向fragment传参数

Bundle args = **new** Bundle();  
args.putString(Constants.***TYPE***, type);  
TabsFragment fragment = **new** TabsFragment();  
fragment.setArguments(args);  
**return** fragment;

**menuType** = getArguments().getString(Constants.***TYPE***);

使用toolbar报错

<https://blog.csdn.net/qq_28702545/article/details/52594675>

**mToolbar** = (Toolbar) findViewById(R.id.***toolbar***);  
setSupportActionBar(**mToolbar**);

主题中添加



接着报错

java.lang.IllegalArgumentException: AppCompat does not support the current theme features: { windowActionBar: false, windowActionBarOverlay: false, android:windowIsFloating: false, windowActionModeOverlay: false, windowNoTitle: false }

接着添加

<**item name="windowNoTitle"**>true</**item**>

showAsAction不可用 ，要引入app命名空间

<**menu   
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"**>

<**item  
 android:id="@+id/action\_notifications"  
 app:showAsAction="ifRoom"  
 android:icon="@mipmap/ic\_notifications"  
 android:title="notifications"**/>

# Icons for Android

PNGs suitable for Android are available from the [material icons library](https://www.google.com/design/icons/). These come in all the supported screen densities so they should look good on any device.

The icons are also available in the [material design icons git repository](https://github.com/google/material-design-icons) in the same combination of colors and sizes named as follows:

\*/drawable-{{density}}/ic\_{{name}}\_{{color}}\_{{size}}dp.png

A density-independent VectorDrawable is provided which is supported from Android Lollipop and later:

\*/drawable-anydpi-v21/ic\_{{name}}\_black\_24dp.xml

The Vector Drawable is currently only available as a black 24dp icon. This is for compatibility with our most standard icon size. To render the icon in a different color, use [drawable tinting available on Android Lollipop](https://developer.android.com/training/material/drawables.html).

When using the Vector Drawable, it may not be necessary to include the xxxhdpi density PNG since it is unlikely a device supporting that screen density does not support Vector Drawables.

getActionBar().setHomeButtonEnabled(**true**);

toolbar 替换 actionbar 后可以

getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);

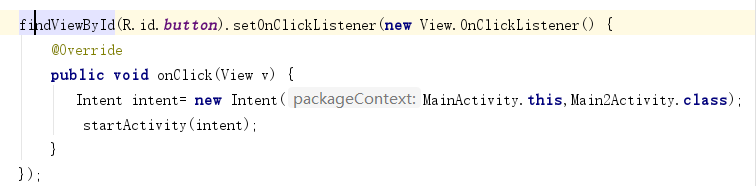
但是 toolbar.setDisplayHomeAsUpEnabled(**true**);报错

设置返回按钮可见，并设置图标

toolbar.setNavigationIcon(R.mipmap.home);

onClick 中启动activity。实例化Intent 时使用的context 要用MainActivity.**this**

单独的**this 代表**OnClickListener



使用toolbar

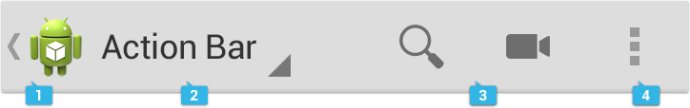
setNavigationIcon（） 和setDisplayHomeAsUpEnabled会互相覆盖

setSupportActionBar(**mToolbar**);  
**mToolbar**.setNavigationIcon(R.mipmap.***ic\_launcher***);  
getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);

setNavigationIcon（）要放在前面才能返回。不然只能显示个图标

**mToolbar**.setNavigationIcon(R.mipmap.***ic\_launcher***);  
setSupportActionBar(**mToolbar**);

<https://www.cnblogs.com/yc-755909659/p/4290784.html>



　　<1> ActionBar的图标，可显示软件图标，也可用其他图标代替。当软件不在最高级页面时，图标左侧会显示一个左箭头，用户可以通过这个箭头向上导航；

　　<2> 如果你的应用要在不同的View中显示数据，这部分允许用户来切换视图。一般的作法是用一个下拉菜单或者是Tab选项卡。如果只有一个界面，那这里可以显示应用程序的标题或者是更长一点的商标信息；

ActionView是一种可以在ActionBar中替换Action按钮的控件，

**app:actionViewClass="android.support.v7.widget.SearchView"**

**app:showAsAction="ifRoom"**



 collapseActionView

声明了这个操作视窗应该被折叠到一个按钮中，当用户选择这个按钮时，这个操作视窗展开。否则，这个操作视窗在默认的情况下是可见的，并且即便在用于不适用的时候，也要占据操作栏的有效空间。一般要配合ifRoom一起使用才会有效果。

**app:actionViewClass="android.support.v7.widget.SearchView"  
app:showAsAction="ifRoom|collapseActionView"**



非正常实现返回

通过调用setDisplayHomeAsUpEnabled()方法来启用ActionBar图标导航功能

actionBar.setDisplayHomeAsUpEnabled(true);

现在重新运行一下程序，结果如下图所示：



可以看到，在ActionBar图标的左侧出现了一个向左的箭头，通常情况下这都表示返回的意思，因此最简单的实现就是在它的点击事件里面加入finish()方法就可以了，如下所示：

switch (item.getItemId()) {

case android.R.id.home:

finish();

return true;

...

}

当点击ActionBar图标的时候，系统同样会调用onOptionsItemSelected()方法，并且此时的itemId是android.R.id.home，所以finish()方法也就是加在这里的了。

如果你还希望在代码中对SearchView的属性进行配置（比如添加监听事件等），完全没有问题，只需要在onCreateOptionsMenu()方法中获取该ActionView的实例就可以了

MenuItem searchItem = menu.findItem(R.id.action\_search);

SearchView searchView = (SearchView) searchItem.getActionView();

// 配置SearchView的属性

......

return super.onCreateOptionsMenu(menu);

有些程序可能还希望在ActionView展开和合并的时候显示不同的界面，其实我们只需要去注册一个ActionView的监听器就能实现这样的功能了

searchItem.setOnActionExpandListener(new OnActionExpandListener() {

}

过时

overflow按钮在有些手机上会显示，而在有些手机上不显示,如下图：



ActionBar最右边的overflow按钮不见，按一下Menu键，隐藏在overflow中的Action按钮就会从底部出来。overflow按钮的显示情况和手机的硬件情况是有关系的，如果手机没有物理Menu键的话，overflow按钮就可以显示，如果有物理Menu键的话，overflow按钮就不会显示出来。在ViewConfiguration这个类中有一个叫做sHasPermanentMenuKey的静态变量，系统就是根据这个变量的值来判断手机有没有物理Menu键的。当然这是一个内部变量，我们无法直接访问它，但是可以通过反射的方式修改它的值，让它永远为false就可以了

overflow中的Action按钮应不应该显示图标，是由MenuBuilder这个类的setOptionalIconsVisible变量来决定的，如果我们在overflow被展开的时候将这个变量赋值为true，那么里面的每一个Action按钮对应的图标就都会显示出来了。赋值的方法当然仍然是用反射了，代码如下所示

用toolbar替换actionbar要注意Window.FEATURE\_ACTION\_BAR

public boolean onMenuOpened(int featureId, Menu menu) {

if (featureId == Window.FEATURE\_ACTION\_BAR && menu != null) {

if (menu.getClass().getSimpleName().equals("MenuBuilder")) {

try {

Method m = menu.getClass().getDeclaredMethod("setOptionalIconsVisible", Boolean.TYPE);

m.setAccessible(true);

m.invoke(menu, true);

} catch (Exception e) {

}

}

}

return super.onMenuOpened(featureId, menu);

}



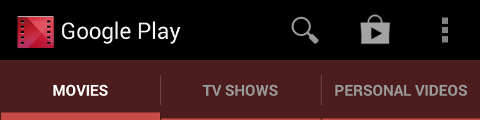
过时

**10.添加导航Tabs**

Tabs的应用可以算是非常广泛了，它可以使得用户非常轻松地在你的应用程序中切换不同的视图。而Android官方更加推荐使用ActionBar中提供的Tabs功能，因为它更加的智能，可以自动适配各种屏幕的大小。比如说，在平板上屏幕的空间非常充足，Tabs会和Action按钮在同一行显示，如下图所示：

https://images0.cnblogs.com/blog/359646/201502/141600267616199.png

而如果是在手机上，屏幕的空间不够大的话，Tabs和Action按钮则会分为两行显示，如下图所示：



下面我们就来看一下如何使用ActionBar提供的Tab功能，大致可以分为以下几步：

1. 实现ActionBar.TabListener接口，这个接口提供了Tab事件的各种回调，比如当用户点击了一个Tab时，你就可以进行切换Tab的操作。

2.为每一个你想添加的Tab创建一个ActionBar.Tab的实例，并且调用setTabListener()方法来设置ActionBar.TabListener。除此之外，还需要调用setText()方法来给当前Tab设置标题。

3.最后调用ActionBar的addTab()方法将创建好的Tab添加到ActionBar中。

actionBar.setNavigationMode(ActionBar.NAVIGATION\_MODE\_TABS);

**11.添加下拉列表导航，和使用tab的目的一样**

1.1 简单介绍

作为Activity内部的另一种导航（或过滤）模式，操作栏提供了内置的下拉列表。下拉列表能够提供Activity中内容

的不同排序模式。

启用下拉式导航的基本过程如下：

<1> 创建一个给下拉提供可选项目的列表，以及描画列表项目时所使用的布局；

<2> 实现ActionBar.OnNavigationListener回调，在这个回调中定义当用户选择列表中一个项目时所发生的行为；

<3> 用setNavigationMode()方法该操作栏启用导航模式；

<4> 用setListNavigationCallbacks()方法给下拉列表设置回调方法。

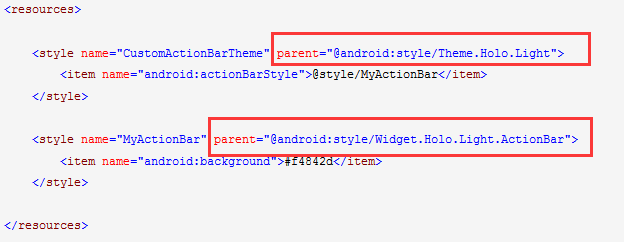
1.2 效果图如下:



// //导航模式必须设为NAVIGATION\_MODE\_LIST

actionBar.setNavigationMode(ActionBar.NAVIGATION\_MODE\_LIST);

如果想要修改ActionBar的背景，我们可以通过创建一个自定义主题并重写actionBarStyle属性来实现。这个属性可以指向另外一个样式，然后我们在这个样式中重写background这个属性就可以指定一个drawable资源或颜色，从而实现自定义背景的功能。

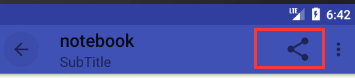


**9.添加Action Provider**

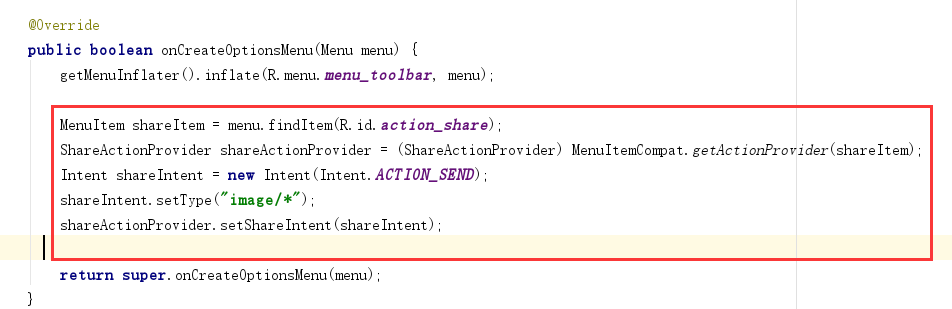
和Action View有点类似，Action Provider也可以将一个Action按钮替换成一个自定义的布局。但不同的是，Action Provider能够完全控制事件的所有行为，并且还可以在点击的时候显示子菜单。

由于每个Action Provider都可以自由地控制事件响应，所以它们不需要在onOptionsItemSelected()方法中再去监听点击事件，而是应该在onPerformDefaultAction()方法中去执行相应的逻辑。

<**item  
 android:id="@+id/action\_share"  
 app:actionProviderClass="android.support.v7.widget.ShareActionProvider"  
 app:showAsAction="ifRoom"  
 android:title="自定义"** />



添加单击响应

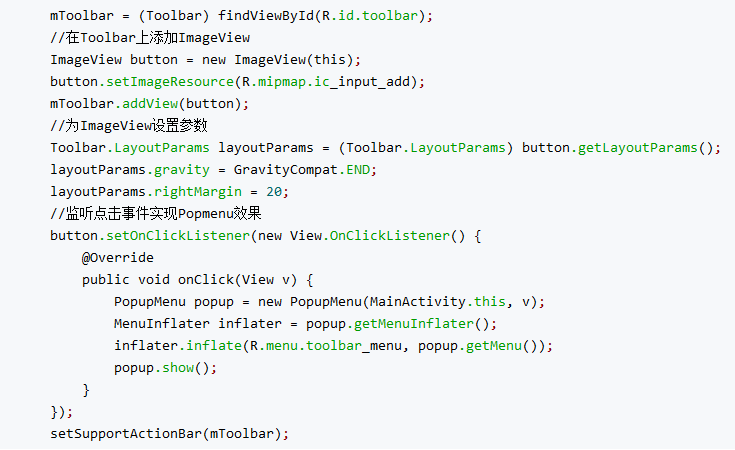


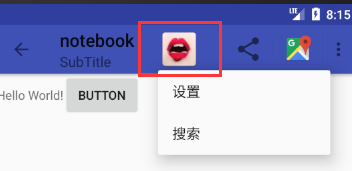
### Toolbar 上实现 PopMenu

PopMenu 是锚到 View 的菜单 ，如果空间足够，就显示在 View 的下方，否则显示在上方

在 Toolbar 上添加 ImageView，并实现 点击弹出Popmenu 效果

现在的显示方式是在 ImageView 的下方，而 overflow 的显示方式是覆盖了 overflow 图标了





我们还可以添加 Popmenu 的点击事件

popup.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {

@Override

public boolean onMenuItemClick(MenuItem item) {

return false;

}

});

使用

**android.support.design.widget.FloatingActionButton**

报错

Error inflating class android.support.design.widget.FloatingActionButton。

Didn't find class "android.support.design.widget.FloatingActionButton" on path: DexPathList[[zip file "/data/app/com.centrin.wuliu.notebook-2/base.apk"],nativeLibraryDirectories=[/data/app/com.centrin.wuliu.notebook-2/lib/x86, /system/lib, /vendor/lib]]

需要引用

implementation **'com.android.support:design:27.1.1'**

外层要用**CoordinatorLayout，使用LinearLayout** 不显示

List<String> list = Arrays.*asList*(**"xxx"**,**"yyy"**,**"zzz"**);  
list.add(**"1"**);

Caused by: java.lang.UnsupportedOperationException

at java.util.AbstractList.add(AbstractList.java:148)

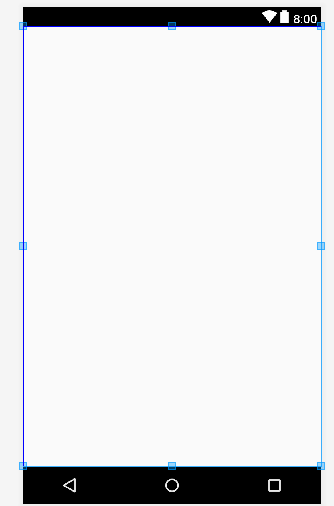
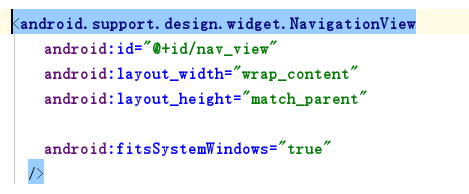
由Arrays.asList() 返回的是Arrays的内部类ArrayList，remove、add等方法AbstractList中是默认throw UnsupportedOperationException而且不作任何操作

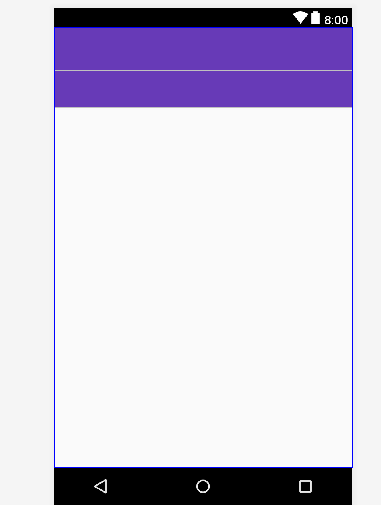
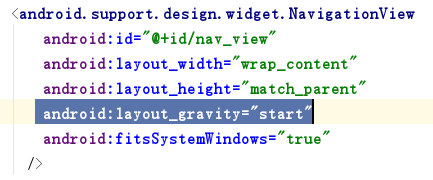
改成

List<String> list = **new** ArrayList<>(Arrays.*asList*(**"xxx"**,**"yyy"**,**"zzz"**));

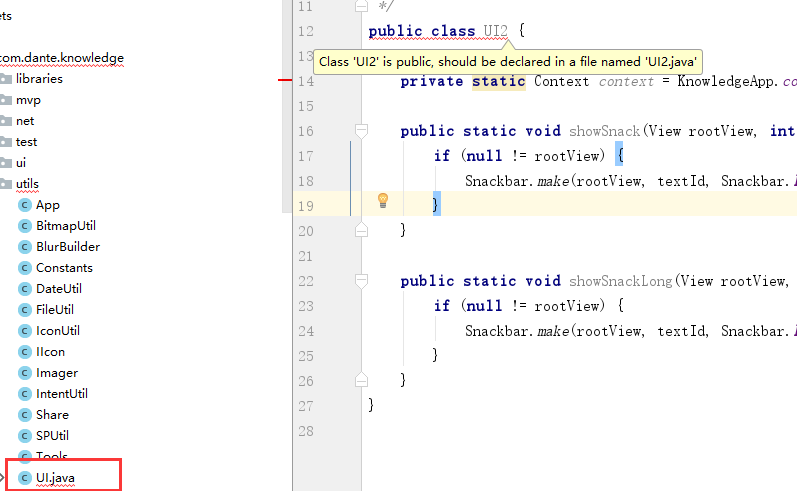
缺少**layout\_gravity**

会完全覆盖窗口





类名要跟文件名一样



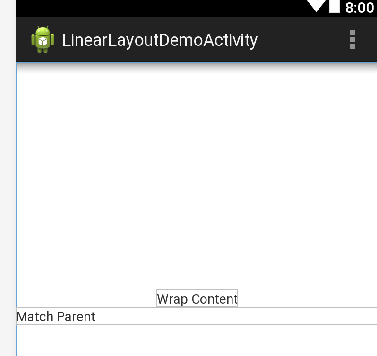
**<RelativeLayout>**

**android:layout\_centerHorizontal="true"  
android:layout\_centerVertical="true"**

**<LinearLayout**

**android:gravity="center">**

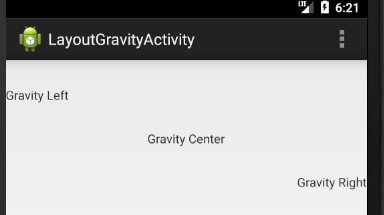
**所有子元素在父布局中都居中**



<**LinearLayout>**

<**TextView android:text="Gravity Left"** />   
<**TextView android:layout\_gravity="center" android:text="Gravity Center"** />  
<**TextView android:layout\_gravity="right" android:text="Gravity Right"** />

</**LinearLayout**>



**android:ems = "10" 的含义 :**



<**requestFocus** />

获得焦点

指的是将对应的控件宽度设为10个字符的宽度。当设置该属性后，一行中最大只能显示设置的宽度。

当设置 ems = "1" 时，//EditText的宽度为一个中文的宽度即2个字符的宽度

https://images2015.cnblogs.com/blog/803469/201611/803469-20161125111847831-388387575.png//所以这里只能看得到86显示,33被遮住了，也不能再输入了  
https://images2015.cnblogs.com/blog/803469/201611/803469-20161125112013925-807568869.png//中文就只显示一个1中文

注意，当 android:layout\_width="match\_parent" 时该属性不会生效

**contentDescription**这个属性必须在用户的Accessible 中的相应属性开启后才能使用。

对于一些视力有障碍的用户，android提供给用户一个很有用的功能，可以帮助使用app。

<**ImageView  
 android:id="@+id/imageView1"  
 android:contentDescription="Demo Image"  
 android:src="@drawable/ic\_launcher"** />

由于这个控件没有text属性，对于一些用户来说，无法理解这个控件是做什么的。这时用户点击这个控件。android系统会自动使用人声朗读控件上android:contentDescription属性说指向的内容

String[] myStringArray = { **"Bruce"**, **"Wayne"**, **"Bill"** };  
ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(**this**,   
 R.layout.***simple\_list\_view\_item***, myStringArray);

listView.setAdapter(adapter);

<**Button android:onClick="firstButtonClicked"  
 android:text="XML onClick"** />

**public void** firstButtonClicked(View v) {  
 SimpleAlertDialog.*displayWithOK*(**this**, **"firstButton clicked via XML handler"**);  
}

ListViewClicks

listView.setOnItemClickListener(**new** OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 String country = **adapter**.getItem(position);  
 SimpleAlertDialog.*displayWithOK*(ListViewClicksActivity.**this**, country);  
 Toast.*makeText*(ListViewClicksActivity.**this**, country, Toast.***LENGTH\_SHORT***).show();  
 }  
   
});

允许在主线程中访问网络

StrictMode.*setThreadPolicy*(  
 **new** StrictMode.ThreadPolicy.Builder().permitNetwork().build());

getActionBar().setTitle(**"Click an Icon"**);

URL url = **new** URL(address);

URLConnection conn;  
 InputStream in;  
 Bitmap bitmap;  
**try** {  
 conn = url.openConnection();  
 conn.connect(); //相当于访问？  
 in = conn.getInputStream();  
 bitmap = BitmapFactory.*decodeStream*(in);   
 in.close();  
} **catch** (IOException e) {  
 bitmap = **null**;  
}

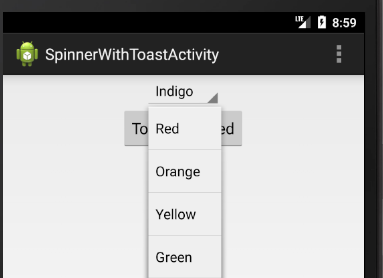
**extends** AsyncTask<Void, Void, Void>

SmartImageDownload

**private void** downloadSmartImageFromUrl(String address) {  
 AsyncHttpClient client = **new** AsyncHttpClient();  
 client.get(address, **new** BinaryHttpResponseHandler() {  
 @Override  
 **public void** onSuccess(**int** statusCode, Header[] headers, **byte**[] image) {  
 Bitmap bitmap = BitmapFactory.*decodeByteArray*(image, 0, image.**length**);  
 ImageView img = (ImageView) findViewById(R.id.***ivSmartImage***);  
 img.setImageBitmap(bitmap);  
  
 }  
  
 @Override  
 **public void** onFailure(**int** statusCode, Header[] headers, **byte**[] binaryData, Throwable error) {  
 error.printStackTrace();  
 }  
 }  
 );  
}



**int** selected = **rdgVal**.getCheckedRadioButtonId();  
RadioButton b = (RadioButton) findViewById(selected);



<**string-array name="spinner\_options"**>  
 <**item**>Red</**item**>  
 <**item**>Orange</**item**>  
 <**item**>Yellow</**item**>  
 <**item**>Green</**item**>  
 <**item**>Blue</**item**>  
 <**item**>Indigo</**item**>  
 <**item**>Violet</**item**>  
</**string-array**>

***simple\_spinner\_item combobox 的样式***

ArrayAdapter<CharSequence> adapter = ArrayAdapter.*createFromResource*(**this**,  
 R.array.***spinner\_options***, android.R.layout.***simple\_spinner\_item***);

*// Set layout style during dropdown 设置弹出菜单项的样式* adapter.setDropDownViewResource(android.R.layout.***simple\_spinner\_dropdown\_item***);  
*// Load data from adapter***spinner**.setAdapter(adapter);

**spinner**.getSelectedItem().toString()

android:ellipsize=”start”—–省略号显示在开头 "...pedia"  
android:ellipsize=”end”——省略号显示在结尾  "encyc..."  
android:ellipsize=”middle”—-省略号显示在中间 "en...dia"  
android:ellipsize=”marquee”–以横向滚动方式显示(需获得当前焦点时)

**tpTime** = (TimePicker) findViewById(R.id.***tpTime***);

String time = **tpTime**.getCurrentHour() + **":"** + **tpTime**.getCurrentMinute();

**class** GridImageAdapter **extends** ArrayAdapter<String>

**public** View getView(**int** position, View convertView, ViewGroup parent) {  
 ImageView v = **new** ImageView(GridViewDemoActivity.**this**);  
 **int** resId = getResources().getIdentifier(getItem(position), **"drawable"**, getPackageName());  
 v.setImageDrawable(getResources().getDrawable(resId));  
 **return** v;  
}

CursorLoader

Cursor c = cursorLoader.loadInBackground();

Unsupported method: BaseConfig.getApplicationIdSuffix().

The version of Gradle you connect to does not support that method.

To resolve the problem you can change/upgrade the target version of Gradle you connect to.

Alternatively, you can ignore this exception and read other information from the model.

将 build.gradle 中的

buildscript {

repositories {

mavenCentral()

}

dependencies {

classpath **'com.android.tools.build:gradle:1.1.1'**

}

}

改成：

buildscript {

repositories {

mavenCentral()

}

dependencies {

classpath **'com.android.tools.build:gradle:2.3.2'**

}

}

Minimum supported Gradle version is 3.3. Current version is 2.2.

**distributionUrl**=**https\://services.gradle.org/distributions/gradle-2.2-all.zip**

改成

**distributionUrl**=**https\://services.gradle.org/distributions/gradle-4.4-all.zip**

Gradle sync failed: Cause: failed to find Build Tools revision 21.1.2

Consult IDE log for more details (Help | Show Log) (4s 405ms)

File->settings| system settings| android sdk| sdk tools| show package details

分享

Intent shareIntent = **new** Intent();  
*// Construct a ShareIntent with link to image*shareIntent.setAction(Intent.***ACTION\_SEND***);  
shareIntent.setType(**"\*/\*"**);  
shareIntent.putExtra(Intent.***EXTRA\_TEXT***, (String)tvTitle.getText());  
shareIntent.putExtra(Intent.***EXTRA\_STREAM***, bmpUri);  
*// Launch share menu*startActivity(Intent.*createChooser*(shareIntent, **"Share Image"**));

**Android Studio 0.5.0** requires **gradle-plugin 0.9.0**

The gradle-plugin **0.9.0** works with **Gradle 1.10 or Gradle 1.11**

Modify your build.gradle script as suggested by Cedric.

**There is a relation between gradle-plugin and the gradle version.** For example

**UPDATED TO 27/11/2015**:

com.android.tools.build:gradle:0.6.+ -> gradle 1.8

com.android.tools.build:gradle:0.7.+ -> gradle 1.9

com.android.tools.build:gradle:0.8.+ -> gradle 1.9/1.10

com.android.tools.build:gradle:0.9.+ -> gradle 1.10/1.11

com.android.tools.build:gradle:0.10.+ -> gradle 1.10/1.11/1.12

com.android.tools.build:gradle:0.11.+ -> gradle 1.10/1.11/1.12

com.android.tools.build:gradle:0.12.+ -> gradle 1.10/1.11/1.12

com.android.tools.build:gradle:0.13.+ -> gradle 2.1

com.android.tools.build:gradle:0.14.+ -> gradle 2.1

com.android.tools.build:gradle:1.0.+ -> gradle 2.2.1-2.3

com.android.tools.build:gradle:1.1.+ -> gradle 2.2.1-2.3

com.android.tools.build:gradle:1.2.+ -> gradle 2.2.1+

com.android.tools.build:gradle:1.3.+ -> gradle 2.2.1+

com.android.tools.build:gradle:1.3.+ -> gradle 2.2.1+

com.android.tools.build:gradle:1.5.+ -> gradle 2.2.1+

com.android.tools.build:gradle:2.0.+ -> gradle 2.10.0+

You can find gradle version used in your project in the file gradle/wrapper/gradle-wrapper.properties

Also there is a relation between gradle-plugin and the IDE version.

Android Studio 0.3.x -> gradle-plugin 0.6

Android Studio 0.4.x -> gradle-plugin 0.7

Android Studio 0.4.3+ -> gradle-plugin 0.8

Android Studio 0.5.x -> gradle-plugin 0.9

Android Studio 0.5.8 -> gradle-plugin 0.9.+ or 0.10.+

Android Studio 0.5.9 -> gradle-plugin 0.9.+ or 0.10.4+

Android Studio 0.6.x -> gradle-plugin 0.11.+

Android Studio 0.8.x -> gradle-plugin 0.12.+

Android Studio 0.8.11+ -> gradle-plugin 0.13.+ / gradle-plugin 0.12.+

Android Studio 0.9.x -> gradle-plugin 0.14.+

Android Studio 1.0.0 -> gradle-plugin 1.0.0

Android Studio 1.1.x -> gradle-plugin 1.0.0/ 1.1.x / 1.2.x

Android Studio 1.2.x -> gradle-plugin 1.0.0/ 1.1.x / 1.2.x

Android Studio 1.3.x -> gradle-plugin 1.0.0/ 1.1.x / 1.2.x / 1.3.x

Android Studio 1.4.x -> gradle-plugin 1.0.0/ 1.1.x / 1.2.x / 1.3.x

Android Studio 1.5.x -> gradle-plugin 1.0.0/ 1.1.x / 1.2.x / 1.3.x / 1.5.x

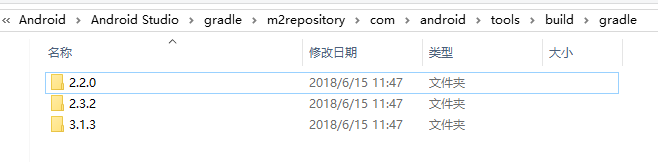
Android Studio 2.0.x -> gradle-plugin 2.0.0

Also gradle 1.12 requires Android Studio 0.5.8+

For updated news you can check this link: <http://tools.android.com/recent>  
For updated doc about the gradle plugin [check here](https://developer.android.com/intl/ru/tools/revisions/gradle-plugin.html).

If you change your plugin version, check compatibility, and then click sync project with your gradle files.It will download a new plugin version if you need it.

1、查看本地安装了哪个版本的Gradle Build Tool

打开Android Studio的安装目录，注意不是SDK安装目录或项目文件所在目录。在Android Studio的安装目录中，找到以下文件夹：   
gradle\m2repository\com\android\tools\build\gradle\   
在最后的这个gradle文件夹中，可以看到本机已经安装了哪些版本的Gradle插件。如图：   


dependencies {  
 classpath **'com.android.tools.build:gradle:2.2.0'**}

Styles work by defining style names associated with a series of properties to apply to a view

If you want to inherit from styles that you've defined yourself, you do not even have to use the parent attribute. Instead, as a shortcut just prefix the name of the style you want to inherit to the name of your new style, separated by a period:

<style name="LargeFont">

<item name="android:textSize">40sp</item>

</style>

<style name="LargeFont.Red">

<item name="android:textColor">#C80000</item>

</style>

<**Button  
 android:id="@+id/Button01"  
 android:layout\_alignLeft="@+id/button2"  
 android:layout\_below="@+id/button2"  
 style="@style/** **LargeFont.Red "  
 android:background="@drawable/selector\_ninepatch\_buttons"  
 android:text="9-Patch Button #2"** />

Instead of applying the style to a particular individual view, you can apply a collection of styles as a **Theme** to an Activity or application.

When you do so, every View within the Activity or application will apply each property that it supports.

This theme contains item nodes that often reference other styles or colors:

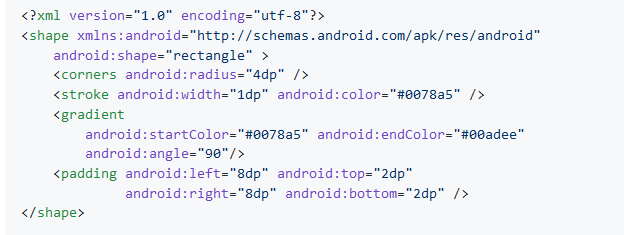
There are at least 17 types of drawables but there are five that are most important to understand:

1. **Shape Drawables** - Defines a shape with properties such as stroke, fill, and padding
2. **StateList Drawables** - Defines a list of drawables to use for different states
3. **LayerList Drawables** - Defines a list of drawables grouped together into a composite result
4. **NinePatch Drawables** - A PNG file with stretchable regions to allow proper resizing
5. **Vector Drawables** - Defines complex XML-based vector images

A shape is simply a collection of properties that are combined to describe a background. The shape can be described with properties such as corners for rounding, gradient for backgrounds, padding for spacing, solid for background colors, and stroke for border.

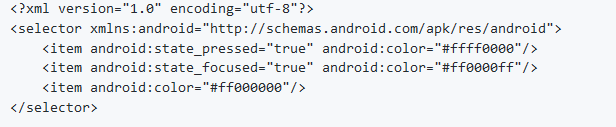
Shapes also support [gradients backgrounds](http://developer.android.com/guide/topics/resources/drawable-resource.html#gradient-element) as well as supporting properties such as startColor, centerColor, endColor, angle. Different gradients such as radial, linear or sweep can be selected using the type property.

Here's an example of a simple linear gradient shape specified in res/layout/drawable/gradient\_shape.xml:



### State List

A [StateListDrawable](http://developer.android.com/guide/topics/resources/drawable-resource.html#StateList) is a drawable object defined in XML that uses several different images to represent the same graphic, depending on the state of the object。and applied to any field that accepts a color value such as the textColor property of a button in a layout file:



### Creating a Layer List

A [LayerDrawable](http://developer.android.com/guide/topics/resources/drawable-resource.html#LayerList) is a drawable object that manages an array of other drawables. Each drawable in the list is drawn in the order of the list—the last drawable in the list is drawn on top. Each drawable is represented by an <item> element inside a single <layer-list> element.

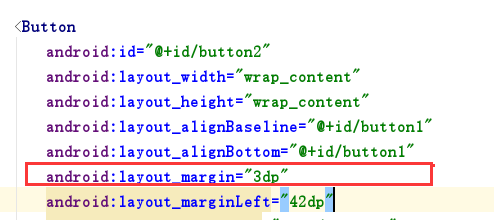
<https://www.cnblogs.com/jhcelue/p/6911371.html>

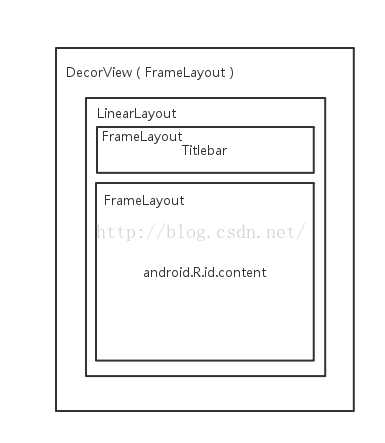
关于layout\_margin

搞Android时间也不短了，对layout\_margin也不陌生了，可近期遇到一个问题让我发现，对它的认识还不够深入全面。大量网络资料上都说，layout\_margin指view距离父view的距离。这个说法不够严谨，正确的说法是，距离**view的相对view的距离**才更准确。

在Linearlayout下。能够觉得是距离父view的距离。但在RelativeLayout下则不然，假设view A已经写定在view B的右側，则view A的layout\_marginLeft就是距离view B的距离，与父View无关。

3dp起作用





Handler内部跟Looper进行关联，Handler的构造方法会默认取当前线程的Looper。Handler找到了Looper也就是找到了MessageQueue，在handler中发消息，就是在想MessageQueue发送消息

总结一下：Handler负责发送消息，Looper负责接受Hander发送的消息，并直接把消息回传给hander自己，MessageQueue就是一个存储消息的容器。

<https://blog.csdn.net/reakingf/article/details/52054598>

Handler() 默认构造方法，与当前线程及其Looper实例绑定。如在主线程中执行new Handler()，那么该handler实例所绑定的便是 UI 线程和 UI 线程绑定的Looper实例。

Handler的作用，它允许我们将Message或Runnable对象发送到当前线程绑定的MessageQueue中，并通过Looper对象不断循环地从队列中获取Message或Runnable对象进行处理。因此，Handler有两个主要的用途:

1. 定时执行messages 和 runnables；
2. 在将一个action入队并在其他线程中执行；

Looper是线程用来运行消息循环(message loop)的类。默认情况下，线程并没有与之关联的Looper，可以通过在线程中调用Looper.prepare() 方法来获取，并通过Looper.loop() 无限循环地获取并分发MessageQueue中的消息

One of the plugins you are using supports Java 8 language features. To try the support built into the Android plugin, remove the following from your build.gradle: apply plugin: 'me.tatarka.retrolambda'

Project build.gradle

移除

dependencies {  
 classpath **'com.android.tools.build:gradle:3.1.2'** *//classpath 'me.tatarka:gradle-retrolambda:3.2.5'*

module build.gradle

移除

*//apply plugin: 'me.tatarka.retrolambda'*

@Inject 注解就如同一个标签，或者说它是一个记号，它是给 Dagger2 看的。它运用的地方有两处。

1. @Inject 给一个类的相应的属性做标记时，说明了它是一个依赖需求方，需要一些依赖。
2. @Inject 给一个类的构造方法进行注解时，表明了它能提供依赖的能力。

ObjectAnimator animate=ObjectAnimator.*ofFloat*(**btnTrans**,**"translationX"**,0,100);

平移前

V/getTranslationX: 0.0

V/getX: 142.0

V/getLeft: 142

平移后

V/getTranslationX: 100.0

V/getX: 242.0

V/getLeft: 142

大多数的情况使用ObjectAnimator就足够了，因为它使得目标对象动画值的处理过程变得足够简单，不用像ValueAnimator那样自己写动画更新的逻辑，

我们通常使用ObjectAnimator设置View已知的属性来生成动画，而一般View已知属性变化时都会主动触发重绘图操作，所以动画会自动实现；但是也有特殊情况，譬如作用Object不是View，或者作用的属性没有触发重绘，或者我们在重绘时需要做自己的操作

# Android常用对话框大全——Dialog

<https://blog.csdn.net/a_zhon/article/details/54578047>

列表对话框

.setSingleChoiceItems(items, 1, new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

Toast.makeText(MainActivity.this, items[which], Toast.LENGTH\_SHORT).show();

}

})

多选列表啦

.setMultiChoiceItems(items, checkedItems, new DialogInterface.OnMultiChoiceClickListener() {

@Override

public void onClick(DialogInterface dialog, int which, boolean isChecked) {

checkedItems[which] = isChecked;

}

})

源码为我们提供了一个.setView()函数，这样我们就可以自定义对话框显示的内容

View view = getLayoutInflater().inflate(R.layout.half\_dialog\_view, null);

final EditText editText = (EditText) view.findViewById(R.id.dialog\_edit);

.setView(view)

上面代码中half\_dialog\_view.xml中我就放置了一个EditText;在这里好多人在找自己布局中的控件时候经常报NullpointException，原因也很简单就是没有使用加载的布局.findViewbyId()。

还想把他的标题或者底部按钮给改了，那么就需要来个完全自定义了，如下：

1. 首先我们需要自定义Dialog的style，让他自己本有的东西全部透明，然后在设置我们自己的内容就可以达到完全自定义的效果了。

Variable is accessed within inner class. Needs to be declared final

java中规定，内部类只能访问外部类中的成员变量，不能访问方法中定义的变量，如果要访问方法中的变量，就要把方法中的变量声明为final（常量）的，因为这样可以使变量全局化，就相当于是在外部定义的而不是在方法里定义的

Any variable defined in a method and accessed by an anonymous inner class must be final. Otherwise, you could use that variable in the inner class, unaware that if the variable changes in the inner class, and then it is used later in the enclosing scope, the changes made in the inner class did not persist in the enclosing scope. Basically, what happens in the inner class stays in the inner class.

# Android 布局优化之include与merge

<https://blog.csdn.net/a740169405/article/details/50473909>