**关于屏幕适配**

**1.屏幕尺寸Screen size**

即显示屏幕的实际大小，按照屏幕的对角线进行测量。

为简单起见，Android把所有的屏幕大小分为四种尺寸：小，普通，大，超大(分别对应：small, normal, large, and extra large)。各个屏幕尺寸换算为dp单位：

Small：

大于：320x426dp

Normal：

大于：320x470dp

Large：

大于：480x640dp

Xlarge：

大于：720x960dp

应用程序可以为这四种尺寸分别提供不同的自定义屏幕布局，平台将根据屏幕实际尺寸选择对应布局进行渲染，这种选择对于程序则是透明的。

**2.** **设备独立像素Density-independent pixel (dp)以及屏幕像素密度**

应用程序可以用来定义UI组件的虚拟像素单元，通过密度无关的方式来描述布局尺寸和位置。一个设备独立像素相当于一个160 dpi屏幕上的物理像素。在程序运行时，系统根据屏幕的实际密度透明的处理任何需要缩放的设备独立像素单元，设备独立像素转换成屏幕实际像素的换算很简单：pixels = dps \* (density / 160).例如在240 dpi的屏幕上，1个设备独立像素等于1.5物理像素.为确保UI组件在不同的屏幕都能合适的展示，强烈建议使用设备独立像素单元来定义你的应用程序UI。

在API Level 18之后，android把不同屏幕像素密度的设备分成ldpi、mdpi、hdpi、xhdpi、xxhdpi、xxxhdpi，对应的dpi分别是：120dpi、160dpi、240dpi、320dpi、480dpi、640dpi，它们之间的缩放比例是3:4:6:8:12:16。一个在ldpi显示9x9像素的位图到mdpi的屏幕中就会变成12x12像素。

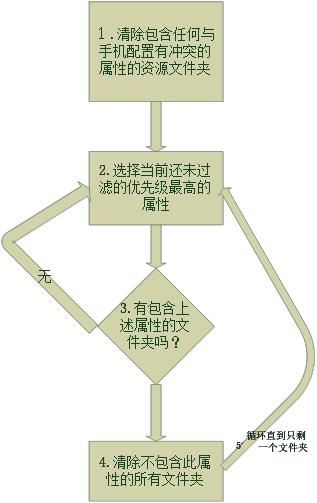
**3. Res下文件夹命名方式**  
　　命名方法与要求  
　　1) 命名不区分大小写;  
　　2) 命名形式：资源名-属性1-属性2-属性3-属性4-属性5.....  
　　资源名就是资源类型名，包括:drawable, values, layout, anim, raw, menu, color, animator, xml;  
属性1-属性2-属性3-属性4-属性5.....就是下列的属性集内的属性，如:-en-port-hdpi;  
　　注意：各属性的位置顺序必须遵守优先级从高到低排列!否则编译不过，属性值见下表，优先级由上到下递减。

|  |  |
| --- | --- |
| **限定符Qualifier** | **值Values** |
| 移动国家码MCC和移动网络码MNC | 手机设备SIM卡上的移动国家码和移动网络码。比如mcc310-mnc004 (美国，Verizon品牌); mcc208-mnc00 (法国,Orange品牌); mcc234-mnc00 (英国，BT品牌).  如果这个设备使用一个无线连接（GSM电话），则MCC来自SIM卡，而MNC来自该设备将要附着的网络。你有时会仅使用MCC，例如包含特定国家合法资源在您的应用程序中。如果您的应用程序指定了MCC/MNC组合的资源，这些资源仅在MCC和MNC都匹配的时候才能使用。 |
| 语言和区域Languageand region | 两个字母的[ISO639-1](http://www.loc.gov/standards/iso639-2/php/code_list.php)语言码和[ISO3166-1-alpha-2](http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html)区域码 (以"r"为前缀)。比如en-rUS,fr-rFR,es-rES.这个代码是大小写敏感的：语言码是小写字母，国家码是大写字母。你不能单独指定一个区域，但是你可以单独指定一个语言，比如en,fr, es, zh. |
| 屏幕方向Screenorientation | 纵向，横向，正方形（port,land, square） |
| 屏幕像素密度Screenpixel density | 92dpi,108dpi等. 当Android选择使用哪个资源时，它对屏幕像素密度的处理和其它限定符不同。在文章后面描述的步骤1[Android如何查找最匹配的目录](file:///\\home\D:\xfchen\ebook\android_dev_guide\developer.android.com\guide\topics\resources\resources-i18n.html#best-match)中，屏幕密度总被认为是匹配的。在步骤4中，如果被考虑的限定符是屏幕密度，Android将选择在那个位置的最佳匹配，而无需继续步骤5。 |
| 触摸屏类型Touchscreentype | 非触摸式，触摸笔，手指（notouch,stylus, finger） |
| 键盘可用方式Whetherthe keyboard is available to the user | 外在键盘，隐藏键盘，软键盘（keysexposed,keyshidden,keyssoft） 如果你的应用程序有一个特定的资源只能通过软件盘使用，则使用keyssoft 值，如果没有keyssoft 资源可用（只有keysexposed 和 keyshidden）并且该设备显示了一个软键盘，那么系统将使用keysexposed 资源。 |
| 首选文本输入方法Primarytext input method | 不支持按键,标准键盘,12键(nokeys,qwerty, 12key) |
| 首选非触摸式导航方法Primarynon-touchscreen navigation method | 不支持导航,滑板,跟踪球,滚轮(nonav,dpad, trackball, wheel) |
| 屏幕分辨率Screendimensions | 320x240,640x480, 等. 更大的分辨率必须先被指定。 |
| SDK版本SDKversion | 设备支持的SDK版本，比如v3。Android1.0SDK是v1，1.1SDK是v2，1.5SDK是v3。 |
| 小版本(Minorversion) | 你目前还不能指定小版本，它总是被设置为0。 |

　　**实例说明**　　1) 把全部属性都用上的例子(各属性是按优先级先后排列出来的)  
values-mcc310-en-sw320dp-w720dp-h720dp-large-long-port-car-night-ldpi-notouch-keysexposed-nokeys-navexposed-nonav-v7  
 2) 上述例子属性的中文说明  
values-mcc310(sim卡运营商)-en(语言)-sw320dp(屏幕最小宽度)-w720dp(屏幕最佳宽度)-h720dp(屏幕最佳高度)-large(屏幕尺寸)-long(屏幕长短边模式)-port(当前屏幕横竖屏显示模式)-car(dock模式)-night(白天或夜晚)-ldpi(屏幕最佳dpi)-notouch(触摸屏模类型)-keysexposed(键盘类型)-nokey(硬按键类型)-navexposed(方向键是否可用)-nonav(方向键类型)-v7(android版本)。

**4. 定位最佳文件夹**

当res下的资源文件按照上述命名要求命名后，android系统是根据如下定位算法来筛选适合当前设备的资源文件。

  
特殊说明：关于屏幕大小size相关的属性不在步骤１的过滤清除条件里（dpi属性和screen size属性）：  
1) 对于dpi属性系统的选择方法是"best match"－－－即如果没有找到准确的属性，就从最高dpi的属性文件夹开始往低dpi的文件夹找。例如：g7手机，应该是values-hdpi,但如果没有values-hdpi，则从最高dpi 的文件夹中找（先找values-xxhdpi到values-xhdpi到values-mdpi到values最后到values-ldpi,注意values在values-ldpi之前先找到，系统认为values比values-ldpi更接近我的values-hdpi属性）;  
  
2) 对于screen size属性系统的选择方法是"向下best match"，即如果没找到准确的属性，只可以接着在比自身属性小的文件夹里找最接近的属性文件夹。例如：g7手机，应该是values-normal,但如果没有values-normal,则可以找出最接近的文件夹(先找values再到values-small,注意，系统认为values比values-small更接近我的values-normal属性，但之后就不会再找values-large与values-xlarge了，因为是"向下best match"，large与xlarge都比normal大)。  
2. 实例说明  
工程有如下文件夹：  
drawable/  
drawable-en/  
drawable-fr-rCA/  
drawable-en-port/  
drawable-en-notouch-12key/  
drawable-port-ldpi/  
drawable-port-notouch-12key/  
手机属性：  
Locale = en-GB   
Screen orientation = port   
Screen pixel density = hdpi   
Touchscreen type = notouch   
Primary text input method = 12key  
  
1) 清除包含任何与手机配置有冲突的属性的资源文件夹（加粗字体被清除）  
drawable/  
drawable-en/  
**drawable-fr-rCA/**  
drawable-en-port/  
drawable-en-notouch-12key/  
drawable-port-ldpi/  
drawable-port-notouch-12key/  
注意：因为dpi属性使用"best match"选择，所以drawable-port-ldpi/  
没被清除。  
2) 选择最高优先级的属性（MCC最高，然后依次向下选择）。  
3)  有包含上述属性的文件夹吗？  
如果没有，跳转步骤２，选择下一优先级高的属性；  
如果有，继续步骤４；  
4) 清除不包含此属性的所有文件夹  
drawable/  
drawable-en/  
drawable-en-port/  
drawable-en-notouch-12key/  
drawable-port-ldpi/  
drawable-port-notouch-12key/  
5) 重复步骤２和３，直到步骤４中只剩下一个文件夹则返回。例子中，屏幕方向是下一个最高优先级的属性，所以可以清除两个文件夹：  
drawable-en/  
drawable-en-port/  
drawable-en-notouch-12key/  
最终找到的文件夹就是drawable-en-port/  
由上述过程可看出，匹配是看最高优先级别的属性，而不是一共匹配到的属性数量，例如：values-en-normal-hdpi的手机，values-en匹配比vaues-normal-hdpi更合适。

**一些技巧跟需要注意的地方：**

* 表格中列出的属性集并不是一直不变的，有些新属性是在高级的android version才加入的，系统在识别到此新属性时会自动加上支持新属性的android最低版本属性，例如：xhdpi是android 2.2(sdk新加入的，则values-xhdpi系统会自动认为是values-xhdpi-v8,这样新属性一定不会在android旧版本中匹配到了。
* 对于指定分辨率的属性(例如：values-hdpi-1024x600，values-hdpi-960x540，values-mdpi-1024x600)，指定分辨率属性并没出现在官网的匹配属性集里，也没找到对于分辨率属性的详细说明，但是使用这个属性很鸡肋，因为android系统的虚拟按键会占用屏幕的一部分，所以设备原来的分辨率是1024x600的结果因为虚拟按键就会使实际分辨率比1024x600低一点从而适配不到1024x600的资源文件。
* android从1.6和更高，Google为了方便开发者对于各种分辨率机型的移植而增加了自动适配的功能   
  <supports-screens   
  android:largeScreens="true"   
  android:normalScreens="true"   
  android:smallScreens="true"   
  android:anyDensity="true"/>   
  **是否支持多种不同密度的屏幕**android:anyDensity=["true" | "false"]   
  如果android:anyDensity="true"   
  指应用程序支持不同密度，会根据屏幕的分辨率自动去匹配。   
  如果android:anyDensity="false"   
  应用程序支持不同密度，系统自动缩放图片尺寸和这个图片的坐标。具体解释一下系统是如何自动缩放资源的。   
  例如我们在hdpi,mdpi,ldpi文件夹下拥有同一种资源，那么应用也不会自动地去相应文件夹下寻找资源，这种情况都是出现在高密度，以及低密度的手机上，比如说一部240×320像素的手机，   
  如果设置android:anyDensity="false"，Android系统会将240 x 320(低密度)转换为 320×480(中密度)，这样的话，应用就会在小密度手机上加载mdpi文件中的资源。   
  **是否支持大屏幕**  
  android:largeScreens=["true" | "false"]   
  如果在声明不支持的大屏幕，而这个屏幕尺寸是larger的话，系统使用尺寸为("normal")和密度为("medium)显示，   
  不过会出现一层黑色的背景。   
  **是否支持小屏幕**  
  android:smallScreens=["true" | "false"]   
  如果在声明不支持的小屏幕，而当前屏幕尺寸是smaller的话，系统也使用尺寸为("normal")和密度为("medium)显示   
  如果应用程序能在小屏幕上正确缩放(最低是small尺寸或最小宽度320dp)，那就不需要用到本属性。否则，就应该为最小屏幕宽度标识符设置本属性   
  来匹配应用程序所需的最小尺寸。
* Android系统采用下面两种方法来实现应用的自动适配：   
  1）布局文件中定义长度的时候，最好使用wrap\_content,fill\_parent, 或者dp 进行描述，这样可以保证在屏幕上面展示的时候有合适的大小   
  2）为不同屏幕密度的手机，提供不同的位图资源，可以使得界面清晰无缩放。   
  对应bitmap 资源来说，自动的缩放有时会造成放大缩小后的图像变得模糊不清，这是就需要应用为不同屏幕密度配置提供不同的资源：为高密度的屏幕提供高清晰度的图像等。   
  3）不要使用AbsoluteLayout   
  4）像素单位都使用DIP，文本单位使用SP
* **在代码中获取屏幕像素、屏幕密度**   
  DisplayMetrics metric = new DisplayMetrics();   
  getWindowManager().getDefaultDisplay().getMetrics(metric);   
  int width = metric.widthPixels; // 屏幕宽度（像素）   
  int height = metric.heightPixels; // 屏幕高度（像素）   
  float density = metric.density; // 屏幕密度（0.75 / 1.0 / 1.5）   
  int densityDpi = metric.densityDpi; // 屏幕密度DPI（120 / 160 / 240）
* **TextView.setText方法中的参数单位是DP。**