软件设计文档

目录

1. [技术选型](#技术选型)
2. [终端支持](#终端支持)
3. [服务器语言](#服务器语言)
4. [web框架](#web框架)
5. [关系型数据库](#关系型数据库)
6. [数据缓存](#数据缓存)
7. [集成与测试工具](#集成与测试工具)
8. [源代码管理](#源代码管理)
9. [架构设计](#架构设计)
   1. [架构表示方式](#架构表示方式)
   2. [架构设计目标与约束](#架构设计目标与约束)
   3. [用例视图](#用例视图)
   4. [逻辑视图](#逻辑视图)
   5. [进程视图](#进程视图)
   6. [实施视图](#实施视图)
   7. [部署试图](#部署试图)
10. [模块划分](#模块划分)
11. [软件设计技术](#软件设计技术)
    1. [面向对象编程](#面向对象编程（截图eclipse，诸多类）)
    2. [MVC设计模式](#MVC设计模式)
    3. [Memcache实现缓存](#Memcache实现缓存)
    4. [PDO的使用](#PDO的使用)
12. [测试用例](#测试用例)
13. **技术选型**
    1. 终端支持：Android

开发语言框架：Java, Android

选择理由：

* + 1. 队伍成员几乎都是使用安卓，而且有几个成员有安卓编程经验，相比其他系统，在安卓系统上较易上手和调试。
    2. 而且，我们这个app相对注重于好友间的互动，相对于使用web，在app端开发会更贴近于受众。
    3. 安卓有足够的市场。
  1. 服务器语言：PHP

选择理由：

* + 1. PHP灵活，上手快、已修改、发布快捷。
  1. web框架：broPHP

选择理由：

* + 1. 开源。
    2. broPHP完全采用面向对象的设计思想，易维护、质量高、效率高、易扩展。
    3. broPHP是基于MVC的三层设计模式，大大提高了代码的可重用性，可依此构造良好的少互扰性的构件，提高了应用程序的灵活性和可配置性。
  1. 关系型数据库：MySQL

选择理由：

* + 1. 开源。
    2. 通用，资料容易查询。
    3. 轻量易用，适合我们这种简易app的开发。
  1. 数据缓存：memcache

选择理由：

* + 1. 当后台管理系统的用户在第一次访问一个页面时，它会将该页面缓存在本地中。以后每次访问时，首先会去检测本地是否有该页面的缓存文件，如果有的话直接拿来显示。而如果本地没有该页面的缓存时，才去服务器获取，这在一定程度上减少了对服务器的访问，降低了服务器和数据库的压力。
  1. 集成与测试工具：Eclipse

选择理由：

* + 1. 开源。
    2. 可以实现多种操作，如浏览代码、编辑代码、编译代码、调试代码，等等。
    3. 对重构功能的支持较好。
  1. 源代码管理：Github

选择理由：

* + 1. 免费。
    2. 简单。
    3. 分布性强大，足以应付我们这个app的开发。

1. **架构设计**
   1. 架构表示方式

下面以一系列的视图（View）来表示系统的软件构架，主要包括用例视图、逻辑视图、进程视图、部署视图、实施视图等，每个视图拥有一个或多个模型（Model）。并围绕相关视图来描述系统的基本结构、组成机制与工作原理等。下面还将系统的构架机制描述也放在了逻辑视图之下。主要使用统一建模语言（UML）来充当相关模型的表达语言。

* 1. 架构设计目标与约束

描述构架设计最主要目标就是满足关键系统功能需求和质量约束，这些功能需求和质量要求对软件构架有重大的影响，并决定了构架的设计。本节同时还列明影响构架的其他相关因素，如软件的复用策略、使用商业构件、设计与实施的策略等。

1. 关键功能需求

按照需求分析文档的规格要求，campus run系统的设计分成了以用户使用功能模块以及管理员功能模块为主的两大模块进行开发，而在此两大模块的的基础上，又细分成了跑步计步，添加删除活动，小游戏等子功能模块。系统的开发主要是为了使campus run系统更加规范化、系统化和程序化，提高信息处理的速度和准确性，提高使用者对运动的用户体验。其功能模块示意图如下所示：

campus run

后台管理系统

用户使用界面

跑步计步

约跑信息查询

游戏

个人管理

后台登录

信息管理

信息查询

用户注册

用户登录

发布约跑信息

* 用户登录模块

用户登录模块主要是提供给用户登录的界面。用户需要在注册界面中注册自己的个人信息，在系统提示成功后，才可以在用户登录界面进行登录。

* 跑步计步模块

跑步计步模块主要是进行跑步活动的过程及历史纪录显示。其中调用了安卓的加速度传感器进行计步的模拟操作，并且运用一定的数学公式进行里程与步数之间的转化。在跑步活动结束时，将当前跑步活动记录下来。并且写入数据库。在后面登录时对上次使用的活动数据进行反馈显示。

* 信息查询模块

这个模块使用一个ListView列表对发布在服务器上面的约跑活动进行显示。点击每一条单独的Item可以显示约跑的活动详情。在详情页面中可以执行加入活动的操作。上方设置条件筛选框，在选框中选择筛选条件并在后台将条件进行显示转化。输出到输出列表中。

* 个人管理模块

这个模块可以管理自己发布的信息，参与的活动信息，还有个人的信息资料查看及修改。并且可以对软件的功能介绍和关于进行查看。

* 游戏模块

游戏模块主要是设置了小游戏，让用户可以在运动之余娱乐身心。并且在运动的过程中积累的里程数可以转化为游戏中的时间奖励，帮助获得更好的分数。

* 后台信息管理模块

这个模块主要是使用框架搭建了一个后台管理平台，并用PHP和前段的Android进行数据交互之后，将信息存储在后台的mySQL数据库，并通过这个平台进行管理员权限上的后台数据手动控制，包括增删查改等，也同时可以在上面完成审核及认证级别的筛选。

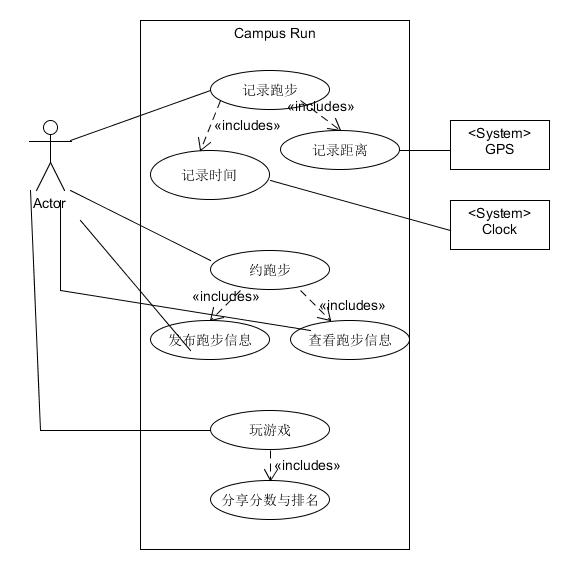
1. 关键质量需求

由于此开发项目针对运动爱好者的日常运动，使用频度较高，使用性要求比较高。为防止对信息资料和管理程序的恶意破坏，要求有较为稳定的访问速度及交互，和用户保密的安全性能。总之，要求稳定、安全、便捷，易于管理和操作。

* + - * 查询速度：不超过1200毫秒（访问超时直接弹出访问失败提示）；
      * 其它所有交互功能反应速度：不超过1500毫秒；
      * 安全性：密码使用加密并在输入框加入特定字符判断防止恶意代码入侵。
      * 可靠性：平均故障间隔时间不低于200小时。
  1. 用例视图
     1. 概述

用例视图从用户使用的角度描述系统构架的基本外部行为特性，通常包含业务用例模型与系统用例模型。业务用例模型不适用于本系统，这里只关注系统用例。这里选取了用例模型中对系统构架的内容产生重大影响的应用场景与用例集合，这些用例代表了系统主要的核心功能，决定了系统构架的基本组成元素。有些用例强调或决定了构架的某些具体然而重要的细节，通常也可以列在本节内，总之所列的用例集合应基本覆盖系统构架的主要方面。

* + 1. 关键用例
* **运动爱好者用例**



**游客的用例说明**

|  |  |
| --- | --- |
| 用例名称 | 简单描述 |
| 记录跑步 | 上次跑步的里程，步数，时间。当前跑步的具体详情，速度信息。调用安卓内置加速度感应器，以及时间接口进行速度时间的记录。 |
| 约跑步 | 在窗口显示约跑步信息列表，可以点击单挑信息ITEM进行查看，并且在弹出的详情页面中加入活动。还可以通过右上角按钮发布个人约跑信息。 |
| 玩游戏 | 内置悠闲小游戏，可以让使用者在闲暇时放松身心。并且游戏与运动挂钩，在跑步中积累的里程数等信息可以在游戏中转化为奖励积分，帮助冲击更高的分数。同时在游戏中设置排行榜，方便好友之间互相查看。 |

* 1. 逻辑视图
     1. 概述

逻辑视图从系统内在逻辑结构的角度描述系统的基本结构与动态行为，通常包括分析模型（Analysis Model）、设计模型（Design Model）以及数据模型（Data Model）等。设计模型说明了系统的组成元素、组织架构和关系，并描述了各组成元素的协作以及状态转换关系等（通过用例实现Use Case Realization予以表达）。本节将分别在系统层次结构模型中描述系统的层次组织结构；在主要的包和子系统中说明系统的具体组成。

* + 1. 系统层次模型

Android

客户端

后台

管理端

HttpClient

HTTP请求

MySQL

数据库

查询请求

ArrayList

Json

SQL语句

SQL

Json

* + 1. 主要的设计包和子系统

com.example.campusrun

postingMessage

runningImplement

* 1. 进程视图
     1. 概述

进程视图从系统运行时刻的角度，描述系统划分为进程、线程的结构，及其动态关系。模型主要说明不同系统角色之间的创建、交互和消息通讯关系等。

* + 1. 角色进程视图
* 跑步

用户

主界面

后台处理

数据库

点击开始跑步

当前运动信息

运动数据

当前数据及历史数据

当前步数速度里程

及历史数据



* 用户发布约跑信息：

用户

主界面

后台处理

数据库

点击开始跑步

发布信息处理

活动数据

是否发布成功

是否发布成功

显示成功界面

或失败提醒

* 游客注册：

用户

主界面

后台处理

数据库

填写个人信息

个人信息去重处理

个人信息数据

是否注册成功

是否注册成功

显示成功提醒

登录界面显示

* 用户修改个人信息：

用户

主界面

后台处理

数据库

修改个人信息

个人信息判断处理

个人信息数据

是否修改成功

是否修改成功

显示成功提醒

更新界面显示

* 游戏：

用户

主界面

后台处理

数据库

游戏操作

游戏信息处理

游戏成绩数据

是否最好成绩

是否最好成绩

最好成绩

是否最好成绩

最好成绩

* 用户加入约跑活动：

用户

主界面

后台处理

数据库

填写跑步信息

跑步信息

跑步信息数据

是否提交成功

是否提交成功

显示成功提醒

刷新列表

* 系统管理员修改后台活动记录：

管理员

主界面

后台处理

数据库

修改活动信息

活动信息

活动信息数据

是否修改成功

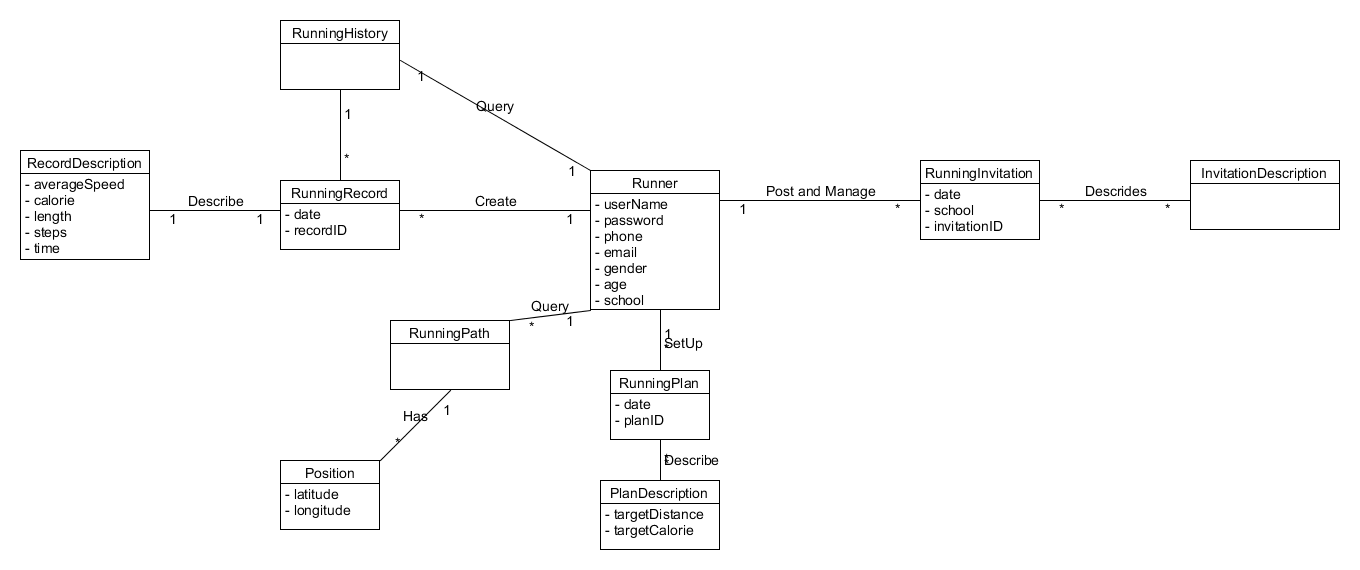
是否修改成功

显示成功提醒

* 1. 实施视图
     1. 概述

本部分从编译与构建的角度，描述系统实施构件的组织结构与依赖关系（主要是编译依赖）。模型包括实施子系统和构件结构，及其依赖关系。同时还表达了逻辑视图中各个包和类分配到实施视图中的子系统和构件的映射关系。

* + 1. 实施模型视图



* 1. 部署试图
     1. 概述

从系统软硬件物理配置的角度，描述系统的网络逻辑拓扑结构。模型包括各个物理节点的硬件与软件配置，网络的逻辑拓扑结构，节点间的交互和讯关系等。同时还表达了进程视图中的各个进程具体分配到物理节点的映射关系。

* + 1. 部署方案视图

用户1

用户2

用户3

Intenet

Web服务器

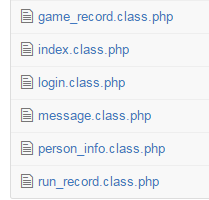
应用器服务器

防火墙

数据库服务器

1. **模块划分**

CampusRun的主要的功能有登录注册、计步、发布约跑消息、游戏和个人信息设置五大块。因此在设计时，我们是通过这五部分将应用分为五个模块，而每个模块里面又包含了其他的小模块，即每个模块里的各个小功能。故根据模块划分的结果，PHP Controller层文件也有如下几个，其中index.class.php不属于任何一个模块，它的作用是在客户端登录后返回后台已经有的约跑消息，交给安卓页面显示出来：



而在上面各个php文件中，其包含的功能如下所示：

1、game\_record用来记录游戏的数据，包括更新最高纪录、排名以及奖励；

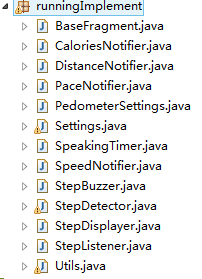
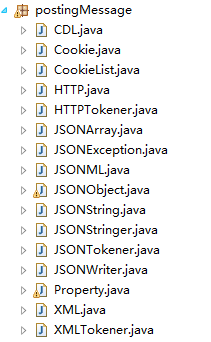
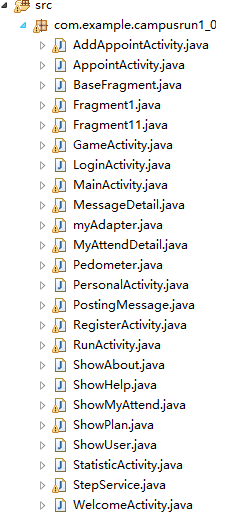
2、login提供了用户注册和登录的功能，包括了对输入的数据有效性进行检测等；

3、message提供了约跑信息的发布、删除、查看功能，以及根据一定的条件对显示的信息进行筛选；

4、person\_info记录了用户的信息，包括姓名、性别、手机、邮箱等属性。用户可以随时设置并更新自己的信息;

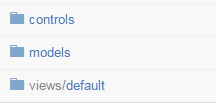
5、run\_record用于记录用户的运动记录，包括了个人上次跑步的时间、距离、消耗的卡路里和个人所有运动的时间、距离以及卡路里。对于这些数据，用户每次在开始新的跑步时，均会更新相关的数据。

1. **软件设计技术**
2. 面向对象编程（截图eclipse，诸多类）

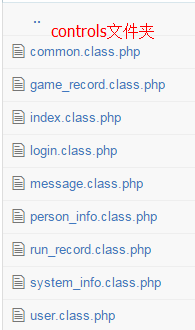
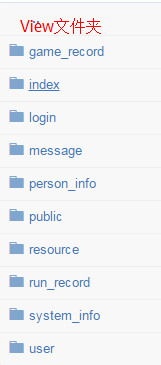
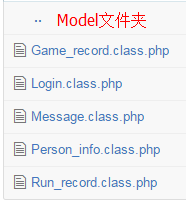


1. MVC设计模式

PHP后台的实现采用MVC模式。客户端在View层提交数据交给Controller层，然后由Controller向Model层发送数据，与数据库进行数据的交互，其中逻辑处理里部分是在Controller层实现的。具体目录组织形式如下图所示：



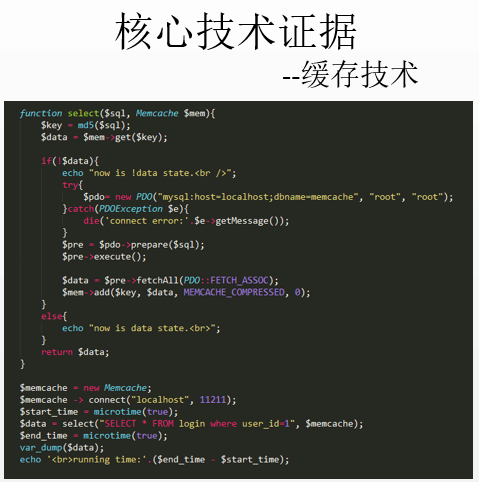
在每个文件夹里面包含了对应的文件，每个文件对应一个模块，而一个模块内包括了多个功能，比如最基本的增删改查。



1. Memcache 实现缓存

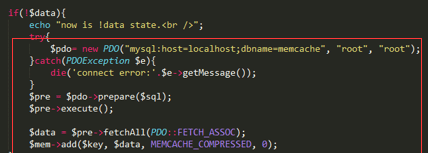
Memcache的原理是在本地中有一个服务器，存储用于经常访问到的数据，包括网站上的一些页面。每次用户向应用所在的服务器请求数据示，它首先会到本地服务器中查找是否已经有所需要的数据，如果有，则直接拿来使用，而如果没有，再到主服务器去获取。这在一定程度上减少了对数据库和服务器的依赖，也降低了服务器的压力。

利用Memcache技术去寻找缓存文件式，它是利用一个Key去寻找的，因为每次在生成本地缓存文件时，都会将该缓存文件与一个Key值绑定，一般情况下Key值是通过将SQL语句进行MD5加密映射后确定。如下所示：



1. PDO的使用

PDO就是一个“数据库访问”抽象层，作用是统一各种数据库的访问接口，能够轻松的在不同的数据库之间切换，使得数据库间的移植容易实现。同Mysql和Mysqli的函数库相比，PDO让跨数据库的使用更具有亲和力。具体使用如下：



1. **测试用例**

|  |  |  |
| --- | --- | --- |
| 用例 | 覆盖模块 | 备注 |
| 点击campus run图标 | 启动欢迎页模块 | 测试软件是否可以正常运行，欢迎页面显示是否正常，登陆页面跳转是否正常  前端显示是否正常 |
| 用户名：sherry  密码：sherry | 登录模块 | 测试已注册用户是否可以正常登陆  后台数据验证是否正常 |
| 用户名：hello  密码：hello  确认密码：hello  电话：1234567  性别：女 | 注册模块 | 测试新用户是否可以正常注册  后台数据接收是否正常 |
| 用户名：hello  密码：hello  确认密码：hello！！  电话：1234567  性别：女 | 注册模块 | 测试注册信息有误时是否可以注册  数据验证是否正常 |
| 跑界面点击start->点击确认，甩动 | 跑步模块 | 测试数据变化是否正常，时间记录是否准确  数据记录是否正常 |
| 跑步模式下点击暂停 | 跑步模块 | 数据记录是否正常 |
| 跑步模式下点击结束，看主页记录 | 跑步模块 | 测试数据是否正常保存读取  后台数据是否记录的读取正常 |
| 滑动约跑界面 | 约跑模块 | 测试约跑信息是否正常显示  后台数据读取是否正常 |
| 学校选择中大，性别选择女，筛选 | 约跑模块 | 测试筛选信息是否正常  后台数据读取是否正常 |
| 点击添加  内容：今晚一起去跑步吧  学校：广外  时间：2015-7-18-20：00:00 | 约跑模块 | 测试信息发布是否正常  后台数据记录读取是否正常 |
| 点击某一约跑条目查看  点击加入 | 约跑模块 | 测试是否可以正常查看约跑消息，是否可以加入约跑  后台记录是否正常 |
| 点击游戏，交替点击left，right | 游戏模块 | 测试游戏使用是否正常，倒计时是否正常 |
| 点击我->个人信息 查看  点击修改 修改信息 | 个人信息模块 | 测试个人信息读取是否正确，是否可以正常修改，修改后信息更新是否正确  后台数据记录读取是否正确 |
| 点击我参与的约跑，发起的约跑 | 个人信息模块 | 测试个人参与记录是否正确  后台数据记录读取是否正确 |
| 点击关于 | 个人信息模块 | 测试系统显示信息是否正确 |

|  |  |  |
| --- | --- | --- |
| 用例 | 结果截图 | 备注 |
| 点击campus run图标 |  | 欢迎页面显示正常，登陆页面跳转显示正常  前端显示正常 |
| 用户名：sherry  密码：sherry |  | 已注册用户可以正常登陆  后台数据验证正常 |
| 用户名：hello  密码：hello  确认密码：hello  电话：1234567  性别：女 |  | 新用户可以正常注册  后台数据接收正常 |
| 用户名：hello  密码：hello  确认密码：hello！！  电话：1234567  性别：女 |  | 注册信息有误时不可以注册  数据验证正常 |
| 跑界面点击start->点击确认，甩动 |  | 数据变化正常，时间记录准确  数据记录正常 |
| 跑步模式下点击暂停 |  | 数据记录正常 |
| 跑步模式下点击结束，看主页记录 |  | 数据正常保存读取  后台数据记录的读取正常 |
| 滑动约跑界面 |  | 约跑信息正常显示  后台数据读取正常 |
| 学校选择中大，性别选择女，筛选 |  | 筛选信息正常  后台数据读取正常 |
| 点击添加  内容：今晚一起去跑步吧  学校：广外  时间：2015-7-18-20：00:00 |  | 信息发布正常  后台数据记录读取正常 |
| 点击某一约跑条目查看  点击加入 |  | 可以正常查看约跑消息，可以加入约跑  后台记录正常 |
| 点击游戏，交替点击left，right |  | 游戏使用正常，倒计时正常 |
| 点击我->个人信息 查看  点击修改 修改信息 |  | 个人信息读取正确，可以正常修改，修改后信息更新正确  后台数据记录读取正确 |
| 点击我参与的约跑，发起的约跑 |  | 个人参与记录正确  后台数据记录读取正确 |
| 点击关于 |  | 系统显示信息正确 |