资源文件夹

1）引用自定义的资源

android:text="@string/hello"

这里使用"@"前缀引入对一个资源的引用--在**@[package:]type/name形式**中后面的文本是资源的名称。在这种情况下，我们不需要指定包名，因为我们引用的是我们自己包中的资源。type是xml子节点名，name是xml属性名：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, HelloDemo!</string>
</resources>
```

2）引用系统资源

android:textColor="@**android:**color/opaque_red" 指定package: android

3）引用主题属性

另外一种资源值允许你引用当前主题中的属性的值。这个属性值只能在样式资源和XML属性中使用；它允许你通过将它们改变为当前主题提供的标准变化来改变UI元素的外观，而不是提供具体的值。

android:textColor="**?**android:textDisabledColor"

注意:这和资源引用非常类似，除了我们使用一个"?"前缀代替了"@"。当你使用这个标记时，你就提供了属性资源的名称，它将会在主题中被查找--因为资源工具知道需要的属性资源，所以你不需要显示声明这个类型(如果声明，其形式就是?android:attr/android:textDisabledColor)。除了使用这个资源的标识符来查询主题中的值代替原始的资源，其命名语法和"@"形式一致：?[namespace:]type/name，这里类型可选。

res/anim/

XML文件，它们被编译进逐帧动画（frame by frame animation）或补间动画(tweened animation)对象

动画资源分为两种，一是实现图片的translate、scale、rotate、alpha四种变化。还可以设置动画的播放特性；
另一种是帧动画，逐帧播放设置的资源
路径位于：res/anim/filename.xml，文件名随意

动画类型

Android的animation由四种类型组成

XML中

alpha	渐变透明度动画效果
scale	渐变尺寸伸缩动画效果
translate	画面转换位置移动动画效果
rotate	画面转移旋转动画效果

JavaCode中

AlphaAnimation	渐变透明度动画效果
ScaleAnimation	渐变尺寸伸缩动画效果
TranslateAnimation	画面转换位置移动动画效果

RotateAnimation 画面转移旋转动画效果

Animation主要有两种动画模式

一种是tweened animation(渐变动画)

XML中	JavaCode
alpha	AlphaAnimation
scale	ScaleAnimation

一种是frame by frame(画面转换动画)

XML中	JavaCode
translate	TranslateAnimation
rotate	RotateAnimation

alpha xml 淡出效果

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
<alpha
    android:fromAlpha="1.0"
    android:toAlpha="0.0"
    android:duration="500"  />
</set>
<!--
    fromAlpha:开始时透明度
    toAlpha:  结束时透明度
    duration: 动画持续时间 -->
```

alpha xml 淡入效果

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
<alpha
    android:fromAlpha="0.0"
    android:toAlpha="1.0"
    android:duration="500"  />
</set>
<!--
    fromAlpha:开始时透明度
    toAlpha:  结束时透明度
    duration: 动画持续时间 -->
```

rotate.xml 旋转效果

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
<rotate
    android:interpolator="@android:anim/accelerate_decelerate_interpolator"
    android:fromDegrees="300"
    android:toDegrees="-360"
    android:pivotX="10%"
    android:pivotY="100%"
    android:duration="10000" />
</set>
<!--
  fromDegrees   动画开始时的角度
  toDegrees     动画结束时物件的旋转角度,正代表顺时针
  pivotX    属性为动画相对于物件的X坐标的开始位置
  pivotY    属性为动画相对于物件的Y坐标的开始位置    -->
```

Scale.xml 缩放效果

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
<scale
    android:interpolator= "@android:anim/decelerate_interpolator"
    android:fromXScale="0.0"
    android:toXScale="1.5"
    android:fromYScale="0.0"
    android:toYScale="1.5"
    android:pivotX="50%"
    android:pivotY="50%"
    android:startOffset="0"
    android:duration="10000"
    android:repeatCount="1"
    android:repeatMode="reverse"  />
</set>

<!--
fromXDelta,fromYDelta       起始时X，Y座标,屏幕右下角的座标是X:320,Y:480
toXDelta，  toYDelta      动画结束时X,Y的座标 --> <!--
interpolator                    指定动画插入器
常见的有加速减速插入器          accelerate_decelerate_interpolator
加速插入器                accelerate_interpolator，
减速插入器                decelerate_interpolator。
fromXScale,fromYScale，          动画开始前X,Y的缩放，0.0为不显示，  1.0为正常大小
toXScale，toYScale，           动画最终缩放的倍数，  1.0为正常大小，大于1.0放大
pivotX，   pivotY          动画起始位置，相对于屏幕的百分比,两个都为50%表示动画从屏幕中间开始
startOffset，                   动画多次执行的间隔时间，如果只执行一次，执行前会暂停这段时间，
                    单位毫秒 duration，一次动画效果消耗的时间，单位毫秒，
                    值越小动画速度越快 repeatCount，动画重复的计数，动画将会执行该值+1次
                    repeatMode，动画重复的模式，reverse为反向，当第偶次执行时，动画方向会相反。
                    restart为重新执行，方向不变 -->
```

translate.xml 移动效果

```xml
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
<translate
    android:fromXDelta="320"
    android:toXDelta="0"
    android:fromYDelta="480"
    android:toYDelta="0"
    android:duration="10000" />
</set>
<!--
fromXDelta,fromYDelta   起始时X，Y座标,屏幕右下角的座标是X:320,Y:480
toXDelta， toYDelta     动画结束时X,Y的座标 -->
```

res/drawable

png、.9.png、.jpg文件，它们被编译进以下的Drawable资源子类型中：

要获得这种类型的一个资源，可以使用Resource.getDrawable(*id*)

位图文件

9-patches（可变尺寸的位图）

为了获取资源类型，使用mContext.getResources().getDrawable(R.drawable.imageId)

图片资源一般使用png格式，使用其他格式的会出现各种问题，貌似不支持gif格式的图片，可是使用Movie来播放gif格式的图片

路径位于：res/drawable

定义格式：可以直接存放图片也可以是xml等配置文件（一般用于自定义组件）

为了获取资源类型，使用mContext.getResources().getDrawable(R.drawable.imageId)

注意：放在这里的图像资源可能会被aapt工具自动地进行无损压缩优化。比如，一个真彩色但并不需要256色的PNG可能会被转换为一个带调色板的8位PNG。这使得同等质量的图片占用更少的资源。所以我们得意识到这些放在该目录下的二进制图像在生成时可能会发生变化。如果你想读取一个图像位流并转换成一个位图(bitmap)，请把图像文件放在res/raw/目录下，这样可以避免被自动优化。

animated-rotate Drawable

animation-list Drawable

帧动画，逐帧播放设置的资源，称为Frame动画

android:oneshot属性代表运行次数，如果为true为只运行一次，false代表运行多次

对应AnimantionDrawable方法中的setOneShot,
android:duration，为显示的时间长度，

对应AnimationDrawable中的addFrame(drawable,duration)方法

```xml
<animation-list xmlns:android="http://schemas.android.com/apk/res/android"
android:oneshot="true">
    <item android:drawable="@drawable/rocket_thrust1" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust2" android:duration="200" />
    <item android:drawable="@drawable/rocket_thrust3" android:duration="200" />
</animation-list>
```

```
//java代码中动画设置代码
 private void animate()
 {
    ImageView view=(ImageView)findViewById(R.id.animationImage);
    view.setVisibility(ImageView.VISIBLE);
    //将自定义动画文件设置imageview背景
    view.setBackgroundResource(R.drawable.frame_animation);
    AnimationDrawable frameAnimation=(AnimationDrawable)view.getBackground();

    if(frameAnimation.isRunning())
    {
        frameAnimation.stop();
    }
    Else
    {
        frameAnimation.stop();
        frameAnimation.start();
    }
 }
```

Bitmap Drawable

1.Android supports bitmap files in a three formats:

.png(preferred), jpg(acceptable), .gif(discouraged)

2.Reference a bitmap way:

a.reference a bitmap file directly
b.resource ID
c.an alias resource ID in XML

Note:

Bitmap files may be automatically optimized with lossless image compression by the aapt tool.
If you plan on reading an image as a bit stream in order to convert it to a bitmap, put your images in the res/raw/ folder instead, where they will not be optimized.

3.file location:

res/drawable/filename.png (.png, .jpg, or .gif)
The filename is used as the resource ID.

4.compiled resource datatype:

Resource pointer to a BitmapDrawable.

5.resource reference:

In Java: R.drawable.filename
In XML: @[package:]drawable/filename

6.example:

a.An image saved at res/drawable/myimage.png

b.layout XML applies the image to a View:

```
<ImageView
    android:layout_height="wrap_content"
```

```
    android:layout_width="wrap_content"
    android:src="@drawable/myimage" />
```

c.Coding

```
Resources res = getResources();
Drawable drawable = res.getDrawable(R.drawable.myimage);
```

Clip Drawable

A drawable defined in XML that clips another drawable based on this Drawable's current Level. Most often used to implement things like progress bars.

1.file location:

res/drawable/filename.xml
The filename is used as the resource ID.

2.compiled resource datatype:

Resource pointer to a ClipDrawable.

3.resource reference:

In Java: R.drawable.filename
In XML: @[package:]drawable/filename

4.syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<clip
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/drawable_resource"
    android:clipOrientation=["horizontal" | "vertical"]
    android:gravity=["top" | "bottom" | "left" | "right" | "center_vertical" |
                "fill_vertical" | "center_horizontal" | "fill_horizontal" |
                "center" | "fill" | "clip_vertical" | "clip_horizontal"] />
```

5.example:

a.define clip drawable in XML file -> res/drawable/clip.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/android"
    android:clipOrientation="horizontal"
    android:gravity="left" />
</shape>
```

b.Apply the clipdrawable to a View in XML layout

```
<ImageView
  android:id="@+id/image"
  android:background="@drawable/clip"
  android:layout_height="wrap_content"
  android:layout_width="wrap_content" />
```

c.Apply the clipdrawable in Java Code -> progressively reveal the image

```
ImageView imageview = (ImageView) findViewById(R.id.image);
ClipDrawable drawable = (ClipDrawable) imageview.getDrawable();
drawable.setLevel(drawable.getLevel() + 1000);
```

//10,000 present that the image is not clipped and comletely visible

Color Drawable

Inset Drawable

A drawable defined in XML that insets another drawable by a specified distance. This is used when a View needs a background that is smaller than the View's actual bounds.

1.file location:

res/drawable/filename.xml
The filename is used as the resource ID.

2.compiled resource datatype:

Resource pointer to a InsetDrawable.

3.resource reference:

In Java: R.drawable.filename
In XML: @[package:]drawable/filename

4.syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<inset
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/drawable_resource"
    android:insetTop="dimension"
    android:insetRight="dimension"
    android:insetBottom="dimension"
    android:insetLeft="dimension" />
```

5.example:

```
<?xml version="1.0" encoding="utf-8"?>
<inset xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/background"
    android:insetTop="10dp"
    android:insetLeft="10dp" />
```

nine-patch Drawable

1.format

A NinePatch is a PNG image in which you can define stretchable regions that Android scales when content within the View exceeds the normal image bounds.

2.Reference Nine-Patch File

a.reference a bitmap file directly
b.resource ID
c.an alias resource ID in XML

3.file location:

res/drawable/filename.9.png
The filename is used as the resource ID.

4.compiled resource datatype:

Resource pointer to a NinePatchDrawable.

5.resource reference:

In Java: R.drawable.filename

In XML: @[package:]drawable/filename

6.example:

a.An image saved at res/drawable/myninepatch.9.png

b.Layout XML applies the Nine-Patch to a View:

```
<Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:background="@drawable/myninepatch" />
```

7.An alias Bitmap resource

file location:

res/drawable/filename.xml

The filename is used as the resource ID.

compiled resource datatype:

Resource pointer to a NinePatchDrawable.

resource reference:

In Java: R.drawable.filename

In XML: @[package:]drawable/filename

syntax:

```
<?xml version="1.0" encoding="utf-8"?>
<nine-patch
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@[package:]drawable/drawable_resource"
    android:dither=["true" | "false"] />
```

example:

```
<?xml version="1.0" encoding="utf-8"?>
<nine-patch xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@drawable/myninepatch"
    android:dither="false" />
```

Rotate Drawable

Scale Drawable

A drawable defined in XML that changes the size of another drawable based on its current level.

1.file location:

res/drawable/filename.xml

The filename is used as the resource ID.

2.compiled resource datatype:

Resource pointer to a ScaleDrawable.

3.resource reference:

In Java: R.drawable.filename

In XML: @[package:]drawable/filename

4.syntax:

```
<?xml version="1.0" encoding="utf-8"?>
 <scale
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/drawable_resource"
    android:scaleGravity=["top" | "bottom" | "left" | "right" | "center_vertical" |
                          "fill_vertical" | "center_horizontal" | "fill_horizontal" |
                          "center" | "fill" | "clip_vertical" | "clip_horizontal"]
    android:scaleHeight="percentage"
    android:scaleWidth="percentage" />
```

5.example:

```
 <?xml version="1.0" encoding="utf-8"?>
 <scale xmlns:android="http://schemas.android.com/apk/res/android"
    android:drawable="@drawable/logo"
    android:scaleGravity="center_vertical|center_horizontal"
    android:scaleHeight="80%"
    android:scaleWidth="80%" />
```

Shape Drawable

It is a generic shape defined in XML.

1.file location:

res/drawable/filename.xml
The filename is used as the resource ID.

2.compiled resource datatype:

Resource pointer to a ShapeDrawable.

3.resource reference:

In Java: R.drawable.filename
In XML: @[package:]drawable/filename

4.syntax:

```
<?xml version="1.0" encoding="utf-8"?>
 <shape
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape=["rectangle" | "oval" | "line" | "ring"] >
    <corners
       android:radius="integer"
       android:topLeftRadius="integer"
       android:topRightRadius="integer"
       android:bottomLeftRadius="integer"
       android:bottomRightRadius="integer" />
    <gradient
       android:angle="integer"
       android:centerX="integer"
       android:centerY="integer"
       android:centerColor="integer"
       android:endColor="color"
```

```
        android:gradientRadius="integer"
        android:startColor="color"
        android:type=["linear" | "radial" | "sweep"]
        android:usesLevel=["true" | "false"] />
    <padding
        android:left="integer"
        android:top="integer"
       android:right="integer"
        android:bottom="integer" />
    <size
        android:width="integer"
        android:color="color"
        android:dashWidth="integer"
        android:dashGap="integer" />
    <solid
        android:color="color" />
    <stroke
        android:width="integer"
        android:color="color"
        android:dashWidth="integer"
        android:dashGap="integer" />
</shape>
```

5.example:

在开发程序时为了兼容不同平台不同屏幕，建议各自文件夹根据需求均存放不同版本图片。
我们可以将已经做好的图片放到该目录下，或者通过自定义XML文件来实现想要的图片，例如我们可以定义shapge_1.xml放到drawable目录下，内容如下：

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"   android:shape="oval">
<!--android:shape="oval"表示所要绘制的图形是一个椭圆，默认是rectangle，长方形-->
<gradient
    android:startColor="#0055ff88"
    android:centerColor="#0055ff00"
    android:centerY="0.75"
    android:endColor="#00320077"
    android:angle="270"
/>
<!--gradient 产生颜色渐变 android:angle 从哪个角度开始变 只有90的整数倍可以 -->
<solid android:color="#ff4100ff"/>
<!--solid表示图形是实心的，填充里面，#ff4100ff为填充颜色-->
<stroke
    android:width="2dp"
    android:color="#ee31ff5e"
    android:dashWidth="3dp"
    android:dashGap="2dp" />
<!-- 描边 采用那样的方式将外形轮廓线画出来，width表示笔的粗细，dashWidth表示小横线的宽度，dashGap表示小横线之间的距离-->
<padding
    android:left="7dp"
    android:top="7dp"
    android:right="7dp"
    android:bottom="7dp" />
<!--和CSS中的padding应该是一个道理-->
<corners android:radius="6dp" />
<!--corners表示是有半径为5像素的圆角-->
</shape>
```

Apply this shape drawable to a Button

```
Resources res = getResources();
Drawable shape = res. getDrawable(R.drawable.shapge_1.xml);

Button button = (Button)findViewByID(R.id.button);
button.setBackground(shape);
```

Selector Drawable

根据状态改变图片显示效果

属性：

android:state_focused

android:state_window_focused

android:state_enabled

android:state_checkable

android:state_checked

android:state_selected

android:state_active

android:state_single

android:state_first

android:state_middle

android:state_last

android:state_pressed

简单示例：

1. 先定义一个名为btnselector.xml文件，代码如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
<item
    android:state_focused="true"
    android:state_pressed="false"
    android:drawable="@drawable/focused"/>
<item
    android:state_focused="true"
    android:state_pressed="true"
    android:drawable="@drawable/focusedpressed"/>
<item
    android:state_focused="false"
    android:state_pressed="true"
    android:drawable="@drawable/pressed"/>
<item
```

```
    android:drawable="@drawable/default"/>

</selector>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item  android:state_pressed="false"android:drawable="@drawable/xxx1" />
    <item  android:state_pressed="true"  android:drawable="@drawable/xxx2" />
    <item  android:state_focused="true"  android:drawable="@drawable/xxx3" />
    <-- 这里还可以加N多效果和动作 只要你用的到 -->
    <item  android:drawable="@drawable/xxx4" />
</selector>
```

2. ImageButton使用btnselector.xml如下：

```
<ImageButton
    android:id="@+id/stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/btnselector"
    android:background="#00000000"/>
```

注意：android:src(android:background)赋值为"@drawable/btnselector"，而不是指向具体的图片资源。

res/layout

XML 文件，被编译成屏幕布局

res/values

可以被编译成很多种类型的资源的XML文件。

注意:不像其他的res/文件夹，它可以保存任意数量的文件，这些文件保存了要创建资源的描述，而不是资源本身。XML元素类型控制这些资源应该放在R类的什么地方。

尽管这个文件夹里的文件可以任意命名，不过下面使一些比较典型的文件（文件命名的惯例是将元素类型包含在该名称之中）：

array.xml 定义数组

Colors.xml定义color drawable和颜色的字符串值(color string values)。使用Resource.getDrawable()和Resources.getColor()分别获得这些资源。

dimens.xml定义尺寸值(dimension value)。使用Resources.getDimension()获得这些资源。

strings.xml定义字符串(string)值。使用Resources.getString()或者Resources.getText()获取这些资源。getText()会保留在UI字符串上应用的丰富的文本样式。

styles.xml 定义样式(style)对象

Array xml

字符串数组

```
arrays.xml如下:

<?xml version="1.0" encoding="utf-8"?>
<resources>

        <string-array name="feed_names">
                <item>新闻</item>
                <item>视频</item>
                <item>国际新闻</item>
                <item>体育</item>
                <item>艺术</item>
                <item>餐饮</item>
        </string-array>
</resources>
```

引用方式: R.array.feed_names

整形数组

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
 <integer-array name="values">
     <item >1</item>
     <item >2</item>
     <item >3</item>
     <item >4</item>
     <item >5</item>
    </integer-array>
</resources
```

Color xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
         <color name="white">#FFFFFF</color> -<!-- 白色 -->
    <color name="ivory">#FFFFF0</color> -<!-- 象牙色 -->
    <color name="lightyellow">#FFFFE0</color> -<!-- 亮黄色 -->
    <color name="yellow">#FFFF00</color> -<!-- 黄色 -->
    <color name="snow">#FFFAFA</color> -<!-- 雪白色 -->
    <color name="floralwhite">#FFFAF0</color> - <!-- 花白色 -->
    <color name="lemonchiffon">#FFFACD</color> - <!-- 柠檬绸色 -->
    <color name="cornsilk">#FFF8DC</color> - <!-- 米绸色 -->
    <color name="seashell">#FFF5EE</color> - <!-- 海贝色 -->
    <color name="lavenderblush">#FFF0F5</color> - <!-- 淡紫红 -->
    <color name="papayawhip">#FFEFD5</color> - <!-- 番木色 -->
    <color name="blanchedalmond">#FFEBCD</color> - <!-- 白杏色 -->
    <color name="mistyrose">#FFE4E1</color> - <!-- 浅玫瑰色 -->
    <color name="bisque">#FFE4C4</color> - <!-- 桔黄色 -->
    <color name="moccasin">#FFE4B5</color> - <!-- 鹿皮色 -->
    <color name="navajowhite">#FFDEAD</color> - <!-- 纳瓦白 -->
    <color name="peachpuff">#FFDAB9</color> - <!-- 桃色 -->
    <color name="gold">#FFD700</color> - <!-- 金色 -->
    <color name="pink">#FFC0CB</color> - <!-- 粉红色 -->
    <color name="lightpink">#FFB6C1</color> - <!-- 亮粉红色 -->
    <color name="orange">#FFA500</color> - <!-- 橙色 -->
    <color name="lightsalmon">#FFA07A</color> - <!-- 亮肉色 -->
    <color name="darkorange">#FF8C00</color> - <!-- 暗桔黄色 -->
    <color name="coral">#FF7F50</color> - <!-- 珊瑚色 -->
    <color name="hotpink">#FF69B4</color> - <!-- 热粉红色 -->
    <color name="tomato">#FF6347</color> - <!-- 西红柿色 -->
</resources
```

引用方式: R.clor.white

语法： <color name="color_name">#color_value</color> 可以保存在 res/values/colors.xml (文件名可以任意)。

xml引用： android:textColor="@color/color_name"

Java引用： int color = Resources.getColor(R.color.color_name)

其中#color_value有以下格式（A代表Alpha通道）：

#RGB

#ARGB

#RRGGBB

#AARRGGBB

Drawable xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <drawable name="white">#FFFFFF</drawable>
    <drawable name="black">#000000</drawable>
</resources
```

引用方式: R.drawable.white

语法： <drawable name="color_name">color_value</drawable> 可以保存在res/values/colors.xml。

xml引用 ：android:background="@drawable/color_name"

java引用：Drawable redDrawable = Resources.getDrawable(R.drawable.color_name)

color_name和上面的一样。个人认为，一般情况下使用color属性，当需要用到paintDrawable时才使用drawable属性。

String xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World!</string>
    <string name="app_name">(*^__^*) 嘻嘻……</string>
</resources
```

引用方式:R.stirng.hello

Dimension xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="one_pixel">1px</dimen>
    <dimen name="double_density">2dp</dimen>
    <dimen name="sixteen_sp">16sp</dimen>
</resources>
```

引用方式:R.dimen.one_pixel

语法：<dimen name="dimen_name">dimen_value单位</dimen>

一般保存为res/values/dimen.xml。

度量单位：

px(象素): 屏幕实际的象素，常说的分辨率1024*768pixels，就是横向1024px，纵向768px，不同设备显示效果相同。

in(英寸): 屏幕的物理尺寸，每英寸等于2.54厘米。

mm(毫米):屏幕的物理尺寸。

pt(点) ：屏幕的物理尺寸。1/72英寸。

dp/dip ：与密度无关的象素，一种基于屏幕密度的抽象单位。在每英寸160点的显示器上，1dp = 1px。但dp和px的比例会随着屏幕密度的变化而改变，不同设备有不同的显示效果。

sp ：与刻度无关的象素，主要用于字体显示best for textsize，作为和文字相关大小单位.

Style xml

定义样式的文件，分为两种用途：

Style：以一个单位的方式用在布局XML单个元素（控件）当中。 例如：我们可以为TextView定义一种样式风格，包含文本的字号大小和颜色，然后将其用在TextView特定的实例。

Theme：以一个单位的方式用在应用中所有的Activity当中或者应用中的某个Activity当中。 比如，我们可以定义一个Theme，它为window frame和panel 的前景和背景定义了一组颜色，并为菜单定义可文字的大小和颜色属性，可以将这个Theme应用在你程序当中所有的Activity里。

系统 Animation style

```
public static final int Animation = 16973824;
public static final int Animation_Activity = 16973825;
public static final int Animation_Dialog = 16973826;
```

```
public static final int Animation_InputMethod = 16973910;
public static final int Animation_Toast = 16973828;
public static final int Animation_Translucent = 16973827;
```

系统 MediaButton style

```
//Media Button 图标

public static final int MediaButton = 16973879;
public static final int MediaButton_Ffwd = 16973883;
public static final int MediaButton_Next = 16973881;
public static final int MediaButton_Pause = 16973885;
public static final int MediaButton_Play = 16973882;
public static final int MediaButton_Previous = 16973880;
public static final int MediaButton_Rew = 16973884;
```

系统 TextAppearance Style

```
public static final int TextAppearance = 16973886;
public static final int TextAppearance_DialogWindowTitle = 16973889;
public static final int TextAppearance_Inverse = 16973887;
public static final int TextAppearance_Large = 16973890;
public static final int TextAppearance_Large_Inverse = 16973891;I
public static final int TextAppearance_Medium = 16973892;
public static final int TextAppearance_Medium_Inverse = 16973893;
public static final int TextAppearance_Small = 16973894;
public static final int TextAppearance_Small_Inverse = 16973895;
public static final int TextAppearance_Theme = 16973888;
public static final int TextAppearance_Theme_Dialog = 16973896;
public static final int TextAppearance_Widget = 16973897;I
public static final int TextAppearance_Widget_Button = 16973898;I
public static final int TextAppearance_Widget_DropDownHint = 16973904;
public static final int TextAppearance_Widget_DropDownItem = 16973905;
public static final int TextAppearance_Widget_EditText = 16973900;
public static final int TextAppearance_Widget_IconMenu_Item = 16973899;
public static final int TextAppearance_Widget_TabWidget = 16973901;
public static final int TextAppearance_Widget_TextView = 16973902;
public static final int TextAppearance_Widget_TextView_PopupMenu = 16973903;
public static final int TextAppearance_Widget_TextView_SpinnerItem = 16973906;
public static final int TextAppearance_WindowTitle = 16973907;
```

系统 Theme style

```
在 AndroidManifest.xml 的 Activity 声明中加入属性 ：

android:theme="@android:style/Theme.NoTitleBar.Fullscreen"

public static final int Theme = 16973829;
public static final int Theme_Black = 16973832;
public static final int Theme_Black_NoTitleBar = 16973833;
```

```
public static final int Theme_Black_NoTitleBar_Fullscreen = 16973834;
public static final int Theme_Dialog = 16973835;
public static final int Theme_InputMethod = 16973908;
public static final int Theme_Light = 16973836;I
public static final int Theme_Light_NoTitleBar = 16973837;
public static final int Theme_Light_NoTitleBar_Fullscreen = 16973838;
public static final int Theme_Light_Panel = 16973914;
public static final int Theme_Light_WallpaperSettings = 16973922;
public static final int Theme_NoDisplay = 16973909;
public static final int Theme_NoTitleBar = 16973830;
public static final int Theme_NoTitleBar_Fullscreen = 16973831;
public static final int Theme_Panel = 16973913;
public static final int Theme_Translucent = 16973839;
public static final int Theme_Translucent_NoTitleBar = 16973840;
public static final int Theme_Translucent_NoTitleBar_Fullscreen = 16973841;
public static final int Theme_Wallpaper = 16973918;
public static final int Theme_Wallpaper_NoTitleBar = 16973919;
public static final int Theme_Wallpaper_NoTitleBar_Fullscreen = 16973920;
public static final int Theme_WallpaperSettings = 16973921;
```

系统 Widget Style

```
public static final int Widget = 16973842;
public static final int Widget_AbsListView = 16973843;
public static final int Widget_AutoCompleteTextView = 16973863;
public static final int Widget_Button = 16973844;
public static final int Widget_Button_Inset = 16973845;
public static final int Widget_Button_Small = 16973846;
public static final int Widget_Button_Toggle = 16973847;
public static final int Widget_CompoundButton = 16973848;
public static final int Widget_CompoundButton_CheckBox = 16973849;
public static final int Widget_CompoundButton_RadioButton = 16973850;
public static final int Widget_CompoundButton_Star = 16973851;
public static final int Widget_DropDownItem = 16973867;
public static final int Widget_DropDownItem_Spinner = 16973868;
public static final int Widget_EditText = 16973859;
public static final int Widget_ExpandableListView = 16973860;
public static final int Widget_Gallery = 16973877;
public static final int Widget_GridView = 16973874;
public static final int Widget_ImageButton = 16973862;
public static final int Widget_ImageWell = 16973861;
public static final int Widget_KeyboardView = 16973911;
public static final int Widget_ListView = 16973870;
public static final int Widget_ListView_DropDown = 16973872;
public static final int Widget_ListView_Menu = 16973873;
public static final int Widget_ListView_White = 16973871;
public static final int Widget_PopupWindow = 16973878;
public static final int Widget_ProgressBar = 16973852;
public static final int Widget_ProgressBar_Horizontal = 16973855;
public static final int Widget_ProgressBar_Inverse = 16973915;
public static final int Widget_ProgressBar_Large = 16973853;
public static final int Widget_ProgressBar_Large_Inverse = 16973916;
public static final int Widget_ProgressBar_Small = 16973854;
public static final int Widget_ProgressBar_Small_Inverse = 16973917;
```

public static final int Widget_RatingBar = 16973857;
public static final int Widget_ScrollView = 16973869;
public static final int Widget_SeekBar = 16973856;
public static final int Widget_Spinner = 16973864;
public static final int Widget_TabWidget = 16973876;
public static final int Widget_TextView = 16973858;
public static final int Widget_TextView_PopupMenu = 16973865;
public static final int Widget_TextView_SpinnerItem = 16973866;
public static final int Widget_WebView = 16973875;

属性名称	描述
android:cacheColorHint	指示该列表总是在固定的单色、不透明的背景下绘制。这允许列表优化其绘制过程
android:drawSelectorOnTop	如果设为真，选择器将绘制在选中条目的上层。否则绘制在下层。默认为假
android:fastScrollEnabled	允许使用快速滚动滑块，可以通过拖动该滑块在列表中快速滚动
android:listSelector	用于在列表中指示当前选中条目的可绘制对象
android:scrollingCache	当为真时，列表滚动使用绘图缓存。该选项使渲染更快，但占用更多的内存。 默认值为真
android:smoothScrollbar	为真时，列表会使用更精确的基于条目在屏幕上的可见像素高度的计算方法。 默认该属性为真，如果你的适配器需要绘制可变高的条目，他应该设为假。 当该属性为真时，你在适配器在显示变高条目时，滚动条的把手会在滚动的 过程中改变大小。当设为假时，列表只使用适配器中的条目数和屏幕上的 可见条目来决定滚动条的属性
android:stackFromBottom	用于 ListView 和 GridView，指示他们的内容栈从底部开始
android:textFilterEnabled	设为真时，列表会过滤根据用户的要求，过滤结果集。列表的适配器必须实现了 Filterable 接口，才能使其可用
android:transcriptMode	设置列表的跳转模式。在跳转模式下，当加入新条目时，列表会滚动到底部， 使新条目可见

Style Items 信息详解

AbsListView 属性

android:choiceMode		Defines the choice behavior for the view.

属性名称	描述
android:cacheColorHint	指示该列表总是在固定的单色、不透明的背景下绘制。这允许列表优化其绘制过程
android:drawSelectorOnTop	如果设为真，选择器将绘制在选中条目的上层。否则绘制在下层。默认为假
android:fastScrollEnabled	允许使用快速滚动滑块，可以通过拖动该滑块在列表中快速滚动
android:listSelector	用于在列表中指示当前选中条目的可绘制对象
android:scrollingCache	当为真时，列表滚动使用绘图缓存。该选项使渲染更快，但占用更多的内存。 默认值为真
android:smoothScrollbar	为真时，列表会使用更精确的基于条目在屏幕上的可见像素高度的计算方法。 默认该属性为真，如果你的适配器需要绘制可变高的条目，他应该设为假。 当该属性为真时，你在适配器在显示变高条目时，滚动条的把手会在滚动的 过程中改变大小。当设为假时，列表只使用适配器中的条目数和屏幕上的 可见条目来决定滚动条的属性
android:stackFromBottom	用于 ListView 和 GridView，指示他们的内容栈从底部开始
android:textFilterEnabled	设为真时，列表会过滤根据用户的要求，过滤结果集。列表的适配器必须实现了 Filterable 接口，才能使其可用
android:transcriptMode	设置列表的跳转模式。在跳转模式下，当加入新条目时，列表会滚动到底部， 使新条目可见

ImageView 属性

```
<item name="android:adjustViewBounds">true</item>
```

是否保持宽高比。需要与 maxWidth、MaxHeight 一起使用，否则单独使用没有效果。

```
<item name="android:clipToPadding">true</item>
```

是否截取指定区域用空白代替。单独设置无效果，需要与scrollY一起使用，效果如下，实现代码见代码部分：

```
<ImageView
    android:background="@android:color/white"
    android:scrollY="-10px"
    android:cropToPadding="true"
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<ImageView
    android:background="@android:color/white"
```

```
    android:scrollY="10px"
    android:cropToPadding="true"
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/'>

<ImageView
    android:paddingTop="10px"
    android:background="@android:color/white"
    android:scrollY="10px"
    android:cropToPadding="true"
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<ImageView
    android:paddingTop="10px"
    android:background="@android:color/white"
    android:scrollY="10px"
    android:cropToPadding="false"
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

```
<item name="android:maxHeight"></item>
```

设置View的最大高度，单独使用无效，需要与setAdjustViewBounds一起使用。如果想设置图片固定大小，又想保持图片宽高比，需要如下设置：

1） 设置setAdjustViewBounds为true；

2） 设置maxWidth、MaxHeight；

3） 设置设置layout_width和layout_height为wrap_content。

```
<item name="android:maxWidth"></item>
```

设置View的最大宽度，同上

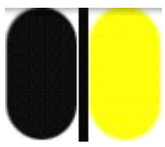
```
item name="android:scaleType">matrix </item>
```

设置图片的填充方式

matrix	0	用矩阵来绘图	
fitXY	1	拉伸图片（不按比例）以填充View的宽高 	layout_height :30px layout_width :120px
fitStart	2	按比例拉伸图片，拉伸后图片的高度为View的高度，且显示在View的左边 	
fitCenter	3	按比例拉伸图片，拉伸后图片的高度为View的高度，且显示在View的中间 	
fitEnd	4	按比例拉伸图片，拉伸后图片的高度为View的高度，且显示在View的右边 	
center	5	按原图大小显示图片，但图片宽高大于View的宽高时，截图图片中间部分显示 	layout_height :60px layout_width :80px padding :10px
centerCrop	6	按比例放大原图直至等于某边View的宽高显示。 	
centerInside	7	当原图宽高或等于View的宽高时，按原图大小居中显示；反之将原图缩放至View的宽高居中显示。 	

```
<item name="android:tint"></item>
```

将图片渲染成指定的颜色。见下图：



左边为原图，右边为设置后的效果，见后面代码。

```
<ImageView
    android:background="@android:color/white"
```

```
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<ImageView
    android:layout_marginLeft="5dp"
    android:background="@android:color/white"
    android:tint="#ffff00"
    android:src="@drawable/btn_mode_switch_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

```
<item name="android:src">@android:drawable/btn_minus</item>
```

设置View的drawable(如图片，也可以是颜色）@android:drawable/btn_minus为系统自带图片

View　　属性

```
<item name="android:background">@android:color/background_dark</item>
```

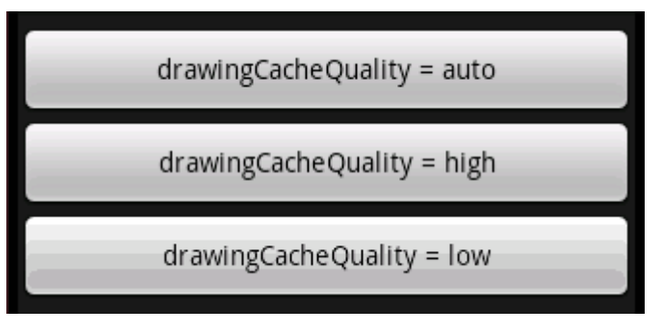
设置背景色/背景图片。可以通过以下两种方法设置背景为透明："@android:color/transparent"和"@null"。
注意 TextView 默认是透明的，不用写此属性，但是 Buttom/ImageButton/ImageView 想透明的话就得写这个属性了

```
<item name="android:clickable">true</item>
```
是否响应点击事件

```
<item name="android:contentDescription"></item>
```
设置View的备注说明，作为一种辅助功能提供, 为一些没有文字描述的View提供说明
如ImageButton。这里在界面上不会有效果，自己在程序中控制，可临时放一点字符串数据

```
<item name="android:drawingCacheQuality">auto</item>
```

设置绘图时半透明质量。有以下值可设置：auto（默认，由框架决定）/high（高质量，使用较高的颜色深度，消耗更多的内存/low（低质量，使用较低的颜色深度，但是用更少的内存）




```
<item name="android:duplicateParentState">true</item>
```
如果设置此属性，将直接从父容器中获取绘图状态（光标，按下等）。 见下面代码部分，注意根据目前测试情况仅仅是获取绘图状态，而没有获取事件，也就是你点一下 LinearLayout 时 Button 有被点击的效果，但是不执行点击事件

```
<LinearLayout
    android:clickable="true"
    android:background="#ff0fff"
    android:layout_width="100dp"
```

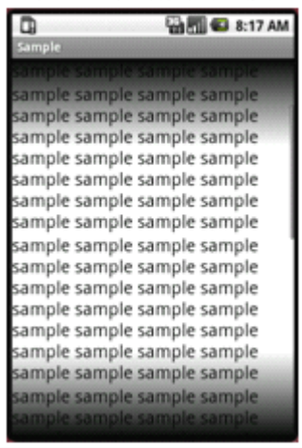
```
    android:layout_height="100dp">
<Button
    android:duplicateParentState="true"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
</LinearLayout>
```

```
<item name="android:fadingEdge">none</item>
```

设置拉滚动条时 ，边框渐变的放向。none（边框颜色不变）
horizontal（水平方向颜色变淡）
vertical（垂直方向颜色变淡）
参照fadingEdgeLength的效果图

```
<item name="android:fadingEdgeLength"></item>
```

设置边框渐变的长度




```
<item name="android:fitsSystemWindows">true</item>
```

设置布局调整时是否考虑系统窗口（如状态栏）

```
<item name="android:focusable">true</item>
```

设置是否获得焦点。若有requestFocus()被调用时，后者优先处理。注意在表单中想设置某一个如EditText获取焦点，光设置这个是不行的，需要将这个EditText前面的focusable都设置为false才行。在Touch模式下获取焦点需要设置focusableInTouchMode为true

```
<item name="android:focusableInTouchMode">true</item>
```

设置在Touch模式下View是否能取得焦点

```
<item name="android:hapticFeedbackEnabled">true</item>
```

设置触感反馈。（译者注：按软键以及进行某些 UI 交互时振动，暂时不知道用法，大家可以找找 performHapticFeedback 或 HapticFeedback 这个关键字的资料看看。）

```
<item name="android:id"></item>
```

给当前View设置一个在当前layout.xml中的唯一编号，可以通过调用View.findViewById() 或Activity.findViewById()根据这个编号查找到对应的View。不同的layout.xml之间定义相同的id不会冲突。格式如”@+id/btnName”

```
<item name="android:isScrollContainer">true</item>
```

设置当前View为滚动容器。这里没有测试出效果来
ListView/ GridView/ ScrollView根本就不用设置这个属性，而EdidText设置android:scrollbars也能出滚动条

<item name="android:keepScreenOn">true</item>

View在可见的情况下是否保持唤醒状态。常在LinearLayout使用该属性，但是模拟器这里没有效果

<item name="android:longClickable">true</item>

设置是否响应长按事件

<item name="android:minHeight"></item> 设置视图最小高度

<item name="android:minWidth"></item> 设置视图最小宽度

<item name="android:nextFocusDown"></item>

设置**下方**指定视图获得下一个焦点。焦点移动是基于一个在**给定方向**查找最近邻居的算法。如果指定视图不存在，移动焦点时将报运行时错误。可以设置imeOptions= actionDone，这样输入完即跳到下一个焦点

<item name="android:nextFocusLeft"></item> 设置**左边**指定视图获得下一个焦点

<item name="android:nextFocusRight"></item> 设置**右边**指定视图获得下一个焦点

<item name="android:nextFocusUp"></item> 设置**上边**指定视图获得下一个焦点

<item name="android:onClick"></item>

点击时从上下文中调用指定的方法。

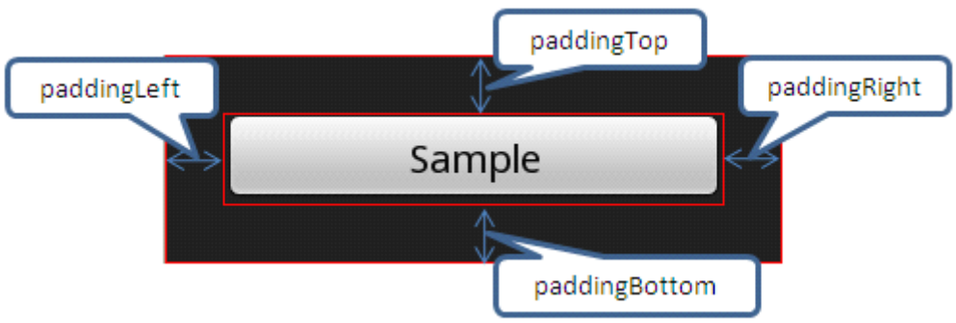
里指定一个方法名称，一般在Activity定义符合如下参数和返回值的函数并将方法名字符串指定为该值即可：

public void onClickButton(View view)

android:onClick="onClickButton"

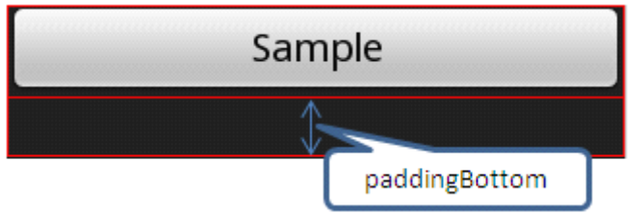
<item name="android:padding"></item>

设置上下左右的边距，以像素为单位填充空白



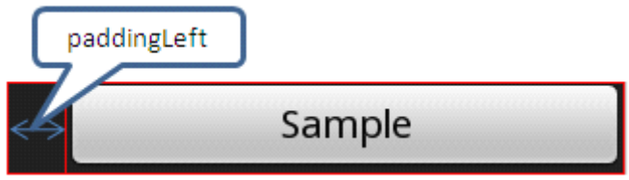
<item name="android:paddingBottom"></item>

设置底部的边距，以像素为单位填充空白




```
<item name="android:paddingLeft"></item>
```

设置左边的边距，以像素为单位填充空白

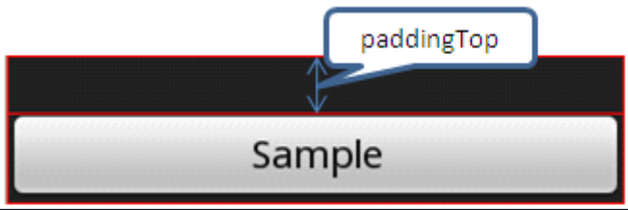



```
<item name="android:paddingRigth"></item>
```

设置右边的边距，以像素为单位填充空白


```
<item name="android:paddingTop"></item>
```

设置上方的边距，以像素为单位填充空白




```
<item name="android:saveEnabled">true</item>
```

设置是否在窗口冻结时（如旋转屏幕）保存View的数据，默认为true，但是前提是你需要设置id才能自动保存，参见这里。

```
<item name="android:scrollX"></item>
```

以像素为单位设置水平方向滚动的的偏移值，在GridView中可看的这个效果

```
<item name="android:scrollY"></item>
```

以像素为单位设置垂直方向滚动的的偏移值

```
<item name="android:scrollbarAlwaysDrawHorizontalTrack">true</item>
```

设置是否始终显示垂直滚动条。这里用ScrollView、ListView测试均没有效果

```
<item name="android:scrollbarAlwaysDrawVerticalTrack">true</item>
```

设置是否始终显示垂直滚动条。这里用ScrollView、ListView测试均没有效果

<item name="android:scrollbarDefaultDelayBeforeFade"></item>
设置N毫秒后开始淡化，以毫秒为单位

<item name="android:scrollbarFadeDuration"></item>
设置滚动条淡出效果（从有到慢慢的变淡直至消失）时间，以毫秒为单位

Android2.2中滚动条滚动完之后会消失，再滚动又会出来，在1.5、1.6版本里面会一直显示着

<item name="android:scrollbarSize"></item>　　设置滚动条的宽度

<item name="android:scrollbarStyle">insideOverlay</item>

设置滚动条的风格和位置

设置值：insideOverlay、insideInset、outsideOverlay、outsideInset

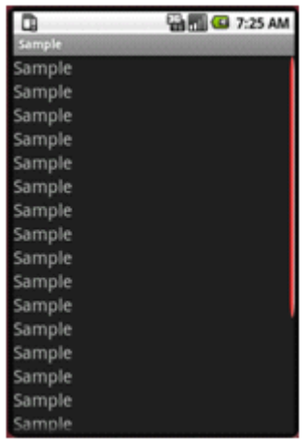
这里没有试出太多效果，以下依次是outsideOverlay与outsideInset效果截图比较：

<item name="android:scrollbarThumbHorizontal"></item>

设置水平滚动条的drawable

<item name="android:scrollbarThumbVertical"></item>

设置垂直滚动条的drawable.




```
<item name="android:scrollbarTrackHorizontal"></item>
```

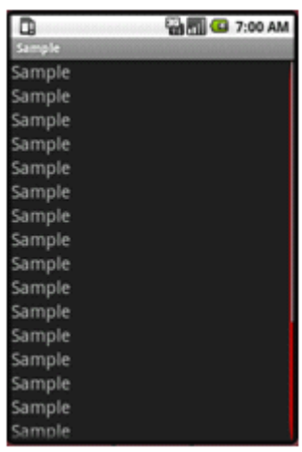
设置水平滚动条背景（轨迹）的色drawable


```
<item name="android:scrollbarTrackVertical"></item>
```

设置垂直滚动条背景（轨迹）的drawable注意直接设置颜色值如”android:color/white”将得出很难看的效果，甚至都不理解这个属性了

这里可以参见ApiDemos里res/drawable/ scrollbar_vertical_thumb.xml和scrollbar_vertical_track.xml

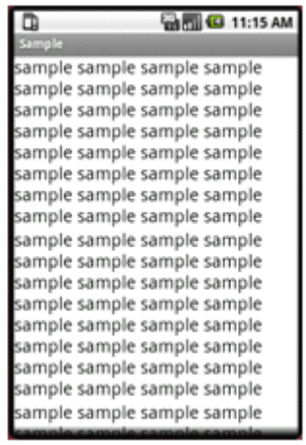
设置代码为：android:scrollbarTrackVertical ="@drawable/scrollbar_vertical_track"




```
<item name="android:scrollbars">none</item>
```

设置滚动条显示。none（隐藏），horizontal（水平），vertical（垂直）

见下列代码演示使用该属性让EditText内有滚动条。但是其他容器如LinearLayout设置了但是没有效果




```
<EditText
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:minHeight="50dp"
    android:background="@android:drawable/editbox_background"
    android:scrollbars="vertical"
    android:maxLines="4">
</EditText>
```

```
<item name="android:soundEffectsEnabled">true</item>
```
设置点击或触摸时是否有声音效果

```
<item name="android:tag"></item>
```
设置一个文本标签。可以通过View.getTag()或 for with View.findViewWithTag()检索含有该标签字符串的View

但一般最好通过ID来查询View，因为它的速度更快，并且允许编译时类型检查

```
<item name="android:visibility">visible</item>
```
设置是否显示View。设置值：visible（默认值，显示）

invisible（不显示，但是仍然占用空间），gone（不显示，不占用空间）

TextView　属性

```
<item name="android:autoLink">none</item>
```
设置是否当文本为URL链接/email/电话号码/map时，文本显示为可点击的链接

可选值(none/web/email/phone/map/all)。这里只有在同时设置text时才自动识别链接，后来输入的无法自动识别

```
<item name="android:autoText">true</item>
```
自动拼写帮助。这里单独设置是没有效果的，可能需要其他输入法辅助才行，效果参见视频

```
<item name="android:bufferType">normal</item>
```
指定getText()方式取得的文本类别。选项editable 类似于StringBuilder可追加字符，

也就是说 getText 后可调用 append 方法设置文本内容。spannable 则可在给定的字符区域使用样式，参见这里 1、这里 2

```
<item name="android:capitalize">none</item>
```
设置英文字母大写类型。设置如下值：sentences仅第一个字母大写；words每一个单词首字母大小，用空格区分单词；

characters每一个英文字母都大写。在模拟器上用PC键盘直接输入可以出效果，但是用软键盘无效果

<item name="android:cursorVisible">true</item>
设定光标为显示/隐藏，默认显示。如果设置 false，即使选中了也不显示光标栏

<item name="android:digits"></item>
设置允许输入哪些字符。如“1234567890.+-*/%\n()”

<item name="android:drawableBottom"></item>
在text的下方输出一个drawable，如图片。如果指定一个颜色的话会把text的背景设为该颜色，并且同时和background使用时覆盖后者

<item name="android:drawableLeft"></item>
在text的左边输出一个drawable（如图片）

<item name="android:drawablePadding"></item>
设置text与drawable(图片)的间隔，与drawableLeft、drawableRight、drawableTop、drawableBottom一起使用
可设置为负数，单独使用没有效果

<item name="android:drawableRight"></iem>
在text的右边输出一个drawable，如图片

<item name="android:drawableTop"></item>
在text的正上方输出一个drawable。在EditView中的效果比较搞笑：  ，居然在文本框里，而且删不了

<item name="android:editable">true</item>
设置是否可编辑, 仍然可以获取光标，但是无法输入

<item name="android:editorExtras"></item>
指定特定输入法的扩展，如“com.mydomain.im.SOME_FIELD”源码跟踪至EditorInfo.extras，暂无相关实现代码

<item name="android:ellipsize">none</item>
设置当文字过长时,该控件该如何显示
有如下值设置：
″start"—﹣省略号显示在开头；
″end"——省略号显示在结尾；
″middle"—-省略号显示在中间；
″marquee" ——以跑马灯的方式显示(动画横向移动)

<item name="android:freezesText">true</item>
设置保存文本的内容以及光标的位置

<item name="android:gravity">center</item>
设置文本位置，如设置成“center”，文本将居中显示

<item name="android:hint"></item>
Text为空时显示的文字提示信息，可通过textColorHint设置提示信息的颜色

<item name="android:imeOptions">normal</item>
设置软键盘的Enter键。有如下值可设置：normal，actionUnspecified，actionNone，actionGo，actionSearch，actionSend，actionNext，actionDone，flagNoExtractUi，flagNoAccessoryAction，flagNoEnterAction。可用’|’设置多个。这里仅设置显示图标之用

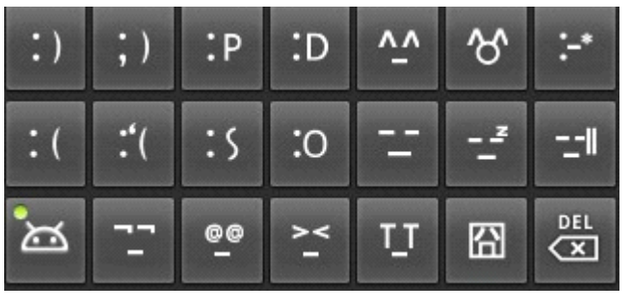
<item name="android:imeActionId"></item>
设置IME动作ID，在onEditorAction中捕获判断进行逻辑操作

<item name="android:imeActionLabel"></item>
设置IME动作标签。但是不能保证一定会使用，猜想在输入法扩展的时候应该有用

<item name="android:includeFontPadding">true</item>
设置文本是否包含顶部和底部额外空白，默认为true

<item name="android:inputMethod"></item>
为文本指定输入法，需要完全限定名（完整的包名）。例如：com.google.android.inputmethod.pinyin，但是这里报错找不到

<item name="android:inputType">none</item>
设置文本的类型，用于帮助输入法显示合适的键盘类型。有如下值设置：none、text、textCapCharacters字母大小、textCapWords单词首字母大小、textCapSentences仅第一个字母大小、textAutoCorrect、textAutoComplete自动完成、textMultiLine多行输入、textImeMultiLine输入法多行（如果支持）、textNoSuggestions不提示、textEmailAddress电子邮件地址、textEmailSubject邮件主题、textShortMessage短信息（会多一个表情按钮出来，点开如下图：）、textLongMessage长讯息？、textPersonName人名、textPostalAddress地址、textPassword密码、textVisiblePassword可见密码、textWebEditText作为网页表单的文本、textFilte文本筛选过滤、textPhonetic拼音输入、numberSigned有符号数字格式、numberDecimal可带小数点的浮点格式、phone电话号码、datetime时间日期、date日期、time时间。部分参考这里



<item name="android:linksClickable">true</item>　　设置链接是否点击连接，即使设置了autoLink

<item name="android:marqueeRepeatLimit">marquee_forever</item>
在ellipsize指定marquee的情况下，设置重复滚动的次数，当设置为marquee_forever时表示无限次

<item name="android:ems"></item>

设置TextView的宽度为N个字符的宽度。这里测试为一个汉字字符宽度，如图： 

<item name="android:maxEms"></item>

设置TextView的宽度为最长为N个字符的宽度。与ems同时使用时覆盖ems选项

<item name="android:minEms"></item>

设置TextView的宽度为最短为N个字符的宽度。与ems同时使用时覆盖ems选项

<item name="android:maxLength"></item>

限制输入字符数。如设置为5，那么仅可以输入5个汉字/数字/英文字母

<item name="android:lineSpacingExtra"></item>　　设置行间距

<item name="android:maxLines"></item>
设置文本的最大显示行数，与width或者layout_width结合使用，超出部分自动换行，超出行数将不显示

<item name="android:minLines"></item>　　设置文本的最小行数，与lines类似

<item name="android:lineSpacingMultiplier"></item> 设置行间距的倍数。如”1.2”

<item name="android:numeric">integer</item>
如果被设置，该TextView有一个数字输入法

有如下值设置：integer正整数、signed带符号整数、decimal带小数点浮点数

<item name="android:password"></item>　　以小点”.”显示文本

<item name="android:phoneNumber">true</item>　　设置为电话号码的输入方式

<item name="android:shadowDx"></item>　　设置阴影横向坐标开始位置

<item name="android:shadowDY"></item>　　设置阴影纵向坐标开始位置

<item name="android:shadowColor"></item>

指定文本阴影的颜色，需要与shadowRadius一起使用。效果： 

<item name="android:shadowRadius"></item>

设置阴影的半径。设置为0.1就变成字体的颜色了，一般设置为3.0的效果比较好

<item name="android:singleLine">true</item>
设置单行显示。如果和layout_width一起使用，当文本不能全部显示时，后面用“…”来表示。如android:text="test_singleLine " android:singleLine="true" android:layout_width="20dp"将只显示“t…”。如果不设置singleLine或者设置为false，文本将自动换行

<item name="android:text"></item>　　设置显示文本

<item name="android:textAppearance"></item>
设置文字外观。如“?android:attr/textAppearanceLargeInverse”这里引用的是系统自带的一个外观，？表示系统是否有这种外观，否则使用默认的外观。可设置的值如下：textAppearanceButton/textAppearanceInverse/textAppearanceLarge/textAppearanceLargeInverse/textAppearanceMedium/textAppearanceMediumInverse/textAppearanceSmall/textAppearanceSmallInverse

<item name="android:textColor"></item>
设置文本颜色

<item name="android:textColorHighlight"></item>
被选中文字的底色，默认为蓝色

<item name="android:textColorHint"></item>
设置提示信息文字的颜色，默认为灰色。与hint一起使用

<item name="android:textColorLink"></item>
文字链接的颜色

item name="android:textScaleX"></item>
设置文字之间间隔，默认为1.0f。分别设置0.5f/1.0f/1.5f/2.0f效果如下：

<item name="android:textSize"></item>
设置文字大小，推荐度量单位”sp”，如”15sp”

<item name="android:textStyle">normal</item>
设置字形[bold(粗体) 0, italic(斜体) 1, bolditalic(又粗又斜) 2] 可以设置一个或多个，用“|”隔开

<item name="android:typeface">normal</item>
设置文本字体，必须是以下常量值之一：normal 0, sans 1, serif 2, monospace(等宽字体) 3]

<item name="android:height"></item>
设置文本区域的高度，支持度量单位：px(像素)/dp/sp/in/mm(毫米)

<item name="android:maxHeight"></item>

设置View的最大高度，单独使用无效，需要与setAdjustViewBounds一起使用。如果想设置图片固定大小，又想保持图片宽高比，需要如下设置：

1） 设置setAdjustViewBounds为true；

2） 设置maxWidth、MaxHeight；

3） 设置设置layout_width和layout_height为wrap_content。

<item name="android:width"></item>

设置文本区域的宽度，支持度量单位：px(像素)/dp/sp/in/mm(毫米)

<item name="android:lines"></item>

设置文本的行数，设置两行就显示两行，即使第二行没有数据

RatingBar 属性

Attribute Name	Related Method	Description
android:isIndicator		Whether this rating bar is an indicator (and non-changeable by the user).
android:numStars		The number of stars (or rating items) to show.
android:rating		The rating to set by default.
android:stepSize		The step size of the rating.

z

<item name="android:addStatesFromChildren">true</item>

android:addStatesFromChildren="true".

属性说明该 viewgroup 的 drawable 属性是否把它的子类的 drawable 的 state 包含进来。

测试中 linearlayout 如果不包含该属性（false），当子 widget 被点击时不会出现被选中的状态。

也就是子类的 state 不会被传递给父类了

<item name="android:allowSingleTap">true</item>

<item name="android:alwaysDrawnWithCache">true</item>

<item name="android:animateOnClick">true</item>

<item name="android:animationCache">true</item>

<item name="android:animationDuration"></item>

<item name="android:autoStart">true</item>

<item name="android:baselineAlignBottom">true</item>

<item name="android:baselineAligned">true</item>

<item name="android:baselineAlignedChildIndex"></item>

<item name="android:bottomOffset"></item>

<item name="android:button">@null</item>

<item name="android:checkMark"></item>

```
<item name="android:checked">true</item>
<item name="android:checkedButton"></item>
<item name="android:childDivider"></item>
<item name="android:childIndicator"></item>
<item name="android:childIndicatorLeft"></item>
<item name="android:childIndicatorRight"></item>
<item name="class"></item>
<item name="android:clipChildren">true</item>
<item name="android:collapseColumns"></item>
<item name="android:columnWidth"></item>
<item name="android:completionHint"></item>
<item name="android:completionHintView"></item>
<item name="android:completionThreshold"></item>
<item name="android:content"></item>
<item name="android:descendantFocusability">beforeDescendants</item>
<item name="android:dial"></item>
<item name="android:disabledAlpha"></item>
<item name="android:divider"></item>
<item name="android:dividerHeight"></item>
<item name="android:dropDownAnchor"></item>
<item name="android:dropDownHeight">fill_parent</item>
<item name="android:dropDownHorizontalOffset"></item>
<item name="android:dropDownSelector"></item>
<item name="android:dropDownVerticalOffset"></item>
<item name="android:dropDownWidth">fill_parent</item>
<item name="android:enabled">true</item>
<item name="android:endYear"></item>
<item name="android:entries"></item>
<item name="android:eventsInterceptionEnabled">true</item>
<item name="android:fadeDuration"></item>
<item name="android:fadeEnabled">true</item>
<item name="android:fadeOffset"></item>
<item name="android:fadeScrollbars">true</item>
<item name="android:fillViewport">true</item>
<item name="android:flipInterval"></item>
<item name="android:footerDividersEnabled">true</item>
<item name="android:foreground"></item>
<item name="android:foregroundGravity">top</item>
<item name="android:foregroundInsidePadding">true</item>
<item name="android:format"></item>
<item name="android:gestureColor"></item>
<item name="android:gestureStrokeAngleThreshold"></item>
<item name="android:gestureStrokeLengthThreshold"></item>
```

```
<item name="android:gestureStrokeSquarenessThreshold"></item>
<item name=android:gestureStrokeType>single</item>
<item name="android:gestureStrokeWidth"></item>
<item name="android:groupIndicator"></item>
<item name="android:hand_hour"></item>
<item name="android:hand_minute"></item>
<item name="android:handle"></item>
<item name="android:headerDividersEnabled">true</item>
<item name="android:horizontalSpacing"></item>
<item name="android:ignoreGravity"></item>
<item name="android:inAnimation"></item>
<item name="android:indeterminate">true</item>
<item name="android:indeterminateBehavior">repeat</item>
<item name="android:indeterminateDrawable"></item>
<item name="android:indeterminateDuration"></item>
<item name="android:indeterminateOnly">true</item>
<item name="android:indicatorLeft"></item>
<item name="android:indicatorRight"></item>
<item name="android:inflatedId"></item>
<item name="android:interpolator"></item>
<item name="layout"></item>
<item name="android:layoutAnimation"></item>
<item name="android:max"></item>
<item name="android:measureAllChildren">true</item>
<item name="android:mode">oneLine</item>
<item name="android:name"></item>
<item name="android:numColumns">auto_fit</item>
<item name="android:orientation">horizontal</item>
<item name="android:outAnimation"></item>
<item name="android:persistentDrawingCache">none</item>
<item name="android:privateImeOptions"></item>
```
提供额外的输入法选项(字符串格式)。依据输入法而决定是否提供
```
<item name="android:progress"></item>
<item name="android:progressDrawable"></item>
<item name="android:prompt"></item>
<item name="android:quickContactWindowSize">modeSmall</item>
<item name="android:scrollHorizontally">true</item>
```
设置文本超出TextView的宽度的情况下，是否出现横拉条
```
<item name="android:secondaryProgress"></item>
<item name="android:selectAllOnFocus">true</item>
```
如果文本是可选择的，让他获取焦点而不是将光标移动为文本的开始位置或者末尾位置。TextView中设置后无效果

```
<item name="android:shrinkColumns"></item>
<item name="android:spacing"></item>
```

```
<item name="android:startYear"></item>
<item name="android:stretchColumns"></item>
<item name="android:stretchMode">none</item>
<item name="style"></item>
<item name="android:tabStripEnabled">true</item>
<item name="android:tabStripLeft"></item>
<item name="android:tabStripRight"></item>
<item name="android:textOff"></item>
<item name="android:textOn"></item>
<item name="android:thumb"></item>
<item name="android:thumbOffset"></item>
<item name="android:topOffset"></item>
<item name="android:uncertainGestureColor"></item>
<item name="android:unselectedAlpha"></item>
<item name="android:useLargestChild">true</item>
<item name="android:verticalSpacing"></item>
<item name="android:weightSum"></item>
```

res/xml

任意的 XML 文件，运行时调用 getResources().getXml(int id)读取

res/raw

直接复制到设备中的任意文件，无需编译，要使用这些资源，调用 getResources.openRawResource(int id)

res/assets

也可以当值任意文件，使用 Activity.this.getAssets().open("xxx.txt");打开

res/menu

描述：菜单可以通过编码实现，也可以通过 XML 文件实现

位置：res/menu/xxx.xml

引用：Java：R.menu.xxx

XML：@[package:]menu/xxx

示例：

game_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/new_game"
        android:icon="@drawable/ic_new_game"
        android:title="new_game" />
```

```
   <item android:id="@+id/help"
        android:icon="@drawable/ic_help"
        android:title="help" />
</menu>
```

重写 `onCreateOptionsMenu(Menu menu)` 方法即可。

```
view plain
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
      Log.d(TAG, "onCreateOptionsMenu() is involed! " + (times++) + " th");
      MenuInflater mInflater = getMenuInflater();
      mInflater.inflate(R.menu.game_menu, menu);

      // 等效下面代码

      //return super.onCreateOptionsMenu(menu);

      return true;// 返回false就不会显示菜单

}
```

ok，运行程序，点击"menu"，显示效果如下：