在本数据库db\_answer中应有数据表question\_bank（题库）、question（试题）、type（试题类型）、option（试题选项）。

1.question\_bank（题库）信息表，如表4-1所示。

表4-1 题库信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 备注 | 是否允许空值 |
| id | int | 11 | 编号 | 否 |
| createtime | varchar | 255 | 创建时间 | 否 |
| name | varchar | 255 | 题库名 | 是 |

2.question（试题）信息表，如表4-2所示。

表4-2 试题信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 备注 | 是否允许空值 |
| id | int | 11 | 编号 | 否 |
| topic | varchar | 255 | 题目 | 否 |
| type | int | 11 | 试题编号 | 否 |
| answer | varchar | 255 | 答案 | 否 |
| bank | int | 11 | 题库编号 | 否 |

3.type（试题类型）信息表，如表4-3所示。

表4-3 试题类型信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 备注 | 是否允许空值 |
| id | int | 11 | 编号 | 否 |
| type | Varchar | 255 | 类型名 | 否 |

4. option（试题选项）信息表，如表4-4所示。

表4-4 试题选项信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 备注 | 是否允许空值 |
| id | int | 11 | 编号 | 否 |
| option | varchar | 255 | 选项 | 否 |
| question | int | 11 | 试题编号 | 否 |

4.5 本章小结

本章对系统的功能进行了总体设计，设计了手机端的页面布局。通过E-R图理清了各个实体之间的关系。在数据库的实现中，分析出系统中所需建立的表，并对各个表之间的关系进行了分析和说明，对表的设计进行了详细的讲解和说明，同时介绍了表的具体实现过程。

1. 详细设计与实现

本章主要介绍系统的功能模块划分和详细设计实现部分，根据需求分析和用户角色定位，对系统功能结构进行划分。利用活动图能够清晰的表示基本事件流和备选事件流的优点，对系统的主要功能进行活动图的建模。并在此基础上逐步实现系统各项功能。

5.1 后台Web端功能设计

### 5.1.1 题库管理模块设计

题库管理模块主要分为数据解析和数据存储。数据解析功能，系统使用Apache POI技术对Word文档进行解析，通过此框架则可以将一整个Word文档解析成Java可识别的String对象，通过对字符串对象的处理来进行解析试题，使用正则表达式来控制识别每一行的数据，此行具体是题目还是选项。先对每一行的数据进行辨别，然后将每行的数据填充到实体类的具体字段中。通过Spring Data的可持久化类进行管理，经过一次事务的提交后将持久化类中的数据插入到具体的数据库中。题库管理的活动图如5-1所示。



图5-1 题库管理活动图

扫描的每一个文档都会生成具体的题库，这个题库的名字就是文档的命名，题库中的选择题和判断题的类型经过筛选后都会进行设置，每一道题目都进行构造后就会生成整个题库的信息。

管理员只需要将制定格式的题库文档上传到服务器中，系统启动后会自动进行文档的扫描和对数据库的插入操作。

### 5.1.2 公告栏管理模块设计

公告栏模块主要用来告知用户版本信息和一些使用提示，需要管理员来修改配置文件中的公告栏内容，公告栏内容中可以任意编写，用户在打开使用的时候会看到公告栏，还有就是在用户升级的时候弹出的对话框会提示系统公告。所以在公告栏中管理员需要写入新版本的更新动态，这样可以吸引用户自动升级。公告栏模块活动图如5-2所示。



图5-2 公告栏管理活动图

### 5.1.3 手机端版本管理模块设计

手机端版本管模块主要面向系统管理员和开发人员。开发人员将新版本的Android端应用打包后发布给系统管理员，系统管理员将此应用的安装包上传到服务器端，然后修改配置文件中的版本号和指定的安装包下载路径即可。

此模块主要用户手机端自动更新，每次发布新版本后都需要更改配置文件，然后用户使用的时候就以服务器版本为准。手机端版本管理模块的活动图如图5-3所示。



图5-3 手机端版本管理模块活动图

5.2 手机Android端功能设计

### 5.2.1 自动更新模块设计

手机端自动更新模块的实现和应用就是为了让此软件有更好的活力和更久的生命周期。一款可以更新维护升级的软件才可以让用户长久使用。而自动更新模块的设计与实现主要依靠服务器提供数据，在手机端进行检测。用户每次打开手机后就会向服务器发送HTTP请求，获取服务器上最新版本号，然后与本机版本号做对比，如果版本号不一致，则再次请求最新应用的下载链接，进行覆盖安装。自动更新模块的活动图如图5-4所示。



图5-4 自动更新模块活动图

### 5.2.2 答题模块设计

答题模块为整个应用的核心模块，整个答题模块分为三个子项：题库解析，试题构造，答案判断。

* + 1. 题库解析

题库解析主要将后台服务器中获取的JSON格式的数据进行解析，生成一个题库数组，然后将数据存入缓存中，以便下次读取，每次进行解析的数据采用二级缓存。先读取手机缓存中的旧数据，然后给用户展示，用户使用的过程中进行联网请求最新题库进行下载。下载后覆盖缓存再次加载题库。

* + 1. 试题构造

试题构造通过已经解析后的数据，通过ViewPageAdapter进行填充ViewPager，在构造过程中进行判断，判断题目类型，如果是单选题和判断题则使用RadioButton来构造成选项如果是多选则使用Checkbox来构造生成选择项。并且够着后将所有题目数量放到整个页面的最下端。将题号设置到制定位置。

* + 1. 答案判断

每次点击查看答案的时候会进行答案判断，如果回答错误则将此题加入错题库。

答题模块的活动图如图5-5所示。



图5-5 答题模块活动图

### 5.2.3 背题功能模块设计

背题功能主要实现用户使用过程中需要学习和进行背题的需求。比如说刚刚开始熟悉题库的时候上来就进行做题错误率非常高，所以先进行题目的学习，与背诵然后在使用答题功能则是更为快速的学习方式。背题活动图如图5-6所示。



图5-6 背题活动图

5.3 后台Web端系统类的设计

后台管理使用Spring Boot框架集成，使用SpringMVC作为整体的控制层，使用SpringData+JPA作为持久化类转化的DAO层，使用Maven进行项目管理控制。系统包图如图5-7所示，项目包名为：pdsu.rj.mzc。一共六个包分别是：

根包下主要放系统启动的入口。

controller包下存放web控制层的代码。

dao层主要实现数据库获取数据的代码。

dto层主要实现数据转化接受的实体类。

entity层主要存放ORM实体类。

service层主要实现系统的业务逻辑代码。

utils层主要存放一些工具类。



图5-7 Web系统包图

系统使用SpringData+JPA作为数据库管理DAO层，在controll层中使用SpringMVC的地址映射关系作为页面的数据获取和手机端接口的获取。

系统主要实体类如下：

1. DtwebApplication类作为SpringBoot项目的整体程序的入口，类似与程序的Main函数。启动此类则启动内置的Tomcat服务器。
2. GlobalExceptionHandler类作为系统全局的异常捕获类，如果系统出现异常则会自动进入此类返回相应的结果。该类可以捕获Sql异常，空指针异常，类型转化异常，传递非法参数异常，算数运算异常，数组传递非法对象异常，下表越界异常，数字格式异常等12种异常消息信息。
3. WebLogAspect类作为系统的整体的日志类，控制着系统日志的打印，使用切面编程思想，将切面放在函数的调用和返回处，每次的调用则会自动打印请求的数据和参数。
4. TimuController类作为手机端数据获取的接口类，里面开放了指定的接口，通过HTTP地址调用就可以返回指定的数据，所有的数据格式由service层获取。
5. TimuService类是题目数据获取的业务处理类，通过调用dao层的数据库中的数据然后经过处理，返回指定格式化的数据，通过调用GsonUtils进行JSON数据格式的转化。
6. QuestionBankRepository类是其中的一个dao层代码，使用SpringData+JPA后所有的dao层代码就很简单，里面只需要写指定的接口方法即可自动生成查询语句。

5.4 手机Android端系统类的设计

手机应用端的设计思想借鉴了MVC设计模式，将整体系统分为视图展示层、逻辑控制层、实体模型层。页面布局构造上使用适配器模式，自定义了控件生产策略。系统包图如图5-8所示，项目包名为：pdsu.rj.mzc.mg。一共八个包分别是：

根包下主要放展示页面和答题页面。

entity包下主要存放一些Gson数据转化后的试题类。

adapter包下主要实现ViewPager的适配器。

activity包下主要存放题目生成的Activity类。

global包下存放全局的控制类。

utils包下存放各种工具类。

test包下存放测试代码。



图5-8 Android系统包图

系统使用XUtils3托管整体Activity控件的注入，使用Xutils3网络模块进行HTTP请求，使用适配器adapter构造生成ViewPager，系统主页的Activity主要用于展示页面效果。系统集成了Utils类用于快速调用。

系统主要数据实体类如下：

* + - 1. Timu类对应后台传递过来的所有题目，包括题干、选项、答案、类型等。
      2. UpdateMsg类对应后台配置信息数据，包括更新的版本，最新版本下载路径，更新的公告栏等。
      3. TiMuViewHolder类对应ViewPager的二级缓存，存储了页面上的控件id以便再次加载能快速读取数据。

系统中主要核心功能类如下：

* + - 1. GetDataUtil类通过调用getdata()方法进行联网获取最新数据存储到UpdateMsg类中在类中主要调用XUtils3的http()方法进行请求网络数据，通过回调函数控制返回数据，类中声明了两个abstract抽象方法:getTimu(ArrayList<Timu> timu)、error(String msg)，所有数据以参数的形式进行传递，对调用方可以进行监听控制。
      2. MyApplication类主要存放各种全局变量，比如说服务器端的url地址和一节启动应用需要获取的上下文Context和启动应用就需要加载的XUtils3框架的注册。
      3. SplashActivity类是实现页面的一个Activity，主要功能是应用的展示页面，内容有公告栏和答题入口等。在应用的onCreate方法中进行注册XUtils3的view控件，然后通过getVersion()方法获取到当前版本号，设置到指定位置。然后调用checkUpdate()方法进行检查升级，在checkUpdate()方法中主要实现了以下业务逻辑：

1. 联网访问服务器升级地址，如果检测到最新版本号与本机当前版本不符合则弹出提示升级对话框。
2. 提示升级对话框，弹出的对话框中有公告内容和两个按钮，确认按钮则调用自动下载功能，取消按钮则不进行升级，进入主页面。
3. 自动下载功能，首先检查手机SD卡是否加载，如果有已经加载储存卡，则将安装文件默认下载到外部储存卡中，如果检测到没有插入内存卡，则将安装包下载到应用目录中。下载功能使用XUtils的下载功能，支持断点续传，支持多线程下载。点击更新后则弹出下载对话框。对话框中显示有下载进度。
4. 覆盖安装Apk，如果下载完成则调用installAPK(File file)方法，此方法集成了系统的解压包程序，可以自动覆盖安装。
   * + 1. MyViewPagerAdapter类是其中一个ViewPager的适配器，一个适配器继承了PagerAdapter必须实现指定的方法，以下是实现的方法的详细解释：
5. setPrimaryItem方法：主要设置当前适配器需要展示的一个子页面。
6. destroyItem方法：主要设置适配器删除一个子页面时调用的析构方法。
7. instantiateItem方法：此方法为主要构造子页面方法，其中做了复杂的逻辑判断，用于控制显示页面的配置。
8. getCount方法：返回整体ViewPager的数量，用户翻页控制。
9. isViewFromObject方法：重用子页面方法，官方说明调用此方法可以使用户更好的体验。
10. notifyDataSetChanged方法：用于ViewPager的强制刷新。
11. getItemPosition方法：设置当前的页面位置，主要用户ViewPager的强制刷新功能。
    * + 1. TestViewPager类是其中一个Activity类，里面封装了题目的ViewPager，在onCreate方法中主要调用了GetDataUtil类中的数据获取方法，然后在回调函数中实现获取数据后的各种加载。

5.5 系统部分功能的实现

### 5.5.1 题目的解析

题目的解析功能就是将word格式的数据进行转化，辨别word文件中每行的数据指定具体意义，然后存入到数据库中。使用Apache POI工具可以将word文件中的文字进行逐行读取，通过正则表达式来进行每行数据的辨别[9]。

通过在TimuController中绑定指定的url访问路径，调用getTiku()方法中调用在TimuService类中的getData()方法。该方法中会扫描配置文件中的文件夹路径，将文件夹中所有的文档进行扫描。逐行分析数据然后存储到数据库中。解析流程图如图5-9所示。类关系图如图5-10所示。



图5-9 解析流程图



图5-10 题目解析类图

通过访问项目下/timu/getTimu接口可以查看数据成功解析转化。如图5-11所示。

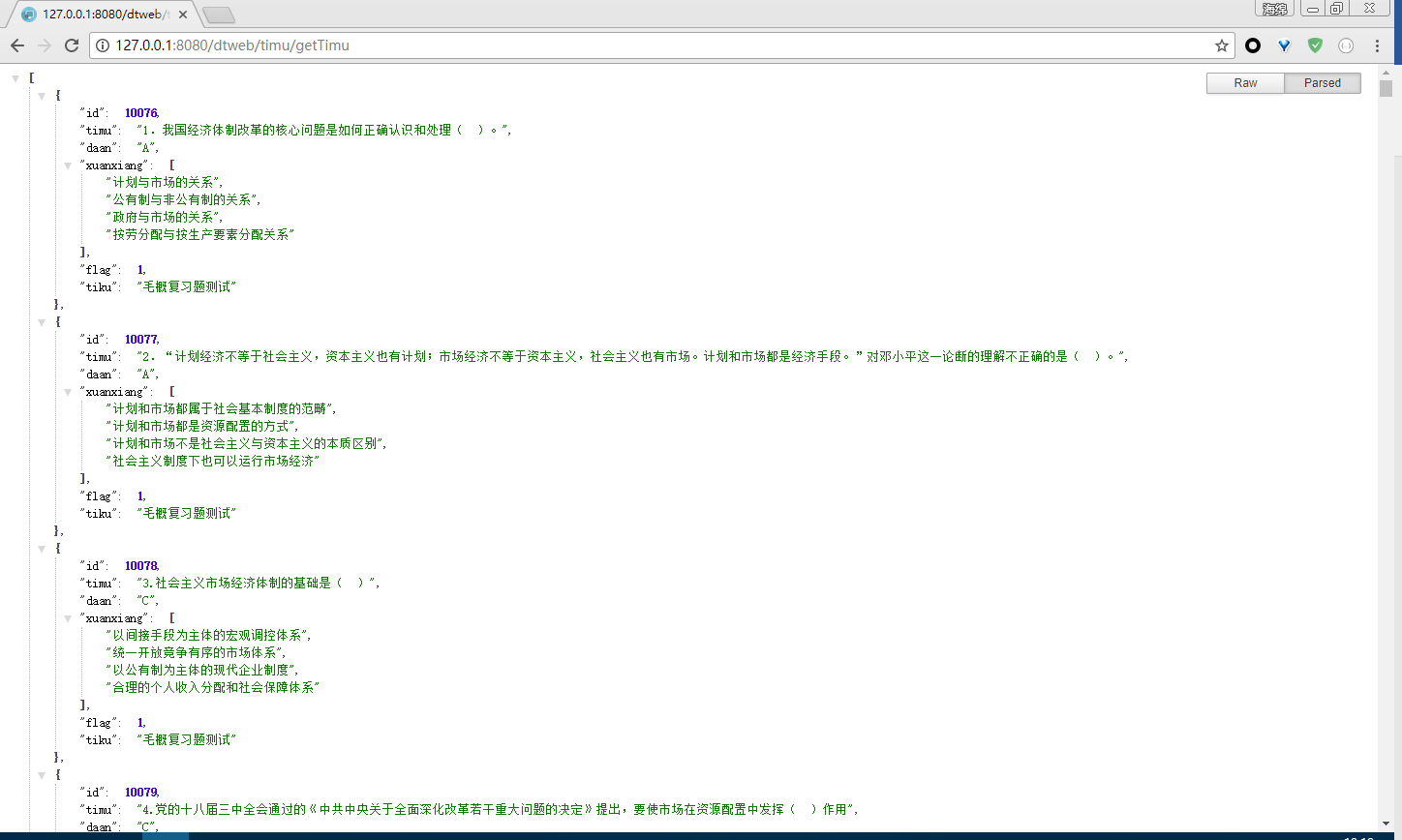


图5-11 题目解析数据图

这个数据是通过Gson进行转化后的JSON数据，手机端通信只需要访问此链接，就可以获取到数据，然后在手机端进行解析即可实时更新题库。

### 5.5.2 手机端题目的构造

对于手机端题目的构造分为三类题目：单项选择题，多项选择题和判断题。通过转化访问HTTP连接请求获取到的JSON数据，获取到题目类型，根据题目类型类进行设置选项的view展示。单项选择题和判断题为RadioButton即单选框，多项选择题为CheckBox多选框。并且判断题生成的单选题的选项为正确和错图。正常选择题的选项为获取到的数据。

打开TestViewPager类调用onCreate()方法，构造生成的ViewPage，在MyViewPagerAdapter构造器中进行逻辑判断，生成答题页面。类图如图5-12所示。



图5-12 题目构造类图

在手机端使用的时候点击查看题库即可查看已经解析过的题目，如图5-13和5-14所示。分别是单选题和多选题的构造展示。



图5-13 题目构造类图



5-14 题目构造类图

### 5.5.3 自动更新的处理

每次打开系统后会调用检查更新模块，在SplashActivity的onCreate方法中会调用checkUpdate()方法，用来联网获取最新版本，如果需要更新则调用showUpdateDialog()方法进行弹出更新对话框。点击更新后调用Update(String apkurl)方法，用来下载安装[10]。类图如图5-15所示。



图5-15 自动更新类图

检测到有新版本的时候回弹出提示框，并且显示出公告内容。如图5-16所示。点击立即更新自动下载，如图5-17所示。

图5-16 更新提示 5-17 更新下载

### 5.5.4 手机端应用数据缓存

答题系统做了缓存系统就是为了避免断网后无法答题的情况，有了数据缓存，在没有网络的时候获取的是本地上次的数据。将题目和公告放入二级缓存中，先读取缓存中的数据进行加载，并且异步请求最新数据，获取到最新数据后加载最新数据，并且覆盖保存到缓存中。

主要通过PrefUtil类实现缓存。当加载数据的时候调用set方法将数据进行缓存，在获取数据的时候get缓存中的数据即可，存储格式为Map(key,value)格式。如图5-18和5-19所示。



图5-18 未连接到互联网



5-19 断网查看题库

### 5.5.5 答题功能

选择题目进行回答的时候会给出答案提示，并且将错题加入错题库中。答案判断控制在MyViewPagerAdapter中的showAnswer(int)方法中，如果回答错误则将错题添加到ErrorVIewPager中的errormap中。并且所有的数据进行了缓存管理。类图如图5-20所示。



图5-20 答题功能类图

点击每道题答题的时候回给出答案提示，效果如图5-21所示。



图5-21答案提示

### 5.5.6 题库管理

超级管理员可以对题库进行维护操作。也就是查看所有的题库，可以点击扫描题库将指定目录下的所有文档都存入数据库。当点击启用某一题库的时候，系统会先禁用所有的题库，然后开启选择的题库，只能有一个题库处于被使用状态。而这个题库就是手机端获取的题库。

运行后台DtwebApplication中的主程序，然后打开后台管理页面。点击题库信息，调用DataTableController返回表格数据。具体的业务逻辑层在QuestionBankService中实现，查询所有的题库然后进行格式化返回，接口中判断了查询条件，如果有查询条件则对查询的结果进行筛选，系统中给出了以题库名为条件进行查询。类图如图5-22所示。



图5-22 题库管理类图

通过前台页面的题库操作就可以控制手机端的数据。管理页面如图5-23所示。

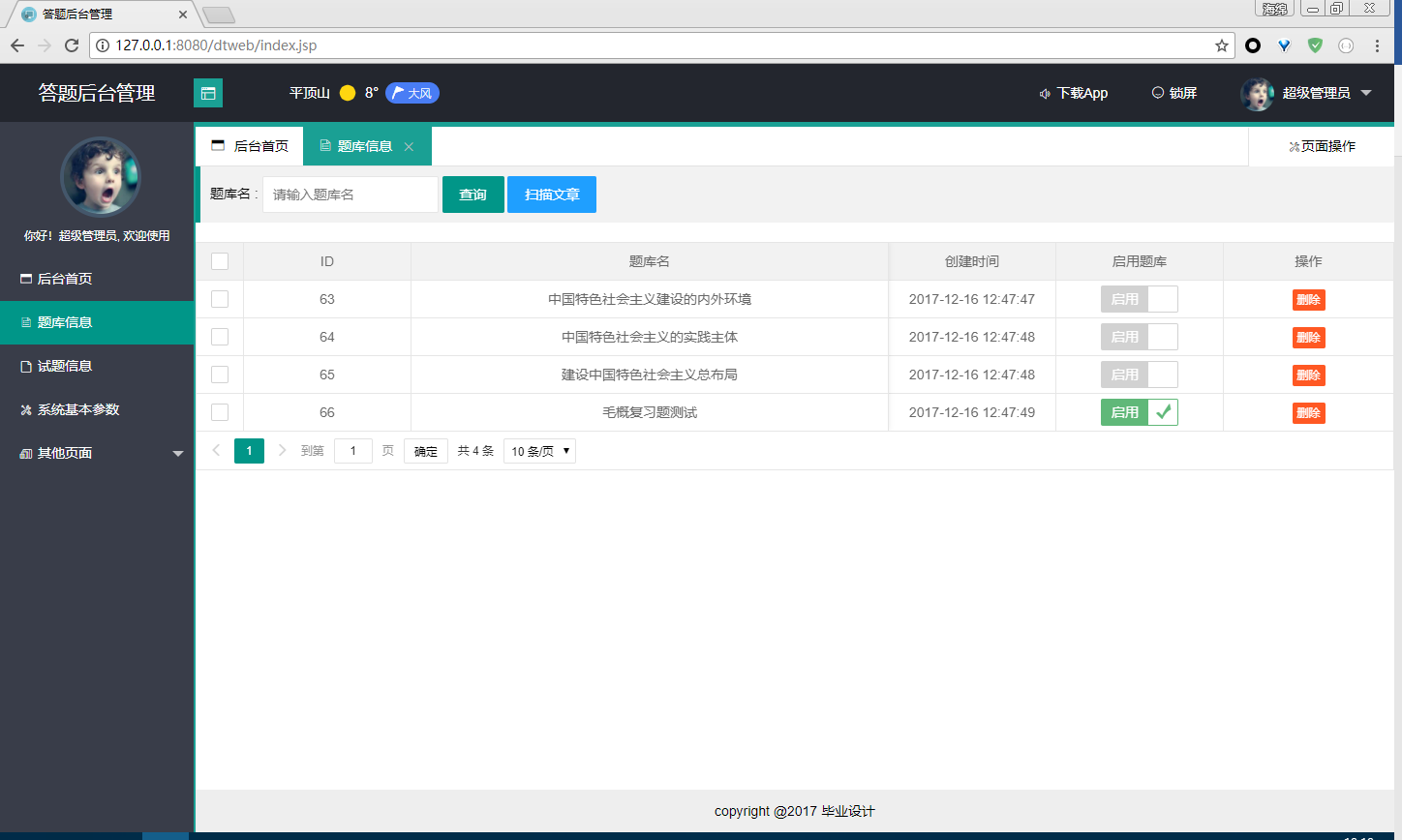


图5-23 题库管理页面

5.6 关键技术使用说明

1. Xutils3主要使用在手机Android端的框架搭建中，普通项目进行布局文件映射则需要多次调用系统方法，非常繁琐和复杂。使用Xutils3框架后整体的Android端应用的开发将简化此类操作，所有的布局文件引入可以直接使用注解映射。使用框架优化的还有联网请求操作，正常情况的Android端对于联网请求必须放在子线程中进行请求，然后在主线程中创建一个Handler进行异步监听，将所有结果放到回调函数中才可以进行处理，如果在主线程中进行联网请求则会对主线程进行堵塞，导致系统崩溃。使用框架后的联网请求就非常轻便，不需要主动的创建线程，框架会自动创建子线程进行处理，只需要在请求的回调函数中进行判断即可，大大的优化了代码的可读性。

对于Xutils3框架的使用让我更加熟悉了Android应用的开发，使得更多的精力放在了系统实现上。

1. Gson作为一个JSON数据格式解析器，由Google公司提供的一个工具类，其强大之处在于快读的处理JSON映射关系，只需要将需要格式化或者转化的数据写成一个实体类将其对应起来，字段名和数据名保持一致，在类中生成get方法和set方法即可完成转换。普通的JSON格式转换复杂并且冗余，一个上百条数据的JSON代码实现可能也需要上百行。并且只是普通的JSON格式，如果有复杂的对象包含对象格式，列表包含对象格式等等。复杂的格式转化加解析采用普通方法会大大增加工作量。本项目中使用Gson工具进行格式转换，整体的前后端的数据交互全部是JSON格式，保证数据的统一[11]。

对于Gson工具的使用让我熟悉了JSON数据传递的重要性，并且更加熟练的使用该工具。

1. Spring Boot是目前最流行的Spring体系的骨架，其设计目的就是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式类进行配置，从而让开发人员不需要定义样板化的配置。通过这种方式，SpringBoot主要优化了Spring体系的大量XML配置以及繁琐的依赖管理。本项目使用SpringBoot框架集成，可以在极少的代码实现复杂的可执行的Web应用。SpringBoot框架依赖Maven工程，先导入Maven项目在pom文件中配置少量的依赖框架。比如说此项目使用的SpringMVC框架、SpringData框架Apache POI框架等。使用Maven托管项目后大量的配置文件都已经被SpringBoot写好了。项目中只需要修改少量的比如数据库的连接池和Web应用启动后绑定的端口号。之前的XML配置现在已经简化为properties文件，使用文件进行配置的优点是每次读取数据不再需要校验判断。整个Web应用的入口是一个Application内部集成了Tomcat服务器，直接以Java应用的方式运行即可启动。

使用SpringBoot集成的项目使我了解学习了J2EE项目开发的前沿知识，使用更好更新的技术开发出的软件更有活力。

1. Apache POI作为一个免费开源的对办工软件Office格式档案的读写功能。本项目中的主要数据存储一个在数据库中，另一个就是在Word文件中。这也是项目中数据库的表结构较为简单的一个方面。使用Apache POI框架提供的对Word格式信息读取，让数据的获取和维护能更加贴近使用者，不需要使用者懂得数据库的维护知识，只需要对常用的Word文件进行修改，所有的数据可同步获取。项目中存放在数据库中的数据只是用来存放数据，作为一个读取的缓存，如果每一次的加载和请求都需要从新读取Word文档的话，会大大增加用户的等待时间，所以将数据读取后存放在数据库中能优化用户体验。使用Apache POI文档数据的获取，对数据的优化和解析就需要配合使用正则表达式。使用正则表达式来对数据进行格式化获取。

使用Apache POI框架使我学习了新的数据存储方式，改变了所有数据存储到数据库中的思维。

5.7 本章小结

本章主要介绍系统的详细设计与实现，给出了详细的功能结构图，清晰的展示了系统的功能，对各个模块进行分析画出活动图及具体实现的截图。并对关键技术使用进行了说明。

1. 测试

软件测试就是利用测试工具按照测试方案和流程对产品进行功能和性能测试。执行测试用例后，需要跟踪故障，以确保开发的产品适合需求。

6.1 测试的意义

软件质量和人们的利益密切相关，因此，人们对软件质量的重视程度越来越高。为了避免由于软件质量给人们带来的一系列经济等各方面的损失，就需要通过软件测试来确保软件的质量。软件工程的总目标是充分利用有限的人力和物力资源，高效率、高质量地完成软件开发项目。软件测试是软件生命周期中一项非常重要的工作，对保证软件的可靠性具有极其重要的意义[12]。

6.2 测试环境

测试环境如下表6-1所示。

表6-1 测试环境表

|  |
| --- |
| 软件环境（相关软件、操作系统等） |
| 操作平台：Windows 10 |
| 数据库：MySQL 5.0 |
| 服务器：Tomcat 8.0 |
| 浏览器：Google Chrome |
| 手机版本：Android4.2.2 |
| 硬件环境（网络、设备等） |
| CPU：2.4GHz |
| 内存：8G |
| 硬盘：256G |
| 手机RAM: 512M |

6.3 功能测试

[功能测试](http://baike.baidu.com/view/651435.htm)也叫[黑盒](http://baike.baidu.com/view/1413605.htm)子测试或[数据驱动测试](http://baike.baidu.com/view/760744.htm)，只需考虑各个功能，不需要考虑整个[软件](http://baike.baidu.com/view/37.htm)的内部结构及代码。一般从软件产品的界面、架构出发，按照需求编写出来的[测试用例](http://baike.baidu.com/view/106882.htm)，输入数据在预期结果和实际结果之间进行评测，进而提出更加使产品达到用户使用的要求。

功能一：超级管理员登录功能

测试目的：测试管理员是否能登录系统

测试方法：黑盒测试

测试数据如表6-2所示。

表6-2 登录系统的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 用例ID | 用户名 | 密码 | 测试结果 |
| 1 | admin |  | 提示必填项不能为空 |
| 2 | admin | 123 | 提示密码错误 |
| 3 | admin | 123456 | 登录成功 |

测试步骤：

1.运行程序，进入“登陆页面”。

2.输入测试数据1，点击登陆，提示必填项不能为空。

3.输入测试数据2，点击登陆，提示密码错误。

3.输入测试数据3，点击登陆，登陆成功，跳转主页面。

功能二：题目搜索功能

测试目的：验证管理员是否可以查询题目

测试方法：黑盒测试

测试数据如表6-3所示。

表6-3 发布物品信息的测试

|  |  |  |
| --- | --- | --- |
| 用例ID | 题目名 | 测试结果 |
| 1 |  | 查询所有结果 |
| 2 | 中国 | 查询题目中有含有“中国”的结果并且高亮显示 |
| 3 | abc | 查询无数据 |

测试步骤：

1.运行程序，登录后选择“试题信息”。

2.输入测试数据1，点击查询，显示所有查询结果。

3.输入测试数据2，点击查询，高亮显示包含“中国”的查询结果。

4.输入测试数据3，点击查询，显示无数据。

6.4 界面测试

界面是[软件](http://baike.baidu.com/view/37.htm)与用户交互的最直接的层，界面的好坏决定用户对软件的第一印象。而且设计良好的界面能够引导用户自己完成相应的操作，起到向导的作用。

表6-4是对题库信息界面进行测试。

表6-4 题库信息界面的测试数据

|  |  |
| --- | --- |
| 测试目标 | 1.对弹出窗口进行测试  2.对显示数据进行测试 |
| 测试范围 | 1.系统主页面  2.题库信息页面  3.题库列表展示  4.启用题库  5.删除题库 |
| 开始标准 | 用户点击“题库信息”页签 |
| 完成标准 | 正确并且分栏显示题库信息，在执行搜索、删除等操作时，界面应能及时更新显示内容，即重新进行信息的正确显示 |
| 优先级 | 高 |

表6-5是对试题信息界面进行测试。

表6-5 试题信息页面的测试数据

|  |  |
| --- | --- |
| 测试目标 | 1.对弹出窗口进行测试  2.对显示数据进行测试 |
| 测试范围 | 1.系统主页面  2.试题信息页面  3.试题列表展示 |
| 开始标准 | 用户点击“试题信息”页签 |
| 完成标准 | 正确并分栏显示试题信息，在搜索、查看等操作时，界面能及时更新内容，可以更加详细的查看信息。 |
| 优先级 | 高 |

6.5 系统性能测试

系统测试的目的在于测试系统与设计需求相比较，及时的发现系统与设计不符或者矛盾的地方，发现这些问题，进一步的进行系统完善工作。主要包括系统压力测试和系统性能测试。

系统压力测试：压力测试的目的在于获取系统能够保证正常运转的极限情况，经测试，本系统可以支持100人同时进行使用不会发生系统稳定性和性能问题。

系统性能测试：系统性能测试主要是模拟用户实际的使用负载来测试系统的响应时间，数据吞吐率以及系统的承载能力。

1. 系统响应时间：在系统压力测试的情况下，本系统的动作响应时间可以达到用户的需求。经过测试，本系统在100个用户同时操作时可以保持0.5秒的响应时间。
2. 数据吞吐率：经过测试，本系统可以在单位时间内完成规定的数据量，可以满足用户的使用需求。
3. 系统承载能力：经过系统压力测试，本系统在保证正常的响应时间的前提下，能够支持100人同时使用，不会产生稳定性问题。

6.6 问题严重程度描述

通过功能测试和界面测试，发现系统中存在一定的问题。现将问题总结如下。

问题严重程度描述如表6-6所示：

表6-6 问题严重程度描述表

|  |  |  |
| --- | --- | --- |
| 问题严重度 | 描述 | 响应时间 |
| 高 | 用户退出后不会跳到登录页面 | 立即处理 |
| 中 | 页签打开后无法刷新最新数据 | 1天以内 |
| 低 | 试题搜索框无法查询特殊符号 | 3天 |

6.7 本章小结

软件测试的目的是为了发现尽可能多的缺陷，通过修复缺陷来降低软件开发及维护的风险。本章利用软件测试的方法对系统进行了功能测试和界面测试，描述了测试中发现的缺陷，会在以后的工作中修复这些缺陷。

1. 结束语

本论文主要设计了一个基于Android与J2EE技术的答题软件的开发，以简单易用，界面友好，运行流畅为主要目标，把所学知识充分结合，完成了该项目的设计开发。

7.1 总结

通过这次系统的实践，将所学知识进行了一次综合的应用，本论文主要论述了一款答题软件的设计与开发工作。通过本论文的研究与设计，开发出了高效实用的答题软件。本文开发答题系统的工作主要包括以下方面：

1. 对答题系统的需求进行调查和分析。
2. 对手机端的系统进行分析，确定本系统实用框架，以及各个功能模块的开发和更新。
3. 学习各个模块技术的使用，运用成熟的J2EE开发技巧。
4. 完成答题软件的开发，为老师和同学提供可靠稳定的答题平台。

7.2 展望

通过对Spring体系的学习，越发的感觉以后的软件开发会更加依赖Spring体系。开发出一款答题软件，经过系统测试，在稳定性和易用性方面达到了基本的要求。系统的自动更新可以让我更好的去维护这款软件。使用本软件可以让学生更好的复习和测试。发展答题软件系统可以更加高效的复习。对现今高效发展具有现实的推进意义。

此次毕业设计的完成过程中，阅读了大量的书籍和网站。最终开发出一套完整答题系统。通过对答题系统的开发，使我的自学能力和编程能力得到了巨大的提高。对框架的使用更加娴熟。比如说Maven对项目的管理，通过pom文件将各个程序的环境依赖关联起来。结合Spring体系使得程序更加明了，可以更好的维护。

对于本系统的展望我希望可以加入用户登录和注册系统，对于每一个用户都拥有一个后台管理。用户可以进入自己的后台管理系统进行操作，比如维护自己的题库系统，扫描上传自己的题库，可以自己查看自己的题库。这样的答题软件更具有灵活性。并且还可以加入分享等操作，分享自己的题库给别人。加入好友系统，互相加好友然后可以添加聊天系统，好友与好友之间可以相互聊天。最后可以加入空间系统，类似于微信一样，可以发布自己的动态。

我相信这个项目未来的发展空间很大，可以有一个自己的圈子。希望未来我能完善此系统为大家提供一个更好的答题软件。

**附 录**

创建试题的核心代码：

public class MyViewPagerAdapter extends PagerAdapter {

private View mCurrentView;

private ArrayList<Timu> timulist;

private Context context;

private boolean showAns; // 是否显示答案

public MyViewPagerAdapter(ArrayList<Timu> timulist, Context context) {

this.timulist = timulist;

this.context = context;

}

/\*\*

\* 设置当前显示的view

\*/

@Override

public void setPrimaryItem(ViewGroup container, int position, Object object) {

mCurrentView = (View) object;

super.setPrimaryItem(container, position, object);

}

/\*\*

\* 获取当前显示的view

\*

\* @return

\*/

public View getCurrentView() {

return mCurrentView;

}

@Override

public void destroyItem(ViewGroup container, int position, Object object) {

container.removeView((View) object);

}

@Override

public Object instantiateItem(ViewGroup container, int position) { // 这个方法用来实例化页卡

View view = View.inflate(context, R.layout.item\_timu, null);

final TiMuViewHolder holder = new TiMuViewHolder();

x.view().inject(holder, view);

showAns = TestViewPager.isShowAns();

final Timu timu = timulist.get(position);

holder.tv\_index.setText(position + 1 + "/" + timulist.size());

RadioGroup group;

if (TextUtils.isEmpty(timu.getYixuan())) {

timu.setYixuan("");

}

switch (timu.getFlag()) {

case 1:// 单选

group = new RadioGroup(context);

for (int i = 0; i < timu.getXuanxiang().size(); i++) {

RadioButton rb = new RadioButton(context);

LayoutParams params = new LayoutParams(LayoutParams.MATCH\_PARENT, LayoutParams.WRAP\_CONTENT);

rb.setLayoutParams(params);

rb.setText(timu.getXuanxiang().get(i));

// 勾选已经答过题的

if (timu.getYixuan().contains("" + (char) (i + 'A'))) {

rb.setChecked(true);

}

group.addView(rb);

}

// 显示答案

if (showAns) {

for (int i = 0; i < group.getChildCount(); i++) {

RadioButton rb = (RadioButton) group.getChildAt(i);

if (timu.getDaan().contains("" + (char) (i + 'A'))) {

rb.setChecked(true);

}

}

}

holder.ll\_xuan.addView(group);

break;

case 2:// 多选

for (int i = 0; i < timu.getXuanxiang().size(); i++) {

CheckBox cb = new CheckBox(context);

LayoutParams params = new LayoutParams(LayoutParams.MATCH\_PARENT, LayoutParams.WRAP\_CONTENT);

cb.setLayoutParams(params);

cb.setText(timu.getXuanxiang().get(i));

holder.ll\_xuan.addView(cb);

// 勾选已经答过题的

if (timu.getYixuan().contains("" + (char) (i + 'A'))) {

cb.setChecked(true);

}

}

// 显示答案

if (showAns) {

for (int i = 0; i < holder.ll\_xuan.getChildCount(); i++) {

CheckBox cb = (CheckBox) holder.ll\_xuan.getChildAt(i);

cb.setChecked(false);

if (timu.getDaan().contains("" + (char) (i + 'A'))) {

cb.setChecked(true);

}

}

}

break;

case 3:// 判断

group = new RadioGroup(context);

RadioButton rb1 = new RadioButton(context);

LayoutParams params = new LayoutParams(LayoutParams.MATCH\_PARENT, LayoutParams.WRAP\_CONTENT);

rb1.setLayoutParams(params);

rb1.setText("正确");

group.addView(rb1);

RadioButton rb2 = new RadioButton(context);

rb2.setLayoutParams(params);

rb2.setText("错误");

group.addView(rb2);

holder.ll\_xuan.addView(group);

// 勾选已经答过题的

if (timu.getYixuan().contains("√")) {

rb1.setChecked(true);

rb2.setChecked(false);

}

if (timu.getYixuan().contains("×")) {

rb1.setChecked(false);

rb2.setChecked(true);

}

// 显示答案

if (showAns) {

if (timu.getDaan().contains("√")) {

rb1.setChecked(true);

rb2.setChecked(false);

}

if (timu.getDaan().contains("×")) {

rb1.setChecked(false);

rb2.setChecked(true);

}

}

break;

}

holder.tv\_timu.setText(timu.getTimu().trim());

view.setTag(holder);

container.addView(view);// 添加页卡

return view;

}

public void showAnswer(int position) {

TiMuViewHolder holder = (TiMuViewHolder) getCurrentView().getTag();

Timu timu = timulist.get(position);

if (showAns) {

return;

}

String xuanze = "";

if (holder.ll\_xuan.getChildCount() > 1) {// 多选

for (int i = 0; i < holder.ll\_xuan.getChildCount(); i++) {

CheckBox checkBox = (CheckBox) holder.ll\_xuan.getChildAt(i);

if (checkBox.isChecked()) {

xuanze = xuanze + (char) (i + 'A');

}

}

} else {// 单选或者判断

RadioGroup rg = (RadioGroup) holder.ll\_xuan.getChildAt(0);

for (int i = 0; i < rg.getChildCount(); i++) {

RadioButton radioButton = (RadioButton) rg.getChildAt(i);

if (radioButton.isChecked()) {

if (rg.getChildCount() > 2) {// 单选

xuanze = "" + (char) (i + 'A');

} else {// 判断

xuanze = i == 0 ? "√" : "×";

}

}

}

}

timu.setYixuan(xuanze);

if (timu.getDaan().equals(xuanze)) {

xuanze = xuanze + "\n答对了";

ErrorViewPager.removeTimu(timu);

} else {

xuanze = xuanze + "\n答错了";

ErrorViewPager.addErrorTimu(timu);

}

ToastUtil.show("您选择的是：" + xuanze + "\n正确答案是：" + timu.getDaan());

}

@Override

public int getCount() {

return timulist.size();// 返回页卡的数量

}

@Override

public boolean isViewFromObject(View arg0, Object arg1) {

return arg0 == arg1;// 官方提示这样写

}

// 以下是为了强行刷新ViewPager

private int mChildCount = 0;

@Override

public void notifyDataSetChanged() {

mChildCount = getCount();

super.notifyDataSetChanged();

}

@Override

public int getItemPosition(Object object) {

if (mChildCount > 0) {

mChildCount--;

return POSITION\_NONE;

}

return super.getItemPosition(object);

}

// ---------------------------

}

参考文献

1. 郑卓远.Android手机上的答题系统设计与实现[J].电脑编程技巧与维护.2016,15:26+50
2. 李燕.基于Android的会员卡信息管理设计与实现[D].西安科技大学,2017
3. 张峰.应用SpringBoot改变web应用开发模式[J].科技创新与应用.2017,23:193-194
4. 高华花.基于J2EE技术的毕业论文管理系统设计与实现[D].大连理工大学,2016
5. 朱永忠.基于Spring MVC框架的IT实训管理平台的研发[J].芜湖职业技术学院学报.2017,1:28-32
6. 彭安金.使用Spring Data和JPA在JavaEE系统中简化持久层[J].电子世界.2017,6:88-89+91
7. 赵鹏.一种基于JSON的Android终端远程获取解析数据的方法[J].微型机与应用.2017,21:57-60
8. 林雪南.基于Apache POI解析Excel文件及内存使用分析[J].电脑编程技巧与维护.2016,23:60-61+98
9. 贾志先.正则表达式在填空题答案设计及评分中的应用[J].智能计算机与应用.2017,2:81-83
10. 关磊.医院系统客户端的自动更新[J].中国医疗设备.2016,3:108-110
11. 徐宝磊.基于JSON的数据交换技术应用研究[J].软件导刊.2017,10:173-175
12. 朱晓敏.软件测试的相关技术应用研究[J].电子测试.2017,1:122-123

**致 谢**

感谢平顶山学院对我几年的培养！

在这大学四年的学习生涯中，我不仅学到了很多知识，而且平顶山学院也教会了我很多做人的准则和严谨的做事态度。

对于大学生来说，毕业设计是一个综合性的、实践性很强的环节，做毕业设计不仅要将所学的知识进行融合，还要学会灵活的变通。大学是一个开放的平台，需要不断的学习新知识来武装自己，老师交给我们的是不仅仅是知识更是解决问题的能力。

感谢张国平老师在我学业上的知指导，讲解不足之处。每当我遇到问题时，张国平老师总是不厌其烦的给我讲解，在这里向张国平老师表示我最诚挚的敬意。他无私奉献的敬业精神令人钦佩，他严谨细致、一丝不苟的作风令我受益终生。

感谢母校四年来对我的培养，感谢各位老师的教育，感谢我的父母，感谢他们对我的关怀和支持。

最后感谢所有帮助过我的人，衷心的祝福你们。