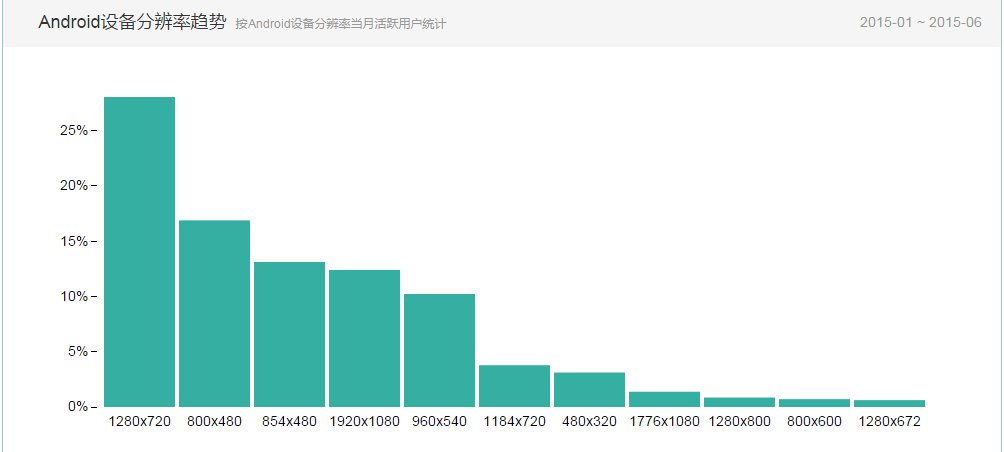
**主流厂商屏幕参数及适配背景：**

2015.01 - 2015.06



2014.01 - 2014.06



从2014到2015年，市场占有率前六位，基本没有变化（480\*320被淘汰出前六，1280 \* 720跻身）。1080 \* 1920逐渐成为旗舰机的主流配置。

**适配的工作没有必要也并不可能将所有的屏幕都完美适配，主流的屏幕可以适配完美，能够兼容其他屏幕就可以了。**

pad和tv的适配比较特殊，可以单独考虑（large tv限定符）。

（注：华为的手机由于有虚拟实体键，所有height比标准小，1776\*1080 和 1184\* 720都是， 都按照标准的来适配就可以了）

**几个重要概念：**

1、 dp、dip、dpi、sp、px

1）px：pixel UI出图单位，屏幕硬件显示的基本尺寸单位像素

2）dp = dip values/dimens里变量的单位：Density independent pixel，会随着手机屏幕的density系数自动计算出pixel，开发者不需要考虑density的问题，所以叫density independent。手机framework系统会根据DisplayMetrics.density作为系数乘以dimens下的dp值，得到实际要显示的px值

3) dpi ： dot per inch 像素密度，是手机的属性值，在values下限定不同手机取值范围的文件夹名称后缀；（标准的dpi是160dpi，此时1dp=1px，不只有160 240等几个值，不同手机有不同的dpi值， 与手机density的关系是dpi = 160 \* density）

4) sp： scale-independent pixel， 手机的系统字体大小会改变scale值，（为适配方便有时候会使用dp,不受系统字体大小影响，否则要显示完全需要动态布局）

2、 mdpi、hdpi、xdpi、xxdpi

用于res文件夹下， drawable和dimens的区分，根据density的范围进行划分，ldpi以及淘汰；这只决定了手机将从哪个文件夹中取值，取值之后的仍然乘以系数density

| **名称** | **像素密度范围** |
| --- | --- |
| mdpi | 120dpi~160dpi |
| hdpi | 160dpi~240dpi |
| xhdpi | 240dpi~320dpi |
| xxhdpi | 320dpi~480dpi |
| xxxhdpi | 480dpi~640dpi |

范围大概上上一个dpi到本层dpi倍率的范围。

**drawable的适配**

1）切图：对于drawable来讲，要根据不同的drawable来切图，切图的大小按照 2:3:4:6:8逐一放大；（mdpi市场占有率也基本很低，xxxhdpi还没有普及，所以一般中间三种dpi来切图）

2）drawable的读取：如果在手机对应的drawable没有取到，会自动从高一等的分辨率中取，而且系统会进行裁剪（从高分辨率中取到会压缩，从低分辨率会放大）。系统会自动压缩放大，之所以还需要每个drawable还放入资源是因为放大和压缩是需要性能成本。

如果不考虑性能成本，那么可以通过以下方式实现一套drawable适配：

例如hdpi 720\*1080 的手机通过drawable-hdpi-sw600dp的限定无法再hdpi中取到drawable，从mdpi中取drawable，系统自动将drawable的大小放大1.5倍，同理xhdpi xxhdpi也从mdpi中取值。这样只需要准备一套mdpi的drawable即可

3）关于drawable图形的适配，网上说，要尽量使用wrap\_content、match\_parent、weight来适配（动态适配所有屏幕），一部分正确吧。match\_parent确实可以做到适配所有屏幕（但是应用有限，一般是背景图等）；weight应用于linearlayout中，而且measure layout的效率不高，不建议在高度重用的界面使用，而且界面层级复杂后有想不到的问题； wrap\_content存在一个问题，对于相同dpi（如hdpi）但是屏幕分辨率不同（一个720\*1280 ，一个640\*960）的手机，wrap\_content的drawable占手机屏幕的比例是不同的，造成无法适配。所以个人建议利用dimens限制drawable的大小，重点适配dimens的大小。

*关于weight在wrap\_content和match\_parent下结果不同的问题。*

*wrap\_content（或android：layout\_width=0dp）下，view显示所占比例和weight成正比，match\_parent正好成反比。*

*这是因为weight的机制是: View的原有宽度+ 剩余空间\*weight*

*如果两个view都是match\_parent，那剩余空间就是负的屏幕宽度，这样结果正好反比。*

**dimens的适配**

市面上的手机分辨率和像素密度之间大部分有一个对应关系（例如 xhdpi 720 \* 1280 density= 2，xxhdpi 1080 \*1920 density = 3，也就是分辨率提升了1.5倍，density也提升了1.5倍，那一套dimens就满足了）这样，只需要一套dimens系统自动就根据density计算好pixel。这样是不是万事大吉了呢？too young，too simple, hdpi（density = 1.5）的设备一般是480\*800左右，不符合这套比例。解决有两种方法：

1 设计xhdpi hdpi两个values，xxhdpi取值xhdpi，

2 为xhdpi增加sw360dp类似的限制符，将满足sw360dp以上的所有屏幕都包含进来，（一般是xh xxh），hdpi采用sw-300dp限制符。

使用sw-dp的方式分类还有一个好处：dimens的取值是根据像素密度来values文件夹里找值，找到之后就乘以手机自己的density。所以如果分辨率和density成正比，那么提供一套dimens值就可以了，系统会自动计算。（xh density = 2对应720 xxh density = 3对应1080）最大的问题就是相同的values里取到的值，乘以相同的density，结果手机的分辨率差别大（比如xh的 width 可能是 640 720 1080）这样就没法适配了。所以利用sw-dp将分辨率进行分类，在分类的基础上，基本能保证这一类内的屏幕分辨率和像素密度是正比的。这样这几个dimens文件中的数值不需要重新计算，都是一样的值。再对一些特殊的机型做特殊处理，特别计算就可以基本适配所有的手机了。

使用限定符：

1 尺寸限定符：

一般在pad或tv，使用不同的layout时使用，layout-large.xml适用于大于7寸的设备（large指设备尺寸）

2 sw限定符 最小宽度限定符，

关于限定符的优先级，以及读取顺序，参考文章二

**layout适配：**

基本是大屏幕（pad等large屏幕）横屏（horizontal）各自有不同的layout布局

官方文档：

<http://developer.android.com/training/basics/supporting-devices/index.html>

参考资料：

一、<http://blog.csdn.net/jdsjlzx/article/details/45891551>

二、<http://blog.csdn.net/zhaokaiqiang1992/article/details/45843661>