## 1.weight与layout\_weight属性

**Weight**属性代表的是控件显示优先级，如Button1 weight=1； Button2 weight=2 ；那么显示出来的就是Button1占据2/(1+2)的空间，Button2占据1/(1+2)的空间。

**Layout\_weight**属性代表的是剩余空间分配，如果Button1 设置了layout\_width=”match\_parent”;layout\_weight=”2”；而Button2设置了layout\_width=”match\_parent”;layout\_weight=”3”那么剩余空间为：(layout\_width-2layout\_width)=-layout\_width；那么Button1占layout\_width-2/(2+3)layout\_width=3/5layout\_width；Button2占layout\_width-3/(2+3)layout\_width=2/5layout\_width；

## 2.四大组件

android系统提供的四大组件分别是活动（Activity）、服务（Service）、广播接收器（Broadcast Receiver）和内容提供器（Content Provider）。凡是在应用中看到的东西，都是放在活动中的；而服务无法看到，它是一直在后台默默运行的，即使用户退出了应用，服务仍然可以继续运行，广播可以允许你的应用接受来自各处的广播消息，比如电话、短信等，你的应用同样可以向外发送广播消息；内容提供器为应用程序之间的数据共享提供了可能，比如读取系统电话簿中的联系人。

## 3.目录解析

gen：这个目录里的内容都是自动生成的，主要有一个 R.java 文件，你在项目中添加的任

何资源都会在其中生成一个相应的资源 id。这个文件永远不要手动去修改它。

res：这个目录下的内容有点多，前面提到的R.java文件也是根据这个目录下的文件自动生成的，图片推荐放在mipmap目录下，布局放在layout目录下，字符串放在values目录下。

AndroidMainfest.xml：这是整个Android项目的配置文件，在程序中定义所有的四大组件都需要在这个文件里注册，另外可以在这个文件中给应用程序添加权限声明，也可以重新指定你创建项目时指定的程序最低兼容版本和目标版本。

## 4.日志工具Log的使用

Log.v()打印最为琐碎，意义最小的日志，对应级别verbose，是android日志中级别最低的一种

Log.d()用于打印一些调试信息，对应级别debug，比verbose高一级

Log.i()用于打印一些比较重要的数据，这些数据应该是你非常想看到的，对应级别info，比debug高一级

Log.w()用于打印一些警告信息，提示存在风险，最好去修复出现警告的地方，对应级别warn，比info高一级

Log.e()用于打印出错信息，对应级别error，比warn高一级

（一般打印信息最好传入第一个参数为类名，便于删选信息）

## 5.Intent

Intent不仅可以指明当前组件想要执行的动作，还可以在不同组件之间传递数据，Intent一般可用于启动活动、启动服务、以及发送广播等场景，使用隐式Intent。

## 6.使用隐式Intent启动其它程序的活动

使用隐式Intent，不仅可以启动自己程序内的活动，还可以启动其他程序的活动，这使得 Android 多个应用程序之间的功能共享成为了可能

Intent intent = new Intent(Intent.ACTION\_VIEW);

intent.setData(Uri.parse("http://www.baidu.com"));

startActivity(intent);

Intent.ACTION\_VIEW：**这是一个 Android 系统内置的动作，其常量值为 android.intent.action.VIEW**。

Uri.parse()：将一个网址字符串解析成一个 Uri 对象，再调用 Intent的 setData()方法将这个Uri对象传递进去，这样就调用了系统默认浏览器去解析这个地址。

还可以在AndroidManifest文件中配置activity如下：

<activity android:name=".ThirdActivity" >

<intent-filter>

<action android:name="android.intent.action.VIEW" />

<category android:name="android.intent.category.DEFAULT" />

<data android:scheme="http" />

</intent-filter>

</activity>

通过建立这样一个活动，让它也能够响应打开网页的Intent。

## 7.intent传递数据和返回数据

传递数据的方法：

String date = “Hello SeondActivity”;

Intent intent = new Intent(FirstActivity.this,SecondActivity.class);

intent.putExtra(“extra\_date”,date);

startActivity(“intent”);

接收端：

Intent intent = getIntent();

String date = intent.getStringExtra(“extra\_date”);

返回数据的方法：

在被调用的activity中绑定一个按钮事件，点击返回参数：

button.setOnClickListener(new OnClickListener(){

@Override

Public void onClick(View v){

Intent intent = new Intent();

intent.putExtra(“date\_return”,”Hello FirstActivity”);

setResult(RESULT\_OK,intent);

Finish();

}

})

**而如果在当前页面不是通过点击按钮，而是通过Back键呢？**

**那么只需重写onBackPressed方法就可以了。**

在调用的这个activity中重写onActivityResult方法得到返回的参数：

@Override

Protected void onActivityResult(int requestCode , int resultCode , Intent date){

Switch(requestCode){

Case 1:

If(requestCode == RESULT\_OK){

String returnedDate = date.getStringExtra(“date\_return”);

}

Default:

}

}

## 8.在Android上发送HTTP请求的两种方式

一种是HttpURLConnection，一种是HttpClient

**HttpURLConnection**：new出一个URL对象，然后传入目标的网络地址，然后调用一下openConnection()就可以了

URL url = new URL(“http://www.baidu.com”);

HttpURLConnection connection = (HttpURLConnection) url.openConnection();

connection.setRequestMethod(“GET/POST”)；//此处的GET/POST分别代表从服务器获得数据，和提交数据给服务器。

接下来可以进行一些自由定制：比如设置连接超时、读取超时的毫秒数等

connection.setConnectTimeout(8000);

connection.setReadTimeout(8000);

之后再调用getInputStream()方法就可以获取服务器中返回的数据流了：

InputStream in =connection.getInputStream();

最后可以使用disconnect()方法将这个Http连接关掉

Connection.disconnect();

**HttpClient:**HttpClient是一个接口，所以无法创建实例，通常情况下都会创建一个DefaultHttpClient实例：

HttpClient httpClient = new DefaultHttpClient();

如果想发起一条GET请求：可创建一个HttpGet对象，并传入目标的网络地址，然后调用HttpClient的execute()方法即可：

HttpGet httpGet = new HttpGet(“http://www.baidu.com“);

httpClient.execute(httpGet);

如果是发起一条POST请求则复杂一些：要通过一个NameValuePair集合来存放待提交的参数，并将这个参数集合传入到一个UrlEncodeFormEntity中，然后调用HttpPost的setEntity()方法将构建好的UrlEncodeFormEntity传入：

HttpPost httpPost = new HttpPost(“http://www.baidu.com”);

List<NameValuePair> params = new ArrayList<nameValuePair>();

Parms.add(new BasicNameValuePair(“username”,”admin”));

Parms.add(new BasicNameValuePair(“password”,”123456”));

UrlEncodedFormEntity entity = new UrlEncodeFormEntity(params,”utf-8”);

httpPost.setEntity(entity);

httpClient.execute(httpPost);

执行execute()方法之后会返回一个HttpResponse对象，服务器返回的所有信息就包含在这里面，如果状态码等于200就说明请求和响应都成功了。

if(httpResponse.getStatusLine().getStatusCod()==200){

//请求和响应都成功了

}

## 9.Activity之间的平滑效果

<set xmlns:android="http://schemas.android.com/apk/res/android"

android:interpolator="@android:anim/accelerate\_interpolator">

<translate android:fromXDelta="-100%p" android:toXDelta="0%p"

android:duration="500" />

</set>

*<!--*

*android:fromXDelta 动画开始的位置，*

*android:toXDelta 动画结束的位置，*

*android:duration动画的时间。*

*-->*

*Java代码中*

//设置切换动画，从右边进入，左边退出

overridePendingTransition(R.anim.*out\_to\_left*,R.anim.*in\_from\_right*;

## 10. 运算符|与&

| ： 会检查每一个 条件的真伪，再做“或”运算 (光棍的不怕死)  
||： 按照条件写的顺序，直到一个为true时，后面的条件则不再检查，直接进入条件   
& ： 会检查每一个 条件的真伪，再做“与”运算  (光棍的不怕死)  
&&： 按照条件写的顺序，直到一个为false时，后面的条件则不再检查，直接跳出

## 11.Looper.prepare()和Looper.loop()

Looper类用来为一个线程开启一个消息循环。默认情况下android中新诞生的线程是没有开启消息循环的。（主线程除外，主线程系统会自动为其创建Looper对象，开启消息循环。）     Looper对象通过MessageQueue来存放消息和事件。一个线程只能有一个Looper，对应一个MessageQueue。

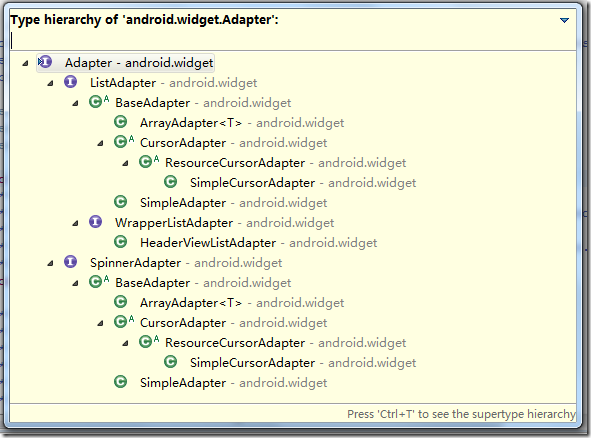
通常是通过Handler对象来与Looper进行交互的。Handler可看做是Looper的一个接口，用来向指定的Looper发送消息及定义处理方法。默认情况下Handler会与其被定义时所在线程的Looper绑定。

在非主线程中直接new Handler() 会报如下的错误: E/AndroidRuntime( 6173): Uncaught handler: thread Thread-8 exiting due to uncaught exception E/AndroidRuntime( 6173): java.lang.RuntimeException: Can't create handler inside thread that has not called Looper.prepare() 原因是非主线程中默认没有创建Looper对象，需要先调用Looper.prepare()启用Looper。

## 12. Adapter

Adapter是连接后端数据和前端显示的适配器接口，是数据和UI（View）之间一个重要的纽带。在常见的View(List View,Grid View)等地方都需要用到Adapter。

Android中所有的Adapter一览：



## 13.使用Intent传递对象的方式：

a)让Person类去实现Serializable接口，这样所有的Person对象就都是可序列化的了

b)在第一个Activity中的写法：

Person person = new Person();

person.setName(“Tom”);person.setAge(20);

……

Intent.putExtra(“person\_data”,person);

c)第二个Activity中的写法:

Person person = (Person) getIntent().getSerializableExtra(“person\_data”);

## 14.Android中使用sql语句查询的时候若要查查询条件为空的字段

应该写成where aim is null；不能用=。

## 15. Activity的四种启动模式：

Activity启动模式设置：

        <activity android:name=".MainActivity" android:launchMode="standard" />

**. standard**

        模式启动模式，每次激活Activity时都会创建Activity，并放入任务栈中。

**. singleTop**

        如果在任务的栈顶正好存在该Activity的实例， 就重用该实例，否者就会创建新的实例并放入栈顶(即使栈中已经存在该Activity实例，只要不在栈顶，都会创建实例)。

**. singleTask**

        如果在栈中已经有该Activity的实例，就重用该实例(会调用实例的onNewIntent())。重用时，会让该实例回到栈顶，因此在它**上面的实例将会被移除栈**。如果栈中不存在该实例，将会创建新的实例放入栈中。

**. singleInstance**

        在一个新栈中创建该Activity实例，并让多个应用共享改栈中的该Activity实例。一旦改模式的Activity的实例存在于某个栈中，任何应用再激活改Activity时都会重用该栈中的实例，其效果相当于多个应用程序共享一个应用，不管谁激活该Activity都会进入同一个应用中。

## 16.Android中的context浅谈：

1）在生命周期上，通过getApplicationContext得到的上下文对象们只要当前的应用程序还存在，那么该对象就会一直存在，对于Activity.this上下文来说，只要当前的activity执行了onDestory方法，这个上下文对象就会一起被系统收回。

2）在应用场景上，如果我们通过一个上下文对象来执行某个动作，且希望一直处于活跃状态，那么应该用getApplicationContext来获取上下文，如数据库的操作。此时，如果采用Activity.this，那么当前Activity调用onDestory方法时，数据库就会关闭，应用程序会产生错误。

## 17.Looper

Looper是每个线程中的MessageQueue的管家，调用Looper的loop()方法后，就会进入到一个无限循环当中，然后每当发现MessageQueue中存在一条消息，就会将它取出，并传递到Handler的handleMessage方法中。

## 18. 对于compileSdkVersion强烈推荐总是使用最新的 SDK 进行编译

。在现有代码上使用新的编译检查可以获得很多好处，避免新弃用的 API ，并且为使用新的 API 做好准备。

可以使用一个高版本的buildToolsVersion去构建一个低版本的compileSdkVersion。

## 19．MUI

1. mui侧边栏使用js控制关闭：

mui('#offCanvasSide').offCanvas('close');//左侧滑栏关闭

2.mui滚动列表使用js控制回到页首：

mui('#container-left-section').scroll().scrollTo(0, 0);//定位至页首;

## 20. *getView*函数为什么重要呢？

因为它是用来刷新它所在的ListView的。它在什么时候调用的呢？就是在每一次item从屏幕外滑进屏幕内的时候，或者程序刚开始的时候创建第一屏item的时候。

1）在自定义Adapter时可以创建一个ViewHolder类来暂存Item对象，为在listview滚动时快速设置值。

ViewHolder holder = null;  
if (convertView == null) {  
 holder = new ViewHolder();  
 LayoutInflater inflater = LayoutInflater.from(mContext);  
 convertView = inflater.inflate(R.layout.treeview\_item,null);  
 holder.icon = (ImageView) convertView.findViewById(R.id.sbtz\_itemicon);  
 holder.treeText = (TextView) convertView.findViewById(R.id.sbtz\_itemtv);  
 convertView.setTag(holder);  
} else {  
 holder = (ViewHolder) convertView.getTag();  
}

static class ViewHolder{  
 ImageView icon;  
 TextView treeText;  
}

2）listview的保留上次点击位置功能：在Adapter中新建一个函数，用来设置，每次点击的item的位置。

public void setSelectedPosition(int pos){  
 selectedPos = pos;  
 notifyDataSetChanged();  
}

然后在Adapter的getView方法中判断：

if (selectedPos == position) {  
convertView.setBackgroundResource(R.color.rowSelectedColor);  
} else {  
 convertView.setBackgroundResource(R.color.white);  
}

## 21．Android为布局添加的来年各种方式：

A）

layout.startAnimation(getAnimation());

protected Animation getAnimation() {  
 Animation anim = new ScaleAnimation(0.0f, 1.0f, 0.0f, 1.0f,  
 Animation.RELATIVE\_TO\_SELF, 0.5f, Animation.RELATIVE\_TO\_SELF,  
 0.5f);// 从0.5倍放大到1倍  
 anim.setDuration(1500);  
 return anim;  
}

B）

layout.setLayoutAnimation(getAnimationController());

## 22．为布局添加以下属性，布局改变时，内容有动画效果

android:animateLayoutChanges="true"

## 23. android编译的apk比eclipse大的解决办法：

在release 的buildTypes下将 minifyEnabled 改为 true。对代码进行混淆，同时去除无用代码。

## 24.android的httpClient与ksoap2的不同：

httpClient是用来进行http请求的 他是Apache 对httpUrlConnection的封装。 ksoap2是android 用来访问webservice 服务的库。

## 25.使用代码触发listview的点击事件：

调用listview的performItemClick方法

listViewLeft.performItemClick(listViewLeft.getChildAt(click\_position),click\_position,listViewLeft.getItemIdAtPosition(click\_position ));

## 26. Toast重复一直弹出:

使用Toast显示提示信息时，避免Toast重复一直弹出，可以将Toast封装起来，通过传入context和content调用。

## 27.android下新建assets 目录：

切换到项目结构目录下，在app/src/main目录下新建directory，命名为assets。

28.自定义view组件时

如果在布局文件中使用，需要重写DemoView(Context context, AttributeSet attrs)的方法。在代码中添加组件时，重写DemoView(Context context)方法。