

iOS Human Interface Guidelines

朱晨 🍌🍌 www.xuexiao.me

译者按:此文是苹果在 2010. 11. 15 合并 iPad HIG 和 iPhone HIG 后的得来的。我用了大概两周的时间将其翻译出来,算是帮助中国的交互设计师和开发者了解苹果设计规范的小礼物。译时仓促,多少会有生涩和错误之处,如有看到,你可以发邮件或者到腾讯微博(totry.psy@gmail.com)告知我。以后有修订后的版本或者其他译作,也会在微博(t.qq.com/totry_zju)和博客(www.xuexiao.me)上给予通知。

目录

| | | |
|-----------------------------------|--------------------|----|
| 第 1 章 | 简介..... | 6 |
| 要旨概览: | 6 | |
| 伟大的 iOS 程序应遵守平台和交互设计原则 | 6 | |
| 伟大的程序起源于简明的定义 | 6 | |
| 伟大的用户体验来源于关注细节 | 7 | |
| 用户期待能在程序中使用苹果的技术 | 7 | |
| 所有的程序都需要进行部分定制 | 7 | |
| 第 2 章 | 平台特点..... | 8 |
| 无论尺寸如何, 屏幕都是最重要的 | 8 | |
| 屏幕的方向是会变的 | 8 | |
| 程序响应手势, 而非点击 | 9 | |
| 每次只在使用一个程序 | 10 | |
| 可以在“设置”中修改偏好 | 11 | |
| 屏幕上“帮助”的效果有限 | 11 | |
| 一个程序只有一个窗口 | 11 | |
| 有两类程序运行在 iOS 上 | 11 | |
| Safari 提供网页界面 | 12 | |
| 第 3 章 | 人机界面设计原则..... | 14 |
| 美 (Aesthetic Integrity) | 14 | |
| 一致性 | 14 | |
| 直接控制 | 15 | |
| 反馈 | 15 | |
| 暗喻 | 15 | |
| 用户控制 | 16 | |
| 第 4 章 | 程序设计策略..... | 17 |
| 明确程序定义 | 17 | |
| 1 列举所有你觉得用户会喜欢的功能点 | 17 | |
| 2 确定你的目标用户 | 18 | |
| 3 通过对目标用户的定义筛选功能点 | 18 | |
| 4 不要就此停下 | 18 | |
| 为设备而设计 | 19 | |
| 拥抱 iOS 界面规范 | 19 | |
| 确保程序在 iPad 和 iPhone 上通用 | 19 | |
| 重新考虑基于 web 的设计 | 20 | |
| 为任务量身定做界面 | 20 | |
| 原型和重述 | 23 | |
| 第 5 章 | 案例分析: 移植到 iOS..... | 24 |
| 从电脑上的 Mail 到 iPhone 上的 Mail | 24 | |
| 从电脑版 Keynote 到 iPad 版 | 25 | |
| 从 iPhone 版 Mail 到 iPad 版 | 28 | |
| 从电脑版 Safari 到 iPad 版 | 32 | |
| 第 6 章 | 用户体验指南..... | 38 |

| | |
|-------------------------------|----|
| 关注主任务..... | 38 |
| 提升用户关注内容的权重..... | 38 |
| 自上而下思考..... | 39 |
| 让用户有逻辑可循..... | 39 |
| 使用方法明显、易用..... | 39 |
| 使用以用户为中心的术语..... | 40 |
| 减少对用户输入的需求..... | 41 |
| 不要重视管理文件的操作..... | 41 |
| 容许协作和联通..... | 42 |
| 弱化设置..... | 42 |
| 品牌宣传要适当..... | 43 |
| 搜索要反应迅速、结果丰硕..... | 44 |
| 要在 App Store 提供精炼的描述..... | 45 |
| 精炼..... | 45 |
| 界面元素要一致..... | 45 |
| 考虑增加真实感..... | 46 |
| 用绝佳的图片取悦用户..... | 47 |
| 处理好改变方向..... | 48 |
| 让目标符合手指的尺寸..... | 50 |
| 使用微妙的动画表达..... | 50 |
| 恰当地支持手势..... | 51 |
| 只在必要的时候要求用户存储..... | 52 |
| 让模态化任务表现地暂时且简单..... | 52 |
| 立即启动..... | 53 |
| 随时准备停止..... | 54 |
| 不要自动退出..... | 54 |
| 有必要的话，展示许可证或者免责声明..... | 54 |
| 适用于 iPad：增强交互性（别只增加功能点）..... | 54 |
| 适用于 iPad：减少全屏转场..... | 55 |
| 适用于 iPad：抑制你的信息层级..... | 55 |
| 适用于 iPad：考虑将浮出层用于不同模态的任务..... | 58 |
| 适用于 iPad：考虑将浮出层用于不同模态的任务..... | 59 |
| 第 7 章 iOS 技术使用指南..... | 60 |
| 多任务..... | 60 |
| 打印..... | 61 |
| iAd 富媒体广告..... | 63 |
| 快速文件预览..... | 65 |
| 声音..... | 66 |
| 理解用户的期望..... | 66 |
| 定义声音的行为..... | 67 |
| 管理声音冲突..... | 71 |
| 处理远程媒体控制事件..... | 72 |
| VoiceOver 和附件..... | 72 |
| 编辑菜单..... | 73 |

| | |
|----------------------------|-----|
| 撤销和重做..... | 75 |
| 键盘和输入视图..... | 75 |
| 位置服务..... | 76 |
| 本地和推送提醒..... | 77 |
| 第 8 章 iOS 界面元素使用指南..... | 80 |
| 栏..... | 80 |
| 状态栏..... | 80 |
| 导航栏..... | 81 |
| 工具栏..... | 83 |
| Tab 栏..... | 84 |
| 内容视图 (Content Views) | 85 |
| 浮出层 (只限 iPad) | 85 |
| 分栏视图 (只限 iPad) | 87 |
| 表格视图..... | 89 |
| 文本视图..... | 95 |
| Web 视图..... | 96 |
| 警告框、操作列表和模态视图..... | 97 |
| 警告框..... | 97 |
| 操作列表..... | 100 |
| 模态视图..... | 102 |
| 控件..... | 104 |
| 活动指示器..... | 104 |
| 日期和时间拾取器..... | 104 |
| 详情展开按钮..... | 105 |
| 信息按钮..... | 106 |
| 标签..... | 106 |
| 网络活动指示器..... | 107 |
| 页码指示器..... | 107 |
| 拾取器..... | 107 |
| 进度指示器..... | 108 |
| 圆角矩形按钮..... | 109 |
| 范围栏 (Scope Bar) | 109 |
| 搜索栏..... | 110 |
| 分段控件..... | 111 |
| 滚动条 (Slider) | 111 |
| 切换器 (Switch) | 112 |
| 文本框 (Text Field) | 112 |
| 系统提供的按钮和图标..... | 113 |
| 工具栏和导航栏中使用的标准按钮..... | 113 |
| Tab 栏中使用的标准按钮..... | 116 |
| 表格等其他界面元素中使用的标准按钮..... | 117 |
| 第 9 章 定制图标和图片指南..... | 117 |
| 程序图标..... | 118 |
| 小图标..... | 120 |

| | |
|--------------------------|-----|
| 文档图标..... | 120 |
| 为 iPhone 制作文档图标..... | 121 |
| 为 iPad 制作文档图标..... | 122 |
| Web 快捷方式图标..... | 124 |
| 导航栏、工具栏和 tab 栏上用的图标..... | 125 |
| 登录图片..... | 126 |
| 为 Retina 屏幕设计画作的技巧..... | 128 |

第1章 简介



要旨概览：

伟大的 iOS 程序应遵守平台和交互设计原则

用户钟爱那些专门为移动设备设计的 iOS 程序。例如，用户非常希望程序能够与设备屏幕相衬，并且能够响应那些用户熟识的手势。虽然用户可能不知道人机交互设计原则，诸如“直接操控”“一致性”，但却能觉察的出遵守原则和违背原则的程序之间的差别。当你开始设计 iOS 程序时，一定要意识到是什么让 iOS 设备如此独特，并且学会使用交互设计原则，以便用户能爱上你的程序。

相关章节：[“平台特点”](#) 和 [“人机界面设计原则”](#)

伟大的程序起源于简明的定义

当有了设计程序的想法后，你必须精确地定义该程序包含哪些特性，目标用户是谁。决定之后，你需要确保该程序的外观和给人的感觉与设备和功能相匹配。

如果你想移植现有的程序到 iOS，需面临的挑战也大同小异。当你为 iOS 重新设计软件

是，从“Mail”“Keynote”等成功的程序上借鉴一些设计中的决策会很有帮助。

相关章节：[“程序设计策略”](#)和[“案例分析:移植到 iOS”](#)

伟大的用户体验来源于关注细节

当设计程序的各个方面时，从如何完成任务，到程序如何启动和关闭，再到如何使用按钮，“用户体验”必须是至高无上的。该指南会从宏观和微观上为你提供参考。

相关章节：[“用户体验指南”](#)和[“iOS 界面元素使用指南”](#)

用户期待能在程序中使用苹果的技术

iOS 提供了很多很棒的技术，诸如多任务、打印、VoiceOver。当用户使用 iOS 设备时，他们认为这些技术是默认就可用的，但这需要你在制作程序时提供支持。如果某种技术适用于你的程序，一定要遵循与其相关的指南。

相关章节：[“iOS 技术使用指南”](#)

所有的程序都需要进行部分定制

如果你的程序致力于提供严肃的、用于产生内容的功能，且只包含一些标准控件，那么你还提供一个在 App Store 和 Home Screens 上都看起来很棒的图标。当你的程序或多或少地包含一些定制皮肤时，你需要清晰地知道应包含哪些图标或图片，以及如何绘制它们。另外，如果你是为 Retina 液晶屏设计图片，就还需要学一些能让这一过程轻松点的技术。

相关章节：[“定制图标和图片指南”](#)

第2章 平台特点

使用 iOS 的设备拥有一些共性，这些特点会影响其程序的使用体验。与这些特性相适应的程序会更加成功，与设备一起为用户提供超凡的使用体验。

无论尺寸如何，屏幕都是最重要的

iOS 设备的屏幕是用户体验的核心。用户不仅在上面浏览优美的文字、图片和视频，还要和多点触摸屏进行交互（即使有时候用户甚至看不到屏幕）。

虽然不同的尺寸和分辨率对程序的用户体验有不同的影响，但有一些原则是通用的。

- 可点击元素的最小尺寸是 44×44 点
- 图片质量的影响显而易见
- 用户最关注的是内容

Table 2-1 Screen sizes of iOS-based devices

| Device | Portrait | Landscape |
|-------------------------------------|-------------------|-------------------|
| iPhone 4 | 640 x 960 pixels | 960 x 640 pixels |
| iPad | 768 x 1024 pixels | 1024 x 768 pixels |
| Other iPhone and iPod touch devices | 320 x 480 pixels | 480 x 320 pixels |

注：像素适用于谈论设备屏幕的尺寸，或在编辑素材的程序中定义图标的大小。点则用来描述一块区域在屏幕上会显示多大。

屏幕的方向是会变的

用户可能在任何时间由于多种原因旋转屏幕。例如，有时用户觉得当前的任务将屏幕竖起来更自然，有时用户觉得横过来放置能看到更多内容。无论原因怎样，用户希望旋转后屏幕依然重点显示此程序的主功能区。

用户经常在桌面上打开程序，所以他们期望程序以与桌面相同的方向打开。由于 iPhone 和 Ipad 在展示 “Home screen” 时的差异，这种期望也有所不同。

- 在 Iphone 和 Itouch 上，桌面只会以竖直的方向展示，Home 键在底部。这使得用户期待程序也以这个角度打开。
- 在 Ipad 上，Home screen 可以以任何角度展示。所以用户会期望程序以与桌面相同的方向打开。

程序响应手势，而非点击

用户使用特定的手指运动，我们称之为手势，来操作 iOS 设备的多点触摸界面。例如，轻敲可以激活按钮，拖动可以滚动长表单，两指分开可以放大图像。

多点触摸界面给用户一种与设备直连，直接操纵屏幕上物体的感觉。

由于内置程序对手势的使用遵从标准原则，所以用户在使用标准手势的时候更加舒适。使用内置程序的经历帮助用户学会了适用于大多数程序的手势词典。

Table 2-2 Gestures users make to interact with iOS-based devices

| Gesture | Action |
|---------|---|
| Tap | To press or select a control or item (analogous to a single mouse click). |
| Drag | To scroll or pan. |
| Flick | To scroll or pan quickly. |

| Gesture | Action |
|----------------|--|
| Swipe | In a table-view row, to reveal the Delete button. |
| Double tap | To zoom in and center a block of content or an image. To zoom out (if already zoomed in). |
| Pinch open | To zoom in. |
| Pinch close | To zoom out. |
| Touch and hold | In editable text, to display a magnified view for cursor positioning. |
| Shake | To initiate an undo or redo action. |

iPhone 和 iPad 都支持多点手势。虽然较大的屏幕也给了更多手指触摸的空间，但这并不意味着多点手势总是最佳选择。

每次只在使用一个程序

屏幕一次只能展示一个程序，当用户切换程序时，前一个程序会退出，其界面也随之消失。

在 iOS4 之前，退出程序意味着其在内存中的数据即刻被清空。而在 iOS4 之后，退出的程序会隐藏到后台，等待再次被调用的机会。这种特性被称为“多任务”，可以将程序保留在后台，直到被再次调用或终结。

大多数程序在转移到后台的时候，会被挂起。被挂起的程序会展示在“多任务选择器（multitasking UI）”中，这帮助用户快速找到近期使用的程序。多任务选择器会出现在屏幕底部，位于当前运行的程序界面或 Home screen 的下侧。



当用户重启挂起的程序时，它能够从退出时所在的那个点迅速恢复，无需重新渲染界面。

用户会偏爱在运行其他程序时把某些程序保持在后台。例如，用户在看电影时又突然想去查看日程表、邮件时，会希望能快速继续中断的播放。

可以在“Setting”中修改偏好

用户需在 iOS 的“Setting”中修改偏好。但若想修改偏好，他们必须退出当前的应用。

偏好往往在设定后很少会改变。虽然有些程序自己内置了“设置”，但并不意味着所有的应用都需要。

屏幕上“帮助”的效果有限

在体验一个程序前，移动设备用户不会去读一大段帮助，既没有时间也没这个欲望。而且，帮助内容会占用宝贵的空间来显示和存储。

基于 iOS 的设备和内置的应用都很符合直觉，易于使用。所以用户并不需要屏幕上展示帮助。这种经验驱使用户期待所有的 iOS 应用都是这些易用。

一个程序只有一个窗口

无论什么样的程序，都只有一个窗口。这个窗口用于放置程序的内容和功能。但是用户不会意识到这个窗口。在 iOS 设备中，用户觉得程序就是依次呈现的一屏又一屏图像(a collection of screens)。

你可以把一屏图像想象成一个离散的视觉状态或者模态。一个程序拥有的屏数或多或少，每一屏都是各种素材和控件的组合 (various combinations of views and controls)。

用户会觉得程序的屏与设备的屏没什么区别，但是程序的屏却可以远远超过设备屏幕的限制。例如，在 iPhone 的“联系人”中，联系人列表只展示了一屏，即使列表的实际长度足够填满好多屏。

有两类程序运行在 iOS 上

基于实现的方式，可将 iOS 上的软件分为两类：

- iOS 程序
- web 内容

iOS 程序是用 iOS SDK 编制的，可以直接运行与 iOS 设备上。就像内置的程序一样，这些 iOS 程序驻留在设备上，可以调用 iOS 设备的资源。用户将这些程序安装在设备上，

就像“Photos, Calender, Mail”等内置程序一样使用它们。

用户可以通过 iOS 设备访问网页时可以浏览 web 内容。Web 内容可以分为三类。

•• Web 应用

Web 应用是指那些能用于完成某种任务并且遵从某种展示标准的网页。它们的表现形式和 iOS 程序类似。

网页有时候会把 safari 工具栏隐藏，这样看起来更像是本地程序。使用 web clip 功能的网页还可以在桌面上建立图标。这样用户就能像运行程序一样打开这些 web 应用。

•• 优化过的网页

优化后网页会更适合 iOS 设备的显示和操作。而且，优化后的网页能为 iOS 显示设备进行适当的缩放，并且检测用户是通过什么样的设备浏览，以便对展示的内容进行调节。

•• 兼容的网页

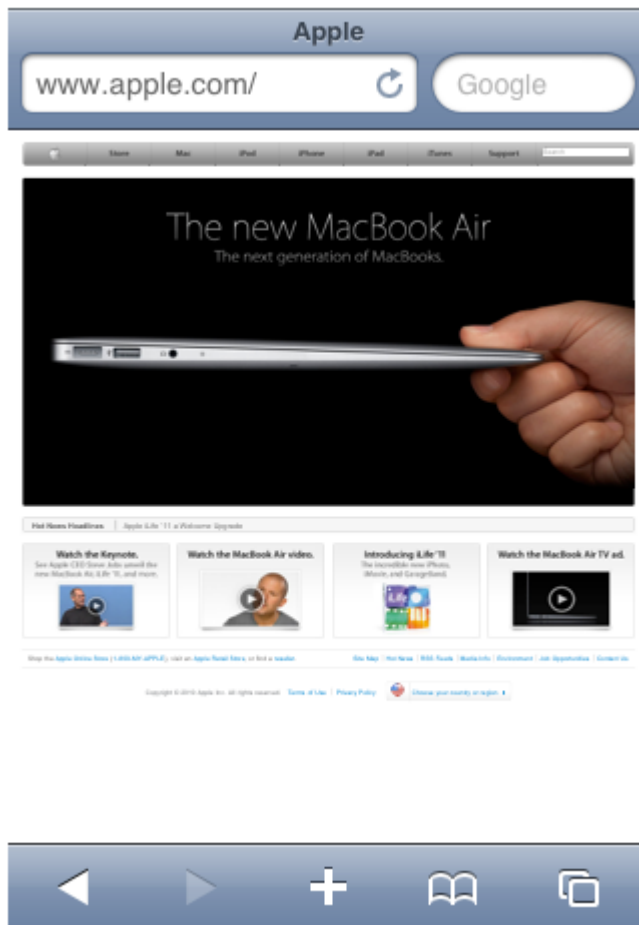
有些网页与 safari 和 iOS 设备的显示和操作兼容。兼容的网页不会为了使用 iOS 设备做更多的优化，但往往 safari 对这些网页的显示也是合适的。

有些应用会将部分区域划拨出来，用于展示 web 内容。这种程序看起来像是本地应用，但其实依赖于网络信息源。

Safari 提供网页界面

iOS 上的 Safari 提供了浏览网页的界面。虽然 iOS 上的 Safari 与电脑上的 Safari 有很多相似之处，但也有很多差异值得注意。

最明显的差异是用户无法改变可视范围（viewport，或译作观察口、视口、视埠）。在电脑上，当用户改变浏览器窗口大小时，可视范围的尺寸也会改变。而在 iOS 上，除非设备的方向旋转了，可视范围才会改变尺寸。iOS 用户可以通过缩放来改变 viewport 的显示范围。iPad 用户缩放网页的需求会比 iPhone 用户少。



iOS 上的 Safari 支持 cookies。Cookies 可以通过保存用户的环境（context）、偏好以及曾输入的数据让用户和网页内容的交互更加流畅。

iOS 上的 Safari 不支持 Flash 和 Java, 或者其他第三方插件。作为弥补, Safari 支持 HTML5 的<audio>和<video>标记, 用以播放音频和视频的流媒体。此外还有 JavaScript 和 CSS3 的变形, 透明和运动效果, 用以展示动态内容。

iOS 上的 Safari 将手势解释为对“如何展示内容”的操控, 而非对内容本身。轻敲可以触发 Safari 给网页发送一个 onclick 事件, 这与鼠标单击相似。但是很多鼠标动作没有相匹配的姿势, 比如说 hover。

iOS 上的 Safari 容许网络应用以全屏模式运行。网络应用可以通过“Web Clip”的图标来登录, 这样能隐藏 Safari 的界面控件, 看起来就更像是一个本地的应用。

第3章 人机界面设计原则

伟大的用户界面会遵从用户界面设计原则，这些原则基于用户思考和工作的方式，而非基于设备的能力。一个费解的、逻辑混乱的、不诱人的界面会让原本很伟大的程序变成一团浆糊。但是一个优美的、符合直觉的界面能够与程序的功能相辅相成，给用户留下良好的印象。

美 (Aesthetic Integrity)

美不是一个衡量程序好不好看，而是程序的外观和与其功能是否相衬。比如说，一个用来产生内容的程序（译者按：比如 word、ppt）往往把它的装饰性元素处理得很低调，并通过使用标准的控件和动作来突显任务。这可以帮助用户获得有关该程序目的和特性的信息。反过来讲，如果这个程序采用了某种鬼灵精怪的界面风格，用户就会陷入冲突的迷雾。

类似地，在那些娱乐性应用的界面上，用户期待界面能够非常漂亮，充满探索趣味。虽然用户不希望在游戏中完成严谨的任务，他们仍然期待游戏的外观可以与体验一致。

一致性

界面一致性能让人们沿用以往学会的知识和技能。保持一致性不是说要盲目地抄袭其他程序。相反，它应当利用那些用户已经熟识的标准和范式。

为了鉴别一个程序是否遵从一致性原则，可以思考如下问题：

- 该程序是否与 iOS 的标准一致？它是否正确地使用了系统提供的控件，外观和图标？它是否将程序与设备的特性有机地结合在一起？
- 该程序是否保持了充分的内部一致性？文案是否使用了统一的术语和样式？同一个图标是否始终代表一种含义？用户是否能预测他在不能地方进行同一种操作的结果？定制的 UI 组件的外观和行为在程序内部是否表现一致？
- 该程序是否与以往诸多版本保持一致？术语和意义是否保持一样？核心的概念没有发生本质变化？

直接控制

当用户直接控制屏幕上的物体，而非通过各种控件时，他们会更深地沉浸在任务中，也更清楚地理解他们行为的结果。iOS 用户很享受在多点触摸屏上直接控制的感觉。手势使得用户对屏幕上的物体拥有更强的操纵感。因为他们可以不再通过鼠标等中介设备控制物体。

例如，用户可以用手指姿势直接缩放一块内容区域，而非通过放大缩小按钮。在一个游戏中，玩家可以直接移动或操纵物体。再例如，游戏里会出现一只锁，用户可以旋转钥匙来打开它。

在 iOS 程序中，用户可以在如下场景体验直接控制

- 旋转或用其他方式移动设备，以影响屏幕上的物体。
- 使用手势操纵屏幕上的物体。
- 看到他们的动作有直接的、可见的结果。

反馈

反馈告知用户他们的行为有何结果，使用户确信程序正在运行中。人们操纵控件时期待即刻的反馈，也期待在较长的流程中能提供状态提示。

内置的程序会为用户的每一个动作提供可觉察的反馈。例如，当用户点击列表项时，该项的背景会变成高光。在那些会持续很多秒的长流程里，一个控件会展示已完成的进度，并在可能的时候提供解释信息。

顺滑的动画会为用户提供有意义的反馈，帮助用户了解动作的结果。例如，列表在添加新项时会向下滚动，帮助用户发现这个显著的变化。

声音同样能为用户提供有用的反馈，但是它不应是唯一的或主要的反馈方式。因为用户的使用场景可能会迫使他们关掉声音。

暗喻

当虚拟的物体和动作是真实世界中物体和动作的暗喻时，用户会立刻明白该如何使用这个程序。经典的例子是文件夹：在真实世界里，用户将东西放在文件夹里，所以他们立刻明白电脑上可以把文件放在文件夹里。

合适的暗喻应该即暗示了使用方法，又避免与它模仿的现实世界里的物体和动作面临同样的限制。例如，用户需要放海量的东西才能把文件夹塞满，而这在现实世界里是不可能的。

iOS 为暗喻提供了充足的空间，因为它支持丰富的动作和图片。用户与屏幕上的物体进行交互，就像在现实世界中操纵同样的物体一样。iOS 系统中的暗喻包括：

- 轻触 iPod 的播放按钮
- 在游戏中拖拉，轻拂或水平滑动物体
- 滑动切换开关
- 轻拂（Flicking over）一叠照片
- 旋转拾取器的拨轮，做出选择（Spinning picker wheels to make choices）

一般而言，暗喻在没有做过多引申时效果会比较好。例如，如果在操作系统里文件夹必须放在书柜里，它就用起来没那么灵光了。

用户控制

应该由用户出发和控制操作，而非程序。虽然程序可以建议某种流程、操作，也可以警示危险的结果，但抛开用户由程序来做决策未免太过荒诞。优秀的程序能够平衡用户的操作权并帮助用户避免犯错。

用户在控件和行为都很熟悉、可以预测结果的时候最有操控感。而且，当动作非常简单直白时，用户可以很容易地理解并记住它。

用户希望在进程开始执行前有足够的机会取消它。而且他们希望能在执行破坏性动作前有再次确认的机会。最后，用户希望能优雅地终止运行中的进程。

第4章 程序设计策略

所以伟大的程序都起源于一个伟大的想法。但这并不意味着将想法孕育成成功的程序是件轻松的事。这一章将介绍一些能用于精炼想法、回顾设计选择（design options）的策略，帮助你设计出人见人爱的程序。

明确程序定义

定义程序是指简明地描述程序要达成的主要目的和目标用户。

在开发早期就完成对程序的定义，能够帮你将一堆想法和属性凝聚一个成用户梦寐以求的产品。在开发过程中，可以以该定义为标准判断潜在的功能点和行为是否靠谱。创建程序定义可以通过以下步骤来完成：

1 列举所有你觉得用户会喜欢的功能点

可以使用头脑风暴。此时你应该找出所有与产品创意相关的任务。不用担心单子列的太长，待会儿还要再做精简。

例如，想象你最初的想法是做一个帮用户采购食物的程序。想象一下，脑袋里就会出现一系列用户感兴趣的任務。例如：

- 创建购物清单
- 获取食谱
- 比较价格
- 寻找商店位置
- 标注食谱
- 获得和使用优惠券
- 浏览烹饪教程
- 探索不同的烹饪方法
- 查找可替代的食材

2 确定你的目标用户

你的用户除了在使用移动设备，期待精致的图片，简洁的交互方式，出色的表现以外，还具备什么样的特性呢？以食材采购为例，你可以判断下列描述是否适合你的用户：

- 经常在家做饭 or 偏爱准备好的餐饮
- 喜欢使用优惠券 or 认为不值得花精力去弄优惠券
- 喜欢搜索奇特的食材 or 只眷恋基本的食物
- 严格遵守食谱 or 只把食谱当灵感参照
- 采购少量多次 or 一次买很多
- 喜欢一次为多个目标集中采购食材 or 只想在回家路上顺便买几样东西
- 笃信品牌 or 方便就好，牌子无所谓
- 每次采购都买类似的东西 or 按照菜谱买东西

考虑完这些问题，挑选三条最符合你目标用户的特性：喜欢实验新菜谱，采购很仓促，尽量节约（在不必花费太多精力时）。

3 通过对目标用户的定义筛选功能点

如果在确定了目标用户的特性后，功能点只剩下寥寥数条，你就得到了它：伟大的程序应该像激光一样准确聚焦在用户想完成的任务上。

比如说，想想在第一步里你为购物程序列数的大量潜在功能点。虽然这些功能点都很有用，但并不意味着每个功能点对用户同样有用。最重要的是，第二步中的目标用户对这些功能点的喜爱程度也不一样。

心里装着目标用户，再来检视功能点清单，最后能将程序聚焦在三个功能点上：创建列表、获得和使用打折券、获得菜谱。

现在可以定义你的程序了，精确地概括程序的功能以及目标用户。好的定义应该是这样的：

“一个帮助喜欢烹饪、主张节俭的用户创建购物清单的工具”

4 不要停

在开发过程中持续始终程序定义去判断功能点、控件和术语是否妥当。例如：

当考虑是否要添加一个功能点时，问问自己它对于你的程序以及目标用户来说是否足够核心。如果不是的话，把它放在一边，它可能是另一个程序的核心组成。例如，你已经确定你的用户喜欢发掘新菜谱，所以强调打包好的蛋糕和菜肴就有失妥当。

为设备而设计

你应该知道这个程序能做什么、目标用户是谁。现在，你要确定此程序能给人“为苹果 iOS 设备而设计”的感觉。这很关键，因为用户对行将安装在 iOS 设备上的程序有很高的期待。如果你的程序让人觉得是为其他设备、或者 web 而设计，用户就不会太珍视它。

拥抱 iOS 界面规范

iOS 用户已经很熟悉内置应用的外观和行为，所以他们期待这些下载来的程序能带来相似的体验。你不会想模仿内置程序的每一个细节，但理解它们所遵循的设计规范会很有帮助。首先要了解 iOS 设备以及运行于其上的程序所具有的特性（详见“[平台特点](#)”这一章）。然后，将以下几点铭记于心：

控件应该是可点击的。按钮、挑选器、滚动条等控件都用轮廓和亮度渐变，这都是欢迎用户点击的邀请。

程序的框架应该简明、易于导航。iOS 为浏览层级内容提供了导航栏，为展示不同组的内容或功能提供了 tab 页签。

反馈应该是微妙且清晰的。iOS 应用使用精确流畅的运动来反馈用户的操作。iOS 程序还可以使用进度条、活动指示器（activity indicator）来指示状态，使用警告给用户以提醒、呈现关键信息。

你可以在“[iOS 界面元素使用指南](#)”这一章掌握控件的使用方法。在为程序设计宏观的用户体验时，确保明白“[用户体验指南](#)”一章中的内容。

确保程序在 iPad 和 iPhone 上通用

如果你正计划为 iPad 和 iPhone 设计程序，要确保该设计方案可以适用两种设备。以下指南可以给你一些帮助：

为设备量身定做程序界面。大多数界面元素在两种设备上通用，但通常布局会有很大差异

为屏幕尺寸调整图片。用户期待在 iPad 上见到比 iPhone 上更加精致的图片。不建议仅仅将 iPhone 上的程序放大到 iPad 的屏幕上。

无论在哪种设备上使用，都要保住主功能。虽然一种版本会为任务提供比另一版更加深入或更具交互性的展示，但不要让觉得他们是在使用两个完全不同的程序。

超越“默认”。没有优化过的 iPhone 程序会在 iPad 上默认以兼容模式运行。虽然这种模式使得用户可以在 iPad 上使用现有的 iPhone 程序,但却没能给用户提供了他们期待的 iPad 体验。

重新考虑基于 web 的设计

如果你的程序移植自 web,那么需要确保你的程序能摆脱网页的感觉,给人 iOS 程序的体验。记住,人们可能会在 iOS 设备上使用 Safari 来浏览你的网页。

这里提供一些能帮助 web 开发者创建 iOS 程序的策略:

关注你的程序。网页经常给访客一堆任务或选项,让他们来挑选,但是这种体验并不适合 iOS 应用。iOS 用户希望程序能像宣称的那样有用,希望能立刻看到有用的内容。

确保你的程序帮助用户做事。用户也许会喜欢再网页中浏览内容,但更喜欢能使用程序完成一些事情。

为触摸而设计。不要尝试在 iOS 应用中复用网页设计模式。熟悉 iOS 的界面元素和模式,并用它们来展现你的内容。菜单、基于 hover 的交互、链接等 web 元素需要重新考虑。

让用户翻页 (scroll)。大多数网页在第一时间将重要的内容认真的展现出来 (Most websites take care to display the most important information “above the fold”)。因为让用户在顶部区域附近没找到想要的内容,就会离开。但在 iOS 设备上,翻页是很容易的,也是意料中。如果为了把所有内容挤在一屏里而缩小字体、压缩空间尺寸,最终可能内容都变得看不清,布局也没法用。

重置主页图标。网页经常在每个页面的顶部放置回主页的图标。iOS 程序不包括主页,所以不必多此一举了。另外, iOS 程序容许人们通过点击状态栏快速回到列表的顶部。如果你在屏幕顶部塞进一个主页图标,想按状态栏就没那么容易了。

为任务量身定做界面

顶级的 iOS 程序能够用清晰的意图和易用性去平衡界面设计。为了在程序中达成平衡,一定要在开发初期将设计考虑进去。因为对品牌化、原创性和市场推广的考虑经常会影响设计决策。始终保持对用户体验的关注是极大的挑战。

使用 iOS SDK 可以随意选择定制界面的程度。由于对定制的程度没有限制,你需要考虑这些定制的界面会怎样影响用户完成任务。当你把任务纳入考虑范围时,想一想用户执行这些任务的频率和环境。

例如,想象一个打电话的软件。这个界面没有使用键盘,而是呈现了一个漂亮、逼真的拨盘。这个拨盘制作精良,所以用户既非常喜爱,也立刻就知道如何去使用它。这个拨盘表现逼真,所以用户在做出拨号动作、听到与众不同的拨号音时会非常开心。但当需

经常拨打没在通讯录中保存的号码时，最初的喜悦很快会被沮丧替代。因为转盘拨号的效率太低了。对于一个帮助用户打电话的程序来讲，这个优美的界面是一个累赘。



另一方面，考虑一下泡泡水平线取样仪，这上面会呈现一个逼真的水平测量管。用户知道如何使用真实的仪器，所以也能立刻知道如何使用它。这个程序即便没有那个漂亮的气泡也能展示水平角度信息，但是这会让程序变得不符合直觉，难以理解。在这种情况下，定制的界面不仅向人们展示如何使用这个应用，也让任务变得简单。



当你在考虑定制的界面给任务带来的是帮助还是障碍时，请记住以下几点：

定制一定是要有据可循的。理想情况下，定制界面能帮助用户完成任务，增强体验。应该让用户的任务来引导界面设计。例如：

- 如果你的程序需要操纵大量的精确数据，用户会偏爱易懂、标准化的控件以及流畅精炼的导航。
- 如果你的程序用于浏览内容，用户就不喜欢比内容还抢眼的界面
- 如果你的程序是个游戏，提供即时的、有情节的体验，人们会期望进入一个充满漂亮图片、交互新颖的奇特世界。

尽可能少的给用户增加认知负担。用户喜欢了标准化控件的使用方法和行为，所以他们不必停下来思考该怎么使用它。当面对那些看起来、用起来不符合标准的控件时，用户之前的经验就失效了。除非你那极富个性的控件能让任务变得很容易，否则用户会讨厌被迫学习只能在此程序中使用的新技能。

保持内部一致性。你的界面越个性，在程序内保持这些控件外观和行为一致性就越重要。如果用户花时间学会使用这些不熟悉的新控件，他们希望这些经验能在整个程序里通用。

在控件和内容间保持差异。因为用户很熟悉标准控件，它们不和内容抢用户的注意。当你设计界面时，要确保它不会和用户关注的内容抢风头。例如，如果你的程序容许人们观看视频，你可能选择自己设计一套播放控件。但是，是否采用标准控件是次要的，更重要的是这些控件是否会在用户开始观看视频时渐隐，在用户轻触屏幕时重现。

在重新设计标准控件前要三思。如果你计划重新制作标准控件，要确保你的控件提供

与标准控件同样多的信息。例如，如果你设计的按钮不是用户印象中那种方方的样子，用户甚至可能看不出它能点。或者，如果你创建一个切换控件，却不能显示两极状态，用户可能不会意识到它是一个双态控件。

确保对定制界面元素进行充分的用户测试。在测试中，观察用户是否能预测控件的功能，使用它是否很简单。例如，如果你的控件尺寸小于 40×40 像素，用户点击它就有困难。或者，如果你的控件对水平滑动（swipe）和轻触（tap）的响应不同，要确保此控件的功能值得用户花费额外精力去留意这些差别。

原型和重述

在你调用庞大的开发资源去实现你的程序前，最好做一个用于用户测试的原型。即使只能让一些同事参与测试原型，你也能从他们对程序功能的新视角和体验中获益良多。

在设计初期，你可以使用纸面原型或者线框图来表现主要的信息展示区和控件（views and control），绘制画面间的切换流程。虽然你能从测试线框图中获得有用的反馈，但有采样不足带来的偏差也会误导测试员。这是因为让用户凭想象获得的体验会与使用细节完善的程序时不同。

如果你的原型能在设备上运行，那么获得的反馈会更加有价值。当用户能与你的程序交互时，他们能在程序中发现那些与预期不一样的地方，或者那些设计的太过复杂。

设计可信原型最简单的方法是选一个 Xcode 的模板，改成一个基本的程序，填充一些合适的内容用以占位。然后，在设备上安装它，以便用户能够尽可能真实的体验。你不需要提供大量的内容，或者让每个控件都有功能。只要测试员能通过点击屏幕上某区域跳转到下一个逻辑关联的页面，或者能完成主要的任务，就足以给用户反馈。想学 Xcode，详见“A Tour of Xcode”。

当基于 Xcode 模板开发你的原型时，你免费获得了很多功能，并能基于反馈对设计作出调整。只需要很短的时间，你就能在定稿并投入资源开发前迭代很多次原型。

第5章 案例分析：移植到 iOS

你很可能准备把一个现有的软件移植到 iOS 平台，而非创建全新的程序。从某些方面来说，从网页或其他平台移植应用要比从草图搭建全新应用更具挑战性。

如果由你负责这些项目，想一下人们会怎样使用 iOS 设备：

- 用户使用 iOS 设备的方式与使用桌上或膝上电脑非常不同，对用户体验的期待也完全不同。将软件从电脑移植到 iOS 设备的征途绝非一马平川。
- 用户经常在走动中或是在嘈杂的环境中使用 iOS 设备。你的工作就是创建爽快的使用体验，把用户拉进来，尽可能又快又简单地把他们想要的内容提供出来。
- 记住二八原则：通常，大多数用户（至少 80%）只使用程序中很少一部分功能。只有很少用户会使用全部功能（少于 20%）。iOS 程序很少需要提供只有资深用户才用得到的功能。

从电脑上的 Mail 到 iPhone 上的 Mail

Mail 是 Mac 上的一个曝光度很高，很好用，也很受好评的程序。也是一个容许用户方便地创建、接收、排序和存储邮件，跟踪事件，创建便签和邀请的强大程序。电脑上的 Mail 通过大量窗口来实现如此强大的功能。

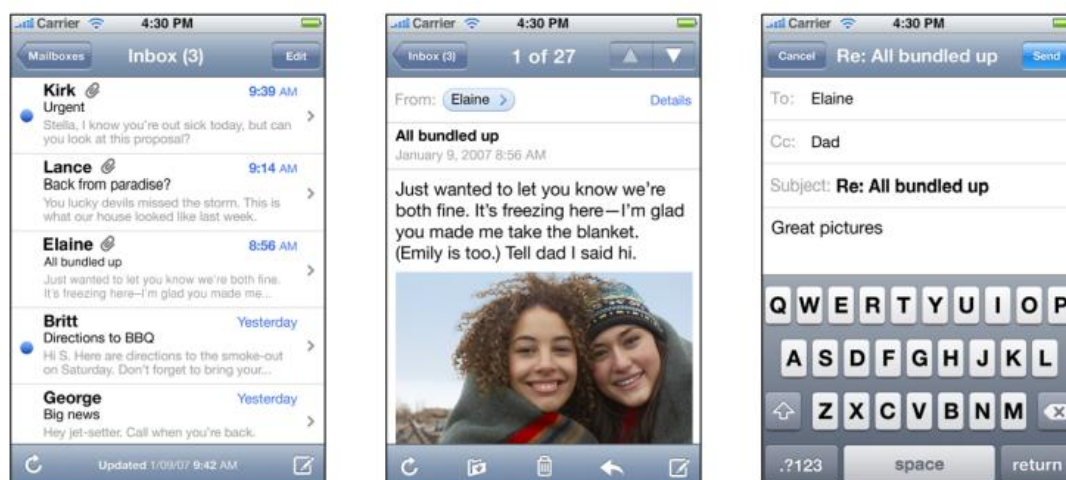
iPhone 版 Mail 只关注电脑版 Mail 的核心功能，帮助用户接收、创建、发送和管理邮件。iPhone 版 Mail 需要把功能浓缩到为如下情景量身定做的界面：

- 简练的外观，把用户关心的内容凸显到最前面、最中央。
- 为不同任务设计不同的视图。
- 信息结构符合直觉、缩放自如。
- 在需要时，提供强大的标记和管理工具。
- 为用户的动作提供微妙且富有表现力的动画作为反馈。

需要留意的是，iPhone 版 Mail 不比电脑版的 Mail 好用。想反，这是专为移动设备用户重新设计的 Mail。通过只关注一部分功能点，并把它们用简洁却诱人的界面展现出来，iPhone 版 Mail 满足了移动用户最核心的需求。

为了将 Mail 的体验迁移到移动环境中，iPhone 版 Mail 在界面上做了些创新：

直白的、高度聚焦的屏幕。每一屏展示一类邮件体验：账户列表、邮件列表、消息列表、消息视图、书写视图。在一屏内，用户通过滚动浏览全部内容。



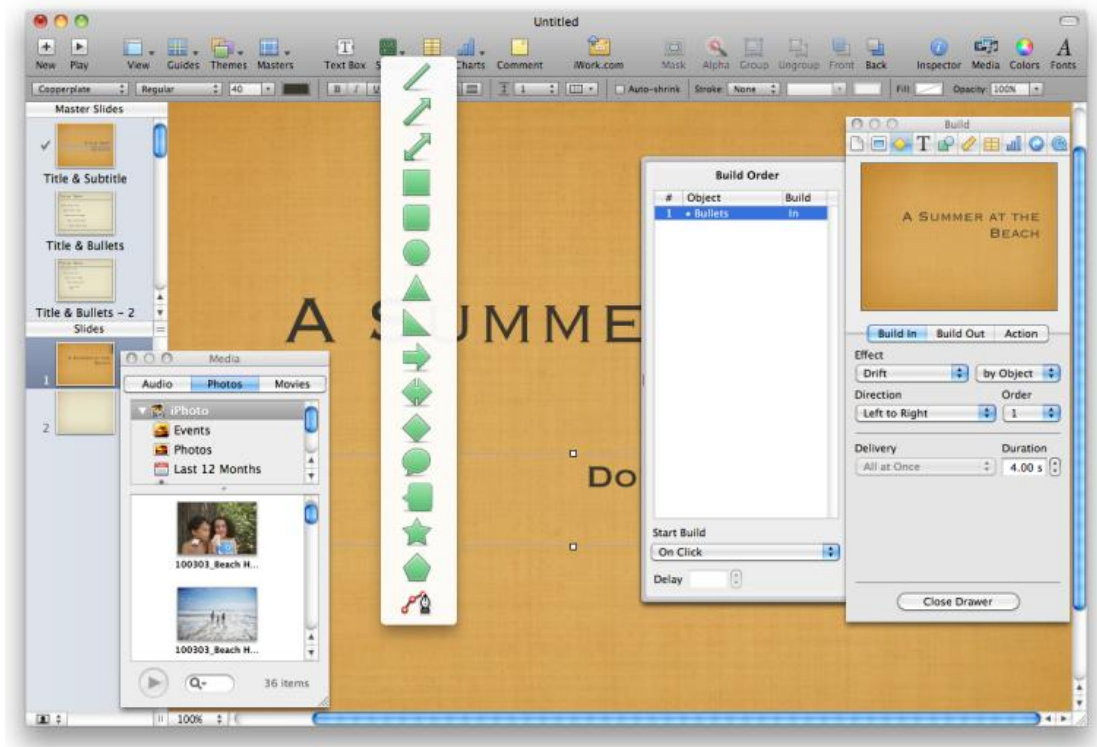
简单、可预测的导航。通过在屏幕上轻击，用户可以从宏观进入微观。每一屏上都会展示标题，告知用户所在的位置，以及一个返回按钮，方便用户找回浏览足迹。

简单、可触摸的控件，一有需要时就出现。因为书写新信息以及检查新邮件可能是用户在任何环境中都想进行的主要操作，iPhone 版 Mail 通过多点触摸屏幕来实现它。当人们浏览信息时，回复、移动、删除等可对信息进行的操作就会出现。



从电脑版 Keynote 到 iPad 版

Keynote 是用于创建世界顶级展示的应用，它非常强大、灵活。用户非常热爱能将易用与细节极其精致的木纹控件融为一体的 keynote。



iPad 版 Keynote 抓住了电脑版 Keynote 的核心功能，通过以下体验让 Keynote 在 iPad 上如鱼得水：

- 关注用户的内容
- 不阉割功能却降低了复杂度
- 提供了简便易用的快捷方式
- 优化了桌面上一些熟悉的特性
- 通过动画提供反馈和沟通

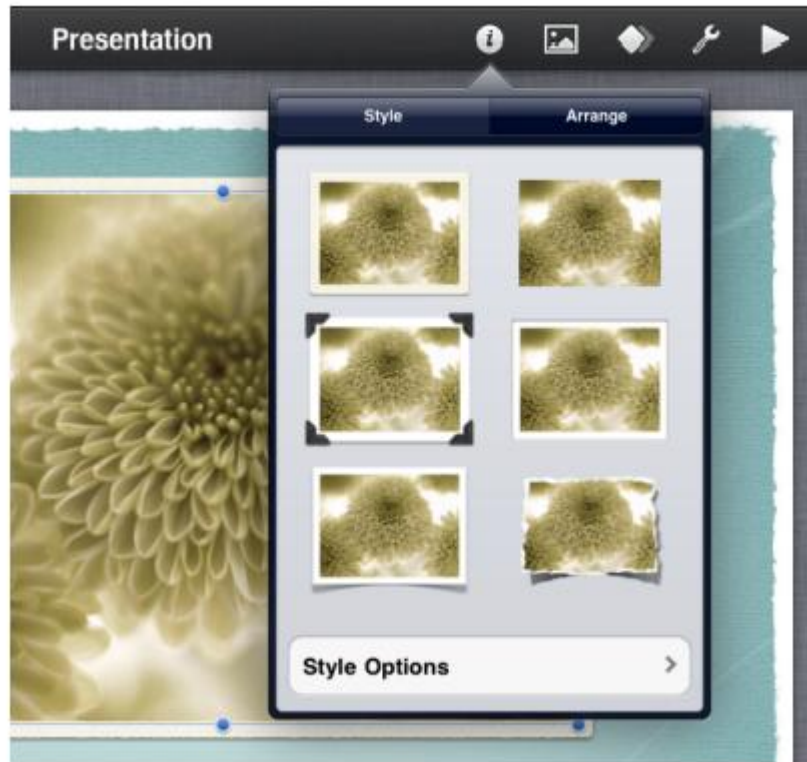
由于 Keynote 使用 iPad 规范提供了用户期待的功能，用户立刻理解如何在 iPad 上使用它。因为可以直接很简单自然地操控内容，新的用户也能很容易学会如何在 iPad 上使用 Keynote。

从电脑版向 iPad 版的移植基于无数事无巨细的修改和重新设计。最明显的改动有这些：

简洁的工具栏。 工具栏数的项目屈指可数。但是这给了用户获得所有创建内容所需工具的一致通道。



听从用户召唤的简洁检视窗口。 Keynote 的检视窗口自动包含与用户所选内容相关的工具和属性调节控件。通常，用户可以在检视窗口的第一屏内完成所有对选中对象的调整。如果要调整那些很少被改动的属性，用户可以进入检视窗口更深的层。



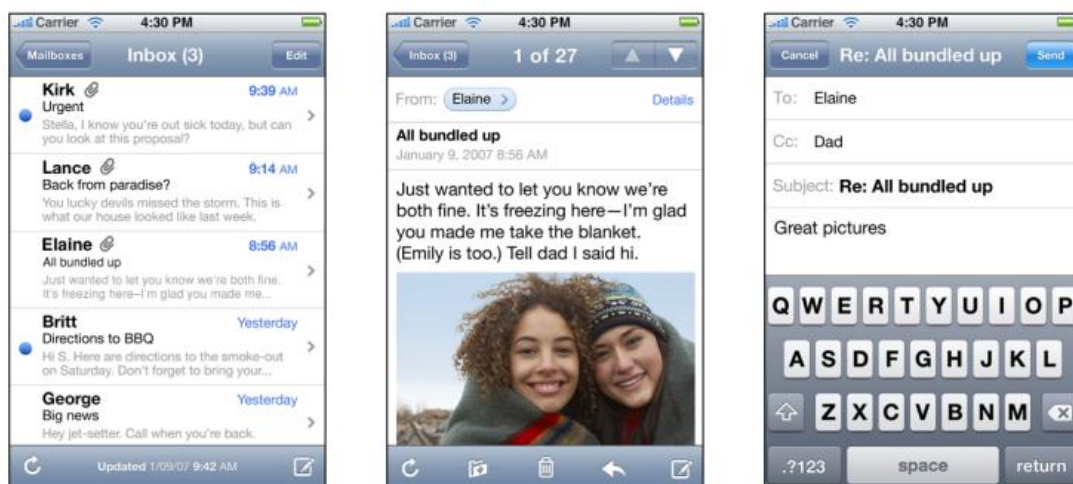
很多预置的样式模板。用户可以利用预置样式很方便地修改图示、表格等的外观和感觉。除了颜色主题外，每一套预置样式还包括表格标题、勾号标记等，用以协调整套主题的风格。



直接控制内容，添加有意义的动画。在 iPad 版 Keynote 上，用户可以拖动一张幻灯片到新位置，扭动物体以旋转它，轻触一张图片以选中它。动画反馈加强了直接控制的印象。例如，幻灯片在移动时会轻轻搏动，当把它放在新位置时周围的幻灯片四散开来为它腾出位置。

从 iPhone 版 Mail 到 iPad 版

Mail 是 iPhone 的首要内置程序之一。用户喜欢它在一屏屏上简洁地组织大量信息的方式。



iPad 版 Mail 实现类相同的核心功能。它有更多的屏幕空间为用户展示消息，有意义的触摸动作，以及时刻可见但又不过分招摇的管理编辑工具。简洁屏幕的视觉稳定性在一屏上呈现了用户需要的信息，尽可能地减少了环境改变。

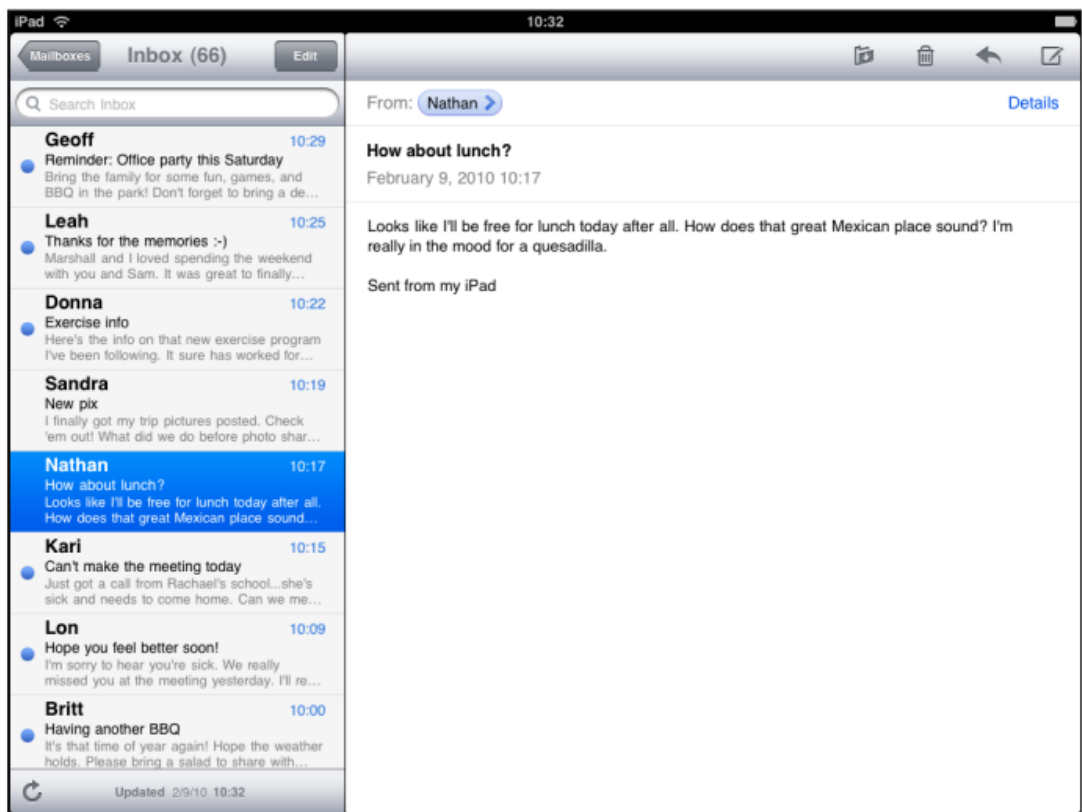
iPhone 版 Mail 和 iPad 版 mail 的差异反映出用户使用两种设备时的用户体验差异：

- iPhone 版 Mail 是为了帮助那些在排队或正走去开会的移动用户处理邮件
- iPad 版 Mail 足以让用户在运动中使用，但它所能提供的丰富体验也鼓励更深层次的使用

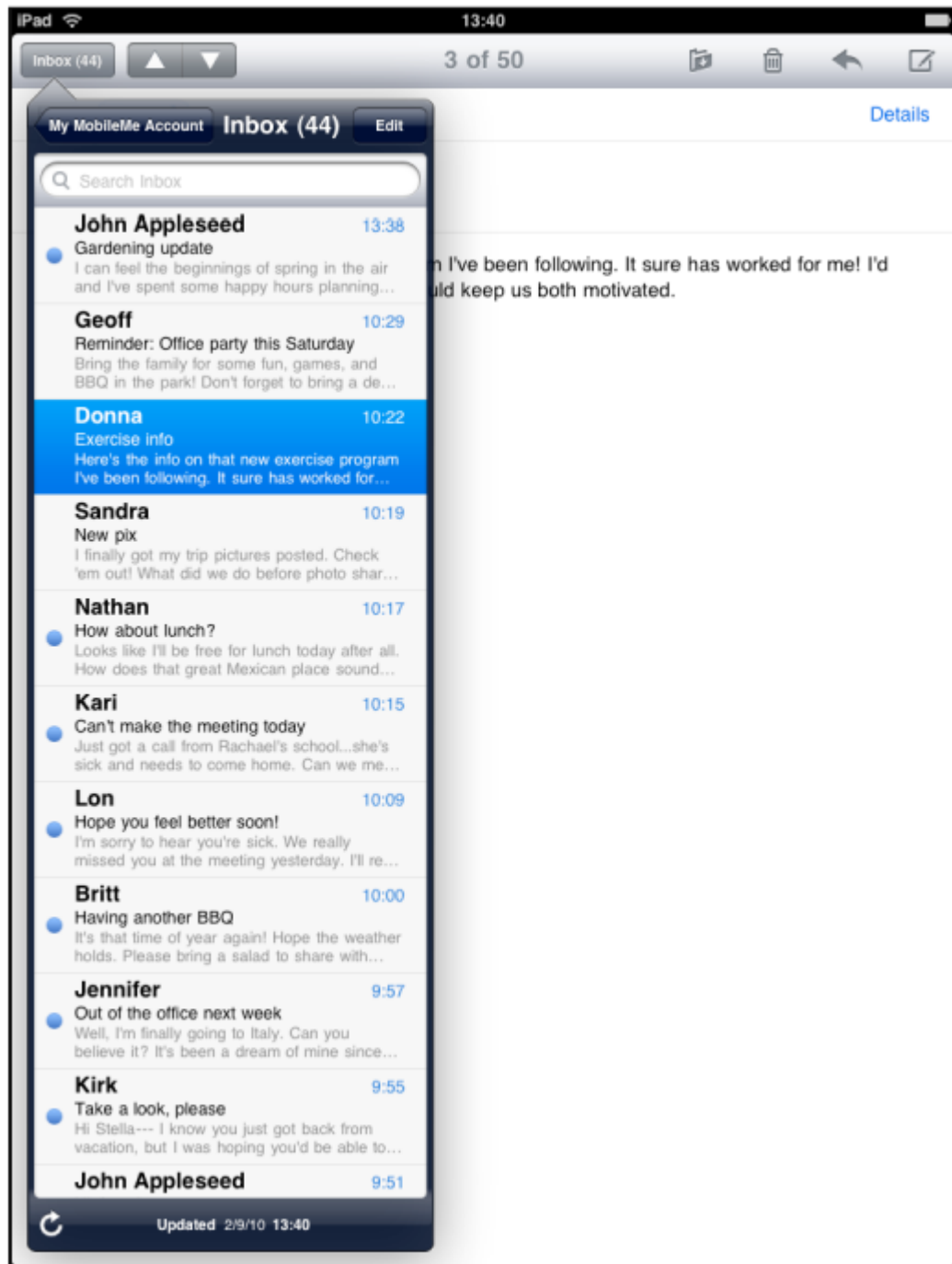
虽然 iPad 版 Mail 的用户体验是为该设备量身定做的，但它没有改变用户使用的核心功能，也没有改变某个功能的位置或是效果。iPhone 版 Mail 用户很容易发现 iPad 版工具栏上的项目以及收件箱结构，也因两者不变的本质而立刻懂得如何使用它。

为了增强移动邮件的体验，iPad 版 Mail 在这些方面优化了界面设计：

为设备的各种方向优化了支持。用户可以在四种方向中使用 iPad 应用。虽然横屏视图（译者按：老外习惯叫风景模式）与竖屏视图（译者按：老外习惯叫肖像模式）的布局略有差异，界面的焦点总维持在用户关注的功能和内容上。



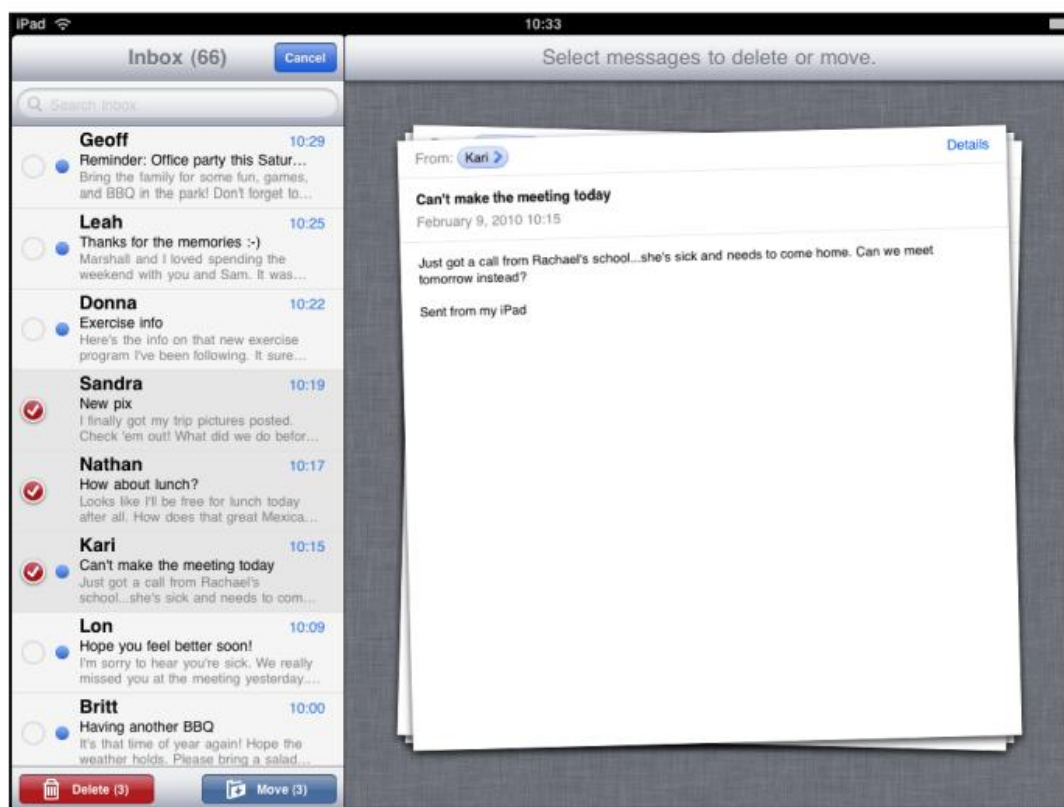
增加对消息内容的关注。iPad 版 Mail 在各个方向上都预留了主要空间用于呈现消息。这包括将工具栏移动到消息的顶部,以增加展示消息的垂直空间。有了这些额外的空间,用户不必翻页就能阅读更多信息。当用户想用竖屏模式浏览信息时,他们仍然能看到当前消息的绝大部分。



扁平化的层级。iPad 版 Mail 通过将所有高于邮件的层级划分到独立的界面元素有效地将邮件层级扁平化了（账户>邮箱>邮件列表>邮件）。在横屏模式下，这块区域就是分栏的左半侧。在竖屏模式下，这个元素是一个弹出层。

极大地减少了全屏转场。因为大多数层级可以通过独立的那部分元素来浏览，用户可以在一屏上获得大多数他想要的的信息。当用户沿层级下行时，只是独立的这部分页面在转场，而非整个屏幕。

具有真实感的邮件。当用户为删除一封邮件而标记它时，这封邮件会在右侧展示，就像一张真的纸。当勾选了更多邮件后，这些邮件会堆叠成一摞纸，摆放的略有参差。



从电脑版 Safari 到 iPad 版

iOS 版 Safari 为在 iOS 设备上浏览网页提供了卓越的体验。用户钟爱平滑的文字和细腻的图片，以及通过旋转设备、轻敲平铺以及分开手指去改变观看方式。

标准的网站在 iOS 设备上可以很好显示。能检测浏览设备、没有 flash 等插件的网站无论在 iPhone 还是 iPad 上都看起来很棒，但也可能会稍被调整。

另外，最成功的网站应该这样：

- 为 iOS 设备专门设计合适的显示范围，页宽与设备宽度相匹配。
- 避免 CSS 绝对定位，以便当用户缩放页面时内容不会错位。
- 使用基于触摸的界面，而非依赖基于指针的交互。

有时，其他的修正也是恰当的。例如：web 程序总是将显示范围设置的比较合适，并隐藏 Safari 的界面。欲知更多，且看“控制显示区域（Configuring the Viewport）”和“调整网络程序（Configuring Web Applications）”

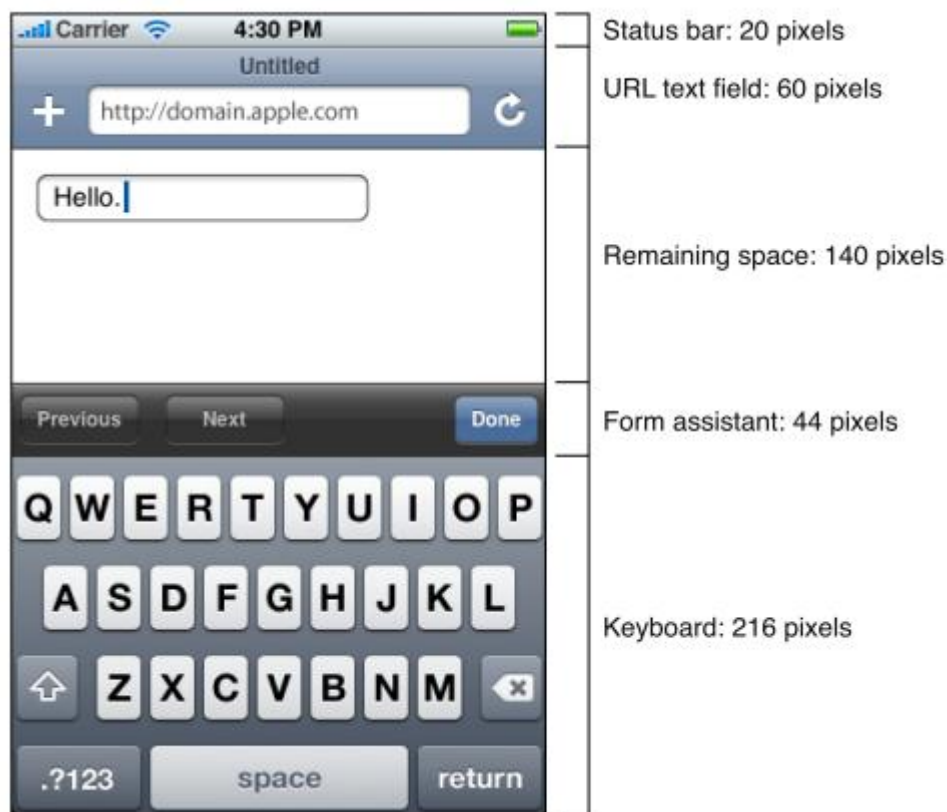
iOS 版 Safari 做了如下优化：

使用定制 CSS 来提供可适应的界面。不同的界面元素适合不同的环境。例如，苹果的网站包括一个页面，展示用户可以观看的视频。当在电脑上用 Safari 浏览时，用户可以点击上一页、下一页或者页码来查看其他片段。



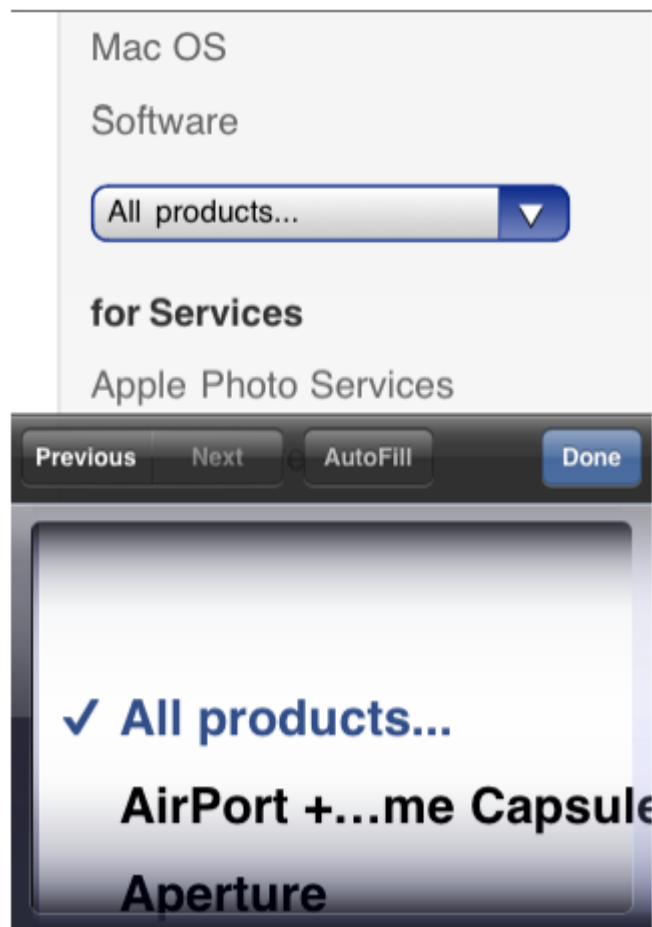
而在 iPhone 上浏览时，这些控件被更易用的按钮替代，按钮上的符号与内置控件的样式相呼应。

将 Safari 与 iOS 键盘相匹配。当键盘以及表单辅助按钮可见时，Safari 在 URL 地址栏下、键盘和表单辅助按钮以上的空间里展示网页。

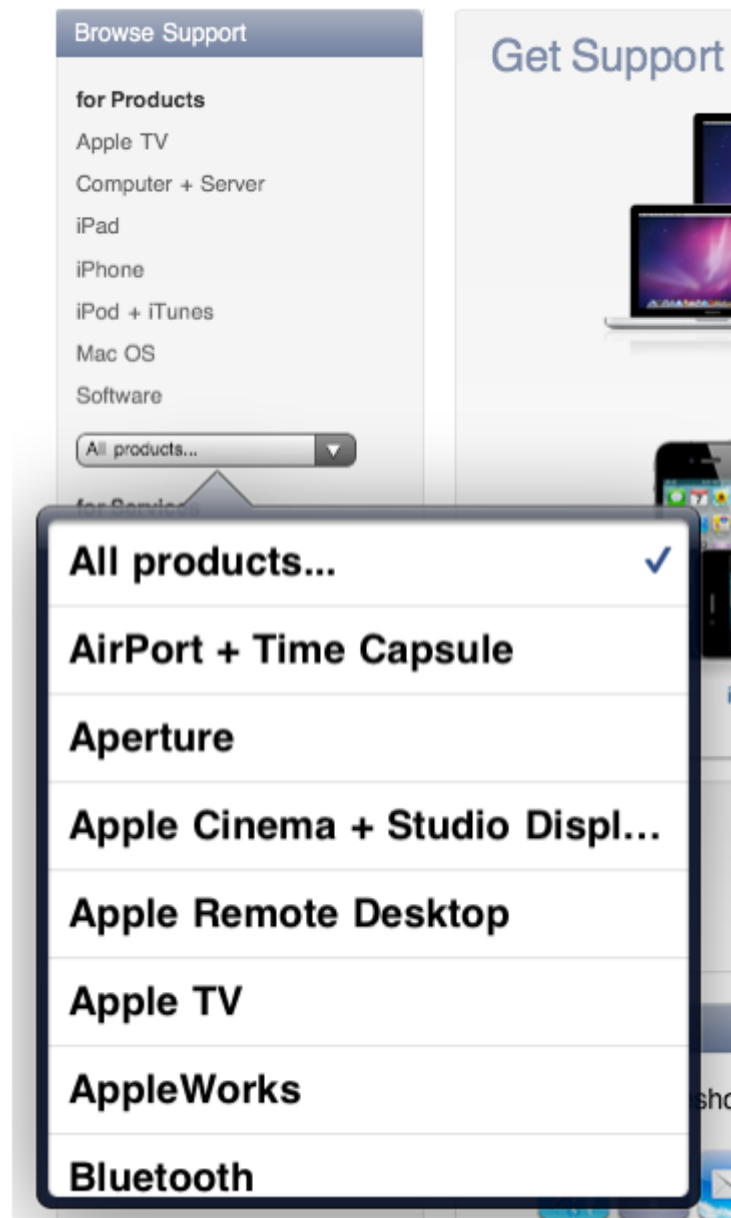


当没有键盘和表单辅助按钮时,会多出 216 像素的垂直空间来展示网页。在横屏模式下,值有两处不同,键盘高 162 像素,表单辅助按钮高 32 像素。

设计了新的下拉菜单控件。在电脑版 Safari 上,下拉菜单的表现方式与 Mac 系统中的应用程序一致,菜单打开,展示所有的项目,甚至在必要的时候超过窗口边缘。iOS 版 Safari,下拉菜单采用新的样式,提供更好的用户体验。例如,在 iPhone 上,下拉菜单显示在一个拾取器里。



在 Ipad 上，下拉菜单显示在一个浮出层里。



使用列表展示 iPhone 网络应用中的数据。iOS 用户习惯了内置程序的列表，所以当在 web 程序中看到列表时，他们更有可能将 web 内容视作程序。在 iPhone 上，表单展示在圆角矩形里，或者边贴边。每种样式都有不同的韵律。

Edge-to-edge list



320 pixels

356 pixels

Rounded rectangle list



320 pixels

356 pixels

第6章 用户体验指南

iOS 设备的用户体验重点在用精简的交互控制用户关心的内容。本章的指南适用于所有 iOS 设备的应用。

关注主任务

当程序使用聚焦于它的主任务时，使用起来一定神清气爽。你对程序的定义能帮助你将程序的功能聚焦在主任务上（见“[明确程序定义](#)”）。为保持专注，你需要明确每一屏上最重要的是什么。

分析每一屏需要什么。当你确定在每一屏上呈现什么时，始终问问自己，这是否是当前用户需要的关键信息或功能。如果答案是否定的，想想这些信息或功能在另一个环境中是否关键，又或者这些东西并没有那么重要。

例如：iPhone 日历关注日子以及发生于某日的事件。用户可以用明确标注的按钮高亮今天，选择浏览方式，以及添加事件。

提升用户关注内容的权重

在游戏中，用户关注感官体验；他并不希望管理、消费或者创造内容。如果你在开发游戏，可以通过提供惬意的故事、漂亮的图片和反馈及时的游戏操控来提升体验。

如果你开发的不是游戏，可以通过为用户感兴趣的信息设计微妙的框架结构来帮助用户关注内容。这里是一些你可以做的事：

减小控件的数量和显著性，降低它们在界面中的权重。Photos 通过在透明的工具栏上放置几个不突兀的控件来达成这一目的。

微妙地设计控件，使它和程序的图片风格一致起来。通过这种方式，控件既容易被发现、理解，也不会太突兀。

在用户不再与使用控件一段时间后，让它们渐隐消失。当用户敲击屏幕时再出现。有时你会想让程序的其他界面部分也渐隐。这对于需要营造卷入感的程序由其合适，因为这空出更多的屏幕空间展示用户想看的内容。例如，Photos 在用户不使用控件一段时间后就将按钮和工具栏隐去，以便让用户更关注内容。当用户想对照片进行操作时，随处轻轻一敲，控件就又出现了。

自上而下思考

屏幕顶部的内容最易被用户看到，因为用户是以如下方式使用 iOS 设备的：

- 用非利手（一般为左手）拿着，或者放在一个平面上，使用利手的手指做手势
- 用一只手拿着，同时用该手的拇指做手势
- 双手端着，用两只手的拇指做手势

把最常用的信息放在顶部附近。这里更容易被看到，也容易触摸。当用户自上而下浏览屏幕时，呈现的信息应从概括向具体渐进，从高级向低级渐进。（from general to specific and from high level to low level）

例如，在游戏中，最常用动作被放置在屏幕的上半部分。下半部分的屏幕用于呈现辅助性信息和控件。

让用户有逻辑可循

让你展现信息的流程富有逻辑，易于被用户预测。另外，一定要提供标记，比如返回按钮，让用户知道他们在哪，如何追溯来时的足迹。

大多数情况，只提供单一的路径。如果很多情景都可能通往某一屏的路径，可以考虑使用能在不同环境出现的模态化的展现方式。（可能是指的弹出层这类元素吧，model view）

使用方法明显、易用

努力让用户一眼就看出你的程序怎么用，因为你不能假设用户都有时间来思考程序是怎么工作的。

第一时间呈现程序的主功能。可以使用这样的方法：

- 尽量减少控件，让用户不必思考该如何选择
- 一致且和实地使用标准控件和手势，以便程序的行为符合用户期望
- 控件名称清晰易懂，让用户明确知道自己在干什么

与内置程序的使用方法范式保持一致。用户知道如何在各层级的屏幕间导航，编辑列表内容，使用 tab 切换程序模式。可以通过不断增强这种体验来让用户简单地使用你的程序。

在内置的 Stopwatch 程序中，用户一眼就能到那个按钮触发秒表，那个按钮用于记录每个人的时间（capture lap times）。



使用以用户为中心的术语

所有用于与用户沟通的文案，都需要保证你的用户能够理解。特别的，避免使用行业术语。多了解你的目标用户，以判断所使用的文案是否适用于他们。

这个 wi-fi 设定使用朴素的语言解释 iOS 如何反馈用户的偏好设置。



减少对用户输入的需求

无论是触摸控件还是使用键盘，输入信息劳神费力。如果你的程序在提供有用信息前要求用户输入一大堆信息，用户就会对你的程序失去兴趣。

平衡用户的输入与你为用户提供的信息。换句话说，为用户输入的每条信息提供尽可能多且有用的信息和功能作为回报。这能让用户觉得他们在向目标前进，而非被你的程序拖后腿。

简化用户的输入方式。例如，可以使用表格或者拾取器，而非文本框。因为选择远比输入来的简单。

在合适的时候，从 iOS 获取信息。用户在设备上存了很多信息。如果你能从设备中很轻易的找到这些信息，就别麻烦用户了。比如说通讯录、日程表等。

不要重视管理文件的操作

虽然 iOS 可以帮助用户创建和管理文件，但这不意味着用户应当清楚地认识 iOS 设备上的文档系统。

iOS 上没有和 Mac OS X Finder 相似的程序。不要要求用户像使用电脑那样管理文件。

不要强迫用户面对任何鼓励他们思考文件元数据的东西。例如：

- 一个展示文件层级的打开或保存对话框
- 文件权限状态的信息

相反，iOS 设备的文档管理程序应该鼓励用户将“内容”视作可供管理的物体。

如果你的程序容易用户创建和编辑文档，就应该提供让用户拿来打开已有文件或创建新文件的文档拾取器（document picker）。它应该：

高度图形化的。用户应该能通过图形化展示的文件迅速找到想要的文件。（比如 iBook 的书架）

容许人们用尽可能少的手势就完成任务。例如，用户在已存在的一堆文档中水平翻页，然后轻触一下想要的那个。

包括一个新建文档功能。不要让用户到处找新建文档的按钮，可以用一个空的占位符图片作为新建文档的按钮。

也可以在你的程序中使用 Quick Look Preview 功能预览文档，即使你的应用无法打开这类文档。欲知更多，详见 [“快速文件预览”](#)

容许协作和联通

iOS 设备是私人物品，但也鼓励与他人进行协作和分享。

在合适的时候，简化与别人的交互，分享诸如位置、设置、最高分等。人们普遍希望分享对他们重要的信息。

大多数程序可以通过容许人们突破程序壁垒，与其他工具共享数据的方式加分。例如，iOS 程序可以作为电脑程序的补充。或者，iPad 程序可能会容许用户与该程序的 iPhone 版通讯。

对于 iPad，要设想多人共用一台 iPad 的情景。例如，两个人可能在屏幕的两边玩对战游戏。或者模拟乐队的程序容许多个人在同一设备上一起玩不同的乐器。

弱化设置

如果可以的话，避免在程序中加入设置模块。设置包含偏爱的程序行为和信息，这些设定很少会改动。用户必须先退出程序，才能设置程序偏好。这绝非你想要的。

当你把程序设计的符合用户期望时，设置的重要性就降低了。如果你需要用户的信息，去问 iOS，而非拷问用户。如果你确信设置不可或缺，请参阅“iOS Application

Programming Guide 中”的“The Settings Bundle”。

让用户在程序中用配置选项（configuration options）来设置偏爱的程序行为。结构选项可以让程序在运行中动态响应用户的设置，用户不想离开程序再来设置它。

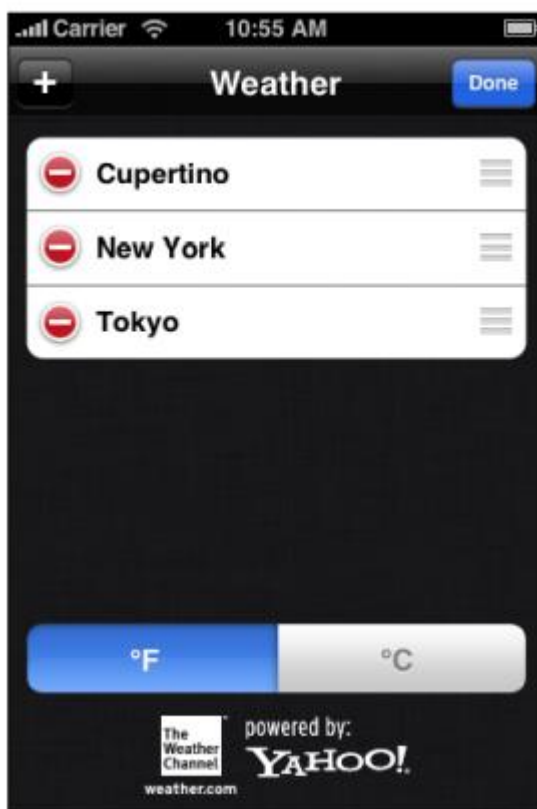
在主界面上提供配置选项，或者在屏幕的背面。选项对任务的重要度和使用频率将决定把控件放在哪里更合适。

■ 在主界面中，放置与主任务相关或者用户要经常更改的选项。

例如，iPad Calendar 允许用户以日、周、年为单位浏览日程表。这些配置选项被放置在主界面上，因为以不同方式查看日程是用户的主要任务，而且会经常切换。

游戏等注重即时体验的程序也应提供配置选项。因为用户经常会在各种体验间往复切换。

■ 在 iPhone app 中，可以将用户很少改变的选项放在屏幕背侧。例如，Weather 的主任务时展示某城市当前的情况和 6 天的预报。虽然选择温度是摄氏还是华氏很重要，但已经改变就不会再经常变动。所以可以把温度单位的设置放在屏幕背后，这样可以方便的拿到，同时也不碍眼。



品牌宣传要适当

精心准备品牌的颜色或图片。当品牌设计的非常微妙、易懂时是最有效的。用户用你

的程序是为了做事情或者娱乐，他们不想被迫看广告。为了最佳的用户体验，你对品牌的宣传要轻轻的。

避免和用户关注的内容抢占空间。例如，在屏幕顶部展示另一个一直呈现的工具栏，啥事儿也不干，只是放个商标，就会占用本应属于内容的空间。可以想想其他不会太打扰用户的方式，比如微妙地处理下屏幕背景（译者按：弄成 80%透明铺在背景上什么的）。

搜索要反应迅速、结果丰硕

在那些需要操控或展示海量数据的程序中，搜索是一个主要功能。如果你需要在程序中提供搜索，建议遵循以下指南：

建立索引，时刻为搜索做好准备。不要等用户请求搜索时才做，你不想让程序给用户留下不好的印象吧？

动态过滤本地信息，以便快速反馈展示结果。最好是在用户输入的时候就动态的过滤信息，在用户继续输入时精炼结果。虽然动态过滤信息通常能提供顶级的用户体验，但并不总是现实的。当动态过滤不现实时，你可以在用户点击键盘上的搜索键时开始搜索。这样做时，要对搜索过程提供进度反馈，这样用户就知道没卡壳。

可能的话，也可以在用户输入时过滤远程数据。虽然过滤用户的输入能提供很好的搜索体验，但如果响应时间可能延迟一到两秒时，要征求用户同意。

在列表或列表索引的顶部放置搜索框。用户希望在这个位置找到搜索框，因为他们已经习惯了 Contacts 等应用中的放置习惯。在这里放置搜索框意味着用户翻滚列表或索引时不会有什么干扰，但又能在需要的时候方便地获取。

只在一些特殊的情境中为搜索提供 tab。如果搜索在你的程序中戏份很重，那你可能会想在一个独立的模式中进行搜索。在 iTunes 中，寻找和获得音乐、播客是程序的重点。用户想要在任何模式下都能迅速找到他们最爱的歌曲、艺术家、播客。所以放置一个随时可获得的搜索 tab 是可以理解的。

有必要的話，放一些空的占位符，等数据可用时再去丰富它。通过这种方式，你可以敏捷地为用户提供信息。在某局域网中不存在的视频门户网站上，用户在点击搜索按钮后触发了搜索请求。如果网速很慢，该网站先展示动态的“loading…”，这样用户就知道搜索在进行中。然后，该网站展示只有文本信息的列表，比如视频标题和用户评分，以及代表视频截图的虚线框。当用户浏览视频标题列表时，视频缩略图就逐渐将虚线框替代掉了。

如果数据可以自然地分为几类，就提供一个 scope bar。Scope bar 包括最多四个 scope 按钮，每个代表一类。例如，Mail 提供的 Scope bar 使得用户可以只关注“收”“发”“标题”，或者在全域内搜索。Scope bar 帮助用户将搜索聚焦，显著地减少结果数量。欲知详情，详见“[范围栏（Scope bar）](#)”。

要在 App Store 提供精炼的描述

App Store 中的描述是一个与潜在用户沟通的绝佳机会。除了准确描述程序、高亮你觉得用户最喜欢的特性外，请遵守如下指南：

避免拼写、语法和标点错误。虽然这样的错误并不会让每个人都心生厌恶，但会给某些人留下不好的印象。

避免使用大些字母。大写字母会吸引用户的注意，但是每个词的每个字都用大写会让用户难以阅读。

写出对具体 bug 的修复。如果新版本的程序对老版的 bug 进行了修复，最好在描述里提一下这部分工作的付出。

精炼

像报社编辑一样，尝试把信息写得像报纸标题那样干练。当界面文案简短直接时，用户能迅速明白意思。找到最重要的信息，简练地表达出来，放在显著的位置。这样用户就不必阅读完一大堆文字才找到想要的信息，或是知道下一步应该怎样。

界面元素要一致

用户期待标准的视图和控件，在所有程序里都有一致的外观和行为。

使用标准控件时尊重推荐的使用方法。这样，用户能在学习程序时利用先前的经验。当 iOS 升级标准控件时，你的程序也能得到更新。

对娱乐性应用来说，有必要定制全套控件。这是因为你在营造独特的环境氛围，用户期待在这类程序中探索如何控制环境。

避免彻底改变执行标准动作的控件的外观。如果你使用不熟悉的控件来执行标准动作，用户会花时间研究如何使用它，还会迷惑为什么你的个性控件干着标准控件的活。

iOS 准许你使用很多内置程序中的标准按钮和图标。例如，你可以在 iPhone 和 iPad 上使用 Mail 的刷新、排序、删除、重播和书写图标。



为避免迷惑用户，绝不要使将标准控件和图标用于其他用途。确保你懂得文档上对标准控件、图标用途的描述。不要依靠你对它外观的解释。欲知详情，详见“[系统提供的按钮和图标](#)”。

除了利用用户已有的经验外，使用系统标准控件还有两个好处：

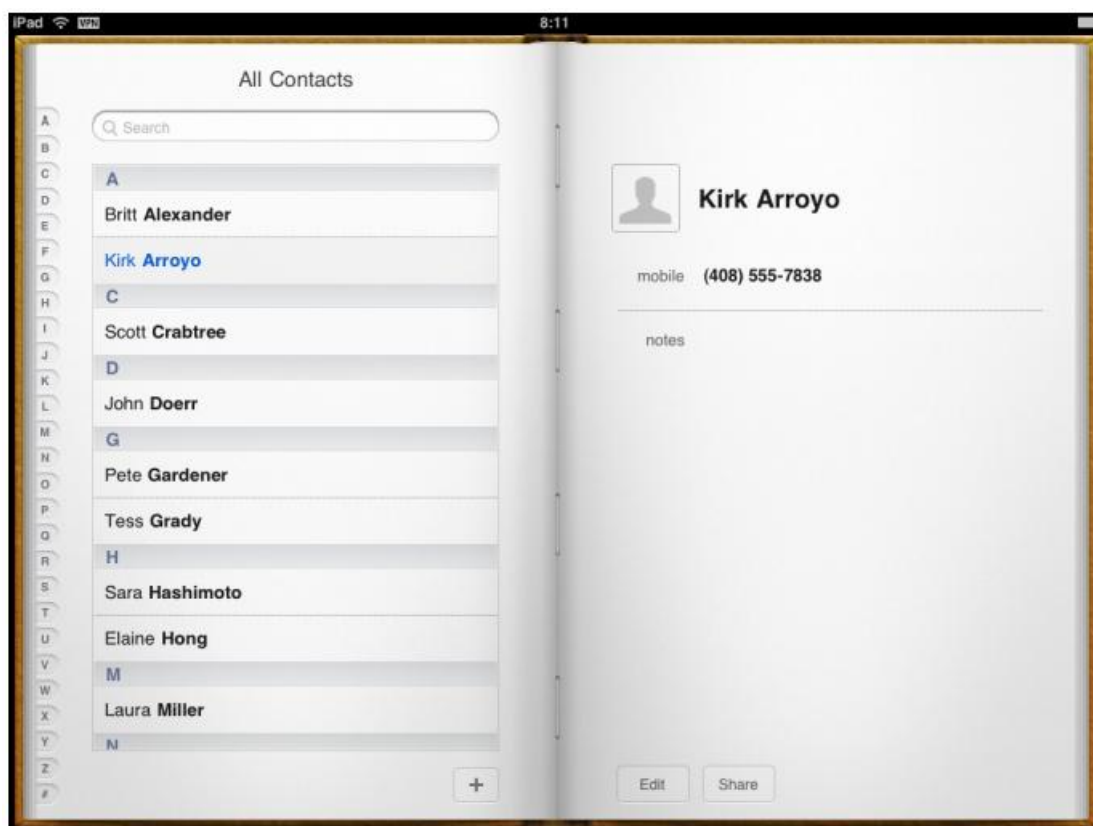
- 减少开发时间，因为你不用再去画皮肤了。
- 增加界面的稳定性，即使将来 iOS 升级会改变标准控件。换言之，虽然图标外观会升级，但它代表的语义是不会变的。

界面编辑器降低了调用标准控件和图标的工作量。请详见“Interface Builder User Guide”里“iPhone OS Interface Objects”

如果你在系统提供的控件和图标里找不到合适的，你可以自己制作按钮和图标。“[导航栏、工具栏和 tab 栏上用的图标](#)”这一章可以提供一些指南。

考虑增加真实感

合适的时候，给你的程序增加一些真实感。通常，程序的外观和行为与真实生活越相近，用户越容易理解如何使用它，也越喜欢用它。例如，用户立刻明白如何使用 iPad 版 Contact 上的地址簿。



在 iPhone 上，用户立刻懂得声音记事簿的作用，并学会使用它，因为它有高精度皮肤以及外观真实的控件。

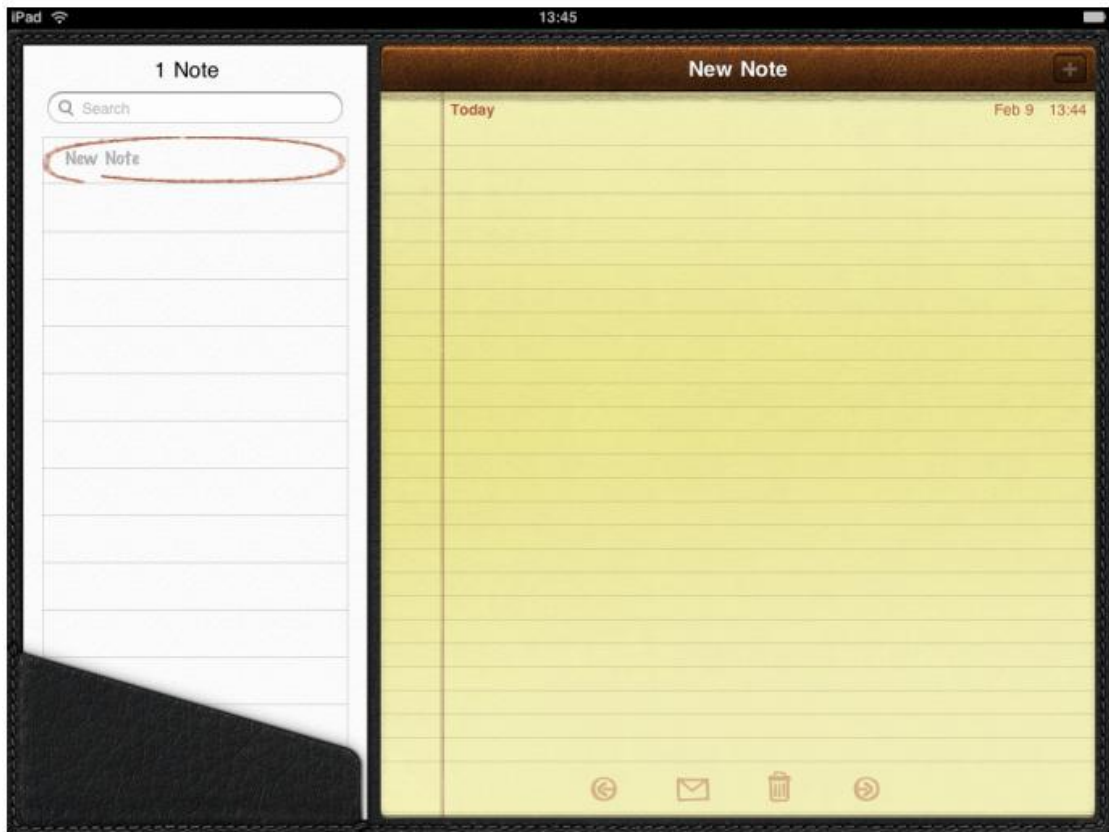


你设计的场景和物体是向用户表达产品灵魂的绝佳机会。别觉得你必须小心翼翼力求完全一致。有时，略作夸张反而比一板一眼看起来更加真实，携带更多意思。用动画强化真实感。运动比外观更强调精确。这是因为人们能接受外观的艺术化，但看到违反物理定律的运动会展晕。尽可能的让你的控件模仿真实物体的运动方式。真实可信的运动提升用户对程序的印象，也愿意花时间来玩它。

用绝佳的图片取悦用户

漂亮的、精致的图片吸引人们使用程序，即使很简单的任务也用得很开心。漂亮的美工也能在用户心中树立品牌形象。iOS 设备对美工有很好的表现力，所以你应该考虑聘用专业的艺术家绘制一流的皮肤。

考虑模仿宝贵的或质地优良的原料的外观。如果木头、皮革、金属的效果适合你的应用，一定花时间确保皮肤看起来真实、贵重。例如，Notes 模仿了优质皮革赫尔金属铆钉的外观。



合适的时候，绘制高精度的画作。大多数情况，放大你的视觉稿并不是一个值得推荐的长期解决方案。应该在比你所需的精度高一级的等级上开始绘画。这样你就可以在压缩它之前添加丰富的细节。这在当你原始稿尺寸是终稿尺寸的整数倍时尤其有用。如果你在图形绘制软件中使用了合适尺寸的网格，就能保证图像在缩小尺寸的过程始终细腻，减少重新锐化的工作量。

确保你的登陆图像和程序图标做工优良。欲知详情，详见“[定制图标和图片指南](#)”

不要把屏幕尺寸的参数写死。这在你希望程序可以在多种 iOS 设备上运行时尤其重要。

处理好方向改变

用户希望能以任何角度使用 iOS 设备。你需要基于你的程序和它支持的功能，确定如何响应用户的殷切期望。

无论什么方向，都保持对主任务的关注。这是优先级最高的准则。用户使用你的程序来浏览或操作他们关注的内容。如果改变方向后，程序不再展示之前的核心内容，会让用户觉得失控了。

如果程序不准备对各个方向提供支持，要再三斟酌。用户希望能在任意角度使用你的程序，你如果能满足这个愿望就太棒了。iPad 用户尤其希望无论他们怎么端着设备，都能正常使用程序。但在某些情况下，程序只能在竖屏模式或横屏模式下运行。如果你的程序只能在一个方向上使用，确保以下几点：

- **以你支持的方向启动，忽略当前的设备方向。**例如，如果你的游戏或者媒体播放程序只以横屏模式显示，就应该在启动的时候就用横屏模式，即使设备是以竖屏模式放置的。这样，如果用户在竖屏模式打开它，他们就知道应该把设备转过来，再浏览内容。
- **避免在界面上用文字等告知用户“把设备转一下”。**以你支持的方向启动已经明确告知用户要旋转设备，不要再添乱了。
- **无论只支持横屏模式还是竖屏模式，要支持这种模式下的两种方向。**例如，如果你的程序只以横屏模式运行，用户无论是以 home 键在左边还是在右边拿着设备，都该能用。而且，如果用户在使用你的程序时把设备转了 180 度，最好你能把内容也跟着转 180 度。

如果你的程序把设备的角度作为一种输入信息，你可以按程序的需要处理旋转。例如，如果你的程序是个游戏，容许用户旋转设备来控制方块移动，屏幕就不用和设备旋转时跟着转了。这种情况下，你应该在游戏开始前让用户选择好以哪种方向来玩。等游戏开始后，就可以按照程序自定义的方式去处理屏幕旋转了。

利用特定的转场动画效果来实现平滑旋转。如果你的屏幕布局很复杂，当用户旋转设备时最好使用“cross-fade”转场。UIViewController 类的 reference 里有对“Handling View Rotations”的解释。

留意加速度。欲知详情，详见“Core Motion Framework Reference”。

在 iPhone 上，在应对设备方向改变时要留意用户的需求。用户经常把手机横过来，是为了想看到更多。如果你仅仅是缩放了一下内容，那么就没满足用户的期望。你应该重新进行页面布局，以更好地适应屏幕。

在 iPad 上，要在所有方向下都能运行，尽力满足用户需求。iPad 的大屏幕诱使用户在想看到更多时旋转屏幕。由于用户不关注设备的外框和 home 键的位置，他们不觉得设备有默认的放置方向。这导致用户希望程序能以他们正在使用的方向运行。你的程序应该尽可能的通过支持各个方向来鼓励用户以各种姿势使用 iPad。

当你的程序需要处理旋转时，请参照以下指南：

- **考虑改变展示辅助信息和功能的方式。**在保证最重要的内容始终在视野范围内的前提下，可以考虑在设备旋转时改变次要信息的呈现方式。

在 iPad 版 Mail 上，账户和收件箱列表是次要内容，邮件是主要内容。在横屏模式下，次要内容放置在分栏的左侧；在竖屏模式下，放置在弹出层里面。

再比如，横屏模式下，游戏的外面有矩形框。转成竖屏模式后，外框需要重绘以适应屏幕。这会导致上下边缘会有大块空白。此时不应把画面纵向拉伸，而应在空白区域填充辅助信息或物体。
- **避免随意改变布局。**应该在各种方向下提供统一的体验，这使得用户可以在旋转设备后沿用使用习惯。例如，如果 iPad 在横屏模式下用网格显示图像，竖屏模式下就不应用列表来展示图像。
- **如果可能的话，应该尽量避免重组信息，重排文字。**尽量在各种方向保持相似的格式，尤其是当用户阅读文字时，避免旋转屏幕后用户找不到刚才读到的地方。

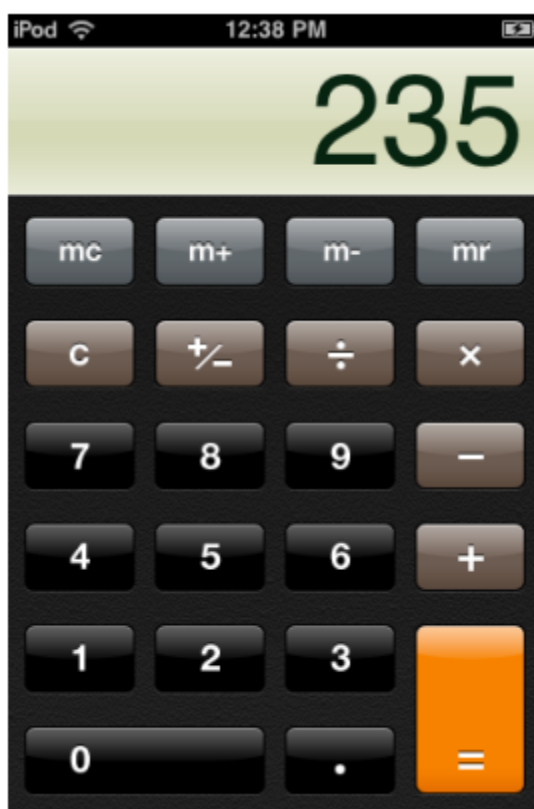
如果重组不可避免，使用动画帮助用户追踪变化。例如，如果你必须在旋转时增减文字的栏数，你也会选择把栏的运动藏起来，简单地让新的排版渐现。为了帮助你设计合适的旋转动作，想想当你旋转真的纸时会对内容的变化有怎样的期待。

- **为每一种方向提供独特的启动图片。**如果每种方向有独特的启动图片，用户就能在每种方向下都能体验到平滑的程序启动。与 iPhone 的桌面不同，iPad 的桌面支持各个方向。所以用户可能会以退出上一个程序的方向启动你的程序。

让目标符合手指的尺寸

iOS 设备的屏幕尺寸会变，但手指的尺寸不会变。请遵循以下指南，以使用户能舒适地使用你的程序。

给你的可触摸元素至少 44×44 像素的面积。iPhone Calculator 就是一个很好的符合指触面积的控件。



如果你的控件更小，或者放置过密，用户必须在触摸前很小心地瞄准目标，而且很可能会按错。这样的话，程序用起来就不那么爽了，甚至会完全不可用。例如，控件太小太密的话，用户必须全神贯注在界面上。

使用微妙的动画表达

动画是很有效的表达方式，它不会妨碍用户的任务、拖用户的后腿。微妙、适当的动画应该做到：

- 表达状态
- 提供有用的反馈
- 增强直接控制的感觉
- 将用户行为的结果视觉化

慎重地添加动画，尤其是在那些非娱乐性程序里。在那些严肃的，用于创造内容的程序里，过多的、无意义的动画会阻碍操作流，降低性能，把用户从任务中诱走。

合适的时候，保持动画效果与内置的程序一致。用户习惯了内置程序的微妙动画。事实上，大多数用户将视图间的平滑转场，设备旋转后的流畅反应和逼真的翻页、反弹视作 iOS 体验的一部分。除非你在设计娱乐性程序，否则就应将定制动画和内置动画协调起来。

程序内部的动画保持一致。就像其他定制一样，保持一致可以让用户依赖从你的程序中学到的经验。

恰当地支持手势

避免将别的动作与标准手势的效果联系起来。避免用定制的手势去诱发标准手势的效果。

使用手势来加速任务的完成，但不要作为唯一的完成手段。虽然用户大多知道很复杂的标准手势，比如水平滑动（swipe）或分开手指，这些姿势却没那么普遍。

例如，当在 Mail 中浏览邮件列表，用户通过展开预览栏，单击其上的删除按钮来删除邮件。用户可以通过两种方式展开删除按钮：

- 单击导航栏中的编辑按钮，每一行预览上都会出现删除控件。然后单击特定行的删除控件，就能展开删除按钮。

这种方式步骤冗余，但容易找，因为它只需要触摸动作，并且是以清晰的“编辑”按钮为起点。

- 在预览栏上水平滑动，删除按钮出现。

这种方式更加快捷，但是需要用户学习，记住水平滑动的手势。

确保总是有一种简单直白的方法完成操作，即使操作步骤多一些。简单的手势允许更关注内容和体验，而非交互动作。

一般而言，避免定义新手势。当你引入新手势时，用户需要费心去发现新手势并记住它。

开发娱乐型程序是个例外。例如，要求用户在列表项上画圈以展示删除按钮将会是非常困惑且难用的。但是在游戏中可能需要用户画圈以旋转某个器件。

确保你采用的手势与程序的功能以及用户的期望相适应。如果用户要经常使用程序中的某个功能，希望越快完成越好，你应该只使用标准姿势。但是，如果你的应用包含逼真的控件，或者环境让用户很有探索欲望，定制手势也是可以的。对于标准手势，请详见[“程序响应手势，而非点击”](#)。

对于 iPad，可以考虑多点触摸手势。iPad 硕大的显示屏为定制多点触摸手势提供了空间。虽然不是每个程序都适用复杂的手势，但它可以为那些用户长时间使用的程序具有更丰富的体验，例如游戏或者创造内容的环境。时刻谨记非标准手势很难被发现，不应是完成某任务的唯一途径。

只在必要的时候要求用户存储

用户应该有信心，他们的工作总是自动保存，除非自己按了取消或删除。如果你的程序帮助用户创建或者编辑文档，要确保它不需要明显的保存按钮。iOS 程序应对保存用户的输入负起责任，可以按周期进行，并在用户打开其他文档或退出程序时进行。

如果主要任务不是产生内容，但允许用户在浏览信息和编辑信息间切换，那么要求用户保存变动是合理的。这种情况下，在展示信息的视图模式下呈现“编辑”按钮是个好办法。当用户触摸编辑按钮时，可以用保存按钮替换它，并增加一个取消按钮。编辑按钮的变化提醒人们现在进入编辑模式。取消按钮可以让用户不保存任何变动，恢复到编辑前的状态。

对于 iPad，保留那些用户在浮出层输入的信息。因为用户有时会不小心把浮出层关掉。更多信息，请详见[“浮出层（只限 iPad）”](#)（p83）

让模态化任务表现地暂时且简单

可能的话，尽量减少进入模态化环境执行任务或提供反馈的次数。iOS 应用应该允许用户与 iOS 任意地交互（nonlinear）。模态通过打断用户的工作流，强迫用户完成一系列操作来剥夺自由。

模态在以下情况最为适用：

- 很有必要获得用户的注意
- 没有用户明确提供的数据，该任务无法完成

用户很喜欢在模态视图里完成一个子任务，因为场景转换清晰且短暂。但是如果子任务太过复杂，用户会在进入模态视图后迷失主任务。当模态是全屏模式的或有很多分支状态时，风险还会加剧。

保持模态任务简短精炼。你应该不想让用户觉得模态任务是嵌在你程序里的一个小程序吧？千万注意别让模态任务有层级结构，因为用户很容易忘记回去的路。如果模态任

务必须包括子任务，一定要给用户一个单一、清爽的路径来浏览层级结构，避免环形路径。

在模态任务中总是提供明显、安全的出口。用户应该总是能够在离开模态时预测他们工作的命运。

如果任务需要分层级的模态，确保用户明白如果他们点击子级页面上的 Done 按钮后会有什么结果。检查一下任务，看看子级页面上的 done 按钮按下后是结束了那一小部分任务还是结束了整个任务。可能得话，避免在子级页面加 Done 按钮，这真的很让人困惑。

立即启动

iOS 程序应该在用户想用它们的时候立刻启动，毫无延迟。在启动时，iOS 程序应该：

展示与应用程序第一屏一样的启动图片。这可以缩短对启动时间的知觉。

避免呈现“关于”窗口或者 splash。一般而言，避免添加任何阻碍用户立刻使用程序的元素。

在 iPhone 上，提供合适的状态栏样式。通常，状态栏要和程序的其他界面部分风格一致。

以合适的默认方向启动。在 iPhone 上，默认的方向竖屏模式。在 iPad 上，默认方向是当前设备的方向。如果你的设备只支持横屏模式，就按横屏模式启动，不用管设备当前的方向。用户会去旋转设备的。

只支持横屏模式的应用应该支持两种方向的横屏模式（Home 键在左或在右）。如果设备已经是横屏模式了，那就直接以此模式启动，除非有别的很好的理由。其他情况下，程序应以 home 键在右侧的方向启动。

避免向用户询问设置信息。应遵照以下指南：

- **只为 80% 用户解决问题。**这样的话，大部分用户不需要设置，因为你的程序已经按他们期望的方式设置好了。如果某个功能只有一点点用户会用到，或者用一次就扔，这样的功能还是放弃吧。
- **从用户以外的地方获得尽可能多的信息。**如果你要用到任何用户在内置程序中储存的信息，可以向系统提出请求。不要让用户再输一次。
- **如果你必须从用户那获得信息，要让用户在程序内进行输入。**然后尽快把这些信息存下来。这样，用户就不用先退出程序才能进到 iOS 设置里。如果用户稍后还需要更改这些信息，他们可以在任何时候到程序的设置模块。

从程序上次离开的位置启动。记住如何回到上一次退出的位置不应是用户的责任。

随时准备停止

iOS 程序需要在用户按下 Home 键的时候停止，然后打开别的程序。所以用户不会点击程序的关闭按钮或是从菜单里选择退出。为了提供好的退出体验，iOS 程序应该：

经常且快速保存用户进度。因为用户可能在任何时间选择退出。

停止的时候保存当前的状态，尽可能的保留细节。这样用户再次打开程序时不会损失细节。例如，如果你的应用展示可翻页的数据，保存当前的翻页位置。

不要自动退出

绝对不要自动退出，因为用户倾向于觉得是程序崩溃了。如果你的程序确实无法像预期的那样工作，你需要告知用户当前的情景，解释他们可以做什么。基于当前情景的危机程度，你有两种选择：

展示一屏吸引人的内容，描述当前的问题，提供修正。这屏内容告知用户程序没出问题。它给予用户控制权，让他们决定是采取补救措施还是忽略报错，又或是按 Home 键然后打开别的程序。

如果只有部分功能失常，那就在用户使用这些功能时弹出警告框。

有必要的话，展示许可证或者免责声明

如果你提供终端用户许可协议 EULA，App Store 会把它展示出来。用户能在获得程序前阅读到它。

可能的话，避免用户在第一次运行程序时必须同意你的协议。没有协议的话，用户可以立刻享用你的程序。但是，即使这是更好的体验，但并不适用所有情况。如果你必须让用户签署协议，要让签署过程与你的界面和谐相处，减少对用户的打扰。

可能的话，在程序描述或者 EULA 中提供免责声明。用户可以在 App Store 中浏览它，你可以平衡商业需求和用户体验。

适用于 iPad：增强交互性（别只增加功能点）

最好的 iPad 用户给用户创新地与内容交互的体验，同时帮助用户完成任务。

忍住添加与主任务无关的功能点。探索让用户看更多、用更多的方法。不要以为 iPad 屏幕大是为了让你把 iPhone 版程序上被砍掉的功能再找回来。

为了让你的程序与众不同,想一想怎么提升用户体验,不要让琐碎的功能点干扰主任务。例如:

- 一个让用户可以读书、持续跟踪阅读列表的程序可以在大屏幕上给用户带来优越的阅读体验。用户不必切换到其他屏幕去管理阅读列表,就可以在弹出层里让用户把喜欢的章节复制进去。这个程序还容许用户添加书签和标注,和他人交换书单,比较阅读进度。
- 一个战斗机游戏可能会在屏幕前蒙一层半透明的头盔。玩家可以触摸逼真的控件迎敌,或者在地图上定位他们。
- 足球游戏可以展示一个更大更真实的球场以及细节更好的球员。用户可以管理队伍,定制球员。它还可以允许用户不离开球场视图就查看球员信息。它还支持多人模式,两个人可以各用一队。
- 记事本程序可能会提供在手绘模式与文字模式间切换的功能。用户在记录的时候可以在这些模式间切换。

适用于 iPad: 减少全屏转场

哪部分内容变了,就只有哪一部分转场。在某些信息变化时,不要呈现全新的一屏,尝试只是更新这一块小区域。只转换部分视图和物体,不要换全屏。大多数时候,不建议做整屏切换。

每次只切换一部分屏幕的话,你的程序看起来会很稳定。这帮助用户追踪他在任务中的位置。你可以使用分栏显示、浮出层等方式取代全屏切换。

适用于 iPad: 抑制你的信息层级

使用大屏幕和 iPad 特制的 UI 元素,让用户可以在一个地方获取更多信息。虽然你不想把太多信息打包到一个屏幕上,你也不想让用户有必须到不同屏幕上找东西的感觉。

一般来说,主要内容集中在主屏幕上,额外信息或工具用辅助手段展示,如 popover。这使人们容易获得他们所需要的功能,无需离开的主任务环境。

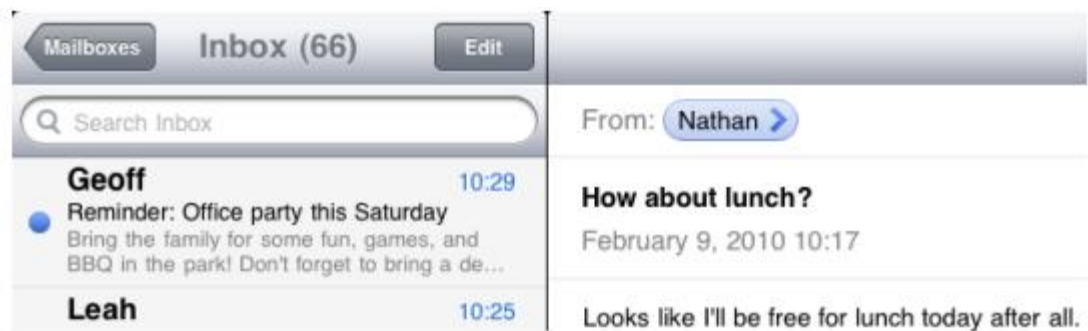
使用 iPad 的大屏幕以及分栏视图和 popover 等界面元素,你不必再像 iPhone 那样每屏展示一级。例如,你可以:

在右侧分栏放置一个导航栏,让用户沿层级结构向上回溯到顶级类目。该类目始终展示在左侧分栏中。这至少可以让你的层级结构极少一级。因为同一时刻屏幕上至少有两

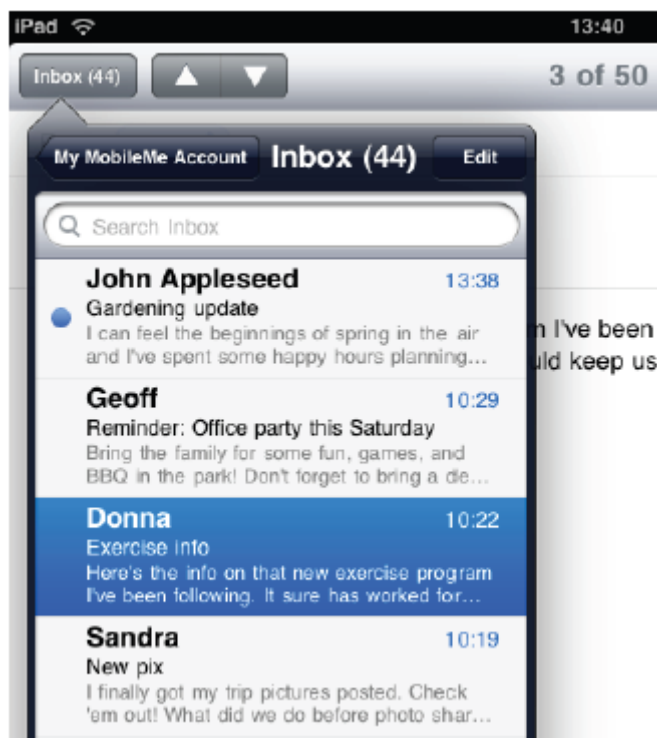
级内容。



在左侧分栏放置一个导航栏，让用户追溯一个比较浅的层级结构。然后，把最细一层的内容呈现在右侧分栏里。这样同样可以通过同时呈现两级内容来简化层级。例如，以风帆视图模式使用 Mail 时就使用了这种设计，在左边分栏展示账户、收件箱、邮件列表，右边分栏展示每一份邮件的内容。



使用浮出层实现操作或呈现用于操作屏幕上事物的工具。浮层可以在当前屏幕的顶部临时地展示操作和工具，这就意味着用户不必转场到其他屏幕来做这些事情。例如，竖屏模式的 Mail 使用浮出层来展示账户、邮件箱和邮件列表的层级结构。



在工具栏上提供分段控件，来切换不同的信息呈现方式和不同的信息分类。这样，你可以通过点击屏幕顶部或底部的一个按钮来展开浮层，切换浏览方式或分类。更多信息，请详见“[工具栏](#)”和“[分段控件](#)”。例如，iTunes 在顶部工具栏提供分段控件，来切换某一类内容的浏览方式。



使用 **tab 栏** 切换不同信息类别或者程序模式。在 iPad 程序上，tab 栏更有可能被用于过滤器或者类目切换，而非模式切换。例如，iTunes 使用 tab 栏切换不同类型的媒体。



可能的话，避免使用 tab 栏切换全屏，因为 iPad 上最好减少全屏切换的场景。

适用于 iPad: 考虑将浮出层用于不同模态的任务

在浮出层打开或处于模态视图下时,用户不能与主屏幕交互,在这一点上两者是相似的。但是模态视图总是模态的,而浮出层可以有两种用法:

- 模态的: 浮出层出现后,降低周围区域的亮度,需要用户主动点退出。这种情况与模态视图非常相似,但浮出层给人的感觉更轻一点。
- 非模态: 浮出层周围的区域不会变暗,用户可以点击浮出层周围的任何区域来清除浮出层(发出激活浮出层的按钮)。这种行为方式让非模态浮出层看起来像从另一种角度观察程序,而非进入另一种状态。

另外,浮出层总是有一个小箭头,指向激活它的按钮或区域。这个线索帮助用户记住他们是从哪里进入现在这个场景中的。这也使模态浮出层看起来只是暂时的,而不像模态那样,不提示是怎么进入到模态中来的。

如果你在 iPhone 中用模态视图来呈现任务,可以换浮出层试试。思考以下问题可以帮助你判断使用浮出层是否合适:

- 该任务是否需要不同类型的输入? Y? 用浮出层吧

虽然键盘兼容浮出层和模态视图,但浮出层更适合展示拾取器或一系列选项。

- 该任务是否需要在多个层级的视图间穿梭? Y? 用浮出层吧

浮出层更适合展示多个页面,因为用户不容易把它和主页面弄混。

- 用户希望能在任务完成前去主页面上干点什么吗? Y? 用非模态浮出层吧

因为用户可以在非模态浮出层周围看到主页面。轻轻一点主页面,浮出层就消失了。记得保留浮出层里的任务进度,用户还可能再回来的。

- 任务足够复杂吗? 是否是程序的主要功能? Y? 用模态视图吧

模态视图伴随很大的转场,帮助用户专注于任务直至完成。模态视图的面积也足够大,可以放置更多的输入控件。

如果此任务是程序的主要功能,但是不是很深,模态化的浮出层会更为合适。这是因为浮出层的轻盈更适用经常执行的任务。

- 如果,任务只执行一次或寥寥数次,比如安装过程?? 用模态视图吧。

当用户只需要做一次任务的时候,他们不是很关心当前的环境。

还有些 popover 的其他用法,比如说提供辅助工具(更多信息,请详见[浮出层\(仅限 iPad\)](#))。而且,iPad 还可以把操作列表放在浮出层里(更多信息,请详见[操作列表](#))。

如果你决定使用模态视图,请查阅有哪些样式可供使用(更多信息,请详见[模态视图](#))

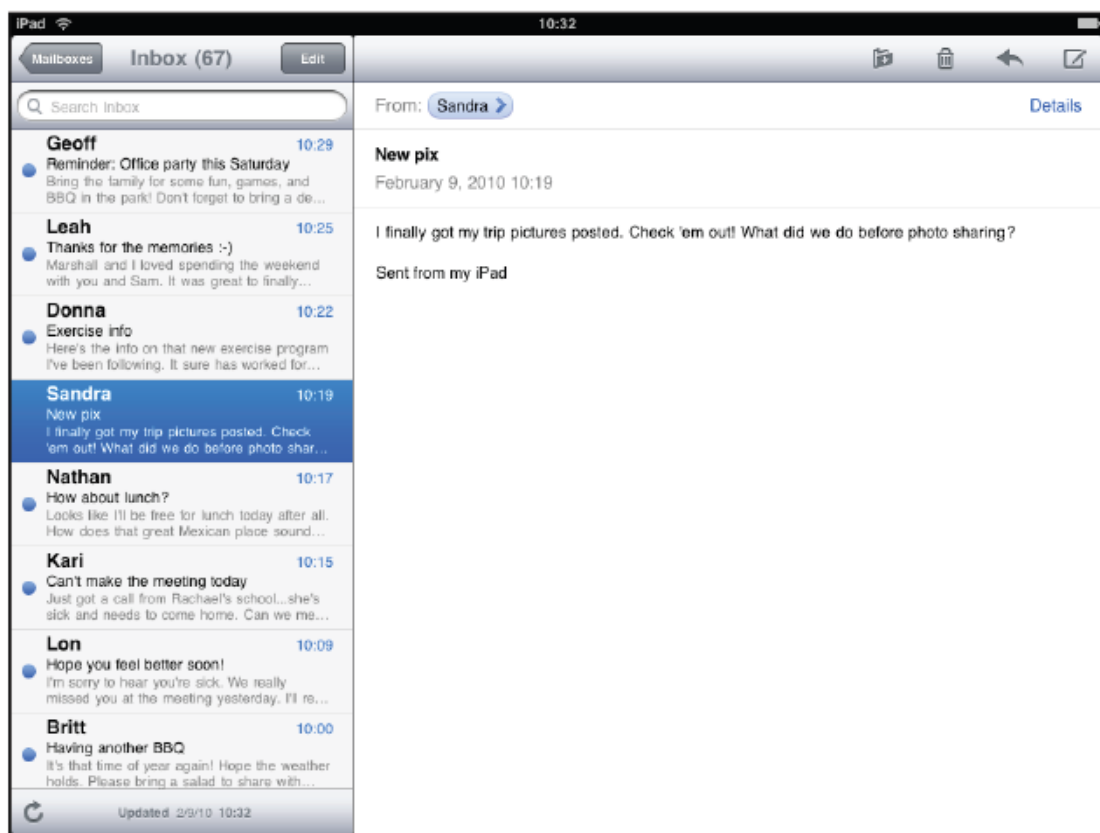
适用于 iPad: 考虑将浮出层用于不同模态的任务

如果你的程序有工具栏，把它移到屏幕顶部吧，别放在下面了。iPad 的屏幕非常宽，顶部的一条工具栏足够放下所有的功能。这给你更多的垂直空间展示核心内容。

例如，iPhone 版 Mail 在浏览邮件页面的工具栏上放置了刷新，管理，删除，回复和书写。



iPad 版 Mail 在邮件上方的工具栏提供了其中四个按钮。刷新按钮移到了邮件箱列表。



第7章 iOS 技术使用指南

iOS 提供很多深得用户喜爱的技术，比如多任务、复制黏贴、推送提醒等。

从用户的角度来说，这些技术已经 iOS 形象的一部分。但开发者只需必须十分小心谨慎，才能实现技术和用户体验的完美结合。

多任务

多任务帮助用户快速地在多个最近使用的程序间切换。当程序退出的时候，实际上是在后台被挂起了。挂起的程序不需要重载界面，所以可以快速恢复。用户可以在多任务管理器中选择最近使用过的程序。



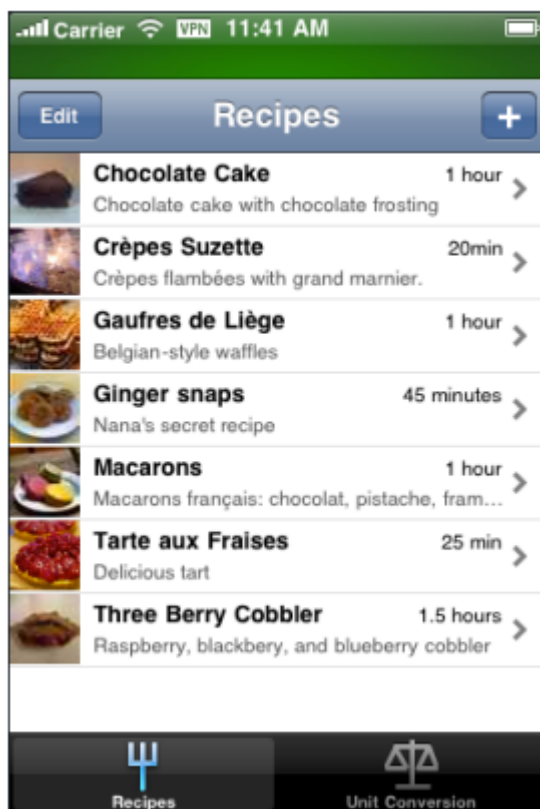
要想在多任务环境中良好生存，必须学会与设备上的其他程序良好相处。这也意味着：

- 对其他程序导致的中断和带来的声音要妥善处理
- 快速、平滑地停止和重启
- 当处于后台时，要对行为举止负责任

遵从这些指南可以帮助你多任务环境中取胜。

随时做好中断和恢复的准备。多任务环境中，你的程序更有可能被后台的程序打断。其他的特性，比如广告和快速切换，也会导致经常被打断。越快越准地保存好当前状态，用户就能更快地重启程序，从中断处继续。

确保你的界面能支持双倍高度的状态栏。双倍高度状态栏用于展示进行中的通话，录音等。如果程序没有准备好，就可能导致布局混乱。例如，界面可能会压缩或被盖住。在多任务环境中，更要留心这一点，因为有了更多的程序可以激活双倍高度的状态栏。你可以在测试时尝试把双倍高度的状态栏调出来，看看是否有问题。



确保那些需要用户注意或参与的信息不被遗漏。例如，如果你的程序是游戏或者媒体浏览器，确保用户在切换到其他程序时不会遗漏重要的内容或事件。当用户切换回来的时候，要让用户觉得好像就没离开过那样。

确保程序的声音展示得当。多任务使得你的程序与其他程序同时运行。也使得你的声音应该在程序进入后台时停止，在程序恢复后继续。为了保证声音可以符合用户的期望，与其他声音和谐相处，请详见“[声音](#)”

恰当地使用本地提醒。程序可以在任意特定时间发送局部提醒，无论是挂起，在后台运行还是根本就没运行。为了用户体验好一点，请不要塞给用户太多的提醒，并遵照“[本地和推送提醒](#)”指南。

合适的话，在后台把用户触发的任务默默完成掉。在触发某任务后，用户通常希望在即使切换到其他程序后，这个任务也会自动进行。如果你的程序正执行用户触发的任务到一半，不再需要用户的指点，那就应该在后台把它做完。

打印

在 iOS4.2 及以后的支持多任务的版本中，用户可以无线打印内容。你可以用内置程序打印图片和 PDF 文档，或者可以使用专业的打印程序来定制格式。iOS 会在选定的打印

机上管理打印顺序。

通常，用户在打印的时候会按标准的打印按钮。接下来，用户可以选择打印机，设置打印属性，按下打印键后开始打印。在 iPhone 上，这个视图以操作列表的方式从屏幕下方出现；在 iPad 上，操作区会出现在指向打印按钮的浮出层里。



用户可以在打印中心里检查打印任务，这是一个只有用户安排打印任务时才运行的后台系统程序。在打印中心里，用户可以查看当前的打印序列，获得打印任务的详情，或者取消任务。

只要在代码里加上几句话，就可以让你的程序支持基本的呃打印功能。为了保证用户体验的质量，请遵照以下指南：

使用提供的打印按钮。用户对这个按钮的意义和行为非常熟悉，所以应该尽可能地使用它。如果你的程序没有工具栏或者导航栏，那就另说了。这样的话，你需要定制可以放在主内容区的打印按钮。

如果打印是当前的主要任务，那就应该展示打印按钮。如果打印不适合当前情景，或者用户不想打印，不要在视图里展示打印相关的功能。

合适的话，为用户提供额外的打印选项。例如，你可以允许用户设置打印范围或拷贝数。



如果用户不能打印，就不要展示打印相关的界面。在展示打印功能前，要确保用户的设备支持打印。

iAd 富媒体广告

在 iOS 以后的版本里，你可以在程序里内嵌广告，用户看到它或点击它后就可以为你创造收入。你要想好，什么时候、怎样来展示广告才能让用户有动力去看，又不影响用户的任务。

你可以在程序里添加广告区，展示由 iAd Network 提供的内容。当用户按在这些区域（banner）上时，广告就有反馈，比如播放电影，展示更多内容或者登陆 Safari 打开网页。这些内容可能会盖住整个界面，或者把程序挤到后台去。

Banner 的尺寸会随设备和摆放方向变化而变化。

Table 7-1 **Banner view dimensions**

| Device | Portrait | Landscape |
|--------|-----------------|------------------|
| iPad | 768 x 66 points | 1024 x 66 points |
| iPhone | 320 x 50 points | 480 x 32 points |

为了保证 banner 广告能与程序贴合更紧密，提供更好的用户体验，请遵照以下指南：

把它放在屏幕底部。这种放置方法会随栏的具体情况略作调整。

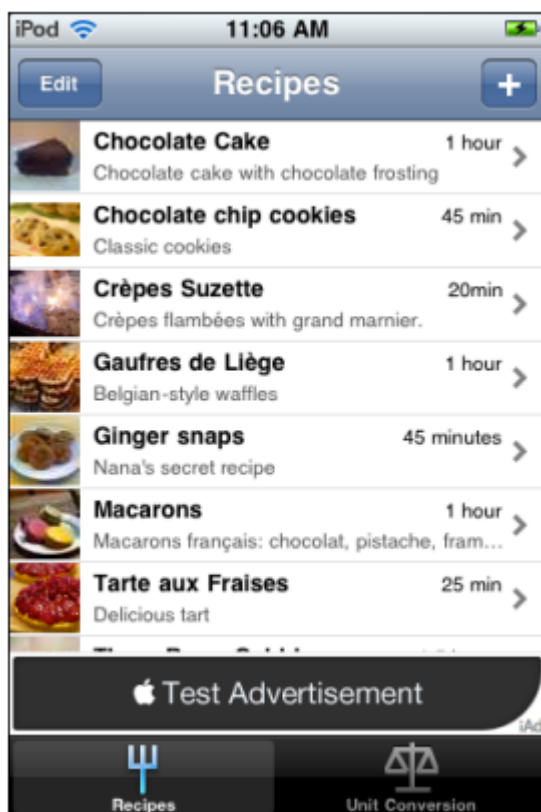
如果屏幕底部没有栏，就在屏幕底部放上 banner。



如果一个栏都没有，就在屏幕底部放上 banner。



如果有工具栏或者 tab 栏，可以把 banner 放在它上面。



确保广告在你的程序里不是那么唐突:你可以选择在哪一屏上展示广告。例如,你可以把它作为主任务各环节的连接处。当用户觉得他们的任务没被打断时,才更有可能留意 iAd。这对于游戏等娱乐性程序更为重要。你不会想用 banner 干扰用户玩游戏吧?

尽可能的在两种方向上都放上 banner 广告。最好让用户在浏览你的广告时不需要切换方向。而且,支持两种方向可以给你留下更广阔的展示空间。

不要让广告在翻页时跟着动。如果你程序中的内容支持需要翻页,要确保 banner 的位置保持固定。

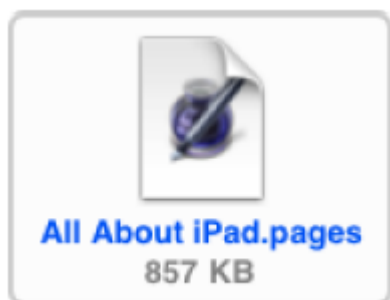
当用户查看或触摸广告时,暂停那些需要用户注意或操作的任务。用户不希望因为看你的广告而漏掉重要的事情,也不希望程序干扰广告的体验。较好的方法,找到那些程序转入后台时会暂停的进程,在用户浏览广告时也暂停同样的进场。

除特殊情况外,不要停止播放广告。一般而言,当用户查看或触摸广告时,你的程序持续运行并接收事件。所以有可能出现一些需要用户注意的突发事件。但是,很少有需要关闭广告的情况。为数不多的一种情况就是 VoIP 电话服务。当有电话进来时,就应该停止播放广告了。

快速文件预览

在 iOS4 以后,即使你的程序不能打开某类文件,用户也可以预览它。例如,用户可能想预览下从 web 或其他源头获得的文档。

在用户预览文件前，他们可以从定制的文件图标里获得一些信息。例如，用户从 email 附件获得文件后，iPad 版 Mail 会展示文档的图标、标题和文件大小。用户可以按下图标来预览文档。



你可以在程序里用别的方式展示文件，全屏展示，或者模态化展示。展示方式应依设备而定。

在 iPad 上，用模态化的方式预览文档。iPad 的硕大屏幕很适合以一种用户可以快速离开的暂态展示预览文档。缩放转场非常适合用于展示预览。

在 iPhone 上，应该采用更加专注的方法，带导航栏的视图更为适合。这可以帮助用户在预览文档时弄清自己从哪里来的，又到了哪里去。虽然在 iPhone 上用模态预览文档也是可以的，但并不推荐这样。iPhone 也不支持缩放转场。

用带导航栏的视图模式预览文档时，会出现预览专用的导航控件(如果已经有了工具栏，这些控件会出现在工具栏上)。

声音

iOS 设备可以实现很棒的声音效果。在你的程序中，声音可能是核心的组成部分，或者偶尔出现增强效果。无论声音扮演的角色如何，你应当知道用户对声音抱着怎样的期望，应该如何来满足它。

理解用户的期望

用户在设备上调节声音，也可能带着有线或无线的耳机。用户也对应该如何控制声音有所期望。虽然你可能觉得有些期望很奇怪，但它们都遵循由用户来控制何时应听到声音的原则，而非由设备。

用户会如下情况切换到静音：

- 避免被不符合期望的声音打扰，比如电话或短信的提醒音
- 避免听到由操作带来的副产品，比如键盘音或其他反馈音、故障音或启动音乐
- 避免听到与玩游戏不太相干的声音，比如偶发的故障音。

例如，在剧院里用户希望把设备调成静音，以免打扰其他观众。这种情况下，用户依然想使用程序，但并不想被出乎意料的声音吓到，比如铃声或新消息提醒音。

对某些刻意设定的声音，当用户的操作触发它时，Ring/Silent 的设置不能关闭它。例如：

- 媒体播放器里的播放不会静音，因为播放是由用户主动请求的。
- 闹钟不会静音，因为它是由用户明确设定为“有声”的。
- 在语言学习软件中的声音片段不应静音，因为用户有明确的欲望想听到它。
- 语音聊天程序中的对话不应静音，因为用户打开它就是为了听声的。

用户使用设备音量键所做的调节会影响所有设备播放的声音。这包括歌曲、程序声音和设备声音。用户总是可以使用音量键静音，无论 Ring/Silent 设置在哪一档上。使用音量按钮调节程序当前的声音时，整个系统的音量都会受影响，铃声除外。

对于 iPhone 来讲，当没有任何程序在播放声音时，调节音量按钮就会调整铃声音量。

用户使用耳机来私下地听声音，同时解放双手。无论这些设备是有线、无线，用户对体验都有一些期望。

当用户插入耳机或联通无线声音设备时，他们是想私下里听这些声音。由于这个原因，他们希望当前正在播放的声音不要暂停。

当用户拔出耳机或与无线设备断开时，他们不想把刚才听到的声音自动共享给其他人。所以，他们希望当前正播放的声音可以暂停，等到用户准备好了再重新播放。

定义声音的行为

有必要的话，你可以为自己的程序设定相对的、独立的音量水平，以便产生合适的声音输出。但是最终的声音输出还是要受系统音量的管制，由音量键或者音量滚动条来调节。这意味着程序音量的控制权还是在用户手中。

合适的话，要给程序加上选择声道（audio route）的控件。即使当用户没有插上或拔下无线声音设备，他也希望能选择不同的声道。为了解决这个问题，iOS 自动提供了让用户选择输出通道的控件。由于切换声道是用户触发的行为，用户期望正在播放的声音不要停。

如果程序只是在主界面上放些伴奏声，就使用 system sound services。SSS 是一种产生警告声、界面声（UI Sounds）和振动的后台服务。当使用 SSS 播放声音时，你不能设置该声音如何与设备上的其他混响，也不能设置如何处理中断或其他设备设置的改变。

如果声音是程序的核心功能之一，使用 `Audio Session Service` 或者 `AVAudioSession` 类。这些类不能直接产生声音，但可以帮你设置该声音如何与设备上的其他混响，如何处理中断或其他设备设置的改变。

在 iPhone 上，无论你用了什么样的技术去控制声音，iPhone 总能打断当前运行中的程序。因为任何程序都不应妨碍到用户接听来电。

在 `Audio Session Service` 中，`audio session` 充当程序与系统的调节者。它最重要的一面是 `category`，可以用来定义程序的声音行为。

为了运用 `Audio Session Service` 的优势，提供用户期待的声音体验，你应当选择与你的声音行为最相符的那一类。这要看你的程序是只在前台时播放声音，还是在后台也能播放。可以参照如下指南：

- **根据 `category` 的语义来选择，而非对行为的细节描述。**这样能保证你的程序能符合用户期望。另外，如果将来类目对应的行为方式被修订了，也能保证你选择类目是合适的。
- **少数情况下可以在标准行为的基础上稍作定制。**类别对应的标准行为与大多数用户的期望一致，所以在自定义前要考虑清楚。例如，你可能想添加“Should Duck”属性，以便你的程序音比其他声音都大（除了铃声），以便符合用户的需求。
- **考虑根据设备当前的声音环境来选择类别。**这也许是合理的，因为用户也许想使用你的程序的同时听着其他声音。这样的话，当用户启动你的程序时，不要暂停用户正在听的声音或强迫用户选择音轨。
- **一般而言，避免在程序运行过程中改变类别。**改变类别的主要原因是你的程序可能需要在不同的时候支持录音和回放。这种情况下，可以在录音类和回放类之间切换，而非只用“播放和录音类”。这是因为选择录音类可以避免在录制过程中有警告声（比如新短信）出现。

表格 7-2 中列述了你可以使用的类别。

Table 7-2 Audio session categories and their associated behaviors

| Category | Meaning | Silenced | Mixes | In Background |
|------------------|--|----------|--|---------------|
| Solo Ambient | Sounds enhance app functionality, and should silence other audio | Yes | No | No |
| Ambient | Sounds enhance app functionality, but should not silence other audio | Yes | Yes | No |
| Playback | Sounds are essential to app functionality, and might mix with other audio | No | No (default) Yes (when the Mix with Others property is added) | Yes |
| Record | Audio is user-recorded | No | No | Yes |
| Play and Record | Sounds represent audio input and output, sequentially or simultaneously | No | No (default) Yes (when the Mix with Others property is added) | Yes |
| Audio Processing | App performs hardware-assisted audio encoding (it does not play or record) | - | No | Yes * |

这里是一些用例，介绍如何选择与用于期望一致的声音类别。

用例 1：帮助用户学习语言的教学程序。你可以提供：

- 当用户触摸某个控件时提供反馈音
- 当用户想听正确的发音时记录单词和词汇

在这个程序里，声音是核心功能。用户使用这个程序来听所学语言的单词和短语发音。所以即使设备的声音锁掉了也应该发音。应为用户需要清楚地听到发音，所以希望其他声音都静下来。

为了提供符合用户期待的声音，你应该使用 playback 类。虽然这个类可以在稍加修改后支持与其他声音混响，这个程序应当使用默认的行为，以便保证其他程序的声音不会带来干扰。

用例 2：Voice over internet protocol (VoIP) 电话程序，你应当提供：

- 获得声音输入
- 播放声音

在这个程序中，声音是核心功能。用户使用该程序与他人联络时，往往有其他程序正在运行。用户希望能在把设备设为静音或设备被锁定时依然能接听电话，并希望在通话过程中其他程序保持安静。用户也希望当程序跑在后台时依然能收到电话。

为了提供符合用户期待的声音，你应该使用 Play and Record 类。另外，只能在需要的时候激活 audio session，以便于在不通话的时候可以使用其他声音模式。

用例 3：允许用户扮演角色执行多种任务的游戏。你应带提供：

- 各种游戏音效
- 音乐音轨

在这个程序中，声音可以提升用户体验，但是并不是主任务的核心。而且，用户也可能想静音玩游戏，或者听着其他音乐玩。

最好的策略是看一下程序启动时用户有没有在听其他声音。不要在开始时让用户选择想听哪个音轨。可以调用 ASS 的函数 `AudioSessionGetProperty`，读取 `kAudioSessionProperty_OtherAudioIsPlaying` 的值。基于这个值，你可以选择 `Ambient` 或者 `Solo Ambient` 类（两种类都支持用户静音玩游戏）。

- 如果程序启动时用户正在听其他声音，应该假设他想继续听，不应该强行用当前程序的音轨代替它。这种情况下，应该选择 `Ambient` 类。
- 如果程序启动时用户没在听其他声音，选择 `Solo Ambient` 类。

用例 4：将用户精确实时地导航到目的地的程序。你应该提供

- 为行程中每一步说出方向
- 少许声音反馈
- 可以继续收听其他声音

在这个程序中，无论它在前台还是后台运行，声音导航提示都是主要任务。基于此，你应该使用 `Playback` 类，允许你在设备被锁或切换到静音、程序跑在后台时也能播放声音。

要想让用户运行程序的同时可以听其他声音，你可以添加 `kAudioSessionProperty_OverrideCategoryMixWithOthers` 属性。但是，你同时也希望用户能在其他声音播放时听到声音指示。这可以通过增加 `kAudioSessionProperty_OtherMixableAudioShouldDuck` 属性来实现。这样你的声音就会比所有其他音轨的声音大（除了 iPhone 上的电话声）。

用例 5：允许用户上传图片和文字到网页的博客程序。你应该提供

- 简短的启动音乐
- 与用户操作相匹配的音效（比如上传成功后的提示）
- 上传失败时的警告音。

在这个程序中，声音可以提升用户体验，但不是核心功能。主任务与声音无关，用户不依赖声音也能成功满足需求。在这个勇利用，你应该使用 `SSS` 来播放声音。这是因为所有该程序对声音的要求都与 `SSS` 的特性相符（播放界面音效和警告音，无视设备是否被锁，无视 `Ring/Silent` 的设置）。

管理声音冲突

有时，来自多个程序播放的声音会冲突。例如，打电话的过程中电话声会和正在播放的其他声音冲突。在多任务环境中，这种冲突的频率会更高。

为了提供符合用户期望的音效，iOS 需要你：

- 确定你的程序会导致怎样的冲突。
- 在冲突解除后选择合适的应对方式。

每一个程序都需要确认可能会导致的冲突，但并不是每一个程序都需要制定应对冲突的方案。这是因为对于大多数程序来说，在冲突解除后继续播放就是一个合适的方案。只有以录音回放为主，提供回放控件的程序需要额外地定义好应对方案。

从概念上来说，根据导致冲突的声音类型以及用户对冲突后的期待来分，共有两种声音冲突。

- 可恢复的冲突。当用户在使用以听觉为核心的程序时被其他声音暂时干扰时，会出现可恢复的冲突。

可恢复中断结束后，有回放控件的程序应该恢复到中断发生前的状态，无论之前是正在播放还是暂停状态。没有回放控件的程序应该恢复到播放状态。（After a resumable interruption ends, an app that displays media playback controls should resume what it was doing when the interruption occurred, whether this is playing audio or remaining paused. An app that doesn't have media playback controls should resume playing audio.）

例如，当用户正用 iPhone 听音乐时，突然来了个 VoIP 电话。用户接了电话，希望在通话过程中音乐保持静音。电话打完后，用户希望回放（playback）型程序能自动恢复播放音乐。这是因为用户在接电话前并没有主动暂停音乐，所以希望打完电话后能继续享受歌声。否则，播放器在电话打完后就应该还是暂停状态。

- 不可恢复的冲突。不可恢复的冲突是由以播放声音为核心功能的程序引起的，比如媒体播放器。

在不可恢复冲突结束后，显示了回放控件的程序不要恢复播放，没有回放控件的程序应该恢复到播放状态。

例如，当用户在听 1 号播放器的音乐时被 2 号播放器的声音打断。用户决定听一会儿 2 号播放。再退出 2 号播放器后，用户不会希望 1 号播放器再自动播放，因为他已主动将 2 号播放器视作了主程序。

以下指南帮助你判断应提供什么样的信息，以及如何在中断结束后继续：

确定程序导致中断的类型。你可以在你的声音播放完毕后通过以下两种方式屏蔽你的声音：

- 如果你的程序导致的是可恢复的中断，给你的声音打上 `AVAudioSessionSetActiveFlags_NotifyOthersOnDeactivation` 的标。

- 如果你的程序导致的是不可恢复的中断，就不用打标了。

这样做可以帮助 iOS 恢复被打断的声音播放。

判断中断结束后你是否应该恢复播放。你可以根据程序所播放声音的体验类型来判断。

- 如果你的程序提供了让用户播放或暂停声音的媒体回放控件，就应该在中断结束后检查 `AVAudioSessionInterruptionFlags_ShouldResume` 标签。

如果存在“Should Resume”标签，你应该：

- 如果你的程序在被打断时正播放声音，那就恢复播放
- 如果你的程序在被打断时没有播放声音，那就不要恢复播放
- 如果你的程序没有提供让用户播放或暂停声音的媒体回放控件，就应该在中断结束后恢复刚才播放的声音，不必再去检查“Should Resume”标签是否存在。

例如，播放音轨的游戏应该在中断结束后立即恢复播放。

处理远程媒体控制事件

从 iOS4.0 起，程序可以在用户使用 iOS 媒体控制器或其他附件时接受远程控制事件。这使得无论你的程序是在前台还是后台，都能从界面以外的地方获得指令。

媒体回放程序尤其需要使当地响应这些事件，尤其是当在后台播放音乐时。

为了承担起保护隐私的责任，请遵守以下指南：

只在合理的时候接受远程控制信号。例如，如果你的程序支持阅读内容，搜索信息和听音频，那么只在用户听音频时才接受远程信号。当用户没在听音频时，就要把接收事件的权限释放出来。这使得用户可以在使用你的程序阅读内容时方便地收听、控制其他程序的音频。

即使某个事件对你的程序没有意义，也不要重新定义事件的含义。用户希望 iOS 媒体控制器和附件的事件在所有程序里具有一致的含义。不要处理与你的程序不需要的事件。所有你处理的事件就要符合用户的期望。一旦重新定义事件的含义，用户就晕了，必须要退出你的程序才能脱离困境。

VoiceOver 和附件

VoiceOver 是为帮助盲人、有视力障碍的用户以及有学习障碍的用户而设计的。



为了确保 VoiceOver 用户能使用你的程序，你需要为视图和控件提供一些描述信息。支持 VoiceOver 并不需要你对界面设计方案做任何调整。

只要你依照标准指南使用标准控件，就不需要什么额外工作。对界面的个性化定制越厉害，就需要更多的说明信息，以便 VoiceOver 能精确地描述你的程序。

编辑菜单

你可以在文本视图、web 视图和图片视图里调出编辑菜单，来执行剪切、粘贴和选择的操作。



你可以调整菜单的行为，给用户更多控制内容的操作。例如，你可以：

- 设定哪些标准命令适合当前的环境。
- 在菜单出现前定义好它的位置，以免遮盖重要的内容。
- 当用户双击唤出菜单时，定义好默认被选中的对象。

你不能：

✳ 更改编辑菜单的颜色和形状。

为了保证在你的程序中编辑菜单符合用户期望，你应该：

展示与当前环境相适应的命令。例如，当什么都没选中时，菜单中不应该有“复制”和“剪切”。同样，如果选中了一些东西，菜单里不应该包含“选择”。如果你要定制编辑菜单的外观，要确保展示的内容与当前环境相匹配。

让菜单和界面布局相适应。iOS 在插入点或选择区的上面或下面，具体由空闲空间来定。菜单指向内容，以使用户知道操作的对象。你可以在菜单出现前用代码确定好它的位置，这样可以避免重要的内容被它遮盖。

两种唤出菜单的手势都要支持。虽然触碰并按住是唤出菜单的主要方式，但也可以在阅读文本时用双击文字同时选中文字并唤出菜单。如果你要自定义编辑菜单，一定要支持两种手势。另外，你可以定义用户双击时默认选中哪些对象。

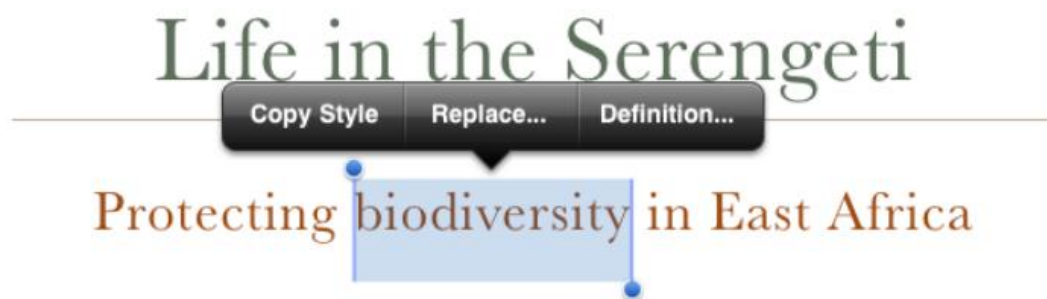
避免在界面上放置编辑菜单里已经有的命令。例如，最好在编辑菜单里完成复制，而非在界面上放置复制按钮。否则用户还需要想一想到底该用哪种。

如果对用户有用的话，可以让静态文本也能被选中。例如，用户会想要复制图片的标题，但不太会想复制 tab 栏的标题或者本屏的标题。

不要让按钮标题变得可以选中。用户选中按钮的标题时会很容易触发按钮。一般而言，与按钮行为相似的元素都不应该被选中。

如果支持“复制”和“粘贴”，也要支持“重做”和“撤销”。用户一改变主意就会很想撤销最近的操作。由于编辑菜单里的命令在执行前无需再次确认，所以应该给用户“重做”和“撤销”的机会。

在 iOS4 以后，你可以在编辑菜单里提供定制的功能。



如果你想把被选中的文字或对象用于与当前环境无关的地方，最好使用操作列表。例如，当用户把选中的内容在网上分享，你最好用操作列表展示一堆 SNS 供用户选择。

如果你想定制编辑菜单，请阅读以下指南：

自定义的菜单项要能够编辑、改变、或者直接操作对用户选中的对象。用户希望编辑菜单上的命令能够在当前环境里直接操作被选中的对象，你定制的命令也应满足这种期望。

把定制的项目列在系统的项目后面。不要把定制命令和系统提供的命令混起来。

保证定制的项目数量合理。不要给用户过多的选择。

定制的菜单命令要简明，精确地概括这条命令的作用。一般而言，项目名称应该是能描述操作的动词。虽然一个大写的词能搞定最好，但非得用词组的时候，要用标题大写样式（四个词以内，大写除冠词、介词、并列连词外的词）。

撤销和重做

晃一晃设备，就能唤出一个对话框，允许用户：

- 撤销刚才输入的词
- 重做刚才撤销的词
- 取消“撤销”

你可以通过定义如下内容，以在更宽的范围内支持撤销

- 用户可以撤销和重做的操作
- 什么时候你的程序会把摇晃视作撤销的命令
- 支持多少步撤销

用简短的话告知用户，他们在撤销什么、重做什么。iOS 在按钮上自动提供“撤销”“重做”（应为会自动补上空格），但你需要提供一两个词，描述一下撤销、重做的是什么。例如，你补上“删除姓名”或“更改地址”，或者直接把标题写成“撤销删除姓名”或“重做更改地址”（“取消”按钮的标题时不能更改或隐藏的）

避免词太长。词太长不好读，太卡。可以用标题大写样式，不要加标点。

不要覆盖摇晃设备的事件。虽然你可以在代码中设置何时将摇晃视作唤出撤销菜单的命令，但如果用户还可以用摇晃作为其他命令的话，这就会让用户迷惑。分析一下程序的交互方式，避免让用户不能预测摇晃后的效果。

除非撤销和重做是很常用的操作，否则不要把系统提供的重做和撤销按钮放出来。记住，摇晃是唤出撤销对话框的首要途径。提供两三种方式反而会让用户迷惑。如果你确信把这些功能放出来很有必要，可以在导航栏里放上系统提供的撤销和重做按钮。更多信息详见“[工具栏和导航栏使用的标准按钮](#)”。

把撤销和重做与当前的工作紧密联系起来，不要和之前的工作联系起来。一般而言，用户希望所做的操作可以立刻生效。

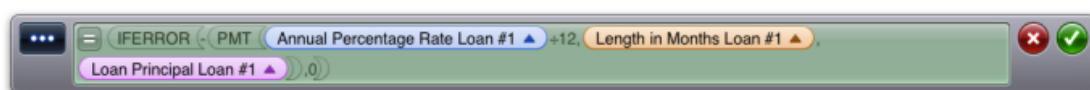
键盘和输入视图

iOS 3.2 及以后的版本可以用定制的键盘取代系统键盘。例如，iPad 版 Numbers 提供了让输入日期、时间更方便的键盘。



如果你要定制输入界面，一定要让它容易理解，而且要看起来可点击。

你也可以定制一些小插件，放置在键盘上方（或自定义位置）。例如，在某些时候，Numbers 会提供让用户帮助快速计算的小插件。



在 iOS4.2 以后的版本里，你可以使用系统提供的标准音来为定制的键盘提供声音反馈。

位置服务

位置服务允许程序确定用户的大致位置，前进方向和设备方向。用户会很喜欢给事物打上位置的标签，或者找到附近的朋友。但是用户也希望在不想分享位置的时候关闭这些服务。

当用户关闭位置服务后，如果新打开的程序还想请求位置数据，就会跳出警告框，提示要想继续使用位置信息必须修改设置。警告框不允许用户在当前程序里修改设置。用户必须到“Settings”里修改偏好。这可以让用户明确知道他在给程序授权位置信息。



为了提供最佳的用户体验，请参照一下指南：

确保用户明白为什么要打开位置服务。当被索要个人信息时，如果用户不明白为什么需要这些信息，就会很疑惑。为了不让用户觉得不舒服，确保只在用户尝试使用明确需要位置信息的功能时弹出警告框。例如，用户可以在没有位置信息时使用 Maps，但是当试图使用定位相关的功能时就会看到警告框。

在弹出警告框前先检查用户的偏好设置。你可以使用核心位置接口来获取这个参数值。获得这个信息可以让你只在必要的时候弹出对话框，或者完全避免弹出它。

只有当没有位置信息就无法实现核心功能的情况下才弹出警告框。这时用户不会觉得烦了。因为他们明白该程序的核心功能必须依赖位置信息。

避免在用户选择需要位置信息的功能前就弹出警告框。这样的话，用户就会疑惑为什么在使用与位置不相干的功能时需要位置信息。

本地和推送提醒

本地和推送提醒允许你在程序没有在前台运行时想用户发送消息。



例如，你可能会想让用户知道：

- 有新消息到达
- 将会发生什么事
- 数据下载好了
- 某些东西的状态变了

本地提醒：程序可以添加排期，由 iOS 系统发布，程序没必要一直在前台运行。例如，日历或者 to-do 程序可以给本地的提醒时间表排期，提醒用户快到的会议或约会。

推送提醒：由程序的远程服务器发送到“苹果推送提醒”服务上，一次性发给所有安装了此程序的设备。例如，用于与他人对战的程序可以一次性更新所有人的动向。

如果本地或推送提醒到达是，你的程序没有运行在前台，你可以通过以下方式获取用户注意：

- 在程序的 Home screen 图标上打上可爱的小绷带标记
- 唤出警告框

你可以在更新绷带标记或者警告框的时候用声音提醒。

如果你的程序正运行在前台，也同样可以收到本地和推送提醒。但是可以用自定义的方式表达给用户。在“settings”里，用户可以将某些或所有程序的推送消息禁用，小绷带、声音或者警告框就不会出现了。但对本地提醒就不能在 settings 里设置了，这要到具体的程序里去设置。

当信息的数量有意义且对时间不敏感的时候可以使用小绷带。绷带是告知用户有多少新项目值得注意的好方法，比如未读消息、待办事项、更新的文档。因为用户只有浏览桌面才会看到这些消息，所以不要拿它来提醒时间紧迫的事。

绷带的外观和位置不能自定义：它是红色的，会出现在图标右上角。



绷带上只有数量，没有文字或标点。

当发布需要用户立刻注意或操作的消息时，可以用警告框。警告框是告知用户新事件或状态改变的好方法。警告框会中断用户的操作流，所以要谨慎使用。



要想让提醒符合用户期望，请遵照以下指南：

保持绷带上的数字实时更新。用户一浏览新信息就要立刻更新绷带上的数字，这样他们就不会觉得刚读完又有新信息来了。

定制警告框上的信息。定制信息将出现在警告框的中央，位于程序名称的下方（程序名称会自动成为警告框的标题）。一个合格的本地或推送消息应该：

- 聚焦于信息，而非用户的操作。避免告知用户应该按哪个键，或描述按某个键的后果。
- 要简短，一两行内放完。如果消息太长，警告框就会被迫翻页。
- 使用句子大写样式，附上适当的标点。可能的话，使用完整的句子。

按钮最好使用定制的标题。警告框上可能会有一或两个按钮。有两个按钮时，关闭按钮会放在左边，操作按钮（action button）会放在右边。如果只有一个按钮，默认显示为 OK。

按操作按钮会关闭警告框并自动登录你的程序。按关闭或者 ok 也会关闭警告框，但不会登录程序。

如果你想为操作按钮使用定制的标题，要让标题明确地描述登录程序后会发生的操作。例如，游戏可能会用“play”作为标题，暗示按下它将会打开程序给用户来玩。要确保标题可以：

- 使用标题大写样式
- 简短，避免在按钮里折行

当设备处于锁定状态时，会显示“滑动以查看”。你定制的按钮标题也可以显示在这里，届时标题会自动小写，取代“查看”。

可以考虑提供登录图片。除了展示已有的登录图片外，你可以在用户通过提醒打开你的程序时提供另一张登录图片。例如，游戏可以显示一张与游戏过程中截图相似的登录图片，而非菜单那一屏。如果不专门为此提供登录图片，iOS 会展示之前的截屏或者其他登录图片。

合适的话，呈现绷带或者警告框时要有伴声。当用户没有看着设备时，声音可以吸引他们的注意。应当只对很重要的信息才配备提醒音。例如，日历可能会在马上要有会议时发出提醒音。或者多人协作任务管理程序会在他人完成了分配的任务时给予提醒。

你可以定制声音，或者使用内置的警告声。如果你要定制声音，确保它简短而专业。不能在有新消息时强迫设备振动，因为这应该听从用户的设置。

第8章 iOS 界面元素使用指南

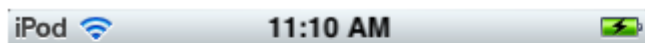
在 iOS 里，UIKit 框架提供了很多你可以在程序中调用的界面元素。当你设计程序界面时，时刻谨记用户希望那些眼熟的视图和控件与内置程序中的行为一致。在你的程序里妥当地使用它们，这会对你有好处的。

栏

状态栏、导航栏、tab 栏以及工具栏在 iOS 中有各自的外观和行为。这些栏不是要求每个程序都必须有。但是，如果使用它，就要用对。这是因为用户在 iOS 设备上对这些栏都很熟悉，包括上面展示的信息以及所具有的功能。

状态栏

状态栏展示于设备和当前环境相关的重要信息。



外观和行为

状态栏总是出现在屏幕顶部，包含网络连接、时间、电量等用户需要的信息。

在 iPhone 上，状态栏的颜色会变。而在 iPad 上，状态栏总是黑色的。

指南

如果你的程序不是游戏或者媒体播放器的话，隐藏状态栏前要考虑清楚。

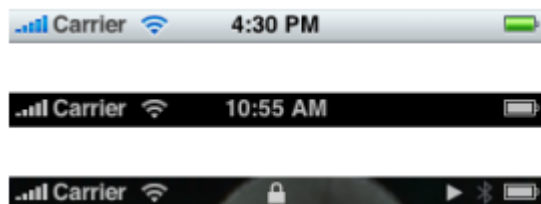
这类应用在运行中会始终隐藏状态栏，你要懂得这种设计的弊端。始终隐藏状态栏意味着用户必须退出你的程序才能看看是否需要给设备充电。

大多数 iPad 程序不需要为了获得额外空间隐藏状态栏，因为状态栏的尺寸也太小了。在 iPad 上，状态栏也表现得很低调，不会和你的程序争抢注意。由于程序界面上边缘的圆角和状态栏的小尺寸，使得状态栏看起来像是设备的边框。

当用户全屏观看媒体时，要把状态栏藏起来。如果你这么做的话，记得在用户轻击屏幕的时候再把状态来呈现出来。除非你有很必要的理由，最好别定义一个手势来召唤状态栏。用户发现不了，发现得了也记不住。

合适的话，展示网络连接情况，这可以提醒用户远程网络连接正在工作。详情请参见“[网络活动指示器](#)”。

在 iPhone 上，定义状态栏的颜色。你可以选择灰色（默认色），透明黑或者半透明黑（alpha=0.5）



要选择和程序风格一致的状态栏。例如，如果导航栏是不透明的，就不要用半透明的状态栏。

在 iPhone 上，设定好状态栏的颜色改变是否用动画来展示。播放动画的话，老状态栏向上滑行直到从屏幕中消失，新的状态栏滑入位。

导航栏

导航栏用于在层级结构的信息中导航，也可以用来管理屏幕信息



欲知更多，详见“[Navigation Controllers](#)”和“[UINavigationController Class Reference](#)。”

外观和行为

导航栏位于屏幕的顶部，上面紧邻状态栏。导航栏通常居中显示当前这一屏的标题。当在层级信息间穿梭时，用户可以触摸导航栏左边的返回按钮回到上一层。用户还可以使用导航栏上与当前内容相匹配的控件来管理屏幕内容。

所有在导航栏里的控件都放在凿出的坑里，这在 iOS 里被称为边框样式（bordered style）。如果你把一个没边框的控件嵌在导航栏里，它会自动变成边框样式。

在 iPhone 里，把屏幕从竖屏转成横屏模式会自动改变导航栏的高度（估计是想在横屏模式留出更多的空间给内容）。在 iPad 上，导航栏的透明度和高度不会随旋转改变。

在 iPhone 上，导航栏的是整屏通栏显示的。在 iPad 上，导航栏可能会嵌在分栏中的某一栏里，不会横贯整屏。

指南

你可以使用导航栏在不同视图间切换，或者在上面放置可以操纵内容的控件。

使用当前视图的标题作为导航栏的标题。当用户浏览到新的层级时，应该做这样两件事：

- 导航栏标题变成新层级的标题。
- 标题左侧出现返回按钮，写着前一级的标题。

确保导航栏上的文字容易懂。系统字体的可读性最强，合适的话你也可以自定义字体。

使用工具栏取代导航栏。如果你需要放置很多控件，或者不需要导航时，就用工具栏吧。

考虑在程序的最顶一级的工具栏放置分段控件（segment control）。如果能够帮助你扁平化信息结构，让用户更容易找到想要的东西，这样做就会很有用。如果你在导航栏上使用分段控件，要给予层次的导航栏返回按钮找好标题。详情请查看 [“分段控件”](#)。

避免用过多的控件填满导航栏，即使看起来好像有足够的空间。除了标题外，导航栏上最多放一个返回按钮，一个操作内容的控件。如果你要在导航栏上放分段控件，就不要放标题或其他控件了。

根据控件的意义选择系统提供的按钮。详情请查看 [“工具栏和导航栏中使用的标准按钮”](#)。如果你决定定制自己的导航栏控件，请到 [“导航栏、工具栏和 tab 栏上用的图标”](#) 查看设计建议。

有必要的时候，定义导航栏的颜色和透明度。

如果你想让导航栏与程序协调起来，可以定制一下导航栏的颜色。如果想让用户更关注栏下的内容，甚至可以把栏弄成半透明的。

如果你要这样定制导航栏，确保它和程序的其外部分外观协调起来。要是使用半透明的工具栏，就不要使用不透明的工具栏。而且，当设备在同一方向上时，不要让不同屏上的导航栏颜色和透明度变来变去的。

避免改变返回按钮的外观和行为。用户依赖标准的返回按钮帮助他们在层级信息中找到回家的路。

不要创建分段的返回按钮（类似于面包屑）



使用分段返回按钮会导致很多问题：

- 分段控件太长，都没空放标题了
- 没法展示某一段的选中态
- 段越多，每一段的可点击区域越小，用户想按某一个不好按
- 当层级很深时，选择层级的哪一部分来展示是个问题

也许你觉得没有分段返回按钮展示的面包屑，用户就会迷路，那说明在你的程序里用户必须进入很深的层级才能获得信息。那么，还是把你的层级扁平化吧。

在 iPhone 上，要考虑到由于设备方向变化导致的导航栏自动改变。确保你定制的导航栏图标适应横屏条件下的窄栏。不要把导航栏的高度写死。

工具栏

工具栏上放着用于操作当前屏幕上物体的控件。



外观和行为

在 iPhone 上，工具栏总在屏幕的底部。但是在 iPad 上它也可能出现在顶部。

工具栏上的控件等宽放置。控件会随着屏上内容的切换而改变，因为内容与用来操纵它的控件是匹配起来的。

在 iPhone 上，横屏切换到竖屏时工具栏高度会自动变小。在 iPad 上，工具栏的高度和透明度是恒定的。

指南

在工具栏上要放那些用户能对当前内容所做的操作，不要用它来切换程序模式（Don't use a toolbar to switch among different modes in an application）。如果你要实现切换的效果，可以用去看“[tab 栏](#)”。

在工具栏上放那些当前情景下用户最常用的功能。你也可以通过在工具栏上放置分段控件，来切换浏览数据的方式，或者程序模式（--！为何与上一段自相矛盾呢？原文：You can also put a segmented control in a toolbar to give people access to different perspectives on your application's data or to different application modes）。欲知更多，详见“[分段控件](#)”。

每个工具栏上的控件至少要保持 44×44 像素的面积。如果控件太密集，用户点击就很困难。

调用系统提供的控件要遵循使用规范。详情请查看“[工具栏和导航栏中使用的标准按钮](#)”。如果你决定定制自己的导航栏控件，请到“[导航栏、工具栏和 tab 栏上用的图标](#)”查看设计建议。

尽量避免在同一个工具栏上混合使用有边框和无边框的控件。使用任何一种都可以，但是不要混着用。

适当的时候，可以定制工具栏的颜色和透明度。如果你想让工具栏和整个程序的外观一致起来，可以去定制颜色。如果要鼓励用户浏览内容，可以把工具栏做成半透明的。如果你用了半透明的工具栏，就不要用不透明的导航栏。设备方向不变时，不要改变在不同屏上使用不同颜色或透明度的工具栏。

在 iPhone 上，要考虑到由设备方向改变引起的工具栏高度变化。确保你定制的工具栏图标与横屏模式下的窄工具条相适应。不要把工具条的高度写死。

Tab 栏

Tab 栏用于切换子任务、视图和模式。



外观和行为

Tab 栏位于屏幕的底部，并且始终可见。Tab 栏上展示图标和文字，同宽，黑底。当用户选中某个 tab 时，这个 tab 的背景色亮起，图片处于高光态。



在 iPhone 上，tab 栏一次只能显示 5 个以内的页签。如果程序有更多的 tab，tab 栏可以展示前四个，第五个放一个“更多”，用列表的方式呈现其余的项目。在 iPad 上，tab 栏可以显示多于 5 个 tab。

Tab 上可以打上小创可贴（红底白字），用以展示与程序相关的信息。

Tab 的透明度和高度不随设备方向改变。

指南

使用 tab 栏来切换对同一组数据的不同浏览方式，或者同一功能的不同子任务。

不要使用 tab 来执行对当前屏幕上元素的操作。如果你要达到这样的目的，可以选用工具栏“[工具栏](#)”。

一般而言，tab 栏是用来管理程序层面信息的。Tab 适合用于主程序界面，因为它可以很好地扁平信息层次，在同一时刻提供进入多个平级信息类的入口。

在 iPad 上，你可能会在分栏或浮出层里下使用 tab 栏，用于切换或过滤内容。但是，在浮出层或分栏底部放置分段控件会更好，因为在视觉上分段控件长的更协调。更多信息请详见“[分段控件](#)”

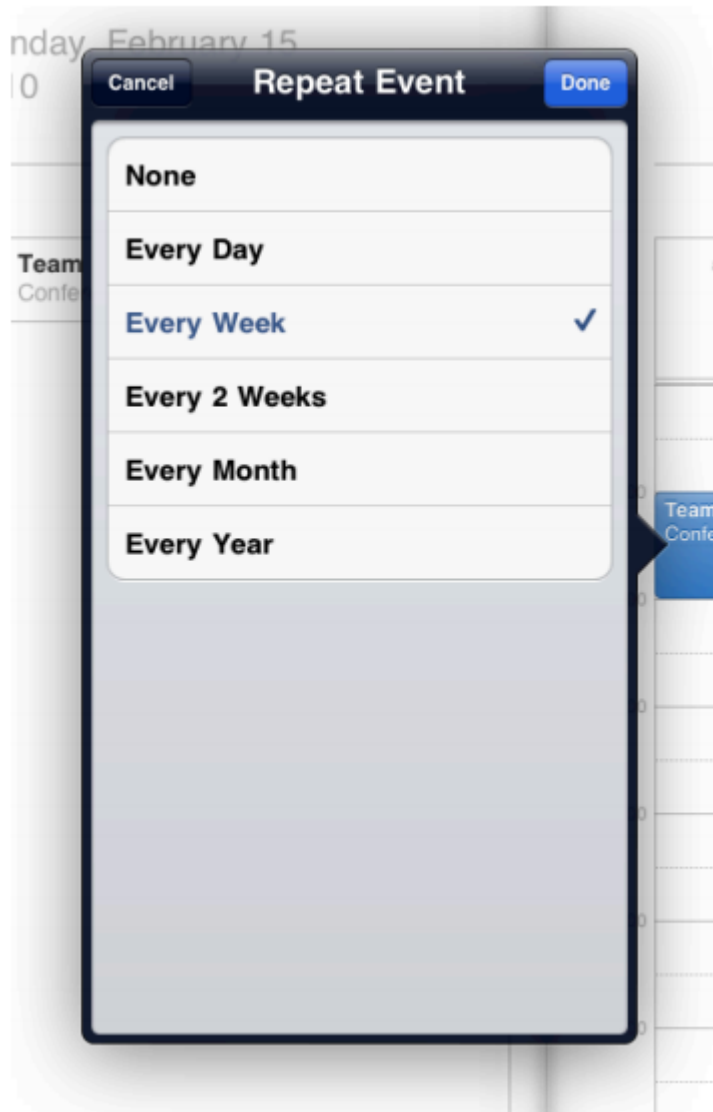
考虑在 tab 上打红色小创可贴，柔和地沟通信息。你可以在 tab 上打上小创可贴，用以显示那个 tab 下的视图或模式有新信息。

调用系统提供的 tab 图标要遵循使用规范。详情请查看“[工具栏和导航栏中使用的标准按钮](#)”。如果你决定定制自己的导航栏控件，请到“[导航栏、工具栏和 tab 栏上用的图标](#)”查看设计建议。

内容视图 (Content Views)

iOS 提供了一些用于定制信息展示方式的视图模板。每一种视图方式都有独特的属性和行为方式，使得它非常适于展示某一类型的信息。

浮出层 (只限 iPad)



浮出层是一种临时的视图方式，可以通过用户点击控件或区域来召唤。

外观和行为

浮出层会悬浮在主屏幕的上面。它还有一个箭头，指向触发它的控件或区域。浮出层上可以展示很多物体和视图，例如：

- 表格、图片、地图、文本、web 或者定制的视图
- 导航栏、工具栏或 tab 栏

- 与当前程序展示的内容相关的控件或物体。

在 iPad 上，操作列表（action sheet）总是以浮出层的形式出现。

指南

你可以这样使用浮出层：

- 与选中或成为焦点的物体相关的信息或者物品列表
- 在竖屏模式下，展示原本横屏模式下的左侧分栏。这样做时，要找一个合适的按钮去召唤它，给按钮合适的名字，把按钮放在屏幕顶部的导航栏或者工具栏上。
- 展示一个操作列表，里面包含于当前屏幕内容紧密相关的一系列选项。

不要提供关闭浮出层的按钮。当浮出层没有继续存在的必要时，就应该自动消失。那么，到哪里才能判断浮出层还有没有利用价值呢？只要参考以下情况：

- 如果浮出层只是提供了一些对主界面有影响的选项或物品，那么，选择一做完，它就应该消失。这种行为与电脑上的“菜单”非常相似。记住，这种行为方式也适用于只显示了操作列表的浮出层：用户一点击列表中的某个按钮，浮出层就关闭。

有时，浮出层在用户做出选择后，也不一定要消失。用户可能想要多选、或者调整当前选择的属性。

包含菜单或 inspector（检视窗口？）应该在用户点击浮出层以外的区域（包括召唤它的按钮）时关闭。在提供餐单的浮出层中，这种动作意味着用户放弃选择，所以不要影响主界面。在展示动作列表的窗口里，这个动作意味着“取消”。

- 如果要在浮出层上执行任务，要展示能完结或取消该任务并同时关闭浮出层的按钮。一般而言，用于执行编辑任务的浮出层上会有“完成”和“取消”。这些按钮提醒用户当前是一个编辑环境，可以选择保留或放弃输入的信息。当用户点击任何一个按钮时，浮出层都应关闭。

如何合理的话，你也可以在用户点击其他区域时不关闭浮出层。若果用户完成的这个任务很重要，这会是个好主意。否则，你应该在用户点击浮层外区域时保存用户输入的信息，就像用户点完成后你会做的那样。

一般而言，在用户点击浮层外区域时保存用户的工作。因为浮层没有明显的解散令，用户可能是不小心碰到了。除非用户点了取消，你才能舍弃他们的工作。

让浮出层的箭头尽量指向唤醒它的元素。这帮助人们记住浮出层是从哪儿冒出来的，与什么任务有关。

确保用户不用看主界面上被遮住的信息就能完成浮出层里的任务。浮出层会遮住它后面的内容，而且浮出层不能被拖到其他地方去。

确保同一时刻屏幕上只有一个浮出层。不应该同时展示多个浮出层（或者定制看起来或行为像是浮出层的东西）。不要同时展示级联的或层叠的浮出层，不要一层套一层。

不要在浮出层上面一层展示模态视图。除非是警告框，否则浮出层上面不要再展示东西。

可以的话，允许咏絮关掉浮出层，然后点一下就打开另一个浮出层。这在基本不同的工具栏按钮分别激活不同的浮出层时尤其有效，因为它不需要用户做过多的操作。

浮出层避免做得太大。太大的浮出层看起来像是把整个屏幕盖住了。它只要和它包含的内容一样大就好，并且指向激活它的地方。

最佳的浮出层宽度应该大于 320 点，小于 600 点。高度没有限制，所以你可以用它展示很长的列表。一般而言，含有任务或操作列表的浮出层最好直接全展示出来，不要让用户翻页。注意，系统可能会把高度和宽度调整到适应屏幕的大小。

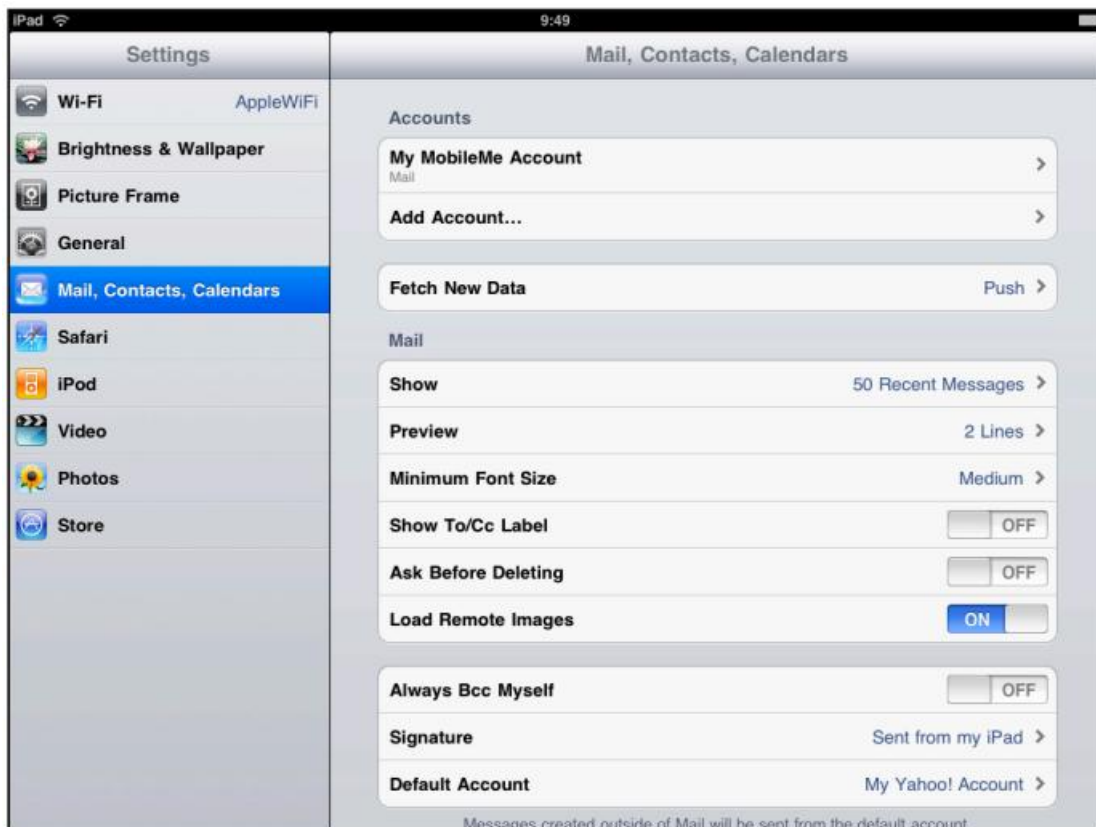
在浮出层里，建议使用标准控件和视图。一般而言使用标准控件和视图的浮出层看起来会很棒，也容易理解。

使用定制背景色或纹理时要留意。确保标准界面元素在你定制的背景上看起来很棒，容易阅读。

合适的话，在展示浮出层时可以动态改变大小。你也许想在展示精简信息和扩展信息时改变浮出层的尺寸。真要改的话，可以用动画。这会非常合适，因为它可以避免让用户觉得好像是出现了一个新的浮出层，把旧的顶掉了。

分栏视图（只限 iPad）

分栏视图要在全屏状态下同时展示并排的两栏。



外观和行为

分栏的左半部分在各种角度下都固定宽度 320 点，用户不能手动调节分栏比例。

两栏都可以拿来展示很多种信息。例如：

- 表格、图片、地图、文本、网页或其他定制的视图
- 导航栏、工具栏和 tab 栏

指南

你可以在分栏视图的左侧展示持久的信息，而在右侧展示子级细节信息。在这种设计模式里，用户在左侧选择项目，右侧展示与此项目相关的信息。

一般而言，一个在横屏下使用分栏视图的程序在设备转成竖屏后，就把左侧分栏的内容移到浮出层里。但你也可以不遵从此模式。如果在你的程序里合理的话，可以始终采用分两栏展示内容的方式。

避免右侧分栏比左边窄。虽然分栏的宽度随你定，但右侧分栏的宽度少于 320 点的话看起来就不太好。

避免两个栏里都有导航栏。都有分栏的话就会让用户很难分辨两栏间的主次从属关系。

一般而言，左侧分栏的选中态要始终展示。这帮助用户理解左侧分栏项目与右侧内容间的关系。这很重要，因为右侧分栏的内容始终在变，但总要保持与左侧分栏选中项目的关系。

表格视图

表格视图在一列里用很多行展示大量内容。

| | |
|-----------------|---|
| G | A |
| Pete Gardener | B |
| | C |
| | D |
| | E |
| Tess Grady | F |
| | G |
| | H |
| | I |
| M.J. Grey | J |
| | K |
| | L |
| | M |
| Jenn Guggenheim | N |
| | O |
| | P |
| | Q |
| Ryan Gunnar | R |
| | S |
| | T |
| H | U |
| Sara Hashimoto | V |
| | W |
| | X |
| | Y |
| Em Hirsch | Z |
| | # |






外观和行为

表格视图在一行行格子里展示信息，可以划分为块或进行归组。用户可以拖动内容，以显示更多行。用户可以点击选中一项，然后用控件增加或删除行，选中多拍，查看某一行的更多信息或者展开另一个表示视图。当用户选中某项目时，表格会短暂地高亮。

如果选中某项导致开始浏览新一屏内容，那么在新一屏滑入位前这一行就会高亮。当用户再回到原来那一屏的时候，原来被选中那一行会再次高光，提醒用户刚才选过什么。

iOS 定义两种表格视图，可以通过外观来区分：

朴素型 (Plain) 表格从屏幕一边开始展示表格，一直到另一边结束。一行里可以有多个内容段，右侧可选择展示一列半透明编号栏。每行开头可以有眉标，尾部可以有角标。默认情况下，每行的背景是白色的。

| | |
|---|---------------------|
|  | Audiobooks |
|  | Compilations |
|  | Composers |
|  | Genres |
|  | Podcasts |
| | |
| | |
| | |
| | |

分组型 (grouped) 表格包含至少一组列表。每一组总是包含至少一项。每一组起始于眉标文字，收尾于脚标文字 (A group can be preceded by header text and followed by footer text. 这一句真不好翻译啊)。默认情况下，分组被放置在蓝色背景上，组内的背景是白色的。分组型表格没有编号。





iOS 提供了一些表格元素，用于拓展表格的功能。除非特别标注，这些元素仅适用于表格视图。

Table 8-1 Table-view elements

| Element | Name | Description |
|---------|--------------------------|---|
| ✓ | Checkmark | Indicates that the row is selected. |
| ➤ | Disclosure indicator | Displays another table associated with the row. |
| ⓘ | Detail disclosure button | Displays additional details about the row in a new view (for information on how to use this element outside of a table, see “Detail Disclosure Button” (page 119)). |
| ☰ | Row reorder | Indicates that the row can be dragged to another location in the table. |
| + | Row insert | Adds a new row to the table. |
| ⊖ | Delete control button | In an editing context, reveals and hides the Delete button for a row. |
| Delete | Delete button | Deletes the row. |

iOS 3 及以后的版本提供了 4 种表格视图样式，为朴素型和分组型表格提供了最常见的表格样式。每一种样式最适合的表达内容有所不同。

默认型。默认型的单元格样式包括默认左对齐的标题，每一行最左边还有可选的图片。

| Default Cell Style | |
|---|-------------------|
|  | Text Label |
|  | Dahlia |
|  | Daisies |

在默认样式的单元格里，文本的外观暗示它代表项目的名称或标题，左对齐也使列表容易浏览。这使得默认样式很适合展示一系列不需要辅助信息就能加以辨别的项目。

副标题型。默认型的单元格样式包括默认左对齐的标题，每一行最左边还有可选的图片。每个标题下面还有副标题。主标题黑色，副标题字号小一点，是灰色的。

| Subtitle Cell Style | |
|---|--|
|  | Text Label Detail text label |
|  | Dahlia This is a dahlia |
|  | Daisies These are daisies |

在副标题型里，字符的显著性暗示它是项目的名称或标题，而副标题外观暗示它包含与主标题相关的辅助信息，左对齐也使列表容易浏览。这种样式适合项目看起来差不多的情况，用户可以利用额外的信息来帮助区分项目。

Value1 这种样式用左对齐的黑色字体显示主标题，用右对齐的小号蓝色字展示副标题。图片不适合这种样式。

| Value 1 Cell Style | |
|--------------------|-------------------|
| Text Label | Detail text label |
| Dahlia | This is a dahlia |
| Daisies | These are daisies |

在这种单元格里，字符的外观暗示它是项目的名称或标题，而副标题的外观暗示它包含与主标题相关的重要信息

左对齐和字体使列表容易浏览，右对齐的副标题将用户的注意吸引到它提供的相关信息上。这种布局尤其适合展示项目当前的状态，可能是该项目对应的子列表中被选中的那一项。

Value2. 这个样式展示右对齐的蓝色子标题，右边紧邻左对齐的黑色主标题。图片不适用本样式。

| Value 2 Cell Style | |
|--------------------|-------------------|
| Text Label | Detail text label |
| Dahlia | This is a dahlia |
| Daisies | These are daisies |

在 Value2 单元格中，右对齐、受限的宽度以及字体暗示这是标题，但左对齐的内容更加重要。

在这种布局下，每一行两部分内容的相接点位置相同。这在两列内容间创建了整齐的一列空隙，用户阅读右侧的主内容会更加轻松。

指南

你可以使用表格视图来高效地展示或多或少的信息。例如以下情景：

- 提供一系列可供用户选择的项目。可以使用勾选标记向用户展示以选中的内容。
当呈现用户点击表格中某一项后出现的一系列选项时，任何一种样式都适用。当呈现用户按下按钮或其他**不在**表格中的元素后出现的列表时，适用朴素型。
- 展示层级信息。朴素型样式尤其适合展示层级信息。列表中的每一项都是通往各种子信息的入口。用户通过在一系列列表中的选择来浏览层级信息。这个图标 ➤ 暗示用户点击该项后还会有子列表。
- 呈现概念上可归组的信息。两种表格样式都允许在各段的分割处添加段眉和段脚，以示分组。分组性表格的圆角可以很清晰地分割各组信息，即使在快速滚动时效果也很好。
- 展示有编号栏的信息。朴素型样式支持编号，可以让用户快速找到需要的信息。编号通常是贴在屏幕右边缘的一栏字符，通常是按字母表排序的字母。用户触摸（或拖拽）编号，展示相关的区域。

如果你要用编号栏，避免在屏幕右侧边缘展示信息（比如这种图标 ➤），因为会与编号栏冲突。

在用户选择列表项的时候提供反馈。 用户希望在点击列表项的时候它能高光一下。点击后用户期望响应的动作会立即执行：出现一个屏新东西或者项目被勾选。

有时，即使与被选中项目相关的辅助细节或控件已经在同一屏出现了，被选中项还处于高光态。但是，我们不鼓励这样做。因为一系列项目，加上选中项，加上相关的细节或控件，会乱成一锅粥。

用动画展示用户对列表项的改变。这可以提供反馈，增强用户直接控制的感觉。在“设置”中，关闭“自动调整时间”后，列表组会慢慢展开，展示设置“时间和日期”的区域。

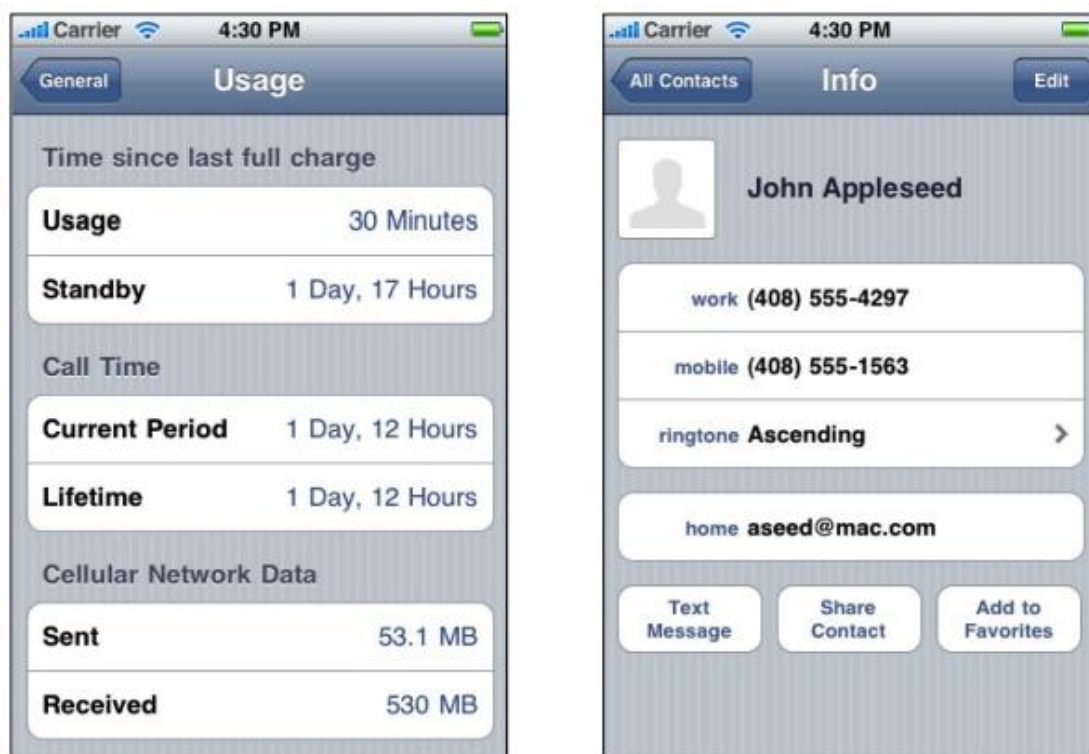
如果表格内容很复杂，避免等到所有信息都载入后再展示。应该立刻展示文本，然后等复杂的信息完备后再展示它们。这种技术可以立刻给用户一些有用的信息，增加程序的反馈性。

考虑在等待载入信息时展示原有信息。如果你的程序展示的信息不太常变，这种技术可以让用户立刻看到一些有用的信息。如果信息经常变动，那就算了。在决定这样做之前，调查一下信息更新的频率以及很依赖新信息的用户所占的比重。

如果数据很复杂或载入很慢，要提供信号，暗示处理进行中。如果表格包含的信息很复杂，可能无法立刻提供有用的信息。这种情况下，不要只展示空格子。因为这会暗示程序挂了。你应该在屏幕中心放一个活动指示器，配上“载入中”这样的字符。这会给用户反馈，让他们知道程序还在继续努力。

朴素型表格的高度不能为适应内容而变化。分组型表格的行高是可以变得，但是在朴素型表格里就看起来很乱。

Value1 和 Value2 的表格很适用于分组型。虽然几种表格样式你可以随便用，但 value1 和 value2 与分组型表格最搭配。例如，“设置”用了 value1，iPhone 版 Contacts 用了 value2。



尽量保持标题简洁，避免被截断。单词和短语被截断后很难阅读和理解。在各种表格样式中，文本截断都是自动的。但这可能因单元格样式和截断位置的不同而引起各种问题。

- 在默认样式中，短的文本标签最好。如果截断不可避免，尽量把重要的词放在前面。
- 在副标题型样式中，短的文本标签也是最好。如果截断不可避免，尽量把重要的词放在前面。如果副标题的文字被截断了，用户可能不会太介意，因为这里的信息是辅助主标题的。
- 在 Value1 样式中，自动截断很难避免。因为要在同一行展示两节信息。但截断也是有必要的，否则就没办法在两节信息间留下分隔空隙了。
- 在 Value2 样式中，截断的字段会破坏两节文字间整齐的空隙。当行与行之间的空隙不一致时，用户会很难浏览右侧的细节信息。

可以通过增加行高来容纳更多的文字，以避免截断。但这会导致如下问题：

- 你需要系统地检查一遍文字是否能被栏充分地包起来。竖屏模式和横屏模式都要检查，因为表格宽度也会有影响。
- 不要在一个方向上是包好的，而在另一个方向上不是
- 不论采用什么表格样式，自适应的行高都会影响表格的整体展示效果。

如果你想用非常规的方式展示表格，最好不要和标准样式差别太大。

文本视图

文本视图可以展示多行文本。

外观和行为

文本视图是一个高度任意的圆角矩形。如果内容过多，超过了便捷显示范围，文本视图会提供拖动功能。

如果文本视图支持用户编辑，当用户在矩形内点击一下后，键盘就会出现。键盘的布局和输入方式由用户的语言设置决定。当用户触摸标有“?.123”的按键时，键盘会变成用于输入数字和标点符号的类型。

指南

你可以对文字的字体、颜色和对齐方式进行设置，但必须要应用到整块文字。也就是说，不能只设置一部分文字的字体。默认的字体系是系统字体，默认的颜色是黑色，因为这样的可读性最高。默认的对齐方式是左对齐。

如果你必须自定义各种颜色和字体，可以使用 web 视图取代文字视图，用 html 定义文字的样式。

你也可以根据需要用户输入的内容类型定义不同的键盘样式。欲知更多，详见“[文本框 Text Field](#)” (p109)。

Web 视图

Web 视图是可以用来展示丰富的 HTML 文本的区域。



外观和行为

除了展示网页信息外，web 视图还会对网页内容进行一些自动化处理，比如将电话号码自动显示为电话超链接。

指南

如果你有网页或者网页 app，可以考虑用 web 视图，然后再给它加一个 app 的壳。

避免使用 web 视图创建一个外观和行为与迷你网页浏览器相似的程序。 用户希望使用 iOS 上的 Safari 来浏览网页内容，所以没必要在你的程序里做重复工作。

警告框、操作列表和模态视图

这三种视图都是当需要获得用户注意、或需要展示额外的信息和功能时临时出现的。当任意一种视图出现时，用户都不能再对主界面进行操作了。

警告框



警告框用于向用户展示对使用程序有重要影响的信息。

外观和行为

警告框浮现在程序中央，覆盖在主程序之上。警告框的外观强调了这样一个事实，它的到来是由于程序或设备的状态发生了重要变动，并不一定是由用户最近的操作导致。用户必须刻意地关掉警告框才能继续进行操作。

警告框通常至少有一个按钮，用户点击后即可关闭窗口。警告框上也总会有标题，并展示额外的辅助信息。警告框的背景色是系统定义的，无法定制。

指南

由于警告框不经常出现，所以一旦出现用户就会谨慎对待。切记限制警告框的数量，确保每一次展示的都是关键、有用的信息。

一般而言，以下情况就木有使用警告框的必要了：

- 仅仅是为了让一些信息显眼点，尤其是那些与程序的常用功能相关的信息。
相反，你应该用与程序样式和谐的方式让重要的信息吸引眼球。
- 向用户提示正常进行中程序的进度。
作为替代，考虑使用“[进度指示器](#)”或者“[活动指示器](#)”向用户提供与进度相关的反馈。
- 对用户触发的操作讨要“确认”
为了要求用户再次确认刚才触发的操作，即使是删除联系人等有潜在风险的操作，也只能使用操作列表（action sheet）
- 不要提示那些用户无能为力的错误。

虽然有必要使用警告框对用户无法弥补的错误进行提示,但最高把这些消息嵌入到界面里。例如,与其每次服务中断时都提示用户一下,不如在最后一次链接成功时提醒用户。

你可以自由定制警告框的标题和内容,按钮的数量,按钮上的文案。但警告框宽度、背景皮肤以及文字的对齐方式不能更改。

在阅读警告文案设计指南的时候,最好明白以下定义:

标题大写样式 (Title-style capitalization): 每一个字母都是大写的,除了冠词、协调连接词和少于四个字符的介词。

句子大写样式 (Sentence-style capitalization): 第一个字母大写,其他字母小写,除非是特定的名词或形容词。

警告文案要简洁地描述当前情景、告知用户他们可以做什么。理想条件下,你的文案要能让用户明白为什么会出现警告,并知道按哪一个按钮是合适的。

标题简短,要在一行内显示。长标题影响用户阅读速度,还有可能要折行显示。



避免只显示一个没有意义的单词。比如“错误”、“警告”

可以的话,使用句子片段。一个简短的,有信息的陈述要比一个完整的句子更容易理解。

表述负面影响时不要扭扭捏捏。用户知道警告框会告诉他们出问题了,提醒危险情景。所以直接把问题讲出来要比和谐的语气有用的多。

尽可能避免使用“你”“你的”“我”“我的”。有时,直接定位用户会带来误会,甚至可以被理解为是一种侮辱。

如果标题包含一个问句,那就使用句子大写样式,以问号结尾。一般而言,如果能避免附加辅助信息,就使用问句作为标题吧。

如果标题包含两个以上的句子,使用句子大写样式,并使用合适的标点结尾。通常是不应该用两个句子做标题的,但如果这样可以省掉辅助句子,也是可以考虑的。

如果要提供辅助信息,就写一个简短的完整句子。使用句子大写样式,并使用合适的标点结尾。



避免消息内容过长。可能的话，保持消息足够的短，一两行足以。如果消息过长的话，就需要滚动翻页，这可不是什么好体验。



避免解释“按某个键后有什么效果”。比如“按‘查看’可以浏览更多信息”。理想状态下，清晰的警告文案和富有逻辑的按钮标签足以给人们足够的信息明白当前的情况和合适的选择。然而，如果你觉得必须提供细致至极的解释，最好遵循这些原则：

- 要用“按”，而非“触摸”“点击”或“选择”，来描述选择某个键
- 不要用引号把按钮标题框起来，但要保证它是首字母大写的。

一定要在两种设备方向上都检测一下警告框。因为横屏模式下，警告框的高会偏小，会和竖屏模式下略有不同。建议你优化一下警告框，以便在两种方向上看起来都很棒。

尽量采用有两个按钮的警告框。两键警告框通常是最有用的，因为方便用户在两种选择间做决策。只有一个按键的警告框不是好警告框，因为这样对于用户没什么意义，不能让用户掌控情景。包含多于三个选项的警告框太过复杂，因尽量避免。如果需要向用户展示多多于两个选择，应该考虑使用操作列表。

认真选择警告框上按钮的颜色。警告框的颜色或深或浅。在有两个按钮的警告框上，左边的按钮总是设色的，右边的总是浅色的。在一个按钮的警告框上，按钮总是浅色的。

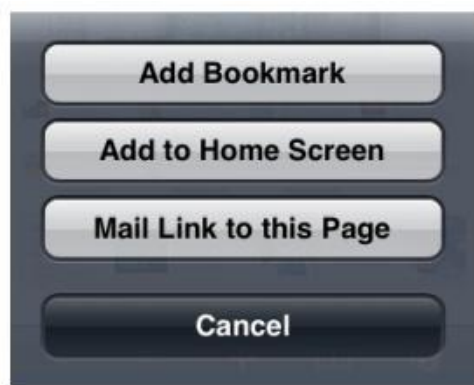
- 在包含危险操作的双按钮警告框上，取消应该放在右侧，用浅色按钮表示。
- 如果双按钮警告框是用于启动某个用户期待的功能，那么取消功能应该放在左侧，用深色按钮表示。

给警告框的按钮赋以简短而富有逻辑的标题。最好的标题包含一到两个单词，与警告框所处的环境相适应。以下指南可以作为准备按钮文案时的参考。

- 使用标题大写样式，不用在结尾加标点
- 优先考虑动词或动词短语，比如“取消”“允许”“恢复”和“忽略”等于警告框直接相关的词汇
- 如果没有更合适的词，那么在表达“同意”时优先考虑“OK”。不要用“是（YES）”“否（NO）”
- 避免使用你，你的，我，我的。按钮标题用了这些词汇后会变得让人费解。

操作列表

操作列表展示一组与用户触发的任务相关的选项。



外观和行为

iPad 上的操作列表与 iPhone 上的有些不同。iPhone 上的操作表总是从屏幕底部出现，悬浮在程序上方。操作列表的两端与屏幕的两条边相接，将其与程序以及最近的操作建立紧密联系。

在 iPad 上，操作列表总是在浮出层中显示，从不与屏幕同宽。操作列表可以引发浮出层出现，也可以出现在一个已有的浮出层里。这两种情况下，在用户的操作以及操作列表的出现间都有强烈的视觉联系。

操作列表始终包括至少两个按钮，让用户选择怎样完成任务。当用户按下某个键后，操作列表消失。操作列表没有标题或者解释性的文字，因为它是在用户的某个操作后立即出现的（这足以说明它的身份）。

指南

这样使用操作列表：

- 提供多种完成任务的方式供用户选择。操作列表允许用户提供与当前情景相符的选择，而不用把这些选择放在一个固定的地方。
- 在完成有潜在危险的任务前获得确认。操作列表的出现提醒用户接下来将面临的危险。这种提醒方式尤为重要，因为用户很可能不小心就触碰到哪里。

在 iPhone 上，将操作列表的背景与导航栏和工具栏协调起来。如果你的程序使用黑色的导航栏和工具栏，那就使用半透明黑色背景。如果你的程序使用默认蓝色的导航栏和工具栏，那就使用默认蓝色背景。

你程序中所有的操作列表应该具有一致的背景色。

在 iPhone 上，提供取消按钮。这种用户能够轻易且安全地离开任务。把取消按钮放在最下面，鼓励用户在做出选择前通读操作列表。默认情况下，取消按钮的颜色与操作列表的背景色一致。

在 iPad 上，选择是否用动画的方式展示操作列表。

- 用没有动画的方式向用户在浮出层外触发的任务提供选择。没有动画的话，操作列表和浮出层同时出现。用这种方式展现操作列表的话，浮出层的箭头应指向用户触发任务的控件或区域。

不要放取消按钮，因为用户可以触碰浮出层外的区域来关闭操作列表，不用选择任何选项。

- 用有动画的方式向用户在浮出层外触发的任务提供选择。有动画时，操作列表从已有的浮出层里滑出来。

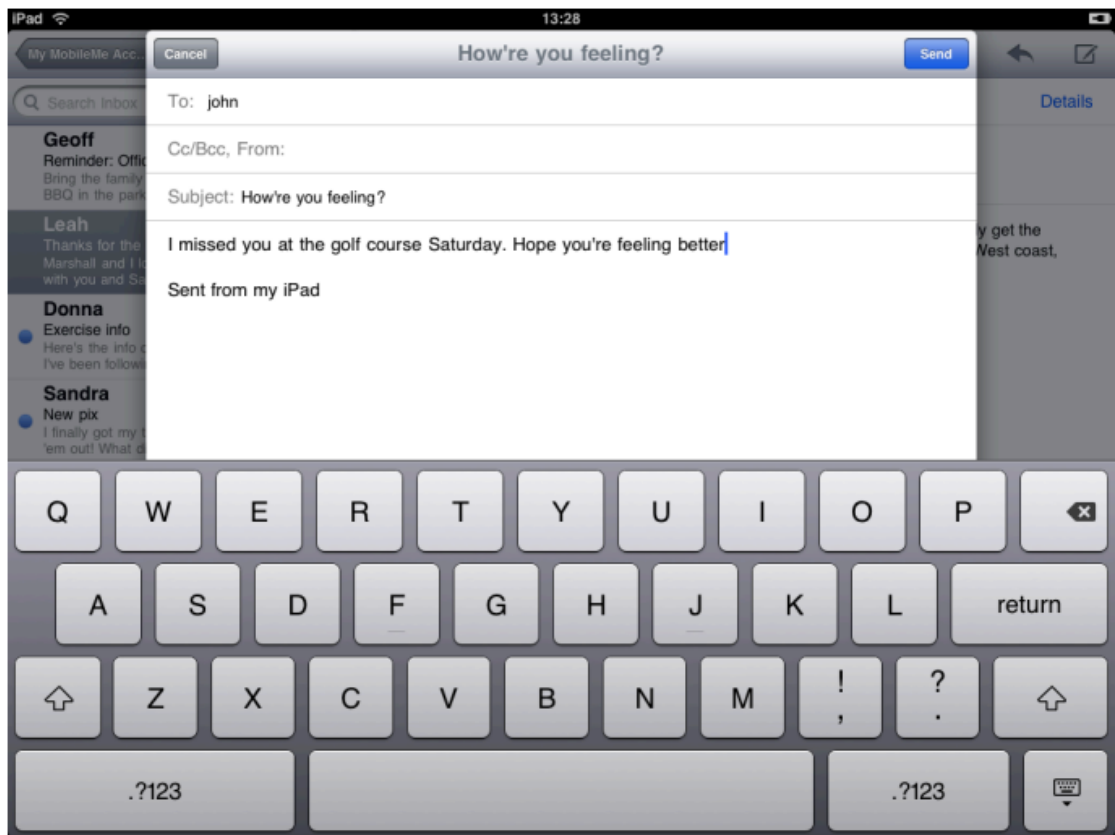
有动画的操作列表应该包含取消按钮，应为用户需要在不关闭浮出层的前提下就关掉操作列表。

无论哪种设备，有潜在危险的按钮要用红色背景。在操作列表顶端展示一个红色按钮，因为离顶部越近越能吸引注意。在 iPhone 上，破坏性的按钮离操作列表底部越远，用户就越不可能在找 Home 键的时候误按到它。

避免操作列表太长，需要翻滚。如果你在操作列表中的按钮过多，它就会需要翻滚。这对于用户来说十分讨厌。因为他们必须花费时间仔细考虑没有选择。而且，翻滚列表又不误按某个按钮是件难事。

模态视图

模态视图用于在当前任务或工作流中提供独立封装的功能。



外观和行为

模态视图会把整个程序屏幕折起来，让用户知觉到一个独立、临时的状态，供他们完成某些任务。

合适的话，模式视图可以拿来呈现文字、以及用于执行某个任务的控件。在模态视图上通常会有完成任务关闭对话框的按钮，或者有个放弃任何改动的“取消”按钮。

指南

当需要在主程序上给用户提供一个子任务的独立空间时，可以考虑使用模态视图。有些子任务需要很多步骤，而且平时没必要把这些控件一直放在屏幕上，这种情况最适合采用模态视图。

在 iPad 上，选择与当前任务以及程序的视觉样式相匹配的模态视图风格。你可以选用：

- **全屏 (full screen)**。覆盖整个屏幕。这种样式适用于那些非常复杂，用户可以在模态的封闭环境中完成的任务。例如：iPod 就用这种形式来完成创建 Genius 播放列表的子任务。
- **页表 (page sheet)**。固定宽度 768 像素，与当前屏幕同高。竖屏模式下，这种模态也能覆盖整个屏幕。横屏模式下，页表没覆盖的屏幕区域蒙上半透明的黑色层，避免用户去操作它。例如，Mail 使用这种方式来展示邮件编辑任务。

- **形式表 (form sheet)**。固定尺寸 540×620 像素，相对屏幕居中对齐。其他没有覆盖的屏幕区域蒙上半透明的黑色层，避免用户去点击。当横屏模式下键盘可见时，形式表向上移动到状态栏下面。这有利于从用户获得信息。
- **相机行事 (current context)**。和父级视图的尺寸一致。这种样式很适合在分栏视图、浮出层等非全屏视图里面展示模态视图。

在 iPad 上，不要在浮出层上面再蒙一层模态视图。不应该在浮出层上再放任何东西，但警告框是个例外。有时，你需要在用户对浮出层有所操作后展示浮出层。那就先把浮出层关掉，再展示模态视图。

在 iPhone 上，让浮出层的外观与主程序一致起来。例如，浮出层上经常会有导航来，上面有助于完成或取消任务的按钮。这种情况下，导航栏的皮肤应该和主程序的导航栏一致起来。

在两种设备上，都应尽可能的在任务界面上展示它的标题。你也可以在视图的其他区域呈现一些文字，完整地介绍当前的任务，提供一些指南。例如，iPhone 版 Message 在用户想要撰写短信时会展现模态视图。这个视图有与主程序相同的导航栏皮肤，标题上写的是“新信息”。



对于两种设备，选择合适的转场效果来展示模态视图。使用与主程序相符的样式，加强用户对“出现了模态视图，场景发生了临时改变”的知觉。可以从如下几种样式进行选择。

垂直：模态视图从屏幕底部滑出来，消失的时候反方向滑走。（默认样式）

翻转：当前的主程序界面从右向左翻转，展现出模态视图。模态视图看起来像是主任务界面的背面。当遣散模态视图时，它水平地总左向右翻转，展现出原有的主程序，

卷边：卷边看来与 iPhone 上的 Maps 页面卷曲效果相仿。当前页面的一角卷曲，展示其下方的模态视图。当用户离开模态视图时，卷曲的部分舒展开来，恢复原状。当你要展示的模式视图不是很大，也不需要太多交互时，可以考虑这种方式。比如用于设置属性选项。

记住，用卷边展示的模式视图不能再用于展示其他的模式视图。

如果你决定改变程序中模式视图的转场效果，避免太过花哨。用户会觉察到其中的差异，并任务这可能代表点什么。因此，最好使用用户容易发现且记住的模式，要逻辑清晰且具有一致性。

控件

控件是用户用于完成操作、浏览信息的界面元素。iOS 提供了大量控件供您使用。

活动指示器

活动指示器暗示任务或过程正在进行中。



外观和行为

当任务在进行中时，活动指示器出现。任务完成后立即消失。用户不与活动指示器交互。默认条件下，活动指示器是白色的。

指南

在工具栏或主视图上展示活动指示器，说明活动正在进行中，但无法展示何时完成。

不要展示静止的活动指示器，因为用户会觉得进程卡死了。

有必要提醒用户进程没有卡死的时候再展示进程指示器，而非展示进程何时完成。

合适的话，调整指示器的尺寸和颜色。

日期和时间拾取器

日期和时间拾取器用于选择时间的各个组成，比如小时、分钟、天、年。



外观和行为

日期和时间拾取器最多可以展示四个独立的轮子，每一个展示一类值，比如月、小时。用户拖拉每个轮子，直到透明的选择栏下出现想要的值。每个轮子上最终的值组成了拾取器的值。

拾取器的尺寸与 iPhone 键盘的尺寸一致。

日期和时间拾取器有四种状态，每一种展示不同数量的轮子，用于选择不同的值。

- **日期和时间。**展示日期，小时和分钟，上午/下午的轮子可选。这是默认模式。
- **时间。**时间模式展示小时和分钟，上午/下午的轮子可选。
- **日期。**日期模式展示月份、天和年。
- **倒计时。**倒计时展示小时和分钟。你可以设置倒计时的总长度，最多 23 小时 59 分钟。

指南

使用日期和时间拾取器让用户对包含多段内容的时间进行设置，比如日、月、年。日期和时间拾取器很容易使用，因为每一部分的取值范围都很小，用户也知道会出现什么。

合理的话，改变分钟轮的步长。默认情况下，分钟轮展示 60 个值（0–59）。如果用户对时间精度要求不高，可以把分钟轮的步长设置的更大些，最高可达 60。例如，对时间精度要求是“刻”，那就展示 0，15，30，45。

在 iPad 上，只在浮出层里展示日期和时间拾取器。iPad 不适合全屏展示日期和时间拾取器。

详情展开按钮

详情展开按钮用于展示与某个物体相关的详情或功能。

外观和行为

用户按详情展示按钮，展示与某物体相关的额外信息和功能。这些额外细节或功能会展示在一个独立的视图里。

当详情展示按钮出现在表格视图的“行”里时，按在行的其他位置上不会激活细节展示按钮，这只会选中行或者触发程序自定义的行为。

指南

通常，详情展示按钮会用在表格视图里，用以引导用户查看更多与某项目相关的细节或功能。当然，你也可以在其他视图模式使用它。

信息按钮

信息按钮展示程序的配置，通常是展示在屏幕的背面。



外观和行为

iOS 有两种信息按钮：深色 i 浅色背景和浅色 i 深色背景。用户按到它时，信息按钮会自动发光闪一下，同时立刻有所响应。比如翻转屏幕展示背面。

指南

使用信息按钮可以展示配置详情或者选项。你可以使用与界面风格最相符的信息按钮样式。

在 iPhone 上，使用信息按钮翻转屏幕，展示更多信息。通常，屏幕的背面展示那些不需要呈现在主界面上的配置选项。

在 iPad 上，避免使用信息按钮翻转整个屏幕。相反，你可以使用信息按钮向人们展示他们可以进入包含更多信息的扩展视图。

标签

标签用于展示各种数量的静态文字。

Enter your passcode

外观和行为

标签用于展示各种数量的静态文字。用户不与标签进行交互，但可以复制文本内容。

指南

你可以使用标签命名或描述界面的某一部分或者提供短消息给用户。标签最好用来展示少量文本。

尽量让你的标签清晰可读。不要为了梦幻字体或炫目的色彩牺牲文字的清晰度。

网络活动指示器

网络活动指示器出现在状态栏上，暗示网络活动正在进行。



外观和行为

当有数据传输时，网络活动指示器出现在状态栏上。当网络活动停止后它就消失。用户不与网络活动指示器交互。

指南

当你的程序调用网络数据的时间稍长时，就应该展示网络活动指示器向用户反馈。如果数据传输很快就完成，就不用展示它了，因为很可能用户还没发现它就消失了。

页码指示器

页码指示器可显示共有多少页视图，当前展示的是第几页。



外观和行为

每一页视图在页码指示器里都用一个圆点展示。圆点的顺序与视图的顺序一致。发光的圆点代表当前打开的视图。用户可以按下发光点左右的点，浏览前一页或后一页。

点的间距不能压缩。竖屏视图模式下总共可以容纳 20 个点。如果你想放置更多的点，多余的点会被裁切掉。

指南

当需要展示一系列同级别的视图时可以使用页码指示器。如果要展示的视图间存在层级关系，就不要用它了，因为页码指示器不能帮助用户记录步骤和路径。

将页码指示器水平居中放置在屏幕底部。这样它既可以总摆在外面，又不会碍眼。不要展示过多的点。

在 iPad 上，考虑在同一屏上展示所有内容。iPad 的大屏幕不适于展示平级的视图，所以对页码指示器的依赖也小一些。

拾取器

拾取器会展示一系列备选值。



外观和行为

日期和时间拾取器起源于拾取器。当使用日期和时间拾取器的时候，用户拨动转盘，直到想要的值出现。在 iPhone 上拾取器的尺寸与键盘一致。

指南

当需要在的一组值中作出选择时，拾取器用起来更简单。当用户对整组值的内容都有所了解时，拾取器会更加适合。这是因为在任意时刻都有很多值是隐藏起来的。如果用户对可选的值并不了解，拾取器就不适合了。

如果需要展示非常大量的值，还是用表格吧，别用拾取器。表格的高度更大，滚动翻页也会更方便。

在 iPad 上，只使用浮出层展示拾取器。全屏展示拾取器不怎么合适吧。

进度指示器

进度指示器用于展示那些可预测完成度（时间、量）的任务或过程的完成情况。



外观和行为

iOS 提供两种进度指示器。两种样式非常相似，但高度有所差别。

默认样式：白色，适合用在程序的主内容区。

栏样式：比默认的细，适合用在工具栏上。也是白色。

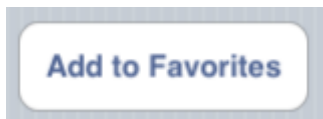
任务或过程进行中时。进度指示器的槽从左向右被逐渐填充。在任何时候，已填充和未填充部分间的比例都能告知用户任务或过程多久能完成。用户不与进度指示器交互。

指南

对可以预测总长度的任务或过程应当提供进度指示器，向用户反馈当前进度，尤其是当用户急切地想知道任务大概还要用多少时间。当进度指示器出现时，用户知道任务还在进行中，这样就能判断是继续等待还是终止进程。

圆角矩形按钮

按钮用于执行特定的操作。



外观和行为

按钮的圆角弧度与分组型表格的外角弧度一致。默认情况下，按钮的背景色是白色。

指南

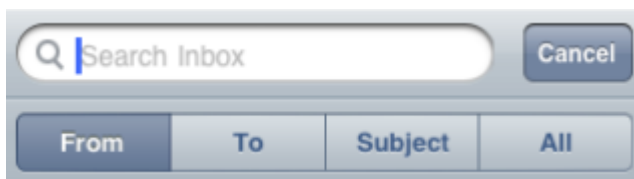
使用圆角矩形按钮可以触发操作。在命名按钮的时候，可以参考以下方式：

- 使用标题大写样式：除冠词、连词、介词外的词都大写，最多四个词
- 避免标题太长：太长的词读起来太卡，用户不容易理解。

你可以定义圆角矩形按钮内展示的标题或图片，也可以定义被按下后如何高亮反馈、标题的外观如何改变。

范围栏（Scope Bar）

范围栏可以允许用户定义搜索的范围，只有在和搜索栏配合时才可以使用。



外观和行为

当搜索栏出现的时候，范围栏紧挨着出现在它下面，除非你使用的是搜索显示控制器（search display controller）。当使用搜索显示控制器的时候，横屏视图模式下范围栏会出现搜索框的右侧。

用户可以点击范围栏上的按钮，切换搜索范围。其外观会和搜索框的设置一致。

指南

当对用户想搜索的范围有明确或典型的分类时，使用范围栏会很有帮助。例如：用户常常想只在邮件中的某一类中进行搜索。

搜索栏

搜索栏可以从用户获得文本，用做筛选的关键字。



外观和行为

搜索栏看起来像圆角的文本框。默认情况下，搜索按钮放在左侧。当用户点击搜索栏，键盘自动出现；当用户输入完毕后，输入的文本就按系统定义的那样处理。

另外，搜索栏还有一些可选的元素：

- **占位符文本。**可以用它来描述控件的作用（例如“搜索”）或者提醒用户是在哪里搜索（例如“Youtube”“Google”）。

- **书签按钮。**这个按钮可以为用户提供便捷的信息输入方式。例如，在 Maps 搜索模式下，用户可以用书签按钮访问标记的地点、最近的搜索和联系人。

只有当文本框里不存在用户输入的或占位符以外的文字时，书签按钮才会出现。因为有了用户输入的文字后，这个位置会放一个清空按钮。

- **清空按钮。**大多数搜索栏包含清空按钮，允许用户点一下就擦除搜索栏中的内容。

当搜索栏 任何非占位符的文字时，清空按钮就会出现。如果没有用户提供的或者非占位符，这个按钮就隐藏起来。因为没必要清空搜索栏的必要。

- **描述性标题。**它会出现在搜索栏上面。例如，它可以是一小段用于提供指引或介绍上下文的短语。

指南

用搜索栏来实现搜索功能，不要用文本框。

你可以在几种标准配色里选取适当的颜色，对搜索框进行自定义：

- 蓝色（与工具栏和导航栏的默认配色一致）。
- 黑色。

分段控件



分段控件是一条分割成多段的线，每一段都像是按钮，可以激活一种视图方式。

外观和行为

分段控件的长度由分段的数量决定，高度是固定的。每一段的宽度依比例而定，取决于分段的总数。当用户单击分段，可使其变成选中态。

指南

使用分段控件提供紧密相连又互斥的选项。

确保每一个选项都可以轻松触摸。为了将每一段尺寸维持在 44×44 像素以上，你需要限制段的数目。在 iPhone 上，分段控件至多可以分成 5 段。

尽可能让每一段的标题一致。因为所有段的宽度都是相同的，如果段上的文字标题长度、风格等不一致，会很难看。

避免同时使用文字和图标。分段控件上面可以放文字，也可以放图标，但只能选用一种。

滚动条 (Slider)

滚动条帮助用户在容许的范围内调整值或进程。



外观和行为

滚动条包含滑轨和滑块，以及可选的图片，用于传达左右两端各代表什么。当用户拖拽滑块，值会实时地连续变化。

指南

用户可以使用滚动条精准地控制值，或者操控当前的进度。

合适的话，可以考虑自定义外观。例如，你可以这样做：

- 水平或者竖直地放置它。
- 自定义宽度，以适应程序。
- 定义滑块的外观，以使用户迅速区分滑块是否可用。

- 为滑轨两端添加自定义的图片，帮助用户理解滑轨的用途。

通常情况下，这些图片表示了最大值和最小值的意思。一个用来控制字体尺寸的滚动条可以在左侧放一个很小的字，在右侧放一个很大的字

- 为滑块在各个位置、控件的各种状态定制不同导轨的外观。

。

切换器 (Switch)

切换器用于切换两种互斥的选择或状态。



外观和行为

切换器展示当前的激活状态，用户滑动空间可以切换状态。当然，按也可以。

指南

在表格视图中展示两种简单、互斥的选项，比如是/否，开/关。所选的两个值要让用户可以预测，这样用户能知道切换后的效果。

你可以使用切换控件改变其他控件的状态。根据用户的选择，新的表单项可能会出现或消失，激活或失活。

文本框 (Text Field)

文本框用于接受一行用户的输入。

外观和行为

文本框有固定的高度。用户按一下文本框后，会出现键盘。当用户按下回车键，文本框会按照程序预设的方式处理输入的字符。

指南

使用文本框来从用户获得少量信息。在用户决定使用文本框前，想想是否有别的控件可以让输入变得简单，比如拾取器或者列表。

可以自定义文本框，帮助用户理解如何使用它。例如，你可以在文本框某一侧放上定制的图片，或者添加系统提供的按钮（比如书签按钮）。一般而言，可以把提示放在文本框左半部，把附加的功能放在右半部。

在右端放置清空按钮。当清空按钮出现时，单击它可以清空文本框的内容。

在文本框里放置提示语，帮助用户理解意图。比如“姓名”“地址”。如果没有其他的文字可放，就可以在这里放置提示语做占位符。

根据要输入的内容选择合适的键盘样式。键盘是主要的输入手段，随着用户的语言而变。例如，你也许想让输入 URL、PIN 或电话号码变简单。iOS 提供几种不同的键盘，每一种都是为输入特定的内容而优化。

系统提供的按钮和图标

为了提高一致性，iOS 提供很多标准控件就用于导航栏、工具栏和 tab 栏。

无论你在做什么类型的程序，都应该熟悉一下使用系统提供的按钮和图片的准则，以便于：




- 正确使用系统提供的图标。
- 避免设计与系统图标相似度太高的图片。













工具栏和导航栏中使用的标准按钮

iOS 提供了很多在工具栏和导航栏中使用的标准按钮。这些按钮适用于两种样式，每一种都有各自适合的用途：





- **有边框样式：**例如，Phone Contacts 导航栏上的 Add 按钮。这种样式对导航栏和工具栏都适合。
- **朴素样式：**例如，Mail 工具栏上的 Compose 按钮。这种样式只适合工具栏。事实上，如果你把导航栏上的按钮定义成有朴素型，它会自动变成导航栏。



所有系统提供的按钮，都是为了某种语意而设计，不要用作任何其他用途。千万不要不看说明文档，只凭图标的外观判断他的用途。为什么要这么强调呢？详见“[界面元素要一致](#)”

| Button | Name | Meaning |
|---|---------|---|
|  | Action | Open an action sheet that allows users to take an application-specific action |
|  | Camera | Open an action sheet that displays a photo picker in camera mode |
|  | Compose | Open a new message view in edit mode |


| | | |
|---|-------------|---|
|  | Bookmarks | Show application-specific bookmarks |
|  | Search | Display a search field |
|  | Add | Create a new item |
|  | Trash | Delete current item |
|  | Organize | Move or route an item to a destination within the application, such as a folder |
|  | Reply | Send or route an item to another location |
|  | Stop | Stop current process or task |
|  | Refresh | Refresh contents (use only when necessary; otherwise, refresh automatically) |
|  | Play | Begin media playback or slides |
|  | FastForward | Fast forward through media playback or slides |
|  | Pause | Pause media playback or slides (note that this implies context preservation) |
|  | Rewind | Move backwards through media playback or slides |

除了这些系统提供的按钮外，你还可以使用“编辑”“取消”“保存”“完成”“重做”“还原”来支持编辑或其他对内容的操作。

| Button | Name | Meaning |
|---|--------|---|
|  | Edit | Enter an editing or content-manipulation mode |
|  | Cancel | Exit the editing or content-manipulation mode without saving changes |
|  | Save | Save changes and, if appropriate, exit the editing or content-manipulation mode |
|  | Done | Exit the current mode and save changes, if any |

| Button | Name | Meaning |
|---|------|------------------------------------|
|  | Undo | Undo the most recent action |
|  | Redo | Redo the most recent undone action |






表格中列述的按钮既适合导航栏，也适合工具栏，但只有带边框的样式。如果你将其定义成不带边框的样式，它也会自动变回来。








在 iOS4 以后，你还可以在工具栏中使用卷页按钮 ，但并不适合导航栏。

卷页按钮为用户提供卷起页角查看更多信息的方式。例如，Maps 允许用户卷起右下角，查看控制地图的按钮。

不要使用卷页按钮切换整个屏幕。如果要切换屏幕，可以用信息按钮代替“[信息按钮](#)”。确保页角不要完全卷上去，还要留一点在外面，以提醒用户这个只是暂态。如果卷上去的页面消失了，这就太像是全屏转场了，卷页的上下文就消失了。




Tab 栏中使用的标准按钮

| Icon | Name | Meaning |
|---|-----------|--|
|  | Bookmarks | Show application-specific bookmarks |
|  | Contacts | Show Contacts |
|  | Downloads | Show downloads |
|  | Favorites | Show user-determined favorites |
|  | Featured | Show content featured by the application |

| Icon | Name | Meaning |
|---|------------|--|
|  | History | Show history of user actions |
|  | More | Show additional tab bar items |
|  | MostRecent | Show the most recent item |
|  | MostViewed | Show items most popular with all users |
|  | Recents | Show the items accessed by the user within an application-defined period |
|  | Search | Enter a search mode |
|  | TopRated | Show the highest-rated items, as determined by the user |

就像其他标准按钮和图标那样，要确保按照说明文档来使用 tab 栏图标。要根据图标的语义来用它，而非只看外表。这样，即使将来系统系统的图标升级了，也能保证意思不变。更多详情详见“[界面元素要一致](#)”

表格等其他界面元素中使用的标准按钮

| Button | Name | Meaning |
|---|------------------|--|
|  | ContactAdd | Display a people picker to add a contact to an item |
|  | DetailDisclosure | Display a new view that contains details about the current item |
|  | Info | Flip to the back of the view to display configuration options or more information. Note that the Info button is also available as a light-colored “i” in a dark circle. |

使用这些按钮时要保证语义正确。要根据图标的语义来用它，而非只看外表。更多详情详见“[界面元素要一致](#)”

虽然详情展示按钮经常使用在表格里，但也有很多其他地方可以用。更多信息请看“[详情展开按钮](#)”。iOS 中也有一些控件是只在表格中使用的。更多信息请看“[表格视图](#)”

第9章 定制图标和图片指南

每一个程序都需要图标和登录图片。同时也建议程序为 iOS 的 Spotlight 搜索结果提供图标。（有必要的話，Settings 里也要）另外，有些程序需要定制图标区代表特定的文档类型或者程序特定的功能，以及工具栏、导航栏、tab 栏的特定模式。

和程序中其他定制图片不一样，这些图标和图片必须满足特定标准，以便 iOS 能妥当地展示它。表格 9-1 里有图标和图片的标准汇总。

为了支持各种分辨率，你需要提供高分辨率版的图标和图片。对于如何制作高分辨率的图片，可以详见“[为 Retina 屏幕设计画作的技巧](#)”。

| Description | Size for iPhone and iPod touch (in pixels) | Size for iPad (in pixels) | Guidelines |
|--|--|--|--|
| Application icon (required) | 57 x 57 114 x 114 (high resolution) | 72 x 72 | “Application Icons” (page 136) |
| App Store icon (required) | 512 x 512 | 512 x 512 | “Application Icons” (page 136) |
| Small icon for Spotlight search results and Settings (recommended) | 29 x 29 58 x 58 (high resolution) | 50 x 50 for Spotlight search results 29 x 29 for Settings | “Small Icons” (page 138) |
| Document icon (recommended for custom document types) | 22 x 29 44 x 58 (high resolution) | 64 x 64 320 x 320 | “Document Icons” (page 138) |
| Web Clip icon (recommended for web applications and websites) | 57 x 57 114 x 114 (high resolution) | 72 x 72 | “Web Clip Icons” (page 142) |
| Toolbar and navigation bar icon (optional) | Approximately 20 x 20 Approximately 40 x 40 (high resolution) | Approximately 20 x 20 | “Icons for Navigation Bars, Toolbars, and Tab Bars” (page 143) |

| Description | Size for iPhone and iPod touch (in pixels) | Size for iPad (in pixels) | Guidelines |
|----------------------------------|--|---|--|
| Tab bar icon (optional) | Approximately 30 x 30 Approximately 60 x 60 (high resolution) | Approximately 30 x 30 | “Icons for Navigation Bars, Toolbars, and Tab Bars” (page 143) |
| Launch image (required) | 320 x 480 640 x 960 (high resolution) | For portrait: 768 x 1004 For landscape: 1024 x 748 | “Launch Images” (page 144) |

对于图片和图标，我们推荐使用 PNG 格式。标准的色深应该是 24bits，外加 8bits 的 alpha 通道。

你不需要只使用 web 安全色。虽然你可以在设计导航栏、工具栏、tab 栏图表时使用 alpha 透明层，但不要在程序图标中使用。

程序图标

用户会把程序图标放在桌面上，按下它可以启动程序。这应该是品牌宣传和视觉设计的完美结合，形成紧密结合的、高度可辨的、颇具吸引力的画作。每一个程序都需要程序图标。

这个图标也会被用在 Game Center 中。

试图在捕获眼球与表意间寻求平衡，以便它能漂亮、真实，并能清楚地传达程序的本质目标。调查一下来自不同文化的用户如何看待你的图片和配色也会很有意思。

为不同的设备创建不同尺寸的程序图标。如果你的程序要适用于所有设备，你需要提供三种尺寸的图标。

为 iPhone 和 iPod touch:

- 57×57

- 114×114 (高分辨率)

为 iPad:

- 72×72

当在桌面上显示你的图标时，会自动添加以下效果:

- 圆角

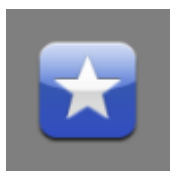
- 投影

- 高光 (除非手动禁用)

例如，普通的 57×57 图标是这样的:



放到桌面上后会变成这个样子:



为确保你的图标与 iOS 提供的加强效果相配，你的图标应当:

- 有 90 度角

- 没有高光效果

- 不使用透明层

程序图标的背景要清晰可见。因为 iOS 自动加了圆角，所以在桌面上的图标要有清晰可见的背景才好看。iOS 加的效果可以保证桌面上的图标有统一的外观，并让用户有按的欲望。如果你的图标没有可见的背景，那么当它出现在桌面上时就看不到圆角。这样的图标看起来不能点击，与用户对桌面图标的期待不一致。

确保你的图片填满了规定区域。如果图标小于推荐的尺寸，或者使用了透明层，它看起来就会像是漂在有圆角的黑色背景上。

例如，程序提供的图标可能有透明层，就像最左侧这颗星星。当 iOS 把它展示到桌面上时，它看起来就像中间这颗（没有高光效果）或者右边这颗（有高光效果）。



在设置了背景图片的桌面上，漂在黑色背景上图标看起来尤其没有吸引力。

为 App Store 做一个 512×512 像素版本的图标。这张图片需要能被立刻识别成你的程序图标，可以拥有更丰富的细节。这个版本的图标不会被附加任何视觉效果。

如果你正在开发 ad-hoc 程序（不通过 App Store 发布，只在 in-house 里展示），同样需要提供 512×512 像素的程序图标。在 iTunes 里也会呈现这个图标。

iOS 也会在其他地方用到这个图标。例如，在 iPad 程序中，如果没有提供文档图标的话，iOS 会用这个版本的大图标自动生成。

小图标

当 iOS 需要在 Spotlight 搜索结果里展示某个程序时，还需要一个小版本的图标。如果需要设置的话，程序还需要在“Settings”里放一个可以与其他内置程序相区分的图标。这个图标应该在一列搜索结果里具有足够的可辨识度。

对于 iPhone 和 iPod touch，iOS 在 spotlight 搜索结果和 settings 里用的是同一个图标。如果你没有提供这个版本，iOS 会把程序图标压缩以做此用。对于 iPhone，应该用这个尺寸的图标：

- 29×29 像素
- 58×58 像素（高分辨率）

对于 iPad，要为 Settings 和 Spotlight 搜索结果提供专门的尺寸：

- 50×50 像素（为 Spotlight 搜索结果准备）
- 29×29 像素（为 Settings）

文档图标

如果你的 iOS 程序定义了自己的文档类型，也应该定制一款图标用以识别它。如果没有提供定制文档图标，iOS 就会把你的程序图标改一下用作默认的文档图标。

例如，用 57×57 像素的程序图标改成的文档图标是这个样子



使用 114×114 像素的高清版图标是这个样子



对于 iPad，使用 72×72 像素程序图标生成的文档图标是这样的



或者，也可以为你的程序定制文档图标。人们会在不同的地方看到你的文档图标，所以最好把它设计得容易记，与程序图标联系紧密。这个图标应该漂亮、表意清晰、细节丰富。

等根据程序是运行于 iPhone 还是 iPad 上来创建不同的图标。

为 iPhone 制作文档图标

对于 iPhone，创建两种尺寸的文档图标

•• 22×29 像素

•• 44×58 像素（高分辨率）

把你画的图套在这个规定的格子里。可以居中放置，或者缩放填充。

例如，如果你提供了一个如左图所示的 22×29 的图标，iOS 会自动生成右图那样的图标。



相似地，如果你做了一个 44×58 像素的图标，iOS 会生成右图那样的图标。



为 iPad 制作文档图标

iPad 版 iOS 使用两种尺寸的图标，64×64 像素和 320×320 像素。最好两种尺寸的图标都准备好，以便在任何环境中都能找到合适的尺寸。

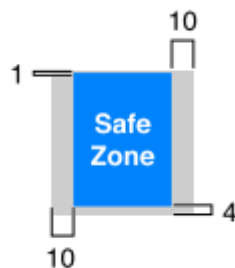
对于两种尺寸，其中都包含了 padding，留下稍小一点的安全区。要确保你的画作适合这个安全区，否则就会被裁切掉。

即使你的画作适合了安全区的尺寸，右上角也总是会被 iOS 添加的卷角效果遮掉一部分。另外，iOS 还会覆盖从黑（上）到透明（下）的渐变。

为了创建 64×64 像素的文档图标，应该：

- 1、 创建 64×64 像素的 PNG 图像
- 2、 加入如下尺寸的 Margin，创建安全区
 - a) 顶部 1 像素
 - b) 底部 4 像素
 - c) 左右各 10 像素

你的安全区应该是长这个样子。



- 3、 把你的画作放在 44×59 像素的安全区里，可以居中或缩放以填充整个安全区。
(注意 iOS 会自动添加卷角和渐变效果)

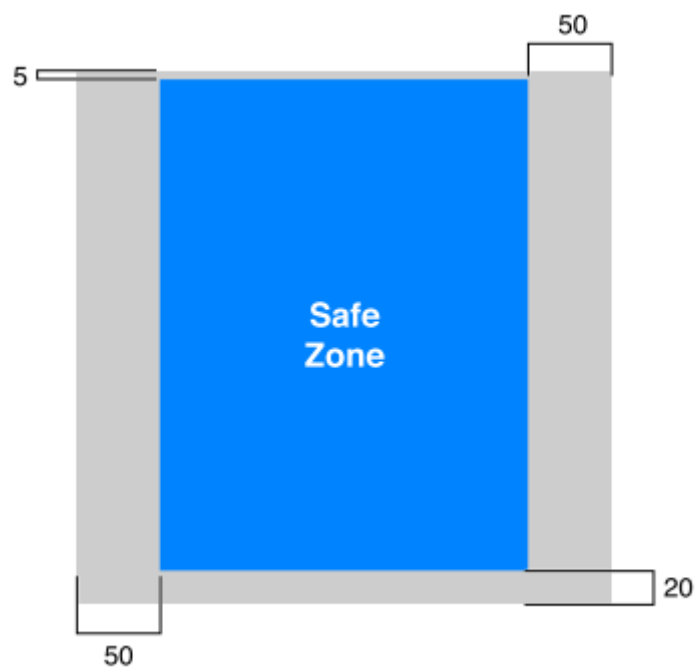


为了创建 320×320 像素的文档图标，应该：

- 1、 创建 320×320 像素的 PNG 图像
- 2、 加入如下尺寸的 Margin，创建安全区

- a) 顶部 5 像素
- b) 底部 20 像素
- c) 左右各 50 像素

你的安全区应该是长这个样子。



- 3、 把你的画作放在 44×59 像素的安全区里,可以居中或缩放以填充整个安全区。(注意 iOS 会自动添加卷角和渐变效果)



Web 快捷方式图标

如果你有 web 小程序或者网站，可以定制一个图标，用户可以把它直接放在桌面上。用户可以按图标直接访问网页内容。你可以让定制的图标代表整个网站或者某个网页。

如果你的网页有独特的图片或者可识别的颜色主题，那最好应用到图标里。但是，为了确保你的图标在设备上看起来很棒，你应该遵照以下指南：

为 iPhone 和 iPod touch 创建如下尺寸的图标：

- 57×57 像素
- 114×114 像素

为 iPad 创建如下尺寸的图标：

- 72×72 像素

就相对程序图标做的那样，iOS 会给图标自动添加一些视觉效果，以便于与其他桌面图标一致。添加的效果有：

- 圆角
- 投影
- 反射高光

例如，57×57 像素的网页图标是这样的：



添加效果后就成了这样



为了确保你的图标与 iOS 添加的效果相得益彰，你的图标应该做到：

- 有 90 度尖角
- 没有高光效果

导航栏、工具栏和 tab 栏上用的图标

首先，你应该尽可能的使用系统提供的按钮和图标来代表标准任务。完整版本的标准按钮和图标以及使用指南可以详见“[系统提供的按钮和图标](#)”

当然，不是每一个任务都是标准的。如果你的程序包含用户经常要执行的任务，就需要创建用于导航栏和工具栏的定制图标来代表这些任务。相似地，如果你的程序用 tab 栏在不同的定制内容和定制模式间切换，就需要为 tab 栏定制图标。

在绘制图标之前，你需要花些时间考虑一些这些图标想表达什么。当你思考时，要让你的图标能够：

简单而富有流线感。太多的细节会让图标显得笨拙，难以辨认

不容易和系统提供的图标搞混。用户应该能一眼把你的图标和标准图标分开。

易懂，容易被接受。要让你的图标能够被大多数用户理解，不会有用户抵触它。

避免使用和苹果产品重复的图片。这些图片都是由产权保护，并且会经常变动。

在思考图标外观的时候，最好依照如下指南：

- 纯白色要有合适的透明度
- 不要包含投影效果
- 使用抗锯齿
- 如果要添加斜面效果，确保光源放在正上方。

工具栏和导航栏上的图标尺寸应如下所示：

- 对于 iPhone 和 iPod：
 - 大概 20×20 像素
 - 大概 40×40 像素（高分辨率版本）
- 对于 iPad：
 - 大概 20×20 像素

tab 栏上的图标尺寸应如下所示：

- 对于 iPhone 和 iPod：
 - 大概 30×30 像素
 - 大概 60×60 像素（高分辨率版本）
- 对于 iPad：
 - 大概 30×30 像素

不要为你的图标提供选中态或按压态。iOS 会为导航栏、工具栏和 tab 栏的图标自动生成这些状态，所以就不用劳神了。因为这些效果是自动叠加的，所以也没法定制。

要让所有图标看起来一样重。要在所有图表间平衡尺寸、细节丰富度以及实心部分。一般而言，把大的小的、细致的粗糙的、空的实的图标混在一个栏上很不好看。

登录图片

为了增强登录时的体验，你必须提供至少一张登录图片。登录图片和程序开启后的第一帧很像。当用户开启程序后，iOS 立刻把它展示出来。等第一屏渲染好了，iOS 再立刻用它把登录图片替换下来。

提供登录图片提升用户体验，不要在如下情况使用它：

- 用作“splash”
- 用作“about”
- 用于品牌推广，除非它真的是你程序的第一屏。

因为用户经常会在程序间切换，所以你应该将登录时间尽量缩短。提供登录图片就可以缩短等待时间的主观体验。

设计与程序启动后第一帧一样的登录图片，除非：

- 文本。登录图片是静态的，所以其中的文本没有办法做定位。
- 可能会变化的界面元素。如果第一帧旋绕出来后有元素会变化，就不要在登录图片中放它了。这样用户就不会觉察到登录图片和第一帧之间的切换。

对于 iPhone 和 iPod touch，创建如下尺寸的登录图片（含状态栏）“

- 320×480 像素
- 640×940（960？）像素（高分辨率）

对于 iPad，创建如下尺寸的登录图片（不含状态栏）“

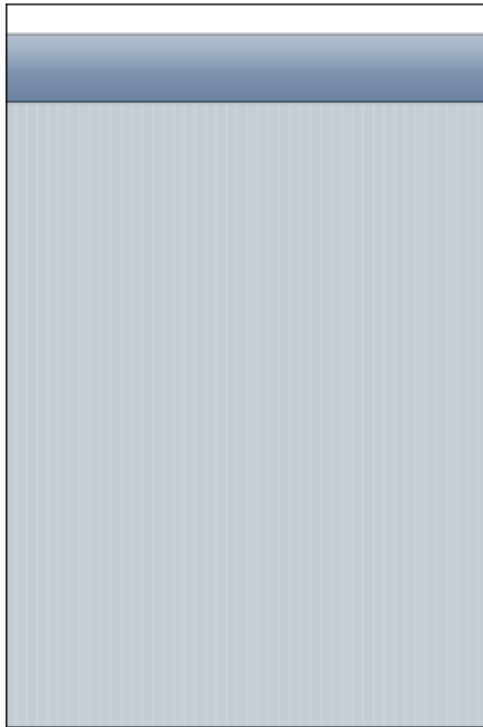
- 768×1004 像素（竖屏）
- 1024×748 像素（横屏）

对于 iPad，应该各种方向的登录图片都提供一份。

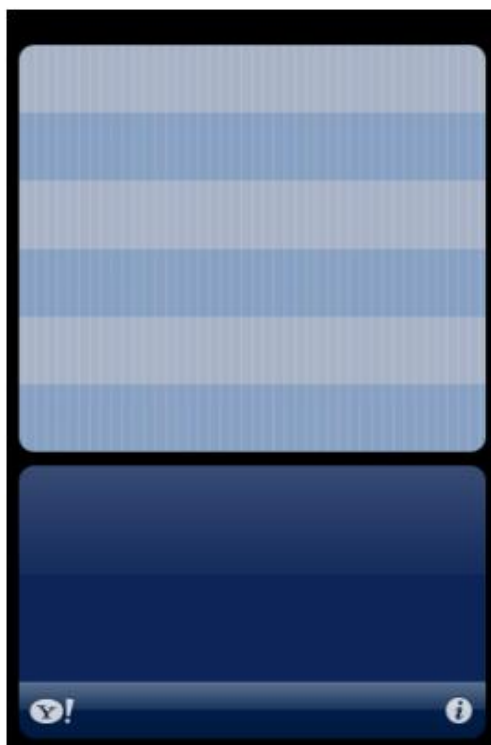
如果你觉得完全依照这些指南做出的登录图片太过朴素，恭喜你，猜对了。记住，登录图片不是为了给用户留下美观的印象，而是为了让用户觉得你的程序启动迅速，使用灵活。

接下来的例子将告诉你登录图片能多朴素：

iPhone Setting 的登录图片只有程序背景，因为这里面的内容都是变来变去的。



iPhone Stocks 的登录图片只有静态背景，因为只有这些是恒定不变的。



为 Retina 屏幕设计画作的技巧

Retina 液晶屏允许你展示高精度的图标和图片。如果仅仅是放大已有的画作，就会错失提供优美、精致图片的机遇。你应该利用已有的素材重新制作大尺寸高质量的版本。

- 纹理丰富。例如，在高精度版的 Settings 和 Contacts 里，铁盒纸张的纹理清晰可见。



- 更多细节。例如，在高精度版的 Safari 和 Notes 里，你可看到更多的细节，比如指针后的刻度和上一张纸撕掉后残留的痕迹。



- 更加真实。例如，高精度版的 Compass 和 Photos 图标通过增加丰富的纹理和细节，变得像是真的指南针和照片。



虽然栏上的图标比程序或者文档图标简单，你也应该在高分辨率版本上增加细节。例如，iPad 里面的艺术家图标是一个歌手的侧面剪影。高分辨率版本的图标看起来和原版本一样，但增加了很多细节。



如下技术可以在设计高分辨率图标时帮大忙：

把原有图片放大至 200%。要使用“nearest neighbor”缩放算法。如果原来的图像不是矢量图形或带有图层样式的话，这样会很管用。最后获得的会是放大的、像素化的图片。你可以在上面再添加更丰富的细节。这种方法可以帮助你节约工作量，保留原有的布局。

如果图片是矢量版的，或者有图层效果，使用默认算法缩放就可以了。

增加细节和深度。不要急着去小元素，因为高分辨率版本给细节留下了很多发挥空间。原来的 1 像素变成了现在的 4 像素。

考虑修整放大的元素。如果你原来的分割线是很细腻的 1 像素，等放大后就会变粗，成为 2 像素宽。但是对于某些线和元素，在放大整体尺寸后还需要再锐化或者让它保留原有尺寸。

考虑为雕刻或投影等效果增加模糊。例如文字的雕刻效果通常是把文字复制一次，然后移动 1 像素。放大之后，这个移位就变成 2 像素了，在高分辨率屏幕上看起来就太细腻了，不真实。为了优化，你可以让移位保留在 1 像素，但是增加 1 像素的模糊来柔化雕刻效果。这仍然会导致 2 像素宽的效果，但是外面这层像素看起来仍然只有半像素宽，看起来也更加舒服。