**摘要**

NFC技术的基本功能是允许某种设备（通常是手机）在限定范围内从另一种设备或NFC标签中收集数据。从某种程度上讲，该技术与蓝牙(Bluetooth)技术很类似。但NFC技术无需对相连接的两款设备都加以设置，而只需通过简单的设备接触，就可建立起无线连接。既然借助于NFC技术，我们能够从标签中提取信息，那么我们可以把支付信息写入到商家专用的标签里面，消费者只需要用支持NFC的手机读取标签中的支付信息，就可以通过手机很方便的完成支付。

**关键字：NFC 近场通信 手机支付 NFC支付**

**Abstract**

The basic function of NFC technology is to allow some equipment (mobile phone most of the time) to collect data from other equipment of NFC tag. From some degree, the technology is very similar to Bluetooth technology. But NFC technology does not need the two equipment which is going to connect to have some extra settings, only by simply touching with each other, a connection is established. Since we can get information from tags with NFC technology, We can write paying information into a tag specially prepared for commercial tenant, Consumers just need to touch the tag with their NFC supported mobile phone, the paying is successfully done conveniently.

Key words:NFC, Near Filed Communication,Mobile Pay, NFC Pay

# 1、研究背景和现状

## 1.1 NFC的起源和现状

大概在 2003 年，当时的 philips 半导体和 Sony 公司计划基于非接触式卡技术发展一种与之兼容的无线通讯技术。飞利浦派了一个团队到日本和sony工程师一起闭关三个月，然后联合对外发布关于一种兼容当前 ISO14443 非接触式卡协议的无线通讯技术，取名NFC(Near Field Communication)。

该技术规范定义了两个 NFC 设备之间基于 13.56MHz 频率的无线通讯方式，在 NFC 的世界里没有读卡器，没有卡，只有 NFC 设备。该规范定义了 NFC 设备通讯的两种模式:主动模式和被动模式。并且分别定义了两种模式的选择和射频场防冲突方法、设备防冲突方法，定义了不同波特率通讯速率下的编码方式、调制解调方式等等最最底层的通讯方式和协议，说白了就是解决了如何交换数据流的问题。该规范最终被提交到ISO标准组织获得批准成为正式的国际标准，这就是 ISO18092，后来增加了 ISO15693 的兼容，形成新的 NFC 国际标准IP2，也就是 ISO21481。同时ECMA（欧洲计算机制造协会）也颁布了针对 NFC 的标准，分别是 ECMA340和 ECMA352，对应的是ISO18092[2] 与ISO21481[3] ，其实两个标准内容大同小异，只是 ECMA 的是免费的，大家可以到网上下载到,而ISO标准是收费的。不过，所幸的是，为了促进标准化，ISO/IEC 18092：2013 和ISO/IEC 21481：2012版均可在ISO官方网站上下载到免费的电子版。

为了加快推动NFC产业的发展，当时的飞利浦、[SONY](http://baike.baidu.com/newpreview?t=1405665908684) 和[诺基亚](http://baike.baidu.com/newpreview?t=1405665908684)联合发起成立了 NFC 论坛，旨在推动行业应用的发展，定义相关基于NFC应用的中间层规范，包括一些数据交换通讯协议　NDEF，包括基于非接触式标签的几种　NFC tag　规范，主要涉及到卡片内部数据结构定义，NFC 设备（手机）如何识别一个标准的 NFC 论坛兼容的标签，如何解析具体应用数据等等相关规范，目的是为了让不同的　NFC设备之间可以互连互通。

NFC具有成本低廉、方便易用和更富直观性等特点，这让它在某些领域显得更具潜力——NFC通过一个芯片、一根天线和一些[软件](http://baike.baidu.com/newpreview?t=1405665908684)的组合，能够实现各种设备在几厘米范围内的通信，而费用仅为2~3欧元。据ABIReasearch有关NFC有最新研究，NFC市场可能发迹于移动手持设备。ABI估计，到2005年以后，市场会出现采用NFC芯片的智能手机和增强型手持设备。到2009年，这种手持设备将占一半以上的市场。研究机构Strategy Analytics预测，至2011年全球基于移动电话的非接触式支付额将超过360亿美元。如果NFC技术能得到普及，它将在很大程度上改变人们使用许多电子设备的方式，甚至改变使用信用卡、钥匙和现金的方式。NFC将会成为智能手机、平板这些移动设备的标配功能，此外，未来数字电视、智能手机、耳机、手表、计步器、汽车、工业生产、环境监控、智能交通、智能家居、智能医疗、健康监测等领域中都会加入NFC功能。

## 1.2 手机支付的现状

手机支付也称为移动支付（Mobile Payment），就是允许移动用户使用其移动终端对所消费的商品或服务进行账务支付的一种服务方式。

2013年是中国移动支付的崛起之年，随着O2O（线上到线下）模式的深入探索、手机支付技术的升级，移动支付给整个支付行业带来了产业形态和商业模式的大变革。2013年第三方移动支付市场交易规模达12197.4亿，同比增速707.0%。2014年第2季度中国第三方支付市场移动支付交易额规模达到16353亿，环比增长0.2%。移动远程支付正快速进入高速成长期，目前市场上出现了琳琅满目的手机支付软件，手机支付是一种必然趋势。



图1 市场主流手机支付软件

## 1.3基于NFC的手机支付系统现状

支付和票务业务是应用最早的NFC业务，因为NFC可以借助于目前已大量普及的支持非接触卡片的基础设施。

2016年2月18日apple pay登录中国大陆，瞬间火爆，当天绑定银行卡3千万多万，而且因太多人绑定，服务器接近宕机。首批12家银行已经支持Apple Pay（中国农业银行，中国银行，上海银行，中国建设银行，中信银行，招商银行，民生银行，广发银行，中国工商银行，兴业银行，中国邮政储蓄银行，上海浦东发展银行）。之后还会增加7家银行（平安银行，光大银行，广州银行，华夏银行，宁波银行，交通银行，北京银行）目前肯德基、麦当劳、711便利店等线下店可以Apple Pay结账，苹果称之后还会增加。加上目前支持apple pay的设备并不算多，可以说当前支持的场景还不多。但是Apple pay无疑给基于NFC的手机支付系统带来了巨大的希望，目前基于NFC的手机支付还有很大的空缺，可以说以后各大手机厂商都会竞相支持NFC功能，各大运营商也在积极推出支持NFC的SIM卡，NFC手机支付前景广阔。

# 2、NFC理论知识：

## 2.1 NFC技术介绍

NFC技术由非接触式射频识别(RFID)及互联互通技术整合演变而来,在单一芯片上结合感应式读卡器、感应式卡片和点对点的功能，能在短距离内与兼容设备进行识别和数据交换。

这项技术最初只是RFID技术和网络技术的简单合并,现在已经演变成一种短距离无线通信技术,发展态势相当迅速。与RFID不同的是，NFC技术具有双向连接和识别的特点，工作于13.56MHz频率范围，作用距离10厘米左右。

NFC的系统主要由NFC芯片和NFC天线组成。NFC芯片具有相互通信功能、计算能力，并具有加密/解密模块(SAM)；NFC天线是一种近场耦合天线，耦合方式是线圈磁场耦合

NFC具有通信距离短、传输速率较低、采用非接触式点对点连接，无需发现和配对、等特点，大多数情况以被动方式连接，反应时间只需0.1s，几乎不消耗电量。

NFC通信不同的传输速率具有不同的帧结构。在106 kb/s的速率下存在三种帧结构：短帧、标准帧、检测帧；速率212 kb/s和424 kb/s的帧结构相同，由前同步码、同步码、载荷长度、载荷和校验码顺序组成。

## 2.2 Mifare UltraLight卡的存储结构

Mifare UltraLight又称为MF0，它是一个低成本、小容量的卡片。低成本，是指它是目前市场中价格最低的遵守ISO14443A协议的芯片之一；小容量，是指其存储容量只有512bit(Mifare S50有8192bit)。

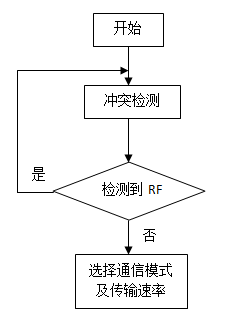
Mifare UltraLight的512bit存储容量分成16个Page，每个Page包含4个字节，在本程序中只用了page4-page7共16个字节，存储结构图如下所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 页号 | Byte0 | Byte1 | Byte2 | Byte3 | 说明 |
| 0 | SN0 | SN1 | SN2 | BCC0 | 只读，存放卡的序列号：Page0前3字节+整个Page1 |
| 1 | SN3 | SN4 | SN5 | SN6 |
| 2 | BCC1 | 保留 | LOCK0 | LOCK1 | 只读，通过设置LOCK0和LOCK1可以讲16个page设为只读 |
| 3 | OTP0 | OTP1 | OTP2 | OTP3 | 可读写，一次性交易计数器，不可逆 |
| 4 | Data0 | Data1 | Data2 | Data3 | 可读写，数据存放区域 |
| 5 | Data0 | Data1 | Data2 | Data3 |
| 6 | Data0 | Data1 | Data2 | Data3 |
| 7 | Data0 | Data1 | Data2 | Data3 |
| 8 | Data0 | Data1 | Data2 | Data3 |
| 9 | Data0 | Data1 | Data2 | Data3 |
| 10 | Data0 | Data1 | Data2 | Data3 |
| 11 | Data0 | Data1 | Data2 | Data3 |
| 12 | Data0 | Data1 | Data2 | Data3 |
| 13 | Data0 | Data1 | Data2 | Data3 |
| 14 | Data0 | Data1 | Data2 | Data3 |
| 15 | Data0 | Data1 | Data2 | Data3 |

*表1 Mifare UltraLight的存储分布图*

## 2.3 NFC初始化过程

NFC设备的默认状态均为目标状态。目标设备不产生射频场，保持静默以等待来自于发起者的指令。应用程序能够控制设备主动从目标状态转换为发起状态。设备进入发起状态后开始冲突检测，只有在没有检测到外部射频场时，才激活自身的磁场。应用程序确定通信模式和传输速率后，开始建立连接传输数据。初始化过程流程图如下：



## 2.4传输过程

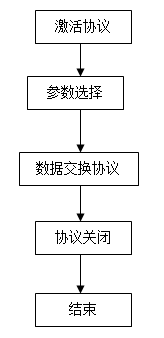
传输协议包含以下三个主要过程：

(1)协议激活。负责发起设备和目标设备间属性请求和参数选择的协商。

(2)数据交换协议。半双工工作方式，以数据块为单位进行传输，包含错误处理机制。

(3)协议关闭。在数据交换完成后，发起设备执行协议关闭过程，包括撤消选中和释放连接。

执行流程图如下：



## 2.5 NFC模式

基于NFC技术的业务支持三种工作模式：

(1)卡模式：将具有NFC功能的设备模拟成一张非接触卡。如门禁卡、银行卡等。

(2)读卡器模式：即作为非接触读卡器使用。

(3)点对点模式：即将两个具备NFC功能的设备链接，实现点对点数据传输。

NFC还有两种工作模式：

主动模式 ：在主动模式下，每一个想要发送数据的设备，都必须产生自己的电磁场

被动模式：只有需要主设备产生电磁场，其他设备，通过负载调制的方法，来和主设备通信，其他设备虽然不产生磁场，但是可以将主设备的磁场强度拉低，从而达到通信的目的，标签卡工作在被动模式下。被动模式主要用在读取标签卡的场景，此时标签的能量，是通过电感耦合的方式从主设备获取的。



图 NFC工作模式电路图

## 2.6 基于NFC的应用

目前基于NFC的应用按照业务应用模式主要分为三类：

卡模式的应用：加油卡、停车卡、公交、优惠券、折扣券、机票、门禁、会员卡等

阅读器模式的应用：广告、信息查询、2013年宝马推出了一款能打开预定酒店的NFC车钥匙、LG在CES上展示多款家用电器，包括一款新的电冰箱、洗衣机、吸尘器机器人和烤箱 ，都配备NFC功能，用户将智能手机触碰到这些智能家电的 NFC 标签处就 可以开始对他们进行控制

点对点模式的应用：电子名片

# 3、基于NFC的手机支付软件系统

本文设计的NFC手机支付软件系统是基于NFC的卡模式的应用，基本工作原理是商家将支付信息储存到NFC标签中，包含收款账户，收款金额，备注等信息。消费者只需要用支持NFC的手机，使用本文设计的软件，就能解析出支付信息，完成支付。

## 3.1 模块设计

为实现上述支付功能，NFC手机支付软件需要如下几个功能模块：

#### 3.1.1 数据库模块

数据库的设计：使用本文设计的软件，消费者需要注册一个帐号，用来保存消费者可用余额、消费记录、以及其他从银行数据库得到的信息，所以软件还需要银行的支持。

为了测试方便与可行，这里使用的是Android系统支持的本地SQLite数据库，为了实现数据库功能，这里编写了一个用户自定义的数据库辅助类SqliteOpenHelper继承SQLiteOpenHelper，这个模块主要完成数据库的创建，该模块代码如下：

**package** com.example.nfcpay;

**import** android.content.Context;

**import** android.database.sqlite.SQLiteDatabase;

**import** android.database.sqlite.SQLiteOpenHelper;

**public** **class** SqliteOpenHelper **extends** SQLiteOpenHelper {

//数据库名：

**private** **static** **final** String ***DBNAME***="nfcpay.db";

//表名

**private** **static** **final** String ***USERINFO***="user";

**private** **static** **final** String ***TRANSLOG***="translog";

**private** **static** **final** **int** ***TESTVERSION***=1;

**public** SqliteOpenHelper(Context context) {

**super**(context, ***DBNAME***, **null**, ***TESTVERSION***);

// **TODO** Auto-generated constructor stub

}

//初始化，创建表

@Override

**public** **void** onCreate(SQLiteDatabase db) {

// **TODO** Auto-generated method stub

String sql1="create table"+" "+***USERINFO***+"(username varchar(20),password varchar(20),account integer)";

String sql2="create table"+" "+***TRANSLOG***+"(date datetime,username varchar(20),action varchar(20),value integer)";

db.execSQL(sql1);

db.execSQL(sql2);

}

//失败后删除，重新创建

@Override

**public** **void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {

// **TODO** Auto-generated method stub

**if**(newVersion>oldVersion)

{

String sql1="drop table if exists"+***USERINFO***;

String sql2="drop table if exists"+***TRANSLOG***;

db.execSQL(sql1);

db.execSQL(sql2);

**this**.onCreate(db);

}

}

}

主要完成创建用户user和交易记录translog这两个数据库

User包含username(用户名)、password(密码)、account(账户金额)三个元素

Log包含data(交易时间)、username(用户名)、action(交易行为)、value(交易金额)四个元素

#### 3.1.2 注册登录模块

注册登录模块完成用户的创建和检查用户的帐号密码是否匹配的功能。

提交注册时，先检查数据库中该用户是否已经存在，如果存在则注册失败，如果不存在则添加到数据库中。处理注册的函数如下：

**private** **boolean** regUser(String userName,String passWord)

{

SQLiteDatabase sdb = helper.getReadableDatabase();

String sql = "select \* from user where username=?";

Cursor cursor = sdb.rawQuery(sql, **new** String[] {userName});

**if**(cursor.moveToFirst()==**true**){

cursor.close();

**return** **false**;

}

**else**

{

String sql1="insert into user(username,password,account) values(?,?,?)";

Object obj1[]={userName,passWord,0};

sdb.execSQL(sql1, obj1);

//用来保存用户交易记录，首次注册时交易记录为存款0

SimpleDateFormat sDateFormat = **new** SimpleDateFormat("yyyy-MM-dd hh:mm:ss");

String curTime = sDateFormat.format(**new** Date());

String sql2="insert into translog(date,username,action,value) values(?,?,?,?)";

Object obj2[]={curTime,userName,"充值",0};

sdb.execSQL(sql2, obj2);

sdb.close();

**return** **true**;

}

}

程序流程图如下：

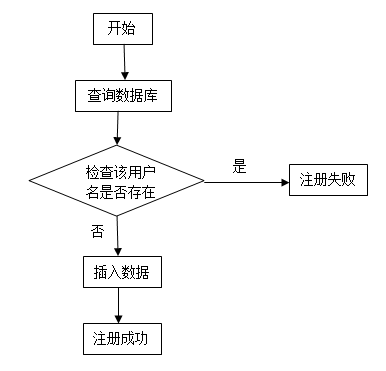


图 注册流程图

提交登录时，检查数据库该用户是否存在，并且密码匹配，如果成功则进入程序的主界面，如果用户不存在或者密码不匹配，则登录失败。处理登录的函数如下：

**private** **boolean** checkUser(String userName,String passWord)

{

SQLiteDatabase sdb = helper.getReadableDatabase();

String sql = "select \* from user where username=? and password=?";

Cursor cursor = sdb.rawQuery(sql, **new** String[] {userName, passWord});

**if**(cursor.moveToFirst()==**true**){

cursor.close();

**return** **true**;

}

**return** **false**;

}

流程图如下：

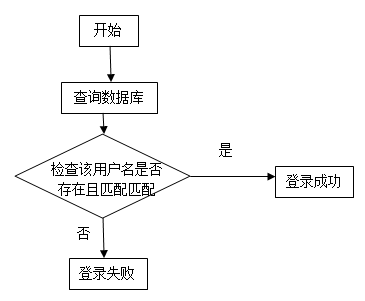


图 登录流程图

#### 3.1.3 交易模块

交易模块是NFC支付软件的核心，主要完成NFC数据的交互和支付功能。

首先通过getDefaultAdapter判断当前手机是否有nfc适配器，如果没有nfc适配器则程序无法正常工作。当检测到NFC标签时，会触发NFC的回调函数onNewIntent，在该函数中调用readTag得到标签中的字符串信息。

NFC读取标签功能实现如下:

**public** **boolean** readTag(Tag tag)

{

String[] techlist=tag.getTechList();

String inPut;

**if**(Arrays.*toString*(techlist).contains("MifareUltralight"))

{

MifareUltralight mifareUltralight=MifareUltralight.*get*(tag);

**try**

{

mifareUltralight.connect();

**byte**[] data=mifareUltralight.readPages(4);

//获取从NFC数据，保存到局部变量inPut中

inPut = **new** String(data,Charset.*forName*("US-ASCII"));

ReadDecode(inPut);

**return** **true**;

}

**catch** (Exception e)

{

// **TODO**: handle exception

**return** **false**;

}

**finally**

{

**try**

{

mifareUltralight.close();

}

**catch** (IOException e)

{

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **false**;

}

}

}

**else**

{

Toast.*makeText*(**this**, "不是MifareUltralightle类型", Toast.***LENGTH\_SHORT***).show();

**return** **false**;

}

}

其中包含了NFC读取数据的完整过程，将读取的数据保存到一个字符串，然后再调用ReadDecode函数按照协商好的格式进行解析。数据格式设计思路如下：

手机从NFC标签获得16个字节流的数据，前8个字节用来保存收款账户，后8个字节用来保存收款金额，目前只支持MifareUltralight标签。

定义数据帧格式如下：

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

+-----+-----+-----+-----+-----+-----+----+-----+-----+-----+-----+----+-----+-----+----+-----+

| | | | | | | | **|** | | | | | | | |

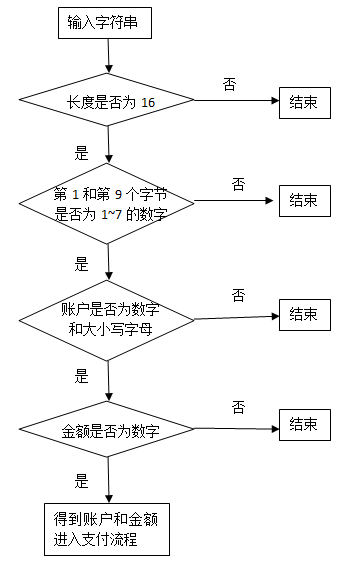
+-----+-----+-----+-----+-----+-----+----+-----+-----+-----+-----+----+-----+-----+----+-----+

第1个字节保存账户的长度，取值1-7，然后从第2个字节开始取出账户；

第9个字节保存金额的长度，取值1-7，然后从第10个字节开始取出金额

第1个和第9个字符必须为1到6的数字，账户名必须只能包含大小写字母和数字，采用正则表达式[0-9a-zA-Z]+进行判断；金额必须为数字，用正则表达式[0-9]+进行判断。如果解析的字符串合法，则进行下一步测支付操作，否则结束本次操作。

读取的数据流处理流程图如下：



*图 数据处理流程*

处理完字符串之后再调用userPay函数进行支付操作，支付操作流程为先查询余额是否充足，如果不够则支付失败，如果余额充足则更新用户数据库，设置账户的余额为交易前的余额减去交易金额，然后将本条交易记录保存到数据库的translog表中，完成交易。

交易功能的代码段如下：

**public** **void** userPay()

{

**if**(account < payValue)

{

Toast.*makeText*(MainActivity.**this**,"余额不足请充值", Toast.***LENGTH\_SHORT***).show();

**return**;

}

account-=payValue;

SQLiteOpenHelper helper = **new** SqliteOpenHelper(**this**);

SQLiteDatabase sdb = helper.getReadableDatabase();

//更新数据库中该用户的余额

String sql1 = "update user set account=? where username=?";

Object obj1[]={account,userName};

sdb.execSQL(sql1,obj1);

//更新交易记录

SimpleDateFormat sDateFormat = **new** SimpleDateFormat("yyyy-MM-dd hh:mm:ss");

String curTime = sDateFormat.format(**new** Date());

String sql2="insert into translog(date,username,action,value) values(?,?,?,?)";

Object obj2[]={curTime,userName,"支付",payValue};

sdb.execSQL(sql2, obj2);

sdb.close();

Toast.*makeText*(MainActivity.**this**,"支付成功", Toast.***LENGTH\_SHORT***).show();

edit\_account.setText(""+account);

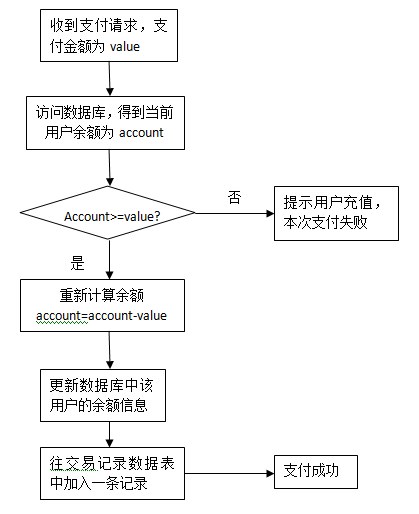
//清空全局变量支付对象和支付金额

payTo = "";

payValue = 0;

}

处理流程图如下：



*图 交易流程*

#### 3.1.4 管理交易记录模块

用于显示用户的交易记录，并提供清除功能。

该部分功能关键之处在于数据库的访问和遍历，访问数据库使用自定义的类SqliteOpenHelper的一个对象，调用该对象的getReadableDatabase方法得到数据库对象，然后使用SQL语句进行查找，将结果保存到一个SimpleAdapter对象中，然后适配到本模块主界面的ListView视图中。显示交易记录的函数实现如下所示：

**public** **void** showLog(String userName)

{

// 创建一个List集合，List集合的元素是Map

List<Map<String, Object>> listItems =

**new** ArrayList<Map<String, Object>>();

SQLiteOpenHelper helper = **new** SqliteOpenHelper(**this**);

SQLiteDatabase sdb = helper.getReadableDatabase();

String sql = "select \* from translog where username=?";

Cursor cursor = sdb.rawQuery(sql, **new** String[] {userName});

**int** logCount = 0;

**while** (cursor.moveToNext())

{

logCount++;

String text = ""+logCount+". "+cursor.getString(0)+" "+cursor.getString(1)+" "+cursor.getString(2)+" ￥"+cursor.getInt(3);

Map<String, Object> listItem = **new** HashMap<String, Object>();

listItem.put("logText", text);

listItems.add(listItem);

}

// 创建一个SimpleAdapter

SimpleAdapter simpleAdapter = **new** SimpleAdapter(**this**, listItems,

R.layout.***logitem***,

**new** String[] {"logText"},

**new** **int**[] {R.id.***logText***});

ListView list = (ListView) findViewById(R.id.***listView***);

// 为ListView设置Adapter

list.setAdapter(simpleAdapter);

}

## 3.2 界面设计

本程序主要有三个界面布局，第一个是登录界面，第二个是交易界面，第三个是交易记录界面

#### 3.2.1 登陆界面

最外层采用竖直摆放的线性布局器组件LinearLayout，里面有一个ImageView，一个相对布局器组件RelativeLayout，一个版权的TextView。

ImageView：用于显示程序的广告语等。

相对布局器RelativeLayout用于制作登录框，内含两个TextView、两个EditText和两个Button。

版权TextView：用于显示程序的开发单位年限等版权信息。

#### 3.3.3 交易界面

交易界面也是程序的主界面，主要用于显示用户信息、提示操作等，交易界面的Activity还完成监听NFC事件的工作，是程序的核心部分。

主要由几个显示用户信息的TextView，和一个显示提示信息的NFC磁场图片ImageView。

#### 3.3.4 交易记录界面

本界面主要用于显示当前用户的交易记录，采用ListView组件，外加一个清除交易记录按钮，该按钮的事件监听器完成清除用户交易记录的功能。

## 3.4 整体设计

#### 3.4.1 MVC设计模式

程序主要遵循MVC的设计思想，实现组件分离各司其职。

组件之间的关系如下图所示：

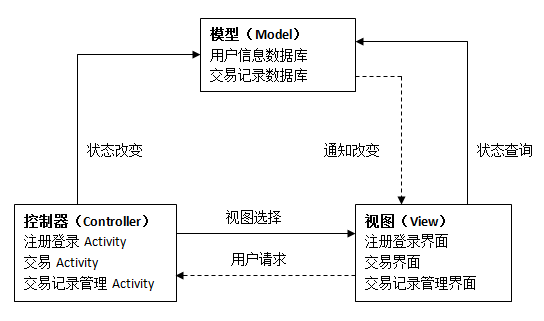


图 组件之间的关系

#### 3.4.2 Activity之间的关系

程序三个Activity之间的跳转方向如下图所示：

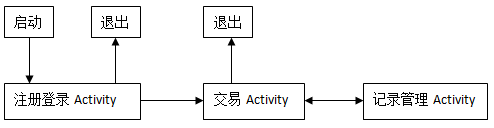


图 Activity之间的跳转关系

开始启动程序只能进入注册登录Activity，从注册登录的Activity可以直接退出，登录成功之后进入交易Activity。

在交易Acitvity可以完成交易，也可以直接退出，还能进入记录管理Activity查看和删除交易记录，但是不能再回到注册登录Activity，此调用为单向。

在记录管理Activity能够返回交易Activity，此过程为双向的，但是从记录管理Activity不能直接退出程序。

各个Activity之间使用SharedPreferences类传递当前登录的用户名等信息。

## 3.5 运行效果

# 4、总结

本次毕业设计课程让我受益匪浅blablabla

## 4.1 遇到的问题和解决方法

问题1：开始设计的时候在支持NFC的手机上调试的，后来拿到一台不支持NFC的手机上运行出现异常

解决方法1：在交易Activity的onCreate方法中对getDefaultAdapter返回的值进行判断，如果为空说明不支持NFC功能，程序后面的部分也不执行。

问题2：在交易记录管理Activity中，只能显示最后一条交易记录，前面的被覆盖

解决方法2：将数据库中每一条交易记录重组成一个字符串，使用simpleAdapter动态加载到界面。

## 4.2 程序改进

程序主要是用作手机支付，因此安全性是必须要考虑的因素。目前程序中使用的数据传输都是明文传输，有不安全因素，今后考虑在NFC交换数据的时候，对数据进行加密处理，在存取数据库数据的时候，也需要一定的加密手段

程序支持的NFC卡类型单一，当前仅支持MifareUltralightle类型的NFC卡片，这一点可以改进为支持全种类的NFC卡。

**致谢**

感谢blablabla

参考文献：

1．RFID技术在手机支付中的应用. 中国移动物联网.

2． ISO/IEC 18092:2013.国际标准组织.

3． ISO/IEC 21481:2012.国际标准组织.

4． 近距离无线通信技术NFC的发展历史.阿拉丁建筑电气论坛.

5． NFC无线功能.来自互联网.

6． 浅析物联网技术应用中RFID与NFC的区别.中国移动物联网.

7． NFC维基百科.维基百科.

8． NFC手机一卡通能否真的把一卡通推向新的发展高潮 .NFC手机一卡通.

9．通讯（NFC）标签种类.易传网.

10． NFC时代门禁技术未来的发展及应用.中国安防行业网.

11. 周虹.手机支付——我国支付领域金融科技发展策略选择[J].中央财经大学学报.29（7）

12. 于淑华.手机支付研究与探索[D].北京邮电大学.211.3

13. 手机支付行业现状及发展趋势分析.中国产业调研网