

IssBase 框架说明文档

最新版本：1.3

日期：2013-09-06

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	廖文新

		11. 完善 IssBaseDemo 以及相关文档	
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

目录

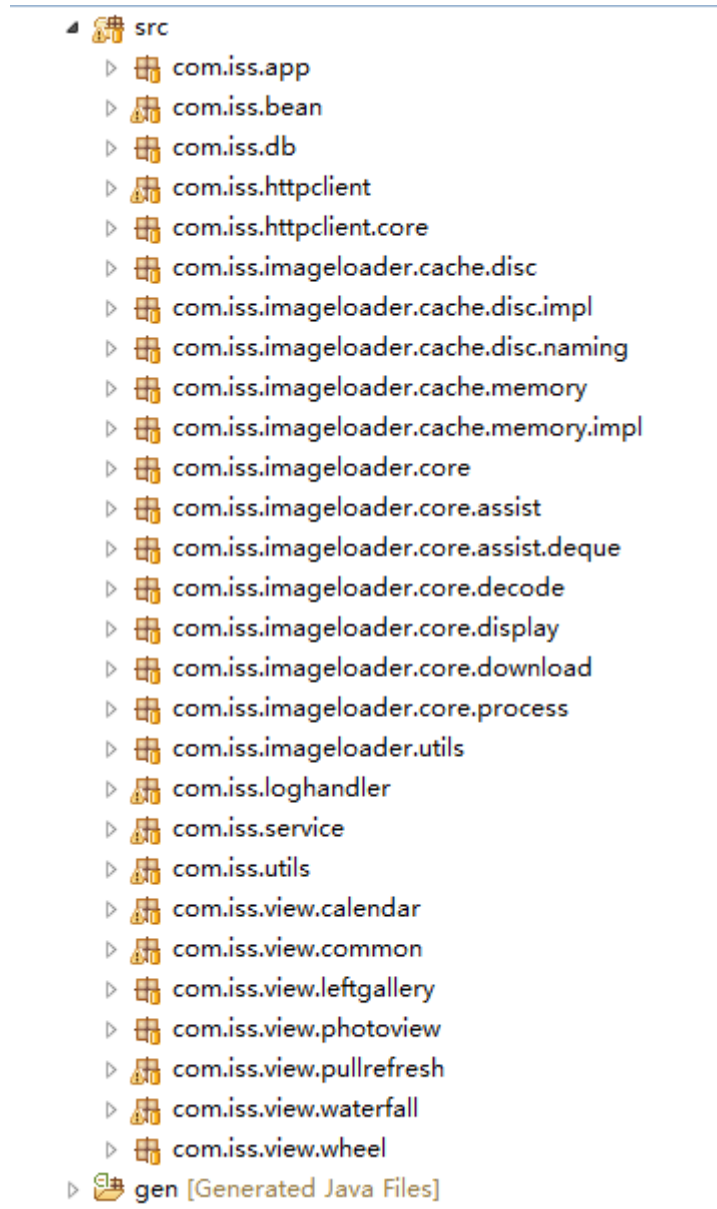
一、概述	1
1.1 工程包结构预览	1
1.2 说明	1
二、包结构说明	2
2.1 com.iss.app	2
2.1.1 IssActivity	2
2.1.2 IssAppContext	2
2.1.3 AbsDialog	2
2.2 com.iss.bean	3
2.2.1 BaseBean	3
2.3 com.iss.db	4
2.3.1 表结构映射	4
2.3.2 配置	4
2.3.3 初始话数据库	4
2.3.4 访问数据库	4
2.3.5 注意	5
2.4 com.iss.httpClient	6
2.4.1 初始化 NormalHttpClient 对象	6
2.4.2 配置请求参数	7
2.4.3 请求头的配置	7
2.4.4 访问网络	7
2.5 com.iss.imageloader	8
2.5.1 权限添加	8

2.5.2	在 Application 中初始化配置.....	8
2.5.3	设置图片显示参数	8
2.5.4	使用 ImageLoader 获取并显示图片	9
2.5.5	Imageloader 可以加载的图片连接.....	9
2.5.6	ImageLoader 中的工具类常用发法	9
2.6	com.iss.loghandler.....	11
2.6.1	在 Manifest 注册用于显示抓取的 log 的 Activity.....	11
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
2.7	com.iss.service.....	12
2.8	com.iss.utils	13
2.8.1	像素单位转换工具类: DimensionPixelUtil	13
2.8.2	图片处理工具类: ImageUtils.....	13
2.8.3	Md5 工具类: Md5Util	13
2.8.4	手机信息处理工具类: PhoneInfoUtil.....	13
2.8.5	字符串工具类: StringUtil	14
2.8.6	时间格式处理工具类: TimeUtil	14
2.8.7	版本工具类: VersionUtils	14
2.9	com.view.calendar	15
2.9.1	CalendarCellView.....	15
2.9.2	CalendarGridView.....	15
2.9.3	CalendarRowView	16
2.9.4	MonthView	16
2.9.5	CalendarPickerView.....	17

2.9.6	MonthCellDescriptor & MonthDescriptor.....	17
2.9.7	使用实例	17
2.10	com.iss.view.common	18
2.10.1	CustomScrollView	18
2.10.2	MarqueeTextView	18
2.10.3	ToastAlone.....	18
2.11	com.iss.view.leftgallery	19
2.12	com.iss.view.photoview	20
2.13	com.iss.view.pullrefresh	21
2.14	com.iss.view.waterfall	22
2.15	com.iss.view.wheel.....	26

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

```
└─ com.iss.app
   ├── AbsDialog.java 87 13-5-13 下午6:07 liaowx
   ├── IssActivity.java 40 13-4-17 下午4:38 liaowx
   └── IssAppContext.java 34 13-4-17 下午3:04 liaowx
```

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：

```
protected abstract void initView();
protected abstract void initData();
protected abstract void setListener();
```

initView(), 通常在该方法里处理各个控件的 findViewById 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

```
@Override
public void onCreate() {
    super.onCreate();
    ImageLoaderConfiguration config = new ImageLoaderConfiguration.Builder(getApplicationContext())
        .threadPoolSize(3)
        .threadPriority(Thread.NORM_PRIORITY - 2)
        .memoryCacheSize(1500000) // 1.5 Mb
        .denyCacheImageMultipleSizesInMemory()
        .discCacheFileNameGenerator(new Md5FileNameGenerator())
        //.enableLogging() // Not necessary in common
        //.offOutOfMemoryHandling()
        .memoryCache(new WeakMemoryCache())
        .build();
    // Initialize ImageLoader with configuration.
    ImageLoader.getInstance().init(config);
}
```

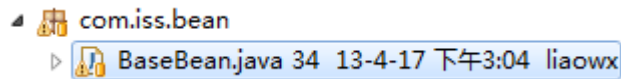
2.1.3 AbsDialog

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：

```
protected abstract void initView();
protected abstract void initData();
protected abstract void setListener();
```

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。

```
/**
 * 将json对象转化为Bean实例
 * @param jsonObj
 * @return
 */
public abstract T parseJSON(JSONObject jsonObj);

/**
 * 将Bean实例转化为json对象
 * @return
 */
public abstract JSONObject toJSON();

/**
 * 将数据库的cursor转化为Bean实例（如果对象涉及在数据库存取，需实现此方法）
 * @param cursor
 * @return
 */
public abstract T cursorToBean(Cursor cursor);

/**
 * 将Bean实例转化为一个ContentValues实例，供存入数据库使用（如果对象涉及在数据库存取，需实现此方法）
 * @return
 */
public abstract ContentValues beanToValues();
```

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

```
@TableColumn(type = TableColumn.Types.TEXT, isIndex = true, isNotNull = true)
public String username;
@TableColumn(type = TableColumn.Types.TEXT)
public String password;
```

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

```
//数据库相关参数设置
DBConfig config = new DBConfig.Builder()
    .addTable(LoginUser.class)
    .setName("sincere.db")
    .setVersion(2)
    .setAuthority("com.iss.sincere")
    .build();
```

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 初始化数据库

```
IssDBFactory.init(getContext(), config);
```

调用 IssDBFactory.init() 方法，传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

```
public static Uri URI_LOGINUSER = IssContentProvider.buildUri(LoginUser.class);
```

调用 ContentResolver 的相关方法来访问数据库：

```
ContentResolver mResolver = context.getContentResolver();
mResolver.insert(URI_LOGINUSER, bean.beanToValues());
```



















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

- └─  com.iss.httpclient
 - └─  NormalHttpClient.java 34 13-4-17 下午3:04 liaowx
- └─  com.iss.httpclient.core
 - └─  AbstractHttpClient.java 34 13-4-17 下午3:04 liaowx
 - └─  BasicRequestHandler.java 34 13-4-17 下午3:04 liaowx
 - └─  ConsoleRequestLogger.java 34 13-4-17 下午3:04 liac
 - └─  HttpDelete.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpGet.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpHead.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpMethod.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpPost.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpPut.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpRequest.java 34 13-4-17 下午3:04 liaowx
 - └─  HttpRequestException.java 34 13-4-17 下午3:04 liaov
 - └─  HttpResponse.java 34 13-4-17 下午3:04 liaowx
 - └─  ParameterMap.java 34 13-4-17 下午3:04 liaowx
 - └─  RequestHandler.java 34 13-4-17 下午3:04 liaowx
 - └─  RequestLogger.java 34 13-4-17 下午3:04 liaowx

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 and its sibling AndroidHttpClient 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 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

```
mClient = new NormalHttpClient(url);
```

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

```
ParameterMap params = mClient.newParams();  
params.add("action", action);  
params.add("email", email);  
params.add("password", password);
```

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

```
mClient.addHeader("Cookie", cookie);
```

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

```
HttpResponse res = mClient.post(path, params);  
result = res.getBodyAsString();
```

调用 NormalHttpClient 的 post()/get()/delete()等方法，传入配置的访问参数，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_EXTERNAL_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

```
<activity
    android:name="com.iss.loghandler.ErrorHandler"
    android:process=":issExceptionProcess"
    android:taskAffinity="system.ErrorHandler" >
    <intent-filter>
        <category android:name="android.intent.category.DEFAULT" />
        <action android:name="android.intent.action.VIEW" />
        <data android:mimeType="errors/itotemUnhandleCatcher" />
    </intent-filter>
</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

```
ErrorHandler.registerNewErrorHandler(this);
ErrorHandler.enableEmailReports("wxliaoc@isoftstone.com", "iss App test");
```


2.7 com.iss.service

2.8 com.iss.utils

- com.iss.utils
 - DimensionPixelUtil.java 104 13-5-27 上午10:41 liaowx
 - ImageUtils.java 99 13-5-16 下午2:58 liaowx
 - Md5Util.java 34 13-4-17 下午3:04 liaowx
 - PhoneInfo.java 105 13-5-27 上午10:57 liaowx
 - StringUtil.java 34 13-4-17 下午3:04 liaowx
 - TimeUtils.java 99 13-5-16 下午2:58 liaowx
 - VersionUtils.java 59 13-5-2 下午6:14 liaowx

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(InputStream 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

- com.iss.view.calendar
 - CalendarCellView.java 108 13-5-29 下午12:12 liaowx
 - CalendarGridView.java 108 13-5-29 下午12:12 liaowx
 - CalendarPickerView.java 108 13-5-29 下午12:12 liaowx
 - CalendarRowView.java 108 13-5-29 下午12:12 liaowx
 - Logr.java 108 13-5-29 下午12:12 liaowx
 - MonthCellDescriptor.java 108 13-5-29 下午12:12 liaowx
 - MonthDescriptor.java 109 13-5-30 下午4:10 liaowx
 - MonthView.java 108 13-5-29 下午12:12 liaowx

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。

6月 2011						
周日	周一	周二	周三	周四	周五	周六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

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

6月 2011

周日	周一	周二	周三	周四	周五	周六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

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;

6月 2011

周日	周一	周二	周三	周四	周五	周六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

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。

6月 2011						
周日	周一	周二	周三	周四	周五	周六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

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 封装了用于处理 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

```
└─ com.iss.view.common
   └─ CustomScrollView.java 115 13-6-4 下午5:39 wxliaoc
   └─ MarqueeTextView.java 115 13-6-4 下午5:39 wxliaoc
   └─ ToastAlone.java 115 13-6-4 下午5:39 wxliaoc
```

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

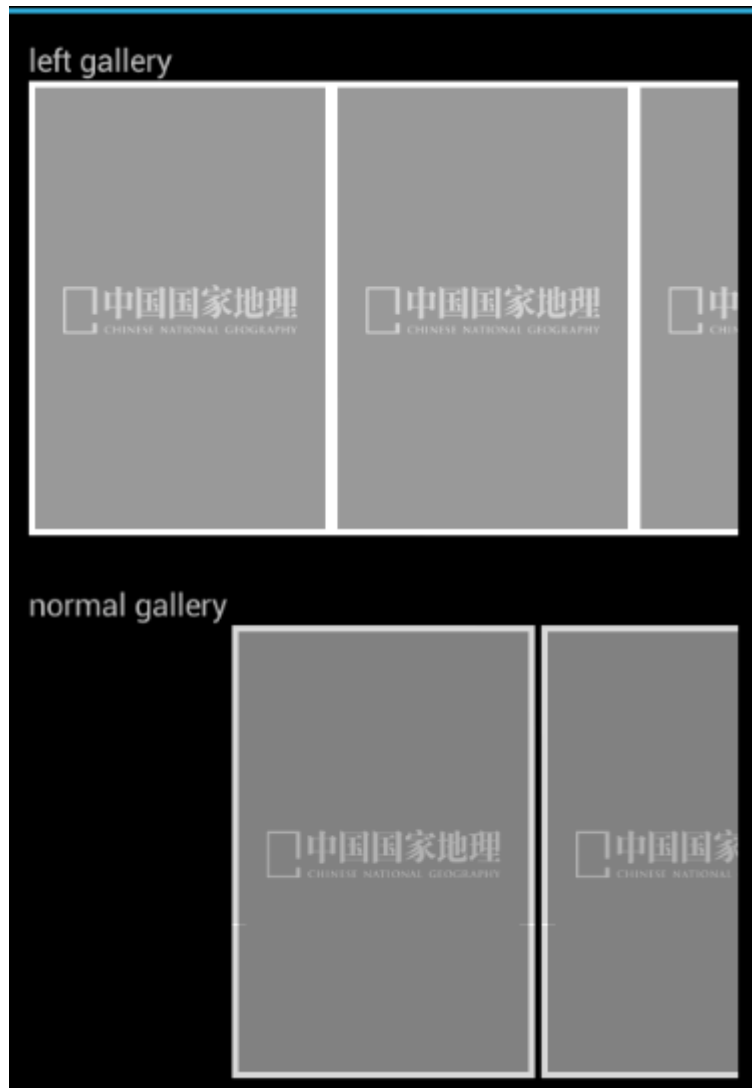
2.10.3 ToastAlone

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

- com.iss.view.leftgallery
 - ↳ AbsSpinner.java 115 13-6-4 下午5:39 wxliaoc
 - ↳ AdapterView.java 115 13-6-4 下午5:39 wxliaoc
 - ↳ Gallery.java 115 13-6-4 下午5:39 wxliaoc

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

```
└─ com.iss.view.photoview
   ├── Compat.java 115 13-6-4 下午5:39 wxliaoc
   ├── IPhotoView.java 115 13-6-4 下午5:39 wxliaoc
   ├── PhotoView.java 115 13-6-4 下午5:39 wxliaoc
   ├── PhotoViewAttacher.java 115 13-6-4 下午5:39 wxliaoc
   ├── ScrollerProxy.java 115 13-6-4 下午5:39 wxliaoc
   ├── SDK16.java 115 13-6-4 下午5:39 wxliaoc
   └── VersionedGestureDetector.java 115 13-6-4 下午5:39 wxliaoc
```

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(OnMatrixChangeListener listener)
|--setOnPhotoTapListener(OnPhotoTapListener listener)
|--setOnViewTapListener(OnViewTapListener listener)
|--setScaleType(ScaleType type)
|--setZoomable(boolean 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

```
<declare-styleable name="PullToRefresh">

    <!-- A drawable to use as the background of the Refreshable View -->
    <attr name="ptrRefreshableViewBackground" format="reference|color" />

    <!-- A drawable to use as the background of the Header and Footer Loading Views -->
    <attr name="ptrHeaderBackground" format="reference|color" />

    <!-- Text Color of the Header and Footer Loading Views -->
    <attr name="ptrHeaderTextColor" format="reference|color" />

    <!-- Text Color of the Header and Footer Loading Views Sub Header -->
    <attr name="ptrHeaderSubTextColor" format="reference|color" />

    <!-- Mode of Pull-to-Refresh that should be used -->
    <attr name="ptrMode">
        <flag name="disabled" value="0x0" />
        <flag name="pullFromStart" value="0x1" />
        <flag name="pullFromEnd" value="0x2" />
        <flag name="both" value="0x3" />
        <flag name="manualOnly" value="0x4" />

        <!-- These last two are deprecated -->
        <flag name="pullDownFromTop" value="0x1" />
        <flag name="pullUpFromBottom" value="0x2" />
    </attr>

    <!-- Whether the Indicator overlay(s) should be used -->
    <attr name="ptrShowIndicator" format="reference|boolean" />

    <!-- Drawable to use as Loading Indicator. Changes both Header and Footer. -->
    <attr name="ptrDrawable" format="reference" />

    <!-- Drawable to use as Loading Indicator in the Header View. Overrides value set in ptrDrawable. -->
    <attr name="ptrDrawableStart" format="reference" />

    <!-- Drawable to use as Loading Indicator in the Footer View. Overrides value set in ptrDrawable. -->
    <attr name="ptrDrawableEnd" format="reference" />

    <!-- Whether Android's built-in Over Scroll should be utilised for Pull-to-Refresh. -->
    <attr name="ptrOverScroll" format="reference|boolean" />

    <!-- Base text color, typeface, size, and style for Header and Footer Loading Views -->
    <attr name="ptrHeaderTextAppearance" format="reference" />

    <!-- Base text color, typeface, size, and style for Header and Footer Loading Views Sub Header -->
    <attr name="ptrSubHeaderTextAppearance" format="reference" />

    <!-- Style of Animation should be used displayed when pulling. -->
    <attr name="ptrAnimationStyle">
        <flag name="rotate" value="0x0" />
        <flag name="flip" value="0x1" />
    </attr>

    <!-- Whether the user can scroll while the View is Refreshing -->
    <attr name="ptrScrollingWhileRefreshingEnabled" format="reference|boolean" />

</declare-styleable>
```

2.14 com.iss.view.waterfall

- com.iss.view.waterfall
 - AbsWaterAdapter.java 115 13-6-4 下午5:39 wxliaoc
 - NetImageView.java 115 13-6-4 下午5:39 wxliaoc
 - WaterView.java 123 13-6-7 上午11:25 wxliaoc

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)

```
<?xml version="1.0" encoding="utf-8"?>
<com.iss.view.pulltorefresh.PullToRefreshWaterView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ptr="http://schemas.android.com/apk/res-auto/com.iss.appdemo"
    android:id="@+id/waterview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingLeft="5dip"
    android:paddingRight="5dip"
    ptr:ptrAnimationStyle="flip"
    ptr:ptrDrawable="@drawable/pulltorefresh_down_arrow"
    ptr:ptrHeaderTextColor="@color/color_ju_yellow"
    ptr:ptrMode="both"
    ptr:ptrScrollingWhileRefreshingEnabled="false" />
```

- 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，

```
@Override
public int getColumns() {
    return COLUMNS;
}
```

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）

```

@Override
public View getView(int position, View convertView, ViewGroup parent) {
    WaterImageView imageView = null;
    if (convertView == null) {
        imageView = new WaterImageView(mContext);
    } else {
        imageView = (WaterImageView) convertView;
    }
    imageView.setScaleType(ScaleType.CENTER_CROP);
    String url = urls.get(position);
    if (url != null) {
        //通过获取随机数来控制图片显示的大小
        int imageType = getRandom();
        switch(position){
            //强制设置前8个view的随机数，目的是为了看到的第一屏瀑布流样式效果一样
            case 0:
                imageType = 2;
                break;
            case 1:
                imageType = 0;
                break;
            case 2:
                imageType = 1;
                break;
            case 3:
                imageType = 2;
                break;
            case 4:
                imageType = 1;
                break;
            case 5:
                imageType = 2;
                break;
        }
    }
}

```

```

        case 6:
            imageType = 2;
            break;
        case 7:
            imageType = 2;
            break;
    }
    switch (imageType) {
        //在这里设置的宽高，ItotemWaterView里会根据实际屏幕宽度来等比拉伸
        case 0:
            imageView.setDefault(R.drawable.loading_bg_192_144);
            imageView.setLayoutParams(new LayoutParams(200, 150));
            break;
        case 1:
            imageView.setDefault(R.drawable.loading_bg_192_192);
            imageView.setLayoutParams(new LayoutParams(200, 200));
            break;
        case 2:
            imageView.setDefault(R.drawable.loading_bg_200_300);
            imageView.setLayoutParams(new LayoutParams(200, 300));
            break;
        default:
            imageView.setDefault(R.drawable.loading_bg_192_192);
            imageView.setLayoutParams(new LayoutParams(200, 200));
            break;
    }
    if (!TextUtils.isEmpty(url)) {
        imageView.setUrl(url);
    }
}
return imageView;
}

```

3.设置 WaterView 的 Adapter:

```

waterAdapter = new WaterAdapter(this);
waterView.setAdapter(waterAdapter);

```

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

```
waterView.setOnResetViewDataListener(new OnResetViewDataListener() {

    @Override
    public void onRecycleData(View view) {
        if (view instanceof NetImageView) {
            ((NetImageView) view).recycle(true);
        }

    }

    @Override
    public void onLoadData(View view) {
        if (view instanceof NetImageView) {
            ((NetImageView) view).reload(false);
        }
    }
});
```

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

