**Mipmap纹理技术简介**

在三维世界中,显示一张图的大小与摄象机的位置有关,近的地方,图片实际象素就大一些,远的地方图片实际象素就会小一些,就要进行一些压缩,例如一张64\*64的图,在近处,显示出来可能是50\*50,在远处可能显示出来是20\*20.

       如果只限于简单的支掉某些像素,将会使缩小后的图片损失很多细节,图片变得很粗糙,因此,图形学有很多复杂的方法来处理缩小图片的问题,使得缩小后的图片依然清晰,然而,这些计算都会耗费一定的时间.

       Mipmap纹理技术是目前解决纹理分辨率与视点距离关系的最有效途径,它会先将图片压缩成很多逐渐缩小的图片,例如一张64\*64的图片,会产生64\*64,32\*32,16\*16,8\*8,4\*4,2\*2,1\*1的7张图片,当屏幕上需要绘制像素点为20\*20 时，程序只是利用 32\*32 和 16\*16 这两张图片来计算出即将显示为 20\*20 大小的一个图片，这比单独利用 32\*32 的那张原始片计算出来的图片效果要好得多，速度也更快.

[**Android**](http://lib.csdn.net/base/android) 在 API level 17 加入了 mipmap 技术，对 bitmap 图片的渲染支持 mipmap 技术，来提高渲染的速度和质量。

mipmap 是一种很早就有的技术了，翻译过来就是**纹理映射技术**。android 中的 mipmap 技术主要为了应对图片大小缩放的处理，在android 中我们提供一个 bitmap 图片，由于应用的需要（比如缩放动画），可能对这个 bitmap 进行各种比例的缩小，为了提高缩小的速度和图片的质量，android 通过 mipmap 技术提前对按缩小层级生成图片预先存储在内存中，这样就提高了图片渲染的速度和质量。

api 中通过 Bitmap 的 public final void setHasMipMap (boolean hasMipMap) 方法可以让系统渲染器尝试开启 Bitmap 的 mipmap 技术。但是这个方法只能建议系统开启这个功能，至于是否正真开启，还是由系统决定。

res 目录下面 mipmap 和 drawable 的区别也就是上面这个设置是否开启的区别。mipmap 目录下的图片默认 setHasMipMap 为 true，drawable 默认 setHasMipMap 为 false。

google 建议大家只把 app 的启动图标放在 mipmap 目录中，其他图片资源仍然放在 drawable 下面。

下面是引用自 google 官方的描述

Mipmapping for drawables   
Using a mipmap as the source for your bitmap or drawable is a simple way to provide a quality image and various image scales, which can be particularly useful if you expect your image to be scaled during an animation.   
Android 4.2 (API level 17) added support for mipmaps in the Bitmap class—Android swaps the mip images in your Bitmap when you’ve supplied a mipmap source and have enabled setHasMipMap(). Now in Android 4.3, you can enable mipmaps for a BitmapDrawable object as well, by providing a mipmap asset and setting the android:mipMap attribute in a bitmap resource file or by calling hasMipMap().

参考：   
[android reference bitmap](http://developer.android.com/reference/android/graphics/Bitmap.html#setHasMipMap%28boolean%29) ：<http://developer.android.com/reference/android/graphics/Bitmap.html#setHasMipMap%28boolean%29>

mipmap主要用于纹理贴图的，对缩放有更好的效果。

mipmap相比drawable，主要是会对sclae有一个优化，性能好一些，占用内存少一些。官方是这样说的：  
Using a mipmap as the source for your bitmap or drawable is a simple way to

provide a quality image and various image scales, which can be particularly useful

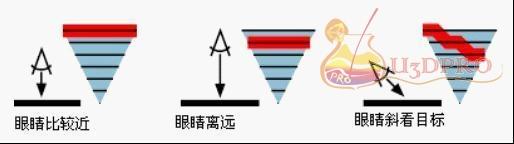
if you expect your image to be scaled during an animation.

最好就是放启动的应用图标。官方给的说明：  
It’s best practice to place your app icons in mipmap- folders (not the drawable- folders) because

# they are used at resolutions different from the device’s current density. 另一遍产得更专业的博文：

原地址：

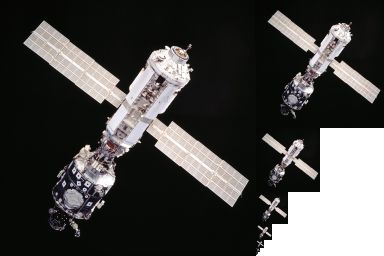
[**MipMap**](http://www.cppblog.com/wc250en007/archive/2011/08/06/152653.html)

首先从MIPMAP的原理说起，它是把一张贴图按照2的倍数进行缩小。直到1X1。把缩小的图都存储起来。在渲染时，根据一个像素离眼睛为之的距离，来判断从一个合适的图层中取出texel颜色赋值给像素。在D3D和OGL都有相对应的API控制接  
口  
  
透过它的工作原理我们可以发现，硬件总是根据眼睛到目标的距离，来玄奇最适合当前屏幕像素分辨率的图层。假设一张32768x32768的mipmap贴图，当前屏幕分辨率为1024\*1024。眼睛距离物体比较近时，mipmap最大也只可能从1024\*1024的Mipmap图层选取texel。再次，当使用三线性过滤（trilinear）时，最大也只能访问2048\*2048的图层选取texel，来和1024\*1024图层中的像素进行线性插值。  
  
为了加快渲染速度和减少图像锯齿，贴图被处理成由一系列被预先计算和优化过的图片组成的文件,这样的贴图被称为 MIP map 或者 mipmap

多级渐进纹理由一组分辨率逐渐降低的纹理序列组成，每一级纹理宽度和高度都是上一级纹理宽度和高度的一半。宽和高不一定相等，也就是说，这些纹理不一定都是正方形。

Direct3D在纹理映射时，自动选择一幅与物体大小最接近的纹理进行渲染。当物体离投影平面较远时，Direct3D会选择一张尺寸较小、分辨率较低的纹理进行渲染；当物体离投影平面较近时，Direct3D会选择一张尺寸较大、分辨率较高的纹理进行渲染。Direct3D将纹理序列看成一条多级渐进纹理链。链头处纹理的分辨率最高，下一级往后依次递减，链尾处纹理的分辨率最低。

Direct3D能估计出多级渐进纹理链中哪幅纹理的分辨率最接近想要的输出结果，然后它将像素映射到纹理空间。当最终显示的图形大小介于任意两级纹理图形之间时，Direct3D将两级纹理的相应元素进行混合后显示。

多级渐进纹理过滤能够有效地提高图形渲染速度，当物体离投影平面较远时，Direct3D会选择一张尺寸较小的纹理进行渲染，而无需经过复杂的诸如各项异性纹理过滤，并且由于这时纹理需要的显存比不使用多级渐进纹理时小，因此能有效地减少纹理载入显存的时间。缺点是对内存的要求比较高  


**设置多级渐进纹理过滤方式**

当最终显示的纹理贴图大小介于任意两级纹理之间时，Direct3D能够取得两级纹理元素进行混合后显示，具体的混合方式由指定的多级渐进纹理过滤方式决定。可以调用函数IDirect3DDevice9::SetSamplerState()设置多级渐进纹理过滤方式，将第一个参数设为纹理层序号，第二个参数设为D3DSAMP\_MIPFILTER表示多级渐进纹理过滤，第三个参数设为在相邻纹理级之间的过滤方式，可取枚举类型D3DTEXTUREFILTERTYPE的任意值。下面的示例代码设置相邻纹理级之间的过滤方式为线性过滤。

g\_device->SetSamplerState(0, D3DSAMP\_MIPFILTER, D3DTEXF\_LINEAR);

如果将第三个参数设为D3DTEXF\_NONE，那么就会一直使用最高一级的纹理，即禁用多级渐进纹理过滤。如果将其设为D3DTEXF\_POINT，就会只使用与图元大小最匹配的一级纹理。如果将其设为D3DTEXF\_LINEAR，Direct3D就将与图元大小最匹配的两级纹理以线性方式混合。

需要注意的是，多级纹理过滤是缩小和放大过滤器的结合。例如，如果将缩小和方法过滤器设为线性过滤，但是多级纹理过滤方式设为最近点采样，Direct3D就会选择与要显示的纹理贴图大小最接近的纹理级别，在该级纹理上完成双线性纹理过滤，并将结果作为像素的值。如果将缩小、放大过滤器和多级渐进纹理都设置为线性过滤，则Direct3D就会在两个最接近的纹理级别上都进行双线性纹理过滤，然后再对相邻两级纹理图形上对应的两个纹理颜色进行加权平均，最后的结果作为单个像素值。这种为了图元中的一个像素，而结合了两幅纹理，共8个像素的技术，称为“三线性过滤”，因为它在纹理的三个方向----u、 v和纹理级别上都进行了线性过滤。

可以通过IDirect3DDevice9::SetSamplerState()函数设置实际渲染时纹理过滤的最大级数，其中需要将第二个参数设为D3DSAMP\_MAXMIPLEVEL，第三个参数设为实际渲染时纹理过滤的最大级数。下面的示例代码设置纹理层0的最大多级纹理过滤级数为16。

g\_device->SetSamplerState(0, D3DSAMP\_MAXMIPLEVEL, 16);

还可以通过将IDirect3DDevice9::SetSamplerState()的第二个参数设为D3DSAMP\_MIPMAPLODBIAS，设置多级纹理映射级数偏移值。如果对某个纹理映射设置正偏移值，得到的图形结果就会比原来的更清晰，但锯齿更多；反之设为负偏移值，得到的图形结果就会更模糊。