技术短文

**应用简介**

QR-Checkin是一款安卓平台上的二维码课程签到应用。其功能为：

**教师账号**

1. 生成课程二维码
2. 查询课程考勤记录

**学生账号**

1. 扫描二维码进行签到
2. 查询课程考勤记录

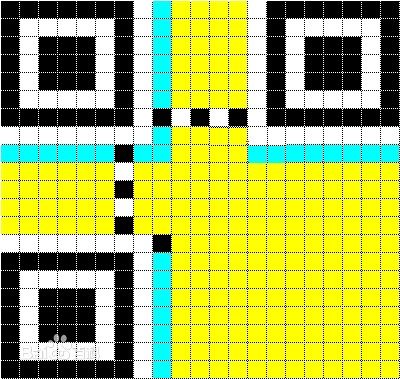
告别纸质版签到的耗时、容易疏漏、整理的不便，告别指纹考勤的排队麻烦、指纹识别不灵敏，开始二维码这一简单快捷的签到方式吧！

**相关技术**

小组应用最核心的部分当算是二维码的相关操作了，这一功能我们采用了zxing来进行实现。

在使用Zxing前我们有必要先了解二维码的原理和工作机制，这样才能更好地完成这一核心功能。二维码说白了就是一维码的平面拓展，在水平和垂直方向都能够存储信息，它利用计算机内部逻辑基础的0、1比特流概念，使用与之相对应的几何形体来表示文字或者数值信息，再通过图像处理设备进行辨析，达到信息自动处理的目的。

二维码分为矩阵式和行列式两种。矩阵式二维码是在一个矩形空间中通过黑白像素的不同分布来形成编码，也就是我们日常熟知的二维码形式。在矩阵元素位置上，出现方点、圆点或其他形状点表示[二进制](http://baike.baidu.com/view/18536.htm)“1”，不出现点表示二进制的“0”，点的排列组合确定了矩阵式二维码所代表的意义。



如上图所示的一个21\*21的矩阵中，[黑白](http://baike.baidu.com/subview/378072/5129411.htm)的区域在QR码规范中被指定为固定的[位置](http://baike.baidu.com/subview/84797/8389933.htm)，称为寻像图形（finder pattern）和定位图形(timingpattern)。寻像图形和定位图形用来帮助解码程序确定图形中具体符号的坐标。黄色的区域用来保存被编码的数据内容以及纠错信息码。蓝色的区域，用来标识纠错的级别（也就是Level L到Level H)和所谓的"Mask pattern",这个[区域](http://baike.baidu.com/view/373715.htm)被称为“[格式化](http://baike.baidu.com/subview/902/5037744.htm)信息”（format information）。

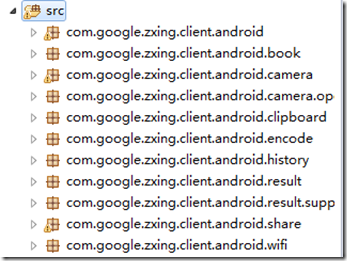
行列式二维码就是在一维码的基础上按需要堆积成二行或多行。它在编码设计、校验原理、识读方式等方面继承了一维条码的一些特点，识读设备与条码印刷与一维条码技术兼容。但由于行数的增加，需要对行进行判定，其译码算法与[软件](http://baike.baidu.com/view/37.htm)也不完全相同于一维条码。

Zxing是一个以将条码扫描器作为独立app能够单独使用或被其它app使用为目的的项目。它的目录分为三个子项，分别是android-integration、androidtest和Barcode Scanner（简称BS）。

Android-intergration的作用是为调用方提供一个简单的方法实现扫描，并接受扫描的结果，且不必了解BS的代码原理，代价是必须把作为独立应用的BS一同与调用方的app安装，否则可能无法使用；

Androidtest则是一个BS的应用实例，由于它依赖android-integration，而且大部分操作都在BS中完成，所以不细说，直接看BS；

Barcode Scanner是zxing的核心，一款功能强大的条码扫描器，它的包结构如图所示：



android：与CaptureActivity直接相关的核心组件。包含了发生震动管理器，闪光灯等等；

book：如果查询的结果是图书信息，用户可以选择查询该书的更进一步的详细信息，该包即包含了搜索与展示书籍的相关类；

camera/camera.open：摄像头相关组件，核心类是CameraManager；

clipboard：剪贴板；

encode：编码功能的各个组件集合。核心类为QRCodeEncoder，最终实施编码的是MultiFormatWriter类；

history：扫描历史管理，核心类是HistoryManager；

result：条码扫描的结果被分为不同的类型，所有的类型都定义在com.google.zxing.client.result.ParsedResultType中，对于不同的类型都有对应的处理方法：xxxResultHandler，所有的ResultHandler都包含在此包中。不同的xxxResultHandler还提供了扫描结果页面要展示几个button，每个button的文本以及需要绑定的事件等等；

result.supplement：对已经扫描并解码的结果做额外处理的工具集。比如扫描出来的是isbn号，如果在设置中选择了“检索更多信息”则会在扫描出isbn号之后自动去网上查询该书的信息，最后将书的信息展示出来，而如果没选中，则只会将isbn号码展示；

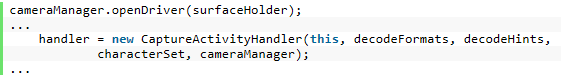
share：分享二维码功能，亦是编码功能的入口所在；

wifi：是WifiResultHandler的辅助类集合。如果扫描到的二维码是对wifi信息的编码，那么最终扫描结果页会展示一个“连接到网络”的按钮，点击此按钮就会自动尝试连接。该包中所包含的类则是链接网络所需的工具类。

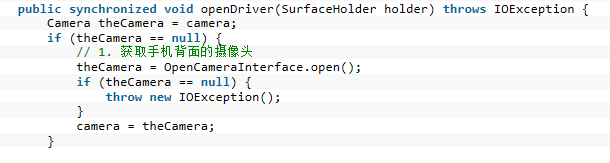
按照功能来分，由扫描、历史和分享三部分组成。利用它进行扫码与解码的流程可简单归纳如下：

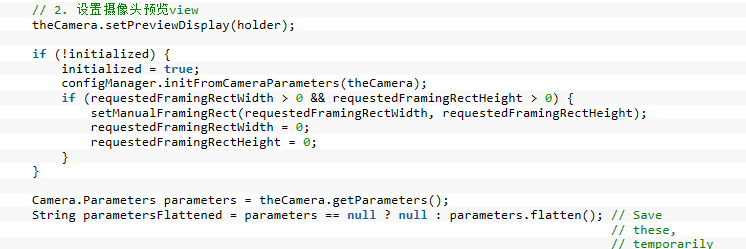
1. 打开预览界面，即借助camera的功能在屏幕中看到预览的条码，根据camera的工作原理可知是调用了Camera.startPreview()即可看到预览界面；
2. 捕捉画面，调用Camera.setOneShotPreviewCallback()，向回调函数Camera.PreviewCallback.onPreviewFrame()中传回当前画面的字节数组。
3. 解析，拿到字节数组以后通过core的逻辑层来进行解码工作，不属于zxing的关注范围。

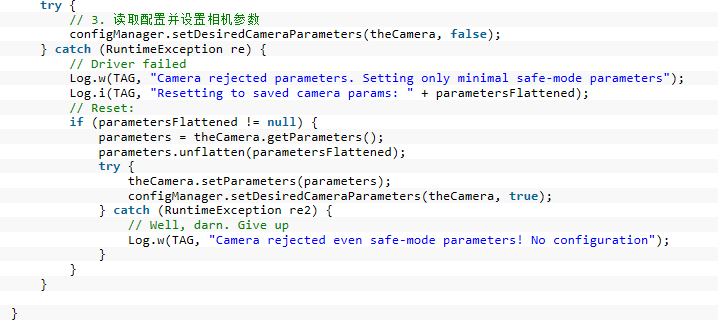
扫描部分可以总体归纳为4个步骤：配置并启动Camera、构建preview与扫描窗口、捕捉画面并解码、将解码结果交给不同的处理函数处理。

第一个步骤中，启动Camera通过CaptureActivity.initCamera来进行，核心代码：

配置Camera则通过CameraManager实现，主要做了三件事：







CameraConfigurationManager是相机辅助类，主要用于设置相机的各类参数。核心方法有两个：

initFromCameraParameters：计算了屏幕分辨率和当前最适合的相机像素

setDesiredCameraParameters：读取配置设置相机的对焦模式、闪光灯模式等等

CaptureActivityHandler类是一个针对扫描任务的Handler，可接收的message有启动扫描（restart\_preview）、扫描成功（decode\_succeeded）、扫描失败（decode\_failed）等等。

第二个步骤，在CaptureActivityHandler.restartPreviewAndDecode中，通过调用activity.drawViewfinder()来构造扫描窗口。

相机的preview界面显示出来后即可开始扫描，所以需要监听preview是否已经显示这个事件，这就是Camera.PreviewCallback的作用。PreviewCallback.onPreviewFrame做的事便是当preview界面展示出来的时候向DecodeHandler发送一个decode消息，DecodeHandler收到该消息后会执行decode方法来解码。

注意，检测并触发捕获画面动作的，是Camera.setOneShotPreviewCallback()这个方法。该函数被调用后，如果预览界面已经打开，就会将包含当前preview frame的byte数组传给回调函数，此时再向DecodeHandler发送decode消息。

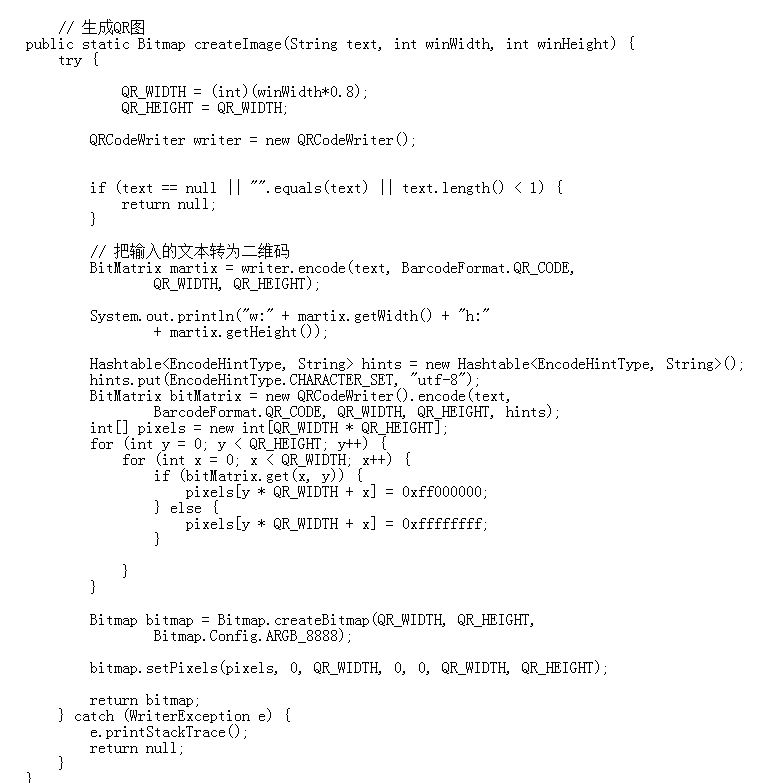
第三个步骤调用DecodeHandler.decode方法；

第四个步骤，当DecodeHandler.decode完成解码后会向CaptureActivityHandler发消息。如果编码成功则调用CaptureActivity.handleDecode方法对扫描到的结果进行分类处理。

历史则是对扫描成功的条码进行一次记录，逻辑相对简单；

分享功能本身比扫描简单很多，我们的软件也没有用到这个功能，故不作赘述。

上面是对扫码与解码的实现阐述，接下来是对二维码生成的实现进行简单的说明。小组实现该功能的思路是将输入的文本转换成二维码图案，使用zxing的QRCodeWriter类，调用QRCodeWriter.encode方法把输入的文本转化成位图矩阵，然后将其通过二异赋值得到的数组保存在像素数组中，最后创建一个新的位图，将这组像素赋给这个位图就生成了一张二维码图，具体代码如下：



**实现细节**

* **服务端**

使用了LAMP架构，数据基本储存在服务端数据库中。数据库表有：

records： 记录学生的签到记录

students： 记录学生的个人信息

teacher： 记录教师的个人信息

courses： 记录课程的相关信息

详细数据库设计请见数据库设计文档。

* **客户端**

用户有两种类型，分别是老师和学生，老师可以查看自己学生的考勤信息并且生成二维码，教师生成的二维码中存储的数据的格式是：

*课程名称+空格+上课时间y/m/d+随机的字符串*

生成二维码后可以将二维码上传到服务器并生成一个URL链接，上课的时候可以用浏览器打开该链接供学生签到。学生扫描二维码成功后，该学生的签到记录便被添加至后台数据库，形成一次签到记录。

这其中需要考虑的是加密，教师在生成二维码的时候可以输入一个随机的字符串，该字符串可以作为密钥来加密二维码的信息，这样学生便无法使用其它二维码扫描工具得知二维码存储的信息了。密钥加密完二维码数据后会被临时存储在后台数据库中，学生客户端在扫描二维码后可以通过后台数据库拿到密钥解密二维码信息，从而得到课程信息，并实现签到。

另外，客户端也有一个数据库，存储的是学生成功签到的记录，避免每次查询记录都从服务端获取。

**应用与改进**

QR-Checkin是一款安卓平台上的二维码课程签到应用。教师可生成当前时间、当前课程的二维码供学生进行扫描签到，同时生成签到记录供教师和学生进行考勤参考。该应用可普及到课程签到、会议签到、工作考勤等场景，方便各场景下的签到和考勤记录管理。

因时间紧促，该应用目前只有简单的生成、扫描二维码和记录查询功能，理想情况下还需要对应用添加“防代签”机制，并且搭配web端进行二维码展示，考勤记录查询、修改、导出等等基础功能。计划中，应用还将加入课程表、请假、通知系统功能，将课程相关的日常场景包含在内，提供便捷的数字课程管理服务。