



微博项目开发文档



中南大学信息院 China 队

2012 年 12 月 21 日

1. 引言.....	4
1.1 编写目的.....	4
1.2 项目背景.....	4
1.3 项目思路.....	4
1.4 参考资料.....	5
2. 任务概述.....	6
2.1 项目总体目标.....	6
2.2 需求概述.....	6
2.3 运行环境.....	7
2.4 开发环境.....	7
3. 相关知识及技术	8
3.1 ANDROID 介绍	8
3.2 OAUTH2.0 验证机制介绍	9
3.2.1 OAUTH2.0 认证授权步骤.....	9
3.2.2 OAUTH2.0 的四角色.....	9
3.2.3 OAUTH2.0 认证授权流程.....	10
3.3 JSON 介绍	11
3.4 图片编辑器 AVIARY 介绍	12
3.6 讯飞语音介绍	13
3.6.1 语音合成.....	13
3.6.2 语音识别.....	14
3.7 摇一摇相关技术介绍.....	14
3.7.1 ANDROID 的传感器.....	14
3.7.2 加速度传感器介绍	15
4.总体设计	16
4.1 总体界面设计	16
4.1.1 登录授权界面设计	16
4.1.2 用户功能界面设计	16

4.2 总体功能设计	17
5. 功能详细设置	20
5.1 用户登录授权	20
5.1.1 功能说明	20
5.1.2 功能结构	20
5.1.3 核心代码	20
5.1.4 用户界面	21
5.2 微博功能	22
5.2.1 功能说明	22
5.2.2 功能结构	22
5.2.3 核心代码	22
5.2.4 用户界面	28
5.3 个人信息	31
5.3.1 功能说明	31
5.3.2 功能结构	31
5.3.3 核心代码	31
5.3.4 用户界面	32
5.4 摇一摇	33
5.4.1 功能说明	33
5.4.2 功能结构	33
5.4.3 核心代码	34
5.4.4 用户界面	36
5.5 系统设置	37
5.5.1 功能说明	37
5.5.2 功能结构	37
5.5.3 核心代码	37
5.5.4 用户界面	40
5.6 退出应用	41
5.6.1 功能说明	41
5.6.2 功能结构	41
5.6.3 核心代码	41
5.6.4 用户界面	42

1. 引言

1.1 编写目的

中南大学信息科学与工程学院结合本院特色推出首届中南大学“IT 精英文化艺术节”。此次活动旨在拓展学生视野，提高学生的网络设计、软件开发、编程设计能力，发掘学生日常思维创新潜力，增加学生思维活跃性，真正做到思维与创新源于生活而服务于生活，为中南学子将来走向社会打下好良好基础。此次活动于 11 月 28 日上午隆重开幕，本团队参加了本次比赛，并且一致选择了参赛题目中南软件方向中的第二题“新浪微博 Android 客户端（Java 开发）”，此篇微博开发文档正是我们团队开发的微博的项目开发文档。

1.2 项目背景

随着经济的发展，各大城市尤其是一线城市的生活节奏明显加强，快节奏成为了人们的生活习惯，这就为微博 140 字的发展提供了很大空间。微博是一种非正式的迷你型博客，是一种可以即时发布消息的类博客系统。微博本质是一种开放的互联网社交服务，但它又不同于社交媒体的满足用户相当价值内容，它满足用户绝对价值内容。同时，移动终端的迅速发展为微博的发展推波助澜。虽然短短几年的发展时间，从名人到普通百姓，许多人都已经形成了一种微博意识，通过互粉成为熟人，然后分享或发布第一时间消息，现在微博整合各方面资源，与传统媒体交融，向社交媒体学习，微博逐渐发展成为了一种媒体+社交的形态。本团队即在微博的基础上进行整合，挖掘出更多的微博功能，开发出更好用更美观更符合大众需求的微博。

1.3 项目思路

应用程序主要是针对的新浪提供的 SDK 而展开的，再深入分析用户可能会用到的功能和用户喜欢的界面 UI 设计，实现新浪微博 Android 版功能，使用户体验更加丰富

和方便，并且在后续版本开放中，可以充分发挥 Android 平台的优势，开发出更多和新浪微博相关的各种插件服务功能。

在应用的开发中，主要要弄清是如何请求数据和发送数据的，对 API 接口需要从最初了解其功能到最终的掌握其方法。考虑到是即时通讯应用软件，并且微博更新的速度较快，主要侧重于“即时收发数据”，并且做到数据发送的完整和迅速。在整个系统中没有设置数据库来存放用户数据，只是在系统中设置了存放系统配置的文件。

1.4 参考资料

- [1].《疯狂 Android 讲义》 李刚 电子工业出版社 2011 年
- [2].《Android 应用开发实战》 李宁 人民邮电出版社 2011 年
- [3].《Google Android SDK 开发范例大全》 余志龙等 人民邮电出版社 2010 年

2. 任务概述

2.1 项目总体目标

本项目的总体目标是开发一个新浪微博 Android 客户端。对于一个 Android 版客户端，针对的是广大的 Android 手机用户，首先要从用户的角度出发，思考哪些是用户所需要的内容，哪些是用户最关心的内容，哪些是最吸引用户的内容等。首先以新浪微博所提到的功能为基础，实现新浪微博的基本功能，包括查看微博，评论和转发微博，发表微博等等。在实现了新浪微博的部分功能或者全部功能后，后续版本再进行插件的开发，扩展更多的微博应用，充分利用微博所提供的用户关心网络，满足用户的需求，探索更多的有价值的插件。本项目突出特色即在满足微博基本功能的基础上，添加了用户所感兴趣的功能，如摇一摇微博（包括摇附近微博，摇好友微博，摇心情微博），语音微博，图像处理功能，达到用户喜爱的目的。

2.2 需求概述

能够稳定运行于当今主流的几种版本的 Android 操作系统上，新浪微博 Android 版主要实现了下面的功能：

1. 登陆验证(Oauth2.0)
2. 用户功能界面的设计
3. 用户公共微博列表
4. 用户好友微博列表
5. 用户微博列表
6. @提到我的微博列表
7. 对指定微博的转发
8. 对指定微博的评论

9. 下拉刷新微博功能
10. 发文字微博，发图片微博，发文字加图片的微博
11. 语音微博，包括语音写微博、语音读微博
12. 对微博图片的处理功能
13. 获取用户的信息并在个人信息界面上显示
14. 摇周边的动态
15. 摇附近朋友
16. 摇好友微博
17. 关注作者微博
18. 开启/关闭声音效果
19. 查看版本更新
20. 用户意见反馈
21. 关于作者团队信息
22. 清空缓存图片
23. 退出登录
24. 欢迎界面

2.3 运行环境

运行 Android 操作系统 2.2 版本以上的手机

2.4 开发环境

开发语言使用 Java，开发工具使用 Eclipse，安装 ADT 插件，同时需要 Android SDK

3. 相关知识及技术

3.1 Android 介绍

Android 是 Google 开发的基于 Linux 平台的开源手机操作系统。它包括操作系统、用户界面和应用程序—移动电话工作所需的全部软件,而且不存在任何以往阻碍移动产业创新的专有权障碍。Google 与开放手机联盟合作开发了 Android,这个联盟由包括中国移动、摩托罗拉、高通、宏达电和 T-Mobile 在内的 30 多家技术和无线应用的领军企业组成。Google 通过与运营商、设备制造商、开发商和其他有关各方结成深层次的合作伙伴关系,希望借助建立标准化、开放式的移动电话软件平台,在移动产业内形成一个开放式的生态系统。

Android 的优势及特点

- 应用程序框架支持组建的重用与替换;
- Dalvik 虚拟机专门为移动设备做了优化;
- 内部集成浏览器该浏览器基于开源的 WebKit 引擎;
- 优化的图形库包括 2D 和 3D 图形库, 3D 图形库基于 OpenGL ES 1.0;
- SQLite 用作结构化的数据存储;
- 多媒体支持包括常见的音频、视频和静态印象文件格式;
- GSM 电话 (依赖于硬件);
- 蓝牙 Bluetooth, EDGE, 3G and WiFi (依赖于硬件);
- 照相机, GPS, 指南针, 和加速度计 (依赖于硬件);
- 丰富的开发环境包括设备模拟器, 调试工具, 内存及性能分析图表, 和 Eclipse 集成开发环境插件。

3.2 OAuth2.0 验证机制介绍

3.2.1 OAuth2.0 认证授权步骤

OAuth2.0是 OAuth 协议的下一版本，很可能是下一代的“用户验证和授权”的标准。

OAuth2.0 关注于使应用开发更加简易安全，验证流程支持网站、手机端应用的开发。

OAuth2.0认证授权三个步骤：

1. 请求授权 Authorization Request
2. 出示访问许可，请求访问令牌 Access Grant&Client Credentials
3. 出示访问令牌 Access Token，请求资源

当应用拿到 Access Token 后，就可以有权访问用户授权的资源了。这三个步骤是对应 OAuth2.0 的三个 URL 服务地址。上面的三个步骤中，每个步骤分别请求一个 URL，并且收到相关信息，并且拿到上步的相关信息去请求接下来的 URL 直到拿到 Access Token。

3.2.2 OAuth2.0 的四角色

OAuth2.0 中有四个角色，分别为资源拥有者、资源服务器、第三方客户端以及鉴权服务器。下面就对四个角色进行简要的阐释：

(1)资源拥有者：一个实体能够授权访问一个受保护的资源。当资源拥有者是一个人，它就称为一个终端用户。

(2)资源服务器：用于接收第三方客户端发送的访问令牌，并返回第三方应用所需的相关数据，或返回相关应用业务的结果。

(3) 第三方客户端：第三方应用系统通过开放平台入口，访问内部服务，从而完成跨域的业务整合。例如，第三方开发者为新浪微博开放平台开发的应用“新浪微博登陆”，可以使用自己的域名，独立运行在该第三方服务器上，通过远程 API 调用完成业务功能。

(4) 鉴权服务器：开放平台中用于鉴别第三方客户端业务请求及颁布访问令牌的服务器。

资源服务器与鉴权服务器可以是分离的,也可以是一体的。

3.2.3 OAuth2.0 认证授权流程



(图 OAuth2.0 认证机制流程图)

上图中描述四种角色的交互过程。大致的流程以下流程：

- A.第三方应用向用户发送认证请求，请求资源拥有者给予授权许可
- B.资源拥有根据实际情况，选择对第三方应用授权内容或者授权与否
- C.第三方应用发送资源所有者的授权信息给授权服务器
- D.授权服务器经过校验后，确认有效，给予访问令牌
- E.第三方应用利用获得的访问令牌访问资源服务器，获取资料
- F.资源服务器以 json 或者 XML 的形式把资料传送给第三方应用

3.3 JSON 介绍

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。易于人阅读和编写，同时也易于机器解析和生成。它基于 JavaScript (Standard ECMA-262 3rd Edition - December 1999) 的一个子集。JSON 采用完全独立于语言的文本格式，但是也使用了类似于 C 语言家族的习惯 (包括 C, C++, C#, Java, JavaScript, Perl, Python 等)。这些特性使 JSON 成为理想的数据交换语言。

JSON 可以将 JavaScript 对象中表示的一组数据转换为字符串，然后就可以在函数之间轻松地传递这个字符串，或者在异步应用程序中将字符串从 Web 客户机传递给服务器端程序。这个字符串看起来有点儿古怪，但是 JavaScript 很容易解释它，而且 JSON 可以表示比名称 / 值对更复杂的结构。事实上大部分现代计算机语言都以某种形式支持它们。这使得一种数据格式在同样基于这些结构的编程语言之间交换成为可能。

JSON 具有以下这些形式：

对象是一个无序的“‘名称/值’对”集合。一个对象以“{” (左括号) 开始，“}” (右括号) 结束。每个“名称”后跟一个“:” (冒号)；“‘名称/值’对”之间使用“,” (逗号) 分隔。

数组是值 (value) 的有序集合。一个数组以 “[” (左中括号) 开始，“]” (右中括号) 结束。值之间使用 “,” (逗号) 分隔。

值 (value) 可以是双引号括起来的字符串 (string)、数值 (number)、true、false、null、对象 (object) 或者数组 (array)。这些结构可以嵌套。

字符串 (string) 是由双引号包围的任意数量 Unicode 字符的集合，使用反斜线转义。一个字符 (character) 即一个单独的字符串 (character string)。

字符串 (string) 与 C 或者 Java 的字符串非常相似。

数值 (number) 也与 C 或者 Java 的数值非常相似。除去未曾使用的八进制与十六进制格式。除去一些编码细节。

3.4 图片编辑器 Aviary 介绍

Aviary 是为开发人员提供的一个强大可定制的图片编辑器，可以插入到 iOS 或 Android，Windows Phone 和 HTML5 的应用程序中。

Aviary 照片编辑器是一个强大的手机图形处理软件。为应用提供了 12 种充满乐趣的功能：

(1) 增强：对原图像附加一些信息或变换数据，有选择地突出图像中感兴趣的特征或者抑制(掩盖)图像中某些不需要的特征，使图像与视觉响应特性相匹配

(2) 滤镜 / 特效：主要是用来实现图像的各种特殊效果

(3) 贴纸：给图片加边框、相框、贴些纸照

(4) 裁剪 / 旋转：水平、竖直、正方向、逆方向旋转图片。裁剪是指按照选定的形状对图片进行剪取

(5) 亮度：亮度是颜色的相对明暗程度

(6) 对比度 / 饱和度：对比度指的是一幅图像中明暗区域最亮的白和最暗的黑之间不同亮度层级的测量，差异范围越大代表对比越大，差异范围越小代表对比越小。饱和度，指的其实是色彩的纯度，纯度越高，表现越鲜明，纯度较低，表现则较黯淡

(7) 锐化：锐化就是使图片的局部清晰一些

(8) 手绘：带有涂鸦、描绘图案的画笔工具进行绘制图片

(9) 文字：给图片配上文字信息

(10) 红眼消除：主要针对人物图片，闪光灯正对人眼拍摄时，视网膜底部反光，在照片上眼睛中间出现红点——红眼。

(11) 牙齿美白：对人物图片上的牙齿进行美白处理

(12) 瑕疵消除：从图片上消除不必要的物体

3.6 讯飞语音介绍

语音是人类沟通最自然便捷的方式，在移动互联网时代将带来人机交互的根本性变革。

移动互联网已迅速成为当今世界发展最快、规模最大和市场前景最好的行业，已吸引众多知名 IT 公司进军该领域。由于现有移动终端设备交互方式存在诸多局限，如键盘太小，输入文字不便；屏幕太小，阅读信息不便；以及无法处理特定场景下的交互，如开车和步行情形。语音技术是人机交互最自然的方式，可以给以上缺陷提供完美的解决方法，移动互联网对语音技术有着天然的需求。

本系统应用语音云开放平台向广大用户的开放科大讯飞全球领先的智能语音技术，主要包括语音合成和语音识别，让移动终端应用可随时随地获取优质的语音服务，具备能“听”会“说”的能力。

3.6.1 语音合成

语音合成，又称文语转换技术，它涉及声学、语言学、数字信号处理、计算机科学等多个学科技术，是中文信息处理领域的一项前沿技术，解决的主要问题就是如何将文字信息转化为可听的声音信息，也即让机器像人一样开口说话。我们所说的“让机器像人一样开口说话”与传统的声音回放设备（系统）有着本质的区别。传统的声音回放设备（系统），如磁带录音机，是通过预先录制声音然后回放来实现“让机器说话”的。这种方式无论是在内容、存储、传输或者方便性、及时性等方面都存在很大的限制。而通过计算机语音合成则可以在任何时候将任意文本转换成具有高自然度的语音，从而真正实现让机器“像人一样开口说话”。

文语转换系统实际上可以看作是一个人工智能系统。为了合成出高质量的语言，除了依赖于各种规则，包括语义学规则、词汇规则、语音学规则外，还必须对文字的内容有很好的理解，这也涉及到自然语言理解的问题。文语转换过程是先将文字序列转换成音韵序列，再由系统根据音韵序列生成语音波形。其中第一步涉及语言学处理，例如分词、字音转换等，以及一整套有效的韵律控制规则；第二步需要先进的语音合成技术，能按要求实时合成出高质量的语音流。因此一般说来，文语转换系统都需要一套复杂的文字序列到音素序列的转换程序，也就是说，文语转换系统不仅要应用数字信号处理技术，而且必须有大量的语言学知

识的支持。

本系统在使用写微博服务后，通过用户客户端接口即可实现文本转换为语音的需求。

3.6.2 语音识别

自动语音识别技术(Auto Speech Recognize，简称 ASR)所要解决的问题是让计算机能够“听懂”人类的语音，将语音中包含的文字信息“提取”出来。ASR 技术在“能听会说”的智能计算机系统中扮演着重要角色，相当于给计算机系统安装上“耳朵”，使其具备“能听”的功能，进而实现信息时代利用“语音”这一最自然、最便捷的手段进行人机通信和交互。

本系统在使用读微博服务后，通过用户客户端接口即可实现语音转换为文本的需求。

3.7 摇一摇相关技术介绍

3.7.1 Android 的传感器

传感器是一种物理装置或生物器官，能够探测、感受外界的信号、物理条件（如光、热、湿度）或化学组成（如烟雾），并将探知的信息传递给其他装置或器官。国家标准对传感器的定义是：“能感受规定的被测量并按照一定的规律转换成可用信号的器件或装置，通常由敏感元件和转换元件组成”。传感器是一种检测装置，能感受被测量的信息，并能将检测的得到的信息，按一定规律变换成为电信号或其他所需形式的信息输出，以满足信息的传输、处理、存储、显示、记录和控制等要求。它是实现自动检测和自动控制的首要环节。

Android 关于传感器 Sensor 有几种类型：

- 方向传感器： `Sensor.TYPE_ORIENTATION`
- 加速度(重力)传感器： `Sensor.TYPE_ACCELEROMETER`
- 光线传感器： `Sensor.TYPE_LIGHT`
- 磁场传感器： `Sensor.TYPE_MAGNETIC_FIELD`
- 距离(临近性)传感器： `Sensor.TYPE_PROXIMITY`
- 温度传感器： `Sensor.TYPE_TEMPERATURE`

在摇动手机的功能中，摇一摇中要用到加速度传感器。

3.7.2 加速度传感器介绍

加速度传感器返回值的单位是加速度，有三个方向的值分别是

`values[0]`: x-axis 方向加速度

`values[1]`: y-axis 方向加速度

`values[2]`: z-axis 方向加速度

其中 x,y,z 方向的定义是以水平放置的手机右下脚为参照系坐标原点

x 方向就是手机的水平方向，右为正

y 方向就是手机的水平垂直方向，前为正

y 方向就是手机的空间垂直方向，天空的方向为正，地球的方向为负

所以说，手机放置的空间位置不同，它三个方向的加速度也不同。在这里，因为一般正常情况下，任意轴数值最大就在 9.8~10 之间，只有在突然摇动手机的时候，瞬时加速度才会突然增大或减少。因此，摇一摇只要监听加速度的改变即可判断摇晃程度。

4.总体设计

4.1 总体界面设计

总体界面设计主要包括下面 2 个方面：登录授权界面、用户功能界面。

4.1.1 登录授权界面设计

登录授权界面主要分为 4 个部分：

第 1 部分：应用程序启动界面。

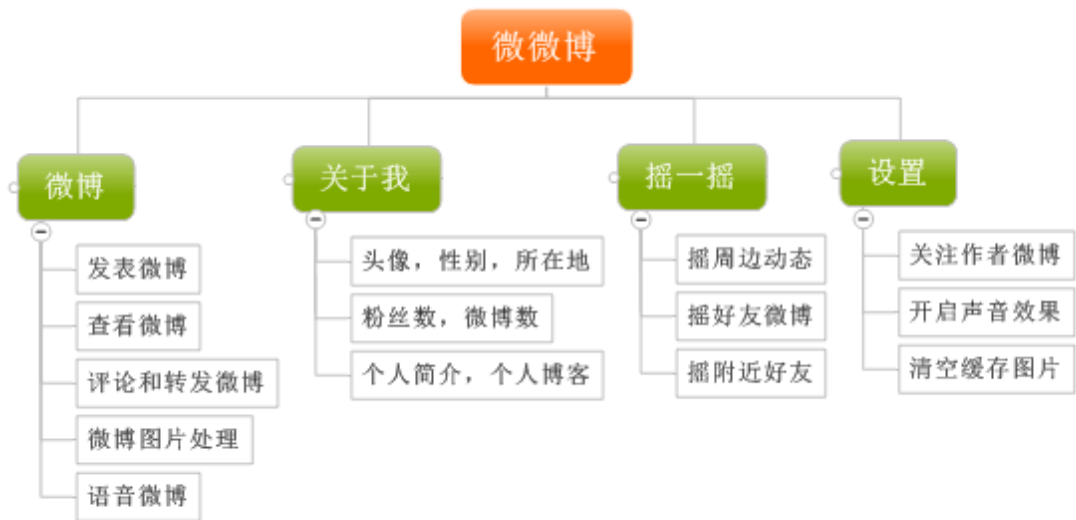
第 2 部分：系统登录注册界面。

第 3 部分：新浪微博登录授权界面。

第 4 部分：欢迎界面。

4.1.2 用户功能界面设计

本系统以微博为核心，功能界面主要分为四个部分：



(图 用户功能界面设计)

第一部分微博：

- 查看微博：包括查看好友微博，我的微博，公共微博以及提到我的
- 评论和转发微博：包括评论微博和转发微博，听微博
- 发表微博：包括发表文字微博和图片微博，说微博
- 图片处理：对于图片可以进行图片处理
- 语音微博：读一条微博内容或写一条微博

第二部分关于我：

包括头像，昵称，性别，所在地，个人简介，个人博客以及粉丝数和微博数等等

第三部分摇一摇：

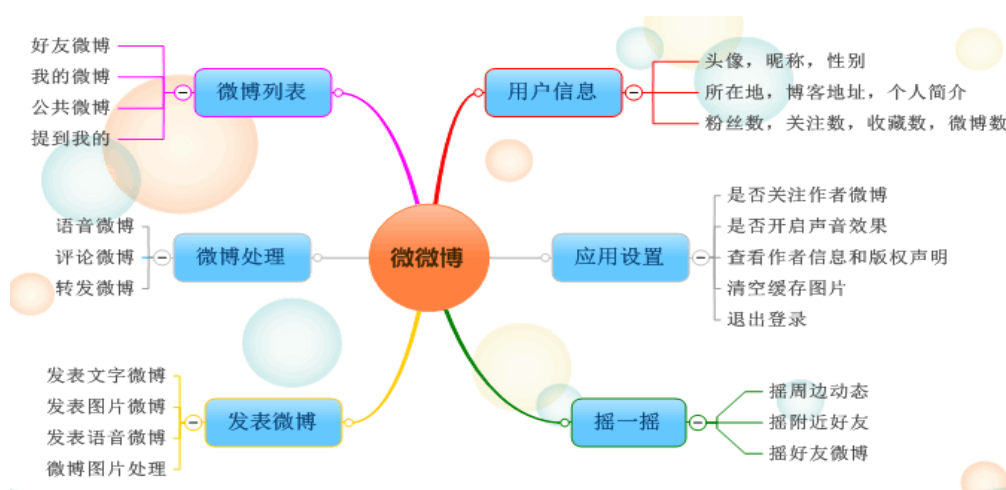
包括摇周边动态，摇附近好友和摇好友微博

第四部分设置：

包括是否关注作者微博，清空缓存图片是否开启声音效果以及查看作者信息和版权声明及退出登录

4.2 总体功能设计

总体功能图如下所示：



(图 系统功能设计)

功能详细：

第 1 部分：微博列表显示。

- 用户好友微博列表
- 用户微博列表
- 公共微博列表
- @提到我的微博列表
- 用户的粉丝最新微博列表
- 用户自己的微博列表
- 每条微博可以进行点击查看详细内容
- 可以对微博进行评论
- 可以对微博进行转发
- 微博列表的下拉刷新

第 2 部分：发表、评论及转发微博

这部分主要实现：接受用户输入的字符，并且传递出去。该部分包括转发、发评论及发微博的文本功能。当用户的输入为空的时候，会提示用户“输入不能为空”。当用户发表文字微博的时候，会动态显示用户还能输入多少字符（一次微博最多可以发 280 个字符，即 140 个汉字）。

第 3 部分：图片处理

这部分是对选中的图片进行处理，处理效果为强化、效果、贴纸、方向、剪切、亮度、对比度、饱和度、锐度、绘图、漂白、消除红眼等。

第 4 部分：用户个人信息显示

这部分主要显示的内容包括：用户图像、昵称、描述、地理位置、粉丝数、好友数、收藏的微博数、发布的微博数。

第 5 部分：摇微博

这部分主要实现：摇动手机，会随机获取一条微博。如果用户选择的类型是摇周边动态，则摇动手机结束后根据用户地理信息获取一条周边的微博，可查看微博详细内容也可以对微博进行转发和评论；如果用户选择的类型是摇附近好友，则摇动手机结束后根据用户的地理信息获取附近微博用户，并显示其最新发表的微博，可查看微博详细内容也可以对微博进行转发和评论；如果用户选择的是好友微博，则会根据用户选择的微博类型摇出好友的图片微博、音乐微博及视频微博。

第 6 部分：应用设置

这部分主要实现：用户的基本设置，如声音设置等。以及查看与该应用相关的基本信息。也可以选择关注作者微博等操作。同时对该应用可以进行意见反馈。进行清空缓存图片操作。使用退出操作。

5.功能详细设置

5.1 用户登录授权

5.1.1 功能说明

用户启动该程序，进入登陆注册界面，在登陆界面完成新浪微博的登陆授权，然后进入欢迎界面




5.1.2 功能结构

如果没有新浪微博账号点击注册，进入注册界面，如果有，点击登录进入新浪微博登陆授权界面，授权成功即可进入欢迎界面

5.1.3 核心代码

```
UserData userdata = UserDataUtil.readUserData(getApplicationContext()); // 得到用户数据
if (NetworkUtil.getNetworkState(Appstart.this) != NetworkUtil.NONE) { // 有网络
    String localToken = userdata.getToken();
    String localExpiresIn = userdata.getExpirestime();
    Intent intent = null;
    if (UserDataUtil.isTokenValid(localToken, localExpiresIn)) { // token还有效
        if (userdata.isFirstRun()) { // 第一次运行，进入欢迎界面
            intent = new Intent(Appstart.this, Whatsnew.class);
        } else { // 不是第一次运行，进入主界面
            intent = new Intent(Appstart.this, MainWeibo.class);
        }
    } else { // 无效，重新授权登陆
        intent = new Intent(Appstart.this, Welcome.class);
    }
    startActivity(intent);
    Appstart.this.finish();
} else { // 没有网络
    ToastUtil.showShortToast(Appstart.this, "网络不可用哟");
}
```

5.1.4 用户界面

<p>5.1.4.1 启动界面</p> 	<p>5.1.4.2 系统登录注册界面</p> 
<p>5.1.4.3 新浪微博授权界面</p> 	<p>5.1.4.4 欢迎界面</p> 

5.2 微博功能

5.2.1 功能说明

用户通过点击选项卡中的“微博”或者左右滑动，进入查看、发表、评论、转发微博功能的界面

5.2.2 功能结构

(1)在微博界面中左上角按钮，用户可以选择发表微博按钮进入写微博界面。

(2)在微博文本输入框写微博。当用户的输入为空的时候，会提示用户“输入不能为空”。当用户发表文字微博的时候，会动态显示用户还能输入多少字符（一次微博最多可以发 280 个字符，即 140 个汉字）。

(3)用户可以上传图片。图片来源可选本地相册和手机拍照，上传的图片可以强化、效果、贴纸、方向、剪切、亮度、对比度、饱和度、锐度、绘图、漂白、消除红眼等操作，也可以取消上传图片

(4)用户编辑好微博内容，点击发送，实现微博发送功能

(5)在微博界面中用户可以选择右上角的下拉框，根据“好友微博”“公共微博”“我的微博”“提到我的”进行微博列表加载显示

(6)对于加载的微博，用户可以下拉微博列表进行刷新，释放则开始刷新

(7)对于每条显示的微博，可点击评论或者转发按钮对微博进行评论或者转发

(8)对于显示的微博，用户可单击显示微博详情，点击“转发”和“评论”也对微博进行转发、评论，点击左上角返回按钮则返回上一界面。

(9)写微博时可以选择发送语音微博，点击语音输入图标，说一段不超过 140 个字的内容，稍等片刻，系统自动转成文本内容出现在微博文本输入框中。

5.2.3 核心代码

5.2.3.1 发表微博

```

if (TextUtils.isEmpty(mContent)) {
    ToastUtil.showShortToast(getApplicationContext(), "朋友，请输入内容");
    return;
}
ToastUtil.showShortToast(getApplicationContext(), "微博发送中,请稍等...");
new Thread(new Runnable() {
    public void run() {
        StatusesAPI api= new StatusesAPI(userData.obtainOAuth2AccessToken());// hjw
        if (!TextUtils.isEmpty(mPicPath)) {
            api.upload(mContent, mPicPath, null, null, HomeWeiboWrite.this);
        } else {
            api.update(mContent, null, null, HomeWeiboWrite.this);
        }
    }
}).start();

```

5.2.3.2 加载微博

```

switch (type) {
case TIMELINE\_PUBLIC:
    paging.setSinceld(sinceldPublic);
    statusWapper = timeline.getPublicTimeline(count, baseAPP);
    sinceldPublic = Long.parseLong(statusWapper.getStatuses().get(o).getId());
    insertTimelineBefore(listPublicTimeline, statusWapper.getStatuses());
    break;
case TIMELINE\_FRIENDS:
    paging.setSinceld(sinceldFriends);
    statusWapper = timeline.getFriendsTimeline(baseAPP, feature, paging);
    if (statusWapper.getStatuses().size() > 0) {
        sinceldFriends = Long.parseLong(statusWapper.getStatuses().get(o).getId());
    }
    insertTimelineBefore(listFriendsTimeline, statusWapper.getStatuses());
    break;
case TIMELINE\_USER:
    paging.setSinceld(sinceldUser);
    statusWapper = timeline.getUserTimelineByUid(userData.getUserid(), paging, baseAPP,
feature);// userid
    if (statusWapper.getStatuses().size() > 0) {

```

```

        sinceIdUser = Long.parseLong(statusWapper.getStatuses().get(o).getId());
    }
    insertTimelineBefore(listUserTimeline, statusWapper.getStatuses());
    break;
case TIMELINE_MENTIONS:
    paging.setSinceld(sinceldMentions);
    statusWapper = timeline.getMentions(paging, o, o, o); // mentions three filters
    if (statusWapper.getStatuses().size() > o) {
        sinceIdMentions = Long.parseLong(statusWapper.getStatuses().get(o).getId());
    }
    insertTimelineBefore(listMentionsTimeline, statusWapper.getStatuses());
    break;
default:
    paging.setSinceld(sinceldPublic);
    statusWapper = timeline.getPublicTimeline(count, baseAPP);
    if (statusWapper.getStatuses().size() > o) {
        sinceIdPublic = Long.parseLong(statusWapper.getStatuses().get(o).getId());
    }
    insertTimelineBefore(listPublicTimeline, statusWapper.getStatuses());
    break;
}

private void showTimeline(int type) {
    if (userData.isSoundPlay()) {
        musicPlayer.start();
    }
    currentType = type;
    hideAllListView();
    rlHomeLoading.setVisibility(View.GONE); //
    switch (type) {
        case TIMELINE_PUBLIC:
            lvPublicTimeline.setVisibility(View.VISIBLE);
            break;
        case TIMELINE_FRIENDS:
            lvFriendsTimeline.setVisibility(View.VISIBLE);
            break;
        case TIMELINE_USER:
            lvUserTimeline.setVisibility(View.VISIBLE);
            break;
        case TIMELINE_MENTIONS:
            lvMentionTimeline.setVisibility(View.VISIBLE);
            break;
    }
}

```


5.2.3.3 评论微博

```
new AsyncTask<Void, Void, Void>() { // AsyncTask: call from UI thread
    protected Void doInBackground(Void... params) {
        Comments cm = new Comments();
        cm.client.setToken(userData.getToken());
        try {
            // cm.createComment(commentContent, statusid);
            Integer comment_ori = isCommentOri ? 1 : 0;
            cm.createComment(commentContent, statusid, comment_ori);
        } catch (WeiboException e) {
            e.printStackTrace();
        }
        return null;
    }
    @Override
    protected void onPostExecute(Void result) {
        ToastUtil.showShortToast(getApplicationContext(), "评论成功");
    }
}.execute();
```

5.2.3.4 转发微博

```
new AsyncTask<Void, Void, Void>() { // AsyncTask: call from UI thread
    protected Void doInBackground(Void... params) {
        Timeline timeline = new Timeline();
        timeline.client.setToken(userData.getToken());
        try {
            Integer comment = isComment ? 1 : 0;
            timeline.Repost(statusid, commentContent, comment);
        } catch (WeiboException e) {
            e.printStackTrace();
        }
        return null;
    }
    @Override
    protected void onPostExecute(Void result) {
        ToastUtil.showShortToast(getApplicationContext(), "转发成功");
    }
}.execute();
```

5.2.3.5 图片处理

```
/**
 * 拍照得到图片
 */
public void takeCameraPicture() {
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    startActivityForResult(intent, ConstantUtil.REQUEST_TAKE_PICTURE);
}

/**
 * 从本地相册选择图片
 */
public void pickMobilePicture() {
    Intent intent = new Intent(Intent.ACTION_GET_CONTENT); // uri --> content://
    /external/images/media/181
    intent.setType("image/*");
    startActivityForResult(intent, ConstantUtil.REQUEST_PICK_PICTURE);
}
```

5.2.3.5 语音微博

```
public void showSynDialog() {
    String source = weiboItem.getContent();
    // 设置合成文本.
    ttsDialog.setText(source, null);

    // 设置发音人.
    String role = mSharedPreferences.getString(getString(R.string.preference_key_tts_role),
        getString(R.string.preference_default_tts_role));
    ttsDialog.setVoiceName(role);

    // 设置语速.
    int speed = mSharedPreferences.getInt(getString(R.string.preference_key_tts_speed),
50);
    ttsDialog.setSpeed(speed);

    // 设置音量.
    int volume = mSharedPreferences.getInt(getString(R.string.preference_key_tts_volume),
50);
```

```

        ttsDialog.setVolume(volume);

        // 设置背景音.
        String music =
mSharedPreferences.getString(getString(R.string.preference_key_tts_music),
        getString(R.string.preference_default_tts_music));
        ttsDialog.setBackgroundSound(music);

        // 弹出合成Dialog
        ttsDialog.show();
    }

    /*
     * (non-Javadoc)缓存成功
     *
     * @see com.iflytek.speech.SynthesizerPlayerListener#onBufferPercent(int, int, int)
     */
    @Override
    public void onBufferPercent(int percent, int beginPos, int endPos) {
        mPercentForBuffering = percent;
        mToast.setText(String.format(getString(R.string.tts_toast_format),
mPercentForBuffering, mPercentForPlaying));
        mToast.show();
    }

    /*
     * (non-Javadoc)播放
     *
     * @see com.iflytek.speech.SynthesizerPlayerListener#onPlayPercent(int, int, int)
     */
    @Override
    public void onPlayPercent(int percent, int beginPos, int endPos) {
        mPercentForPlaying = percent;
        mToast.setText(String.format(getString(R.string.tts_toast_format),
mPercentForBuffering, mPercentForPlaying));
        mToast.show();
    }
}

```

5.2.4 用户界面

<p>5.2.4.1 加载微博</p> 	<p>5.2.4.2 选择微博类型</p> 
<p>5.2.4.3 写微博</p> 	<p>5.2.4.4 发表评论或者转发</p> 

5.2.4.5 图片处理 1



5.2.4.6 图片处理 2



5.2.4.7 图片处理 3



5.2.4.8 图片处理 4



5.2.4.9 语音微博写微博 1



5.2.4.10 语音微博写微博 2



5.2.4.11 语音微博读微博 1



5.2.4.12 语音微博读微博 2



5.3 个人信息

5.3.1 功能说明

用户通过点击选项卡中的“关于我”或者左右滑动，进入查看个人详细信息的界面

5.3.2 功能结构

- (1)在该界面中用户可以查看个人的昵称、头像、性别
- (2)可以查看在线状态、博客地址、所在位置个人简介
- (3)可以显示该用户的粉丝数量；显示用户所关注的用户数量
- (4)显示用户发表过的所有微博数量；显示收藏的微博数量

5.3.3 核心代码

```
// set user info to controls
private void setUserInfo() {
    rlInfoLoading.setVisibility(View.GONE);
    sv_userinfo_info.setVisibility(View.VISIBLE);
    tv_info_screenname.setText(user.getScreenName());
    tv_info_location.setText(user.getLocation());
    if (user.getOnlineStatus() == 0) {
        tv_info_online_status.setText("不在线");
    } else if (user.getOnlineStatus() == 1) {
        tv_info_online_status.setText("在线");
    }
    if (user.getUrl() == null || "".equalsIgnoreCase(user.getUrl())) {
        tv_info_blogurl.setText("暂无博客");
    } else {
        if (user.getUrl().length() > BLOG_URL_MAX_LENGTH) {
            tv_info_blogurl.setText(user.getUrl().substring(0,
BLOG_URL_MAX_LENGTH) + "...");
        } else {
            tv_info_blogurl.setText(user.getUrl());
        }
    }
    if (user.getDescription() == null) {
        tv_info_description.setText("暂无简介");
    }
}
```

```

    } else {
        tv_info_description.setText(user.getDescription());
    }
    tv_info_followers_count.setText(user.getFollowersCount() + "");
    tv_info_friends_count.setText(user.getFriendsCount() + "");
    tv_info_statuses_count.setText(user.getStatusesCount() + "");
    tv_info_favourites_count.setText(user.getFavouritesCount()+ "");
    if (user.getGender().equalsIgnoreCase("f")) { // female
        iv_info_gender.setImageResource(R.drawable.userinfo_female);
    } else if (user.getGender().equalsIgnoreCase("m")) {
        iv_info_gender.setImageResource(R.drawable.userinfo_male);
    } else { // not know
        iv_info_gender.setVisibility(View.GONE);
    }
    WeiboUtil.restoreBitmap(CacheUtil.PROFILE_CACHE_PATH,
        user.getProfileImageUrl(), tabinfohandler, iv_info_photo,
        IMAGE_TYPE_PROFILE);

```

5.3.4 用户界面

5.3.4.1 个人信息界面



5.4 摇一摇

5.4.1 功能说明

用户通过点击选项卡中的“摇一摇”或者左右滑动，进入摇一摇功能的界面

5.4.2 功能结构

（1）如果用户选择的类型是摇周边动态，则摇动手机结束后根据用户地理信息获取一条周边的微博，可查看微博详细内容也可以对微博进行转发和评论

（2）如果用户选择的类型是摇附近好友，则摇动手机结束后根据用户的地理信息获取附近微博用户，并显示其最新发表的微博，可查看微博详细内容也可以对微博进行转发和评论

（3）如果用户选择的是好友微博，则会根据用户选择的微博类型摇出图片微博、音乐微博及视频微博

5.4.3 核心代码

```
// nearby weibo
private void shakeNearbyWeibo() throws WeiboException {
    Location location = locationManager.getLastKnownLocation(network_provider); // network
    if (location == null) {
        location = locationManager.getLastKnownLocation(gps_provider); // gps
    }
    shake_result = MATCH;
    Place place = new Place();
    place.client.setToken(userData.getToken());
    StatusWapper statusWapper;
    if (location != null) {
        statusWapper = place.nearbyTimeline(location.getLatitude(),
location.getLongitude()); //
    } else {
        statusWapper = place.nearbyTimeline(28.15675335, 112.934191); // my location
    }
    Random random = new Random();
    int randomint = random.nextInt(statusWapper.getStatuses().size());
    status = statusWapper.getStatuses().get(randomint);
    if (status == null) {
        shake_result = NO_MATCH;
    }
}

// post shake nearby weibo
private void postShakeNearbyWeibo() {
    tv_shakeweiboitem_name.setText(status.getUser().getScreenName());
    tv_shakeweiboitem_distance.setText(status.getDistance() + "M");
    String content = status.getText();
    if (content.length() > MAX_CONTENT_LENGTH) {
        content = content.substring(0, MAX_CONTENT_LENGTH) + "...";
    }
    tv_shakeweiboitem_content.setText(content);
    User user = status.getUser();
    if (user.getGender().equalsIgnoreCase("f")) { // female
        iv_shakeweiboitem_gender.setVisibility(View.VISIBLE);
        iv_shakeweiboitem_gender.setImageResource(R.drawable.user_info_female);
    } else if (user.getGender().equalsIgnoreCase("m")) { // male
        iv_shakeweiboitem_gender.setVisibility(View.VISIBLE);
    }
}
```

```

        iv_shakeweiboitem_gender.setImageResource(R.drawable.user_info_male);
    } else {// not know
        iv_shakeweiboitem_gender.setVisibility(View.GONE);
    }
    WeiboUtil.restoreBitmap(CacheUtil.PROFILE_CACHE_PATH,
user.getProfileImageUrl(), shakeHandler,
        iv_shakeweiboitem_head, ConstantUtil.IMAGE_TYPE_PROFILE);
    TranslateAnimation downAnimation = new
TranslateAnimation(Animation.RELATIVE_TO_SELF, of,
        Animation.RELATIVE_TO_SELF, of, Animation.RELATIVE_TO_SELF, -1.2f,
Animation.RELATIVE_TO_SELF, of);
    downAnimation.setDuration(1000);
    downAnimation.setFillAfter(true);
    rl_shake_result.setVisibility(View.VISIBLE);
    rl_shake_result.startAnimation(downAnimation);

    rl_shake_result.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
            Intent intent = new Intent(ShakeWeibo.this, ResultWeiboItem.class);
            intent.putExtra("status", new WeiboItem(status));
            startActivity(intent);
        }
    });
}

```

5.4.4 用户界面

<p>5.4.4.1 摇一摇类型选择</p> 	<p>5.4.4.2 摇一摇</p> 
<p>5.4.4.3 摇出一条微博</p> 	<p>5.4.4.4 查看摇出的微博详情</p> 

5.5 系统设置

5.5.1 功能说明

用户通过点击选项卡中的“设置”或者左右滑动，进入查看设置功能的界面

5.5.2 功能结构

- (1)在该界面中用户可以关注或取消关注作者微博
- (2)可以开启或关闭声音效果
- (3)了解版本更新状况
- (4)进行意见反馈。将@作者发一条微博。
- (5)了解开发团队情况。了解作者团队的信息将进入另一个界面，内容包括版本、团队和作者及版权。
- (6)清空缓存图片
- (7)退出登录。

5.5.3 核心代码

5.5.3.1 关注作者微博

```
//follow
private void followAuthor() {
    ToastUtil.showShortToast(getApplicationContext(), "正在关注作者微博，稍等哈...");
    new AsyncTask<Void, Void, Void>() { // AsyncTask: call from UI thread
        protected Void doInBackground(Void... params) {
            Friendships fm = new Friendships();
            fm.client.setToken(userData.getToken());
            try {
                fm.createFriendshipsByld(ConstantUtil.AUTHOR_UID);
            } catch (WeiboException e) {
                e.printStackTrace();
            }
            return null;
        }
    }

    protected void onPostExecute(Void result) {
```

```

        isFollowd = true;
        tv_settings_followauthor.setText("已经关注了");
        ToastUtil.showShortToast(getApplicationContext(), "成功关注作者微博！");
        userData.setFollowauthor(isFollowd);
        UserDataUtil.updateLocalToken(getApplicationContext(), userData);
    }
    }.execute();
}

```

5.5.3.2 开启声音效果

```

cb_settings_sound.setOnCheckedChangeListener(new OnCheckedChangeListener() {
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
        userData.setSoundPlay(isChecked);
        UserDataUtil.updateLocalToken(getApplicationContext(), userData);
    }
});

public static void updateLocalToken(Context context, UserData userData) {
    SharedPreferences.Editor editor =
context.getSharedPreferences(ConstantUtil.TINYWEIBO, Context.MODE_PRIVATE)
    .edit();
    editor.putString(USERID, userData.getUserid());
    editor.putString(TOKEN, userData.getToken());
    editor.putString(EXPIRETIME, userData.getExpirestime());
    editor.putString(NICKNAME, userData.getNickname());
    editor.putString(PROFILE, userData.getProfileimage());
    editor.putBoolean(FOLLOW, userData.isFollowauthor());
    editor.putBoolean(SOUND, userData.isSoundPlay());
    editor.putBoolean(FIRSTRUN, userData.isFirstrun());
    editor.commit();
}

```

5.5.3.3 清空缓存图片

```

// clear cache
public void rl_settings_clearcache(View v) {
    ToastUtil.showShortToast(getApplicationContext(), "正在清空缓存图片，稍等哈...");
}

```

```

new AsyncTask<Void, Void, Void>() {// AsyncTask: call from UI thread
    protected Void doInBackground(Void... params) {
        CacheUtil.clearCache();
        return null;
    }

    protected void onPostExecute(Void result) {
        ToastUtil.showShortToast(getApplicationContext(), "缓存图片已清空！");
    }
}.execute();
}

// clear cache
public static void clearCache() {
    deleteDirectory(IMAGE_CACHE_PATH);
    deleteDirectory(PICTURE_CACHE_PATH);
    deleteDirectory(PROFILE_CACHE_PATH);
}

// delete files in the specified directory
private static void deleteDirectory(String cachePath) {
    File file = new File(cachePath);
    if (file.exists()) {
        File[] files = file.listFiles();
        for (int i = 0; i < files.length; i++) {
            files[i].delete();
        }
    }
}
}

```

5.5.4 用户界面

<div>5.5.4.1 用户设置主界面</div> <div></div>	<div>5.5.4.2 关于我们</div> <div></div>
<div>5.5.4.3 意见反馈</div> <div></div>	<div>5.5.4.4 退出登录</div> <div></div>

5.6 退出应用

5.6.1 功能说明

用户按 Menu 键，将弹出菜单，只有退出一个选项，或者按下 Back 键，直接显示退出界面

5.6.2 功能结构

如果退出本系统，同时将不能接收新的信息。

5.6.3 核心代码

```
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (keyCode == KeyEvent.KEYCODE_BACK && event.getRepeatCount() == 0) {
        if (menu_display) {
            menuWindow.dismiss();
            menu_display = false;
        } else {
            Intent intent = new Intent();
            intent.setClass(MainWeibo.this, DialogExit.class);
            startActivity(intent);
        }
    } else if (keyCode == KeyEvent.KEYCODE_MENU) {
        if (!menu_display) {
            mainMenu = inflater.inflate(R.layout.main_menu, null);
            menuWindow = new PopupWindow(mainMenu,
                android.view.ViewGroup.LayoutParams.FILL_PARENT,
                android.view.ViewGroup.LayoutParams.WRAP_CONTENT);
            menuWindow.showAtLocation(findViewById(R.id.rl_mainweibo),
                Gravity.BOTTOM | Gravity.CENTER_HORIZONTAL,
                0, 0);
            btn_menu_close = (LinearLayout)
                mainMenu.findViewById(R.id.btn_menu_close);

            btn_menu_close.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View arg0) {
                    Intent intent = new Intent();
                }
            });
        }
    }
}
```

```

        intent.setClass(MainWeibo.this, DialogExit.class);
        startActivity(intent);
        menuWindow.dismiss();
    }
    });
    menu_display = true;
} else {
    menuWindow.dismiss();
    menu_display = false;
}
return false;
}
return false;
}
}

```

5.6.4 用户界面

5.6.4.1 选择退出按钮



5.6.4.2 确定退出

