# Android UI 设计指南（自译）

# 一、图标设计指南

创建一个统一的外观和整体的用户界面效果以增加产品的价值，精简的图形样式还能让用户觉得 UI 更专业。

本文提供的信息能帮助你为应用的用户界面的各个部分创建的图标与 Android2.X 框架的一般样式相匹配。以下的指南将帮助你创建一个完美而且统一的用户体验。

下面文档讨论关于 Android 应用程序常见类型图标的使用详细指南：

**启动图标**

启动图标是您的应用程序在设备的主界面和启动窗口的图形表现。

**菜单图标**

菜单图标是当用户按菜单按钮时放置于选项菜单中展示给用户的图形元素。

**状态栏图标**

状态栏图标用于应用程序在状态栏中的通知。

**Tab 图标**

Tab 图标用来表示在一个多选项卡界面的各个选项的图形元素。

**对话框图标**

对话框图标是在弹出框中显示，增加互动性。

**列表视图图标**

使用列表视图图标是用图形表示列表项，比如说设置这个程序。

想更快的创建你的图标，可以导向 Android 图标模板包。

## ● 使用 Android 图标模板包

Android 的图标模板包是模板设计、纹理和图层样式的集合，使您更容易的创建在这份文档中符合指南指引的图标。我们建议您下载模板包文档再设计图标。

图标模板提供了 Adobe Photoshop 格式的文档（PSD），以确保在为 Android 平台创建标准的图标提供层和设计处理。你可以加载模板文件到任何兼容的图片编辑程序，但是你所使用的基于该应用程序的层和处理方式可能会有所不同。

## ● 提供密度特定的图标集

Android 是为多种设备设计的，因此必须提供一系列的图片尺寸和解决方案。当你为应用设计图标时，最重要的是要让你的应用安装在任何设备上都是保持一致的。就你所知道的支持多屏幕文档中，Android 平台让你提供的图标能更直接的在任何设备上不管这个设备的尺寸和精度都以同一种方式准确的实现。

在一般情况下，推荐的方法是为下表中所列出的广义上的三种屏幕密度各创建一个单独的图标。当应用运行时，Android 平台将检查屏幕的规格然后从特定文件中加载适合的图标

资源。欲了解关于如何存储特定密度资源信息，请参阅屏幕大小和密度的资源目录。

有关如何创建和管理多种密度的图标集的提示，参阅设计师的提示。

表 1：三个屏幕密度所需要的图标尺寸：

| **图标类型** | **标准尺寸（以像素为单位），广义的屏幕精度** | | |
|---|---|---|---|
| | 低精度 *(ldpi)* | 中精度*(mdpi)* | 高精度 *(hdpi)* |
| 启动图标 | 36 x 36 px | 48 x 48 px | 72 x 72 px |
| 菜单图标 | 36 x 36 px | 48 x 48 px | 72 x 72 px |
| 状态栏图标 (Android 2.3 以后版本) | 12w x 19h px (preferred, width may vary) | 16w x 25h px (preferred, width may vary) | 24w x 38h px (preferred, width may vary) |
| 状态栏图标(Android 2.2 及之前版本) | 19 x 19 px | 25 x 25 px | 38 x 38 px |
| Tab 图标 | 24 x 24 px | 32 x 32 px | 48 x 48 px |
| 对话图标 | 24 x 24 px | 32 x 32 px | 48 x 48 px |
| 列表图标 | 24 x 24 px | 32 x 32 px | 48 x 48 px |

- **设计师建议**

这里有几点建议对你为应用程序设计图标或者绘制设备也许有用。这个建议是以 Adobe Photoshop 或者一种位图和矢量编辑软件为基础的。

- **图标资源命名规则**

按字母排列顺序命名文件以便将相关资源规整到一个目录里面，特别是，它有助于使用一个共同的前缀为每个图标类型。例如：

| **资源类型** | **前缀** | **举例** |
|---|---|---|
| 图标 | `ic_` | `ic_star.png` |

| | | |
|---|---|---|
| 启动图标 | ic_launcher | ic_launcher_calendar.png |
| 菜单图标 | ic_menu | ic_menu_archive.png |
| 状态栏图标 | ic_stat_notify | ic_stat_notify_msg.png |
| Tab 图标 | ic_tab | ic_tab_recent.png |
| 对话图标 | ic_dialog | ic_dialog_info.png |

值得注意的是不必使用任何类型的相同前缀-目的是方便您。

**创建一个工作文档来整理多精度文件**

支持多尺寸的屏幕精度一个图标就需要创建多个版本，为了让多个副本文件安全并且容易查找，我们建议在你的工作空间里每个资源文件创建 一个资源包。如下：

```
assets/...
    ldpi/...
        _pre_production/...
            working_file.psd
        finished_asset.png
    mdpi/...
        _pre_production/...
            working_file.psd
        finished_asset.png
    hdpi/...
        _pre_production/...
            working_file.psd
        finished_asset.png
```

此结构相似的特定精度结构，您将把您的应用资源最终存储为完整资源。因为你的工作环境和应用程序结构类似，你可以快速的决定哪些资源必须拷贝到每个应用程序目录。通过分辨率区分资源同样可以帮助你通过文件名发现多精度是否一致，这点很重要，因为不同精度的相同资源必须使用相同的文件名。

以下对比，这是一个典型的应用的资源目录结构：

```
res/...
    drawable-ldpi/...
        finished_asset.png
    drawable-mdpi/...
        finished_asset.png
    drawable-hdpi/...
        finished_asset.png
```

- **最好使用矢量图形**

诸如 Adobe Photoshop 的图像编辑软件允许使用矢量形状、栅格图层和图层效果相结合。如果可能，使用矢量图形，以便在需要时资源能随意缩放而不会都是细节和边缘清晰度。

使用矢量图形也可以很容易的在较小精度的像素边界对其边缘。

- **以最大尺寸开始**

你需要针对不同屏幕精度创建不同资源，如表 1，最好以大尺寸图标的设计或者所有图标大小的倍数尺寸开始设计。比如，启动图标根据屏幕精度需要 72px、48px、36px 三种尺寸。如果你以 864*864 的尺寸开始绘制启动图标，当你创建最终资源的图标尺寸时就能更容易和清晰的调整图标。

他同样有益于在高精度的对象中提供被推荐的安全边界的大尺寸增加指引（也称为指南）。继续以上的例子，每条指南，启动图标包含 72x72 的完整资源包括 60x60px（56x56 的矩形图标区域）。在 864x864 的大尺寸中，这相当于在水平和竖直方向上在需要 72px 的边缘。

- **缩放时，位图图片需要重新绘制**

如果缩放图形的话，要从矢量图层进行缩放，而不是位图图层，如果在高精度的环境中，这些位图图层需要重新绘制。比如说在一个中等精度下 60x60 位图图形的圆要在高分辨下以 90x90 的大小显示就需要重新绘制。

- **当保存图像资源时，删除一些没用的资源。**

为了帮助图像资源尽可能小，可以从文件中删除一些不必要的头文件，如 Adobe Fireworks 的元数据或 Adobe Photoshop 头文件。删除 Photoshop 头文件的步骤如下：

1. 选择文件>存储为 Web 和设备所用格式
2. 在“存储为 web 和设备所用格式”面板，预设选择“PNG-24”，设置底下选项仍为“PNG-24”，勾选“透明度”（如果图像含有透明度）
3. 选择“存储”

也可以使用 PNG 文件的优化工具，如 OptiPNG 或 Pngcrush。

- **保证不同精度的相应资源使用相同的文件名**

同样的图标资源文件在每个精度中必须使用同样的文件名，但是要存储在具体精度目录里。这允许系统根据设备屏幕的特点查找并加载适合的资源。出于这个原因，确保资源在每个目录是一致的文件名，而不使用特定的精度后缀名。

# 1. 启动图标

启动图标是在设备主屏幕和启动窗口代表应用程序的图形。

用户打开主屏幕下方的触摸屏的启动图标。启动界面一打开将看到所有安装应用的图标。用户选择一个应用程序，并通过点击启动图标或任何硬件导航控件，如轨迹球或者 D-pad 打开他。

正如你所看到的，在提供多个特性精度图标设备时，你必须创建低精度、中精度和高精

度的三种屏幕图标。这将确保图标正确的显示在被安装的设备的应用程序上。可以看看设计师建议这一节的建议如何使用多套图标的问题。

- **Android Market 的应用程序图标**

如果您在 **Android Market** 发布应用程序，你必须提供在开发者平台应用程序上传时的 512x512px，高精度的图标。此图标将被用于在 Android Market 的不同地点，但不会取代你的启动器图标。

如何更容易的将高精度图标缩放到 512*512 的提示和建议，在设计师建议章节中有讲。

对于高精度的应用程序图标，在 Android Market 的信息和规格，请参阅下面的文章：

应用程序图形资源（Android Marker 帮助说明）

- **Android2.0 以后版本**

启动 Android2.0，启动图标就展示在面前，而不是四分之三透视。下面的指南描述了如何设计 Android2.0 之后版本的启动图标的设计。

**样式**

创建启动图标，应遵循以下总体风格的原则。这个准则并不意味着限制你设计图标，而是强调用相同的方法在设备上分享您的图标，如图 1


**Figure 1.** Example launcher icons for Android 2.0 and greater.

**简洁时尚**

符合当下的流行趋势，避免过度使用隐喻。

**高度简化图标**

• 小尺寸也易于识别，不宜太复杂。

• 尝试抓住程序的主要特征（例如，用音像表示音乐 icon）

• 使用自然的 轮廓和形状，看起来几何化和有机化，不失真实感。

• 图标采用前视角，几乎没有透视，光源在顶部

**不光滑但有质感**

图标应该采用非镜面，有质感的材料。见“材料和颜色”章节，获取更多信息。

**前视角和顶部光源**

Android2.0 之后版本，Android 启动图标采用前视角，几乎没有透视，光源在顶部。

注：Android 适用于单独的文字标签，当现实启动图标时使用应用程序名称。所以应该避免图标里嵌入文本而不是专注于设计一个独特难忘的图标。

**尺寸和定位**

启动图标要有多样化的形状和样式，但又要形成统一的视觉风格，其尺寸和定位也要统一。

图 2 说明了定位图标内资源的各种方式。图标的实际大小应该小于资源的实际边界，以创建一个一致的视觉体验和阴影效果。如果图标是矩形或者接近矩形，图标尺寸应该更小。

为了表示推荐大小的图标，图 2 中的例子每个图片包括了三个不同的指导框。

• 红色框代表图标的完整尺寸

• 蓝色框式代表图形尺寸，比图标尺寸小，图形之外的空间用于显示阴影和特殊效果；

• 橙色框代表当图标为方形时的边框范围。正方形的外框小于其他形状的图标，原因是两种类型图标的达到统一的视觉权重。

**Figure 2.** Launcher icon sizing and positioning inside the bounds of the icon asset.

高精度屏幕的启动图标尺寸：
外框：72x72px
图标：60x60px
方形图标：56x56px
中精度屏幕的启动图标尺寸：
外框：48x48px
图标：40x40px
方形图标：38x38px
低精度屏幕启动图标尺寸：
外框：36x36px
图标：30x30px
方形图标：28x28px

**材质和颜色**

启动图标要有触感、明亮和有质感的材质，即使图标只是简单的形状，但也要尝试一些取之于现实世界的材质来表现。

Android 启动图标通常由一个大的基本形状，一个中立和主色调组成的较小的形状。图标可能使用一个保持有相当对比度的中性色。如果可能的话，每个图标不使用一个以上的主色。

图标应该使用包括一系列暗淡的和基本的的色调。不能用太饱和的色调。

启动图标推荐使用的色调如图 3，你可以从中选择元素的基础色和高亮色。也可以使用

调色板中的颜色和白色到黑色的渐变的叠加。这需要创建该图标从上到下的透视，并保持色彩的不饱和度。

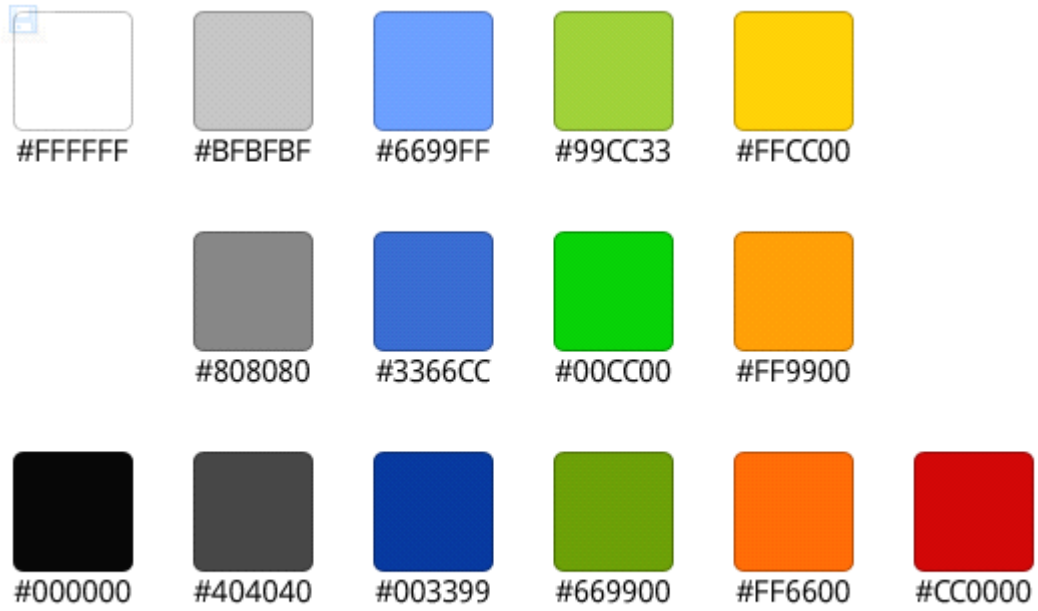



**Figure 3.** Recommended color palette for icons.

可以通过如图 4 所示的调色板颜色的几种突出材质，组合成如图 5 所示的组合。为了使您开始设计图标，图标模板包括 Photoshop 里的材质、颜色、渐变方式的文件（ic_launcher_template/example_materials.psd）

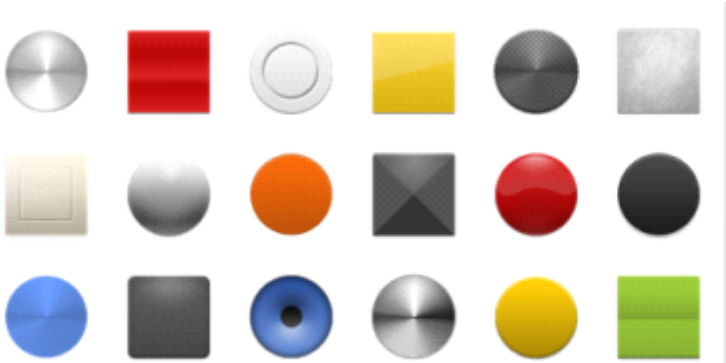


**Figure 4.** Example materials that you can use to create your icons.

**Figure 5.** Examples of materials combined with base and highlight colors from the recommended palette.

**特效**

启动图标是采用从上往下的透视角度的，通过阴影创建层次感。图标可以使用不同的纹理和灯光效果，但是必须是从上往下的透视角度。

为了保持统一性，所以的图标都需要使用相同的阴影效果，如图 6 所示。


**Figure 6.** Style, light and effects for launcher icons.

注：所有的像素尺寸为中等精度，并应适当的缩放分辨率。

1. 光影：从上而下，并适应合适的高光细节
2. 投影：#000000 75%透明度；90° 投影；距离 1px；大小 3px
3. 材质：触感，出现使用现实中的材质（例如图像中的单色噪音）

**该做什么，不该做什么**

下面是一些创建应用程序图标时需要考虑的那些“该做和不该做”的例子。

Android 启动图标该做…

- 现代、简约、磨砂、触觉和纹理
- 正面视图，且从上而下的光影，整体性，局限于调色板颜色。

Android 启动图标不该做…

- 古董、过于复杂、光泽、平面向量
- 旋转、裁切、过饱和

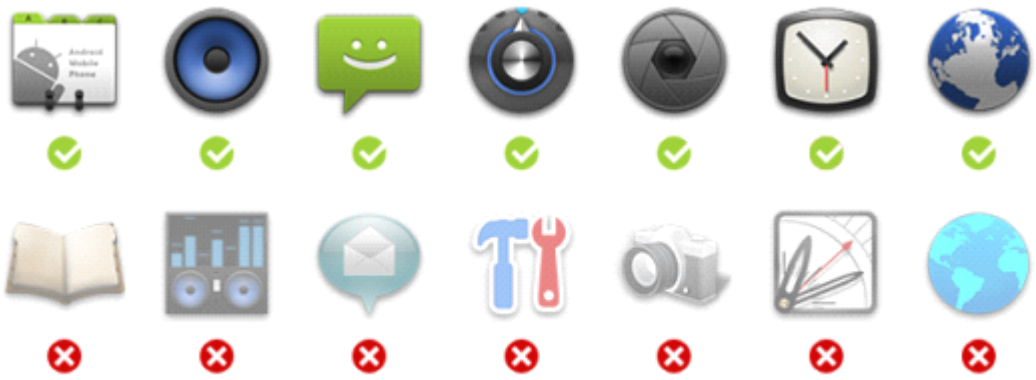


Figure 7. Side-by-side examples of "do's and don'ts" for Android launcher icons.

**图标示例**

以下例子展示 Android 应用的高精度启动图标。这些图标仅供参考-请不要再使用这些图标


- **Android1.6 及更早版本**

如下指南描述了如何设计 Android1.6 及更早版本的启动图标。Android1.6 一些版本的图标是采用一个固定角度的简化 3D 图标。如图 8 所示的角度：

**结构**

• 启动图标的底部可以朝上或者朝前

• 启动图标的表面应大部分使用启动图标调色板。使用一个或多个高亮颜色重点强调凸显的特性

• 所有的启动图标必须有圆角，使他们看起来友好且简介，如图 8 所示。

• 所有指定的尺寸都基于一个 250x250px 的如 Adobe Illustrator 矢量图形编辑器里，图标的画板边界内适合画板的大小。

• 最后使用 Adobe Photoshop 图像编辑器缩小输出为一个透明的 PNG 文件，不包括背景颜色。

• 在 Adobe Photoshop 中创建的图标模板都在图标模板包里。

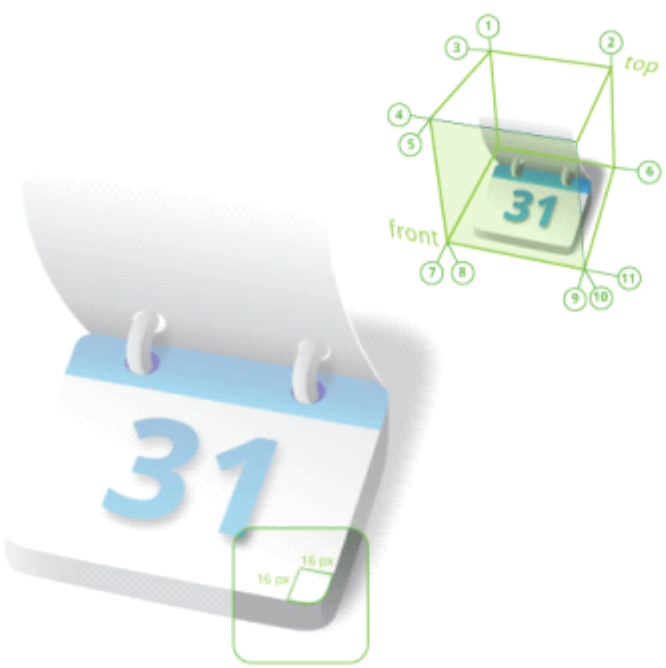



**Figure 8.** Rounded corners and perspective angles for launcher icons (90° is vertical).

1. 92°
2. 92°
3. 173°
4. 171°
5. 49°
6. 171°
7. 64°
8. 97°
9. 75°
10. 93°
11. 169°

**高光、特效和阴影**

启动图标通过使用光影和阴影定义 3D 图标。光源在左前方，因此投影在右后方。

**启动图标调色板**

白色

RGB：255 255 255

用于高亮边缘

浅灰渐变

1.RGB：255 255 255

2.RGB：217 217 217

用在图标前部

中灰渐变

1. RGB：190 190 190
2. RGB：115 115 115

用于图标边缘（阴影部分）

深灰渐变

1.RGB:100 100 100

2.RGB：25 25 25

用于图标的阴影细节部分

黑色

RGB：0 0 0

作为阴影的基础色

**步骤：**

1. 用 Adobe Illustrator 创建一个基本形状，使用“启动图标：结构”这一章所描述的角度。形状和特效应该在 250x250px 的画板里
2. 添加深度、形状并创建启动图标结构的圆角
3. 添加细节和颜色。渐变为左前方的光源照射。
4. 以正确的角度和模糊的效果创建阴影
5. 使用像 Adobe Photoshop 工具导入图标，而缩放出适合图像的 48x48px 的透明背景图片。
6. 导出透明的 48x48 大小的 Png 文件

## 2. 菜单图标

菜单图标是当用户点击菜单按钮出现的属性菜单的一种图形表示。他们的绘制角度是前视图和灰色过度。菜单图标不得以 3D 和其他角度出现。

正如“提供特定精度的图标集”这一章所描述，为高精度、中精度、低精度屏幕创建特定的图标集。这将确保图标在应用程序的各种设备中正确显示。请看“设计师建议”这一节，

建议如何创建多设备的图标。

**最终导出不包括背景颜色的透明的 PNG 文件的图片。**

创建图标的 Adobe Photoshop 文件的模板在“图标模板包”里。

提醒： Android2.3 与以前的版本相比，图标的风格和大小都有所变化

1. 图标有一个更大的安全框架；图标的内容区域比实际尺寸还小；最终的资源并没有改变。
2. 调色板更淡一些
3. 无外发光效果
4. 菜单图标可在深色或浅色背景中呈现

## ● Android2.3 及更高版本

以下指南描述如何为 Android2.3 及更高版本设计菜单图标

### 尺寸和定位

菜单图标可以使用各种形状和样式，在同一个资源内部必须创建统一的大小和定位，以达到统一的视觉权重。

如图 1 所示的资源内部多种方式的位置表示。图标的实际大小比资源的 宽度还小，以便创建一致的视觉权重。如果图标是正方形或者接近正方形，那么图标将会更小。

为了表示图标的尺寸，如图 1 所示包括了三种不同的指导框

- 红色的框代表完整资源的宽度
- 蓝色的框是推荐的图标的实际外框。这个图标框小于整体资源的宽度，目的是让不同图标形状保持一致的视觉权重
- 橙色的框代表当图标为正方形时被推荐的实际宽度。正方形的图标框小于其他图标框，为了确保两种类型有统一的视觉权重。

**Figure 1.** Menu icon sizing and positioning inside the bounds of the icon asset.

高精度屏幕的菜单图标尺寸
完整尺寸：72x72px
图标尺寸：48x48px
方形图标：44x44px
中精度屏幕的菜单图标尺寸
完整尺寸：48x48px
图标尺寸：32x32px
方形图标：30x30px
低精度屏幕的菜单图标尺寸
完整尺寸：36x36px
图标尺寸：24x24px
方形图标：22x22px

**样式、颜色和特效**

菜单图标为平面的、正视图和灰度渐变，稍微一点凹陷和其他一点特效，如下，是为了创建一点深度。当逻辑恰当时，菜单图标有圆角。

为了保持一致性，所有的菜单图标应如图 2 所示，使用相同的色调和特效。

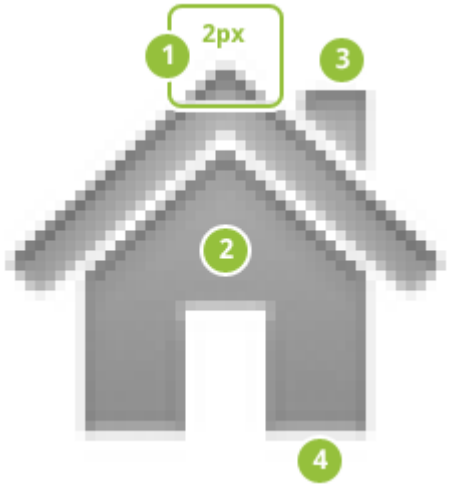



**Figure 2.** Style, light and effects for menu icons.

注：所有的尺寸为中精度，因此其他精度的需要缩放到合适尺寸。

1. 圆角宽度：恰当的 2px 圆角半径
2. 渐变填充：90°，从#8c8c8c 到#b2b2b2
3. 内 阴 影：#000000,20%不透明度
   角度 90°
   距离 2px
   大小 2px
4. 内 斜 面：深度 1%
   方向：向下
   大小：0px
   角度：90°
   高度：10°
   高光模式：#ffffff，70%不透明度
   阴影模式：#000000,25%不透明度

**该做什么，不该做什么**

以下是一些“该做和不该做”的例子，当设计应用程序菜单图标时需要考虑到的

**图标例子**

以下展示用于 Android 平台标准的高精度菜单图标。

警：因为这些资源在各个版本之间有可能不同，所以不要把这些资源与用于 Android 平

台资源的设计指导中。如果你想使用任何图标和内部绘制资源，可以存储本地一些图标副本或者在应用程序资源中绘制，然后从应用程序代码中引用本地副本。这样一来，即使系统的副本更改了，仍能保持图标外观一致。注意以下格子不是完整的。

- **Android 2.0 及更早版本**

接下来的指南描述在 Android2.2 及更早版本如何设计菜单图标。菜单图标在 Android2.2 及以下版本是以前视图形式展现。菜单图标的元素不能以 3d 视图或者其他视图展示。

**结构**

• 为了保持一致性，所有的菜单图标应使用相同的色调和特效。更多信息，请查看“色调”章节

• 菜单图标当符合逻辑时带有圆角，如图 3 所示，符合逻辑的带有圆角的区域是屋顶而不是建筑主体。

• 这个页面的所有特定尺寸为基于 48x48px 的画板带有 6px 的安全间距

• 菜单图标的图层效果（外发光）一些发光、特效和阴影当需要时可以放在 6px 的安全间距里。基础形状都应该放于安全框架里。

**• 最后必须以透明的 PNG 文件输出**

• 创建菜单图标的 Adobe Photoshop 模板在图标模板包里。

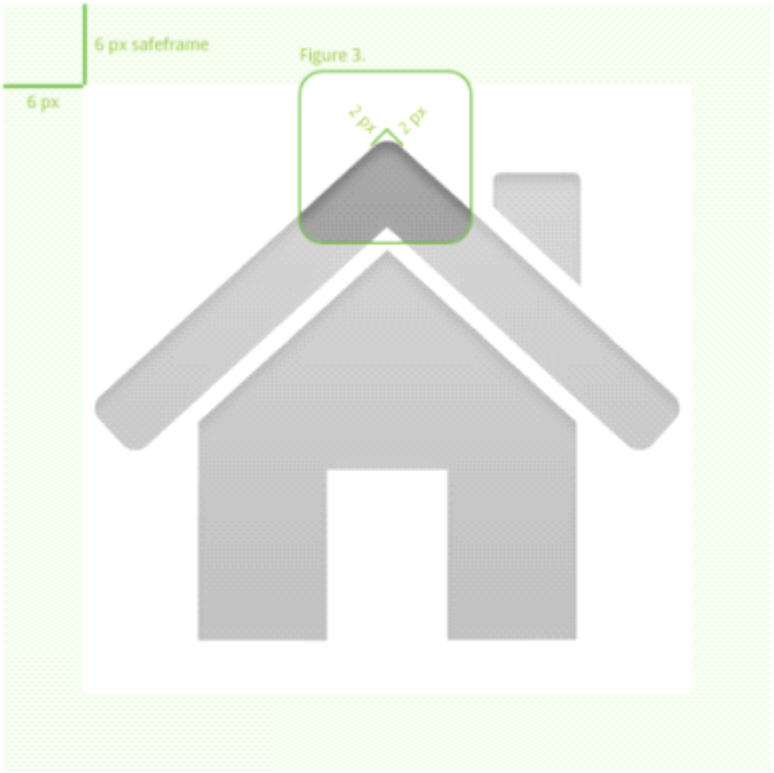



**Figure 3.** Safeframe and corner-rounding for menu icons. Icon size is 48x48.

### 发光、样式和阴影

菜单图标是正视图的，轻微的凹陷和一些其他特效，如下图所示，是为了创建深度效果。


**Figure 4.** Light, effects, and shadows for launcher icons.

1. 前　部：使用基础的色调渐变填充
2. 内阴影：黑色、20%不透明度
   角度 90°、距离：2px
   大小 2px
3. 外发光：白色 55%不透明度
   扩展 10% 大小 3px
4. 内斜面：深度 1% 、方向 向下、大小 0px
   角度 90°、高度 10°
   高光：白色 70%不透明度
   阴影：黑色 25%不透明度

**色调**

白色
RGB：255 255 255
用于外发光和内斜面高光

渐变填充
1. RGB：163 163 163
2. RGB：120 120 120

用于颜色填充

黑色
RGB：0 0 0
用于内阴影和内斜面阴影

**步骤**

1. 用 Adobe Illustrator 创建一个基本形状
2. 形状导出到 Adob Photoshop 中缩放到 48x48px 的透明背景中，留下安全边框
3. 如图 4 增加图层样式
4. 导出带有透明度的 48x48 的 PNG 图标

## 3. 状态栏图标

状态栏图标用于应用程序在状态栏的信息。

正如“提供特定精度图标集”所描述，需要创建适用于低、中和高三种精度屏幕的特定图标集。这是确保应用程序在多种设备上能正常显示和安装。参考“设计师建议”章节的建议如何创建多精度图标集。

**最后应不包括颜色背景导出透明的 PNG 图标。**

创建图标所需要的 Adobe Photoshop 文件模板在“图标模板包”里。

**警：**Android2.3 以前版本在装叹了图标的样式和尺寸上有所不同，提供所有 Android 版本支持的开发者必须注意：

1. Android2.3 及更高版本的状态栏图标放于 `drawable-hdpi-v9`, `drawable-mdpi-v9`, 和 `drawable-ldpi-v9` 三个文件中。
2. 之前版本的状态栏图标放于 `drawable-hdpi`, `drawable-mdpi`, 和 `drawable-ldpi` 三个文件中

- **Android2.3 及更高版本**

一些描述在 Android2.3 及更高版本如何设计状态栏图标

**大小和位置**

状态栏图标要使用简单的图像和框架，这些图标将缩放在最终资源所在的位置。

如图 1 所示多种图标在资源内的位置。图标的实际尺寸比资源尺寸 还小。**状态栏图标的宽度可能会有一点点的不同。**

为了指出图标的推荐尺寸，图 1 中每个例子都包括两种不同的矩形指导框：

1. 红色的框代表资源的完整尺寸
2. 蓝色的框代表图标的实际尺寸。图标框的垂直方向的尺寸比实际资源框略小，为了让多种图标形状保持视觉权重的一致性


**Figure 1.** Status bar icon sizing and positioning inside the bounds of the icon asset.

高精度屏幕（hdpi）的状态栏图标大小：
完整尺寸：24w x 38h px（首选，宽度会有所变化）
图标尺寸：24w x 24h px（首选，宽度会有所变化）
中精度屏幕（mdpi）的状态栏图标大小：
完整尺寸：16w x 25h px（首选，宽度会有所变化）
图标尺寸：16w x 16h px（首选，宽度会有所变化）
低精度屏幕（ldpi）的状态栏图标大小：
完整尺寸：12w x 19h px（首选，宽度会有所变化）
图标尺寸：12w x 12h px（首选，宽度会有所变化）

**样式、色调和特效**

状态栏图标是平的、亚光的和正视图的。


**Figure 2.** Style and effects for status icons.

注：所有的像素尺寸都是以中精度表示的，因此其他精度的需要适当缩放

1. 渐变填充：90° 从#828282 到#919191
2. 内 阴 影：#ffffff，10%不透明度
   角度 90°
   距离 1px
   大小 0px
3. 内部内容：内部内容通过减法减去外部形状，内部完全透明像素组成。

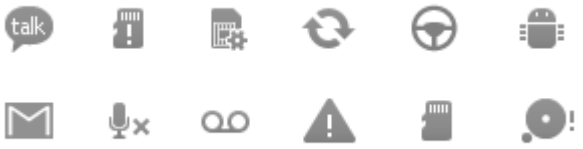
**该做什么，不该做什么**

以下是一些创建状态栏图标时的一些“该做和不该做”的例子。

**图标示例**

如下展示了在 Android 平台中标准的高精度的状态栏图标。

**警：**由于这些资源在各个版本中可能有所不同，所以不要把这些资源与用于 Android 平台资源的设计指导中。如果想用任何图标或者内置绘制资源，你可以把这些图标存储备份或者在应用程序资源中绘制，然后从应用程序代码中引用本地副本。这样一来，即使系统的副本更改了，仍能保持图标外观一致。注意以下格子不是完整的。




- **Android2.2 及更早版本**

接下来介绍如何设计 Android2.2 及更早版本的状态栏图标

**结构**

- 圆角必须始终运用于状态栏图标的基本形状，如果图 3 的细节图。
- 所有的特定尺寸基于 25x25px 的画板尺寸中，并有 2px 的安全边框
- 状态栏图标可以在安全边框范围内重叠左右两边，但是顶部和底部不能重叠。
- 最后应以带透明度的 PNG 文件输出
- 创建状态栏图标所需要的 Adobe Photoshop 模板在“图标模板包”里


**Figure 3.** Safeframe and corner-rounding for status bar icons. Icon size is 25x25.

**发光、特效和阴影**

状态栏图标有略微凸出，高对比度且图片正视图以便能在小尺寸中更清晰的表示

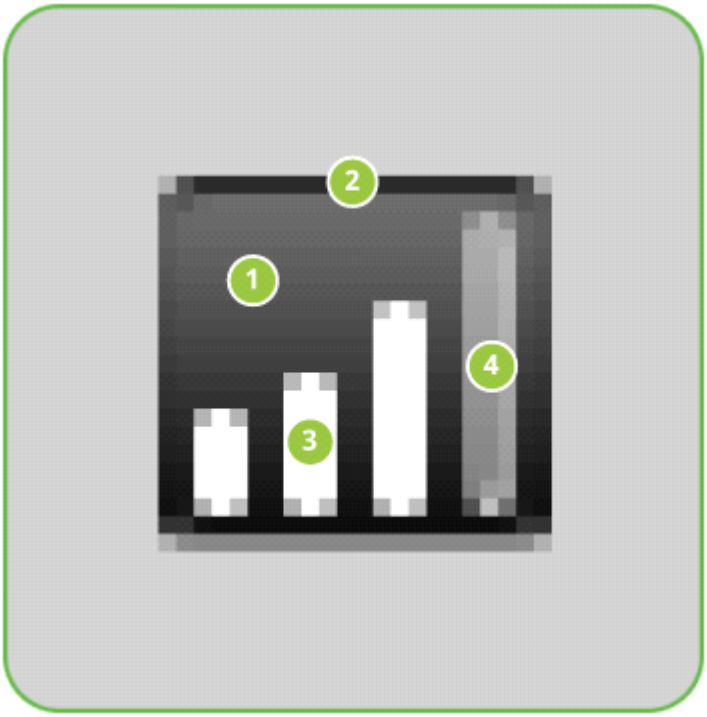



**Figure 4.** Light, effects, and shadows for status bar icons.

1. 前部：使用色调的颜色渐变填充
2. 内斜面：深度 100% 、方向 向下
   大小 0px 角度 90° 高度 30°
   高光：白色 75%不透明度
   阴影：黑色 75%不透明度
3. 细节：白色
4. 不可用细节：色调中的灰色渐变
   内斜面：平滑、 深度 1%、方向 向下、大小 0px、角度 117° 、高度 42° 、高光 白色 70%不透明度、无阴影

**色调**

只有一小部分手机的状态栏图标使用全彩，其他状态栏图标都使用纯色。

白色
RGB：255 255 255
用于图标内高光细节

灰度渐变
1. RGB：169 169 169
2. RGB：126 126 126

用于图标内不可用细节

填充渐变
1. RGB：105 105 105
2. RGB：10 10 10

用于颜色填充

黑色
RGB：0 0 0
用于内斜面阴影

**步骤**

1. 用 Adobe Photoshop 创建一个 25x25px 透明背景的基本形状，留出安全边界，保持上部和下部都留有 2px 空间
2. 增加圆角如图 3。
3. 增加高光、特效和阴影如图 4
4. 导出 25x25 的带透明度图标的 PNG 文件

## 4. Tab 图标

Tab 图标用于在多个 tab 界面表示单个 tab 的图形元素，每个 tab 图标有两种状态：未选中和选中。

就像“提供特定精度图标集”所描述的，需要创建适用于低、中和高三种精度屏幕的特定图标集。这是确保应用程序在多种设备上能正常显示和安装。参考“设计师建议”章节的建议如何创建多精度图标集。

**最后应不包括颜色背景导出透明的 PNG 图标。**

创建图标所需要的 Adobe Photoshop 文件模板在“图标模板包”里。

**警：**tab 图标的样式在 Android2.0 与更早版本有大大的不同，提供所有 Android 帮的开发者需要注意：

1. Android2.0 及更高版本的 tab 图标放于 drawable-hdpi-v5, drawable-mdpi-v5, and drawable-ldpi-v5 三个文件中
2. 之前的版本放于 drawable-hdpi, drawable-mdpi, and drawable-ldpi 三个文件中
3. 设置 Android：targetSdkVersion5 或者更高版本在“SDK 使用说明”里的“应用程序清单，”

## ● 提供两种 Tab 状态

Tab 图标有两种状态：未选中和选中。提供图标多种状态，开发者必须为每个图标创建“绘制状态列表”，图像使用不同的 UI 状态就需要不同的 XML 文件列表。

例如，一个 tab 标签分别为“朋友”和“同事”，你可以使用如下的小标签：

```
res/...
    drawable/...
        ic_tab_friends.xml
        ic_tab_coworkers.xml
    drawable-ldpi/...
        ic_tab_friends_selected.png
        ic_tab_friends_unselected.png
        ic_tab_coworkers_selected.png
        ic_tab_coworkers_unselected.png
    drawable-mdpi/...
        ic_tab_friends_selected.png
        ic_tab_friends_unselected.png
        ic_tab_coworkers_selected.png
        ic_tab_coworkers_unselected.png
    drawable-hdpi/...
        ...
    drawable-ldpi-v5/...
        ...
    drawable-mdpi-v5/...
        ...
    drawable-hdpi-v5/...
        ...
```

上面列出的 XML 文件列表需要绘制选中和未选中的两种图标。例如，

“ic_tab_friends.xml:”代码

```xml
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- selected state -->
    <item android:drawable="@drawable/ic_tab_friends_selected"
          android:state_selected="true"
          android:state_pressed="false" />
    <!-- unselected state (default) -->
    <item android:drawable="@drawable/ic_tab_friends_unselected" />
</selector>
```

- **Android2.0 及之后版本**

接下来介绍如何设计 Android2.0 及更高版本的 tab 图标

**尺寸和定位**

Tab 图标应使用简单的形状和样式，必须放置于最终资源之内。

如图 1 所示多种形状的图标在资源内部的位置，图标的实际尺寸必须比资源尺寸还小。

为了指明图标的推荐尺寸，如图 1 每个图标有三种不同的指导框：

• 红色的框代表资源的完整尺寸

• 蓝色框代表图标的实际尺寸，图标的实际尺寸比完整尺寸还小，并允许特殊图标特殊处理。

• 橙色框代表图标为方形时的推荐框，为了使两种类型保持视觉权重的一致性，因此方形图标框会比其他两个图标框还小。

**Figure 1.** Tab icon sizing and positioning inside the bounds of the icon asset.

高精度屏幕(hdpi)的 tab 图标尺寸:

完整尺寸：48x48px

图标尺寸：42x42px

中精度屏幕(mdpi)的 tab 图标尺寸:
完整尺寸：32x32px
图标尺寸：28x28px
低精度屏幕(ldpi)的 tab 图标尺寸:
完整尺寸：24x24px
图标尺寸：22x22px

**样式、色调和特效**

Tab 图标是平的、亚光的正视图。
Tab 图标有两种状态：选中和未选中。

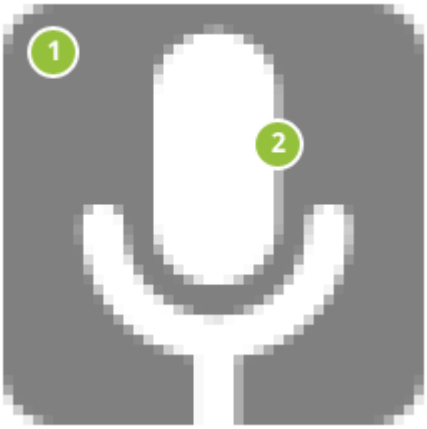



图 2：未选中 tab 图标的样式和特效

注：所有尺寸为中精度图标的尺寸，若其他精度请缩放到合适的尺寸。

1. 填充颜色：#808080
2. 内部内容：内部内容以减去外部形状，并让内部内容完全透明

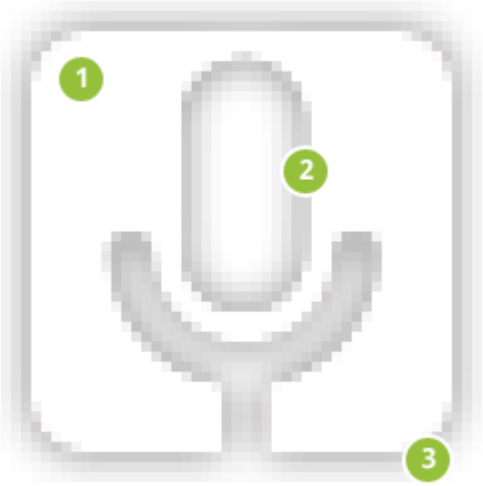



图 3 选中 tab 图标的样式和特效

注：所有尺寸为中精度图标的尺寸，若其他精度请缩放到合适的尺寸。

1. 填充颜色：#ffffff
2. 内部内容：内部内容以减去外部形状，并让内部内容完全透明

3. 外发光：#000000,25%不透明度，大小 3px

**该做什么，不该做什么**

以下是一些创建 tab 图标时的一些“该做和不该做”的例子。

**图标示例**

如下展示了在 Android 平台中标准的高精度的状态栏图标。

**警：**由于这些资源在各个版本中可能有所不同，所以不要把这些资源与用于 Android 平台资源的设计指导中。如果想用任何图标或者内置绘制资源，你可以把这些图标存储备份或者在应用程序资源中绘制，然后从应用程序代码中引用本地副本。这样一来，即使系统的副本更改了，仍能保持图标外观一致。注意以下图标不是完整的。

## ● Android1.6 及更早版本

接下来介绍如何设计 Android1.6 及更早版本的 tab 图标。

**结构**

• 未选中的 tab 图标渐变填充和特效与菜单图标一致，就是少了外发光。

• 选中图标和未选中图标很类似，就是有一个更深的内阴影，并且前景色和对话图标一致。

• tab 图标有 1px 的安全边框，这是避免圆形边缘出现重叠。

• 此页面的所有指定尺寸基于 32x32px。在 Photoshop 模板保持四周 1px 的边缘。

**Figure 3.** Safeframe and fill gradient for unselected tab icons. Icon size is 32x32.

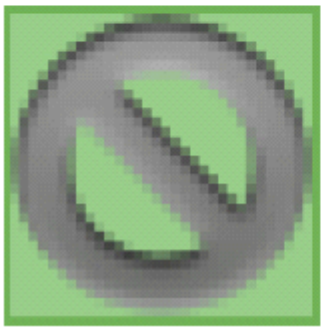


**Figure 4.** Safeframe and fill gradient for tab icons in selected state. Icon size is 32x32.

**未选中 tab 图标**

**发光，特效和阴影**

未选中图标和选中图标类似，但有一个更深的内阴影，并且填充渐变色与对话图标一致。


**Figure 5.** Light, effects, and shadows for unselected tab icons.

1. 前景部分：渐变叠加->角度 90°、底部颜色：RGB223 223 223、顶部颜色：RGB249 249 249、底部颜色位置：0%、顶部颜色位置：75%
2. 内阴影：黑色 10%不透明度、角度 90°、距离 2px、大小 2px
3. 内斜面：深度 1%、方向 向下、大小 0px、角度 90°、高度 10°、高光：白色 70%不透明度、阴影：黑色 25%不透明度

**步骤**

1. 用 Adobe Illustrator 创建基本形状
2. 导入这个形状到 Adobe Photoshop 中，并缩放成 32x32px 的透明度图片
3. 为为选中状态添加效果，如图 5
4. 导出带透明度的 32x32 大小的 PNG 文件

## 选中 tab 图标

选中图标的渐变填充和效果与菜单图标类似，但没有外发光


Figure 6. Light, effects, and shadows for selected tab icons.

1. 前景部分：使用调色板的渐变填充
2. 内阴影：黑色 20%不透明度、角度 90°、距离 2px、大小 2px
3. 内斜面：深度 1% 、方向 向下、大小 0px、角度 90°、高度 10°、高光：白色 70%不透明度、阴影：黑色 25%不透明度

**调色板**

填充渐变

1. RGB：163 163 163

2. RGB：120 120 120

在未选中图标上使用颜色填充

**步骤**

1. 在 Adobe Illustrator 中创建基本形状
2. 导入到 Adobe Photoshop 中并缩放到 32x32px 的带透明背景的画板中
3. 为选中状态增加特效，如图 6 中所示
4. 导出 32x32 的 PNG 透明图标文件。

# 5. 对话框图标

对话框图标出现在弹出对话框中，提示信息的作用。这些图标使用亮颜色的渐变和内阴影以在黑色背景中突出。

就像“提供特定精度图标集”所描述的，需要创建适用于低、中和高三种精度屏幕的特定图标集。这是确保应用程序在多种设备上能正常显示和安装。参考“设计师建议”章节的建议如何创建多精度图标集。如表 1 所示列出各个精度的图标大小。同样的，参考“设计师建议”这一章节建议如何设计多种设备的图标集。

表 1：展示 3 个广泛意义上的屏幕精度的最终对话框图标尺寸

| 低精度屏幕 *(ldpi)* | 中精度屏幕 *(mdpi)* | 高精度屏幕 *(hdpi)* |
|---|---|---|
| 24 x 24 px | 32 x 32 px | 48 x 48 px |

**最后导出带透明的 PNG 文件，不能有背景颜色。**

创建图标所需要的 Adobe Photoshop 文件模板在“图标模板包”里。

## ● 所有 Android 版本

接下来介绍如何为所有 Android 版本设计对话框图标。

**结构**

• 对话框图标包括 1px 的安全边界。基本形状应在安全边界以内，但圆形可以覆盖安全边界。

•此页面的所有指定尺寸基于 32x32px 画板中。在 Photoshop 模板保持四周 1px 的边缘。

图 1：对话框图标以渐变色填充，图标大小为 32x32。

**发光、特效和阴影**

对话框图标是一个平面的正视图。为了能在黑色背景中有所突出，使用了亮一点的渐变和内阴影。


Figure 2. Light, effects, and shadows for dialog icons.

1. 前景：渐变填充->角度 90°、底部：RGB 223 223 223、顶部：RGB：249 249 249、底部颜色位置：0%、顶部颜色位置：75%
2. 内阴影：黑色、25%不透明度、角度-90°、距离：1px、大小 opx

**步骤**

1. 在 Adobe Illustrator 中创建基本形状
2. 导出到 Adobe Photoshop 中缩放到一个透明的 32x32px 大小的画板图像中
3. 为合适填充增加特效，如图 2
4. 导出 32x32 的 PNG 透明图标文件。

## 6. 列表视图图标

列表视图图标类似于对话框图标，区别是当在亮的资源背景上使用一个内阴影。这类图标的设计只用于列表视图中，比如在设置应用中。

就像“提供特定精度图标集”所描述的，需要创建适用于低、中和高三种精度屏幕的特

定图标集。这是确保应用程序在多种设备上能正常显示和安装。参考“设计师建议”章节的建议如何创建多精度图标集。如表 1 所示列出各个精度的图标大小。同样的，参考“设计师建议”这一章节建议如何设计多种设备的图标集。

表 1：展示 3 个广泛意义上的屏幕精度的最终对话框图标尺寸

| 低精度屏幕 *(ldpi)* | 中精度屏幕 *(mdpi)* | 高精度屏幕 *(hdpi)* |
|---|---|---|
| 24 x 24 px | 32 x 32 px | 48 x 48 px |

**最后导出带透明的 PNG 文件，不能有背景颜色。**

创建图标所需要的 Adobe Photoshop 文件模板在“图标模板包”里。

- **所有 Android 版本**

  接下来介绍如何为所有 Android 版本设计列表视图图标

  **结构**

  • 列表视图图标通常由 1px 的安全边界，但圆形区域可以在安全边界中重叠。
  • 所有指定尺寸基于 32x32px 画板中。在 Photoshop 模板保持四周 1px 的边缘。

**Figure 1.** Safeframe and fill gradient for list view icons. Icon size is 32x32.

### 发光、特效和阴影

列表视图图标在一个内阴影中的正视图平面。由亮色渐变和内阴影组成，在黑色背景中更突出。


Figure 2. Light, effects, and shadows for list view icons.

1. 内阴影：黑色、57%不透明度、角度 120°、混合模式：正常、距离 1px、大小 1px
2. 背景：黑色、系统标准颜色　这些图标只在列表视图中显示。

注：列表视图图标在 Photoshop 中的 32x32 的画板中，没有安全边界。

**步骤**

1. 在 Adobe Illustrator 中创建基本形状
2. 导出到 Adobe Photoshop 中缩放到一个透明的 32x32px 大小的画板图像中
3. 为合适填充增加特效，如图 2
4. 导出 32x32 的 PNG 透明图标文件。

# 二、Widget 设计指南

Widget 是在 Android1.5 时引入的特性之一。Widget 可以让用户在主屏幕界面及时了解应用程序显示的重要信息。标准的 Android 系统内置多个 Widget，如：模拟时钟、音乐播放器等。

用户在主屏幕(HomeScreen)界面的空白区域长按,选择菜单的”小部件”项,即可随意选取所需的部件并显示在主屏幕上.

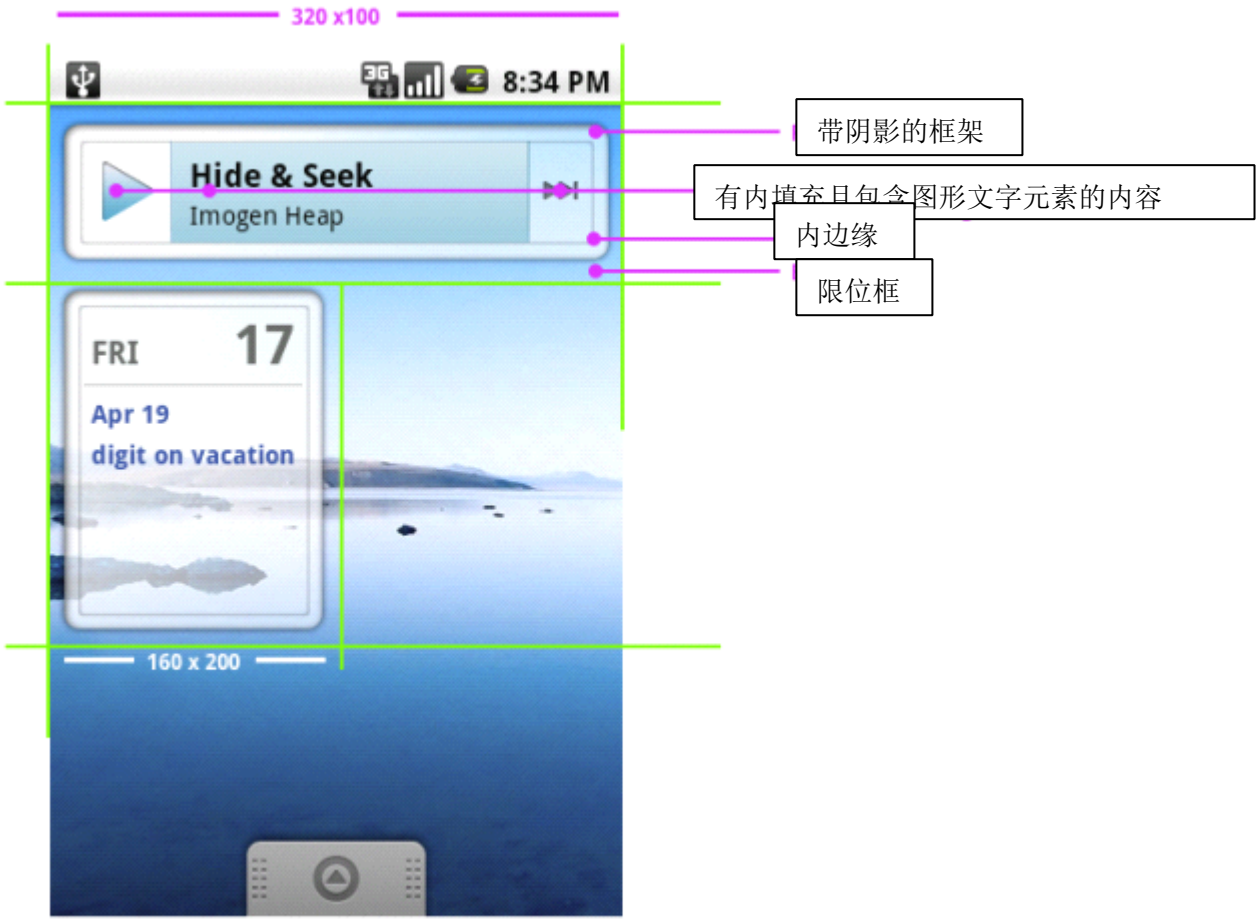

此文档介绍了如何设计一个 Widget，使得和其它 Widget 以及主屏幕其它元素保持美观一致. Android 团队也于此文档介绍了 Widget 源图的一些设计标准，还有 Widget 制图的技巧诀窍.

对于开发 Widget 的相关信息， 可参考开发者指南的 AppWidgets 章节和 AppWidgets 博客.

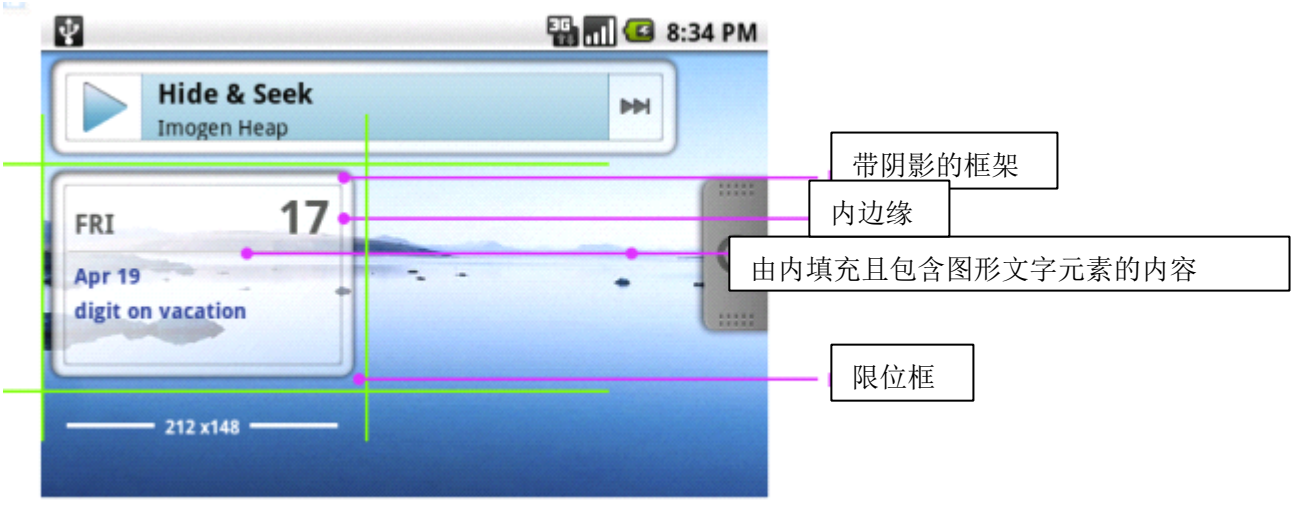
## ● 标准 Widget 解析

典型的 Android Widget 主要有三个组成部分：一个限位框，一个框架，还有 Widget 的图形控件以及其它元素. 设计周全的 Widget 会在限位框边缘&框架之间，及框架内边缘&Widget 的控件之间都保留一些内填充(内补白). Widget 的外观被设计得与主屏幕的其它 Widget 相匹配，并以主屏幕的其它元素为依据对齐；它们亦使用标准的阴影效果. 此文档说明了所有的相关细节.

**纵向---Widget 的标准尺寸**




**横向—Widget 的标准尺寸**




- **设计 Widget**

1. 设置 Widget 的限位框尺寸
   要在最小的 Widget 尺寸中，让应用程序的最有用和及时的数据最有效的显示在 Widget 中。用户将权衡 Widget 的有用性和是否占用主屏幕的空间，所以 Widget 越小越好。

所有 Widget 必须适应于六个支持 Widget 尺寸的边界框，或者说，需要一对适应横向和纵向的尺寸，所以当用户切换屏幕方向时，Widget 看起来要友好些。

“标准的 Widget 尺寸”如图示，有 6 种尺寸（3 种垂直方向和 3 种水平方向）

2. 选择一个合适的框架

“标准的 Widget 框架”如图示有 6 种标准框架尺寸，这些链接可以下载使用。Widget 不一定使用这些框架，但如果这样做，Widget 更能与其他 Widget 风格上统一。

3. 图形上使用标准的阴影效果

此外，若不使用这些效果，“标准 Widget 阴影”说明了标准 Widget 使用的 Photoshop 设置。

4. Widget 中有按钮，需要有三种状态（正常、按下和选中）

可以下载“播放按钮的三种状态的 Photoshop 文档”，来自于音乐播放 Widget，分析三个标准按钮效果可以使用 Photoshop。


5. 完成源图的绘制并调整比例和对齐

“Widget 对齐的技巧和窍门”这一章节介绍了在标准框架内 Widget 图形的对齐技巧，另外还有一些其他 Widget 的图形技巧。

6. 正确的图形文件保存 Widget

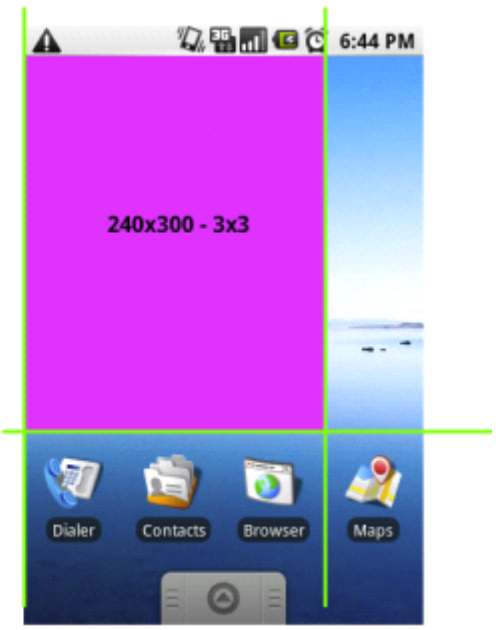
“窗口图形文件格式”介绍了如何正确保存 Widget 图形文件

## ● 标准 Widget 尺寸

有 6 种 Widget 大小，基于主屏幕的 4x4（纵向）或者 4x4（横向）的网格单元。这些为 6 个标准控件尺寸的边界框尺寸。典型的 Widget 内容不要超出这些尺寸的边缘线，但是在限位框里填充一个框架，正如“设计 Widget 时”所介绍的。

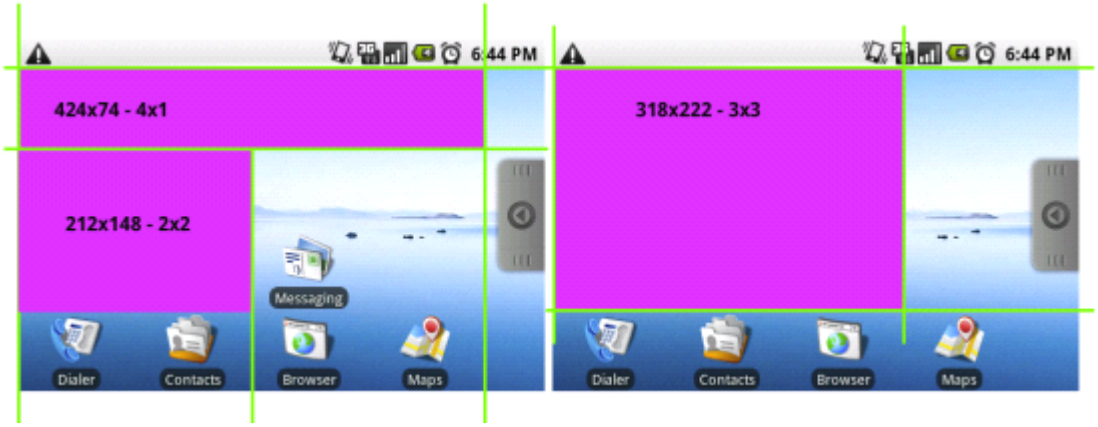
纵向，每个单元格为 80px 的宽度、100px 的高度（如图所示纵向的单元格），三种纵向的 Widget 尺寸如下：

| 单元格 | 像素 |
|---|---|
| 4 x 1 | 320 x 100 |
| 3 x 3 | 240 x 300 |
| 2 x 2 | 160 x 200 |


横向，每个单元格 106px 的宽、74px 的高，三种横向的 Widget 尺寸如下：

| 单元格 | 像素 |
| --- | --- |
| 4 x 1 | 424 x 74 |
| 3 x 3 | 318 x 222 |
| 2 x 2 | 212 x 148 |




- **标准 Widget 框架**

针对六种标准 Widget 框架尺寸中都有一个标准框架，你可以点击如图所示的框架下载 PSD 框架文件，通过这个文件来设计你的 Widget。

4x1_Widget_Frame_Portrait.psd

/docs/images/widget_design/4x1_Widget_Frame_Portrait.psd

3x3_Widget_Frame_Portrait.psd

/docs/images/widget_design/3x3_Widget_Frame_Portrait.psd

2x2_Widget_Frame_Portrait.psd

/docs/images/widget_design/2x2_Widget_Frame_Portrait.psd

4x1_Widget_Frame_Landscape.psd

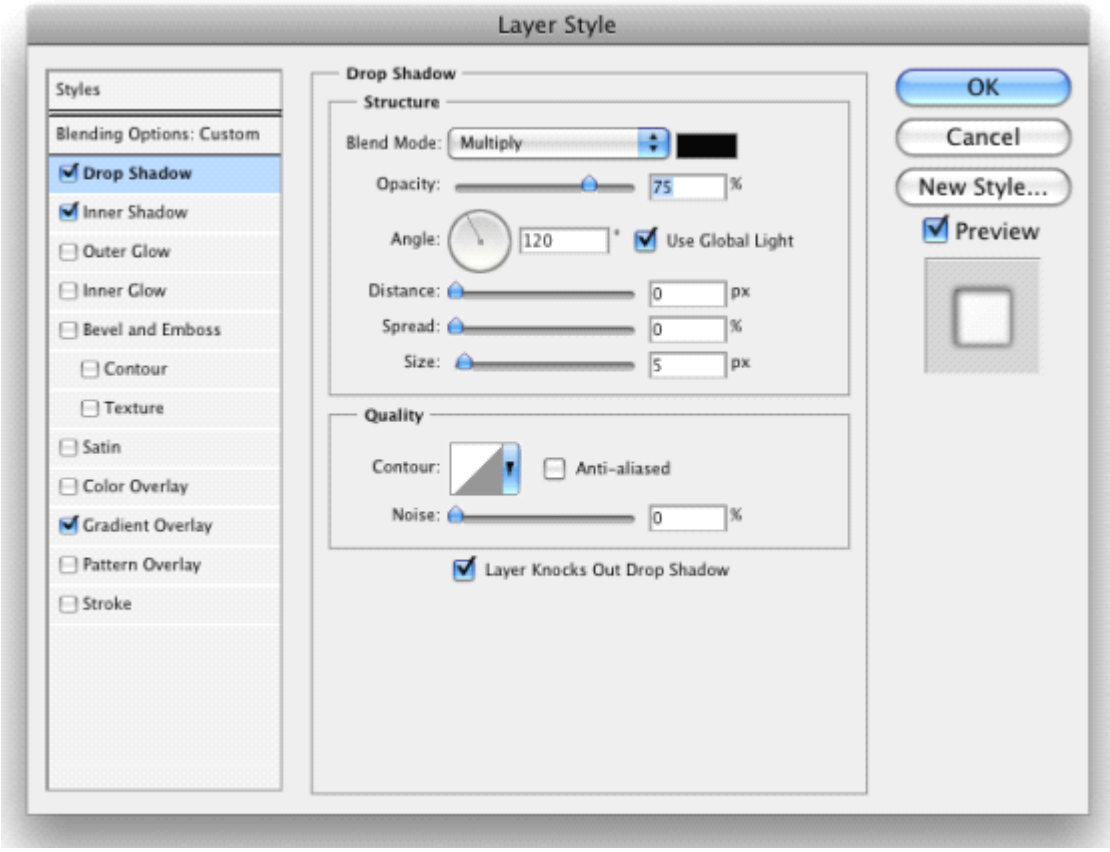
/docs/images/widget_design/4x1_Widget_Frame_Landscape.psd

3x3_Widget_Frame_Landscape.psd

2x2_Widget_Frame_Landscape.psd

- **标准 Widget 阴影**

在 Widget 图层应用阴影特效，使用底下的 Photoshop 图层面板设置与其他标准的 Android Widget 相匹配。

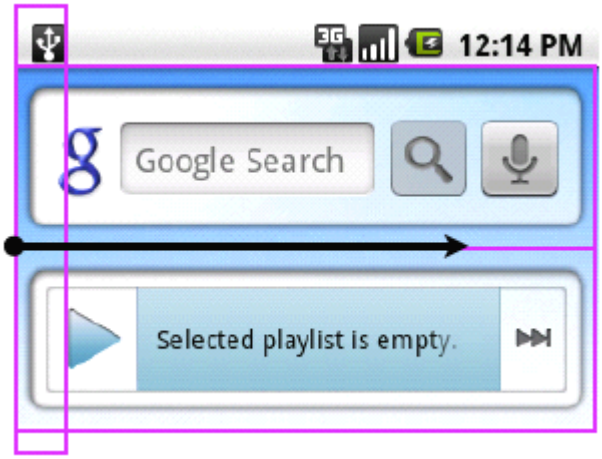



- **Widget 绘制技巧诀窍**

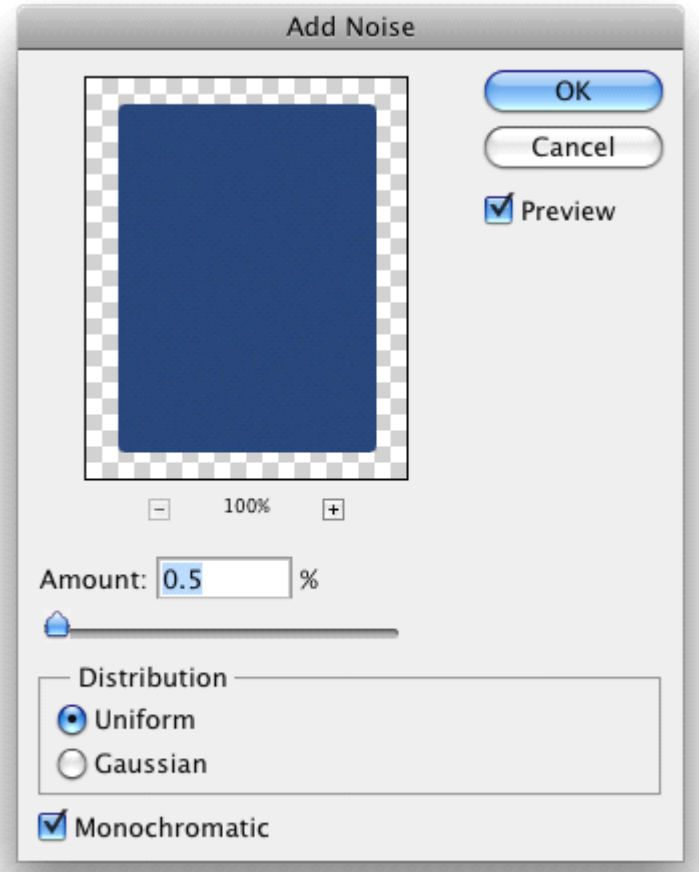
Android 团队展示了一些用于对齐 Widget 图片和标准 Widget 限位框和框架的技巧，使数个 Widget 以及主屏幕上的其他元素在视觉上对齐，除此之外还有一些创建 Widget 的技巧。

使用从 Android SDK 模拟器的屏幕截图工具，让你的 Widget 控件对齐搜索控件和主屏幕上的其他元素的形状和阴影。

从全尺寸单元格裁剪 Widget 的富余部分，包括任何填充空间。（底下是 4x1Widget，减去 320x100px 的资源）



• 为了减少 Widget 输出的条状色块，应用 Photoshop 的“添加杂色”来设置图片



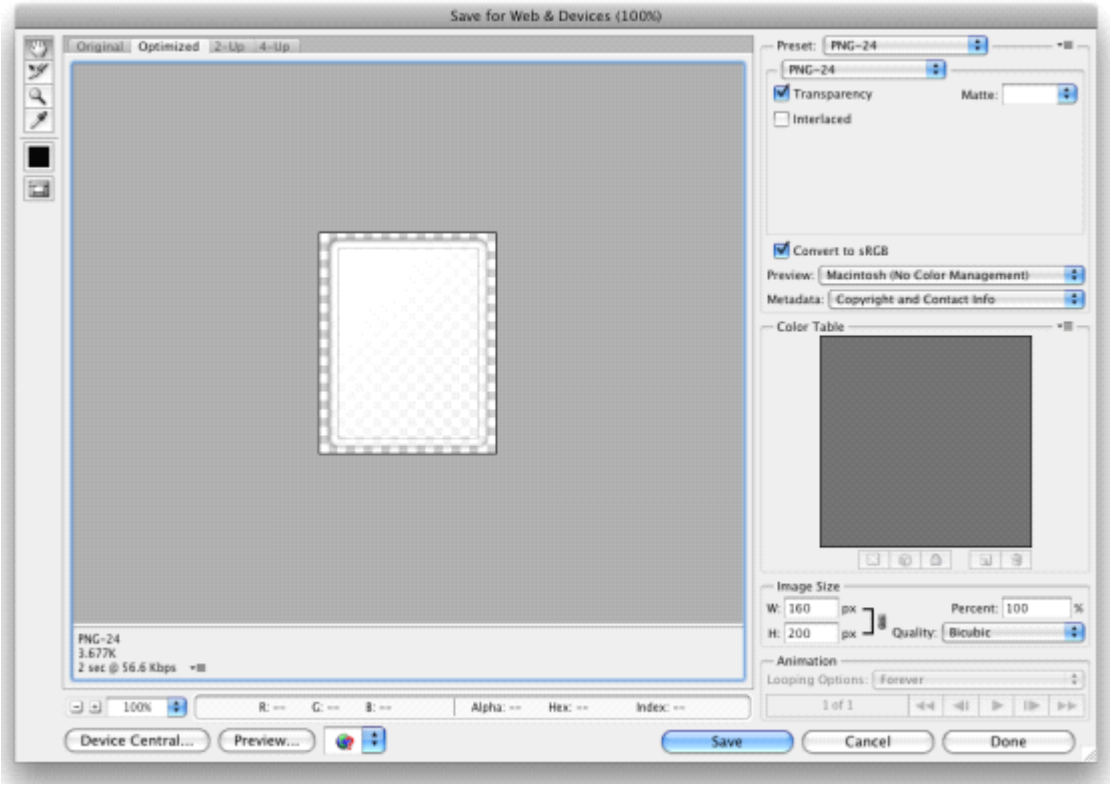

• 使用“9-patch”图片格式技术缩放图片并设置内容区域内填充（查看 /docs/guide/developing/tools/draw9patch.html）

注：当前 Android Widget 模板被设计为用于普通的渐变角度，意思是“9-修补”技术不能用于优化资源尺寸。无论如何，“9-patch”图片格式技术都能设置内容区域的内填充。

• 在某些情况下，低精度的设备，可能引起视觉带抖动问题。为了解决这个问题，应用程序开发人员应该通过“代理”为 XML 定义绘制资源：这种技术引用下的原始图片，“background.9.png”这种情况下，指示设备需要抖动。

## ● Widget 的图形文件格式

保存 Widget 图片使用合适的透明图片框，用 PNG-24 格式和 8 位色调




# 三、Activity and Task 设计指南

这篇文档主要讲述 Android 应用框架的核心原则。站在高层来说，以用户为中心来设计良好的交互程序，对于应用设计者或是开发者来说是非常重要的。

下面用例子来阐述了 activities 和 tasks 的一些底层原则和机制 ，例如导航，多任务，activity 重用，意图和 activity 栈。这篇文档也着重讨论了一些设计决策，针对如何利用好它们去设计你的应用程序 UI。

这篇文档中的例子均是 Android 应用程序，包括默认应用程序（比如拨号器）、Google 应用程序（比如地图）。你可以自己在 Android 模拟器上或是 Android 手机上去试验这些例子。如果你使用 Android 手机试验时，可能未提供本文档中的某些例子。

再看这篇文档之前，请确保您看过本篇中的 Design Tips 章节。此篇算做是 Application Fundamentals 文档的的一部分（特指 Tasks and Back Stack 章节），它对于程序员来说覆盖了整个底层机制。

# Applications, Activities, Activity Stack and Tasks

理解 Android 系统中的四个基本概念对你是很有帮助的，它们分别是：

- Applications
- Activities
- Activity Stack
- Tasks

## Applications

一个 Android 应用程序其实就是由一个或者多个 Activity 组成。它们被捆绑在一起并存放进.apk 文件中，这就是 Android 应用程序。Android 中有着丰富的应用程序，比如邮件、日历、地图定位、文本消息、联系人、照相机、打电话、音乐播放器，系统设置等应用。

一般情况下，桌面上都会有 Android 应用程序快捷图标，用户可以选择某一个图标来启动应用程序。

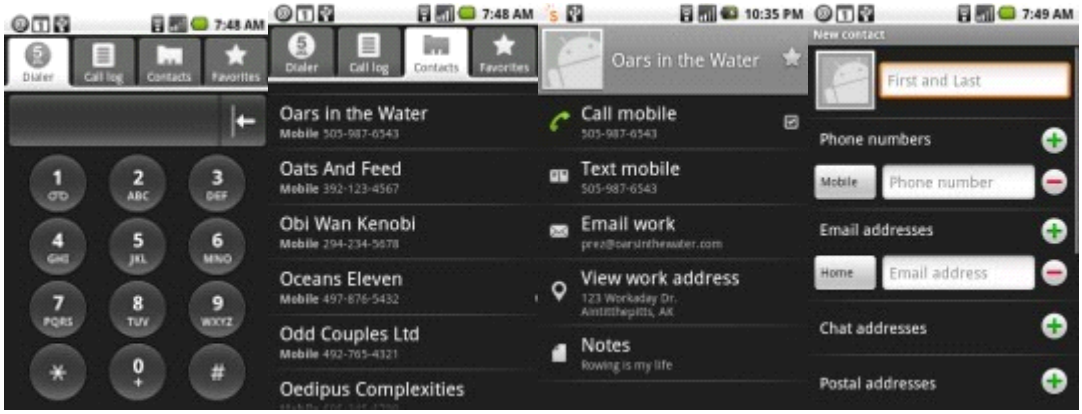
## Activities

Activity 是 Android 应用程序的主要组成部分，activity 可以是你自己创建的，当然，还可以是其它应用程序中的 activity。它们都是在运行时绑定上的，以便于应用程序扩展其自身的功能，它们一旦组合在一起，就会如同一个应用程序一样。每个 activity 都有其独特的 UI、明确的功能，诸如打电话、看照片、听音乐等。任何一个应用程序都应至少有一个 activity。

当使用 Android 手机时，用户在屏幕上一个接一个地滑动或是点击图标启动 activity，完全不会注意到底层的行为——他们体验是无缝的。activity 后面有 activity，task 后面还会有 task。

Activity 可以处理特定的数据类型和接受一相关的动作。每个 activity 都有其各自的生命周期，互不干扰；并且它们都可以被用户或者系统独立启动、运行、暂停、恢复运行、停止、重新开始。正因为这个独立性，activity 可以以不同的方式被其它的 activity 进行替换。

比如打电话的应用程序就包含了四个 activity：打电话、联系人列表、查看联系人、添加联系人，如下图：




打电话　　联系人列表　　查看联系人　　添加联系人

下面的应用程序同样也包含了很多 activity：

- 邮件 - 查看文件夹、邮件列表、邮件，发送邮件和设置邮件账号。
- 日历 - 查看天、星期、月、议程，编辑事件、首选项。
- 照相机 - 运行照相机、查看图片列表、图片，编辑图片，运行录像机，查看录像列表和录像。
- 游戏 - 玩游戏和安装游戏。
- 地图 - 查看地图上的位置，查看朋友的位置以及他们的详细信息（朋友的位置、状态、照片）。

Activity 是 Android 应用中最为突出的组件，其余组件分别为：service、content provider、broadcast receiver。更多 activities 的详情，参见 Application Components。

## ● Activity Stack

用户之所以能够从一个 activity 转到下一个 activity，是因为 Android 系统针对 activity 而设计了一个线性的导航历史以供用户追溯访问，这就是 activity 栈，也称为 back stack。当用户启动了一个新的 activity，它就被添加进 activity 栈，以便按 BACK 键时能够返回到上一个 activity。然而，用户不能按 BACK 键就直接返回到桌面（除非 activity 的前一个是桌面才可以）。

activity 栈里面存放的只能是 activity，而视图、窗体、菜单和对话框则不能。也就是说，如果你可以让用户从屏幕 A 跳转到屏幕 B，当用户按 BACK 键时，他就应该会回到屏幕 A，那屏幕 A 必须是一个 activity。有个例外情况就是，你的应用程序需要利用 BACK 键控制自身的导航，那就要自己重新设定 BACK 键的导航功能。

- **Tasks**

任务则是一系列的 activity 集合，它能使用户完成既定的操作，而又不用去关心这些 activity 是哪个应用程序里面的，除明确指定一个新任务之外（参见“中断任务”小段），那么其他 activity 都属于当前任务的一部分。再次注意的是，这些 activity 可是任意应用程序中的其中一个，也就是说不管它们所属的应用程序是否相同。举个例子，用户打开了联系人的程序，任务随之也会启动，他选择了 email 地址准备发邮件，这时跳转到了 email activity，之后他要添加附件，需要在图库中挑选图片。这里面，联系人，email，图片图库都是不同的应用程序。

当一个 activity 启动时，任务也随之启动的话，那个 activity 就是根 activity。启动 activity 一般有这么几种方式，应用程序发射器、桌面快捷方式、最近任务切换器。Android 系统内部一旦有任务，那么按 BACK 键就可以回到上一个 activity。Activity 栈可以是多个任务的组成部分。

下面是关于任务的例子，以供参考：

✉ 发送文本消息并含有附件

✉ 观看 YouTube 视频并以邮件的方式向其他人分享。

**中断任务** ——任务中有一个重要特性就是能使用户中断当前正在做的操作（他们的任务）而去执行其它操作，当然他们也可以回到之前的任务上，也就是说支持同时运行多任务并且来回切换它们。

这里有两种情况来开始其它任务，并且都可以返回到原先的任务上。

✉ 打开通知：用户接收到通知并打开查看它。

✉ 用户转去做其它操作：用户在桌面启动。

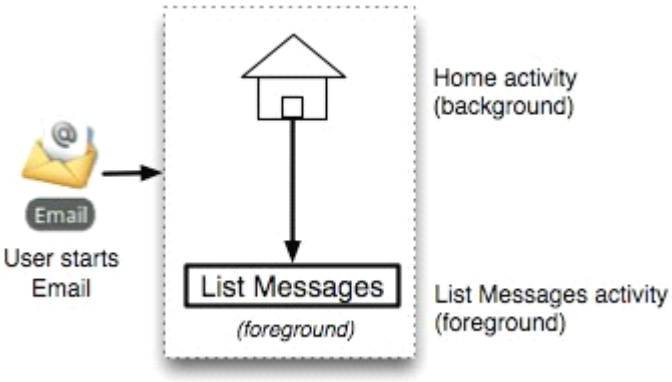
当然，也有例外的。除了刚才提到的两种方式，另外还有一种打开新任务的情况，就在其内部开启一个 activity。例如，在邮件中以新任务的方式打开地图 activity 或是打开一个浏览器 activity，当按 BACK 键时就又会回到邮件 activity 中。

## Activities 和 Tasks 之旅

下面的例子阐述了应用程序的基本原则，主要有 activities，activity 栈，回退键，任务和意图；并展示了系统是如何响应用户请求的，例如用户开始了一个应用程序，用户不断的切换 UI，程序内部就是利用在不同任务之间切换 activities 的。下面的许多例子你都可以在 Android 手机上运行起来。

**在桌面上开始一个 Activity**

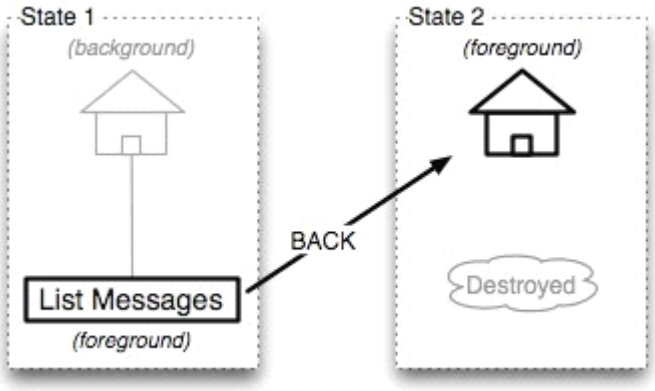
桌面是启动应用程序的主要地方，比如在桌面上点击应用程序图标就能将其打开，用户第一眼看到的就是应用程序中的主 activity。如下图，所描述的是用户在桌面点击 Email 图标所发生的事情：




- **用 BACK 和 HOME 键进行导航**

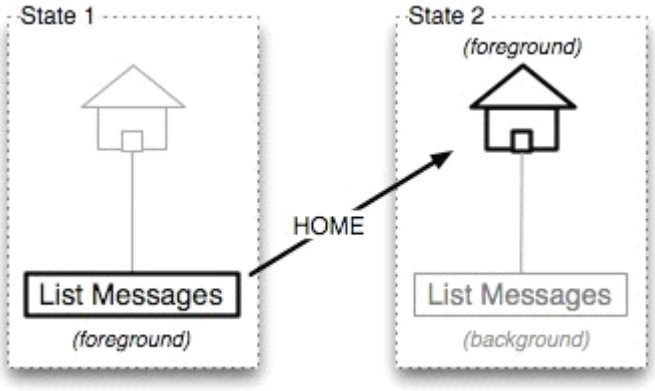
Activity 保持或者丢掉其状态完全取决于用户是怎样离开这个 activity 的——使用 HOME 键还是 BACK 键。

默认情况下，按下 BACK 键来结束（销毁）当前 activity 并为用户显示上一个 activity。例如下图，用户在桌面上打开了邮件，当前 activity 显示着邮件列表。用户将列表滚动条往下拉以便看到后面的邮件，这时用户按了 BACK 键，那么 Android 就会销毁这个邮件信息列表 activity 并返回之前的 activity（桌面）。然后用户又重新打开邮件，还是那个列表，但是滚动条又回到了起始位置上。




上面的例子中，当按下 BACK 键就返回到了桌面，由于那是用户在上一次看到的 activity。不过如果用户从其他 activity 里面跳转到邮件列表，然后按下 BACK 键则回到了先前的那个 activity（这里只是说明一下 BACK 键的作用）。

相比之下，下面的图就是用 HOME 键离开邮件列表 activity 而不是 BACK 键，那么当前 activity 就呈 stop 状态并移置后台而不是销毁。当再次打开邮件列表 activity 时状态保持不变。




其它情况：有些应用程序则不是如上面所说的那样。例如联系人和图库，用户在桌面打开联系人后查看了某个联系人的资料，接着再次打开联系人时，就不会显示之前的 activity 了。这是因为联系人的主 activity 有四个标签，是为了让用户能够看到全部的功能特性。

此外，也不是所有 activity 都是当按下 BACK 键之后销毁掉的。例如用户开始播放音乐，接着按下 BACK 键，却不会影响音乐的播放。即使它的 activity 不再可见，音乐应用程序依然

会在状态栏上提示着用户。注意：你也可以让 activity 不再可见时停止掉或是继续在后台运行，但后者更适合像音乐这样的应用程序。

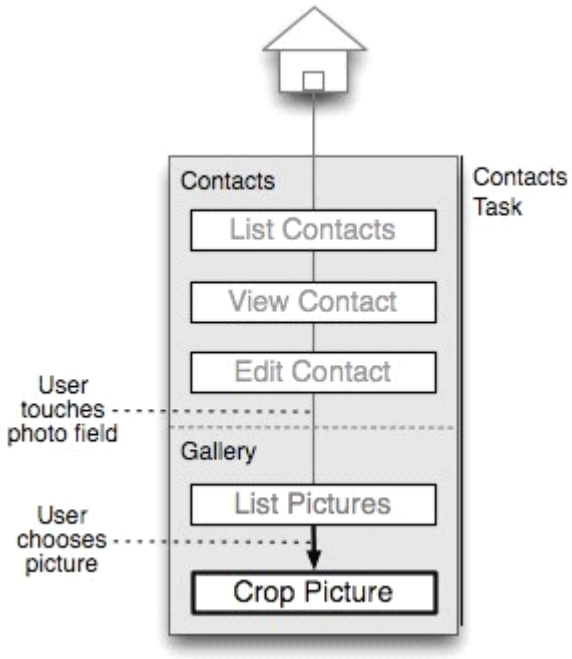
- **重用 activity**

有两个应用程序中，它们分别也有两个 activity：activity A 和 activity B。A 的部分功能需要调用 B 的已实现功能，那么 B 就叫被重用。

**联系人重用图库来获取图片** — 联系人 activity 中会有联系人的照片，但是照片一般存放在图库里面，所以联系人要重用图库的功能来获取图片，图库 activity 就是重用的绝佳例子。下面的图画出了重用的流程。具体流程是这样的：用户打开了联系人，查看某一个联系人的资料并想编辑他的照片，这时，打开了图库 activity，对图片进行设置并保存，那个联系人的图片也就相应的改变了。

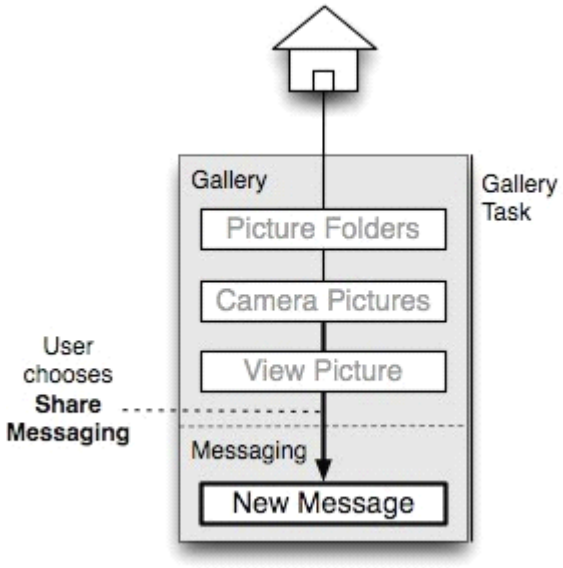
注意图库最终会返回给联系人一张图片。 下一个例子讲述一个 activity 的重用并且不返回任何结果。同样需要注意下面的插图是说明通过 activity 或是 activity 栈来实现历史导航——用户可以通过每个 activity 用任何方式回到桌面。

当开始设计一个应用程序时，一个不错的想法就是怎样能够在重用其它应用程序中的 activity 或是你的 activity 怎样被其它应用程序重用。如果用一样的 intent filter（已经存在了一个 activity）再添加一个 activity，那么系统会为用户显示出一个选择 UI，供用户选择使用那个 activity。




**图库重用短信来与其他人分享图片**。分享也是不同应用程序之前重用的好例子。如下图所示，用户打开了图库，从中挑选了一张图片并点击了共享菜单，选择“短信”。这时，就打开了短信 activity，在其里面写些文字和附加上那张图片之后发送出去。用户现在在短信 activity 当中，如果想回到图库 activity，就按 BACK 键返回。

注意这里的短信 activity 并没有给图库返回任何的东西。



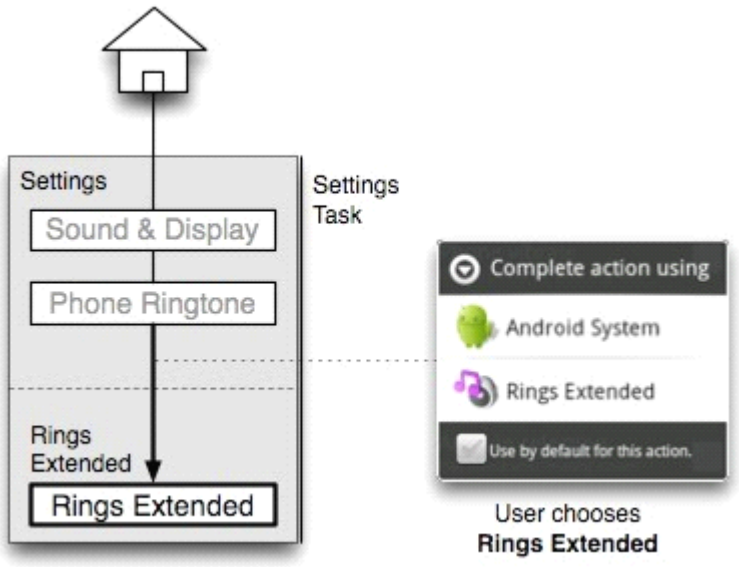

这些例子都在阐述任务——一系列的 activities 都在实现同一个目标。每一个例子中的 activity 都是从两个不同的应用程序中完成本职工作的。

- **替换 activity**

这个例子描述的是不同应用程序中的两个 activity 互相替换，activity A 替换 activity B。这种一般发生在 activity A 比 activity B 的功能更为强大一些。

换句话来说,A 和 B 妥妥得等价，当然就可以实现 A 替换 B。这个例子中的联系人应用程序重用了 activity，A 和 B 虽然是完全不同的 activity，但是它们两个彼此形成了互补，使程序更加的强大。

在这里例子中，用户下载了一个手机铃声的 activity，称之为“铃声扩展”。用户这时进入到“设置 > 声音&显示 > 手机铃声”里面，系统会展示两个可用 activity 供用户选择。此时弹出的对话有一个选项是让你设置“是否默认使用此 activity”，选中它。当用户选择“铃声扩”时，以后在加载的时候就替换了 Android 默认铃声的 activity 了。

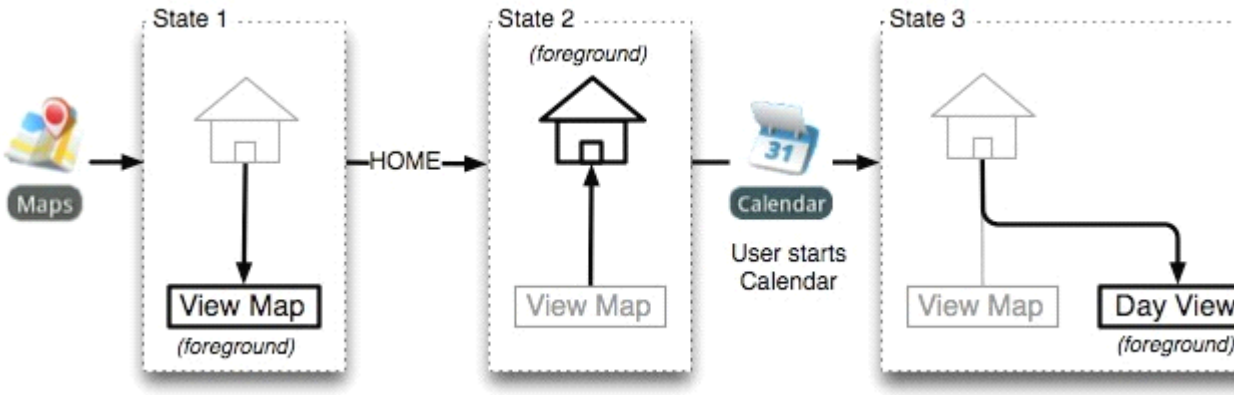



- **多任务**

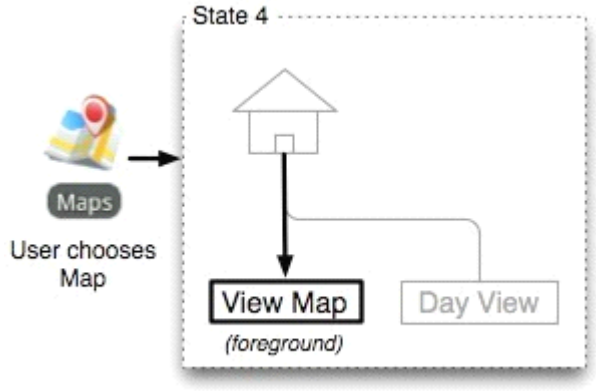
如前所述，当一个 activity 启动后，用户还可以回到桌面启动第二个 activity，第一个 activity 则不会被销毁还是继续运行着，我们换个例子来说明这一小节——地图应用程序。

✉ 状态 1：用户打开了地图应用程序并查询一个地址。这时，用户该说了，网络太 TMD 慢了！因为地图定位是需要一些时间的。

- 状态 2：用户准备做些其它事情，按下 HOME 键，不过这样做不会干扰地图应用程序，还是保持其加载地图的状态。
- 状态 3：地图 activity 现在是在后台运行着，桌面在来到了前台。这时用户打开了日历 activity，比如查看今天是星期几。




- 状态 4：用户回到桌面，重新打开 Map，这时地图已经全部加载完毕了。



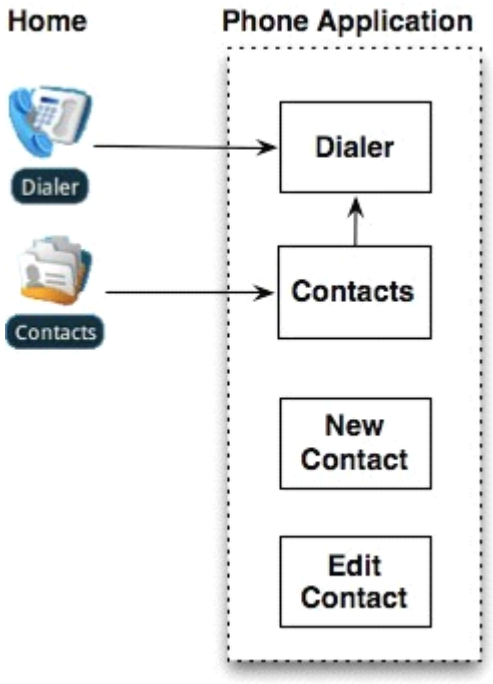

以上两个应用程序“地图”和“日历”是两个不同的任务，因此 Android 支持多任务模式。

## ● 两个入口点

相对于应用程序来说，必须至少要有一个入口点，也就是至少要有一个 activity。桌面上的图标就代表着每个应用程序的入口点，同样也可以在其它应用程序中启动，当然，它们的入口点都在其内部。

而电话应用程序就有两个入口点：联系人和打电话。用户进入到联系人里面选择了一个电话号码并拨打该电话。如下图的图所示，用户打开联系人，也就是启动了联系人的 activity，然后选择了一个电话号码随之进入了打电话的 activity，最后拨打它。

一旦用户在应用程序里，它们就可以通过标签、菜单项、列表项、屏幕上的按钮或其他用户界面访问诸如新增联系人和编辑联系人。




- **意图(Intents)**

用户点击一个 mailto:的连接时，这实际上就被看作是一个意图，发邮件的意图。

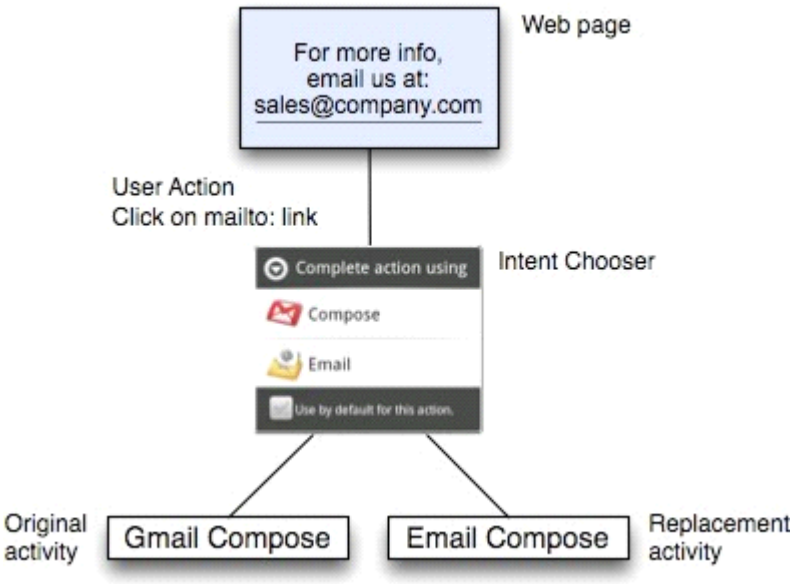
关于意图有三点要说明：

✉ 如果是显式意图，Android 就会立即启动那个 activity。

✉ 如果是隐藏意图，Android 先去 intent filter 寻找合适的 activity 再启动。

✉ 如果有多个合适的意图，Android 就会列出一个意图选择列表供用户选择。

下面就举用户发邮件的例子，此时用户的 Android 上有两个邮件应用程序，当他在页面点击了 mailto:链接的时候，Android 会提示给他一个对话框，其中有两个可用的程序供其选择（Gmail 和 Email）。




下面列举一些常用的意图和其对应的 activity：

- 查看联系人列表：对应联系人列表查看 activity
- 查看指定的联系人：对应联系人查看 activity
- 编辑指定的联系人：对应联系人编辑 activity
- 发邮件：对应邮件 activity
- 拨打电话：对应电话拨打 activity
- 查看图片列表：对应图片列表查看 activity
- 查看指定的图片：对应图片查看 activity
- 裁剪指定的图片：对应图片裁剪 activity

意图必须由两部分构成：动作和数据。

- 动作：由上面的意图列表中可得出，查看、编辑、打电话、裁剪
- 数据：由上面的意图列表中可得出，联系人的列表、指定的联系人、电话号码、图片列表、指定的图片。

注意：任何在桌面上启动的应用程序都是显式意图，目的是指定其内部特有的那个 activity。同理，应用程序也可以在内部以显式意图的方式启动自身的 activity，外部 activity 都是访问不到它们的。

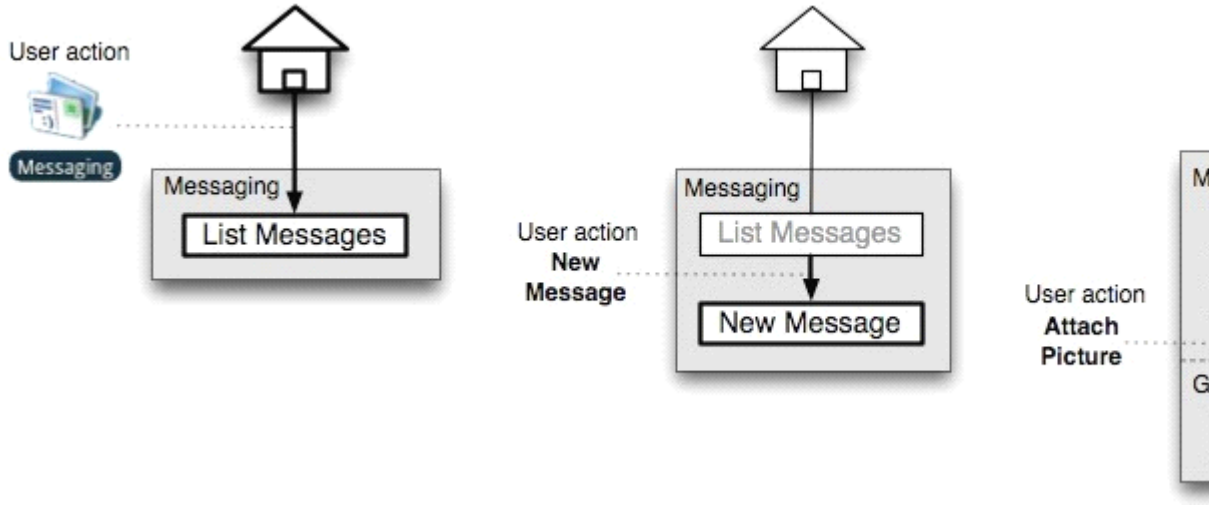
关于意图更多信息，参见 Intents and Intent Filters。

- **切换任务**

下面的例子描述的是用户如何在两个任务之间进行切换。

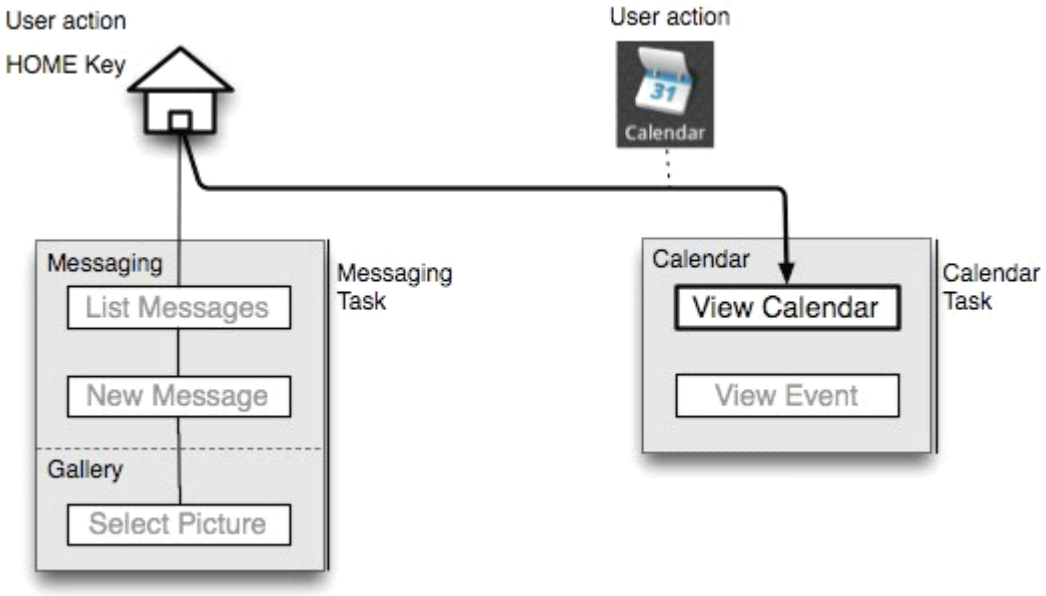
1. **开始第一个任务**。你想要发送一条短消息并附加一张图片。你会这样操作：

   桌面 > 短消息 > 新的短信息 > 菜单 > 附件 > 图片。最后一步启动了图库 activity 来选择一张照片。注意图库是另外的一个应用程序。



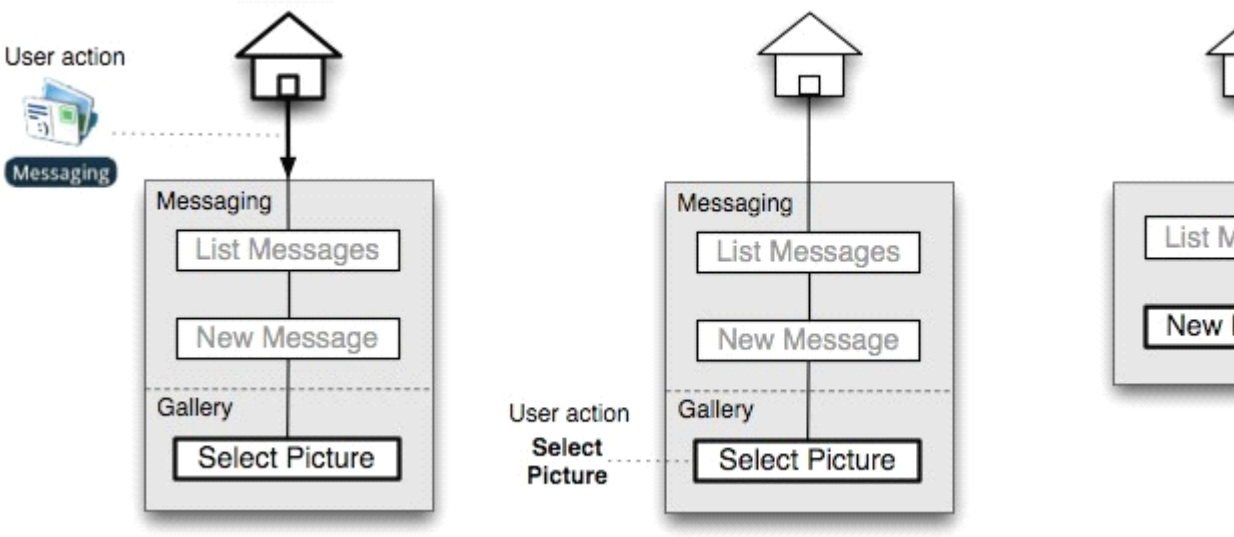

   在选择照片之前，可以先去桌面打开日历，目的是为了开始第二个任务。

2. **开始第二个任务**。你会这样操作：桌面 > 查看日历。从桌面上打开日历，就等于是开始一个新任务了。



3. 切换到第一个任务并完成后面的操作。

查看完日历之后，继续回到先前的任务上：桌面 > 短消息，此时进入的并不是短消息 activity，而是图库 activity，也就是之前离开的 activity。然后你就可以选择图片并发送短消息出去了，也就完成了第一个任务。




## 设计小贴士

下面的提示和指导都是针对应用设计者和开发者而提出的。

- **使用显式意图来防止外部应用调用你的 activity**

如果你不想自己的 activity 被外部使用，就别在 manifest.xml 里面配置 intent-filter。这样的话，你的 activity 就只会在应用程序内部来启动了，同样也避免了安全漏洞。反之，创建一个意图并指定明确要启动的组件，这就是显示意图，在这个例子中，就不需要 intent filters。Intent filters 可以发布所有的应用程序，当你创建了一个 intent filter 时，其它应用程序就可以访问到你的 activity 了，至于它们怎么用，你就不知道了，这意味着不经意间形成了安全隐患。

- **如果使用外部的 activity，但却没有匹配上，该怎么办？**

有这一种情况，你利用 Intent 去调用外部应用程序中的 activity，但遗憾的是，那个应用程序并没有安装进手机里，因此我们需要妥善的处理这种情况。

（译者注：官方提出了两个不太完善的解决方案，我们来看下：）

1. 在启动那个 activity 之前用 intent 先对其测试一下。
2. 如果启动 activity 会失败的话，则捕获它的异常信息。

以上更多信息请参阅官方文档提供的博客文章：Can I use this Intent?。

该博文中提供了一种比较好的解决方式，正如其提供的样例代码中的 isIntentAvailable()方法，我们可以在初始化阶段调用它；如果该应用不存在的话，我们就给用户提示一条消息，告诉他某某应用不存在，请去 Google Market 下载等友好信息。如果要让意图决定显示哪个 activity，那我们就使用 startActivity()或是 startActivityForResult()来启动 activity。

## ● 思考：以怎样的方式来启动 activity

做为 Android 设计者或开发者，完全取决于用户如何启动你的应用程序，而应用程序则是由一系列的 activity 组成，用户会从 Home 或是其他应用程序中启动这些 activity。

### ✉ 在桌面点 icon 来启动应用程序主 activity

如果你的应用程序是独立运行的，它应该是用户在屏幕上触摸应用程序的 icon 或是任务选择器当中来启动（这个机制需要在 manifest.xml 中配置 intent filter，action 为 MAIN，category 为 LAUNCHER）。

### ✉ 在其它应用程序中启动你的 activity

这种方式就意味着你的 activity 是可重用的，也就是隐式意图。许多应用程序中的数据都需要共享给其它用户的，例如，email、文本消息、上传下载等。

还会有一种情况是这样，就是当用户选择了一个功能，正好有一个或多个 activity 符合用户的这种需求，就会向用户提供一个 activity 列表供其选择。举一个具体的例子，Gallery（图库），它能让用户查看并共享图片，这时用户选择了“共享”菜单，Android 系统会在 intent filter 中寻找适合该请求的 activity，如果有多个，就会以列表的形式展现给用户，供其选择。在这个例子中，intent filter 能找到 Email、Gmail、Messaging、Picasa 等。

当其它的 activity 启动了你的 activity 时，会根据需求给它们返回一个结果。

- **启动一个 activity 并需要返回一个结果**

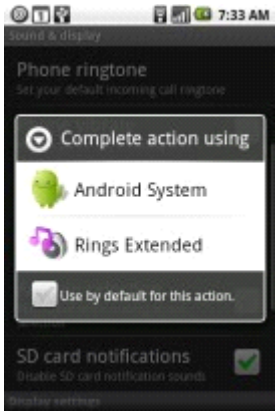
官方称这种方式为 closed loop，也就是说当启动一个 activity 之后，会返回一个结果回来。再拿上面那个例子来说，当用户完成上传或者发送的过程之后，会将图片信息返回给图库。这个例子中的上传过程所用到的 activity 就是由外部的 Gallery 启动的。（这种方式需使用 startActivityForResult()）

- **启动一个 activity 不需要返回结果**

官方称这种方式为 open-ended。举个例子，在 Email 中可定位一个住址，那么应用程序便会启动地图 activity 来定位地址，完成之后不会再给 Email 返回任何的结果；此时用户可以按 BACK 键回到 Email 中来。（这种方式需使用 startActivity()）

- **只从其他应用程序中启动 activity** -前面所说的例子，Gmail、消息、Picasa（在 Gallery 启动）都是 activity，它们均从桌面上的 icon 来启动的，与之形成对比的是，像裁剪图片和添加附件则不是在桌面启动的，因为这些都不是独立运行的。

实际上，并非所有的应用程序都有 icon 以供启动，它们都算作是一种小小的应用而已，因为它们使用并不频繁，而且其启动点都嵌在已有的应用程度当中。例如，Android 手机里面的打电话程序，其内部有个铃声设置功能，它存在于 Android 手机里面的设置（Setting）菜单里面，你也可以使用同样的 Intent 开发出一个定制的铃声设置应用，这样，在用户需要改变铃声时，会向其展示出两个铃声设置应用，一个是 Android 内置的，另一个就是你开发出来的。如下图：




铃声设置并不是经常使用，而且其定义的功能也很明确，所以也就不需要在桌面提供应用程序 icon 了。

- **不同的图标能够启动多个相同的应用程序 -**由于 Android 应用程序的运行代码均存在于.apk 文件中，因此就把这个文件看作是一个应用程序。我们甚至还可以让其内部存在两个主 activity，也就是两个应用程序启动入口点。

  Camera.apk（照相机）就是一个非常好的例子，它内部就含有两个独立的主 activity，Camera 和 Camcorder（摄像机）；它们均拥有自己的 icon 并且独立运行；在用户角度上来看，这就是两个应用程序。它们都共享使用一个镜头、在 Gallery 里面保存图片等。

  实现这样的功能其实很简单，只需将它们都关联到不同的任务上即可。（每个 activity 都其各自的任务，每个任务都有不同的亲缘性(affinity)。（这个例子的两个应用它们所在的两个包是"com.android.camera"和"com.android.videocamera"，有兴趣的可以深入研究）。

  联系人和拨号器也是同一个应用两个主 activity 的典型例子。

- **应用程序部件 -** 我们也可以将应用程序以部件的形式嵌进桌面上或是其它应用程序中并能它们持续更新。

## ● 允许你的 activity 添加到当前任务中

如果你的 activity 是在外部应用程序已经启动的话，那么也允许把它们添加到当前的任务中来（或者是已存在的任务——它有自身的 affinity），这样做的话能会使用户能够在其它任务和你的 activity 之间进行自由切换。但不包括你的 activity 仅有一个实例的情况。

对于这种行为，你的 activity 应该有一个 standard 或 singleTop 的启动模式，而不是 singleTask 或 singleInstance，这样，你的 activity 就会以多实例的模式来运行。

## ● 通知，应该能让用户更容易的返回上一个 activity

利用后台运行的服务能够给用户发出他们感兴趣的事件消息。下面举个例子，主要是以 Calendar 为主，这个例子含两部分，一个是以 Email 的形式发出即将来临的消息，另一个是当有新消息时就发出通知。

我们来模拟一个应用场景，当某个用户处于 activity A 中，这时获取到了 activity B 发出的通知，他打开了这个通知，也就是进入到了 activity B，当用户按下 BACK 键，他应该回退到 activity A。

下面的具体流程描述了当用户响应通知时，activity 栈是怎样工作的：

1. 首先，用户在 Calendar 中设置了一个开会通知，也就是创建了一个新的事件，并将已写进 Email 中的部分信息复制到该事务上。
2. 其它用户选择 桌面 > Gmail。
3. 他们打开 Gmail，接收到来自 Calendar 发出的一个开会通知。
4. 接着他们打开了那条通知，进入到 Calendar activity 中，并查看会议的简要说明。
5. 这时用户进入到其里面查看更为详细的内容（就是在第一步当中复制的信息）。
6. 当用户完成查看的操作时，按下了 BACK 键。他们回到了 Gmail 上，也就是打开通知的那个地方。

但上面的流程在默认情况下却不是这样的。

通常情况下主要有两种方式来发出通知：

- **通知专用的 activity -** 接着上面的应用场景来说，某用户接受了一条 Calendar 通知，用户首先进入 Gmail，查看详细的内容就要进入 Calendar activity 中；用户查看完后，按下 BACK 键必须要返回到 Gmail activity 上。实现此功能的前提是，Calendar activity 不能有与 Calendar 或其它 activity 同样的亲缘性(affinity)，也就是将该亲缘性设为空字符串即可。下来解释一下为什么这么做。

  那个 Calendar activity 拥有其默认的任务亲缘性（taskAffinity），当按下 BACK 键时（如上述第六步）回到了 Calendar，而不是 Gmail，这就是上面那样做的主要原因。特定应用程序中的所有 activity 都具有相同的任务亲缘性，因此 Calendar activity 的亲缘性匹配了 Calendar 的任务，这个任务是在第一步当中运行起来的,那第四步就表示打开 Calendar activity，又回到了 Calendar 的任务中，所以最后返回的还是 Calendar activity。但这不是我们想要的结果，只有将任务亲缘性设为空字符串才能解决这个问题。

- **选择已有的 activity，但只会展示其初始的状态 -**例如，用户在与 Gmail 交互的过程中进入到了其它 Activity，稍后再在回到 Gmail activity 时显示要显示其初始的状

态，而不是先前的状态。首先，你要确保通知触发器起作用时，intent 的标识是“clear top”，所以当 activity 启动时，它显示的是初始化之后的 activity，防止 Gmail 再次来到前台时还是用户上一次的看到的那些状态。（你需要在 intent 对象中设置 FLAG_ACTIVITY_CLEAR_TOP 标志）

另外还有其它方式去处理通知，比如让一个 activity 到前台并设置好其显示的指定数据，比如短信息。

一般情况下都是以新任务的形式来启动通知 activity 的（也就是说，在 intent 对象中设置 FLAG_ACTIVITY_NEW_TASK），这么做是避免这个任务成为另一个任务中的一部分。

- **请使用通知系统，而不要使用对话框来代替通知**

如果你的后台服务在某个时刻要通知给用户一个消息，那么请使用标准通知系统-，不要使用 dialog 或 toast 来通知。这两个会直接弹出来提示用户，再说通俗一些就是会突然打断用户的当前操作，这是一个极为不友好的用户体验。通知系统在这方面就做得就比较好，用户可以在适当的时候从屏幕上方拉下通知列表以便回应消息。

- **不要重新设置 BACK 键的功能，除非你有绝对的需要**

BACK 键的主要功能就是从当前的 activity 回退到上一个 activity，就是所谓的导航功能。大多数的 activity 都是一些比较通用的操作，诸如查看联系人列表，查看照片等，如果按 BACK 键，就直接返回先前调用它的 activity 就好了，不需其它的功能需求。

但要考虑一个问题，如果是应用程序非常得大，并需要细粒度的 BACK 键来加以控制该如何呢？例如 Google 浏览器，已经打开了几个 web 页面和地图页面，其中有一些关于地图数据的图层面板，我们需要在它们之间进行切换操作，也就是说在其内部通过 BACK 键来对其进行回退导航，而不是针对整体的 activity。

接上面的例子继续说，地图应用程序展现给用户不同的数据图层面板：有用来显示查询结果的定位信息、有显示朋友的位置、有显示街道方向的路线等。地图应用程序将这些图层面板保存在自身的历史记录里面，所以需要 BACK 键来进行回退导航。

同样地，浏览器使用浏览器窗体为用户展示多个的 web 页面，每个窗体都有它自身的历史导航，也就是桌面操作系统上的浏览器中的标签。例如，你在 Android 浏览器中的一个窗体上打开 Google 进行查询，并点击一条查询结果，那么这个结果页面会在当前窗体上打开，然后按 BACK 键就会回到了查询页面。总结一句话就是，当前窗体是从先前的窗体上跳转过来的，这时按 BACK 键就会回到了先前的窗体。如果用户一直按 BACK 键的话，最后就会离开浏览器所在的 activity，回到了桌面。

这些例子就是你有绝对的需要才要重新设置 BACK 键功能的理由。

# 四、菜单设计指南

菜单保存有一系列的隐藏的指令（用户操作），并通过按钮键或者手势访问。菜单命令执行操作和导航到您的应用程序或其他应用程序的其他部分提供了一种操作。菜单做为放置功能和导航之一有效的释放屏幕空间，用按钮或者在应用程序内容区域中的其他用户内容控件。

Android 提供可以用于提供功能或导航的两种类型菜单。两者之间，你能够为你的应用程序组织功能和导航。简述：

• 属性菜单包括应用全局通用的 activity 或开始相关的 activity 的首要功能。典型应用是用户按实际键盘“MENU”键。

• 关联菜单是当前选定项的次要功能。典型应用时长按一个项出现的菜单。就像在功能菜单，操作可以在当前窗口中运行或者其他窗口

除了最简单的应用程序外，其他都有菜单。系统自动放置菜单并提供用户能接受的标准方式。在这种意义下，通过熟悉的和可靠的方式为用户能接受所有应用程序的功能。所有菜单都悬浮在 activity 之上并比全屏小，因此应用程序的边缘仍然可见。这是一种视觉提醒，一旦菜单消失，它的使用是一个中介的操作。

下面开始介绍菜单。

## 菜单纵览

注意：你手机上的菜单样式和屏幕布局有可能与本文中的截图范例有所差异；此问题是由于不同版本的 Android 系统或不同型号的手机而造成的.

**1.选项菜单**

选项菜单上的多个指令(指令,即功能指令,译者注)是全局适用的,且可并行作用于当前进程 (原文为 Activity,类似 PC 上的进程,下同.译者注),也可启动另一个进程.却不适用(文本)内容里的被选中中项.
(程序员就容易理解这段,大意是选项菜单可针对运行中的程序启动某些功能)

在大部分的手机上,用户按下"MENU"键就会在屏幕下方显示选项菜单.而用户再次按下"MENU"键或"返回"键就会关闭选项菜单.实际上,想要关闭任何菜单都可统一使用"返回"键.(重复按下"MENU"键或者点击屏幕空白处也能实现同样效果.) 并需要注意在不同手机上的操作方法.

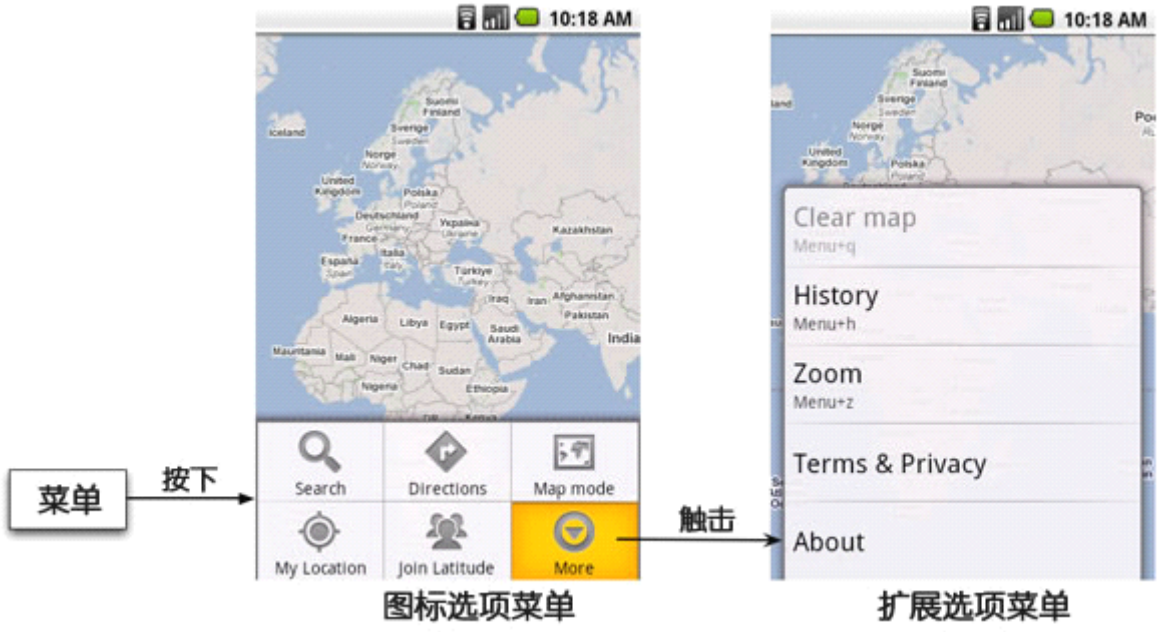
每个进程有属于它们的操作方式和选项菜单.一个程序的多个进程会有不同的选项菜单.

比如,在 Email 程序的邮件列表里面,选项菜单有邮件搜索,邮件排序,刷新列表,和更改邮箱设置等的功能.在 Email 程序的撰写模式下,选项菜单也有差异之处,例如多出了复制副本,添加附件和删除邮件这几个功能.

为了处理数目众多的菜单项,选项菜单通常以两步来逐级呈现.

•图标选项菜单 – 初次按下 MENU 键,屏幕底部会显示几个带图标且不可滚动的网格.(G1 手机上会显示 6 个典型按钮.)

•扩展选项菜单 – 如果这个进程的菜单项很多(超过 6 个),选项菜单的最后一个图标会标记"更多"–选中后会弹出一个包含多个菜单项目的列表,此列表有时还可以滚动.
(非常清晰的逻辑,很好地指导了菜单的设计技巧)




在 Android 的某些固件版本,用户可以长按(touch&hold,下同.译者注)"MENU"键来弹出快捷键–图标菜单上的文字,一会儿显示指令名称,一会儿显示快捷键(若有的话).

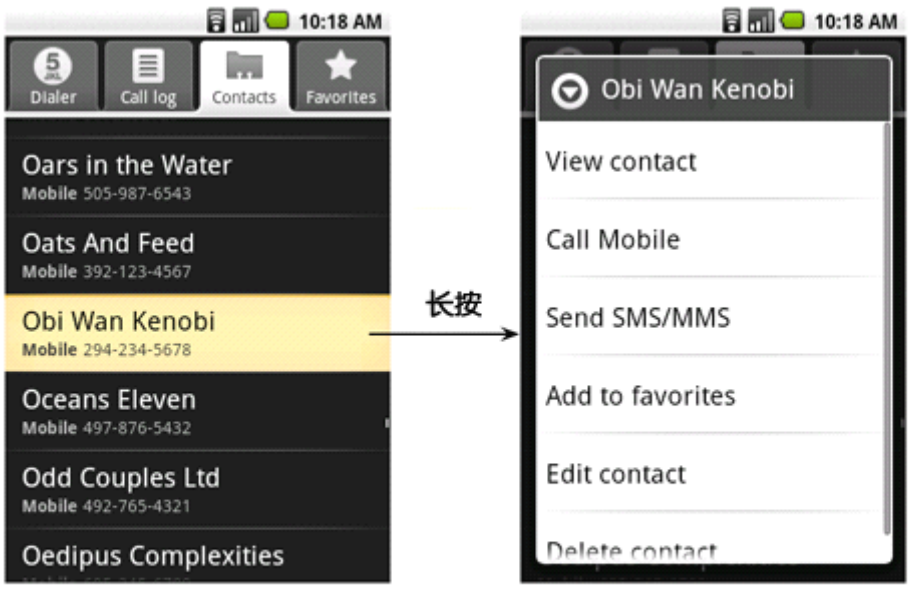
**2.关联菜单**

关联菜单类似于桌面操作系统(如Win7,译者注)的右键菜单.这便于在任何地方都能启动一些指令.

如下图所示,用户可以长按屏幕上的内容来打开关联菜单 (若有的话).关联菜单其实是被选中内容的一些可操作指令的列表.指令也可以成为当前进程的一部分,系统也能通过被选中内容去启动另一个进程进行操作.

例如,在邮件列表中,用户长按邮件信息会弹出一个含有阅读,归档,删除等指令的关联菜单 .

用户亦可长按屏幕某些位置来打开关联菜单.比如当用户准确地在主屏幕界面(Home screen)空白处长按,同样会显示一个关联菜单; 此处的图标菜单项也能点击.



列表视图 关联菜单

### 3.关联菜单是快捷方式

综上所述,如果用户在联系人"Obi Wan Kenobi"上长按,则会打开一个关联菜单,上面提供的指令能够执行一系列(与此联系人相关的)功能.

点击关联菜单会激活最直观的指令 - 例如 "查看联系人".我们建议最直观的指令同样可在关联菜单的首项(菜单的第一位,译者注)列出.如这个例子,直接点击 "Obi Wan Kenobi" 这个名字,和在关联菜单中点击"查看联系人" 所实现的功能是一样的.
(某些功能有多个入口或多种触发方式,能够适合不同熟练度的用户使用)

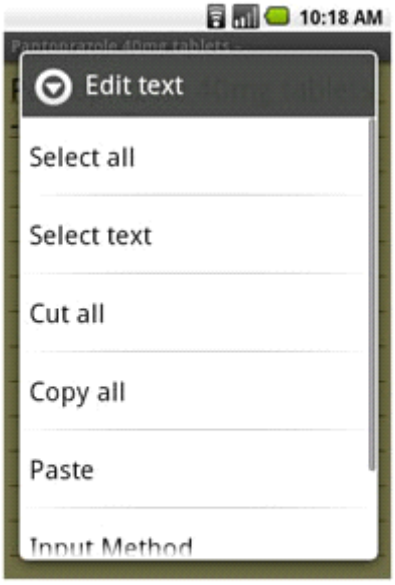
同样要注意 ,下文中的截图,左右两副图可实现的功能都是相同的.点击"查看联系人"后,左图关联菜单上的各种指令会分离成右图的选项菜单、图标按钮和常规菜单项.(大家对比下面两个图的功能项就明白了,译者注)

所以,使用关联菜单被认为是快速执行常规操作的一种捷径.关联菜单比起某些常显式的按钮或选项菜单,出现的机会更少.很多用户从未发现或者使用过关联菜单.因此,关联菜单上的每个指令也应该在界面上利用多种形式(比如图标啦,按钮啦之类,译者注)直观地显

示. 在下一节的说明中, 比如 "选择文字" 这种操作指令也许只在关联菜单出现. 同样, 比如浏览器之类的富互联网应用(原文为 rich applications, 译者注), 或某些包含联网的应用, 关联菜单上的一些些指令在其它地方也无法使用.
(慎用关联菜单这种隐性操作, 因为在交互设计里面, 操作分两种:显性与隐性;隐性操作无法明确被操作对象的关系和产生的结果, 所以大家就明白了为何手机上的左右软键在屏幕上面有软键功能名称)


**4.关联菜单上的文本指令**

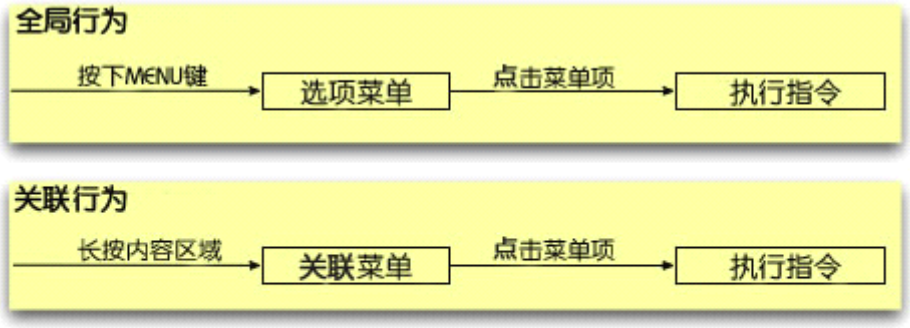
任何内容的文字链接与文字区域, 系统都统一提供一些操作选项, 并且适用于所有程序: 比如 "选择全部","选择文字","复制全部", 和 "添加至字典"这几个指令. 如果文字区域是可编辑的, 则会有另外的操作, 比如 "剪切全部" 与 "切换输入法". 又或者剪贴板内有文字的话, 则会显示"粘贴". 系统会在文字链接与文字区域中的关联菜单自动插入适当的菜单项, 就如下图所示.
(不同场景中, 菜单上的项目也有相应的变化, 以便于使用)




纯文字形式的关联菜单

**5.选项菜单与关联菜单的区别**

选项菜单适用于全局性的控制,而关联菜单适用于内容项.下面的图表中,用户通过操控菜单,然后点击菜单项来执行一个动作或打开一个对话框.




全局行为用选项菜单来执行
关联行为用关联菜单来执行

**6.屏幕中的固定指令**

某些指令能直接固定在屏幕上,典型例子是文字按钮,图形按钮,以及列表项.这种布局方式是目前为止最易于被用户发现的–用户无需按下按钮就能够直接看到指令(名称).这种增强可见性的方法需要衡量界面的空间和控件的占位大小,否则视觉设计上会显得凌乱.(合理安放菜单,需要细心思考菜单的:1.出现时机;2.出现形式;3.界面外观)

# 指南

正确选择菜单的呈现方式,保持菜单的一致性,是设计优秀程序的关键因素.后续指南的目的是,协助用户体验设计者及程序开发者更好地了解以上内容.

**分离全局指令的特定选项**

选项菜单内,适合安放全局性控制当前进程的指令,或者将指令固定显示在屏幕上; 关联菜单内,适合安放针对当前被选中项的指令.(无论如何,指令也可以作为进程的一部分来运行或启动另一个进程.)

你可以视功能作用来判定如何安放菜单: 如果(某个)指令对屏幕上的被选中内容(或特定位置)起效,就将指令安放到内容的关联菜单里.如果(某个)指令不是对特定的内容或位置起作用,就安放在选项菜单里.系统会以下列方式强制执行这些特定的指令.当你按下 MENU 键打开选项菜单,被选中内容则变为未选,因此也不起作用.

这里举一个特定选项的例子 - 用户在联系人列表里长按人名.关联菜单将包含典型的指令 "查看联系人","呼叫联系人",和 "编辑联系人".

**优先安放最频繁使用的操作**

由于屏幕高度有限,一些菜单会滚动.设计时着重安放重要的指令在首项,而无需滚动就可查看.至于选项菜单,在图标选项菜单上优先排列最频繁使用的操作,用户必须选择"更多"来查看其它指令.对于在相同位置安放类似指令也是非常有用的 - 例如,搜索图标可能一直排在选项菜单的首位,为多个进程提供搜索.

关联菜单上,最直观的指令应当优先排前,然后再按使用频度递减来排序指令,(关联菜单)底部则安放极少使用的指令.

**不要仅在关联菜单上安放指令**

如果用户无需使用关联菜单就能够完全使用你的程序,那么你设计得非常好!一般来说,倘若某些程序不得不使用关联菜单的话,那么你需要在其它位置也显示同样的指令.

在打开关联菜单之前,无法从视觉上识别其是否存在(反之,选项菜单起码有个 MUNE 键),所以关联菜单不是特别容易发现.因此,关联菜单的指令图标同样会显示在屏幕.例如,用户既可在联系人列表的人名上长按,从打开的关联菜单里点击拨打电话,亦可在"查看联系人细节"界面点击电话号码来拨出电话.

**关联菜单的首项指令应是最直观的**

如上所述, 关联菜单首项指令的功能最好与直接点击时所产生的功能一致.以上两个例子都是最直观的操作.
(我举个例子吧,例如打开链接,有 2 种方式:直接点击该链接,或者,长按链接然后弹出关联菜单的第一项指令"打开链接".两个方式实现的功能都是一致的)

**直接点选内容应执行最直观的操作**

在你的程序里,当用户点击任何可操作文本 (比如链接及列表项)或图像(比如照片图标),应当执行一个最符合用户期望的操作.
(符合期望值,的确是良好体验的基础)

一些基本操作范例:

- 点选一个图片则执行"查看图片"
- 点选一个多媒体的图标或文件名则执行 "播放"
- 点选一个链接则执行"打开链接"

•点选一个地点则执行"定位地点" (地图程序内)
需要注意的是,在不同的情景里选择同样的项目有可能调用不同的功能:
•在联系人程序里,点选一个联系人会执行"查看详细"
•在即时聊天程序里,点选一个联系人会执行"开始聊天"
•在邮件程序里,焦点处于收件人栏,在联络簿点选一个联系人则会执行"添加至收件人列表"

**关联菜单和被选中项须有所联系**

当用户长按某个项,弹出的关联菜单应包含被选中项的名称.因此,创建关联菜单的时候,必须包含被选中项的标题名称,这样用户才清晰地知道关联了什么.例如,若用户选择一个联系人叫"Joan of Arc",就把人名放在关联菜单的标题栏(利用 setHeaderTitle 函数).同样地,一个编辑联系人的指令应称作"编辑联系人",而不是"编辑".

**仅将最重要的指令固定在屏幕上**

把一些指令放入菜单里,就能让屏幕空出位置而显示更多内容.另一方面,在进程的内容区域固定显示一些指令(如按钮/图标),让指令操作更显眼及易用.

这里有几个重要理由来解释为何要在屏幕上固定某些指令:
•凸显指令,确保指令比较明显而不被用户忽视.
如:应用商店里的"购买"按钮.
•更快捷地调用重要指令,减少打开菜单的冗长乏味感.
如: 图片浏览程序里的下一张/上一张按钮或放大/缩小按钮.
•程序运行过程中一些需确认的指令.
如:图片程序里的保存/放弃按钮.
•用于对话框及向导界面.
如:确定/取消按钮.
•操作更直接.
如:在主屏幕界面拖拽一个程序图标至垃圾桶 .

**选项图标菜单多使用短名称**

如果图标选项菜单里某个文本标签过长,系统会将它拦腰截断.所以"Create Notification"这个指令名称会被截断得像"Create…ication"一样.你无法控制系统截断指令名称,所以, 最佳的方法是保持名称简洁.于 Android 的某些版本,图标被选中后,相关功能描述文字会如字幕般呈现,并可滚动阅读上面的文字.

**对话框不该有选项菜单**

当对话框出现后,按下 MENU 键不应有任何反应.这也适用于某些看起来像对话框的进程.小型的对话框比全屏的对话框更易于查看,上面一般有 0~3 个按钮,且不可滚动,有时还可能有单选框或复选框.

基本原理是,显示对话框后就不能有选项菜单.用户在(显示对话框的)过程中无法启动另一个全局任务 (而这也是对选项菜单的规定).

**如果进程无选项菜单,则不要显示任何信息**

当用户按下 MUNE 键后,若这里没有选项菜单,系统不会有任何反应.我们建议你不要执行任何指令(比如显示信息之类),这样整个程序的菜单会非常一致,继而带来更良好的用户体验.

**弱化或隐藏当前内容的不可用菜单项**

有时某个菜单项的功能无法执行 - 例如,在浏览器里, "前进"按钮必须在你曾按过"后退"按钮之后方可使用.我们建议:

•选项菜单内 - 禁用无效的菜单项,让其变灰.这种方法适用于图标选项菜单以及"更多"菜单.图标选项菜单突然由 6 项变成 5 项时会让人困惑,因此我们用同样的方式(变灰而不隐藏)来处理"更多"菜单.
(这里有点啰嗦,估计大意是:没用的菜单项就变灰,而不是突然隐藏它)

•关联菜单内 - 隐藏无效的菜单项.这样会使菜单更短,而用户也仅能看见可使用的选项(也能减少菜单滚动).