**原文**

<http://developer.android.com/guide/topics/resources/more-resources.html>

**更多资源类型**

本页定义了其它一些具体的资源类型，包括：

[Bool](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#Bool)

存放布尔值的XML资源。

[Color](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#Color)

存放颜色值的XML资源（十六进制颜色）。

[Dimension](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#Dimension)

存放数量值的XML资源（带计量单位）。

[ID](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#ID)

为应用程序的资源和控件提供唯一标识的XML资源。

[Integer](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#Integer)

存放整数值的XML资源。

[Integer Array](http://developer.android.com/guide/topics/resources/more-resources.html#IntegerArray#IntegerArray)

提供整数数组的XML资源。

[Typed Array](http://developer.android.com/guide/topics/resources/more-resources.html#TypedArray#TypedArray)

提供[TypedArray](http://developer.android.com/reference/android/content/res/TypedArray.html)（用于Drawable对象数组）的XML资源。

**Bool**

用XML格式定义的布尔值。

**注意：**bool是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把bool资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/*filename*.xml

文件名可随意指定。<bool>元素的名称name将被用作资源ID。

资源引用：

Java代码：R.bool.*bool\_name*

XML代码：@[*package*:]bool/*bool\_name*

语法：

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

<resources>

    <[bool](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#bool_%E8%8A%82)

        name="*bool\_name*"

        >[true | false]</bool>

</resources>

元素：

<resources>

**必填项**。必须是根节点。

无属性。

<bool>

布尔值：true或false。

属性：

name

*String类型。*布尔值的名称，用作资源ID。

示例：

存放在res/values-small/bools.xml XML的文件：

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

<resources>

    <bool name="screen\_small">true</bool>

    <bool name="adjust\_view\_bounds">true</bool>

</resources>

以下应用程序代码取出bool值：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

boolean screenIsSmall = res.[getBoolean](http://developer.android.com/reference/android/content/res/Resources.html#getBoolean%28int%29)(R.bool.screen\_small);

以下布局（layout）XML将bool资源用于属性：

<ImageView

    android:layout\_height="fill\_parent"

    android:layout\_width="fill\_parent"

    android:src="@drawable/logo"

    android:adjustViewBounds="@bool/adjust\_view\_bounds" />

**Color**

用XML格式定义的颜色值。用RGB值和alpha通道指定颜色值。可以在任何接受十六进制颜色值的地方使用color资源。还能在XML里用到drawable资源时使用color 资源（比如：android:drawable="@color/green"）。

颜色值总是以（#）字符开头，后面跟着Alpha-红-绿-蓝信息，格式如下之一：

       #*RGB*

       #*ARGB*

       #*RRGGBB*

       #*AARRGGBB*

**注意：**color是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把color资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/colors.xml

文件名可随意指定。<color>元素的名称name将被用作资源ID。

资源引用：

Java代码：R.color.*color\_name*

XML代码：@[*package*:]color/*color\_name*

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_color)>

    <[color](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#color_%E8%8A%82)

        name="*color\_name*"

        >*hex\_color*</color>

</resources>

元素：

<resources>

**必填项**。必须是根节点。

无属性。

<color>

十六进制表示的颜色值。如上所述。

属性：

name

*String类型。*颜色的名称，用作资源ID。

示例：

存放在res/values/colors.xml的XML文件：

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

<resources>

   <color name="opaque\_red">#f00</color>

   <color name="translucent\_red">#80ff0000</color>

</resources>

以下应用程序代码取出color资源：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

int color = res.[getColor](http://developer.android.com/reference/android/content/res/Resources.html#getColor%28int%29)(R.color.opaque\_red);

以下布局（layout）XML将color资源用于属性：

<TextView

    android:layout\_width="fill\_parent"

    android:layout\_height="wrap\_content"

    android:textColor="@color/translucent\_red"

    android:text="Hello"/>

**Dimension**

用XML格式定义的数量值。数量值是用数字后跟度量单位来指定的。例如：10px, 2in, 5sp。Android支持以下度量单位：

dp

分辨率无关的像素（Pixel）单位，一种基于屏幕的物理（像素）分辨率的抽象单位。此单位基于一个160 dpi（每英寸点数）的屏幕，所以*160dp常常是1英寸*且与屏幕像素分辨率无关。dp和像素的比率会随着屏幕密度而变化，但不一定成正比。建议用于在layout里指定View尺寸 ，这样UI在不同屏幕上能自动缩放而显示出相同的大小。（“dip”和“dp”同义，编译器都可接受，虽然“dp”更近似于“sp”。）

sp

缩放无关的像素单位，类似于dp，但还会根据用户的字体大小设置进行缩放。建议用于指定字体大小，这样根据屏幕分辨率和用户设置都能自动调整。

pt

点，基于屏幕实际尺寸，对应1/72英寸。

px

像素，与屏幕实际像素一致。这是个不建议使用的单位，因为在不同设备上的实际表现会差异很大，每种设备每英寸的像素数可能不同，屏幕上的总像素数亦可能更多或更少。

mm

毫米，基于屏幕物理尺寸。

in

英寸，基于屏幕物理尺寸。

**注意：**dimension是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把dimension资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/*filename*.xml

文件名可随意指定。<dimen>元素的名称name将被用作资源ID。

资源引用：

Java代码：R.dimen.*dimension\_name*

XML代码：@[*package*:]dimen/*dimension\_name*

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_dimension)>

    <[dimen](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#dimension_%E8%8A%82)

        name="*dimension\_name*"

        >*dimension*</dimen>

</resources>

元素：

<resources>

**必填项**。必须是根节点。

无属性。

<dimen>

度量值，用浮点数表示，后跟一个计量单位（dp、sp、pt、px、mm、in），如上所述。

属性：

name

*String类型。*度量的名称，用作资源ID。

示例：

存放在res/values/dimens.xml的XML文件：

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

<resources>

    <dimen name="textview\_height">25dp</dimen>

    <dimen name="textview\_width">150dp</dimen>

    <dimen name="ball\_radius">30dp</dimen>

    <dimen name="font\_size">16sp</dimen>

</resources>

以下应用程序代码取出dimension资源：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

float fontSize = res.[getDimension](http://developer.android.com/reference/android/content/res/Resources.html#getDimension%28int%29)(R.dimen.font\_size);

以下layout XML将dimensions用于属性：

<TextView

    android:layout\_height="@dimen/textview\_height"

    android:layout\_width="@dimen/textview\_width"

    android:textSize="@dimen/font\_size"/>

**ID**

用XML格式定义的资源唯一ID。对应<item>元素里指定的名称，Android开发工具在R.java类中创建一个唯一的整数。可用来标识应用程序资源（比如：UI布局中的一个[View](http://developer.android.com/reference/android/view/View.html)）,或者在应用程序代码中被用作一个唯一的整数（比如：对话框的ID或一个返回值）。

**注意：**ID是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把ID资源和其他简单类型资源一起放入一个<resources>元素下。而且，请记住ID资源不代表一个实际的资源项，而只是一个可与其他资源绑定的唯一ID，或是一个用于应用程序代码中的唯一整数。

文件位置：

res/values/*filename.xml*  
文件名可随意指定。

资源引用：

Java代码：R.id.*name*

XML代码：@[*package*:]id/*name*

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_ID)>

    <[item](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#item_ID)

        type="id"

        name="*id\_name*" />

</resources>

元素：

<resources>

**必填项**。必须是根节点。

无属性。

<item>

定义一个唯一的ID。不含值，只含属性。

属性：

Type

必须是“id”。

name

*String类型。*ID的唯一名称。

示例：

存放在res/values/ids.xml的XML文件：

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

<resources>

    <item type="id" name="button\_ok" />

    <item type="id" name="dialog\_exit" />

</resources>

以下layout段将“button\_ok”用于Button控件ID：

<Button android:id="**@id/button\_ok**"

    style="@style/button\_style" />

注意android:id的值：ID引用里不含加号“+”了，因为这个ID已经在上面的ids.xml中定义过了。（如果XML资源里用加号指定一个ID—类似格式android:id="@+id/name"—那就意味着“name”命名的ID还不存在并需要创建它。）

以下代码段示例用“dialog\_exit”ID作为对话框的唯一标识：

[showDialog](http://developer.android.com/reference/android/app/Activity.html#showDialog%28int%29)(**R.id.dialog\_exit**);

在同一个应用程序里，在生成对话框时“dialog\_exit”ID用作条件比较：

protected Dialog [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29)(int id) {

    Dialog dialog;

    switch(id) {

    case **R.id.dialog\_exit**:

        ...

        break;

    default:

        dialog = null;

    }

    return dialog;

}

**Integer**

用XML格式定义的整数资源。

**注意：**integer是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把integer资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/*filename.xml*

文件名可随意指定。<integer>元素的名称name将用作资源ID。

资源引用：

Java代码：R.integer.*integer\_name*

XML代码：@[*package*:]integer/*integer\_name*。

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_integer)>

    <[integer](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#integer_%E8%8A%82)

        name="*integer\_name*"

        >*integer*</integer>

</resources>

元素：

<resources>

**必填项**。必须是根节点。

无属性。

<integer>

一个整数。

属性：

name

*String类型。*整数的名称。用作资源ID。

示例：

存放在res/values/integers.xml的XML文件：

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

<resources>

    <integer name="max\_speed">75</integer>

    <integer name="min\_speed">5</integer>

</resources>

以下应用程序代码取出整数资源：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

int maxSpeed = res.[getInteger](http://developer.android.com/reference/android/content/res/Resources.html#getInteger%28int%29)(R.integer.max\_speed);

**Integer Array**

用XML格式定义的整数数组。

**注意：** integer array是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把integer array资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/*filename.xml*

文件名可随意指定。<integer-array>元素的名称name将用作资源ID。

编译后的资源数据类型：

指向整数数组的指针。

资源引用：

Java代码：R.array.*string\_array\_name*

XML代码：@[*package*:]array.*integer\_array\_name*

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_integer_array)>

    <[integer-array](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#integer_array%E8%8A%82)

        name="*integer\_array\_name*">

        <[item](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#item_integer_array)

            >*integer*</item>

    </integer-array>

</resources>

元素：

<resources>

**必填项。**必须是根节点。

无属性。

<integer-array>

定义整数数组。包含一个或多个<item>子元素。

属性：

name

*String类型。*数组的名称。作为资源ID用于引用数组。

<item>

整数。可以是指向另一个整数资源的引用。必须是<integer-array> 的子元素。

无属性。

示例：

存放在res/values/integers.xml的XML文件：

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

<resources>

    <integer-array name="bits">

        <item>4</item>

        <item>8</item>

        <item>16</item>

        <item>32</item>

    </integer-array>

</resources>

取出integer array的应用程序代码：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

int[] bits = res.[getIntArray](http://developer.android.com/reference/android/content/res/Resources.html#getIntArray%28int%29)(R.array.bits);

**Typed Array**

用XML格式定义的[TypedArray](http://developer.android.com/reference/android/content/res/TypedArray.html)。用于创建其它资源的数组，比如drawable。注意数组元素不必是同一类型的，可以创建多种资源组成的数组。但必须小心处理数组内不同的数据类型，利用[TypedArray](http://developer.android.com/reference/android/content/res/TypedArray.html)的get...()属性正确地读取每个数据项。

**注意：**typed array是简单类型资源，是用名称（name）属性（而非XML文件名）来直接引用的。因此，在一个XML文件里，可以把typed array资源和其他简单类型资源一起放入一个<resources>元素下。

文件位置：

res/values/*filename*.xml  
文件名可随意指定。<array>元素的名称name将用作资源ID。

编译后的资源数据类型：

指向[TypedArray](http://developer.android.com/reference/android/content/res/TypedArray.html)的指针。

资源引用：

Java代码：R.array.*array\_name*

XML代码：@[*package*:]array.*array\_name*

语法：

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

<[resources](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#resources_typed_array)>

    <[array](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#array%E8%8A%82)

        name="*integer\_array\_name*">

        <[item](http://www.cnblogs.com/over140/archive/2011/10/24/2222255.html#item_typed_array)>*resource*</item>

    </array>

</resources>

元素：

<resources>

**必填项。**必须是根节点。

无属性。

<array>

数组定义。包含一个或多个<item>元素。

属性：

android:name

*String类型。*数组的名称。作为资源ID用于引用数组。

<item>

资源。可以是指向一个资源的引用，或是一个简单数据类型。必须是<array>的子元素。

无属性。

示例：

存放在res/values/ arrays.xml的XML文件：

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

<resources>

    <array name="icons">

        <item>@drawable/home</item>

        <item>@drawable/settings</item>

        <item>@drawable/logout</item>

    </array>

    <array name="colors">

        <item>#FFFF0000</item>

        <item>#FF00FF00</item>

        <item>#FF0000FF</item>

    </array>

</resources>

以下程序代码取出每个数组并读取第一个数组元素：

Resources res = [getResources()](http://developer.android.com/reference/android/content/Context.html#getResources%28%29);

TypedArray icons = res.[obtainTypedArray](http://developer.android.com/reference/android/content/res/Resources.html#obtainTypedArray%28int%29)(R.array.icons);

Drawable drawable = icons.[getDrawable](http://developer.android.com/reference/android/content/res/TypedArray.html#getDrawable%28int%29)(0);

TypedArray colors = res.[obtainTypedArray](http://developer.android.com/reference/android/content/res/Resources.html#obtainTypedArray%28int%29)(R.array.icons);

int color = colors.[getColor](http://developer.android.com/reference/android/content/res/TypedArray.html#getColor%28int,%20int%29)(0,0);