概述

- 疯狂Android讲义（288页-380页，第八章 Android数据存储和IO）
  - 第六章
    - 深入学习各种资源的定义和使用，包括java代码和xml以及如何实际访问
      - values、drawable、Property Animation、原始XML资源、layout、menu、style、theme和Attribute
    - 如何做国际化和自适应
  - 第八章
    - 使用使用SharedPreferences获取，存储数据

疯狂Android第六章

Values

字符串、颜色、尺寸

资源类型	资源文件的默认名	对应于R类中的内部类的名称
字符串资源	/res/values/strings.xml	R.string
颜色资源	/res/values/colors.xml	R.color
尺寸资源	/res/values/dimens.xml	R.dimen
- Android中的颜色值是通过红(Red)、绿(Green)、蓝(Blue)三原色以及一个透明度(Alpha)值来表示的，颜色值总是以井号(#)开头，就是Alpha-Red-Green-Blue的形式。其中Alpha值可以省略，如果省略了Alpha值，那么该颜色默认是完全不透明的。
  - `#RGB`: 分别指定红、绿、蓝三原色的值(只支持0~f这16级颜色)来代表颜色。
  - `#ARGB`: 分别指定红、绿、蓝三原色的值(只支持0~f这16级颜色)及透明度(只支持0~f这16级透明度)来代表颜色。
  - `#RRGGBB`:分别指定红、绿、蓝三原色的值(支持00~ff这256级颜色)来代表颜色。
  - `#AARRGGBB`:分别指定红、绿、蓝三原色的值(支持00~ff这256级颜色)以及透明度(支持00~ff这256级透明度)来代表颜色。
- 定义
  - 字符串资源文件位于/res/values/目录下，字符串资源文件的根元素是<resources...>，,该元素里每个<string..>.子元素定义一一个字符串 常量，其中<string../>元素的 name 属性指定该常量的名称，<string../> 元素开始标签和结束标签之间的内容代表字符串值
    - ```
      <string name="hello">Hello World</ string>
      ```
  - 颜色 资源文件位于/res/values/目录下，.颜色资源文件的根元素是<resoures...>,.该元素里每个<color...>子元素定义一个字符串常量，其中<colo...元素的name属性指定该颜色的名称，<color..>元素开始标签和结束标签之间的内容代表颜色值.
  - 尺寸资源文件位于/res/values/目录下，尺寸资源文件的根元素是<resources..>，该元素里每个<dimen.../>子元素定义一个尺寸常量，其中<dimen.../>元素的name 属性指定该尺寸的名称，<dimen...>元素开始标签和结束标签之间的内容代表尺寸值.
      ```

- Android 允许使用资源文件来定义boolean 常量。在/res/values/目录下增加一一个bools.xml 文件，该文件的根元素也是<resources.../>，根元素内通过<bool...>子元素来定义boolean常量，整形integer也类似

国际化

- 引入国际化的目的是为了提供自适应、更友好的用户界面。程序国际化指的是同一个应用在不同语言、国家环境下，可以自动呈现出对应的语言等用户界面，从而提供更友好的界面。
- 为了给这些消息提供不同国家、语言的版本，开发者需要为values 目录添加几个不同的语言国家版本。不同values文件夹的命名方式为: `values-语言代码-r国家代码`
- 在不同语言的国际化资源文件中所有消息的key是相同的，在不同国家、语言环境下，消息资源key对应的value不同。

数组

- Android采用位于/res/values目录下的arrays.xml文件来定义数组资源，定义数组时XML资源文件的根元素也是<resorc.../>,该元素内可包含如下三种子元素。
  - <array....>子元素: 定义普通类型的数组，例如Drawable数组。
  - <string-array...>子元素: 定义字符串数组。
  - <integer-array..> 子元素:定义整型数组。
- 为了能在Java或Kotlin程序中访问到实际数组，Resources 类提供了方法。
  - String[] getStringArray(int id):根据资源文件中字符串数组资源的名称来获取实际的字符串数组。
  - int[] getIntArray(int id):根据资源文件中整型数组资源的名称来获取实际的整型数组。
  - TypedArray obtainTypedArray(int id):根据资源文件中普通数组资源的名称来获取实际的普通数组。
    - TypedArray代表一个通用类型的数组，该类提供了getXxx(int index)方法来获取指定索引处的数组元素。

Drawable

- 图片资源是最简单的Drawable 资源，只要把 `*.png`、`*jpg`、`*.gif` 等格式的图片放入/res/drawable-xxx目录下，Android SDK就会在编译应用中自动加载该图片，并在R资源清单类中生成该资源的索引。
  - 注意：Android要求图片资源的文件名必须符合Java或Kotlin标识符的命名规则;否则，Android SDK无法为该图片在R类中生成资源索引。
- 在程序中获得实际的Drawable对象：Resources 提供了getDrawable(int id)方法，该方法即可根据Drawable资源在R资源清单类中的ID来获取实际的Drawable对象。

StateListDrawable资源

- StateListDrawable用于组织多个Drawable对象。当使用StateListDrawable作为目标组件的背景、前景图片时，StateListDrawable对象所显示的Drawable对象会随目标组件状态的改变而自动切换。
- StateListDrawable对象的XML文件的根元素为<selector../>,该元素可以包含多个<item.../>元素
  - android:color或android:drawable:指定颜色或Drawable对象。
  - android:state_ xxx: 指定一个 特定状态。比如激活、已勾选、可勾选、可用、开始
- StateListDrawable不仅可以让文本框里文字的颜色随文本框状态的改变而切换，也可让按钮的背景图片随按钮状态的改变而切换。StateListDrawable的功能非常灵活，它可以让各种组件的背景、前景随状态的改变而切换。

LayerDrawable资源

- LayerDrawable包含一个Drawable数组，系统将会按这些Drawable对象的数组顺序来绘制它们，索引最大的Drawable对象将会被绘制在最上面。
- LayerDrawable对象的XML文件的根元素为<layer-list...>,该元素可以包含多个<item..>元素，item元素的属性：
  - android:drawable: 指定作为LayerDrawable元素之一的 Drawable对象。
  - `android:id`:为该Drawable对象指定一个标识。
  - android:buttom|top|left|right: 它们用于指定一个长度值，用于指定将该Drawable对象绘制到目标组件的指定位置。

ShapeDrawable资源

- ShapeDrawable用于定义一个基本的几何图形(如矩形、圆形、线条等)
- ShapeDrawable的XML文件的根元素是<shape.../>元素，子元素有corners、gradient、padding、size、solid、stroke该元素可指定属性
  - android:shape=["rectangle" |"oval" I "line" I "ring"]:指定定义哪种类型的几何图形。

ClipDrawable资源

- ClipDrawable代表从其他位图上截取的一个“图片片段”。在XML文件中定义ClipDrawable对象使用<clip../>元素
  - clip元素的属性
    - android:drawable:指定截取的源Drawable对象。
    - android:clipOrientation:指定截取方向，可设置水平截取或垂直截取。
    - android:gravity:指定截取时的对齐方式。
  - 使用ClipDrawable对象时可调用setLevel(int level)方法来 设置截取的区域大小，当level为0时，截取的图片片段为空;当level为10000时，截取整张图片。

AnimationDrawable资源

- AnimationDrawable代表一个动画Android既支持传统的逐帧动画(类似于电影方式，一张图片、一张图片地切换)， 也支持通过平移、变换计算出来的补间动画。
- 定义补间动画的XML资源文件以<set...>元素作为根元素，该元素内可以指定如下4个元素。
属性	说明
alpha	设置透明度的改变。
scale	设置图片进行缩放变换。
translate	设置图片进行位移变换。
rotate	设置图片进行旋转。
  - 定义动画的XML资源应该放在/res/anmi/路径下，Android Studio 创建Android应用时，默认不会包含该路径，开发者需要自行创建该路径。
  - 元素都可指定一个android:interpolator属性，该属性指定动画的变化速度，可以实现匀速、正加速、负加速、无规则变加速
  - 如果程序想让<set.../>元素下所有的变换效果使用相同的动画速度，则可指定android:shareInterpolator="true"

- 在Java或Kotlin代码中获取实际的Animation对象，可调用AnimationUtils的loadAnimation(Context ctx, int resId)方法。使用View的startAnimation（ ）方法可以执行动画。

屏幕自适应

- 屏幕资源需考虑的方面。
  - 屏幕尺寸:屏幕尺寸可分为small (小屏幕)、normal (中等屏幕)、large (大屏幕)、xlarge (超大屏幕) 4种。
    - 屏幕分辨率：Android 默认把drawable 目录(存放图片等Drawable 资源的目录)分为drawable-ldpi、drawable- mdpi、drawable-hdpi、 drawable xhdpi、drawable-xxhdpi 这些子目录，它们正是用于为不同分辨率屏幕准备图片的。
  - 屏幕分辨率: 屏幕分辨率可分为ldpi (低分辨率)、mdpi (中等分辨率)、hdpi (高分辨率)、xhdpi (超高分辨率)、xxhdpi (超超高分辨率) 5种。
    - 屏幕尺寸:可以为layout、values 等目录增加后缀small、normal、 large 和xlarge,分别为不同尺寸的屏幕提供相应资源。
  - 屏幕方向:屏幕方向可分为land (横屏)和port (竖屏)两种。
- 最低屏幕要求
  - xlarge屏幕尺寸至少需要960dpx720dp。
  - large屏幕尺寸至少需要640dpx480dp。
  - normal屏幕尺寸至少需要470dpx320dp。
  - small屏幕尺寸至少需要426dpx 320dp。

PropertyAnimation 资源

- Animator代表一个属性动画，但它只是一个抽象类，通常会使用它的子类: AnimatorSet、ValueAnimator、ObjectAnimator、TimeAnimator。
- 定义属性动画的XML资源文件能以如下三个元素中的任意一个作为根元素。
  - <set.../>: 它是一一个父元素，用于包含<objectAnimator...>.、<animator...>或<set..>.子元素，该元素定义的资源代表AnimatorSet对象。
  - <objectAnimator.../>: 用于定义ObjectAnimator动画。
  - <animator../>: 用于定义ValueAnimator动画。

原始XML资源

- Android 应用有一些初始化的配置信息、应用相关的数据资源需要保存，一般推荐使用XML文件来保存它们，这种资源就被称为原始XML资源。
- 定义：原始XML资源一般保存在/res/xml/路径下，Android Studio创建Android应用时,/res/目录下并没有包含xml子目录，开发者应该自行手动创建xml子目录。Android应用对原始XML资源没有任何特殊的要求，只要它是一份格式良好的XML文档即可。
- 为了在Java或Kotlin程序中获取实际的XML文档，可以通过Resources的方法来实现。
  - XmlResourceParser getXml(int id):获取XML文档，并使用一个XmlPullParser来解析该XML文档，该方法返回一个解析器对象(XmlResourceParser是XmlPullParser的子类)。
  - InputStream openRawResource(int id):获取XML文档对应的输入流。
  - 大部分时候直接调用getXml(int id)方法来获取XML文档，并对该文档进行解析。Android默认使用内置的Pull解析器来解析XML文件。
  - 除使用Pull解析之外，也可使用DOM或SAX对XML文档进行解析。一般的Java或Kotlin应用会使用JAXP API来解析XML文档。

        - Pull解析方式有点类似于SAX解析，它们都采用事件驱动的方式来进行解析。当Pull 解析器开始解析之后，开发者可不断地调用Pull解析器的next()方法获取下一一个解析事件(开始文档、结束文档、开始标签、结束标签等)，当处于某个元素处时，可调用XmlPullParser的getAttributeValue()方法来获取该元素的属性值，也可调用XmlPullParser的nextText( ) 方法来获取文本节点的值。
        - 如果开发者希望使用DOM、SAX或其他解析器来解析XML资源，那么可调用openRawResource (int id)方法来获取XML资源对应的输入流，这样即可自行选择解析器来解析该XML资源了。

Layout

- Layout资源文件应放在/res/layout/目录下，Layout 资源文件的根元素通常是各种布局管理器，比如LinearLayout、TableLayout、FrameLayout等。

Menu

- Android菜单资源文件放在/res/menu目录下，菜单资源的根元素通常是<menu...>元素，<menu../>元素无须指定任何属性。

Style

- Style：Android样式包含一组格式， 为一个组件设置使用某个样式时，该样式所包含的全部格式将会应用于该组件。
- Android的样式资源文件也放在/res/values/目录下，样式资源文件的根元素是<resources..>元素，该元素内可包含多个<style...>子元素，每个<style...> 子元素定义一个样式。
    - <style...>元素属性
        - name:指定样式的名称。
        - parent: 指定该样式所继承的父样式。当继承某个父样式时，该样式将会获得父样式中定义的全部格式。当然，当前样式也可以覆盖父样式中指定的格式。
    - 在<style...>元素内可包含多个<item...>子元素，每个<item...> 子元素定义一一个格式项。

Theme

- 主题资源的XML文件通常也放在/res/values/目录下，主题资源的XML文件同样以<resources...>元素作为根元素，同样使用<style...> 元素来定义主题。
- 主题与样式的区别主要体现在:
    - 主题不能作用于单个的View组件，主题应该对整个应用中的所有Activity起作用，或对指定的Activity起作用。
    - 主题定 义的格式应该是改变窗口外观的格式， 例如窗口标题、窗口边框等。
- 使用
    - 在代码中使用setTheme()方法
    - 在AndroidManifest.xml文件中对指定应用、指定Activity 应用主题，使用android:theme属性
- Android中提供了几种内置的主题资源，这些主题通过查询Android.R.style类可以看到。

Attribute

- 当在XML布局文件中使用Android系统提供的View组件时，开发者可以指定多个属性，这些属性可以很好地控制View组件的外观行为。如果用户开发的自定义View组件也需要指定属性，可以使用属性资源

- 属性资源文件也放在/res/values目录下，属性资源文件的根元素也是<resources../>元素，该元素包含如下两个子元素。
    - attr子元素:定义一个属性。
    - declare-styleable 子元素:定义一个styleable 对象，每个styleable 对象就是一组attr属性的集合。

原始资源

- 类似于声音文件及其他各种类型的文件，只要Android没有为之提供专门的支持，这种资源都被称为原始资源。Android 的原始资源可以放在如下两个地方。
- 位置
    - 位于/res/raw/目 录下，Android SDK会处理该目录下的原始资源，AndroidSDK 会在R清单类中为该目录下的资源生成一个索引项。
        - Android SDK会为位于/res/raw/目录下的资源在R清单类中生成一个索引项
    - 位于/assets/目录下，该目录下的资源是更彻底的原始资源。Android 应用需要通过AssetManager来管理该目录下的原始资源。
        - AssetManager是一个专门管理/assets/目录下原始资源的管理器类, AssetManager提供了如下两个常用方法来访问Assets资源。
            - InputStream open(String fileName):根据文件名来获取原始资源对应的输入流。
            - AssetFileDescriptor openFd(String fileName): 根据文件名来获取原始资源对应的AssetFileDescriptor。AssetFileDescriptor 代表了一项原始资源的描述，应用程序可通过AssetFileDescriptor来获取原始资源。

数据存储和IO

- 数据的持久化

SharedPreferences

概述

- SharedPreferences保存的数据主要是类似于配置信息格式的数据，因此它保存的数据主要是简单类型的key-value对。
- SharedPreferences接口主要负责读取应用程序的Preferences数据
- SharedPreferences方法

方法	说明
boolean contains(String key)	判断SharedPreferences是否包含特定key的数据。
Map<String, ?> getAll()	获取SharedPreferences数据里全部的key-value对。
getXxx(String key, xxx defValue)	获取SharedPreferences 数据里指定key对应的value。如果该key不存在，则返回默认值defValue。其中xxX可以是boolean、float、int、 long、String等各种基本类型的值。

- SharedPreferences接口并没有提供写入数据的能力，通过SharedPreferences. Editor才允许写入，SharedPreferences调用edit()方法即可获取它所对应的Editor 对象。

- SharedPreferences. Editor方法
SharedPreferences.Editor clear()	**清空SharedPreferences里所有数据。**
SharedPreferences.Editor putXxx(String key, xXx value)	向SharedPreferences 存入指定key对应的数据。其中xxx可以是boolean、float、 int、long、String等各种基本类型的值。
SharedPreferences.Editor remove(String key)	删除SharedPreferences里指定key对应的数据项。
boolean apply()	当Editor编辑完成后，调用该方法提交修改。
commit()	与applyO功能类似，但commit()会立即提交修改:而apply()在后台提交修改，不会阻塞前台!

- SharedPreferences本身是一个**接口**，程序无法直接创建SharedPreferences 实例，只能通过Context提供的getSharedPreferences(String name，int mode)方法来获取SharedPreferences实例
  - mode参数可选值
    - Context.MODE PRIVATE:指定该SharedPreferences数据只能被本应用程序读写。
    - Context.MODE WORLD_ READABLE: 指定该SharedPreferences数据能被其他应用程序读，但不能写。
    - ContextMODE WORLD_WRITEABLE:指定该SharedPreferences 数据能被其他应用程序读写。

存储位置和格式

- SharedPreferences数据保存在 `/data/data/<package name>/shared _prefs` 目录下，SharedPreferences数据以XML格式保存。
  - SharedPreferences 数据文件的根元素是<map..>元素，该元素里每个子元素代表一一个key-value对，当value是整数类型时，使用<int...> 子元素;当value是字符串类型时，使用<string.../> 子.....依此类推。