客户名字和logo

餐厅订餐系统-客户端详细设计文档

wang linglong

日期：2014-12-28

文档版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 修订日期 | 修订人 | 审核人 | 变更内容 |
| V0.1 | 2014/11/30 | Wang LingLong |  | 初始化 |
| V0.2 | 2014/12/8 | Wang LingLong |  | 补充详细内容 |
| V0.3 | 2014/12/8 | Wang LingLong |  | 更新类图和时序图 |
| V0.4 | 2014/12/27 | Wang LingLong |  | 更新反馈的问题 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1 文档介绍 1](#_Toc407461870)

[1.1 文档目的 1](#_Toc407461871)

[1.2 文档范围 1](#_Toc407461872)

[1.3 缩写词列表 1](#_Toc407461873)

[1.4 参考内容 1](#_Toc407461874)

[2 模块设计 2](#_Toc407461875)

[2.1 菜品浏览 2](#_Toc407461878)

[2.1.1 设计图 2](#_Toc407461879)

[2.1.2 功能描述 2](#_Toc407461880)

[2.1.3 菜品浏览类图 3](#_Toc407461881)

[2.1.4 菜品浏览时序图 3](#_Toc407461882)

[2.1.5 函数说明 4](#_Toc407461883)

[2.2 点菜 6](#_Toc407461884)

[2.2.1 设计图 6](#_Toc407461886)

[2.2.2 功能描述 6](#_Toc407461887)

[2.2.3 类图 7](#_Toc407461888)

[2.2.4 时序图 7](#_Toc407461889)

[2.2.5 函数说明 7](#_Toc407461890)

[2.3 下单 9](#_Toc407461891)

[2.3.1 设计图 9](#_Toc407461893)

[2.3.2 功能描述 9](#_Toc407461894)

[2.3.3 类图 10](#_Toc407461895)

[2.3.4 时序图 10](#_Toc407461896)

[2.3.5 函数说明 11](#_Toc407461897)

[2.4 历史订单查询 12](#_Toc407461898)

[2.4.1 设计图 12](#_Toc407461900)

[2.4.2 功能描述 12](#_Toc407461901)

[2.4.3 类图 13](#_Toc407461902)

[2.4.4 时序图 13](#_Toc407461903)

[2.4.5 函数说明 13](#_Toc407461904)

[3 存储数据设计 15](#_Toc407461905)

[3.1 应用设置 15](#_Toc407461907)

[3.1.1 Android常用存储方式说明 15](#_Toc407461908)

[3.1.2 本例实现方式说明 15](#_Toc407461909)

[4 数据缓存设计 17](#_Toc407461910)

[4.1 功能描述 17](#_Toc407461912)

[4.2 实现方式 18](#_Toc407461913)

[4.3 函数说明 19](#_Toc407461914)

# 文档介绍

## 文档目的

本文描述eOrder餐厅订餐系统架构设计，用于指导开发人员进行系统详细设计；需求人员进行系统需求实现评估；测试人员进行系统测试用例编制。

## 文档范围

本文档主要指导客户端开发人员进行编码工作，并供测试人员等项目其他人员参考。

## 缩写词列表

|  |  |
| --- | --- |
| 缩写词 | 解释 |
| MVC | MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## 参考内容

# 模块设计



## 菜品浏览

### 设计图



### 功能描述

浏览菜单

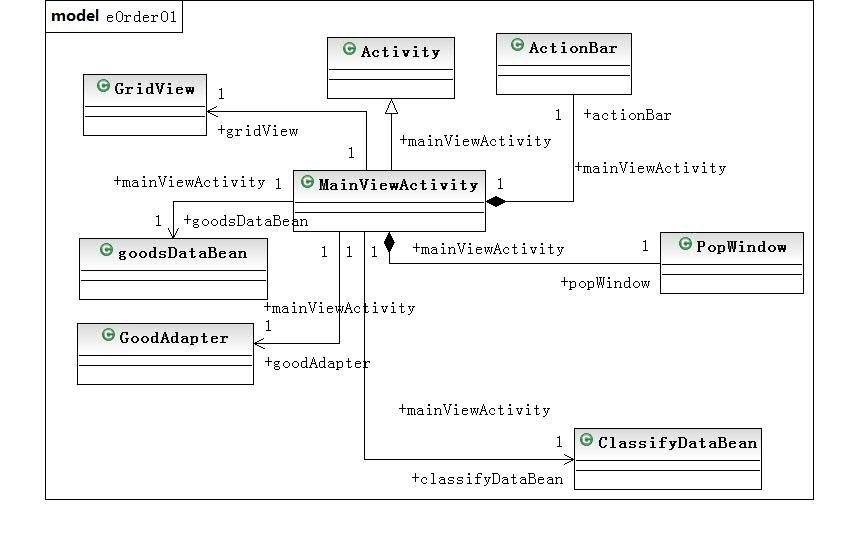
1. 用户打开手机上的电子菜单应用

2. 应用按菜单分类显示第一大类第一分页的菜品信息

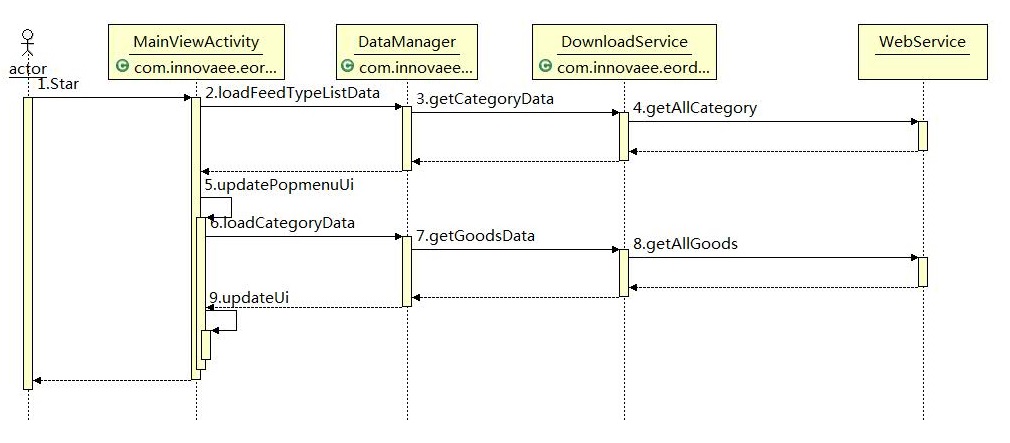
3. 用户点击菜单屏幕的上下滑动翻动菜单页

4. 用户点击菜单左上角的菜单分类筛选器直接跳转到该菜品分类的第一页

### 菜品浏览类图



### 菜品浏览时序图



### 函数说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.mobile.view | | |
| 类名/接口名 | MainViewActivity | | |
| 父类 | Activity | | |
| 实现接口 | N/A | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| feedTypeList | List<FeedType> | 所有分类类型列表，记录分类种类状态 | |
| gridView | GridView | 九宫格显示控件 | |
| goodsAdapter | GoodsAdapter | GridView的数据绑定器 | |
| goodsListData | List<GoodsDataBean> | 当前分类下所有菜品的数据列表 | |
| categoryListData | List<CategoryDataBean> | 服务器下发的分类类型原始数据 | |
| selectOrderGoods | List<GoodsDataBean> | 当前已经选中的菜品列表 | |
|  |  |  | |
| 视图部分 |  |  | |
| actionBar | ActionBar | 自定义ActionBar | |
| orderActionView | View | 订单数量显示view | |
| orderCountView | TextView | 订单数量显示view里数量TextView | |
| feedTypePopup | PopupWindow | 分类选择弹出菜单 | |
| feedTypeName | TextView | ActionBar上显示的当前查看分类 | |
| popupMenuList | ListView | 分类弹出菜单中listview，和PopupWindow一起使用 | |
| displayMetrics | DisplayMetrics | 显示工具类 | |
| orderHestoryInput | EditText | 订单记录查询会员号输入编辑器 | |
| progressBar; | ProgressBar | 读取数据进度条 | |
|  |  |  | |
| 消息部分 |  |  | |
| handler | Handler | 消息处理Handler | |
| MSG\_UPDATE | static int | 初始化完成消息 | |
| MSG\_INITDATA | static int | 数据初始化完成消息 | |
| MSG\_UPDATE\_POPMENU | static int | 更新PopMenu消息 | |
| MSG\_ORDER | static int | 下单的消息 | |
| MSG\_DELALL | static int | 删除所有已经选中的菜品的消息 | |
| MSG\_UPDATE\_COUNT | static int | 更新选中的菜品的数目的消息 | |
| MSG\_UPDATE\_FAILUI | static int | 显示失败的消息 | |
| receiver; | BroadcastReceiver | 广播接收器 | |
|  |  |  | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| onCreate | 无 | Bundle savedInstanceState | Activity创建函数，系统调用 |
| onCreateOptionsMenu | boolean是否已经处理 | Menu paramMenu | Menu创建函数，系统调用 |
| onOptionsItemSelected | boolean是否已经处理 | Menu paramMenu | Menu点击处理函数，系统调用 |
| initView | 无 | 无 | 初始化view，控件函数 |
| initData | 无 | 无 | 初始化数据，监听函数 |
| initActionBarCustomView | 无 | 无 | 初始化ActionBar自定义View |
| openSetting | 无 | 无 | 进入设置界面 |
| openAbout | 无 | 无 | 进入关于界面 |
| oepnOrderHestory | 无 | int userId | 进入订单记录界面 |
| openMyOrder | 无 | 无 | 进入我的订单界面 |
| changeFeedType | 无 | FeedType paramFeedType | 切换当前的FeedType，并更新显示 |
| initFeedTypePopup | 无 | 无 | 初始化Popmenu菜单 |
| clickOnFeedType | 无 | View paramView | 处理Actionbar中FeedType自定义View点击事件 |
| clickOnOrderHestory | 无 | View paramView | 处理ActionBar中orderHestory自定义View点击事件 |
| updateMyOrderCount | 无 | int count | 更新我的订单数目 |
| updateUi | 无 | 无 | 更新GridView UI |
| updatePopmenuUi | 无 | 无 | 更新Popmenu UI |
| loadCategoryData | 无 | Int id 菜品分类id | 加载某个分类菜品列表数据 |
| loadFeedTypeListData | 无 | 无 | 加载所有分类列表信息 |
| changeCategoryToFeedType | List<FeedType> | List<CategoryDataBean> categoryListData | 转换分类数据为切换分类所需的数据类型 |

## 点菜



### 设计图



### 功能描述

点菜下单

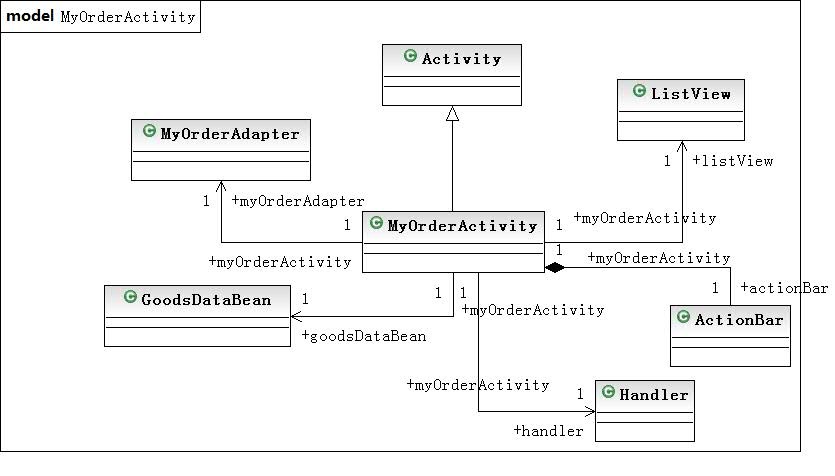
1. 用户单击菜品图片，菜品加入页面下方的购物车图标

2. 用户点击页面上方的购物车图标，进入订单详情页面

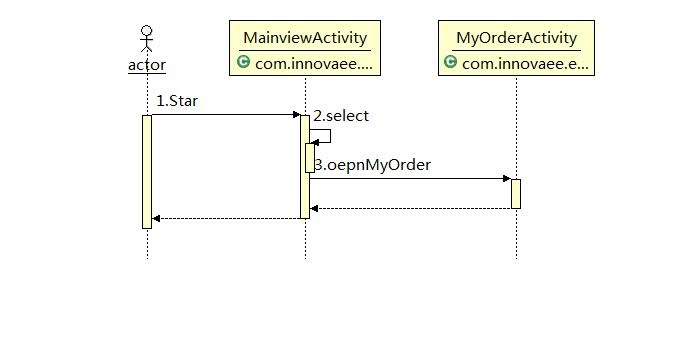
3. 用户上下滑动查看订单详情

4. 用户点击返回按钮回到菜单首页

### 类图



### 时序图



### 函数说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.mobile.view | | |
| 类名/接口名 | MyOrderActivity | | |
| 父类 | Activity | | |
| 实现接口 | N/A  N/A | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| selectOrderGoods | List<GoodsDataBean> | 已经选择商品列表 | |
| 视图部分 |  |  | |
| listView | ListView | 列表显示控件，显示订单项 | |
| myOrderAdapter | MyOrderAdapter | 订单数据绑定器 | |
| actionBar | ActionBar | ActionBar显示器 | |
| allPirce | TextView | 总价格显示框 | |
| delAllBtn | Button | 全部删除按钮 | |
| orderBtn | Button | 下单按钮 | |
| 消息部分 |  |  | |
| mHandler | Handler | 消息接收Handler | |
| MSG\_UPDATE | int | 内部消息定义，刷新消息 | |
| MSG\_INITDATA | int | 初始化数据完成消息 | |
| MSG\_UPDATE\_COUNT | int | 更新显示的下单数目消息 | |
| MSG\_RESET\_ADAPTER | int | 重新设置适配器消息 | |
| ACTION\_INTENT\_DELALL | String | 删除所有已选项广播 | |
| ACTION\_INTENT\_CHANGECOUNT | String | 改变菜品数量广播 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| onCreate | 无 | Bundle savedInstanceState | Activity创建函数，系统调用 |
| onOptionsItemSelected | boolean是否处理 | MenuItem item | ActionBar上的按键消息处理函数，系统调用 |
| initView | 无 | 无 | 初始化控件View |
| initData | 无 | 无 | 初始化控件数据，监听器 |
| getAllPrice | Double总价 | 无 | 得到当前订单的总价 |
| openOrderActivity | 无 | 无 | 跳转到下单界面 |
| sendBroadcastToMainActivity | 无 | 无 | 发送改变数目广播到主界面，用于主界面刷新数目 |
|  |  |  |  |
|  |  |  |  |

## 下单



### 设计图



### 功能描述

1. 用户点击订单详情页面的下单按钮，系统显示订单附加信息页面

2. 用户点击订单附加信息中桌号对应的下拉框选择桌号

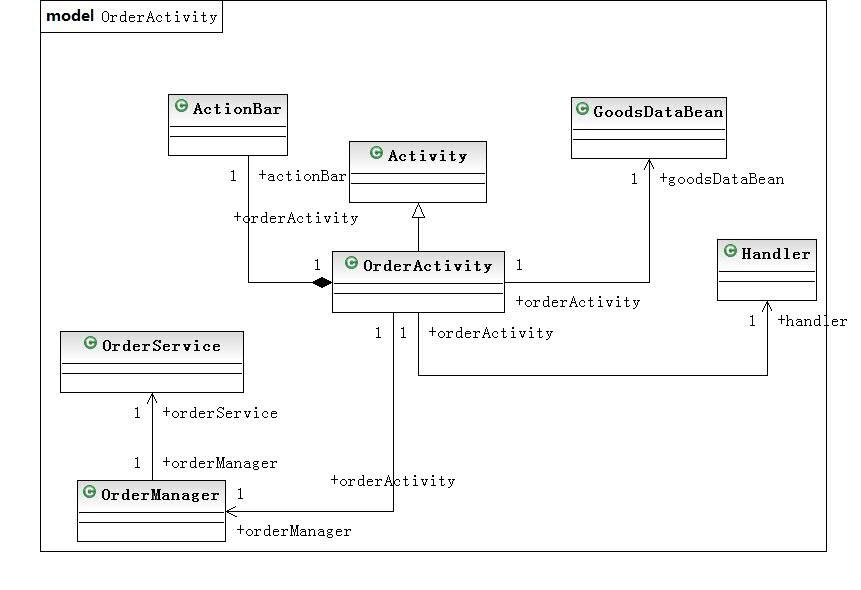
3. 用户点击输入员工号对应的的照相机图标进行的员工号对应的二维码扫描到订单

4. 用户在订单信息页面会员号输入框内输入顾客会员号，查看会员等级对应的折扣信息，系统显示会员信息和打折信息

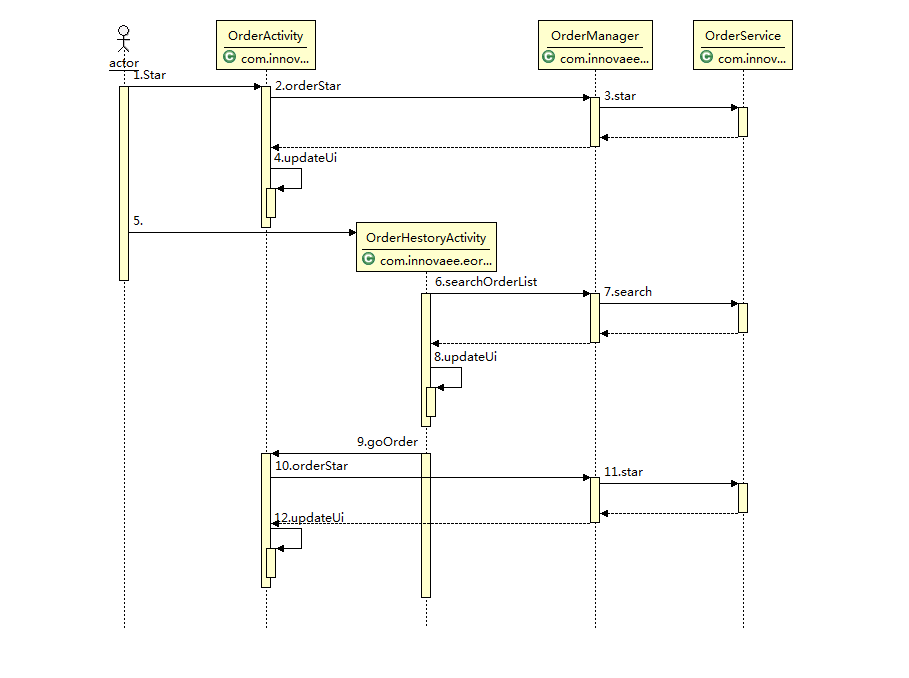
5. 点击下单按钮，系统生成订单

6. 系统显示Msg401

### 类图



### 时序图



### 函数说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.mobile.view | | |
| 类名/接口名 | OrderActivity | | |
| 父类 | Activity | | |
| 实现接口 | N/A | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| selectOrderGoods | List<GoodsDataBean> | 已经选择商品列表 | |
| discount; | Double | 折扣 | |
| 视图部分 |  |  | |
| inputEmployeeId | EditText | 输入员工号文本编辑器 | |
| inputTableId | EditText | 输入桌号文本编辑器 | |
| inputUserId | EditText | 输入会员号文本编辑器 | |
| qrcodeBtn | Button | 二维码扫描按钮 | |
| discountBtn | Button | 获取折扣按钮 | |
| okBtn | Button | 确认按钮 | |
| originalPriceTxt | TextView | 原始价格文本显示器 | |
| discount | TextView | 折扣文本显示器 | |
| realPriceTxt | TextView | 实际价格文本显示器 | |
| allPriceTxt | TextView | 最终价格显示器 | |
| listView | ListView | 订单列表显示ListView | |
| orderAdapter | MyOrderAdapter | 订单数据绑定器 | |
| 消息部分 |  |  | |
| ACTION\_INTENT\_CHANGECOUNT | String | 改变菜品数量广播 | |
| MSG\_UPDATE | int | 刷新界面消息 | |
| MSG\_INITDATA | int | 初始化数据完成消息 | |
| MSG\_UPDATE\_COUNT | int | 更新显示数量消息 | |
| MSG\_UPDATE\_DISCOUNT | int | 更新折扣信息消息 | |
| MSG\_ORDER\_SUCCESS | int | 下单成功消息 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| onCreate | 无 | Bundle savedInstanceState | Activity创建函数，系统调用 |
| onOptionsItemSelected | boolean是否处理 | MenuItem item | ActionBar上的按键消息处理函数，系统调用 |
| initView | 无 | 无 | 初始化控件View |
| initData | 无 | 无 | 初始化控件数据，监听器 |
| getAllPrice() | Double | 无 | 计算菜品的总价格 |
| sendBroadcastToMainActivity | 无 | 无 | 发送广播到主界面，更新显示的数目 |
| displayPrice | 无 | 无 | 刷新显示价格等TextView |
| updateDiscountUi | 无 | 无 | 更新折扣等ui |
| orderSuccessful | 无 | 无 | 下单成功更新ui |
| getDiscountData | 无 | String userId | 获取某个会员号的信息 |
| setListViewHeightBasedOnChildren | 无 | ListView listView | 显示界面的时候需要动态计算listview的高度 |
| orderToService | 无 | List<GoodsDataBean> selectOrderGoods | 下单到服务器 |
| isNumeric | boolean | String str | 判断字符串是否为数字 |

## 历史订单查询



### 设计图



### 功能描述

历史订单查询

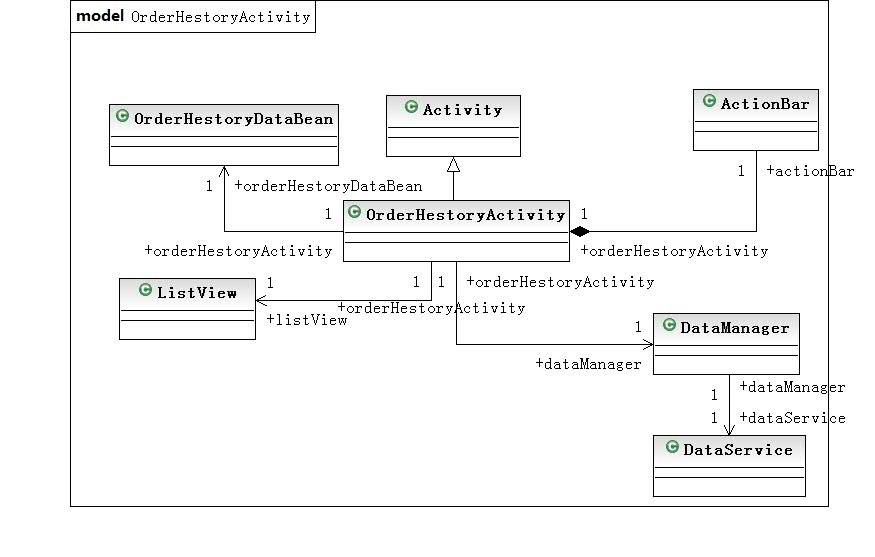
1. 用户在菜单首页的上方的会员号对应的文本框内输入会员号

2. 点击历史订单按钮

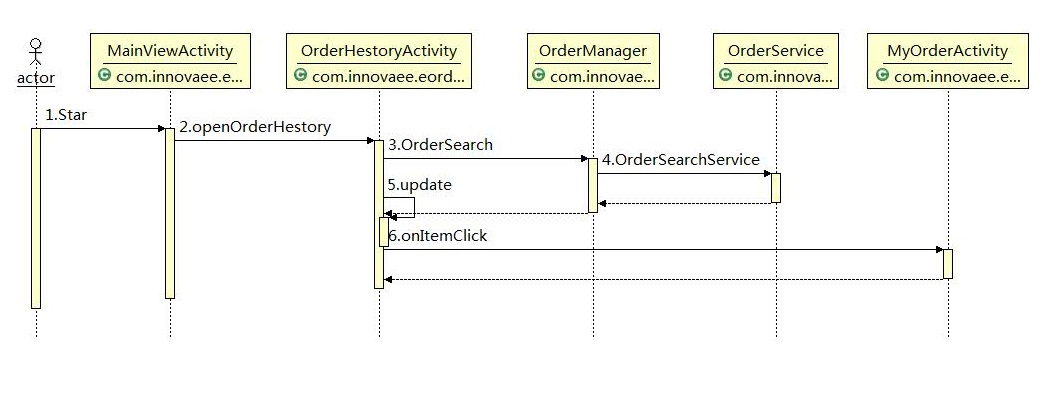
3. 系统显示用户点菜历史订单

4. 用户从上往下滑动，查看订单内菜品信息

### 类图



### 时序图



### 函数说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.mobile.view | | |
| 类名/接口名 | OrderHestory | | |
| 父类 | Activity | | |
| 实现接口 | N/A | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| orderHestoryList | List<OrderHestoryDataBean> | 历史记录订单列表 | |
| userId | String | 会员id | |
| 视图部分 |  |  | |
| listView | ListView | 历史订单列表ListView | |
| orderHestoryAdapter | OrderHestroyAdapter | 订单数据绑定器 | |
| orderHestoryDataBean | OrderHestoryDataBean | 历史订单记录数据结构 | |
| orderHestoryList | List<OrderHestoryDataBean> | 历史记录订单列表 | |
| actionBar; | ActionBar | 界面ActionBar | |
| 消息部分 |  |  | |
| handler | Handler | 消息handler | |
| MSG\_UPDATE | int | 刷新UI消息 | |
| MSG\_INITDATA | int | 初始化数据完成消息 | |
| MSG\_UPDATE\_FAIL | int | 显示失败提示消息 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| onCreate | 无 | Bundle savedInstanceState | Activity创建函数，系统调用 |
| onOptionsItemSelected | boolean是否处理 | MenuItem item | ActionBar上的按键消息处理函数，系统调用 |
| initView | 无 | 无 | 初始化控件View |
| initData | 无 | 无 | 初始化控件数据，监听器 |
| updateUi | 无 | 无 | 更新ListView ui |
| updateFailUi | 无 | 无 | 更新显示失败 |
| openMyOrder | 无 | 无 | 进入我的订单 |
| loadOrderHestoryListData | 无 | String userId | 获取会员的历史订单 |

# 存储数据设计



## 应用设置

### Android常用存储方式说明

在Android应用程序中，数据存储操作是必不可少的。因此，Android系统一共提供了四种数据存储方式。分别是：SharePreference、SQLite、Content Provider和File。这几种方式有着各自的特点，分别适用于不同的场景。

SQLite： SQLite是一个轻量级的数据库，支持基本SQL语法，是常被采用的一种数据存储方式。Android为此数据库提供了一个名为SQLiteDatabase的类，封装了一些操作数据库的API。

SharedPreference： 除SQLite数据库外，另一种常用的数据存储方式，其本质就是一个xml文件，常用于存储较简单的参数设置。

File： 即常说的文件（I/O）存储方法，常用语存储大数量的数据，但是缺点是更新数据将是一件困难的事情。

ContentProvider: Android系统中能实现所有应用程序共享的一种数据存储方式，由于数据通常在各应用间的是互相私密的，所以此存储方式较少使用，但是其又是必不可少的一种存储方式。例如音频，视频，图片和通讯录，一般都可以采用此种方式进行存储。每个Content Provider都会对外提供一个公共的URI（包装成Uri对象），如果应用程序有数据需要共享时，就需要使用Content Provider为这些数据定义一个URI，然后其他的应用程序就通过Content Provider传入这个URI来对数据进行操作。

而本应用主要是为了存储设置参数，参数均比较简单，所以这里比较合适的是选择SharePreference方式。

### 本例实现方式说明

本程序客户端只需要存储简单数据到手机端，比如手机的设置项等等简单参数，所以采用Android中应用比较广泛，轻量级的存储类SharedPreferences．

在Android系统中，这些信息以XML文件的形式保存在 /data/data/　PACKAGE\_NAME /shared\_prefs 目录下。

存储实现方式：

1. 打开Preferences，名称为setting，如果存在则打开它，否则创建新的Preferences

SharedPreferences settings = getSharedPreferences(“setting”, 0);

2. 让setting处于编辑状态

SharedPreferences.Editor editor = settings.edit();

3. 存放数据

editor.putString(“name”, ”xxxxx”);

editor.putString(“URL”, ”xxxxx”);

4. 完成提交

editor.commit();

存储数据读取实现方式：

1. 获取Preferences

SharedPreferences settings = getSharedPreferences(“setting”, 0);

2. 取出数据

String name = settings.getString(“name”, ”默认值”);

String url = setting.getString(“URL”, ”default”);

以上就是Android中SharedPreferences的使用方法，其中创建的Preferences文件存放位置可以在Eclipse中查看：

DDMS->File Explorer /<package name>/shared\_prefs/setting.xml

# 数据缓存设计



## 功能描述

网络应用通常会采用缓存技术，可以进一步大大缓解数据交互的压力，又能提供一定的离线浏览，缓存一般适合下面这样的应用．

1. 提供网络服务的应用

2. 数据更新不需要实时更新，哪怕是3-5分钟的延迟也是可以采用缓存机制。

3. 缓存的过期时间是可以接受的

这样所带来的好处：

1. 减小服务器的压力

2. 提高客户端的响应速度

3. 一定程度上支持离线浏览

常用的Android缓存方式有下面几种方式：

1. 软引用方式SoftReference。

SoftReference 的特点是它的一个实例保存着一个 Java 对象的软引用，该软引用的存在不妨碍垃圾收集器线程对该 Java 对象的回收。也就是说，一旦SoftReference 保存着一个 Java 对象的软引用之后，在垃圾收集器线程对这个 Java 对象回收之前， SoftReference 类所提供的 get() 方法都会返回 这个Java 对象的强引用。另外，一旦垃圾线程回收该 Java 对象之后， get() 方法将返回 null 。

但是目前不推荐使用这种方式。从Android 2.3 (API Level 9) 开始，垃圾回收开始强制的回收掉 soft/weak 引用 从而导致这些缓存没有任何效率的提升。另外，在 Android 3.0 (API Level 11)之前，这些缓存的Bitmap数据保存在底层内存(native memory)中，并且达到预定条件后也不会释放这些对象，从而可能导致程序超过内存限制并崩溃。

1. 使用内存缓存。

内存缓存提高了访问图片的速度，但是要占用不少内存。 LruCache类（在API 4之前可以使用Support Library 中的类 ）特别适合缓存Bitmap， 把最近使用到的Bitmap对象用强引用保存起来（保存到LinkedHashMap中），当缓存数量达到预定的值的时候，把不经常使用的对象删除。

但是使用该种方式，很明显会比较占用内存，所以使用的时候要特别小心。

1. 使用磁盘缓存。

在访问最近使用过的图片中，内存缓存速度很快，但是您无法确定图片是否在缓存中存在。像GridView 这种控件可能具有很多图片需要显示，很快图片数据就填满了缓存容量。

同时您的程序还可能被其他任务打断，比如打进的电话 — 当您的程序位于后台的时候，系统可能会清除这些图片缓存。一旦用户恢复使用您的程序，您还需要重新处理这些图片。在这种情况下，可以使用磁盘缓存来保存这些已经处理过的图片，当这些图片在内存缓存中不可用的时候，可以从磁盘缓存中加载从而省略了图片处理过程。当然， 从磁盘载入图片要比从内存读取慢很多，并且应该在非UI线程中载入磁盘图片。

## 实现方式

本应用也采用了缓存技术，因为本客户端主要是对网络图片的读取比较频繁，需要不断的刷新和加载图片，所以保存网络下载的图片比较重要．客户端采用了对图片进行缓存的技术，结合了上面两种方式，采用了方式2和方式3结合的方式来保存图片，及将从网络端下载的图片保存到缓存和本地sdcard上，使用的时候会先去缓存获取图片，如果获取不到图片再去sdcard获取相应的图片．

实现类如下：

private static final int DEFAULT\_SIZE = 5 \* 1024 \* 1024;

/\*\*

\* SDcard storage path

\*/

private final static String STORAGE\_PATH = Environment

.getExternalStorageDirectory().getPath()

+ "/eOrderMobile";

/\*\*

\* Soft cache for bitmaps kicked out of hard cache

\*/

// private static LruCache<String, SoftReference<Bitmap>> softBitmapCache =

// new LruCache<String, SoftReference<Bitmap>>(

// 5);

private static LruImageCache softBitmapCache = new LruImageCache(

DEFAULT\_SIZE);

private static ImageDataManager instance = null;

＊＊＊＊＊＊

｝

## 函数说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.mobile.util | | |
| 类名/接口名 | ImageDataManager | | |
| 父类 | N/A | | |
| 实现接口 | OnImageLoaderListener | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| softBitmapCache | LruImageCache | 图片缓存 | |
| imageThreadPool | ThreadPoolExecutor | 下载Image的线程池 | |
| activeQueue | HashMap<String, ImageDownloadTask> | 活跃任务队列 | |
| waitTasksQueue | Queue<Runnable> | 被reject的任务队列 | |
| rejectedExecutionHandler | RejectedExecutionHandler | 任务被拒绝执行的处理器 | |
| scheduledRunnable | ScheduledRunnable | 调度Runnable | |
| scheduledExecutorService | ScheduledExecutorService | 调度线程池 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getThreadPool | ThreadPoolExecutor | 无 | 获取线程池的方法，因为涉及到并发的问题，我们加上同步锁 |
| downloadImage | 无 | String url, int hashCode,  OnImageLoaderListener listener | 先从内存缓存中获取Bitmap,如果没有就从SD卡或者手机缓存中获取，SD卡或者手机缓存 没有就去下载 |
| cancelTask | 无 | 无 | 取消正在下载的任务 |
| getBitmapFromCache | Bitmap | String url | 从缓存获取图片 |
| getBitmapFromSdCard | Bitmap | String url | 从sdcard获取图片 |
| saveBmpToSd | 无 | String url, Bitmap bitmap | 保存图片到sdcard |
| hideMedia | 无 | String folder | 在媒体库中隐藏文件夹内的媒体文件 |
| autoCleanCache | 无 | String dirPath | 清理缓存 |
| isSdExits | boolean | 无 | 判断sdcard是否存在 |
| getSdAvailable | int | 无 | 得到sdcard可用空间 |
| initRejectedExecutionHandler | 无 | 无 | 初始化任务被拒绝执行的处理器的方法 |
| hasMoreWaitTask | boolean | 无 | 是否还有等待任务的方法 |
| removeTask | boolean | int hashCode, String url | 删除任务 |
| destory() | 无 | 无 | 释放函数 |
|  |  |  |  |