**目录**

1. 项目介绍
2. UI
3. Controller
4. Server

一．项目简介：

Chat 安卓app是由我们小组五位成员共同开发的聊天软件，主打特点是摇一摇震动对方手机提醒对方接受消息以及消息已读功能。前端采用了安卓studio实现。后台基于mysql的tornado系统。前端后台采用了http通信，服务器向前端发送消息采用了“心跳”策略。经过ab等测试工具测试，在1G一核的商业云服务器下，能支持每秒400多人的并发访问。

二．UI设计：

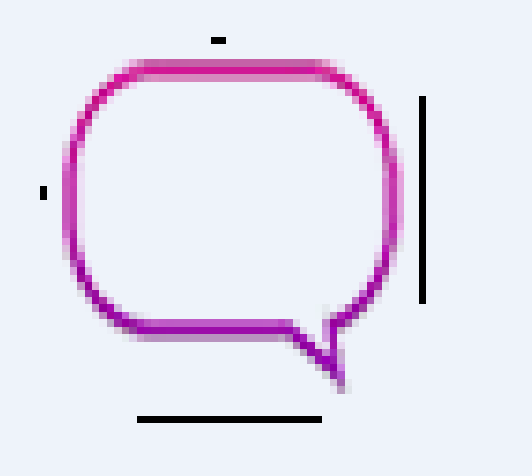
1.整体ui框架fragmentLayout

fragmentLayout是一个很好用的界面管理框架，它的样式是使应用的界面可以进行复用，即在一个activity下支持多个界面的显示，各界面之间的显示切换可以通过下方的fragmentTabHost进行控制，在一个activity复用多个界面，在显示刷新速度上很快，大大的优化了操作的体验。

2.9.png的使用

.9图片即图片后缀名前有.9的图片，如pic.9.png、pic1.9.jgp，诸如此类的图片就称为.9图片。.9图片的作用是在图片拉伸的时候特定的区域不会发生图片失真，至于哪些部分不会失真那么得看你的图片是如何绘制的。

在本项目中，我们使用该类型的图片对聊天的内容进行包装，做出像交互式聊天的界面，外观如下。



在该图中可以见到在上下左右各有一条明显的黑线，其中上边和左边的黑线较短，这是用来在包裹内容很大的时候用作拉伸的区域，即若横向文字很多需要横向拉伸时，会在上方黑线所表示的那一段区域内进行复制拉伸该边框，纵向也是同理。而右边和下边的黑线则是表示了文字显示的部分，只有在黑线对应的范围内才能显示文字，随着边框拉伸，这两条也会随之拉伸。

黑线的附加可以使用android studio的自带工具，但同样，由于只是一像素宽度的黑线，在手头没有工具时使用photoshop等工具也可以制作。

3.传感器的使用

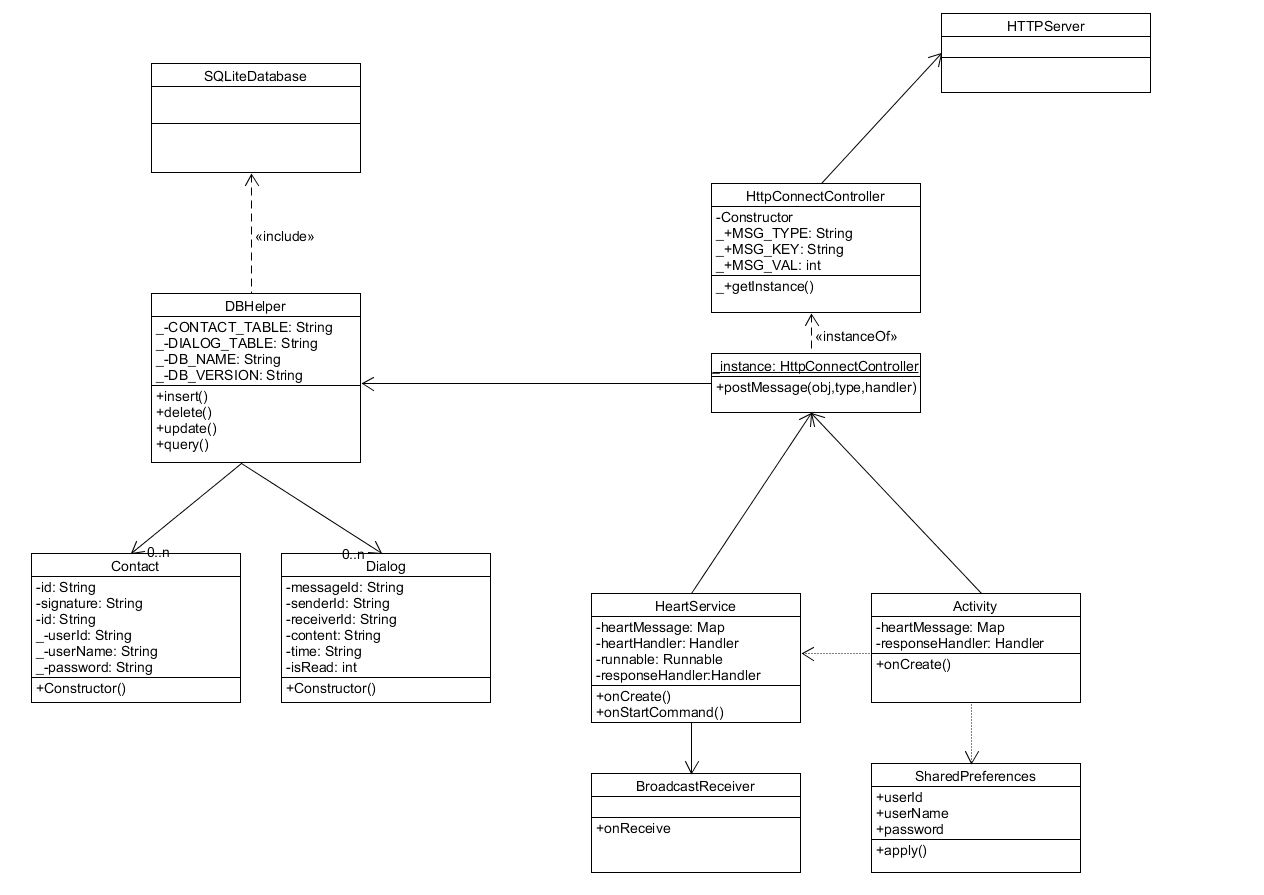
该应用中使用了简单的传感器来进行摇一摇操作的实现，主要使用的就只有加速度传感器了，加速度传感器会向其存储数组中给出xyz三个方向上加速度的值，为了仿真摇一摇的动作，我们使用了加速度的变化量，对应手在摇的时候的方向瞬间变化的动作。

4.广播和Notification

该应用中，我们使用了Notification在应用在后台时给出聊天消息的通知，使用到了广播和Notification，广播负责对接收到的新消息进行广播使得在各个界面甚至后台时能够捕捉到新消息到来，从而使广播接收器能够收到消息并进行相应的动作，这里的动作便是构造一个新的Notification消息，使得用户可以及时的看到在手机上方弹出的消息进行回复。

三．Controller:

设计架构：



四．Server:

1. 组织框架：

后台基于tornado

数据库基于mysql

链接数据库采用了torndb

后台架构：

---src

---api.md api文档

---app.py 程序主入口

---settings.py 设置文件，包含数据库链接需要的用户名密码等

---my.sql 数据库初始化文件

---urls.py 路由设置

handler文件夹

---base.py等 具体的路由处理文件

Logconfig文件夹

---logconfig.py 日志设置

myssqlDb文件夹

---mysqlDb\_helper.py 所有与数据库操作有关的命令，增删改查

1. 接口介绍：

详情参考api.md接口文档

方法全部为post，格式全为json

关键技术说明：

1. 心跳服务：

移动端保持与服务器通信的解决方案

我们参考了各大网站保持通信的方式，有建立长tcp连接的，由服务器push信息给客户端。不过更多的论坛、网站运用的还是这种“心跳策略”。我们曾尝试过保持tcp链接不close，发现最多只能支持百人左右的链接就无法建立新的链接了。因而最终我们决定采用“心跳策略”。在“心跳”中提醒客户机应该去哪个接口提取新信息，当没有新消息的时候返回403。我们最终心跳的频率定在了每秒一次，其实更多时候是403的返回值的，对客户机和服务器的压力都不是很大。

1. 一键部署

后台的技术栈是ubuntu + mysql-client + python2.7 + tornado

采用了docker实现了一键部署，利用了网络上的镜像以及对他人编写好的dockercompose进行修改。mysql-server部分我们打算专门处理，这在各大公司也是很常见的。本次我们采用了一齐部署在同一台云服务器下，实现了公网访问。