

移动端网页设计

----设计部 黄卉 2016.05.12 -06.27

移动端网页设计

- ▶ 1、Web App、Native App和Hybrid App
- ▶ 2、Web App中的rem单位
- ▶ 3、Android开发中的dp和sp单位
- ▶ 4、Android前端开发注意事项

1、*Web App*、*Native App*和 *Hybrid App*

- ▶ 1.1 三者的基本含义
- ▶ 1.2 三者的对比
- ▶ 1.3 如何判断手机app中哪些是Web App，哪些是Native App

1.1 三者的基本含义

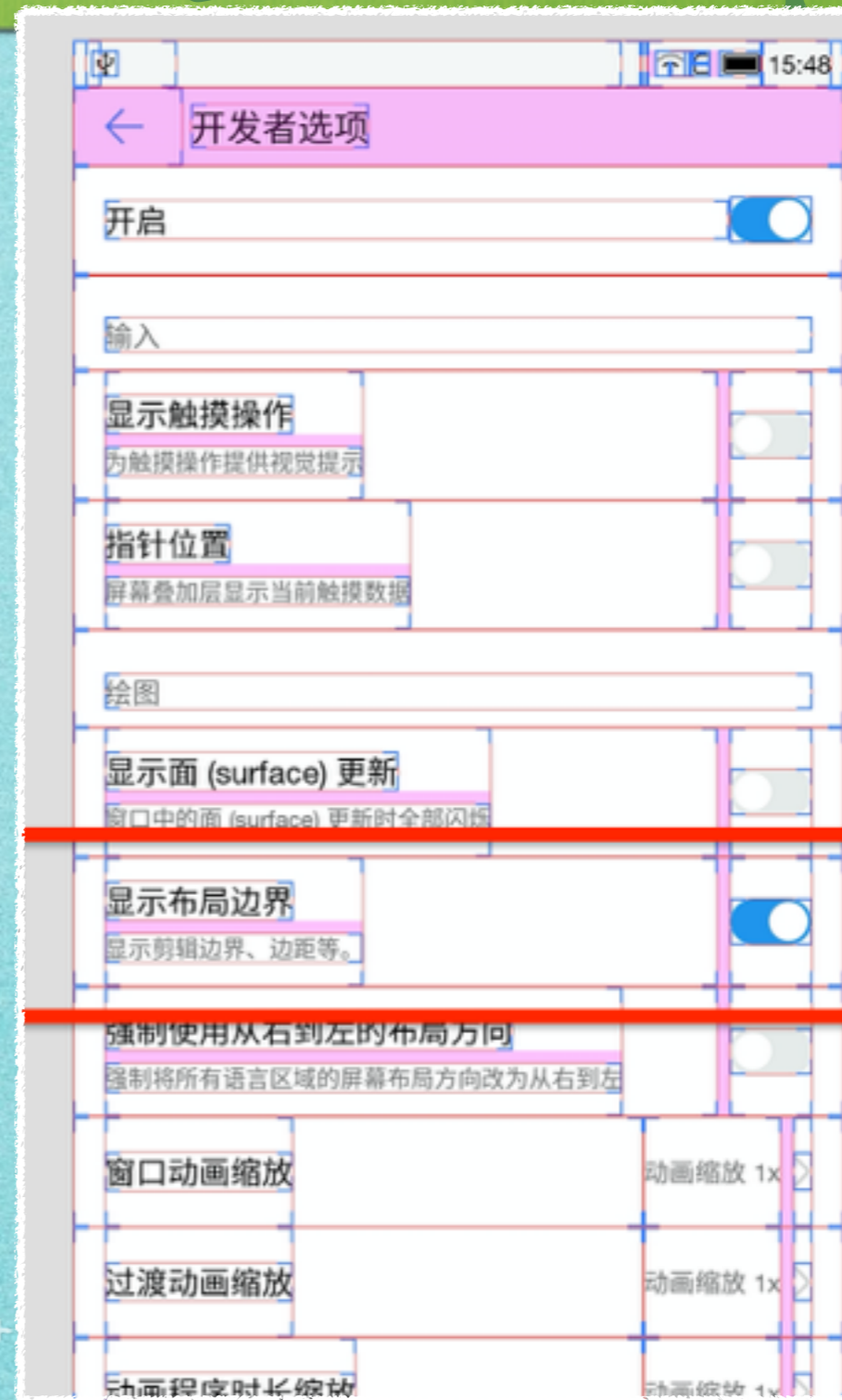
- ▶ Web App（网页应用）：Web App就是运行于网络 and 标准浏览器上，基于网页技术开发实现特定功能的应用，也就是通过html来实现的基于WEB的应用。
- ▶ Native App（原生应用）：Native App是一种基于智能手机本地操作系统如iOS、Android、WP并使用原生程式编写运行的第三方应用程序,也叫本地app。一般使用的开发语言为JAVA、C++、Objective-C。
- ▶ Hybrid App（混合应用）：Hybrid App是指介于web-app、native-app这两者之间的app,它虽然看上去是一个Native App，但只有一个UI WebView，里面访问的是一个Web App。

1.2 三者的对比

	Web App	Hybrid App	Native App
开发成本	低	中	高
维护更新	简单	简单	复杂
体验	差	中	优
Store或者Mark认可	不认可	认可	认可
安装	不需要	需要	需要

1.3 如何判断手机app中哪些是Web App，哪些是Native App

- ▶ 以乐视1s为例：“设置”-»“开发者选项”-»勾选“显示布局边界”，具体见右图



app首页，native app



帮助中心，web app



“今日头条”app首页



“今日头条”具体新闻页面



- ▶ 总结：区分app中的web App和Native App
- ▶ 终极目标：实现手机跨平台！
- ▶ 推荐学习新技术：ReactNative或者是HBuilder推出的一些html5框架可以实现跨平台！

2、*Web App*中的*rem*单位

- ▶ 2.1 了解手机端浏览器
- ▶ 2.2 rem是什么
- ▶ 2.3 rem作用
- ▶ 2.4 viewport用法
- ▶ 2.5 rem如何使用

2.1 了解手机端浏览器

- ▶ 现有四大内核浏览器：Trident (IE浏览器)、Gecko(FireFox)、Webkit (Chrome/ Safari / UC)、Presto: (Opera)
- ▶ 手机浏览器：UC、百度、欧朋、QQ、海豚、safari、Chrome，这些浏览器都是基于webkit内核的。
- ▶ WebApp用chrome调试即可，但具体情况具体分析。

2.2 rem是什么

- ▶ rem (font size of the root element) 是指相对于根元素的字体大小的单位。简单的说它就是一个相对单位。
- ▶ em (font size of the element) 是指相对于父元素的字体大小的单位。

2.3 rem的作用

- ▶ rem作用：实现强大的屏幕适配布局
- ▶ rem主要针对web app，web page也可以用

rem

流式
布局

固定
宽度

响应式
布局

viewport

流式布局

- ▶ 就是采用百分比来作为单位确定每一块的width，高度用px来固定住。虽然可以让各种屏幕都适配，但是显示的效果极其的不好。只有几个尺寸的手机能够显示完美的效果。（亚马逊、携程、兰亭）

固定宽度

- ▶ 网站设置固定的宽度，超出部分留白。大屏幕下页面小，小屏幕下页面超出有横滚动条。

响应式布局

- ▶ 一般大型企业复杂性的网站不使用方法（工作量大、维护性难），只有中小型的门户或者博客类站点才会采用该方法从web page到web app一步到位。

viewport

- ▶ 设置viewport进行缩放，也非常高效，但部分元素会不清晰。（天猫webApp首页）

- ▶ rem实现方法，修改简单，快速、高效，修改一个元素，能看到不同分辨率下的显示效果不同。
- ▶ 设置viewport，实际元素看起来和屏幕一样
- ▶ 设置media query，不同分辨率下的效果设置

2.4 viewport用法

- ▶ viewport用法如下：

```
<meta name="viewport"
  content="
    height = [pixel_value | device-height] ,
    width = [pixel_value | device-width] ,
    initial-scale = float_value ,
    minimum-scale = float_value ,
    maximum-scale = float_value ,
    user-scalable = [yes | no] ,
    target-densitydpi = [dpi_value | device-dpi | high-dpi | medium-dpi | low-dpi]"
/>
```

- ▶ app中web app每个页面都用到了viewport，具体代码如下：
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">

▶ viewport中content属性列表


content属性	说明
width 和 height	控制 viewport 大小，可以指定一个值或者特殊的值
initial-scale	初始缩放，即页面初始缩放程度。浮点数，是页面大小的一个乘数。
minnum-scale maximum-scale	最小（大）缩放，即页面允许的最大缩放程度。浮点数，用以指出页面大小与屏幕大小相比的最大乘数。
user-scalable	用户调整缩放，即用户是否能改变页面缩放程度。 yes 是允许用户对其进行改变，反之为 no 。
target-densitydpi	屏幕像素密度，由屏幕分辨率决定的，通常定义为每个英寸点的数量（ dpi ）

▶ 设置了viewport和不设置viewport的区别，以学信网为例



2.5 rem如何使用

```
html{
  font-size:20px;
}
.btn {
  width: 6rem;
  height: 3rem;
  line-height: 3rem;
  font-size: 1.2rem;
  display: inline-block;
  background: #06c;
  color: #fff;
  border-radius: .5rem;
  text-decoration: none;
  text-align: center;
}
</style>
</head>
<body>
<input type='button' class='btn' value='TestBtn' />
</body>
```



- ▶ rem是通过根元素（html）进行屏幕适配的，设置html的font-size控制其他dom元素的大小
- ▶ webkit, html的font-size默认为16px

▶ 如何计算不同分辨率下font-size的值

- ▶ 现在设计稿基本是按照640px的宽度作为标准尺寸（其实这个尺寸不一定是640，可以是320，或者480，又或是375，可变动，只是参考值）。
- ▶ 这里的屏幕对比比例也是可以改变的，并不是一定根据这个来，而是根据实际设计图样式来定。

宽度	320	384	480	640
屏幕对比比例	0.5	0.6	0.75	1
Html font-size	10px	12px	15px	20px
元素宽度(px)	100px	120px	150px	200px
元素宽度(rem)	10rem	10rem	10rem	10rem

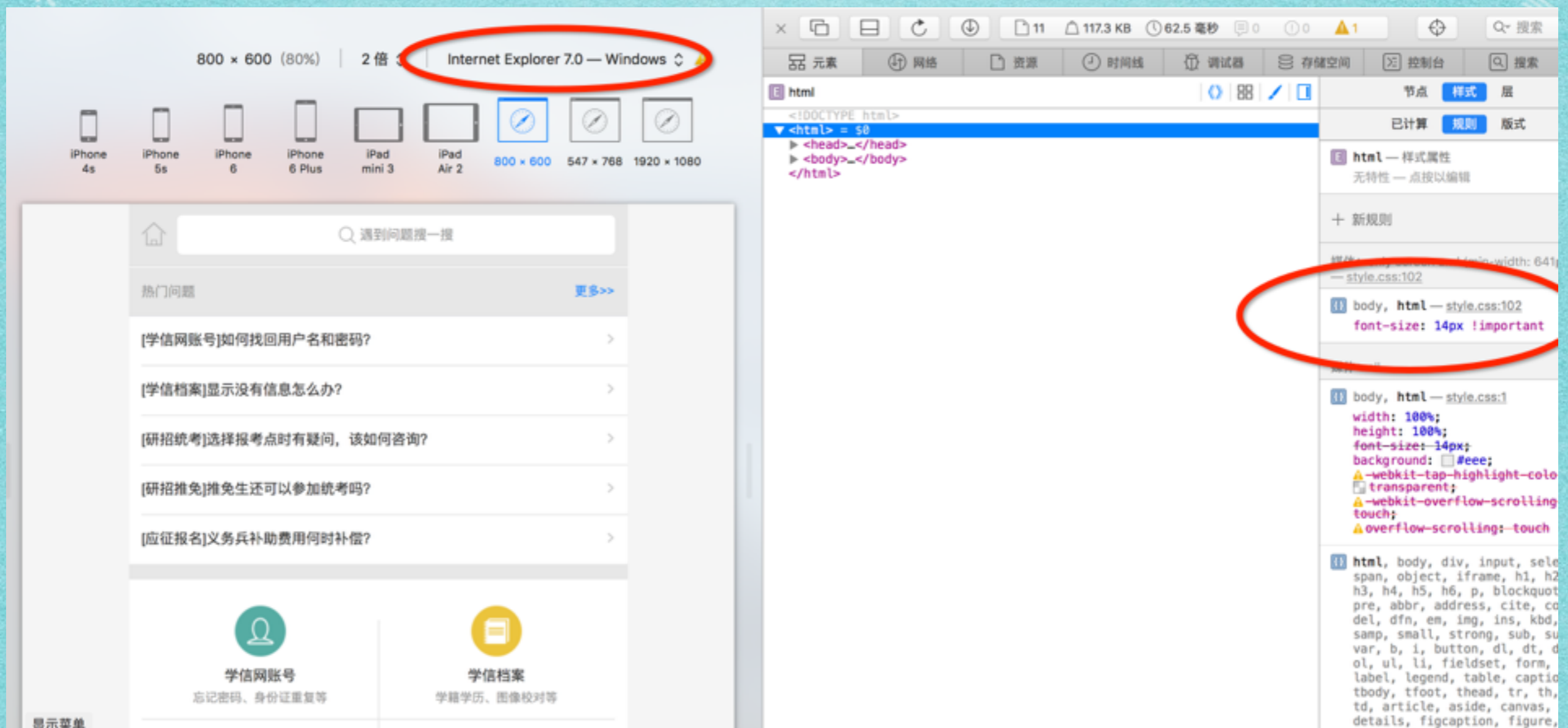
▶ 计算font-size的方法

- ▶ js计算：根元素的font-size可以通过js来动态计算，这样的好处是所有设备分辨率都能兼容适配，淘宝首页目前就是用的JS计算。
- ▶ media：但其实不用JS也可以做适配，一般我们在做web app都会先统计自己网站有哪些主流的设备，然后去针对那些设备去做media query设置也可以实现适配，例如右侧。

```
html {  
  font-size : 20px;  
}  
@media only screen and (min-width: 401px){  
  html {  
    font-size: 25px !important;  
  }  
}  
@media only screen and (min-width: 428px){  
  html {  
    font-size: 26.75px !important;  
  }  
}  
@media only screen and (min-width: 481px){  
  html {  
    font-size: 30px !important;  
  }  
}  
@media only screen and (min-width: 569px){  
  html {  
    font-size: 35px !important;  
  }  
}  
@media only screen and (min-width: 641px){  
  html {  
    font-size: 40px !important;  
  }  
}
```

▶ 解决rem兼容性问题IE7、8:

▶ `<!--[if lt IE 9]><script src="http://t3.chei.com.cn/common/wap/help/js/respond.js"></script><![endif]-->`



▶ WebApp开发总结总结：

- ▶ 先设置viewport，让页面的width等于设备的width
- ▶ 然后设置html的font-size为一个基础值
- ▶ 再根据实际情况来，设置页面后MediaQuery
- ▶ MediaQuery兼容性处理
- ▶ 推荐学习：flexbox弹性盒子布局和触摸touch事件Swiper插件！

3、*Android*开发中的 *dp*和*sp*单位

- ▶ 3.1 手机类型和尺寸
- ▶ 3.2 pc端常用单位px
- ▶ 3.3 长度单位dp
- ▶ 3.4 字体单位sp

3.1 手机类型和尺寸

- ▶ Android开发自动生成的5个尺寸文件：Ldpi、Mdpi、Hdpi、XHdpi、XXhdpi
- ▶ 同时也是5种典型的手机屏幕和尺寸，具体如下：

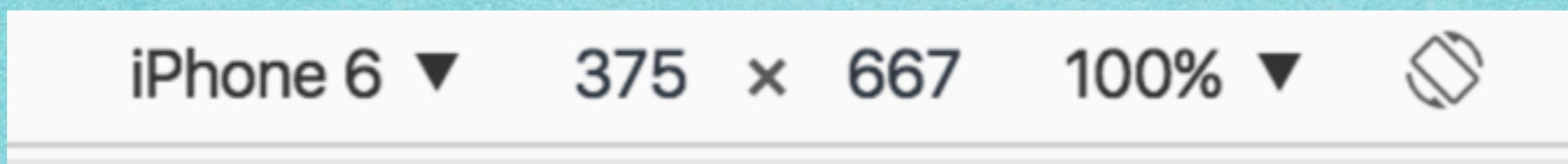


- ▶ 这里的px、dpi是什么意思？

▶ iPhone6s和iPhone6plus的对比

▶ $750 \times 1334 \rightarrow 375 \times 667$, 2倍, 2就是dpr, 密度系数

▶ $\sqrt{(750^2 + 1334^2)} / 4.7 = 326 \text{ ppi}$



▶ ppi和dpi, 都是单位英寸的像素密度

▶ $\text{ppi} = \sqrt{(\text{长度像素数}^2 + \text{宽度像素数}^2)} / \text{屏幕英寸}$

型号	分辨率 (px)	屏幕尺寸	PPI
iPhone4/4s/4c	640*960	3.5	326
iPhone 5s	640*1136	4.0	326
iPhone 6	750*1334	4.7	326
iPhone 6 Plus	1080*1920	5.5	401
红米Note	720*1280	5.5	267
红米手机/1s	720*1280	4.7	312
三星 S4	1080*1920	5	441
三星 Note3	1080*1920	5.7	373

- ▶ **px像素**： 屏幕上一个物理像素点，不同设备显示效果不同；
- ▶ **分辨率**： 手机屏幕垂直和水平方向上的像素个数，
640px*960px；
- ▶ **屏幕尺寸**： 指对角线长度，而不是手机长度，3.5英寸；
- ▶ **ppi**： 屏幕密度标准，数字影像的解析度，每英寸所拥有的像素数，120，160，240，320，480等。ppi越高图片越清晰
- ▶ **dpi**： 是印刷上的计量单位，每个英寸上，所能印刷的网点数。即每英寸像素数量。和ppi表达意思一样，ppi是用在计算机屏幕上，dpi用到具体的印刷设备上。

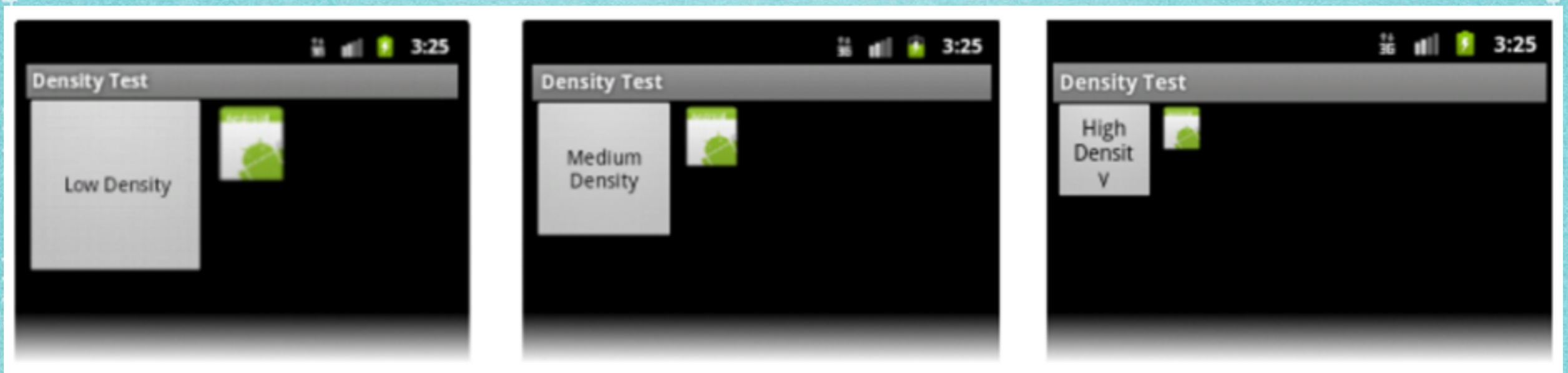
	ldpi	mdpi	hdpi	xhdpi	xxhdpi
ppi	120	160	240	320	480
默认缩放比	0.75	1.0	1.5	2.0	3.0

▶ dp和px之间的换算

				
1dp=0.75px	1dp=1px	1dp=1.5px	1dp=2px	1dp=3px
Ldpi 120dpi	Mdpi 160dpi	Hdpi 240dpi	XHdpi 320dpi	XXHdpi 480dpi
			对应的设计尺寸 720*1280	对应的设计尺寸 1080*1920

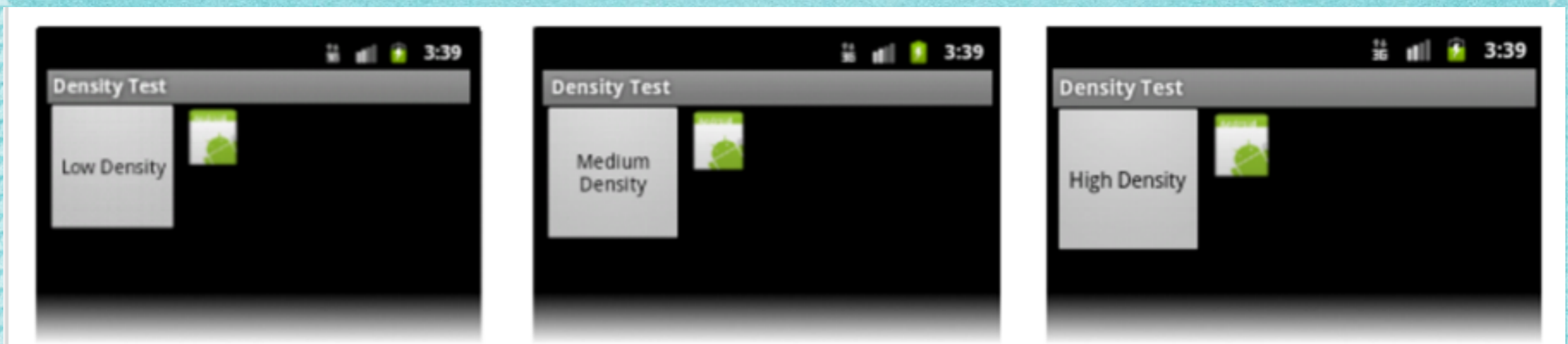
3.2 pc端常用单位px

- ▶ px: 即像素, 1px代表屏幕上一个物理的像素点;
- ▶ px单位在Android开发中不被建议使用, 因为同样100px的图片, 在不同手机上显示的实际大小可能不同, 如下图所示。



3.3 长度单位dp

- ▶ dp是界面设计的尺寸规范标注中，作为间距单位的。
- ▶ 不同移动设备上显示效果相同，因为每个移动设备的ppi不同，px不同，但占手机屏幕比例相同。



- ▶ 当ppi=320时，1dp=2px，ppi不同px不同，才能确保不同浏览器下显示效果相同。

dp	8	16	24	32	48	72
px (120dpi)	6	12	18	24	36	54
px (160dpi)	8	16	24	32	48	72
px (240dpi)	12	24	36	48	72	108
px (320dpi)	16	32	48	64	96	144

注：从表中可以看出，除去低分辨率的屏幕（120dpi）在视觉设计中，要求所有元素必须是可以被4整除的元件。

3.4 字体单位sp

- ▶ sp: 与缩放无关的抽象像素 (Scale-independent Pixel)。
- ▶ sp和dp很类似, 但唯一的区别是, Android系统允许用户自定义文字尺寸大小: 小、正常、大、超大等等。
- ▶ 当文字尺寸是“正常”时 $1\text{sp}=1\text{dp}=0.00625$ 英寸, 而当文字尺寸是“大”或“超大”时, $1\text{sp}>1\text{dp}=0.00625$ 英寸。类似我们在windows里调整字体尺寸以后的效果——窗口大小不变, 只有文字大小改变。

▶ sp和dp换算方法相同，当ppi=320时，1sp=2dp

默认的界面规格	480*800, PPI=240	720*1280, PPI=320	1080*1920px ppi=480
Text Size Micro	18px=12sp	24px=12sp	36px = 12sp
Text Size Small	21px=14sp	28px=14sp	42px = 14sp
		32px=16sp	48px = 16sp
Text Size Medium	27px=18sp	36px=18sp	54px = 18sp
		40px=20sp	60px = 20sp
Text Size Large	33px=22sp	44px=22sp	66px = 22sp

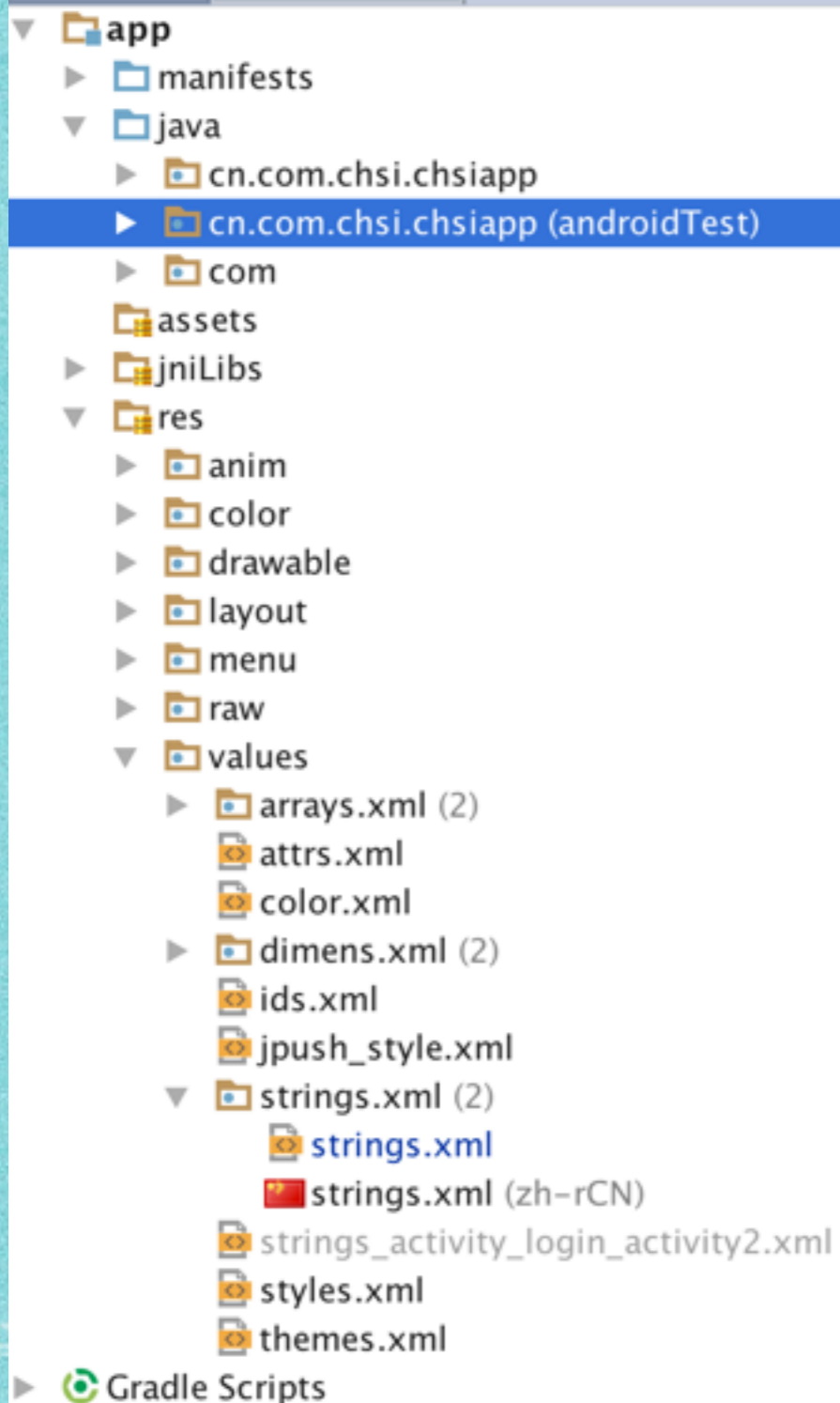
▶ 总结:

- ▶ Android手机尺寸有5种, Ldpi、Mdpi、Hdpi、XHdpi、XXhdpi, 对应的默认缩放比为0.75、1.0、1.5、2.0、3.0
- ▶ $ppi = \sqrt{(\text{长度像素数}^2 + \text{宽度像素数}^2)} / \text{屏幕英寸}$, ppi越高屏幕越清晰。Android手机对应ppi: 120、160、240、320、480.
- ▶ 在android开发中, 确保不同手机上显示效果一样, 间距单位用dp, 字体大小单位用sp。

4、*Android*前端开发注意事项

- ▶ 4、1 项目文档结构及作用
- ▶ 4、2 什么是.9图片
- ▶ 4、3 如何写xml布局文件
- ▶ 4、4 常用控件
- ▶ 4、5 控件必填项
- ▶ 4、6 常用属性值
- ▶ 4、7 常用的5大布局
- ▶ 4、8 布局示例

4.1 项目文档结构及作用



- ▶ java代码里面实现js事件触发等方法
- ▶ res/anim, 动画文件
- ▶ res/color, 文本颜色textColor相关的文件
- ▶ res/drawable, 放背景background属性相关的文件和图片
- ▶ res/layout里面放的xml文件==html布局文件
- ▶ res/mens, 菜单属性, 引导页、启动页, 每个页面的title等
- ▶ res/values, 下面的文件, 如下:
 - ▶ arrays.xml, 测试用的数据来源
 - ▶ color.xml, 放所有颜色相关的文件
 - ▶ dimens.xml, 放尺寸的文件
 - ▶ string.xml是国际化的地方, 中英文
 - ▶ styles.xml == css文件
 - ▶ themes.xml 放app主题配置相关的文件

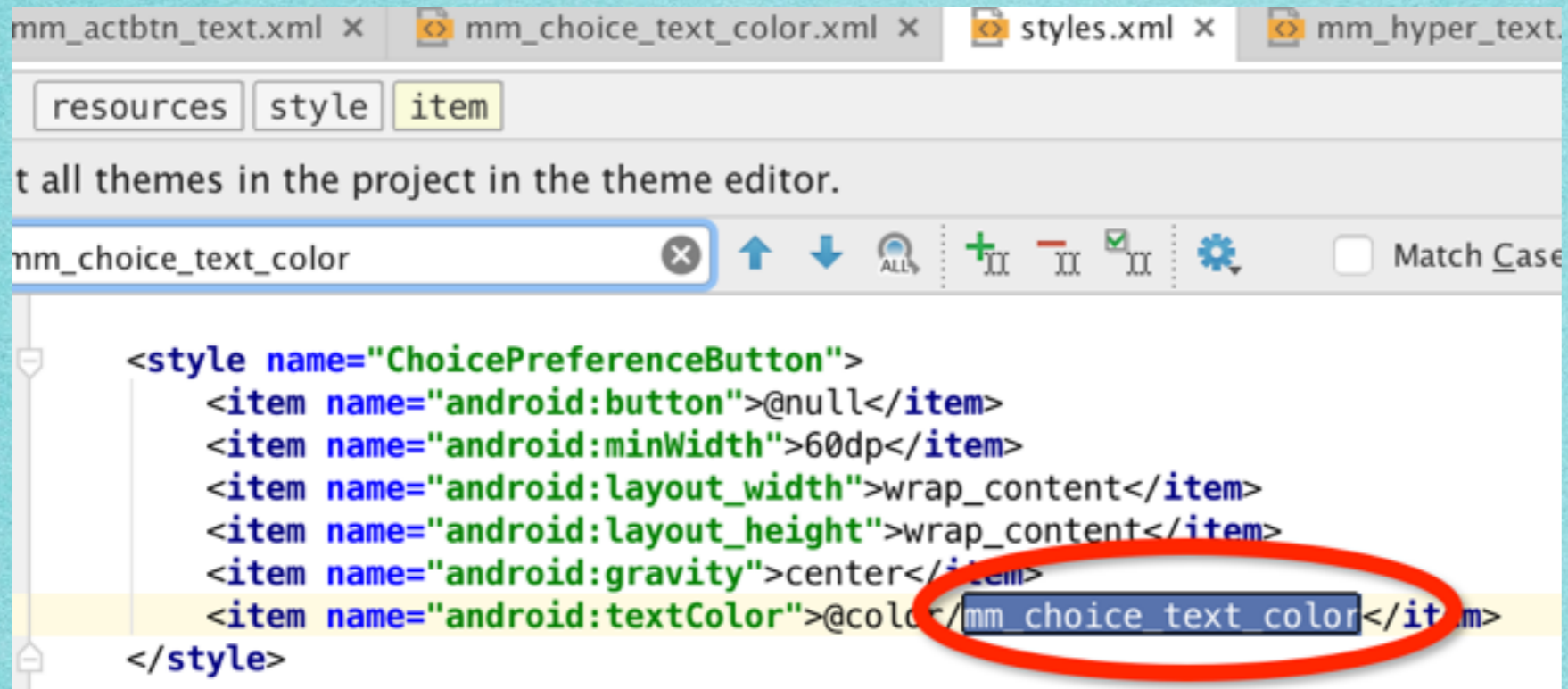
- ▶ res/color里面的xml文件和 res/values/color.xml文件的区别（前者有状态，后者只有样式）
- ▶ res/color/mm_choice_text_color.xml引用的地方，在res/values/styles.xml里面，通过 android:textColor引用，也可以在布局文件中引用
- ▶ 引用方式都是@color/（res/color/文件名）或者(res/values/color.xml中定义item的文件名)

```
mm_actbtn_text.xml x mm_choice_text_color.xml x mm_hyper_text.xml  
  
<?xml version="1.0" encoding="UTF-8"?>  
<selector  
  xmlns:android="http://schemas.android.com/apk/res/android">  
  <item  
    android:state_selected="true"  
    android:color="@color/white" />  
  <item  
    android:state_checked="true"  
    android:color="@color/white" />  
  <item  
    android:color="@color/black" />  
</selector>
```

```
color.xml x  
resources color  
<?xml version="1.0" encoding="UTF-8"?>  
<resources>  
  <color name="result_view">#b0000000</color>  
  <color name="viewfinder_mask">#60000000</color>  
  <color name="possible_result_points">#c0ffff00</color>  
  <color type="drawable" name="action_bar">#36aaa4</color>  
  <color type="drawable" name="share_background">#ffaafffb</color>  
  <color name="white">#ffffff</color>  
  <color name="black">#000000</color>  
  <color name="darkgrey">#ff333333</color>  
  <color name="grey">#666666</color>  
  <color name="bottomgrey">#5f6971</color>  
  <color name="blue">#ffccddff</color>  
  <color name="sgrey">#ffdddddd</color>  
  <color name="slightgrey">#ffcccccc</color>  
  <color name="toasterro">#9fd524</color>  
  <color name="lightgrey">#ff888888</color>  
  <color name="semitransparent">#80000000</color>  
  <color name="lightransparent">#a0000000</color>  
  <color name="transparent">#00000000</color>  
  <color name="navpage">#FFE1E8EB</color>  
  <color name="green">#ff3fa5aa</color>  
  <color name="lightgreen">#ff5aaeb2</color>  
  <color name="background_tab_pressed">#663385E5</color>  
</resources>
```

color/drawable 包下一些属性	说明
android:state_pressed	是否按下，如一个按钮触摸或者点击。
android:state_focused	是否取得焦点，如选择了一个文本框
android:state_hovered	光标是否悬停， 4.0 的新特性
android:state_selected	被选中
android:state_checkable	组件是否能被 check
android:state_checked	被 checked 了
android:state_enabled	能够接受触摸或者点击事件
android:state_activated	被激活
android:state_window_focused	应用程序是否在前台

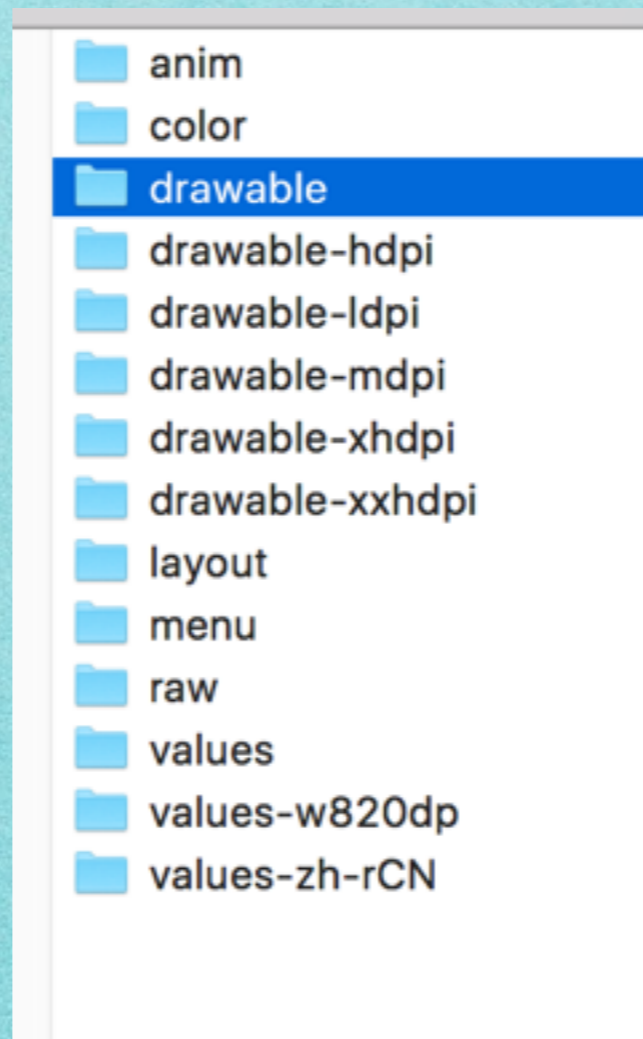
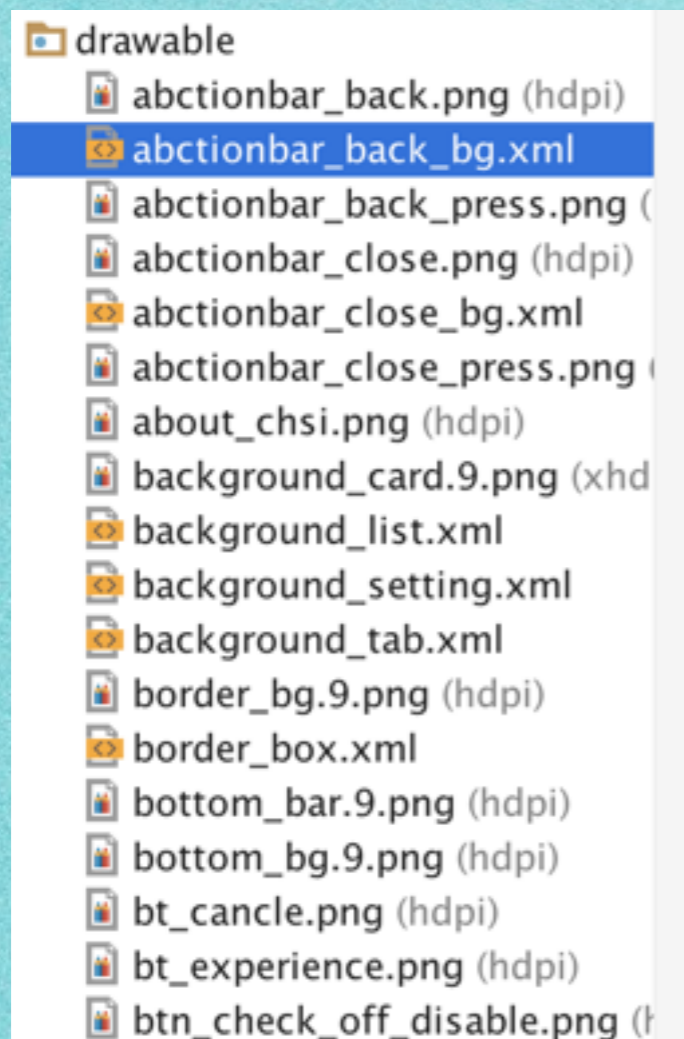
▶ res/color里面的xml文件如何引用?



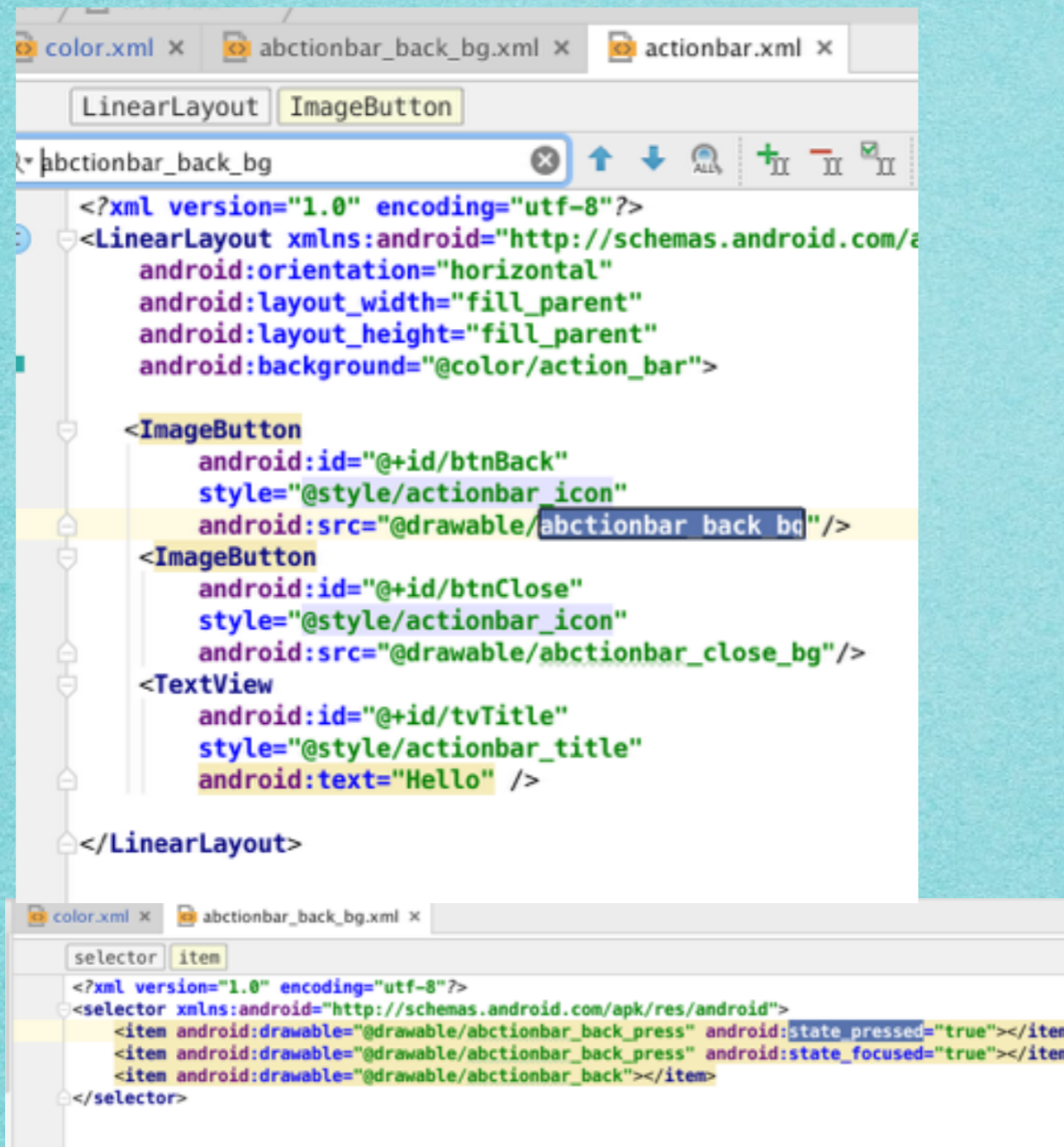
```
resources style item
t all themes in the project in the theme editor.
mm_choice_text_color
<style name="ChoicePreferenceButton">
  <item name="android:button">@null</item>
  <item name="android:minWidth">60dp</item>
  <item name="android:layout_width">wrap_content</item>
  <item name="android:layout_height">wrap_content</item>
  <item name="android:gravity">center</item>
  <item name="android:textColor">@color/mm_choice_text_color</item>
</style>
```

```
<TableRow
  style="@style/index_table_row">
  <TextView
    android:text="学信网"
    style="@style/index_table_text"
    android:textSize="16sp"
    android:textColor="@color/mm_actbtn_text"
  />
  <TextView
    android:text="http://www.chsi.com.cn"
    style="@style/index_table_text"
    android:textColor="@color/white"/>
</TableRow>
```

- ▶ drawable, 实际有6个文件夹
- ▶ drawable放xml文件, background属性
- ▶ drawable-hdpi、ldpi、mdpi、xhdpi、xxhdpi, 5类文件夹放png图标, 5中尺寸。
- ▶ drawable-hdpi除了放png图片外, 还放默认的.9图片



- ▶ res/drawable里面的xml文件如何引用，android:src或者是android:background



The image shows two screenshots from an IDE. The top screenshot displays the XML code for an ActionBar background, where the `android:src` attribute of an `ImageButton` is highlighted. The bottom screenshot shows a selector XML file with three `item` elements, where the `android:drawable` attribute is highlighted.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/action_bar">

    <ImageButton
        android:id="@+id/btnBack"
        style="@style/actionbar_icon"
        android:src="@drawable/abactionbar_back_bg"/>

    <ImageButton
        android:id="@+id/btnClose"
        style="@style/actionbar_icon"
        android:src="@drawable/abactionbar_close_bg"/>

    <TextView
        android:id="@+id/tvTitle"
        style="@style/actionbar_title"
        android:text="Hello" />

</LinearLayout>
```

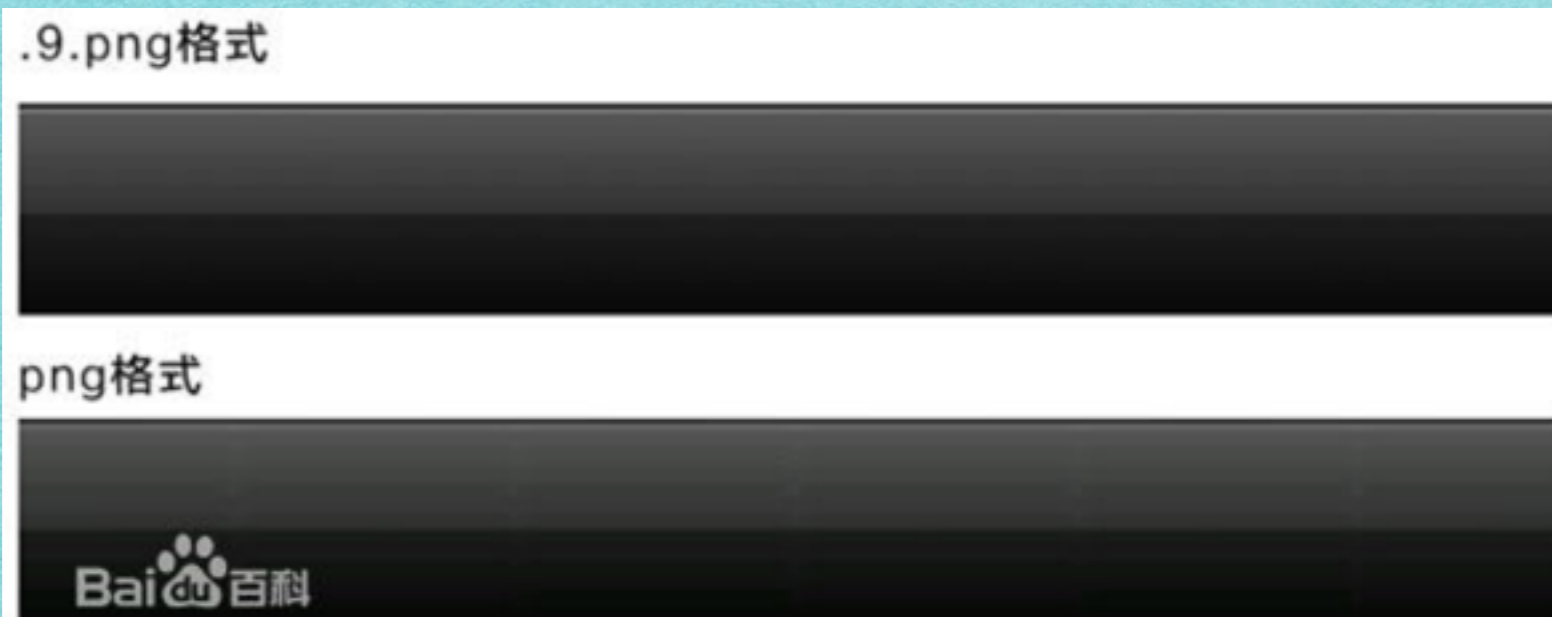
```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@drawable/abactionbar_back_press" android:state_pressed="true"></item>
    <item android:drawable="@drawable/abactionbar_back_press" android:state_focused="true"></item>
    <item android:drawable="@drawable/abactionbar_back"></item>
</selector>
```

4.2 什么是.9的图片

- ▶ 点九即 .9 ，是andriod平台的应用软件开发里的一种特殊的图片形式，文件扩展名为：.9.png



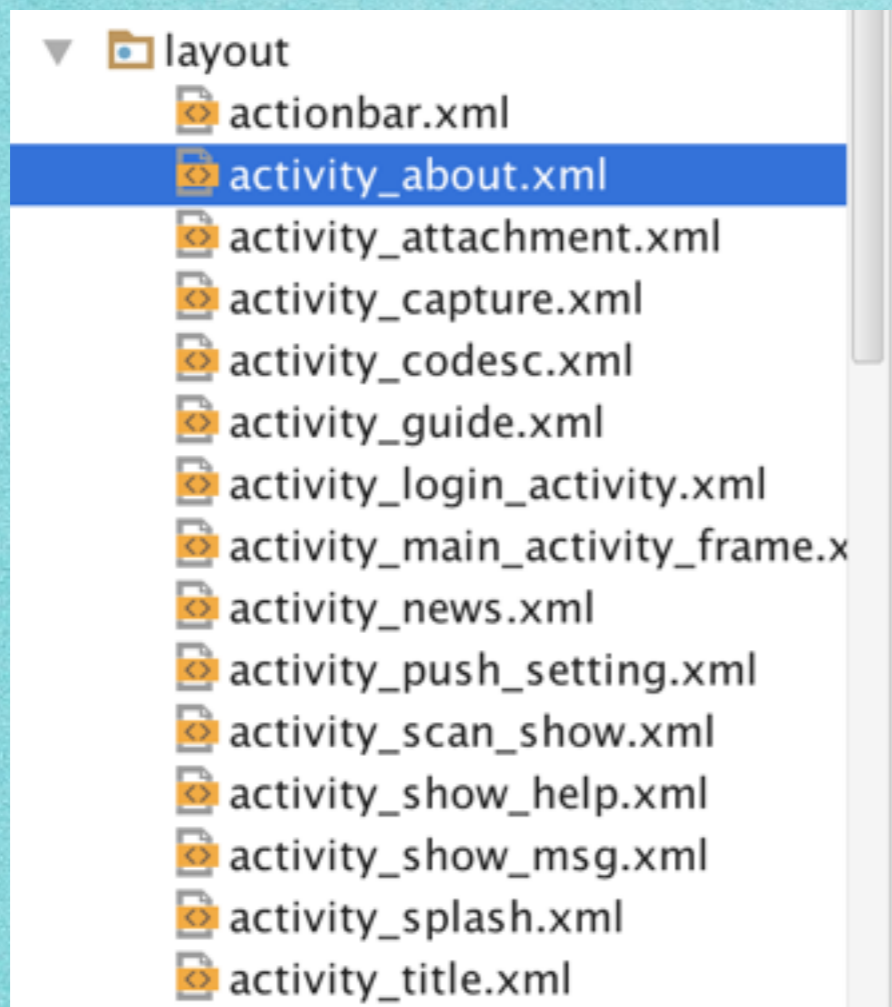
- ▶ 作用：在图片拉伸的时候特定的区域不会发生图片失真



- ▶ 如何绘制：
 - ▶ 利用Android开发中的工具draw9patch制作
 - ▶ 利用photoshop制作
- ▶ 疑问，能否放.jpg?
- ▶ 具体参考：《.9图片讲解》 <http://www.cnblogs.com/vanezkw/archive/2012/07/19/2599092.html>

4.3 如何写xml布局文件

- ▶ 布局文件，也就是在res/layout下面的xml文件，一个页面一个xml文件。



```
activity_login_activity.xml
color.xml x activity_about.xml x activity_login_activity.xml x abctionbar_bar

LinearLayout ScrollView LinearLayout

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:paddingBottom="@0dp"
    android:paddingLeft="@0dp"
    android:paddingRight="@0dp"
    android:paddingTop="@0dp">

    <!-- Login progress -->
    <ProgressBar
        android:id="@+id/login_progress"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginTop="100dp"
        android:visibility="gone" />

    <ScrollView
        android:id="@+id/login_form"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:id="@+id/email_login_form"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <AutoCompleteTextView
                android:id="@+id/email"
                style="@style/login_input"
                android:layout_marginTop="15dp"
                android:hint="邮箱/手机号/身份证号"
                android:inputType="textEmailAddress" />

        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

```
<View
    android:layout_width="fill_parent"
    android:layout_height="1dp"
    android:background="#eeeeee"
    android:visibility="visible" />

<EditText
    android:id="@+id/password"
    style="@style/login_input"
    android:hint="密码"
    android:imeActionId="@+id/login"
    android:imeActionLabel="登录"
    android:imeOptions="actionUnspecified"
    android:inputType="textPassword" />

<View
    android:layout_width="fill_parent"
    android:layout_height="1dp"
    android:background="#eeeeee"
    android:visibility="visible" />

<Button
    android:id="@+id/email_sign_in_button"
    style="@style/btn_login"
    android:background="@drawable/login_btn"
    android:text="登录" />

</LinearLayout>
</ScrollView>
</LinearLayout>
```

- ▶ LinearLayout布局，线性布局
- ▶ ProgressBar，正在加载中
- ▶ ScrollView，滚动区域
- ▶ LinearLayout布局里面为具体的元素：AutoCompleteTextView、view、EditText、Button

4.4 常用控件

- ▶ app中现用的控件和对应的html
- ▶ 这里只是用到了一些常规的控件，基于android2.2版本的控件
- ▶ 如果是用Android Material Design，则根据引入的库不一样，用到的控件和效果也不一样

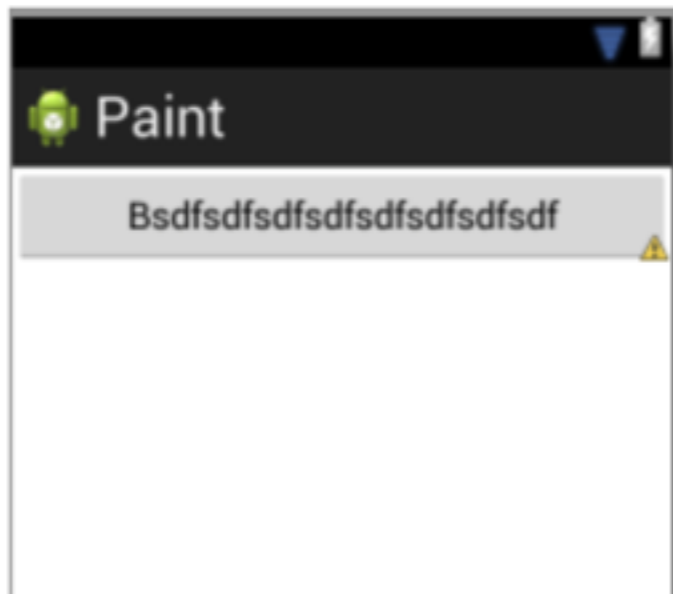
控件名	描述	html
TextView	显示文本，不允许编辑 (android:autoLink)	p、span
EditText	可编辑输入框	input、textarea
Button、RadioButton、 ImageButton、ButtonGroup	按钮、单选按钮、 图片按钮、按钮组	button、type='radio'
ImageView	图片	img
TableRow	tableLayout 里面的行	tr
WebView、ListView	引入 jsp 、 xml 页面的区域	include
View、ScrollView	视图区域、滚动区域	-
ProgressBar	正在加载中	-
ToggleSwitch、Switch	开关、滑块	-

4.4 控件必填项

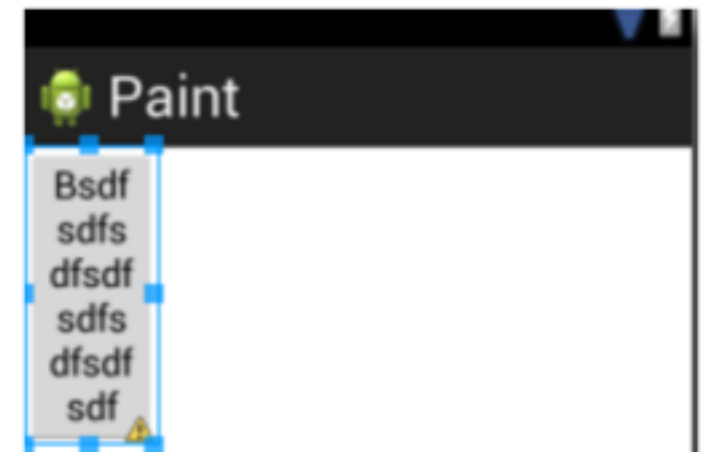
- ▶ layout_width和layout_height属性值（必填）：“match_parent”、“wrap_content”、“fill_parent”
 - ▶ “match_parent”和“fill_parent”实际上是一样的，Android2.2之后“fill_parent”改为“match_parent”
 - ▶ match_parent: 表示填充父元素
 - ▶ wrap_content: 表示大小刚好足够显示当前控件里的内容

- ▶ layout_width和width属性区别：只有当其属性为wrap_parent时，width属性才有效。

```
<Button  
  android:id="@+id/button1"  
  android:layout_width="match_parent"  
  android:width="100dp"  
  android:layout_height="wrap_content"  
  android:layout_gravity="center"  
  android:gravity="right"  
  android:text="Bsdfsdfsdfsdfsdfsdfsdf" />
```



```
<Button  
  android:id="@+id/button1"  
  android:layout_width="wrap_content"  
  android:width="50dp"  
  android:layout_height="wrap_content"  
  android:text="Bsdfsdfsdfsdfsdfsdfsdf" />
```



4.6 常用属性

- ▶ 只列出了app中常用的属性值和对应的css
- ▶ 每个控件都有对于的特殊属性
- ▶ 其他特殊的属性，可以进一步研究

属性名	描述	取值	CSS
android:background、src	背景	@drawable/	background
android:textSize	文字大小	sp	font-size (px)
android:textColor	字体颜色	#FFcccccc、red、@color	color
android:padding	内边距	dp	padding (px)
android:layout_margin	外边距	dp	margin (px)
android:visibility	是否可见	gone、visible、invisible	display:none/visibility:visible、hidden
android:singleline	单行显示	true、false	white-space: nowrap;
android:ellipsize	省略显示	end、start、middle、marquee	text-overflow: ellipsis

4.7 常用的5大布局

- ▶ 1、LinearLayout: 线性布局，分垂直布局（`android:orientation="vertical"`）和水平布局（`android:orientation="horizontal"`），在LinearLayout里面可以放多个控件，但是一行（列）只能放一个控件。
- ▶ 2、FrameLayout: 单帧布局，所有控件都放置在屏幕左上角（0,0），可以放多个控件，但是会按控件定义的先后顺序依次覆盖，后一个会直接覆盖在前一个之上显示，如果后放的比之前的大，会把之前的全部盖住（类似于一层层的纸张）。
- ▶ 3、AbsoluteLayout: 绝对布局，可以直接指定子控件的绝对位置（例如：`android:layout_x="60dp" android:layout_y="32dp"`），这种布局简单直接，但是由于手机的分辨率大小不统一，绝对布局的适应性比较差。
- ▶ 4、RelativeLayout: 相对布局，其子控件是根据所设置的参照控件来进行布局的，设置的参照控件可以是父控件，也可以使其他的子控件。
- ▶ 5、TableLayout: 表格布局，是以行列的形式来管理子控件的，在表格布局中的每一行可以是一个View控件或者是一个TableRow控件。而TableRow控件中还可以添加子控件。

LinearLayout

LinearLayout

TextView

TextView

TextView

LinearLayout

TextView

TextView

TextView

TableLayout

TableRow

TextView

TextView

TableRow

TextView

TextView

TableRow

TextView

TextView

LayoutDemo



4.8 布局实例

- ▶ app中用到了4种布局：LinearLayout、AbsoluteLayout、RelativeLayout、TableLayout
- ▶ FrameLayout布局很少用到
- ▶ app中消息列表listview里面的具体布局文件message_item.xml，用到了：LinearLayout、AbsoluteLayout、RelativeLayout
- ▶ RelativeLayout布局属性值

```

message_item.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="horizontal">
6
7     <AbsoluteLayout
8         android:layout_width="80dp"
9         android:layout_height="wrap_content">
10
11         <ImageView
12             android:id="@+id/header"
13             android:layout_width="wrap_content"
14             android:layout_height="wrap_content"
15             android:paddingBottom="15dp"
16             android:paddingLeft="15dp"
17             android:paddingTop="15dp" />
18
19         <TextView
20             android:id="@+id/msgcount"
21             style="@style/tab_rb_text"
22             android:layout_width="18dp"
23             android:layout_height="18dp"
24             android:layout_x="60dp"
25             android:layout_y="10dp"
26             android:textSize="12sp" />
27     </AbsoluteLayout>
28
29     <LinearLayout
30         android:layout_width="match_parent"
31         android:layout_height="wrap_content"
32         android:orientation="vertical">
33
34         <RelativeLayout
35             android:layout_width="match_parent"
36             android:layout_height="wrap_content">
37
38             <TextView
39                 android:id="@+id/title"
40                 android:layout_width="wrap_content"
41                 android:layout_height="match_parent"
42                 android:layout_marginBottom="8dp"
43                 android:layout_marginRight="70dp"
44                 android:layout_marginTop="18dp"
45                 android:ellipsize="end"
46                 android:singleLine="true"
47                 android:textColor="@color/black"
48                 android:textSize="16sp" />
49
50             <TextView
51                 android:id="@+id/time"
52                 android:layout_width="match_parent"
53                 android:layout_height="match_parent"
54                 android:layout_alignParentRight="true"
55                 android:layout_alignParentTop="true"
56                 android:gravity="right"

```



▶ RelativeLayout相对布局的属性值

`android:layout_above="@id/xxx"` --将控件置于给定ID控件之上

`android:layout_below="@id/xxx"` --将控件置于给定ID控件之下

`android:layout_toLeftOf="@id/xxx"` --将控件的右边缘和给定ID控件的左边缘对齐

`android:layout_toRightOf="@id/xxx"` --将控件的左边缘和给定ID控件的右边缘对齐

`android:layout_alignLeft="@id/xxx"` --将控件的左边缘和给定ID控件的左边缘对齐

`android:layout_alignTop="@id/xxx"` --将控件的上边缘和给定ID控件的上边缘对齐

`android:layout_alignRight="@id/xxx"` --将控件的右边缘和给定ID控件的右边缘对齐

`android:layout_alignBottom="@id/xxx"` --将控件的底边缘和给定ID控件的底边缘对齐

`android:layout_alignParentLeft="true"` --将控件的左边缘和父控件的左边缘对齐

`android:layout_alignParentTop="true"` --将控件的上边缘和父控件的上边缘对齐

`android:layout_alignParentRight="true"` --将控件的右边缘和父控件的右边缘对齐

`android:layout_alignParentBottom="true"` --将控件的底边缘和父控件的底边缘对齐

`android:layout_centerInParent="true"` --将控件置于父控件的中心位置

`android:layout_centerHorizontal="true"` --将控件置于水平方向的中心位置

`android:layout_centerVertical="true"` --将控件置于垂直方向的中心位置

▶ 相对布局示例，app中新闻列表页面的xml文件 listview_fragment_news.xml

```
listview_fragment_news.xml x
1  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     xmlns:tools="http://schemas.android.com/tools"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent">
5     <RelativeLayout
6         android:layout_width="match_parent"
7         android:layout_height="match_parent"
8         android:minHeight="70dp"
9         android:paddingBottom="10dp"
10        android:paddingLeft="15dp"
11        android:paddingRight="15dp"
12        android:paddingTop="10dp">
13        <!-- 标题 -->
14        <TextView
15            android:id="@+id/fragment_news_listview_title"
16            android:layout_width="wrap_content"
17            android:layout_height="match_parent"
18            android:layout_alignParentLeft="true"
19            android:layout_toLeftOf="@+id/fragment_news_listview_photo"
20            android:ellipsize="end"
21            android:maxLines="1"
22            android:singleLine="true"
23            android:textColor="#333333"
24            android:textSize="16sp" />
25        <ImageView
26            android:id="@+id/fragment_news_listview_photo"
27            android:layout_width="60dp"
28            android:layout_height="40dp"
29            android:layout_alignParentRight="true"
30            android:layout_alignParentTop="true"
31            android:layout_marginLeft="10dp"
32            android:contentDescription="@string/action_settings" />
33        <!-- 时间 -->
34        <TextView
35            android:id="@+id/fragment_news_listview_time"
36            android:layout_width="wrap_content"
37            android:layout_height="wrap_content"
38            android:layout_alignParentBottom="true"
39            android:layout_marginTop="10dp"
40            android:textColor="#666666"
41            android:textSize="12sp" />
42        <!-- bottom边线 -->
43    </RelativeLayout>
44 </RelativeLayout>
```



▶ 表格布局示例，app中的首页fragment_fragment_page_home.xml

```
fragment_fragment_page_home.xml • listview_fragment_news.xml × activity_main_activ
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical">
6
7     <ImageView
8         android:id="@+id/btn_scan"
9         style="@style/index_image"
10        android:src="@drawable/index_scan"/>
11
12    <ScrollView
13        android:layout_width="match_parent"
14        android:layout_height="match_parent"
15        android:overScrollMode="never"
16        android:scrollbarStyle="insideOverlay">
17
18        <TableLayout
19            style="@style/index_table"
20            android:shrinkColumns="1">
21
22            <TableRow
23                style="@style/index_table_row"
24                android:paddingTop="8dp">
25
26                <TextView
27                    android:id="@+id/xlcx_title"
28                    style="@style/index_table_title"
29                    android:text="@string/xlcx_title" />
30
31                <TextView
32                    android:id="@+id/zxyz_title"
33                    style="@style/index_table_title"
34                    android:text="@string/zxyz_title" />
35            </TableRow>
36            <!-- bottom边线 -->
37            <View style="@style/set_list_bottom" />
38
39            <TableRow style="@style/index_table_row">
40
41                <TextView
42                    android:id="@+id/content_xlcx"
43                    style="@style/index_table_text"
44                    android:drawableLeft="@drawable/index_xlcx"
45                    android:text="@string/xlcx" />
46                <TextView
47                    android:id="@+id/content_xjxl"
48                    style="@style/index_table_text"
49                    android:drawableLeft="@drawable/index_yzbg"
50                    android:text="@string/xjxl" />
51            </TableRow>
52            <!-- bottom边线 -->
53            <View style="@style/set_list_bottom" />
```



▶ 总结:

- ▶ 4、1 项目文档结构及作用
 - ▶ 4、2 什么是.9图片
 - ▶ 4、3 如何写xml布局文件
 - ▶ 4、4 常用控件
 - ▶ 4、5 控件必填项
 - ▶ 4、6 常用属性值
 - ▶ 4、7 常用的5大布局
 - ▶ 4、8 布局示例
- ▶ 新技术学习: Android Material Design, 新控件的引入和学习

总结

▶ 讲解内容总结:

- ▶ 1、Web App、Native App和Hybrid App
- ▶ 2、Web App中的rem单位
- ▶ 3、Android开发中的dp和sp单位
- ▶ 4、Android前端开发注意事项

▶ ppt制作总结:

- ▶ 平时接触新技术的时候，进行及时整理和记录
- ▶ 看不同的网站上的讲解，结合实际开发进行合理判断，不可盲目。
- ▶ 不懂得地方需要反复查看和分析。
- ▶ 资源共享，共同学习。

推荐的学习网站

- ▶ W3C: <http://www.w3school.com.cn>
- ▶ 菜鸟教程: <http://www.runoob.com>
- ▶ 极客学院wiki: <http://www.jikexueyuan.com>
- ▶ 慕课网: <http://www.imooc.com>
- ▶ 51CTO学院: <http://edu.51cto.com>

The end, Thanks !