**IssBase框架说明文档**

**最新版本**：1.4

**日期：**2013-10-14

|  |  |  |  |
| --- | --- | --- | --- |
| **时间** | **版本** | **更新说明** | **作者** |
| 2013-04-17 | V1.1 | 1.将ItotemBase转为IssBase，完成初始化导入  2.加入基于HttpURLConnection的网络请求封装  3.删除com.itotem.network包  4.替换图片缩放控件为PhotoView  5.更新ImageLoader包 | 廖文新 |
| 2013-06-07 | V1.2 | 1.补充完善DimensionPixelUtil工具  2.去掉styles\_activity.xml里Activity样式的透明属性设置  3.添加ImageUtils、VersionUtils工具类  4.加入AbsDialog，修改Dialog样式  5.加入数据库模块  6.更新日历控件  7.更新android-support-v4包，关联源码  8.更新PhotoView控件  9.更新下拉刷新控件  10.更新瀑布流控件  11.完善IssBaseDemo以及相关文档 | 廖文新 |
| 2013-09-06 | V1.3 | 1.优化网络模块  2.加入缩放、旋转、拖拽的手势封装  3.修改IssActivity的实现方式  4.更新ImageLoader包  5.加入动画包  6．加入多方向的ViewPager：OrientationViewPager | 廖文新 |
| 2013-10-14 | V1.4 | 1. 加入Jenkins相关脚本文件 2. 修改WheelView控件，支持滚动变化 | 廖文新 |

**目录**

[一、概述 1](#_Toc369533574)

[1.1 工程包结构预览 1](#_Toc369533575)

[1.2 说明 1](#_Toc369533576)

[二、包结构说明 2](#_Toc369533577)

[2.1 com.iss.app 2](#_Toc369533578)

[2.1.1 IssActivity 2](#_Toc369533579)

[2.1.2 IssAppContext 2](#_Toc369533580)

[2.1.3 AbsDialog 2](#_Toc369533581)

[2.2 com.iss.bean 3](#_Toc369533582)

[2.2.1 BaseBean 3](#_Toc369533583)

[2.3 com.iss.db 4](#_Toc369533584)

[2.3.1 表结构映射 4](#_Toc369533585)

[2.3.2 配置 4](#_Toc369533586)

[2.3.3 初始话数据库 4](#_Toc369533587)

[2.3.4 访问数据库 4](#_Toc369533588)

[2.3.5 注意 5](#_Toc369533589)

[2.4 com.iss.httpclient 6](#_Toc369533590)

[2.4.1 初始化NormalHttpClient对象 6](#_Toc369533591)

[2.4.2 配置请求参数 7](#_Toc369533592)

[2.4.3 请求头的配置 7](#_Toc369533593)

[2.4.4 访问网络 7](#_Toc369533594)

[2.5 com.iss.imageloader 8](#_Toc369533595)

[2.5.1 权限添加 8](#_Toc369533596)

[2.5.2 在Application中初始化配置 8](#_Toc369533597)

[2.5.3 设置图片显示参数 8](#_Toc369533598)

[2.5.4 使用ImageLoader获取并显示图片 9](#_Toc369533599)

[2.5.5 Imageloader可以加载的图片连接 9](#_Toc369533600)

[2.5.6 ImageLoader中的工具类常用发法 9](#_Toc369533601)

[2.6 com.iss.loghandler 11](#_Toc369533602)

[2.6.1 在Manifest注册用于显示抓取的log的Activity 11](#_Toc369533603)

[2.6.2 在需要配置loghandler的Activity的OnCreate中注册错误监听，并配置接收错误日志的邮箱 11](#_Toc369533604)

[2.7 com.iss.service 12](#_Toc369533605)

[2.8 com.iss.utils 13](#_Toc369533606)

[2.8.1 像素单位转换工具类：DimensionPixelUtil 13](#_Toc369533607)

[2.8.2 图片处理工具类：ImageUtils 13](#_Toc369533608)

[2.8.3 Md5工具类：Md5Util 13](#_Toc369533609)

[2.8.4 手机信息处理工具类：PhoneInfoUtil 13](#_Toc369533610)

[2.8.5 字符串工具类：StringUtil 14](#_Toc369533611)

[2.8.6 时间格式处理工具类：TimeUtil 14](#_Toc369533612)

[2.8.7 版本工具类：VersionUtils 14](#_Toc369533613)

[2.9 com.view.calendar 15](#_Toc369533614)

[2.9.1 CalendarCellView 15](#_Toc369533615)

[2.9.2 CalendarGridView 15](#_Toc369533616)

[2.9.3 CalendarRowView 16](#_Toc369533617)

[2.9.4 MonthView 16](#_Toc369533618)

[2.9.5 CalendarPickerView 17](#_Toc369533619)

[2.9.6 MonthCellDescriptor & MonthDescriptor 17](#_Toc369533620)

[2.9.7 使用实例 17](#_Toc369533621)

[2.10 com.iss.view.common 18](#_Toc369533622)

[2.10.1 CustomScrollView 18](#_Toc369533623)

[2.10.2 MarqueeTextView 18](#_Toc369533624)

[2.10.3 ToastAlone 18](#_Toc369533625)

[2.11 com.iss.view.leftgallery 19](#_Toc369533626)

[2.12 com.iss.view.photoview 20](#_Toc369533627)

[2.13 com.iss.view.pullrefresh 21](#_Toc369533628)

[2.14 com.iss.view.waterfall 22](#_Toc369533629)

[2.15 com.iss.view.wheel 26](#_Toc369533630)

[三、JenKins打包使用说明 27](#_Toc369533631)

[3.1 debug打包 27](#_Toc369533632)

[3.1.1 apk\_create.sh脚本说明 27](#_Toc369533633)

[3.1.2 apk\_install\_start.sh 脚本说明 27](#_Toc369533634)

[3.1.3 apk\_stop\_uninstall.sh 脚本说明 27](#_Toc369533635)

[3.1.4 build.xml 脚本说明 27](#_Toc369533636)

[3.1.5 custom\_rules.xml文件修改 28](#_Toc369533637)

[3.2 normal打包 28](#_Toc369533638)

[3.2.1 apk\_create.bat脚本说明 28](#_Toc369533639)

[3.2.2 build.xml脚本说明 29](#_Toc369533640)

[3.2.3 ids.cfg文件 29](#_Toc369533641)

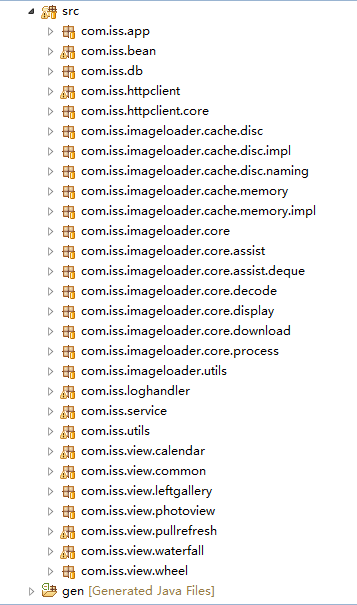
[3.3 Jenkins服务器配置 29](#_Toc369533642)

[3.3.1 在Jenkins上创建新任务 29](#_Toc369533643)

[3.3.2 新建任务的配置 30](#_Toc369533644)

# 一、概述

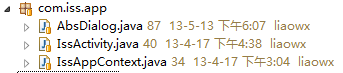
## 1.1 工程包结构预览



## 1.2 说明

# 二、包结构说明

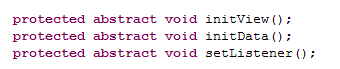
## 2.1 com.iss.app



### 2.1.1 IssActivity

IssActivity是一个抽象类，继承了Activity，复写了几个使用几率较高的方法，方便开发使用。

继承IssActivity的类必须实现下面几个方法：



initView(),通常在该方法里处理各个控件的findView等

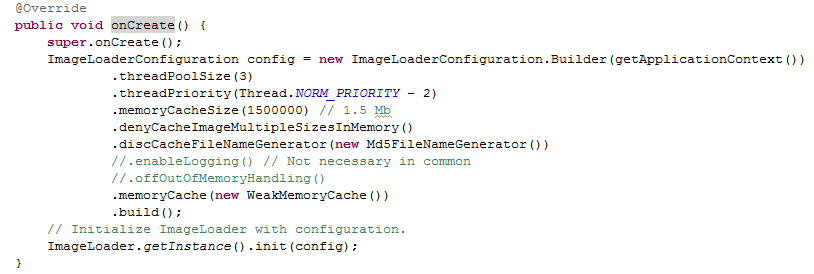
initData(),通常在该方法里初始化Activity涉及的数据

setListener(),给各个控件添加监听

后续此类需完善功能，加上各个控件的注解方式处理，以简化代码

### 2.1.2 IssAppContext

集成了Application，主要在onCreate方法里对ImageLoader做初始化处理，需要用到ImageLoader的工程要集成该类



### 2.1.3 AbsDialog

AbsDialog是一个抽象类，继承自Dialog，方便开发中使用，跟IssActivity一样，继承AbsDialog需实现如下方法：



使用实例参考：Sincere—com.iss.sincere.dialog包中的各种自定义Dialog

## 2.2 com.iss.bean



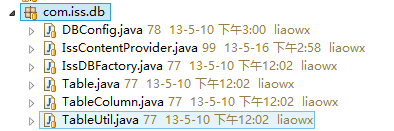
### 2.2.1 BaseBean

BaseBean是一个抽象类，实现了Serializable接口，项目里边定义的实体类需继承该类，需实现如下方法。



后续该类将和数据库的架构相结合

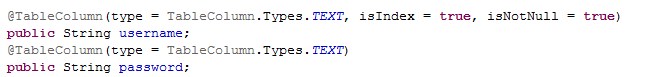
## 2.3 com.iss.db



该包采用orm方式，对系统的sqlite数据库使用ContentProvider的方式进行了简单封装。使用步骤如下（以Sincere工程为例，需求：LoginUser类为登录用户的实体类，登录的用户需记录到数据库，我们需要建立一个LoginUser表，来存储登录的用户，并且为username建立索引，以提高检索速度。）

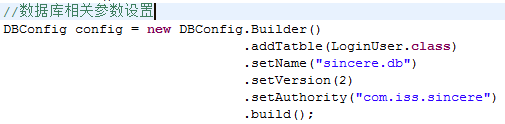
### 2.3.1 表结构映射

参考：Sincere--com.iss.sincere.bean—LoginUser



使用@TableColumn注解可以轻松为数据库映射表的字段、字段类型、索引关系等

### 2.3.2 配置



通过DBConfig类来配置数据库文件名，版本号，以及需要关联生成表的类等

### 2.3.3 初始话数据库



调用IssDBFactory.init()方法，参入数据库的配置参数，来初始化数据库。初始化数据库的过程需要继承IssContentProvider，在init()方法中实现初始化逻辑

### 2.3.4 访问数据库

通过IssContentProvider的buildUri方法获取相应的表访问的Uri



调用ContentResolver的相关方法来访问数据库：



参考—com.iss.sincere.db.DBUtils，建议根据需求封装一个统一的Utils类来访问数据库

### 2.3.5 注意

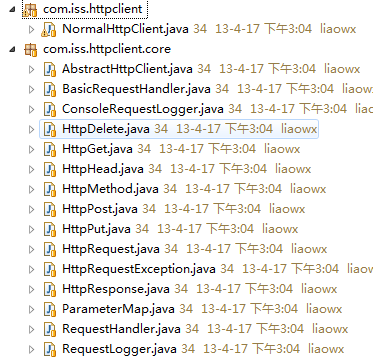
由于数据库架构用到了注解和反射，所以如果使用默认的混淆打签名包时，会出问题。需修改混淆配置文件proguard-project.txt，将用到注解和反射的包屏蔽混淆。如下：

-dontwarn com.iss.db.\*\*

-keep class com.iss.db.\*\* { \*;}

-keepattributes \*Annotation\*

## 2.4 com.iss.httpclient



该包是基于HttpURLConnection封装的网络请求包，是在一个开源项目上做的修改。Apache HttpClient 和 HttpURLConnection的比较，如下是google官方博客的原话，推荐项目中采用HttpURLConnection。测试发现HttpURLConnection在性能和速度上均优于Apache HttpClient。

**Apache HTTP Client**

[DefaultHttpClient](http://developer.android.com/reference/org/apache/http/impl/client/DefaultHttpClient.html) and its sibling [AndroidHttpClient](http://developer.android.com/reference/android/net/http/AndroidHttpClient.html) are extensible HTTP clients suitable for web browsers. They have large and flexible APIs. Their implementation is stable and they have few bugs.But the large size of this API makes it difficult for us to improve it without breaking compatibility. The Android team is not actively working on Apache HTTP Client.

**HttpURLConnection**

[HttpURLConnection](http://developer.android.com/reference/java/net/HttpURLConnection.html) is a general-purpose, lightweight HTTP client suitable for most applications. This class has humble beginnings, but its focused API has made it easy for us to improve steadily.

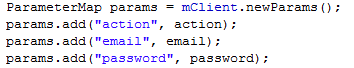
使用说明：

### 2.4.1 初始化NormalHttpClient对象



调用NormalHttpClient的构造方法，传入主url来初始化一个NormalHttpClient

### 2.4.2 配置请求参数



通过NormalHttpClient的newParams()方法获取一个集合类对象，然后像集合类中添加相关请求参数

### 2.4.3 请求头的配置



调用NormalHttpClient的addHeader()方法，可以给每次请求的请求头设置一些参数，如Cookie等

### 2.4.4 访问网络



调用NormalHttpClient的post()/get()/delete()等方法，参入配置的访问参数，和对应接口具体访问路径，来访问网络，返回一个HttpResponse对象，通过该对象获取接口放回的数据，注意HttpResponse可能为null，需判断处理

## 2.5 com.iss.imageloader

该包对网络图片获取及显示进行了相关封装，整合的开源项目Android-Universal-Image-Loader。

### 2.5.1 权限添加

网络访问权限：

<uses-permission android:name=”android.permission.INTERNET”/>

存储卡读写权限：

<uses-permission android:name=”android.permission.WRITE\_EXTERANL\_STORAGE”/>

### 2.5.2 在Application中初始化配置

ImageLoaderConfiguration config = new ImageLoaderConfiguration.Builder(context)

.threadPriority(Thread.NORM\_PRIORITY - 2)

.denyCacheImageMultipleSizesInMemory()

.discCacheFileNameGenerator(new Md5FileNameGenerator())

.tasksProcessingOrder(QueueProcessingType.LIFO)

.enableLogging()

.memoryCache(new LruMemoryCache(2 \* 1024 \* 1024))

.memoryCacheSize(2 \* 1024 \* 1024)

.discCacheSize(50 \* 1024 \* 1024)

.discCacheFileCount(100)

.build();

ImageLoader.getInstance().init(config);

### 2.5.3 设置图片显示参数

DisplayImageOptions options = new DisplayImageOptions.Builder()

.showStubImage(R.drawable.ic\_stub)

.showImageForEmptyUri(R.drawable.ic\_empty)

.showImageOnFail(R.drawable.ic\_error)

.resetViewBeforeLoading()

.delayBeforeLoading(1000)

.cacheInMemory()

.cacheOnDisc()

.preProcessor(...)

.postProcessor(...)

.extraForDownloader(...)

.imageScaleType(ImageScaleType.IN\_SAMPLE\_POWER\_OF\_2) // de

.bitmapConfig(Bitmap.Config.ARGB\_8888) // default

.decodingOptions(...)

.displayer(new SimpleBitmapDisplayer()) // default

.handler(new Handler()) // default

.build();

### 2.5.4 使用ImageLoader获取并显示图片

imageLoader.displayImage(imageUri, imageView, displayOptions, new ImageLoadingListener() {

@Override

public void onLoadingStarted(String imageUri, View view) {

...

}

@Override

public void onLoadingFailed(String imageUri, View view, FailReason failReason) {

...

}

@Override

public void onLoadingComplete(String imageUri, View view, Bitmap loadedImage) {

...

}

@Override

public void onLoadingCancelled(String imageUri, View view) {

...

}

});

### 2.5.5 Imageloader可以加载的图片连接

// from Web

String imageUri = "http://site.com/image.png";

// from SD card

String imageUri = "file:///mnt/sdcard/image.png";

// from content provider

String imageUri = "content://media/external/audio/albumart/13";

// from assets

String imageUri = "assets://image.png";

// from drawables (only images, non-9patch)

String imageUri = "drawable://" + R.drawable.image;

### 2.5.6 ImageLoader中的工具类常用发法

ImageLoader |

| - getMemoryCache()

| - clearMemoryCache()

| - getDiscCache()

| - clearDiscCache()

| - denyNetworkDownloads(boolean)

| - handleSlowNetwork(boolean)

| - pause()

| - resume()

| - stop()

| - destroy()

| - getLoadingUriForView(ImageView)

| - cancelDisplayTask(ImageView)

MemoryCacheUtil |

| - findCachedBitmapsForImageUri(...)

| - findCacheKeysForImageUri(...)

| - removeFromCache(...)

DiscCacheUtil |

| - findInCache(...)

| - removeFromCache(...)

StorageUtils |

| - getCacheDirectory(Context)

| - getIndividualCacheDirectory(Context)

| - getOwnCacheDirectory(Context, String)

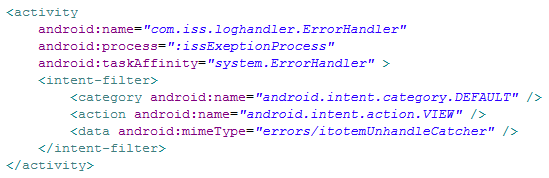
PauseOnScrollListener

## 2.6 com.iss.loghandler

该包多用于debug测试阶段，在程序崩溃时会触发，将错误log及设备信息抓取显示，并可以配置发送给开发者。

使用步骤:

### 2.6.1 在Manifest注册用于显示抓取的log的Activity

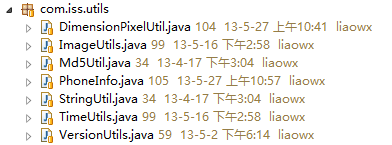


### 2.6.2 在需要配置loghandler的Activity的OnCreate中注册错误监听，并配置接收错误日志的邮箱



## 2.7 com.iss.service

## 2.8 com.iss.utils



### 2.8.1 像素单位转换工具类：DimensionPixelUtil

DimensionPixelUtil

|-- getDimensionPixelSize(int unit, float value, Context context)

|-- dip2px(Context context, float value)

|-- px2dip(Context context, float value)

|-- sp2px(Context context, float value)

|-- px2sp(Context context, float value)

### 2.8.2 图片处理工具类：ImageUtils

ImageUtils

|-- readDrawable(Context context, int resId)

|-- readDrawable(Context context, int resId, Config bitmapConfig)

|-- readDrawable(Resources res, int resId, Config bitmapConfig)

|-- readDrawable(Resources res, File file)

|-- readBitmap(Context context, int resId)

### 2.8.3 Md5工具类：Md5Util

Md5Util

|-- getMD5Str(String str)

|-- byte2hex(byte[] b)

### 2.8.4 手机信息处理工具类：PhoneInfoUtil

PhoneInfoUtil

|-- printCallLog(Context context)

|-- hideSoftInputMode(Context context, View windowToken)

|-- getMetaData(Context context, String key)

|-- getPhoneNumber(Context context)

### 2.8.5 字符串工具类：StringUtil

StringUtil

|-- inputToString(InputString inputStream, String encoding)

### 2.8.6 时间格式处理工具类：TimeUtil

TimeUtil

|-- getFormatDate(String format)

|-- getFormatDate1()

|-- getFormatDate2()

|-- getFormatDate3(String date)

|-- getFormatDate4(String date)

|-- getFormatDate5(String date)

|-- getFormatTime1()

|-- getFormatTime2()

|-- getFormatTime3(String time)

|-- getFormatTime4(String time)

|-- getFormatTime(long time, String format)

|-- getFormatLeaveDay (String day, int leaveDay)

|-- getFormatBeforeDay (String day)

|-- getFormatNextDay (String day)

|-- getWeekDay(String inputDate)

|-- isInsideTime(String timeslot, String time)

|-- compareTime(String time1, String time2)

|-- compareDate(String date1, String date2)

### 2.8.7 版本工具类：VersionUtils

VersionUtils

|-- hasFroyo()

|-- hasGingerbread()

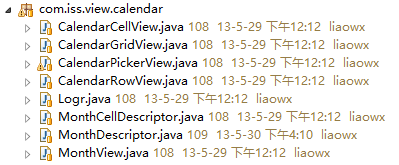
|-- hasHoneycomb()

|-- hasHoneycombMR1()

|-- hasJellyBean()

|-- getAppVersionName(Context context)

## 2.9 com.view.calendar



日历控件的可定制程度非常高，所以不太好封装，在这里介绍下每个类，以便如果样式不合适的项目，可以在此基础上做修改和封装。



### 2.9.1 CalendarCellView

该类继承了TextView,日历中每一个日期的显示都是一个CalendarCellView



### 2.9.2 CalendarGridView

该类继承自ViewGroup，为整个日历区域，其主要功能是读取布局文件的子view（CalendarRowView），做日历样式的排版，和绘制边框线



### 2.9.3 CalendarRowView

继承自ViewGroup，用于封装一排view，并根据自身的宽度计算每一个子view的宽度，如果子View是CalendarCellView，则高也设置跟宽一样。并将点击事件传递给子view；



### 2.9.4 MonthView

如下图，MonthView由一个TextView和一个CalendarGridView组成，是一个完整的月份视图。



### 2.9.5 CalendarPickerView

该类继承了ListView，子View的都是MonthView，通过ListView的Adapter来控制子view的刷新加载

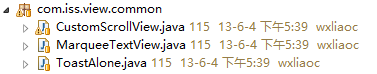
### 2.9.6 MonthCellDescriptor & MonthDescriptor

这两个是辅助类，MonthDescriptor封装了用与处理MonthView的信息，MonthCellDescriptor封装了用于处理CalendarCellView的信息以及日期的选取模式等等

### 2.9.7 使用实例

参考IssBaseDemo – com.iss.example.view.calendar.CalendarActivity

## 2.10 com.iss.view.common



### 2.10.1 CustomScrollView

该控件继承ScrollView，普通的ScrollView里边嵌套有Gallery，ViewPager之类的控件时，会有手势上的冲突，使用该控件可以解决这个问题。

### 2.10.2 MarqueeTextView

该控件继承TextView，显示单行，如果文字长度超出显示范围，自动以跑马灯形式展示

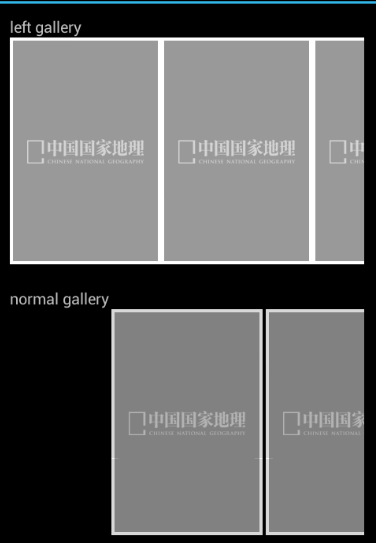
### 2.10.3 ToastAlone

对Toast进行了简单封装，避免使用系统Toast时，多次点击，会弹出多个Toast的问题

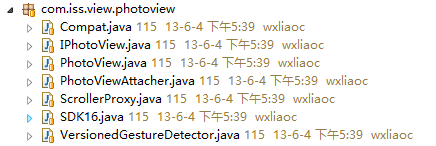
## 2.11 com.iss.view.leftgallery



如下图，leftgallery包实现了一个默认居左显示的Gallery（系统Gallery默认居中显示），可根据项目的需求使用该控件，使用方式与系统Gallery一致。



## 2.12 com.iss.view.photoview



一个图片缩放控件，支持双击缩放和手势缩放，可实现类似于系统相册的图片浏览

该控件的核心类为PhotoViewAttacher,同过将一个ImageView跟PhotoViewAttacher绑定，可以实现相关的缩放功能。

PhotoView类其实就是继承了ImageView，然后绑定了一个PhotoViewAttacher

PhotoViewAttacher常用方法：

PhotoViewAttacher

|--getMinScale()

|--getMidScale()

|--getMaxScale()

|--getScale()

|--getScaleType()

|--setMinScale(float minScale)

|--setMidScale(float midScale)

|--setMaxScale(float maxScale)

|--setOnLongClickListener(OnLongClickListener listener)

|--setOnMatrixChangeListener(OnMatrixChangedListener listener)

|--setOnPhotoTapListener(OnPhotoTapListener listener)

|--setOnViewTapListener(OnViewTapListener listener)

|--setScaleType(ScaleType type)

|--setZoomable(booean zoomable)

|--zoomTo(float scale, float focalX, float focalY)

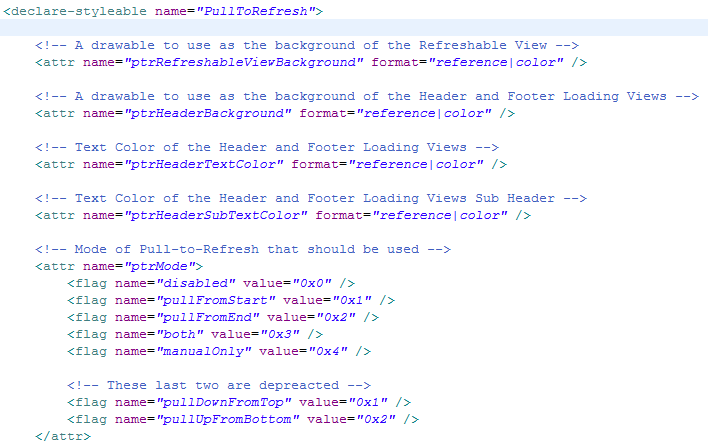
使用实例参考：

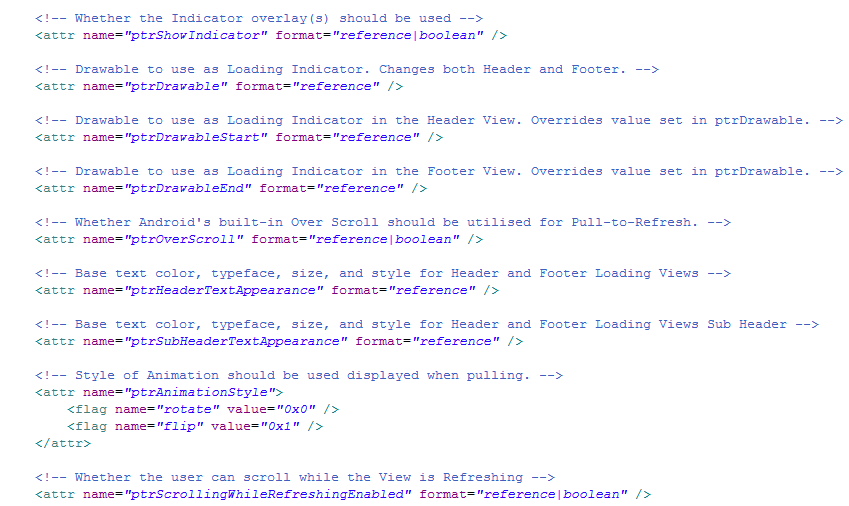
IssBaseDemo—com.iss.example.view.photoview—SimpleSampleActivity/ViewPagerActivity

## 2.13 com.iss.view.pullrefresh

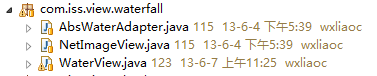
下拉刷新控件，这个比较常用，注意该控件有很多可配置属性，通过配置能满足项目的各种刷新需求，详见attrs.xml

使用示例：参考IssBaseDemo—com.iss.example.view.pullrefreshview中的例子





## 2.14 com.iss.view.waterfall



这是一个轻量级的瀑布流控件，采用了一个比较合理的图片回收管理机制，能够有效的避免内存溢出问题。

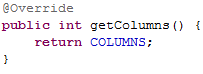
使用说明：

1.在xml布局文件中使用瀑布流控件WaterView或者PullToRefreshWaterView(带刷新功能)



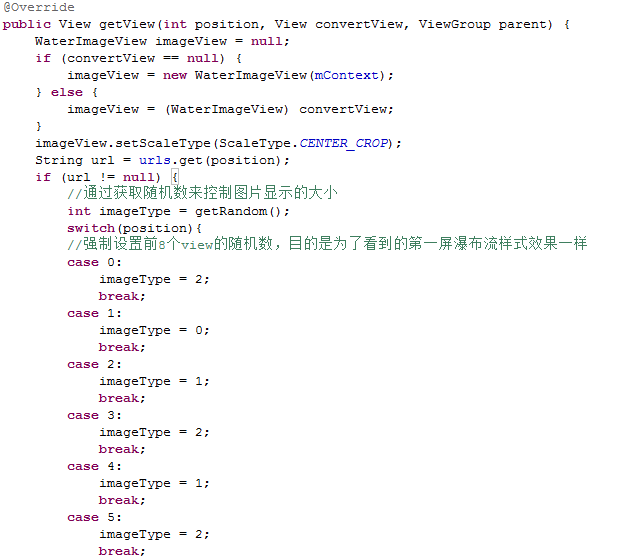
2.实现用于WaterView的Adapter，需继承AbsWaterAdapter，

通过实现getColumns()来设置瀑布流的列数，



最重要的一点，getView()方法里边，每个View在瀑布流控件里的实际大小是根据列数得出宽度之后，将view的宽高做等比缩放处理的

（Demo里边的例子，为了实现每次进入看瀑布流界面时的样式一样，采用了固定前几个控件大小的变法）

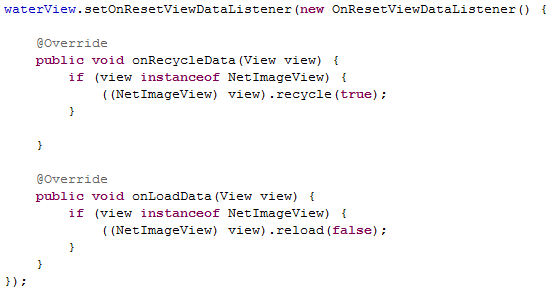




3.设置WaterView的Adapter；



4.使用瀑布流控件时需要注意，每一个子View的数据的加载和回收都是通过回调监听来实现的，如下示例，需设置WaterView的OnResetViewDataListener监听才能实现数据的加载和回收



## 2.15 com.iss.view.wheel

这是一个仿iphone的picker控件，具体效果如下图，使用示例请参考IssBaseDemo—com.iss.example.view.wheelview中的例子



# 三、JenKins打包使用说明

Jenkins服务器负责的工作

1．生成自动化测试包+执行自动化测试脚本

2．生成安装包，提交Installer服务器，供客户及测试人员下载使用

准备工作：

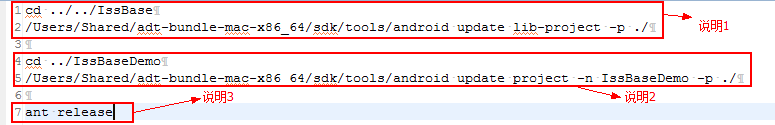
1．在工程新建jenkins-debug、jenkins-normal、test三个文件夹，其中jenkins-debug文件夹用于放置打自动化测试包的相关脚本，jenkins-normal文件夹用于放置正常打包脚本，test文件夹用于放置测试脚本，这三个文件夹都需要提交svn。

2．在JenKins上新建项目任务，每个项目需要配置两个任务，一个用于生成自动化测试包并执行测试脚本，一个用于生成普通安装包并提交到installer。

## 3.1 debug打包

### 3.1.1 apk\_create.sh脚本说明

该脚本用于在指定工程的根目录下生成build.xml和local.properties文件，生成的build.xml文件引用了sdk中的打包脚本，当我们在工程根目录下运行ant命令时，默认会调用这个生成的build.xml文件自动打包了



***说明1：***这两行命令用来更新library工程，如果没有library，删除这两行，如果有多个，按此格式添加

cd ../../IssBase 表示将当前目录切换到library工程，

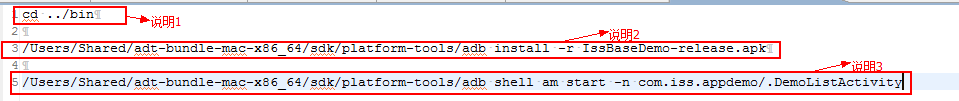
更新library的命令：android update lib-project -p xxx，“-p xxx”表示工程位置，“./” 表示当前目录

***说明2：***这两行命令用来更新主工程，

更新主工程命令：android update project -n xxx -p xxx，“-n xxx”表示工程名字，这里必须按命名规则指定，生成的apk是根据这个来命名的，“xxx-release.apk”

***说明3：***这个命名为调用ant，执行当前目录生成的build.xml文件

### 3.1.2 apk\_install\_start.sh 脚本说明



***说明1：***定位到bin目录，因为生成的apk在这个目录下

***说明2：***调用adb命令安装apk，这里需要需改apk文件名

***说明3：***调用adb命令启动程序，请修改对应的启动类

### 3.1.3 apk\_stop\_uninstall.sh 脚本说明



***说明：***调用adb命令关闭、卸载程序，请修改对应包名

### 3.1.4 build.xml 脚本说明

该脚本一共执行8个target（详情请查看原文件）

target1：Clean工程

target2：复制monekytalk的jar包，和custom\_rules.xml文件

target3:生成apk（分windows、mac平台）

target4:安装启动apk（分windows、mac平台）

target5:执行测试脚本

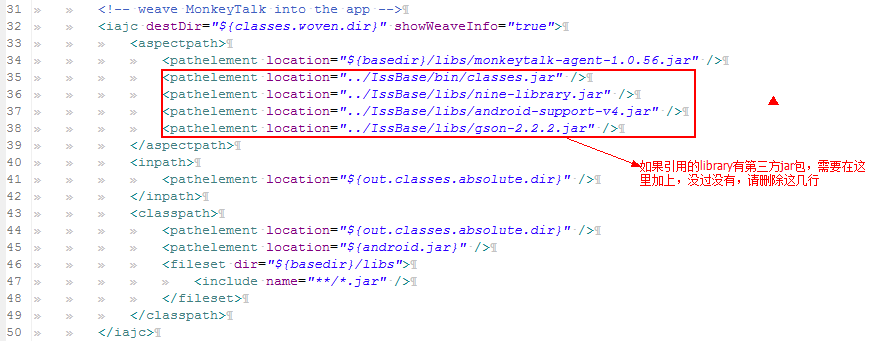
target6:关闭卸载apk（分windows、mac平台）

target7:删除target2复制的文件

注意需要修改的地方：该脚本target5中测试脚本需修改成相对应脚本文件，如下



### 3.1.5 custom\_rules.xml文件修改



## 3.2 normal打包

### 3.2.1 apk\_create.bat脚本说明

这个跟debug打包一样，请参考上面的说明

### 3.2.2 build.xml脚本说明

该脚本通过读取ids.cfg文件循环执行打包操作，并将生成的apk文件复制一份到installer服务器所使用文件夹，需要修改的地方

a.修改<project>标签name属性值，按邮件提到的命名规则来起名

b.如果多渠道打包，非友盟的，需修改渠道名，替换“UMENG\_CHANNEL”

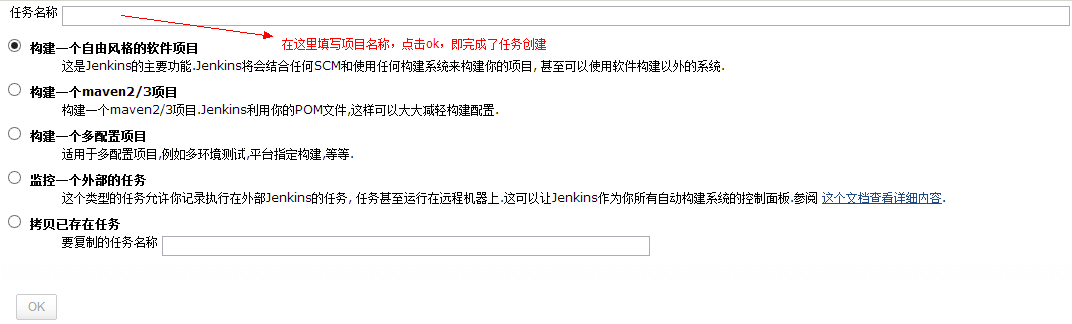
### 3.2.3 ids.cfg文件

添加渠道号，如果无渠道，至少保留一个渠道号

## 3.3 Jenkins服务器配置

### 3.3.1 在Jenkins上创建新任务





在Jenkins上新建任务很简单，如图，点击“新Job”，填写项目名称，选择“构建一个自由风格的软件项目”，ok确定，就可以了。

需要注意，任务命名规则：项目名称\_平台\_构建类型

项目名称：按项目拼音，首字母大写

平台：Android/AndroidPad

构建类型：Debug/Normal

Eg：宝宝家手机版---BaoBaoJia\_Android\_Debug/BaobaoJia\_Android\_Normal

新浪汽车 ----SinaQiChe\_Android\_Debug/SinaQiChe\_Android\_Normal

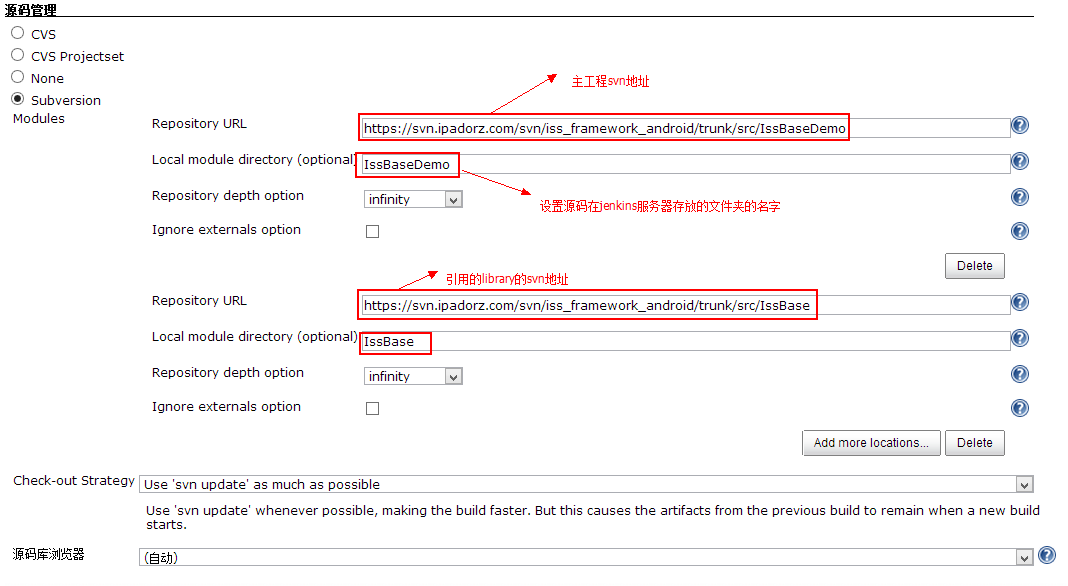
### 3.3.2 新建任务的配置

这里注意下，新任务的配置，Debug任务和Normal任务有一些区别，应该先配置Normal任务，再配置Debug任务

1. **Normal任务的配置(以IssBaseDemo\_Android\_Normal的配置为例)：**

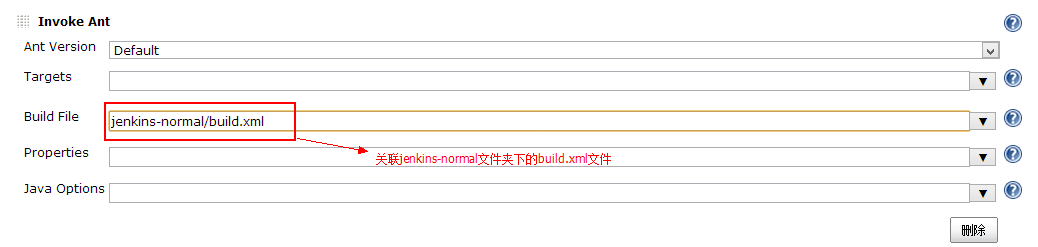
a-1：源码管理（用于关联svn相关工程）

选择Subversion，我们这里有两个svn工程，选择Add more locations，多加一个工程



a-2：构建（关联Ant脚本）

选择Invoke Ant，点“高级”，配置Ant脚本路径



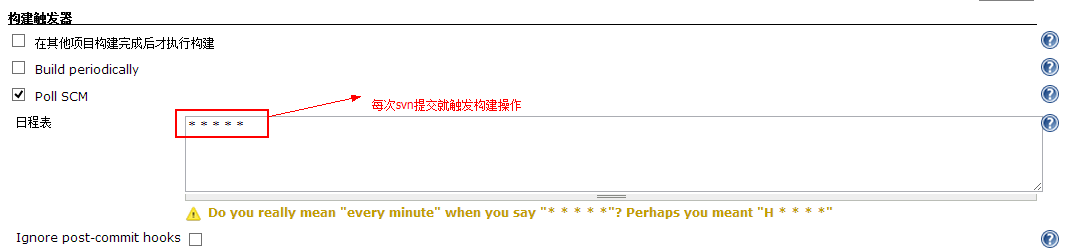
a-3：配置完成，点击保存

1. **Debug任务的配置**

b-1：源码关联（同a-1）

b-2：构建触发器（配置jenkins构建的规则，如每天固定时间点触发构建，或者每次svn提交代码触发构建等）

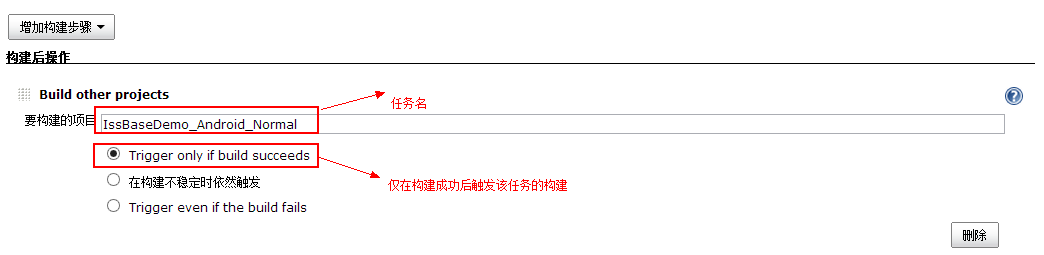
选择Poll SCM，填写触发规则，“\* \* \* \* \* \*”表示每次svn有提交就触发构建，具体规则点查看



b-3：构建，同a-2（注意文件位置修改成“jenkins-debug/build.xml”）

b-4：构建后的操作

选择Build other projects，我们要在Debug任务构建完成后，自动构建Normal任务



b-5：配置完成，保存

注意，如果需要在windows系统调试ant脚本，请把jenkins-debug/windows目录里的bat脚本放到jenkins-debug目录，并且电脑需要配置的环境变量：

1.JAVA\_HOME --------------------------------------------JDK所在路径

2.ANDROID\_HOME--------------------------------------------Android SDK所在路径

3.ANT\_HOME --------------------------------------------Ant所在路径

4.AspectJ\_HOME--------------------------------------------Aspect所在路径