访问资源

内容快览

- 可以使用 `R.drawable.myimage` 等来自 `R.java` 的整型数在代码中引用资源
- 可以使用 `@drawable/myimage` 等特殊 XML 语法在资源中引用资源
- 您还可以通过 `Resources` 中的方法访问您的应用资源

关键类

- `Resources`

本文内容

另请参阅

- 提供资源
- 资源类型

您在应用中提供资源后（提供资源中对此做了阐述），可通过引用其资源 ID 来应用该资源。 所有资源 ID 都在您项目的 `R` 类中定义，后者由 `aapt` 工具自动生成。

编译应用时，`aapt` 会生成 `R` 类，其中包含您的 `res/` 目录中所有资源的资源 ID。 每个资源类型都有对应的 `R` 子类（例如，`R.drawable` 对应于所有可绘制对象资源），而该类型的每个资源都有对应的静态整型数（例如，`R.drawable.icon`）。这个整型数就是可用来检索资源的资源 ID。

尽管资源 ID 是在 `R` 类中指定的，但您应该永远都不需要在其中查找资源 ID。 资源 ID 始终由以下部分组成：

- *资源类型*：每个资源都被分到一个“类型”组中，例如 `string`、`drawable` 和 `layout`。如需了解有关不同类型的详细信息，请参阅资源类型。
- *资源名称*，它是不包括扩展名的文件名；或是 XML `android:name` 属性中的值，如果资源是简单值的话（例如字符串）。

访问资源的方法有两种：

- **在代码中：**使用来自 `R` 类的某个子类的静态整型数，例如：

  `R.string.hello`

  `string` 是资源类型，`hello` 是资源名称。当您提供此格式的资源 ID 时，有许多 Android API 可以访问您的资源。请参阅在代码中访问资源。
- **在 XML 中：**使用同样与您 `R` 类中定义的资源 ID 对应的特殊 XML 语法，例如：

  `@string/hello`

  `string` 是资源类型，`hello` 是资源名称。您可以在 XML 资源中任何应该存在您在资源中所提供值的地方使用此语法。请参阅在 XML 中访问资源。

在代码中访问资源

您可以通过以方法参数的形式传递资源 ID，在代码中使用资源。例如，您可以设置一个 `ImageView`，以利用 `setImageResource()` 使用 `res/drawable/myimage.png` 资源：

```
ImageView imageView = (ImageView) findViewById(R.id.myimageview);
imageView.setImageResource(R.drawable.myimage);
```

您还可以利用 `Resources` 中的方法检索个别资源，您可以通过 `getResources()` 获得资源实例。

语法

以下是在代码中引用资源的语法：

`[<package_name>.]R.<resource_type>.<resource_name>`

- `<package_name>` 是资源所在包的名称（如果引用的资源来自您自己的资源包，则不需要）。
- `<resource_type>` 是资源类型的 `R` 子类。
- `<resource_name>` 是不带扩展名的资源文件名，或 XML 元素中的 `android:name` 属性值（如果资源是简单值）。

如需了解有关各资源类型及其引用方法的详细信息，请参阅资源类型。

访问原始文件

尽管并不常见，但您的确有可能需要访问原始文件和目录。如果确有需要，则将您的文件保存在 `res/` 中不起作用，因为从 `res/`读取资源的唯一方法是使用资源 ID。您可以改为将资源保存在 `assets/` 目录中。

保存在 `assets/` 目录中的文件*没有*资源 ID，因此您无法通过 `R` 类或在 XML 资源中引用它们。您可以改为采用类似普通文件系统的方式查询 `assets/` 目录中的文件，并利用 `AssetManager` 读取原始数据。

不过，如果只需要读取原始数据（例如视频文件或音频文件）的能力，则可将文件保存在 `res/raw/` 目录中，并利用 `openRawResource()` 读取字节流。

用例

有许多方法接受资源 ID 参数，您可以利用 `Resources` 中的方法检索资源。您可以通过 `Context.getResources()` 获得 `Resources` 的实例。

以下是一些在代码中访问资源的示例：

```
// Load a background for the current screen from a drawable resource
getWindow().setBackgroundDrawableResource(R.drawable.my_background_image) ;

// Set the Activity title by getting a string from the Resources object, because
//  this method requires a CharSequence rather than a resource ID
getWindow().setTitle(getResources().getText(R.string.main_title));

// Load a custom layout for the current screen
setContentView(R.layout.main_screen);

// Set a slide in animation by getting an Animation from the Resources object
mFlipper.setInAnimation(AnimationUtils.loadAnimation(this,
        R.anim.hyperspace_in));

// Set the text on a TextView object using a resource ID
TextView msgTextView = (TextView) findViewById(R.id.msg);
msgTextView.setText(R.string.hello_message);
```

注意：切勿手动修改 `R.java` 文件 — 它是在编译您的项目时由 `aapt` 工具生成的。您下次编译时所有更改都会被替代。

在 XML 中访问资源

您可以利用对现有资源的引用为某些 XML 属性和元素定义值。创建布局文件时，为给您的小部件提供字符串和图像，您经常要这样做。

例如，如果您为布局添加一个 `Button`，应该为按钮文本使用字符串资源：

```
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/submit" />
```

语法

以下是在 XML 资源中引用资源的语法：

`@[<package_name>:]<resource_type>/<resource_name>`

- `<package_name>` 是资源所在包的名称（如果引用的资源来自相同的包，则不需要）
- `<resource_type>` 是资源类型的 `R` 子类
- `<resource_name>` 是不带扩展名的资源文件名，或 XML 元素中的 `android:name` 属性值（如果资源是简单值）。

如需了解有关各资源类型及其引用方法的详细信息，请参阅资源类型。

用例

在某些情况下，您必须使用资源作为 XML 中的值（例如，对小部件应用可绘制图像），但您也可以在 XML 中任何接受简单值的地方使用资源。例如，如果您具有以下资源文件，其中包括一个颜色资源和一个字符串资源：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <color name="opaque_red">#f00</color>
   <string name="hello">Hello!</string>
</resources>
```

您可以在以下布局文件中使用这些资源来设置文本颜色和文本字符串：

```
<?xml version="1.0" encoding="utf-8"?>
<EditText xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:textColor="@color/opaque_red"
    android:text="@string/hello" />
```

在此情况下，您无需在资源引用中指定包名称，因为资源来自您自己的资源包。 要引用系统资源，您需要加入包名称。 例如：

```
<?xml version="1.0" encoding="utf-8"?>
<EditText xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:textColor="@android:color/secondary_text_dark"
    android:text="@string/hello" />
```

> **注：**您应该始终使用字符串资源，以便将您的应用本地化为其他语言。如需了解有关创建备用资源（例如本地化字符串）的信息，请参阅提供备用资源。 如需查看将您的应用本地化为其他语言的完整指南，请参阅本地化。

您甚至可以在 XML 中使用资源创建别名。例如，您可以创建一个可绘制对象资源，将其作为另一个可绘制对象资源的别名：

```
<?xml version="1.0" encoding="utf-8"?>
<bitmap xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@drawable/other_drawable" />
```

这听起来多余，但对使用备用资源可能很有帮助。阅读更多关于创建别名资源的内容。

引用样式属性

您可以通过样式属性资源在当前应用的风格主题中引用某个属性的值。 通过引用样式属性，您可以不采用为 UI 元素提供硬编码值这种方式，而是通过为 UI 元素设置样式，使其匹配当前风格主题提供的标准变型来定制这些元素的外观。引用样式属性的实质作用是，“在当前风格主题中使用此属性定义的样式”。

要引用样式属性，名称语法几乎与普通资源格式完全相同，只不过将 at 符号 (`@`) 改为问号 (`?`)，资源类型部分为可选项。 例如：

`?[<package_name>:][<resource_type>/]<resource_name>`

例如，您可以通过以下代码引用一个属性，将文本颜色设置为与系统风格主题的“主要”文本颜色匹配：

```
<EditText id="text"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textColor="?android:textColorSecondary"
    android:text="@string/hello_world" />
```

在以上代码中，`android:textColor` 属性指定当前风格主题中某个样式属性的名称。Android 现在会使用应用于 `android:textColorSecondary` 样式属性的值作为 `android:textColor` 在这个小部件中的值。由于系统资源工具知道此环境中肯定存在某个属性资源，因此您无需显式声明类型（类型应为 `?android:attr/textColorSecondary`）— 您可以将 `attr` 类型排除在外。

访问平台资源

Android 包含许多标准资源，例如样式、风格主题和布局。要访问这些资源，请通过 `android` 包名称限定您的资源引用。例如，您可以将 Android 提供的布局资源用于 `ListAdapter` 中的列表项：

```
setListAdapter(new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, myarray));
```

在上例中，`simple_list_item_1` 是平台为 `ListView` 中的项目定义的布局资源。您可以使用它，而不必自行创建列表项布局。 如需了解详细信息，请参阅列表视图开发者指南。