|  |  |  |
| --- | --- | --- |
| **文件状态**  **[√] 草稿**  **[ ] 讨论稿**  **[ ] 正式发布** | **文档类型** | **软件详细设计文档** |
| **文件标识** |  |
| **版 本** | **1.0** |
| **作 者** | **颜振松、易万鑫、王驰、高浩然、刘延钊、崔玉涛** |
| **完成日期** | **2016.4.17** |

项目名称： Running cat

文档名称：软件详细设计文档

文档修订

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **日期** | **更改人** | **描述（注明修改的条款或页）** |
| 1.0 | 2016.4.17 | 颜振松、易万鑫 | 初始版本 |
| 2.0 | 2016.4.26 | 王驰 | 接口功能增加 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

批准人签字

|  |  |  |
| --- | --- | --- |
| **职务** | **姓名** | **日期** |
| 项目经理 | 颜振松 | 2016.4.17 |
| 客户经理 | 鲁悦 | 2016.4.17 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[1. 引言 5](#_Toc448761062)

[1.1 编写目的和范围 5](#_Toc448761063)

[1.2 术语表 5](#_Toc448761064)

[1.3 参考资料 5](#_Toc448761065)

[1.4 使用的文字处理和绘图工具 5](#_Toc448761066)

[2. 模块设计 6](#_Toc448761067)

[2.1 用例图 6](#_Toc448761068)

[2.2 功能设计说明 6](#_Toc448761069)

[2.2.1 注册登录模块 6](#_Toc448761070)

[2.2.2 跑步实时记录模块 11](#_Toc448761071)

[2.2.3 宠物养成模块 15](#_Toc448761072)

[2.2.4 上传分享模块 17](#_Toc448761073)

[2.2.5 个人信息模块 19](#_Toc448761074)

[3. app接口设计 21](#_Toc448761075)

[3.1 用户注册界面 21](#_Toc448761076)

[3.2 用户登陆界面 22](#_Toc448761077)

[3.3 个人界面 23](#_Toc448761078)

[3.4 资料界面 23](#_Toc448761079)

[4. 数据库设计 24](#_Toc448761080)

[4.1 数据库概念设计 24](#_Toc448761081)

[4.2 数据库逻辑设计 24](#_Toc448761082)

[4.3 数据字典 25](#_Toc448761083)

# 引言

## 编写目的和范围

在软件概要设计的基础上，对软件做详细设计，主要解决实现该系统需求的程序模块详细设计及实现问题。在下一阶段的编程实现中，编程人员可以参考此详细设计，在详细设计对软件所做的模块结构设计的基础上，对软件进行编程实现。在以后的软件测试以及维护阶段也可以参考此说明书，以便于了解在详细设计过程中所完成的各模块设计结构，或在修改时找出在本阶段设计的不足之处。

## 术语表

|  |  |  |
| --- | --- | --- |
| **序号** | **术语或缩略语** | **说明性定义** |
| 1 | json | 一种轻量级的数据交换格式。 |
| 2 |  |  |
|  |  |  |

## 参考资料

|  |  |  |  |
| --- | --- | --- | --- |
| 资料名称 | 作者 | 文件编号、版本 | 资料存放地点 |
|  |  |  |  |

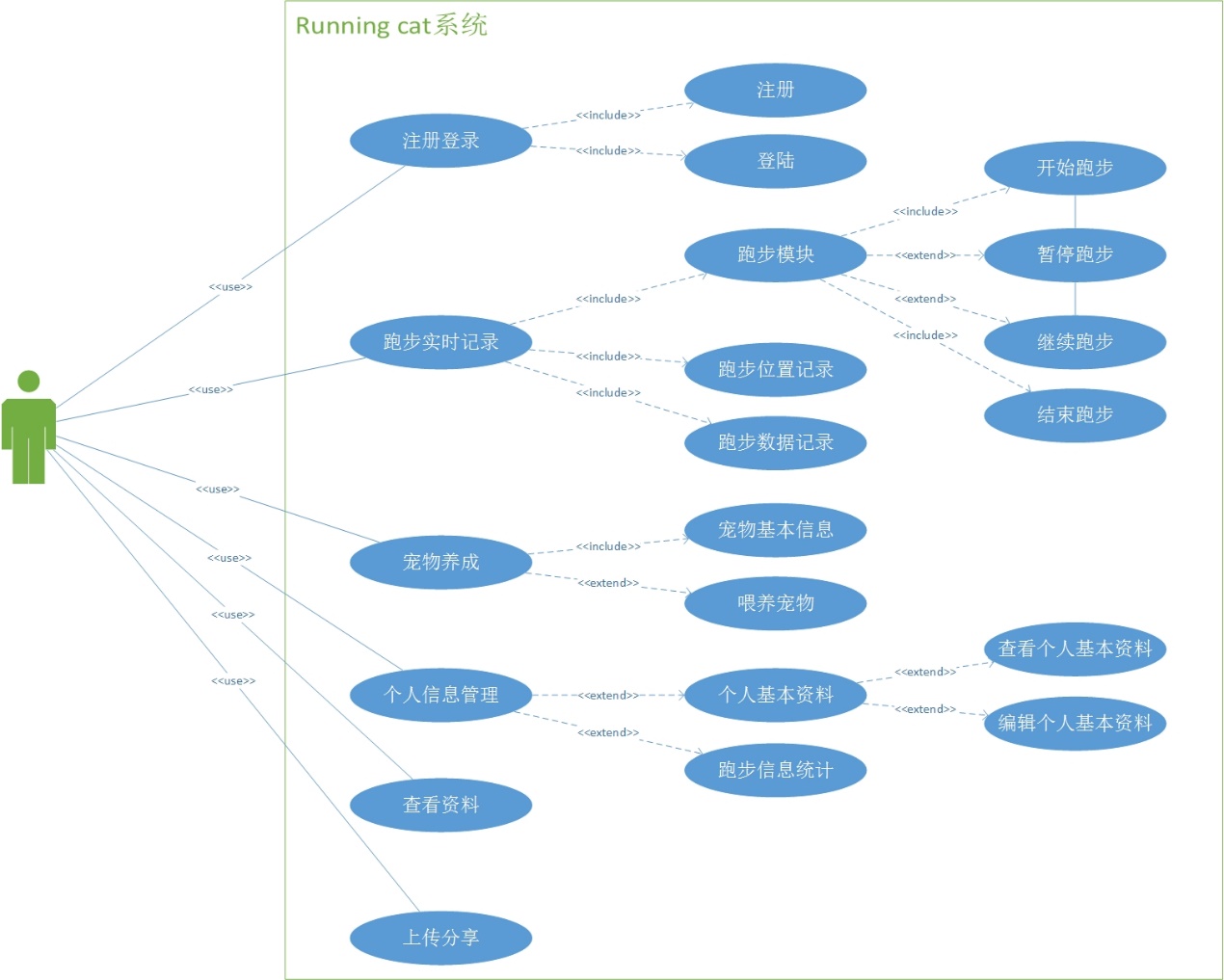
## 使用的文字处理和绘图工具

文字处理软件：word

绘图工具：在线画图工具processon

# 模块设计

## 用例图



## 功能设计说明

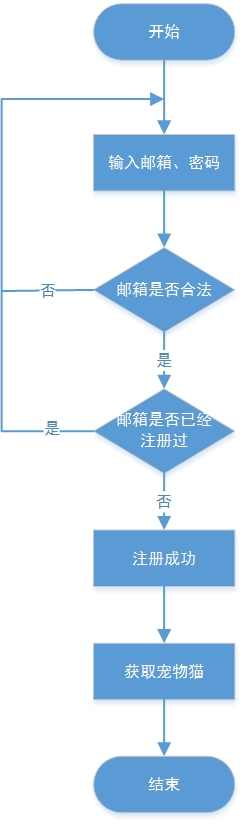
### 注册登录模块

第一个模块是注册登录模块，主要实现用户的注册和登录功能。

#### 注册模块

将这一模块分为注册和登录两个模块，然后这是注册模块。

##### 注册流程图



##### 功能描述

用户通过邮箱，输入密码进行注册，若注册邮箱合法且之前未被注册过，则注册成功，并获得一只宠物猫。

##### 输入数据

用户输入自己要注册的邮箱和密码进行注册，然后后台数据库遍历数据表，对输入的邮箱进行检测，判断邮箱号是否合法以及是否被注册，如果没有，则注册成功。

##### 输出数据

当注册成功之后，系统会给出注册成功的信息，并且将用户的邮箱和密码保存在数据库中。

这个时候用户得到一只宠物猫，宠物猫有等级，饥饿度，经验值，这些数据将以数值的形式显示出来。

##### 业务算法和流程

用户输入注册信息之后，数据库对用户输入的账户信息在数据表中进行检索，如果用户的注册信息正确，那么将用户输入的注册信息保存到数据库中。

##### 数据设计

create table User(

nickname varchar(20),

catExp integer,

catFood integer,

id char(20) primary key,

createDate date,

password varchar(50),

);

如图，设计的数据表中包括用户的昵称、账户、密码、宠物信息。当用户注册成功的时候，将在数据库中自动生成这些数据

##### 源程序文件说明

此模块主要是通过RegisterActivity类来实现，这个类继承自Activity类，实现用户的信息注册功能。

##### 函数说明

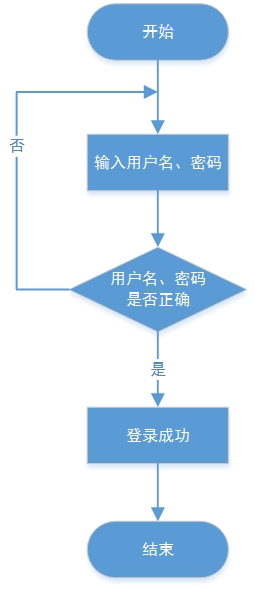
* 1. RegisterActivity 类用来加载布局文件和实现同服务器数据的交换。

register.xml 是布局文件，显示布局中应有的控件。以TableLayout的布局形式来显示 Space, TextView, EditText, Button等控件。

* 1. onCreate 方法用来显示布局和交换数据。

#### 登录模块

##### 登录流程图



##### 功能描述

用户输入用户名和密码登录，查询数据库是否存在该用户，并且用户名和密码均正确，则登录成功。

##### 输入数据

用户输出自己的用户名和密码，然后后台数据库对用户表进行遍历，检测输入的用户名和密码，如果用户名和密码匹配，判断用户名和密码是否正确。

##### 输出数据

如果用户名和密码正确，那么系统会提示登录成功，并跳转进入系统主页，如果用户名和密码错误，那么系统将会直接提示用户名或密码错误，并且不会登录。这个时候，用户将会得到个人的所有信息。

##### 业务算法和流程

用户输入登录信息之后，数据库在数据表中进行检索，如果用户的登录信息和数据表匹配，那么数据库将会将用户的个人信息返回给前端，于是用户就能得到自己的所有信息。

##### 数据设计

create table User(

age integer,

height integer,

weight float,

sex char(1),

nickname varchar(20),

location varchar(20),

catExp integer,

catFood integer,

id char(20) primary key,

createDate date,

password varchar(50),

allDist float

);

在用户登录成功的时候，将会得到上面数据表中设计的所有数据，包括用户的年龄，体重，身高，性别，昵称，账户，密码，宠物信息，跑步日期以及跑步的距离。并且也会知道得到这些数据的数据类型。

##### 源程序文件说明

此模块主要是通过LoginActivity类来实现，这个类继承自Activity类，实现用户登录功能。

##### 函数说明

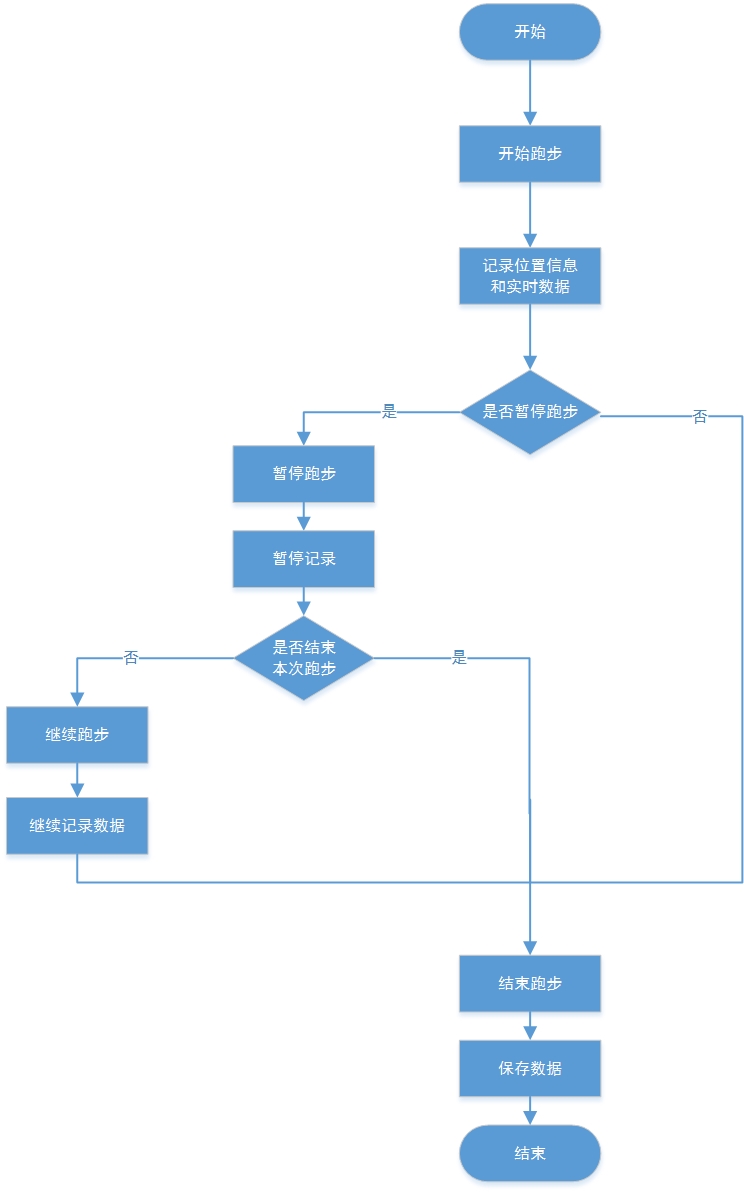
* 1. LoginActivity 类用来加载布局文件和实现同服务器数据的交换。

Login.xml 是布局文件，用来显示该布局中的各种控件。以TableLayout的布局形式来显示 Space, TextView, EditText, Button等控件。

* 1. onCreate 方法用来显示Login中的布局和交换各种数据。

### 跑步实时记录模块

#### 跑步记录模块



##### 功能描述

用户开始跑步，同时开始记录位置信息和实时数据。跑步过程中，用户可以选择暂停跑步（PAUSE），同时暂停记录数据，之后可以选择继续跑步，或者直接结束本次跑步。

##### 输入数据

用户在开始跑步的时候，点击Start按钮开始跑步，与此同时，当人在移动的时候，根据GPS定位，从而实时记录人跑步的时间、轨迹和路程。用户如果想暂停跑步，那么点击暂停按钮，此时系统将会停止记录用户的跑步信息。当用户点击stop按钮的时候，将会停止跑步，这个时候将会得到用户此次跑步的总距离和总时间。

##### 输出数据

系统将会显示用户跑步的距离、花费的时间以及运动的轨迹。跑步的距离和时间将以数值的形式显示，而运动的轨迹将会在地图中以连线的形式表示出来。

##### 数据设计

用户此次跑步的数据首先是会被系统记录下来，然后上传到数据表，这个时候，数据表得到信息之后，将会更新，从而记录一个用户跑步总距离和总次数的最新值。

create table User(

createDate date,

allDist float

);

数据表中这两个数据就是分别表示用户跑步的总距离和总时间。

##### 业务算法和流程

根据GPS定位，得到用户跑步的距离和花费的时间，系统将这些数据传给数据库，数据库在数据表中找到对应的数据，然后将其进行运算，得到新的数据，再将数据传给前端，从而系统得到数据更新，并显示在界面。

##### 源程序文件说明

这个模块由MapActivity , GsonSevice , RealtimeTrackDate 三个avtivity类和map.xml , map1.xml两个布局文件来实现。

（1）map.xml和 map1.xml 是活动中的两个不同的布局：

map.xml是一个LinearLayout的总体布局。其中包括EditText，MapView，Chronometer，ImageButton等四种控件。

map1.xml是一个LinearLayout的总体布局。其中包括EditText , MapView , Chronometer, ImageButton等四种控件。

（2）GsonSecive：用来解析Json.

（3）RealtimeTrackDate: 用来存储轨迹的各种数据。

其中的全局变量：

**public int status**; *// 状态码，0为成功***public int size**; *// 返回结果条数，该页返回了几条数据***public int total**; *// 符合条件结果条数，一共有几条符合条件的数据***public** List<Entities> **entities**;  
**public** String **message**; *// 响应信息,对status的中文描述*

**Entities里的局部变量：**

**public** String **create\_time**; *// 创建时间 格式化时间 该时间为服务端时间***public** String **modify\_time**; *// 修改时间***public** RealtimePoint **realtime\_point**; *// 实时轨迹信息*

RealtimePoint中的局部变量：

**public** List<Double> **location**;*// 经纬度 Array 百度加密坐标***public** String **loc\_time**;*// 该track实时点的上传时间 UNIX时间戳 该时间为用户上传的时间*

（4）MapAcitivity 用来显示map 和map1的布局，以及实现各种数据的交换。

其中的全局变量：

**private** Chronometer **timer**;  
**int miss** = 0;  
**int gatherInterval** = 1; *//位置采集周期 (s)***int packInterval** = 10; *//打包周期 (s)*String **entityName** = **null**; *// entity标识***long serviceId** = 114202;*// 鹰眼服务ID***int traceType** = 2; *//轨迹服务类型 traceType - ( 0 : 不建立长连接, 1 : 建立长连接但不采集数据, 2 : 建立长连接并采集数据 )***private static** OnStartTraceListener *startTraceListener* = **null**; *//开启轨迹服务监听器***private static** MapView *mapView* = **null**;  
**private static** BaiduMap *baiduMap* = **null**;  
**private static** OnEntityListener *entityListener* = **null**;  
**private** RefreshThread **refreshThread** = **null**; *//刷新地图线程以获取实时点***private static** MapStatusUpdate *msUpdate* = **null**;  
**private static** BitmapDescriptor *realtimeBitmap*; *//图标***private static** OverlayOptions *overlay*; *//覆盖物***private static** List<LatLng> *pointList* = **new** ArrayList<LatLng>(); *//定位点的集合***private static** PolylineOptions *polyline* = **null**; *//路线覆盖物***private** Trace **trace**; *// 实例化轨迹服务***private** LBSTraceClient **client**; *// 实例化轨迹服务客户端*

##### 函数说明

该类中每个函数的作用：

// 用来加载布局，用来实现不同布局之间的切换。用来实现点击事件。用来实现地图轨迹服务和计时功能

// 用来实现同服务器之间数据的交换

**protected void** onCreate(Bundle savedInstanceState；

//用来实现从布局1切换到布局2

**public void** jumpToLayout2()；

//用来实现从布局2切换到布局1

**public void** jumpToLayout1()；

//初始化各个参数

**private void** init()；

*// 将秒转化成小时分钟秒***public** String FormatMiss(**int** miss)；

//初始化设置实体监听器

**private void** initOnEntityListener()；

//百度地图鹰眼轨迹追踪开始。

**private void** initOnStartTraceListener()；

//轨迹刷新线程

**private class** RefreshThread **extends** Thread；

//查询实时线路

**private void** queryRealtimeTrack()；

//展示实时线路图,数据以Json的形式存取

**protected void** showRealtimeTrack(String realtimeTrack)

//画出实时线路点，连接成线

**private void** drawRealtimePoint(LatLng point)

//启动刷新线程

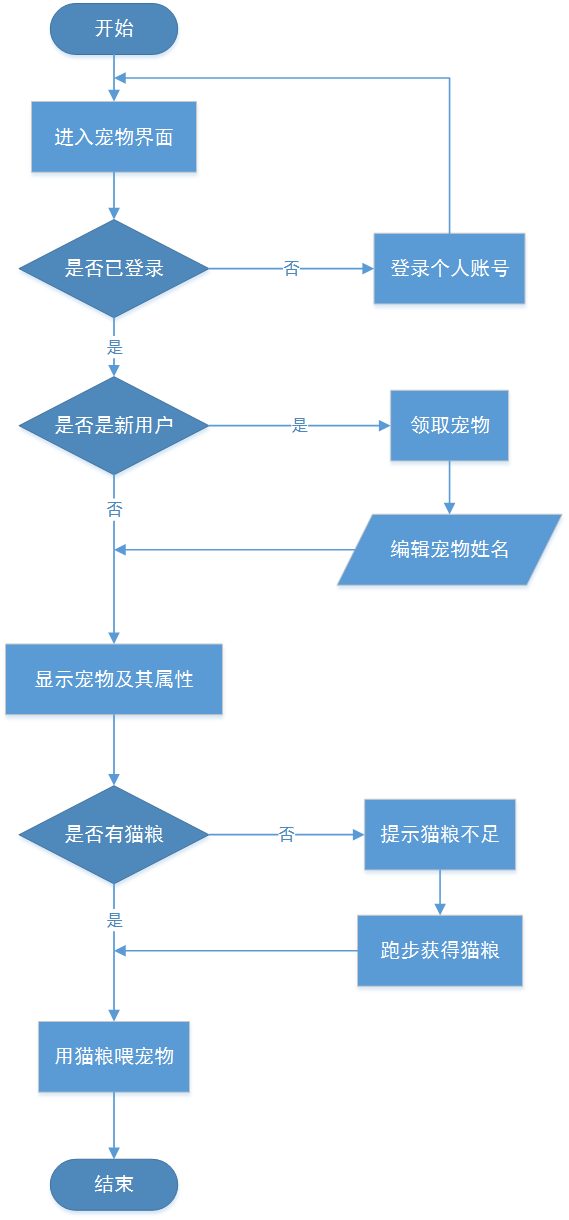
**private void** startRefreshThread(**boolean** isStart)

//获取手机的Imei码，作为实体对象的标记值

**private** String getImei(Context context)

### 宠物养成模块

##### 宠物养成模块



##### 功能描述

新用户第一次注册登陆成功之后，进入主界面，其中会出现宠物挑选，其中有许多种类的小猫供用户挑选。用户可以根据自己的喜好选取一只进行饲养。以后本界面就会出现这只小猫的家，小猫会对主人撒娇和对话。同时旁边也会显示小猫的姓名和成长状态，即以柱状图和数值方式显示饥饿度、经验值、等级。在该界面也会以图像和数值的方式显示猫粮剩余量，当没有猫粮时会提醒跑步换取猫粮。

##### 输入数据

首先用户在成功注册账号的时候即可获得一只宠物猫。在使用系统的过程中，用户可以获得猫粮，然后可以选择输入喂给宠物的猫粮数量，从而使得自己的宠物成长。

##### 输出数据

用户可以得到当前宠物猫的饥饿度、经验值、等级，然后这些数据将会以图像和数值的方式显示出来。当用户给宠物喂食的时候，宠物的饥饿度、经验值以及等级会增加，不喂食的时候饥饿度会下降。当经验值满的时候，显示宠物升级，与此同时用户界面上的经验值置为0。

##### 业务算法和流程

用户输入需要喂给宠物的猫粮数量，数据库获得这个数值之后将会和原先数据库中宠物信息的数值进行运算，最好得到一个新的宠物信息数值，然后将这个数值发送给前端，从而系统得到新的数据。

##### 数据设计

create table User(

catExp integer,

catFood integer,

);

这两个分别表示宠物的经验值和宠物的饥饿度，这两个以数字的形式在宠物界面显示，当喂食或者宠物成长的时候，这些值就会随着发生改变。

##### 源程序文件说明

宠物养成模块主要由FirstActivity这个类实现，这个类所在的程序包含在Running\_cat1项目下的com.example.lucas.running\_cat1中。

##### 函数说明

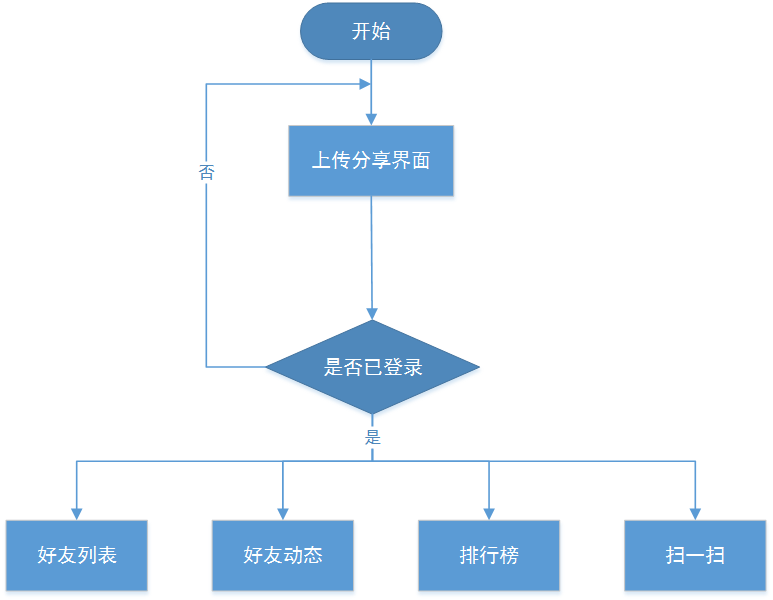
* 1. FirstActivity 类用来加载布局文件和实现同服务器数据的交换。

one.xml 是布局文件，显示布局中应有的控件。以TableLayout的布局形式来显示 Space, TextView, EditText, Button等控件。

* 1. onCreate 方法用来显示布局和交换数据。

### 上传分享模块

##### 上传分享模块



##### 功能描述

用户通过该模块可以查阅一些跑步时的注意事项和规范动作，还可以查看一些跑友的跑步心得和与跑步相关的文章，新闻等等。

##### 输入数据

用户只需要登录账户，进入上传分享模块之后，输入自己想要分享的文章、心得、动态，然后分享上传，这样用户的好友就可以看到你分享的东西了。

##### 输出数据

用户登录之后，在上传分享模块就可以得到好友列表，好友最新的动态，以及跑步的排行榜等信息。其中主界面将会以列表的形式存放最新的动态信息。

##### 源程序文件说明

上传分享模块主要由SecondActivity这个类实现，这个类继承自Activity类，这个模块我们目前还处于待开发的状态。这些文件都是在Running\_cat1项目下的com.example.lucas.runnng\_cat1中，目前主要用来推送最新动态。

##### 函数说明

在这个类中，主要有以下两个函数，这两个函数主要包含在SecondActivity类中。

protected void onCreate(Bundle saveInstanceState){

}

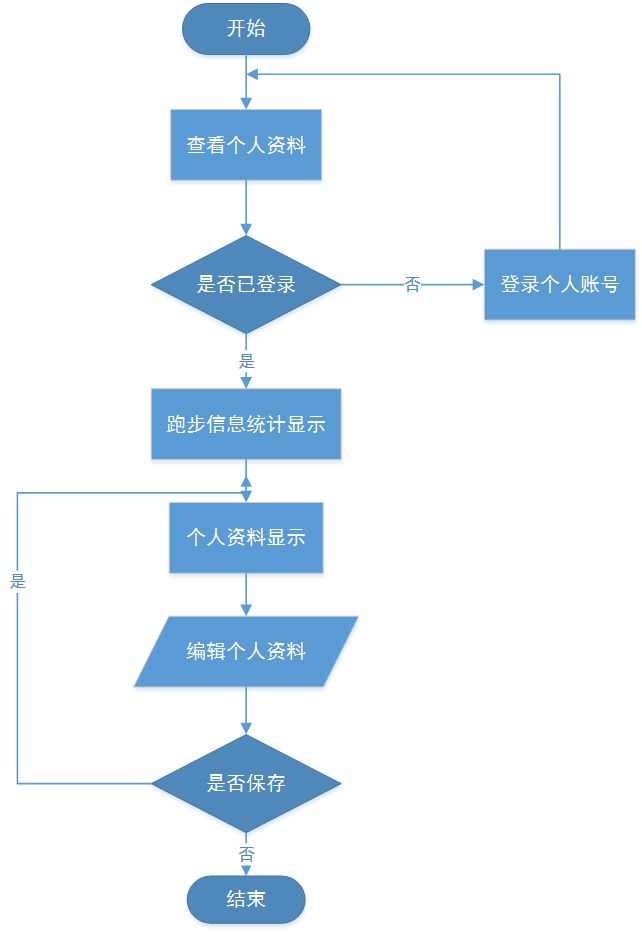
这个函数是当前类的主要函数onCreate函数其作用是创建一个activity，并且在activity中进行其他控件和活动的操作。

public void onClick(View v)

这个函数的功能是使用显示Intent来打开新的活动。

### 个人信息模块

##### 个人信息模块



##### 功能描述

用户登录账号后，可进入个人信息模块，查看个人基本资料，也可编辑个人资料并保存修改。同时在该模块可以显示用户之前的跑步信息，主要包括：累计距离、累计时间、平均时速。

##### 输入数据

用户只需要输入登录信息之后，在个人信息界面即可得到其当前的所有个人信息资料。

##### 输出数据

输出信息包括用户的个人资料，如身高、体重、昵称、性别等，以及用户跑步的累计距离、累计时间、平均时速。这些数据将以数子的形式直接在当前界面显示。

##### 业务算法和流程

整个数据的流程大致是：用户登录账号信息，然后数据库进行验证信息，验证成功之后，数据库将个人信息发送给前端，系统得到个人资料信息。

用户修改个人资料：用户登录到界面的时候，如果需要修改个人资料，那么在个人资料界面修改成功之后，用户选择是否保存，将直接保存上传到数据库。

##### 数据设计

create table User(

age integer,

height integer,

weight float,

sex char(1),

location varchar(20),

catExp integer,

catFood integer,

createDate date,

allDist float

);

这个数据表描述了个人信息界面所能得到的所有数据的名称和类型。

##### 源程序文件说明

个人信息模块主要由FifthActivity这个类实现，这个类继承自Activity这个类，所有Java文件包含在com.example.lucas.running\_cat1包里面。然后布局文件在Rnning\_cat1项目的layout中，然后在类中定义了几个button，链接到新的activity，有比如fouth\_heightactivity，fouth\_weightactivity，fouth\_placeactivity等，这几个新的activity中包含的是用户的身高、体重、所在地等数据。每一项数据对应一个activity类，然后这些数据由ListView控件来实现点击和更改，当用户需要对其个人资料进行更改的时候，只需要在ListView对其所想要更改的数据进行点击，即可响应点击事件。

##### 函数说明

在这个类中，主要有以下两个函数，这两个函数主要包含在FifthActivity类中。

protected void onCreate(Bundle saveInstanceState){

}

这个函数是当前类的主要函数onCreate函数其作用是创建一个activity，并且在activity中进行其他控件和活动的操作。

public void onClick(View v)

这个函数的功能是使用显示Intent来打开新的活动。

# app接口设计

注：传参一律用post方式，key=request，value为一串json格式字符串。格式如：

request={

“para”:{

“id”:”id”,

“password”:”1234”,

}

}

para:请求体，客户端请求所需要传入的参数，依据不同的接口而定。

## 用户注册界面

请求地址：http://159.203.225.151/RunningCat/register.php

request={

“para”:{

“id”:”mail”, (由用户输入)

“password”:”1234”, (由用户输入)

“rePassword”:”1234” (由用户输入)

}

}

返回信息如下：

{

“head”:{

“code”:”000”,

“msg”:”注册成功”

}

}

{

“head”:{

“code”:”001”,

“msg”:”注册失败”

}

}

## 用户登陆界面

请求地址：http://159.203.225.151/RunningCat/login.php

request={

“para”:{

“id”:”mail”,

“password”:”password”

}

}

返回信息:

{

“head”:{

“code”:”002”,

“msg”:”登录失败”

}

}

{

“head”:{

“code”:”003”,

“msg”:”登录成功”

}

“para”:{

“id”:”id”,

“createDate”:””,

“allDist”:””,

“location”:””,

“nickname”:””,

“sex”:””,

“weight”:””,

“height”:””,

“age”:””,

“catFood”:catFood,

“catExp”:catExp

}

}

## 个人界面

请求地址：http://159.203.225.151/RunningCat/person.php

request={

“para”:{

“id”:id,

“catFood”:catFood,

“catExp”:catExp

}

}

返回信息

{

“head”:{

“code”:”006”,

“msg”:”修改成功”

}

}

{

“head”:{

“code”:”007”,

“msg”:”修改失败”

}

}

## 资料界面

请求地址：http://159.203.225.151/RunningCat/news.php

request={

“para”:{

“getnews”:””

}

}

返回信息:

{

“head”:{

“code”:”004”,

“msg”:”获取失败”

}

}

{

“head”:{

“code”:”005”,

“msg”:”获取成功”

}

“para”:{

“newsnames”:””,

“newsurls”:””

}

}

## 好友界面

请求地址：<http://159.203.225.151/RunningCat/friend.php>

//好友列表

request={

“para”:{

“getfriendlist”:true

}

}

返回信息:

{

“head”:{

“code”:”006”,

“msg”:”获取失败”

}

}

{

“head”:{

“code”:”007”,

“msg”:”获取成功”

“friendlist”:””

“newfriendlist”:””

}

}

//好友请求

request={

“para”:{

“sendrequest”:””

}

}

返回信息:

{

“head”:{

“code”:”008”,

“msg”:”发送失败”

}

}

{

“head”:{

“code”:”009”,

“msg”:”发送成功”

}

}

//接受好友

request={

“para”:{

“id”:””

“accept”:true/false

}

}

返回信息:

{

“head”:{

“code”:”010”,

“msg”:”发送失败”

}

}

{

“head”:{

“code”:”011”,

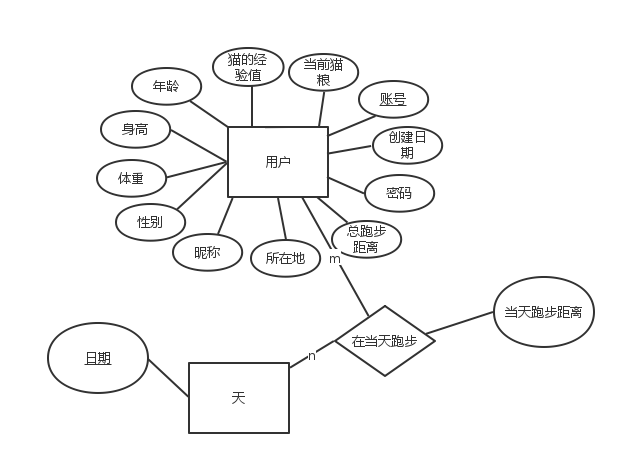
“msg”:”发送成功”

}

}

# 数据库设计

4.1 数据库概念设计



4.2 数据库逻辑设计

用户（帐号，创建日期，密码，总跑步距离，所在地，昵称，性别，体重，身高，年龄，当前猫粮，猫的经验值）

在当天跑步（用户帐号，日期，当天跑步距离）

天（日期）

4.3 数据字典

4.3.1

数据项名称：年龄

含义说明：标识用户年龄大小

数据表中名称：age

类型：integer

取值范围：1-99

4.3.2

数据项名称：身高

含义说明：标识使用者身高

数据表中名称：height

类型：integer

取值范围：0-300

取值含义：数值单位为cm。

4.3.3

数据项名称：体重

含义说明：标识用户身体重量

数据表中名称：weight

类型：float

取值范围：0-999

取值含义：数值单位为kg。

4.3.4

数据项名称：性别

含义说明：标识用户性别

数据表中名称：sex

类型：char

长度：1

取值范围：’男’或’女’

4.3.5

数据项名称：昵称

含义说明：用户为自己取另一个称呼

数据表中名称：nickname

类型：char

长度：20

取值范围：只能由汉字、数字、字母组成

4.3.6

数据项名称：所在地

含义说明：标识用户所在地区

数据表中名称：location

类型：char

长度：20

取值范围：XX省XX市XX区

4.3.7

数据项名称：账号

含义说明：唯一标识用户的号码，是该用户的邮箱

数据表中名称：id

类型：char

长度：50

取值范围：符合邮箱格式

4.3.8

数据项名称：创建日期

含义说明：标识用户注册账户的日期

数据表中名称：createDate

类型：date

4.3.9

数据项名称：密码

含义说明：用户登录的唯一指令，使用md5加密

数据表中名称：password

类型：char

加密前最大长度：20

加密后最大长度：50

取值范围：加密前的密码只能字母和数字由组成

4.3.10

数据项名称：总跑步距离

含义说明：标识用户总共奔跑的距离

数据表中名称：allDist

类型：float

取值含义：随奔跑距离增加，单位为km

4.3.11

数据项名称：当前的猫粮

含义说明：标识当前拥有的猫粮数量

数据表中的名称：catFood

类型：integer

4.3.12

数据项名称：猫的经验值

含义说明：标识用户所拥有的经验值数量

数据表中的名称：catExp

类型：integer

4.3.13

数据项名称：当天跑步距离

含义说明：标识当天所奔跑公里数

数据表中的名称：dist

类型：float

取值含义：单位为km

4.3.14

数据项名称：日期

含义说明：标识当天的日期

数据表中名称：date

类型：date