



超人的电话亭
Supermancall.com

苹果 iOS13 设计规范

—



前言

Foreword

从 iOS7 到现在，市面上已经很久没有出现过完整并便于查阅的苹果设计规范内容了。对于英语不好的设计师、产品经理、测试人员，只能在网上查阅那些破碎的片段，始终无法窥探 iOS 规范神秘的全貌。

在我的 UI 培训授课过程中，很多学员都抱怨过这个问题，而常规的培训内容是无法完全将规范的所有内容一一作出解答的，所以出于责任和义务，我组织并翻译了这套规范。

为了便于使用，只是通过网页的图文是无法满足需要的，所以我们排版制作成 PDF 的格式，可以保存在各种设备查看。并且，在制作的过程中我们还考虑了打印成册的情景，对于使用的色彩、线条的粗细都做过优化。

本次翻译大概有5.6万个单词，感谢所有自愿加入并参与翻译的我的学员们：**刘璐、克里斯、昆、呸呸、旦若霞、金琳、吡咯、璟、杉杉、圆圆、拉面、晓玲、鸡肚儿、MAX、SkyLee**

之后，我会不断更新新的版本，包括优化和苹果官方更新的内容，请关注我的公众号 “超人的电话亭” 并回复 “iOS规范”，或扫描下放二维码：



目录

Contents

前言	i
目录	ii
规范预览	1
1.1 原则iOS设计原则	1
1.2 界面元素	4
App 架构	6
2.1 加载	6
2.2 模态	8
2.3 导航	11
2.4 用户引导	14
2.5 授权许可	16
2.6 设置	18

用户交互	21
3.1 3D touch	21
3.2 音频	22
3.3 身份验证	25
3.4 数据键入	29
3.5 拖放	31
3.6 反馈	37
3.7 文件处理	38
3.8 手势	40
3.9 触觉	44
3.10 近场通信技术 (NFC)	50
3.11 撤销和重做	52
系统功能	55
4.1 增强现实	55
4.2 多任务处理	69
4.3 iPad 上的多窗口	72
4.4 通知	74
4.5 打印	79
4.6 快速查看	80

4.7 评级和评价	81
4.8 截图	84
4.9 电视提供商	85
视觉设计	89
5.1 适应性和布局	89
5.2 动效	101
5.3 品牌化	102
5.4 色彩	104
5.5 深色模式	112
5.6 材质	114
5.7 术语	117
5.8 排版	119
5.9 视频	129
图标和图像	133
6.1 图像尺寸和分辨率	133
6.2 应用图标	135
6.3 自定义图标 (iOS12 及更早的版本)	140
6.4 启动画面	144

6.5 系统图标	146
----------	-----

UI栏 153

7.1 导航栏	153
7.2 搜索栏	156
7.3 状态栏	158
7.4 标签栏	161
7.5 工具栏	163

内容视图 165

8.1 动作菜单	165
8.2 活动视图	166
8.3 警告框	168
8.4 精选	171
8.5 图像视图	173
8.6 地图	174
8.7 翻页	176
8.8 弹出框	177
8.9 滚动视图	179
8.10 分屏视图	181

8.11 表元素	183
8.12 文本视图	187
8.13 网页视图	188
控件	190
9.1 按钮	190
9.2 情景菜单	194
9.3 编辑菜单	196
9.4 标注	198
9.5 页面指示器	199
9.6 选择器	200
9.7 进程指示器	203
9.8 刷新内容控件	206
9.9 分栏控件	207
9.10 滑块	208
9.11 分档器	209
9.12 开关	210
9.13 文本框	211
扩展	214

10.1 自定义键盘	214
10.2 文件应用	217
10.3 主屏幕快捷操作	218
10.4 消息	220
10.5 照片编辑	226
10.6 共享和动作	228
10.7 小部件	230
术语	233
11.1 Apple 支付	233
11.2 增强现实	247
11.3 GameKit	258
11.4 HealthKit	259
11.5 HomeKit	263
11.6 iCloud	272
11.7 内购	274
11.8 Live Photos	276
11.9 ResearchKit	278
11.10 社交媒体	288
11.11 钱包	290

其他 293

12.1 SF Symbols 293

12.2 辅助功能 300

12.3 轻按和重压 303

12.4 Siri 305

第一章

规范预览

Overview

—

1.1 原则



iOS设计原则

作为一名 App 设计师，你就有机会发布一款能够登上 App Store 榜首的卓越产品。为此，你的应用在质量和功能上必须精益求精。

iOS 与其他平台不同，主要是下面三大原则：

- **清晰：**整个系统中，任何字号的文字都必须清晰易读，图标表达含义

准确易懂，修饰恰到好处，以功能驱动设计。留白、颜色、字体、图形和其他界面元素能够巧妙地突出重点内容并传达交互性。

- **顺应：**流畅的动效和清晰美观的界面有助于用户理解内容并与之交互，且不会干扰用户。当内容占据整屏时，半透明和模糊处理通常会暗示其他更多的内容。减少使用边框、渐变和阴影，使界面尽可能轻量化，从而突显内容。
- **纵深：**清晰的视觉层和生动的动效赋予界面层次感，使其富有活力并有助于理解。使可触发界面元素更容易被找到能提升体验的愉悦感，让用户在触发相应功能或者获取更多内容时不至于茫然无措。当用户浏览内容时，流畅的过渡能够提供纵深感。

设计原则

要想扩大影响力和延展性，在你设计 App 时，请牢记以下原则：

整体美感：

整体美感体现在一款 App 的视觉外观、交互行为与其功能结合的优异程度。例如，一款协助用户完成重要任务的 App 应该使用不易察觉且不会造成干扰的图形、标准化控件和可预知的交互行为，从而使用户聚焦在任务本身。反之，一款沉浸式体验的 App（如游戏），需要提供一个有吸引力的界面，在鼓励用户探索的同时为用户带来无穷的乐趣和激动。

一致性

一致的应用程序通过使用系统提供的界面元素、众所周知的图标、标准的文

本样式和统一的术语来实现近似、一致的规范和范式。该应用程序以符合人们预期的方式结合功能和操作行为。

直接处理

屏幕内的内容能够直接处理这一点足够吸引用户并促进他们理解内容。用户在旋转设备或使用手势影响屏幕内容时系统直接处理并返回结果，因此他们可以在行动后直接看到结果。

反馈：

反馈能够响应交互操作，呈现结果，便于用户了解情况。系统自带的 iOS 的 App 对用户的每个操作都提供了明确的反馈。

- 交互元素在点击时会被高亮显示
- 进度指示器显示了需要长时间运行的操作进度
- 动效和声音使用户能够更清晰地感知交互行为的结果

隐喻：

当一个 App 的虚拟对象和行为与用户所熟悉的体验相似时——无论这种体验来源于现实生活亦或是数字世界，用户就能够更快速地学会使用这款 App。隐喻在 iOS 中能够起作用是因为用户与屏幕在进行物理上的交互。

- 可以通过移动分层视图来显示被遮挡的内容
- 可以拖拽并滑动对象
- 可以切换开关，移动滑块，滚动数值选择器
- 可以通过轻扫来翻阅书籍和杂志

用户控制：

在 iOS 中，用户是决策者，而不是App。App可以对用户行为提出建议，对可能造成严重后果的行为发出警告，但不应该直接替用户做决策。优秀的App 在用户主导和避免不想要的结果之间找到平衡。为了让用户拥有掌控性，App 可以使用用户熟悉且可预知的交互元素，让用户二次确认破坏性的行为，并且保证可以停止正在执行中的操作。

1.2 界面元素

大多数的 iOS App 使用了来自 [UIKit](#) 的组件进行搭建，这是一个定义了基本界面元素的编程框架。这个框架让各种 App 在视觉上达到一致的同时还提供了高度的个性化。UIKit 元素是灵活且常见的。且它们是可适配的，让你能够设计一个在任何 iOS 设备上看起来都很棒的 App，而且能够在系统发布新版本的时候自动更新。由 UIKit 提供的界面元素可以分为以下三种：

栏 (Bars) : 告知用户现在在App中所处的位置，提供导航，而且还可能包含按钮或者其它用来触发功能和交流信息的元素。

视图 (Views) : 包含用户在App内最关注的信息，例如文本、图形、动画和交互元素的容器。视图允许使用诸如滚动、插入、删除和排列之类等交互行为。

控件 (Controls) : 触发功能和传递信息。控件包括按钮、开关、输入框和进度指示器。

除了定义 iOS 界面，UIKit 还定义了你的 App 能够采用的功能。通过这个

框架，你的 App 可以对触摸屏上的手势做出回应，还可以实现一些例如绘画、辅助和打印的功能。

iOS 也和其他编程框架和技术紧密结合，例如 Apple Pay、HealthKit 和 ResearchKit，帮助你设计出一个强大且使人惊叹的 App。

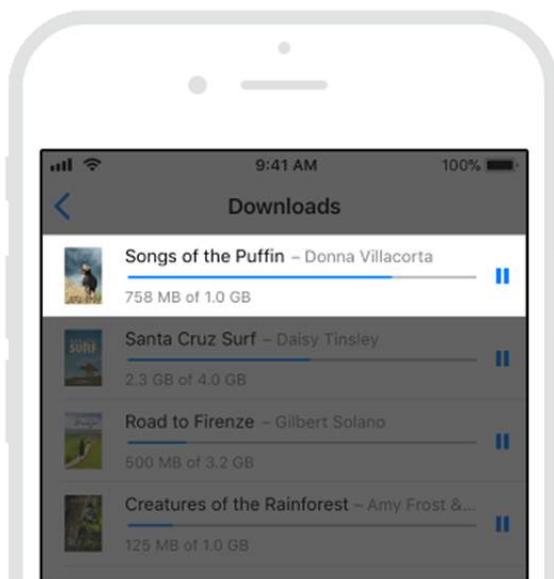
第二章

App 架构

App Architecture

2.1 加载

当内容在加载时，一片空白静止的屏幕，让你的 App 好像被冻结了一样，不仅让用户感到困惑和失望，而且很有可能会使用户离开你的 App。

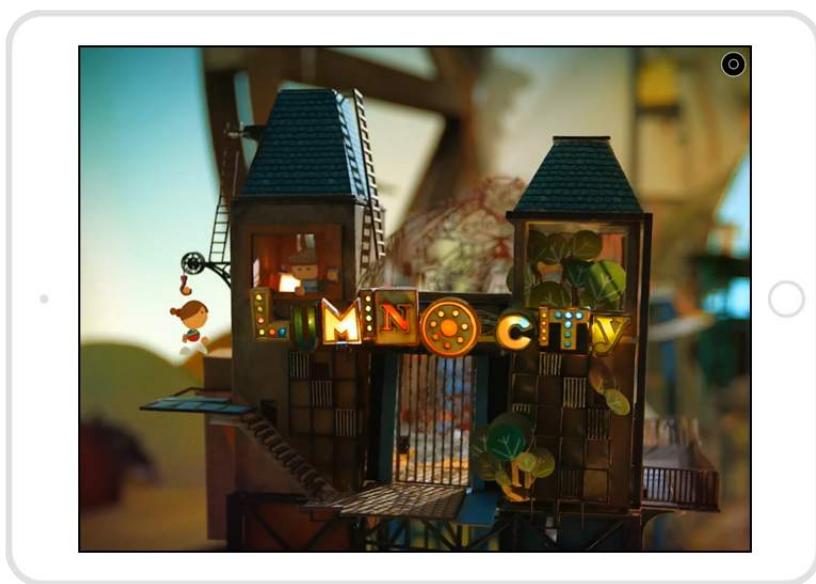


要清晰的表明加载的状态：至少，展示一个动态旋转图来表明有任务正在进行中。更好的做法是，显示更清晰明确的加载进度，这样用户就能知道他们

还需等待多久。

加快显示内容的速度：不要让用户等着且只看到加载进度，要尽快显示内容，可以通过占位符、图形或者动画，表明哪些内容还未加载完成。当内容加载成功后，替换掉之前的占位元素。有可能的话，在播放动画或用户操作菜单、导航的过程中，就提前在后台预加载要显示的内容。

通过启发或娱乐用户来填充加载的过程：尝试一下展示游戏秘籍、娱乐性的动画或者其他有趣的占位图。



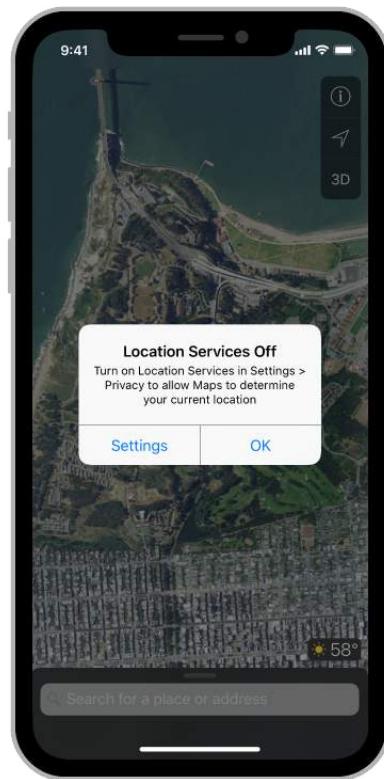
自定义加载画面：尽管有标准化的进度指示，但它们有时会不符合当前的使用场景。这时设计符合你的 App 或游戏风格的自定义动画和元素，可以实现更具沉浸式的使用环境。

更多指南，请参阅 [进度指示器](#)。

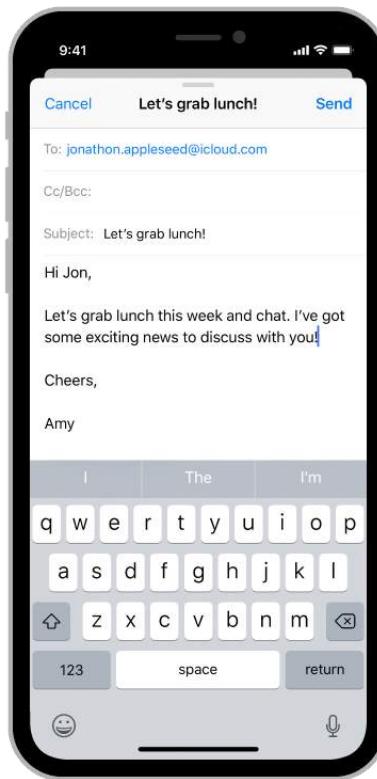
2.2 模态

模态是一种设计技巧，它以一种临时的、与用户当前情景所隔离开的模式呈现内容，并且需要一个明确的操作才能退出。以模态方式呈现内容可以：

- 帮助人们专注于一个独立的任务或一组密切相关的选项
- 确保人们接收到关键信息，并在必要时对其进行操作



警告



模态视图

iOS 提供了 Alerts、Activity Views (或 Share Sheets) 以及 Action Sheet 供您的 APP 在一些特殊情况下使用。为了在您的应用中显示自定义的模态内容，iOS13 及更高版本支持以下呈现样式。

遮片 (sheet)

遮片样式显示为部分覆盖底层内容的卡片，并使所有未覆盖区域变暗以防止与它们发生交互。父视图或前一张卡片的上半部分在当前卡片后面可见，以帮助人们记住他们在打开卡片时暂停的任务。人们通过以下方式关闭卡片：

- 从屏幕顶部向下滑动
- 当卡片内容滚动到顶部时，从屏幕上的任何位置向下滑动
- 点击按钮

当使用遮片作为非沉浸的模态内容的形式时不允许出现复杂的任务。

全屏 (Full Screen)

全屏样式覆盖整个屏幕，完全覆盖了之前的视图，最大限度地减少了视觉干扰。人们通过点击按钮来关闭全屏模态视图。

使用全屏模式视图可以获得沉浸式的内容，比如视频、照片或相机预览，或如标记文档或编辑照片这样可以从全屏模式中获得更好体验的复杂任务。

NOTE

在诸如分屏窗格，弹出窗口或其他非全屏视图这种复杂环境下，应改用遮片样式呈现模态内容。

必要时才使用模态：只有当人们的注意力集中在做出关键选择或执行与当前任务不同的任务时，才使用模态。模态使人们脱离当前的情景并需要手动关闭，因此必须拥有明显的收益时才能使用它。

为传达必要且可操作的信息保留警告弹窗：通常，会出现警告弹窗是因为哪

里出现了问题。由于警告弹窗会中断操作体验并需要点击才能关闭，因此很重要的一点是：让人们认为这种打扰是有道理的。相关规范，请参阅[警告弹窗](#)。

保持模态任务的简单、简短且注意力集中：不要在你的 App 中创建另一个 App。如果一个模态框太复杂，当用户看到时，就会忽略他们之前暂停的任务。当创建一个包含多层级视图的模态框时，请格外谨慎，因为用户可能不知道如何返回原先的步骤。如果模态框必须含有子视图，那就提供单级的跳转路径以及清楚的“完成”路径。避免在完成任务之外的地方使用“完成”按钮。

请永远保留模态框的“关闭”按钮：例如，您可以使用“完成”或“取消”。包含按钮可确保模态能够使用辅助技术，并提供备选的关闭手势。

必要时，关闭模态之前让人们获得确认来避免数据丢失：无论人们使用手势还是按钮来关闭视图，如果操作可能导致用户生成内容丢失，请提供解释情况的 Action Sheet，并为人们提供解决方法。

不要在弹出框上方显示模态框：尽管你可以在弹出窗内部显示模态框，但是除了警告框之外，其他任何元素都不应该覆盖在弹出框之上。在极少数情况下，用户完成弹出框的指示后，还会显示模态框。那么，请先关闭弹出框，再展示模态视图。

尽可能使用能够明确说明模态任务的标题：当人们进入模态任务时，他们会从之前的情景中切换出来，因此最好使这个新的情景清晰。您还可以在视图的其他部分中提供更全面的任务描述或提供指导。

让模态框的视觉风格与你的 App 相符：一个模态视图可能包含导航栏。在

这种情况下，请使用与你App内的导航栏一样的视觉外观。

选择合适的过渡动画显示模态框：使用与您的应用程序风格相符的过渡动画，增强用户对临时情景切换的感知。默认过渡动画为模态框从屏幕底部垂直向上滑出，当它被关闭时向下滑走。在整个应用中统一过渡动画。

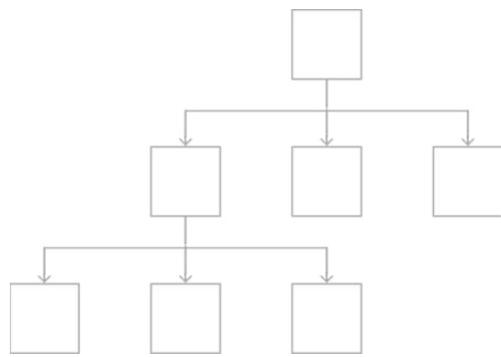
开发者请参阅[UIViewController](#) 和[UIPresentationController](#)。

2.3 导航

在一个 App 的导航栏达不到用户的预期时，他们往往是意识不到的。你的工作就是在不引起注意的情况下，创造一种能够适应结构和目的的导航。导航应该让用户觉得自然和熟悉，并且不应该主导界面或者抢走页面内容的风头。在 iOS，主要有三种导航结构。

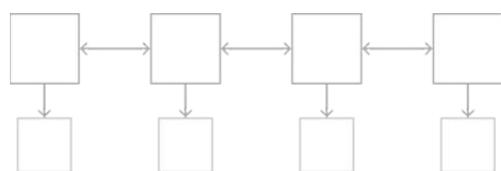
分层导航

在每个页面都只做一次选择，直到到达目标页面。要想到达另外的目标页面，你必须重回之前的步骤或是从头开始重新选择。“设置” 和 “邮件”就是采用这种导航结构。



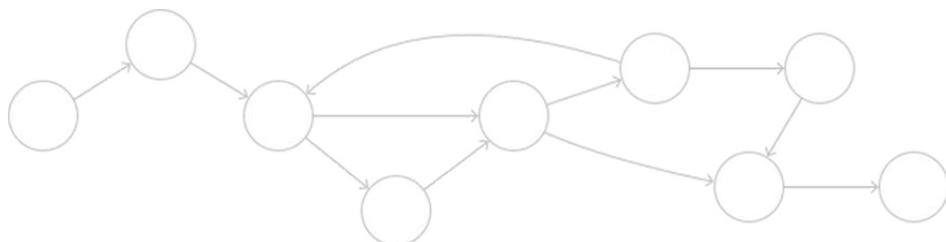
扁平导航在不同的内容类别间切换

“音乐” 和 “App 商店” 就是采用这种导航结构。



内容驱动或是体验驱动式导航

在内容中自由地转换，或是内容本身定义导航。游戏、阅读以及其它沉浸式 App 一般都采用这种导航结构。



有的App结合了多种导航形式。例如，采用了扁平导航的 App 也可能在每个类别里使用分层导航。

始终提供清晰的路径：用户应该一直知道他处于 App 的什么位置，以及如何去往下一个目标页面。无论导航的样式如何，都应该确保内容间的路径是合理的、符合预期的并且容易追溯的。一般来说，每个页面只提供一条路径。如果他们需要在多个情景下看到某一屏的内容，那么考虑采用操作列表、警告框、弹出层或是模态框的形式来展示这些内容。β

了解更多内容，请参阅 [Action Sheets](#), [Alerts](#), [Popovers](#), 和 [Modality](#)。

设计一个能够简单快速地访问内容的信息结构：合理地组织你的信息结构，保证只需要用最少次数的点击、滑屏和跳转就能访问相应的内容。

使用触摸手势来制造流畅感：让用户能轻松地在界面内跳转，而感受不到阻力。例如，你可以让用户在屏幕边界轻扫来返回到上一屏。

使用标准的导航组件：可能的话，使用标准的导航控件比如页面管理、底部栏、分段控件、表格视图、精选视图和分割视图。用户已经熟悉了这些控件，他们很自然地就知道如何玩转你的 App。

使用导航栏访问分层内容：导航栏内的标题能够显示当前的层级位置，使用返回按钮能够轻易地回到上一个位置。

了解更多指南，请参阅 [Navigation Bars](#)。

使用底部栏来显示内容相符的类目或是功能模块：底部栏让用户能够快速简单地在不同类别中切换，而不受当前位置的限制。

了解更多指南，请参阅 [Tab Bars](#)。

当有多个显示同类型的内容的页面时，请使用页面控件：页面控件能够清楚

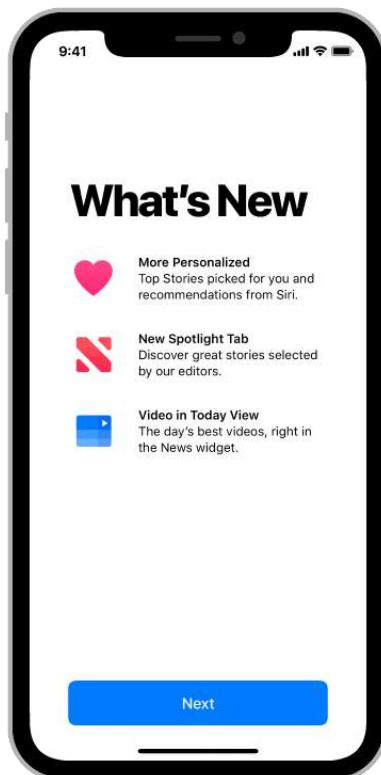
地表示总页数，以及当前所在页。“天气”App 就使用了页面控件来显示不同地理位置的天气页面。了解更多指导，请参阅 [Page Controls](#)。

提示：分段控件和工具栏不具备导航功能。使用分段控件能够将信息整理到不同的类别。使用工具栏可以为当前内容提供交互控件。

了解更多信息，请参阅 [Segmented Controls](#) 和 [Toolbars](#)。

2.4 用户引导

App 的启动过程是你第一个接触和联系新老用户的时机。请设计一个快速、有趣并有启蒙意义的用户引导。



提供启动画面：在 App 打开时启动画面会在 App 加载初始内容的同时进行显示，让用户感觉你的 App 能够快速响应。因为这个画面很快就会

被 App 的首页替代，所以除非出现有限制性的文字和可交互的元素外，它应该尽量与首页相似。了解更多，请参阅 [Launch Screen](#)。

选择适当的显示方向启动用户引导页面：如果你的App同时支持竖屏和横屏模式，那么应该以设备当前的方向启动。如果你的App只在一个方向运行，则应该始终以该方向启动，并让用户旋转设备来使用。除非有迫不得已的原因，否则无论设备是向左旋转还是向右旋转，处于横屏模式的App都应该正确地适配方向。了解更多信息，请参阅 [Adaptivity and Layout](#)。

快速响应：避免出现启动画面、菜单和说明时间过长，要让用户快速进入App。如果你的App需要教学或是介绍步骤，为用户提供一个跳过的选项，并且不要对老用户展示这些。

提前设想用户可能会需要的帮助：主动考虑用户何时会遇到麻烦。例如，一个游戏可以在暂停或是角色前进受阻时提供一些小秘籍。当用户错过启动画面时，让他们可以重新观看教程。

只在教程中展示最关键的内容：虽然为新用户提供引导没错，但是好的设计是不需要太多教学的，设计首先要确保App是易懂的。如果你的App需要过多的引导，那么请重新审视你的设计。

让学习过程变得有趣而且易于学习：让用户通过操作来学习比阅读一长串说明来的更有趣和有效。通过动效和交互循序渐进地教学。避免展示看起来似乎可以操作的屏幕截图作为教学说明。

避免在一开始就要求用户填写资料：用户会想要App马上可以使用。App是为大多数人设计的，让少数想要更多服务的人，通过填写资料来满足更多需求。尽可能地从设备设置和默认设置，或者通过同步服务例如 iCloud 来

获取信息。如果 App 必须要求用户填写资料，那么刚开始在 App 内提示用户，并让用户稍后在 App 设置中自行修改。

避免在 App 内显示接受许可协议和免责声明：在你的 App 被下载之前，直接在苹果商店显示接受许可协议和免责声明。如果你必须将这些东西放在你的 App 里，那么以一种自然的方式将其融入，避免干扰用户体验。

在你的 App 重新启动时，恢复之前已有的状态：不要让用户再操作一遍关闭或退出 App 之前的步骤。保存并且复原 App 的已有状态，这样用户就能继续上次他们未完成的操作了。

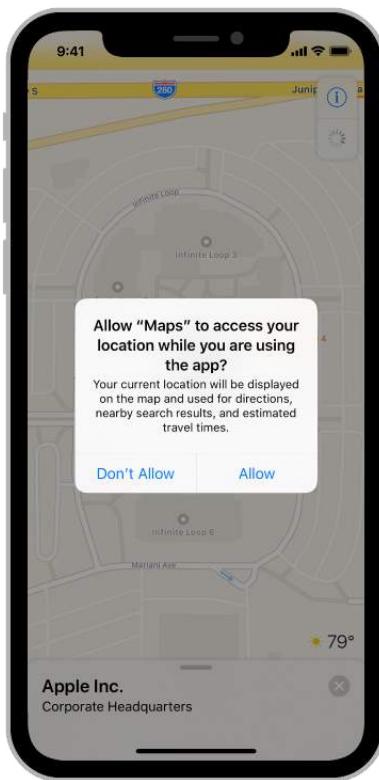
不要太快或是太频繁地要求用户对你的 App 评分：太快或是太频繁地要求评分会让人厌烦，并且会减少你收到的有用反馈的数量。为了鼓励填写有效的反馈，在要求评分之前，给用户足够的时间使用 App。在评分时，始终提供“退出”的选项，并且永远都不要强迫用户对你的 App 进行评分。

不要鼓励重启：重新启动耗费时间，并且让你的App看起来既不可靠又不易用。如果你的App出现储存或者其它问题，导致它无法运行只能重启系统，那么你应该解决这些问题。

2.5 授权许可

只有用户对 App 予以授权，App 才能获取用户的私人信息，比如当前位置、日历、联系人、提醒事项以及照片。虽然 App 获得这些信息后，用户使用起来会比较方便，但他们还希望能够掌控自己的私人数据。比如，用户希望能够根据实际位置自动标记照片，或是寻找附近的朋友，但是他们又同

时希望能有关闭这些功能的选项。



只有在App确实需要获取私人信息时，再向用户发起请求：用户自然会质疑获取个人信息的目的，尤其是他们发现当前的请求没有什么必要时。确保授权请求只出现在用户使用某些需要个人数据的功能时才出现。比如，一个 App 只有在启用位置跟踪的功能时，再请求获得当前位置。

当需求不明显时，向用户解释为什么你的 App 需要这些信息：你可以在系统提供的授权请求警告框上添加自定义文本。使用明确且有礼貌的文本，这样用户就不会感到有压力。保证文本言简意赅，并且注意段落要求和首字母大小写等问题。没有必要包含你的 App 名字，系统已经替你在警告框上写明了App 名称。

在 App 一启动时就请求授权那些对运行 App 至关重要的信息：如果用户明确地知道你的 App 只有获取这些个人信息才能运行，那么他们就不会反

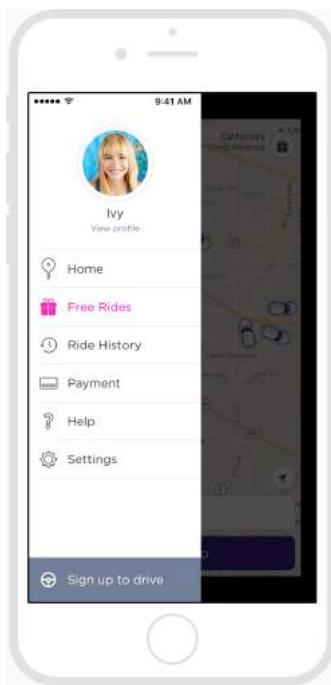
感。

不要随便请求位置信息：在获得位置信息之前，检查系统以查看位置服务是否已经被启用。使用这个方案，可以延迟提醒，直到使用需要获取该信息的功能时才进行提醒，甚至可能完全避免提醒。

学习如何实现定位功能，请参阅 [Location and Maps Programming Guide](#)。

2.6 设置

有些 App 可能需要一开始就让用户决定设置或配置选项，但是大部分 App 可以避免或是延迟这些操作。成功的 App 能够一开始就让大多数用户流畅的使用，并且同时提供了一个便捷的方式去调整体验。当你设计的App满足大部分用户的需求，就可以减少他们对设置的操作了。



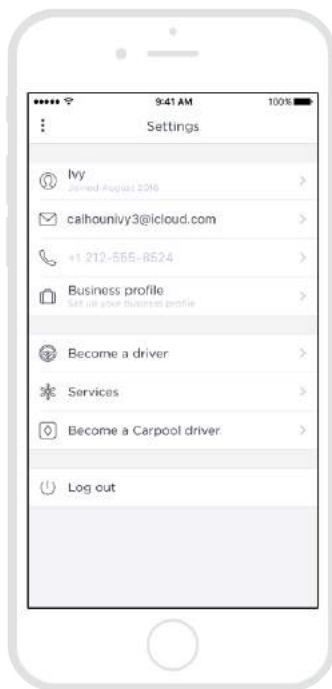
推断你可以从系统中得到什么：如果你需要关于用户、设备或是环境的信

息，那么尽可能地向系统请求而不是直接询问用户。比如，如果你想要知道用户的邮编来提供本地化的选项时，可以向用户请求获取他们的当前位置（译者注）。

注：系统会自动根据位置生成右边，电话的区号同理的问题

慎重地考虑设定选项的优先级：App 设置主界面是一个放置关键或时常变更选项的合理位置，如果权重不是特别高的选项应该将它们安排到次要页面中

把不经常更改的选项放到系统设置里：设置 App 是更改系统配置的核心地带，但是用户必须离开App才能到达。因此在你的 App 中直接改变设置会更加方便。



如果你的App必须提供很少需要改动的设置选项，请参阅 [Preferences and Settings Programming Guide](#) 中的 [Implementing an iOS Settings Bundle](#) 部分。

适时的提供通往“设置”的快捷方式：如果你的 App 包含引导用户前往设置的文本，比如 “去设置>我的 App >隐私>定位服务”，请提供一个能够

自动打开该界面的按钮。

了解如果实现这个行为，请参阅 [UIApplicatio](#) 中的 [Settings Launch URL](#) 部分。

第三大章

用户交互

User Interaction

3.1 3D touch

3D Touch 为触控交互提供了一个全新的维度。用户可以在支持的设备中，通过不同力度按压屏幕来访问额外的功能。App 会通过展示菜单、显示额外内容、播放动画等方式来进行反馈。使用 3D Touch 功能不需要学习新的交互手势，只要轻轻按压屏幕就可以轻易发现额外的内容。

主屏交互

在运行 iOS13 或更高版本设备的主屏幕上，人们可以通过长按 APP 图标来呼出一个情景菜单（在有 3D touch 的设备上重按可以更简单地呼出菜单）。应用的情景菜单可以让用户快速执行 APP 的一些特定操作，或者浏览一些有趣的内容。比如日历提供了「新建日程」的快捷按钮，还显示日程表上的「下一个日程」。详情请参阅 [Home Screen Actions](#) 和 [Widgets](#)。

动态照片

App可以支持动态照片，按压照片可以用来查看动态的照片，包含动作和声音来再现拍照时前后的片刻。

了解相关设计指导，请参阅 [Live Photos](#)。

3.2 音频

用户通过音量键、静音键、耳机声控和屏幕上的音量调节滑块控制声音。很多第三方配件也有声控功能。音频可以通过内部或外部扬声器、耳机输出，甚至通过支持 AirPlay 或是蓝牙的设备来无线输出。无论声音是你 App 的重要功能还只是一个点缀，你都应该知道用户对声音有什么期待并且去满足他们。

静音

用户将他们的设备调至静音来避免被突发的声响（比如电话铃声和短信提示声）打扰。他们也想要关闭没有意义的声音，包括按键声、音效、游戏背景音乐以及其它音频反馈。当设备被设置成静音后，只能播放用户确认打开过的声音，比如媒体播放中的声音、闹钟、音频和视频信息。

音量

无论是使用设备按键或是屏幕中的滑块，用户都期望音量的变化会改变整个

系统音量，包括音乐声和 App 内的音效。但是提示铃声音量是例外，它只能在音频没有播放的情况下，被单独调节。

耳机

在私密环境中，用户使用耳机听声音，并且能够解放双手。当插入耳机时，用户希望声音能够自动继续播放而不被中断。当拔掉耳机时，他们希望播放能够立即暂停。

设计良好的音效体验

有时自动调整音量，但不是总体音量：你的 App 可以调整特定的音量来达到很棒的音响效果。但是最终的音量还是由系统音量来控制。

可以允许重置音频：用户经常会想要选择一个不同的音频输出设备。比如，他们会想要用客厅的立体音响、车载收音机或是苹果电视来听音乐。请支持这个功能除非你有十足的理由不这么做。

使用系统自带的音量窗口来调节音量：音量窗口是调整音频的最好方法。这个窗口是可以自定义的，它包括一个音量的滑块，甚至还包括了一个重置音频输出的控件。

了解开发者指南，请参阅 [MPVolumeView](#)。

使用系统自带音效来实现短音和振动：

了解开发者指南，请参阅 [System Sound Services](#)。

如果声音对你的 App 十分重要, 请先设置音频类型: 不同的音频类型中, 有的允许声音被静音键静音、有的与其它声音混合、或是允许 App 在后台时播放。根据不同的类别和当前设备的音频状态来选择合适的方案, 然后将其配置到你的音频中去。例如, 非必要情况下, 请不要打断用户正在收听的来自另一个 App 的音乐。总之, 当你的 App 在运行时, 最好不要改变其原有类型, 当然那种需要录制和重放不同时间音频的 App 除外。

开发者指南, 请参阅 [Audio Session Programming Guide](#)。

类别	含义	特性
个人环境	声音不是必要的, 但是它会使其他音频静音。例如有背景音乐的游戏。	受静音键控制。 不和其他声音混合。 不在后台播放。
环境	声音不是必须的, 也不会使其他音频静音。例如一个游戏允许用户在游戏时播放另一个 App 的音乐来代替游戏本身的背景乐。	受静音键控制。 可以和其他声音混合。 不在后台播放。
播放	声音是必要的而且可能会和其他声音混合。例如, 有声读物或者外语学习的教育 App , 用户可以离开 App 后也能听到其内容。	不受静音键控制。 可能也可能不与声音混合。 可在后台播放。
录制	声音可录制。例如, 一个做笔记的App 可以提供录音模式。如果这种App 需要允许用户播放录制的笔记, 那么它有可能要把类别转换到回放类。	不受静音键控制。 不和其他声音混合。 可在后台录制。
播放并录制	声音可能会同时进行录制和播放。例如, 一个拥有音频短信或者视频电话功能的 App 。	不受静音键控制。 可能也可能不与声音混合。 可在后台录制和播放。

当时可以重新播放被打断的音频: 有时正在播放的音频会被另一个 App 的音频中断。暂时性中断 (如来电铃声) 被认为是可恢复的。永久中断 (如 Siri 打开的音乐播放列表) 被认为是无法恢复的。当一个可恢复的中断发生时, 你的 APP 应该在中断结束时自动重新播放被打断的音频。例如, 一个正在播放配乐的游戏和一个播放音频的多媒体 App 都应该可以恢复播放。

让其他 App 知道何时将暂时性的音频播放完毕：如果你的 App 可能会暂时中断其他 App 的音频，它应该对音频会话进行适当的标记，这样其他 App 就能知道何时可以安全的恢复。

更多开发者指南，请参阅 [AV Foundation](#) 中的 [AVAudioSessionSetActiveOptionNotifyOthersOnDeactivation](#)。

只在特定时间对声音控件有反应：不管你的 App 是在前台还是后台运行，用户可以从你 App 的外部控制音频播放，比如控制中心或耳机控制。如果你的 App 正在音频相关的环境中播放音频，或者连接到一个支持 AirPlay 的设备上，那么它可以对音频控制做出回应。否则，当控件激活时，你的 App 不能停止其他 App 正在播放的音频。

不要重新定义声音控件：人们希望音频控制在所有 App 中都能保持一致。永远不要重新定义声音控件。如果你的 App 不支持相应控件，那么它就不应该对它们做出响应。

3.3 身份验证

可以通过让用户进行身份验证的条件来为他们提供有价值的服务，例如个性化体验、访问附加功能、购买内容或同步数据。如果你的 App 需要认证，那保持登录过程的快速、简单和自然，就不会让你的 App 的体验感变差。



如果您使用「Apple 账户登录」，那么使用自动填充密码：此功能自动生成并填写密码和安全代码，以便人们可以在验证界面中花费更少的时间。所有应用程序都应该支持此功能。开发者请参阅 [支持密码自动填写](#)。

将登陆尽量往后排序：当人们在做一些有用的事情之前被迫登录时，他们往往会选择放弃这样的 App。在强制用户登录前给他们一个爱上你的 App 的机会。在购物 App 中，让用户可以在启动时能立即浏览你的商品，并且只有在他们准备好购买的时候才需要登录。在一个流媒体 App 中，也在登录前让用可以搜索你的内容，看看你的 App 能提供些什么。

解释身份验证的优点，以及如何注册：如果你的 App 需要验证，请在登录屏幕上展示简短友好的说明描述使用这个流程的原因及好处。另外，请记住，并不是所有使用你的 App 的人从一开始就有账户。确保你描述了如何获得一个账户，或者提供一个简单的 App 内的注册方式。

显示合适的键盘类型来减少数据输入：例如要求填写 Email 地址时，显示 Email 键盘屏幕，其中要包含常用数据输入的快捷键。有关相关指导，请参阅 [键盘](#)。有关可用键盘类型的完整列表，请参阅 [UI文本输入特性](#) 中的 [UI 键盘类型](#)。

永远不要使用“密钥”这个术语：当生物识别认证失效时，密钥是用于解锁用户的 iOS 设备，并与 Apple Pay 进行身份验证的。

了解有关 Apple Pay 验证设计指南，请参阅 [Apple Pay](#)。

脸部识别和触控



尽量让你的App支持生物识别认证：Face ID 和 Touch ID 是人们所信任的比较安全、熟悉的认证方法。如果用户启用了生物识别认证，你可以假设他们理解它的工作原理，喜欢这份便利，并且愿意随时使用它。请记住，用户可能会选择在他们的设备上禁用生物识别认证，因此你的 App 要准备好应对这种情况。

向用户展示一个简单的验证方法：当人们不必选择如何进行身份验证时，这是最直观的一种方法。给他们一个简单的选项，比如 Face ID，在初始方法失败时，请求用户名和密码，在返回刚才的操作。

仅在响应用户操作时启动用户认证：选择一些显眼且明确的操作指示，比如点击一个按钮，确认用户是想主动进行身份验证的。在 Face ID 的情况下，它也提高了用户面对摄像头的可能性。

始终标识身份验证方法：例如，使用 Face ID 在你的 App 上登录的按钮，应该被命名为“用 Face ID 登录”，而不只是“登录”。

准确使用验证方法：不要在支持 Face ID 的设备上引用 Touch ID。同样，也不要在支持 Touch ID 的设备上引用 Face ID。要检查设备的功能并使用适当的术语。

了解有关开发者指南，请参阅 [L^ABiometryType](#)。

总之，不要在你的 App 中提供选择生物识别身份验证的设置：如果在系统层面启用了生物识别身份验证，那么就假定用户想要使用它。如果你实现了一个 App 特定的设置，那么你的 App 会要求用户进行生物识别认证，但是实际上在系统范围内它是被禁用的。

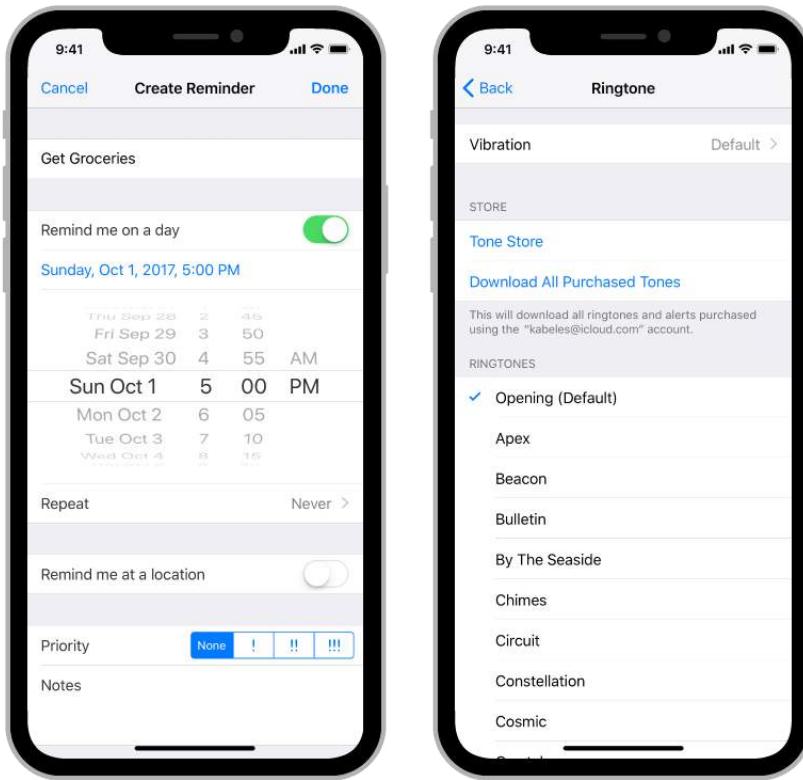


不要使用自定义图标来标识系统身份验证功能：当用户看到类似 Touch ID（指纹）和 Face ID 的系统图标时，他们会以为要进行身份验证。而使用相同图形的自定义图标则会让用户产生混乱，尤其是当这些图标，变着颜色和大小，出现在不符的使用环境中，就像变成了一个按键，或者出现在 App 的设置上。

开发者指南，请参阅 [Local Authentication](#)。

3.4 数据键入

无论是点击界面元素还是使用键盘，键入信息都是一个乏味的过程。当一款 App 还没有提供实际的帮助，就要求用户大量的键入内容，会导致进程变慢，用户就可能会很快失去耐心，甚至完全放弃这个 App。



尽量显示选项：数据输入高效化。例如，考虑使用选择列表来代替打字，因为从提前设定好的选项中进行选择，要比打字容易多了。

尽可能的从系统中获取信息：不要强迫用户提供那些可以自动或是在用户许可内就能获取的信息，比如联系人或是日历信息。

提供合理的默认值：尽可能地预填可能性最大的信息值。提供准确的默认值可以进行快速决策。

只有在收集完必要的信息之后才能进行下一步：在启动“下一步”或“继续”按钮之前，确保所有需要的字段都有值。使用可视化按钮，提示用户可以继续进行下一步。

立刻验证填写信息：当你在填写完一份冗长的表格后，又不得不回到前面去纠正错误是让人十分沮丧的。尽可能在输入后立即检查字段值，这样用户就可以马上纠正它们（例如输入用户名后立马检查格式和是否被注册，而不需

要在输入完密码后再一起验证)。

只有在必要时使用文本信息：只要在真正需要的信息，才使用字段表达。

将信息列表化以高效预览：在列表和集合中，选择一个选项是更容易的。考虑将信息列成表，以字母顺序排序或以另一种逻辑方式排序，从而可以快速扫视和选择（译者注：如商品类目以字母排序的方式在列表中展现出来）。

在文本框中显示提示，以助沟通：当字段中没有其他文本时，文本字段可以包含诸如“电子邮件”或“密码”之类的占位符文本。当占位符文本足以表述清楚的话，不要再使用单独的标签来进行描述。

3.5 拖放

只要一根手指，用户就可以将选定的照片、文本或其他内容，从一个位置拖到另一个位置来进行移动或复制，然后松开手指将其放下。



触摸并按住选择的内容使它看起来会上升并附着在用户的手指上。当内容被

拖动时，动画和视觉提示要识别你想放置的位置。当无法放下，或者拖拽只能复制而不是移动时，该系统会显示别的标识来提醒用户。

请参阅 [UIKit 中的 Drag and Drop](#)。

原位置和目的地

拖放包括将选中的内容从原位置转移到目的地。这些位置可以在同一集合中，如文本视图，或分屏模式下两端的文本视图。例如，在 Notes 中，用户可以将选中的文本拖到同一条笔记中的新位置。在 Reminders 中，用户可以将单个提醒从一个列表中拖出来，并将其放入另一个列表中。

在 iPad 上，原位置和目的地的位置也可以在不同的 App 中，从而产生跨 App 的交互，比如从 Safari 的网页中拖出一张照片到 Mail 里的新邮件中。

在拖拽内容的同时，用户可以通过[多任务处理](#)、退出到主屏幕或从屏幕底部向上滑动显示快捷键来访问另一个 App。

注意：在App程序之间拖放的内容是复制文件，而不是内容本身（译者注：即拖动只是将内容复制到另一个应用中而不是移动存储位置）。

支持拖放功能

拖放是一种高效的、直观的功能，用户期望在整个系统中都能实现。如果你的App包含或生成了文本，照片，视频，音频，或者其他用户想要移动，复制或插入的内容，你的App应该支持拖放功能。

所有可选择和可编辑的内容都能进行拖放：可选择的内容应该是可拖动的，

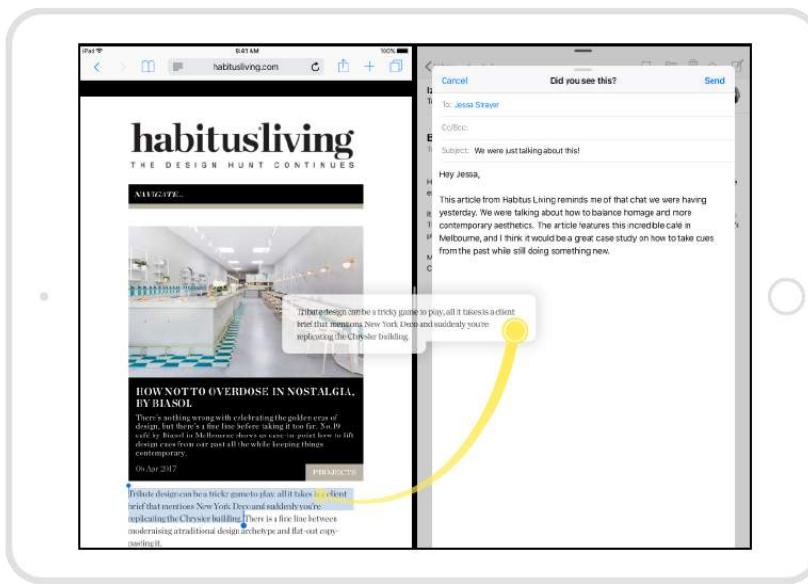
可编辑的内容也是可以放置内容的。还要确保你的 App 能够支持在这些区域复制粘贴。

允许将内容放到控件上：总之，配置控件要支持数据输入或选择(比如文本字段)接受可放置的内容。

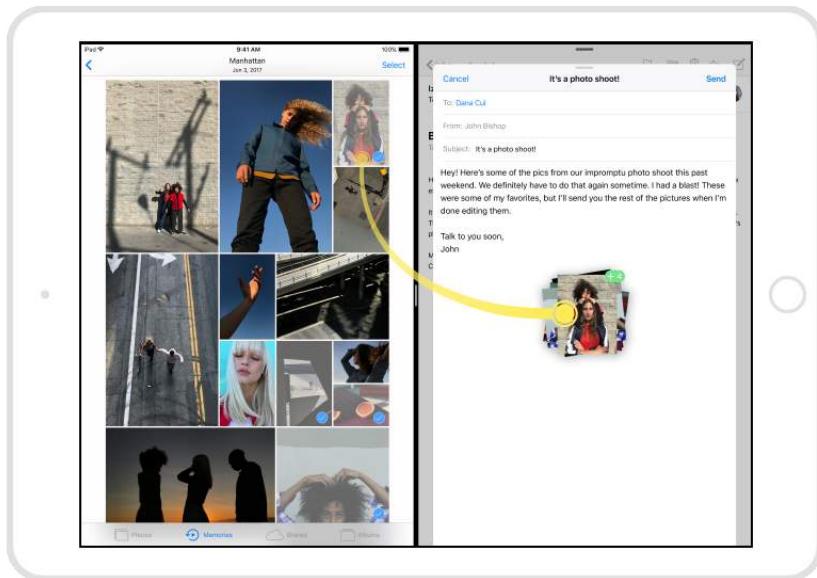
尽可能使用标准文本视图和输入框：系统自带的这些组件都包含对拖放功能的原生支持。

了解更多相关指导，请参阅 [Text Fields](#) 和 [Text Views](#)。

了解更多开发者指南，请参阅 [UITextField](#) 和 [UITextView](#)。



为提高效率，考虑支持多项拖放：在很多 App 里，用户可以用一根手指拖动一个内容，同时用另一根手指轻敲其它内容，就会在选择的第一个内容的手指下方显示出堆叠效果。然后，用户可以将它们像组一样拖放到指定的位置。例如，主屏幕允许用户根据这个方法选中多个 App 程序图标，并将其拖放到一个文件夹中。在 APP 中，如“照片”，提供了用户可以在拖拽之前选择多个对象的选择模式。



确定在你的 App 中拖放内容是移动还是复制：一般来说，当原位置和目的地的载体是相同时(在文档中拖动文本)，才可以移动，而当它们不同时(在文档之间或 App 程序之间进行拖动)，则使用复制。当然，情况并非总是如此，重点是拖放应该是直观的。在“提醒”中，列表之间拖动操作是一种移动而不是复制，因为这是用户想要的。而在 App 程序之间拖放内容就总是复制了。

用户是应该可以撤销拖放的：通常情况下，当用户无意中将内容放到错误的位置时，他们应该能够使用撤销来将App返回到之前的状态。也就是说，被放置的内容可以被移除，如果被放置的内容是从你的App程序里移出的，那么它应该恢复到原来的位置。

考虑用一下弹簧加载：有了弹簧加载，用户可以通过拖动已选择的内容，并在不放置的情况下暂停，来激活某些控件，比如按钮和分栏组件。例如，在邮件中，选择的消息可以被拖动到导航栏的 Back 按钮上，以到达邮箱里的其他位置。永远不要让弹簧加载成为激活控件的唯一方法。把它当作一种可以被发现的小惊喜。在大多数情况下，一个弹簧加载的控件也应该可以响应

一个点击手势。

开发者指南, 请参阅 [UISpringLoadedInteraction](#)。

提供可拖动的内容

可以自定义拖动项的预览: 通常, 在用户手指下显示被拖动的预览效果应该是半透明内容, 表明拖动正在进行中, 并使用户能够看到被拖动内容下面的目的地。

尽可能的提供多个拖拽数据的显示类型, 从高保真到低保真排序: 例如, 当提供线条稿时, 你的 App 可以按照顺序依次提供一个 PDF 矢量显示, 一个具有透明度的无损 PNG 图像, 以及一个没有透明度的有损的 JPEG 图像。这样, 用户就可以选择最高质量的显示来导入到目的地。

将自定义对象的本地版本作为最丰富的数据形式: 例如, 一个允许人们拖拽图表的 App 应该首先显示本地的图表对象。然后, 它应该为那些不支持图表对象的 App 提供可替代的图像版本。

当你 App 的内容传输会比较耗费时间或资源密集的时候, 文件要提供程序扩展功能: 文件提供程序扩展, 并确保完成, 即使你的 App 不再运行也能做到。请注意, 只有用户放下内容后传输才会开始。

了解更多开发者指南, 请参阅 [NSFileProviderExtension](#)。

当你的 App 需要时间传输内容时请提供进度指示: 如果内容需要下载或大文件需要时间复制时, 请提供进度信息。至少提供内容的总大小, 这样就可以计算出剩余的时间, 并显示恰当的进度指示器。

了解更多开发者指南，请参阅 [NSProgress](#)。

接收放下的内容

使用视觉提示来识别潜在目的地，并预览放置内容的效果：高亮显示、插入点光标和动画都是识别目的地的好方法。当内容被拖过来时，视图可以微妙地闪烁并改变颜色，或者段落可以分开为拖动的图像腾出空间。当屏幕上有一个可能的目的地时，一次只识别一个。如果原位置和目标容器是相同的，那么可以不需要高亮显示，除非内容完全从原位置中拖出来，然后重新进入。当内容被放下时或不再位于目的地之上时，确保不显示高亮。

自动滚动目的地的内容：当内容被拖到目的地的边界之外时，你的 App 可能需要判断滚动显示当前视图遮挡的内容，或者允许用户继续拖动到完全不同的目的地。如果你的 App 让用户进行滚动拖放，那就定义一个区域，当拖拽的项目被放置在上面时，这个区域会自动滚动。例如，当内容被拖到正文区域的顶部或底部时，邮件中的长草稿会自动滚动。标准文本视图和输入框会自动采用这种方式。

提取和展示要能展现出放置的丰富内容：例如，你的 App 可以为一个图表提供的几种表现形式。如果你的 App 支持图表，它可以提取并显示出原始图表对象。如果你的 App 不支持图表，它可以提取并显示图表的图像版本。**只提取放置内容的相关部分：**例如，如果用户想将某个联系人拖到邮件的收件人中，只需提取名称和电子邮件即可，而不用联系人的地址信息。

当内容放置后，在表视图和精选视图中显示占位符：占位符会暂时指示当内容完成传输后，将停留在哪里。

在放置内容需要时间进行传输时显示进度：默认情况下，当 App 程序之间发生耗时的转移时，系统会显示一个 App 模式警报。可以使用自定义进程的显示方式，比如在表视图或集合视图中显示占位符的进度指示器，这样就不会阻止用户使用你的 App 了。请注意，只有用户放下内容后传输才会开始。

为放置的内容启动一个进程时提供反馈：如果用户将内容放置到发起任务的一个控件上，例如，将视频上传至共享站点，就会显示任务已经开始并通知用户其进度。

放置失败时通知用户：如果放置的内容不能插入，可能是因为文件传输中断，要通知用户放置内容失败。

用适当的样式来放置文本：当原位置和目的地支持相同样式的文本属性时，放置的文本应该保留其原始字体、字型、大小和其他属性。否则，放置文本应该采用目的地的样式。

提供一种直观的方式，当用户不能立即撤销拖放时，可以选择退出：例如，一个分享性的 App 会在放下内容之前提供一个中间的附加表。这个附加表可以用来提供其他内容，比如状态消息（译者注：如内容数量和大小），同时还提供一个取消按键。当照片被拖进共享的照片流时，App 就显示了这种方式。

3.6 反馈

反馈让用户知道 App 现在在做什么，发现他们下一步应该做什么，并且了

解操作的结果。



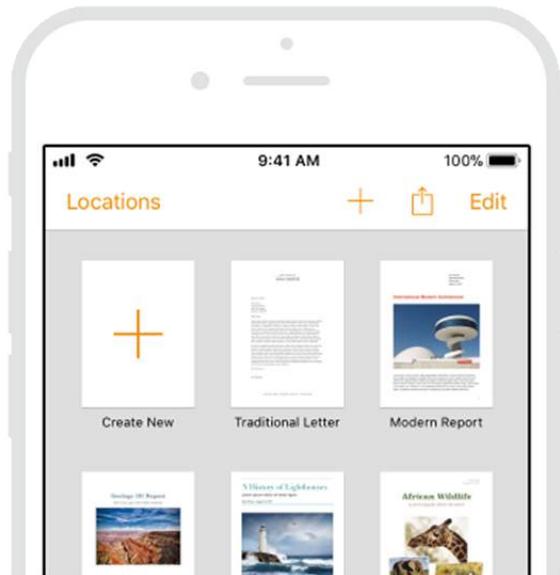
将状态和其他类型的反馈整合到你的界面中：理想情况下，用户可以在不操作或不被打扰的情况下获得重要信息。例如，邮件在通过邮箱导航时，巧妙地显示了工具栏中的状态信息。这些信息不会和屏幕上的主要内容抢风头，但用户可以在任何时候快速浏览。

避免不必要的警报框：警报框是一种强大的反馈机制，但应该只用于传递重要的信息。如果用户看到太多包含无关紧要信息的警报框，他们很快就会学会忽略以后的警报。

了解更多指南，请参阅 [Alerts](#)。

3.7 文件处理

用户在创建、查看和操作文件时不用考虑文件系统。如果你的App需要运行文件时，尽可能地减少文件处理的时间。



让用户相信文件即时保存，除非主动取消或删除：一般来说，不要让用户去即时保存文件。相反，在用户打开和关闭文件，以及切换到另一个 App 程序时，会定期自动地保存更改。在某些情况下，例如在编辑现有文件时，保存和取消选项可能仍然是有意义的，因为它们可以帮助确认何时编辑的内容已经被保存。

不要提供创建本地文件的选项：用户通常希望所有的文件都可以在他们的所有设备上使用。只要有可能，你的 App 就应该通过 iCloud 等服务来支持云文件存储。

设计一个直观的图形文件浏览界面：理想情况下，使用用户熟悉的系统文档应用来浏览。如果你设计了一个自定义的文件浏览器，请确保它是直观且高效的。文件浏览器在高度图形化的情况下运行得最好，也提供了文件的可视化展示。为了高效引导用户，可以考虑提供一个新的文档按键，这样用户就不必去其他地方创建新文档了。

让用户在不离开 App 程序的情况下预览文件：你可以使用 Quick Look 功能让用户查看来自 Keynote、Numbers 和 Pages 的内容，以及 PDF 文

档、图片或某些其它格式的文件，即使你的 App 并没有真正打开它们。

请参阅 [Quick Look](#)。

与其他App共享文件：如果可以的话，你的 App 可以通过文档的扩展功能（[document provider extension](#)）与其他 App 共享文件。也可以让用户浏览和打开其他 App 的文件。

了解更多开发者指南，请参阅 [Document Picker Programming Guide](#)。

3.8 手势

用户通过在触摸屏上使用手势来与 iOS 设备交互。操作手势与内容产生了紧密的个人联系，增强了对屏幕的直接操纵感。

一般使用标准操作手势：用户熟悉了标准的手势，就不喜欢被迫学习不同的方法来做相同的事。在游戏和某些沉浸式App中，自定义手势是这种体验的乐趣之一。在一般 App 中，最好使用标准的手势（[standard gestures](#)），这样用户就不用费力去发现或记住它们了。

避免使用标准手势执行非标准操作：除非你的 App 是一个极具可玩性的游戏，否则重新定义标准手势会变得混乱和复杂。

不要禁用系统的屏幕边缘手势：除了标准的手势之外，一些额外的手势还可以调用全局操作，比如在屏幕的边缘通过滑动屏幕来显示主屏幕(在支持的 iphone 上)、控制中心、通知中心和 Dock (在 iPad 上)。用户依靠这些手势来操作每一个 App 程序。在极少情况下，像游戏这样的沉浸式 App 可能

需要自定义的屏幕边缘手势，这些手势优先于系统的手势操作——第一次滑动调用此 App 的自定义手势功能，而第二次则调用系统手势。这种行为(被称为边缘保护)应该要尽量避免，因为这会使得用户难以访问系统操作。

基于界面的导航和操作提供补充性的快捷手势，而不是替换：只要有可能，提供一种简单、直观的方法来引导或执行操作，即使这意味着需要额外的点击。许多的系统 App 提供了清晰可点的返回上一页的按钮的导航栏。但用户也可以通过从屏幕的一侧滑动来返回。在 iPad 上，用户可以按下 Home 键退出到主屏幕，或是使用四指捏合的手势。

使用多点手势来增强 App 的体验：虽然涉及多个手指同时操作的手势不适用于每一个 App，但是他们能够丰富一些 App 的体验，譬如游戏和绘图 App。例如，一个游戏可能包含多个屏幕控制，比如操纵杆和发射按钮，就可以同时进行操作。

了解更多开发者指南，请参阅 [UIGestureRecognizer](#)。

标准手势

用户通常期望以下的标准手势在整个系统和每个 App 程序中都是相同的。



点击：激活一个控件或者选择一个对象。

拖拽：让一个元素从一边移动到另一边，或者在屏幕内拖动元素。



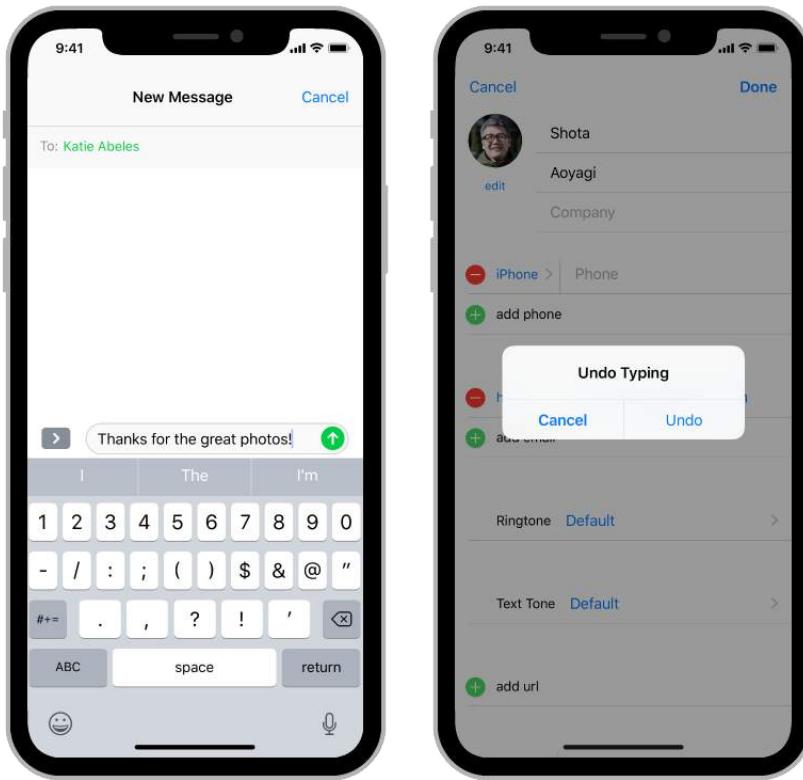
滑动：快速滚动或平移。

横扫：单指返回上一页，显示分屏视图中的隐藏视图，滑出列表行中的删除按钮，或在轻按显示操作列表。在 iPad 中四指操作用来切换 App。



双击：放大并居中内容或图片，或者缩小已放大过的。

捏合：向外张开时放大，向内捏合时缩小。



长按: 当在可编辑或可选文本中执行时，显示用于光标定位的放大视图。在某些与集合视图类似的视图中操作，进入对象可编辑的状态。

摇晃: 撤销或重做。

旋转: 旋转图像或者视图。

3.9 触觉

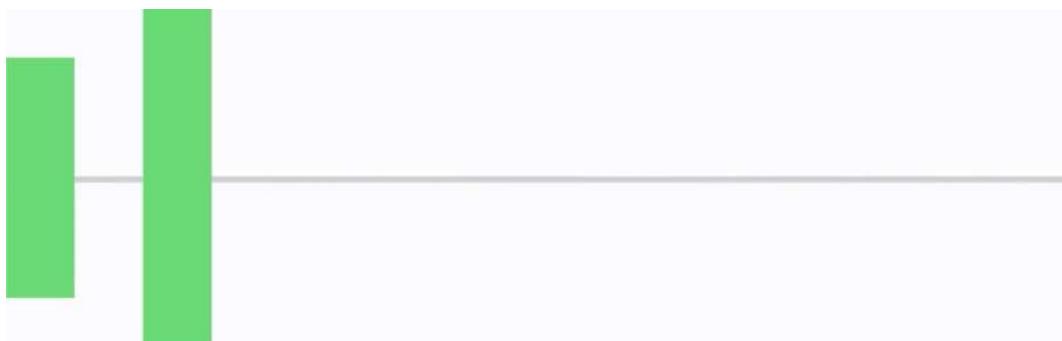
Haptics 线性马达将用户的触觉联系了起来，优化了与屏上界面的交互体验。例如当确认 Apple Pay 的交易时，除了提供视觉和听觉上的反馈，系统还会进行振动。Haptics 同时能够增强触控手势，诸如在选择器内滚动或者切换开关时提供振动。

在支持 Haptics 的 iPhone 型号上，您可以通过多种方式在应用内添加它：

- 使用标准的 UI 组件，如[开关](#)，[滑块](#)和[选择器](#)，它们默认提供 Apple 设计的系统振动。
- 必要时，使用反馈生成器来触发几种在通知、碰撞（如触底和触顶）或 选择中的预设振动模式。开发者请参阅 [UIFeedbackGenerator](#)。
- 如果您需要更深入的控制 Haptics，可以编写并使用您自定义的振动 模式。请参阅 [Creating Custom Haptic Patterns](#)。

当您使用系统预设的震动时，iOS 会管理反馈的强度和模式。例如，打开或 关闭开关会自动触发一次轻微的振动；而通知会触发如下所示的振动：

通知 - 成功



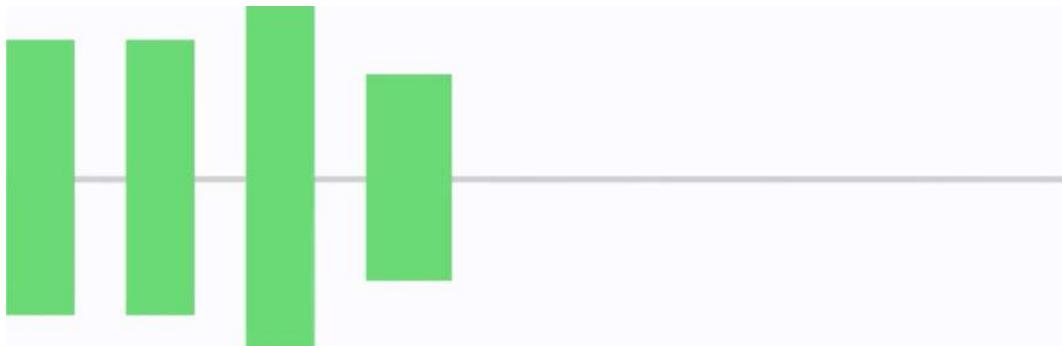
表示任务或操作（例如存入支票或解锁车辆）已完成。

通知 - 警告



表示任务或操作（例如存入支票或解锁车辆）已出现某种警告。

通知 - 失败



表示任务或操作 (例如存入支票或解锁车辆) 失败。

碰撞 - 轻



提供补充视觉的物理隐喻。例如，当视图滑动到头或两个对象相撞时，用户可能会感到“啵”的一声。

碰撞 - 中



提供补充视觉的物理隐喻。例如，当视图滑动到头或两个对象相撞时，用户可能会感到“啵”的一声。

碰撞 - 重



提供补充视觉的物理隐喻。例如，当视图滑动到头或两个对象相撞时，用户可能会感到“啵”的一声。

选择



表示选项正在快速更改。例如，用户在滚动选择器时感觉到轻敲。此反馈旨在将一些列离散的值联系起来，而不是进行选择或确认选择。

无论是使用系统提供的振动还是自定义的振动，最关键的是，振动是为了提升用户的体验，而不是分散用户的注意力。

Haptics 的设计

设计能够与人们触觉相连的界面，可以把它们所熟悉的物理世界搬进你的 APP。无论您如何在应用中使用振动，以下原则都可以帮助您为人们提供有吸引力的体验。

在每个振动和触发物之间建立清晰的因果关系：理想情况下，用户能够清楚地知道您的应用为什么会出现振动。如果一个振动无法加强因果关系，那么它可能会让人感到困惑，而且是无来由的。

使用振动可以补充应用中的其他反馈：当您应用程序的视觉，听觉和触觉反馈协调一致，就像在物理世界中一样，那么用户的体验就更加连贯，用起来更加自然。

请谨慎地使用振动：例如，当界面提供了很长一串值时使用振动；而如果你想用振动来为你的 APP 增添新鲜感，那么往往会觉得花里胡哨的。此外，宁愿将振动添加到少量重要的交互中，而不是大量琐碎的交互一股脑儿都使用振动，那样会让用户不知所措。

让振动始终如一：一致的振动反馈能够帮助人们在特定的操作和特定的振动模式间建立一种触觉感知上的联系。举个例子，如果在你的游戏中，当游戏角色任务失败时触发一种振动，用户会将这种振动与负面的结果联系起来，而一旦你将同样的振动赋予一个正面的结果比如升级，那么用户就会感到困惑。

不要过度使用振动：有时候，偶尔出现的振动能够让用户觉得合理，但如果频率一旦变高，它就会变得烦人。通常来说，不要因为设计了一个拖沓或者重复的震动反馈，而让体验变差。最好的振动体验，是让人意识不到它的存在，但一旦将它关掉又会想念它的状态。

一定要测试应用中的振动：不同的用户对振动有不同的偏好和敏感度，所以你需要尽可能多的用户参与测试振动反馈。

保证振动可关闭：允许用户关闭振动，并确保即使没用振动，用户依然能够

享受你的应用。

在游戏中，思考有哪些方法能够通过使用自定义振动来增强玩家的体验：除了在玩家与游戏内物品进行交互时触发振动之外，你还可以考虑将振动与游戏内的事件结合起来。举个例子，振动既可以大的方面提升体验比如碰撞或者打击，也可以从微妙的地方提升体验比如渐近的脚步或者逼近的危险。

请注意，振动可能会影响其他方面的体验：Haptic 线性马达会产生足够的物理力量让用户感知到设备在振动，但是确保振动不会影响到与相机、陀螺仪或者麦克风等其他方面相关的体验。

创建自定义的振动模式

自定义的振动模式能够模拟诸如弹弓释放或者车辆驶过减速带的触感。自定义振动同样可以根据用户的输入情况，或者不同的情景动态变化。比如，当游戏角色从树上跳下时，玩家感受到的振动比平地起跳更强。

在 iOS13 及更高版本的系统中，Haptics 核心提供了两种生成自定义震动的基础构建模块。

- 瞬态事件，像点击或者脉冲那样的短促、紧凑的振动。比如在主屏上点击手电按钮。
- 连续事件，感觉像是持续的振动，例如消息中激光效果的体验。

除了上述两种构建模块，你还能控制振动的锐度和强度。锐度可以这样理解：它把振动抽象为一种能够让触觉感知到的波形。通过指定锐度，你可以将你想象中的振动体验传达给系统。例如：你可以利用锐度值来表达柔软、

圆润，或有机，或清脆、精确，亦或是机械感的振动。顾名思义，强度代表了震动的力度。

通过结合瞬态和连续事件，改变锐度和强度，并结合可选的音频内容，您可以创建各种不同的振动体验。开发者请参阅 [Core Haptics](#)。

3.10 近场通信技术 (NFC)

NFC 使设备能够在几厘米的范围内进行无线信息交换。在支持此项功能的设备上运行 iOS App，可以使用 NFC 扫描来读取现实世界相关的电子标签上的数据。例如，用户可以扫描一个玩具来连接视频游戏，购物者可以扫描店内的标志来拿到优惠券，或者零售员工可以扫描产品来跟踪库存。

开发者请参阅 [Core NFC](#)。

应用内标签读取

一个 App 可以支持单个或多个对象的扫描，当用户需要扫描某件东西时，系统会弹出一个扫描表。



不鼓励用户接触物理对象：要扫描标签，iOS 设备只需简单地靠近标签，而不需要触碰。当要求用户扫描对象时，使用“扫描”和“靠近”等术语，而不是“点击”和“触碰”。

使用通俗易懂的术语：一些人可能不熟悉 NFC 这个概念。为了让它变得通俗易懂，应避免提及技术上的、面向开发者的术语，比如核心 NFC 、近场通信、NFC 和标签。相反，要使用大多数人都能明白的友好会话式的术语。

应该这样	不应该这样
请扫描【对象名称】	请扫描 NFC 标签
请把你的 iPhone 靠近【对象名称】以了解更多。	请使用 NFC 扫描，点击手机上的【对象】

为扫描提供简洁的指导说明：提供完整的带有结束标点符号的句子。确定要扫描的对象，并适当地修改文本以便进行后续扫描。保持文本简短，避免截断。

应该这样	不应该这样
把你的 iPhone 靠近【对象名称】以了解更多。	现在，把你的iPhone 靠近另一个【对象名称】

了解更多开发者指南，请参阅 [Core NFC](#)。

背景标签读取

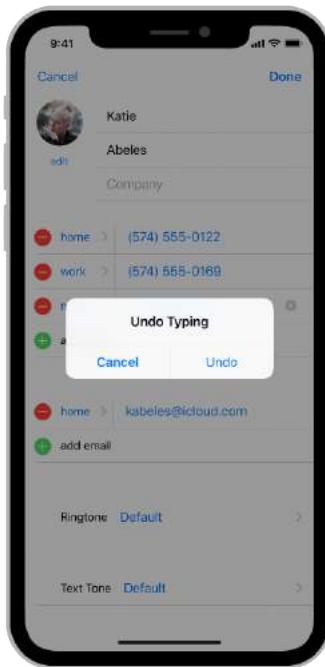
背景标签读取允许用户任何时候都能扫描 NFC 标签，不需要事先打开 APP 再启动扫描。在支持背景标签读取的设备上，系统会自动地在你的附近搜寻兼容的标签，不管屏幕有没有点亮。在检测标签并与某一个 APP 匹配之后，系统将会显示一条通知，用户可以点击这条通知来将标签数据发送到应用中处理。注意，背景读取会在以下场景禁止使用：已启动了 NFC 扫描页的情况下；钱包或者 Apple Pay 正在被使用的情况下；相机正在被使用的情况下；设备处于飞行模式；设备重启后处于锁定状态下。



同时支持背景标签读取和应用内标签读取。您的应用仍然需要提供应用内扫描标签的方式，因为有些设备并不支持背景标签读取。

3.11 撤销和重做

许多 App 允许用户通过摇晃设备来撤销和重做某些操作，比如打字或删除。当以这种方式启动时，会出现一个警告框提示用户确认或取消撤销（重做）。



简要描述一下将要撤销或重做的操作：撤销和重做的警告提示框标题会自动地包含“撤销”或是“重做”这样的前缀（以及后面的空格）。你需要在前缀后面提供额外的一两个词语用来描述什么会被撤销或是重做。例如，你可以创建警报提示，例如“撤销名称”或“重做地址更改”。

如果你已经使用摇晃手势进行撤销和重做，则不要将其用于其他操作：尽管你可以通过编程方式给摇晃手势赋予多种功能，但却冒着让用户困惑的风险，这样就会让你的 App 更难用。

节制地使用撤销和重做按钮：当 App 提供多种方法来执行相同的任务时，会让用户感到困惑。如果你的 App 真的需要专有的撤销和重做按钮，请使用标准系统提供的图标，并将它们放在一个预期的位置，比如导航栏中。

只在当前情境中执行撤销和重做操作：撤销和重做必须对当前的（而非之前

的) 情境有明确直接的影响。

了解更多开发者指南, 请参阅 [Undo Architecture](#)。

第四大章

系统功能

System Capabilities

4.1 增强现实

ARKit是苹果公司的增强现实（AR）技术，提供身临其境，引人入胜的体验，将虚拟物体与真实世界无缝融合。AR应用通过设备的相机在屏幕上呈现了生动的物理世界视图。三维虚拟物体叠加在这个视图上，营造它们实际存在的感觉。用户可以重新调整设备的方向，以便从不同的角度探索对象，尽可能符合现实经验，可以使用手势和移动与对象进行交互。

设计引人入胜的体验

使用整个显示器：投入尽可能多的屏幕来查看和探索物理世界和应用中的虚拟对象。避免让控件和信息扰乱屏幕，增加沉浸式体验。

在放置逼真的物体时创造令人信服的错觉：并非所有AR体验都需要真实的虚拟对象。那些确实应该存在于物理环境中的物体，为获得最佳效果，需要设计具有逼真纹理的细腻的3D素材。然后使用ARKit提供的信息将物体定位在检测到的真实世界表面上，并适当缩放对象，在虚拟物体上提供环境光照

条件，在真实世界表面上投射虚拟物体阴影，并在相机位置改变时更新视觉效果。

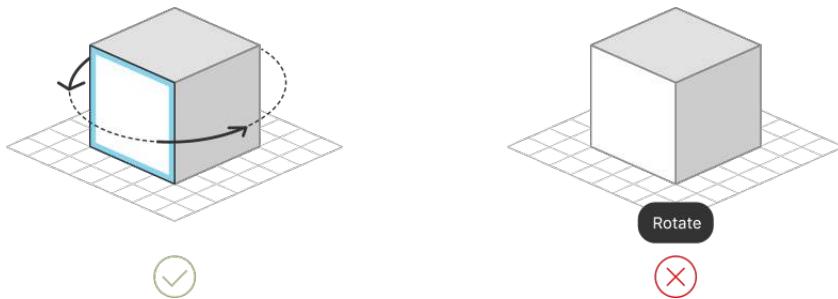
预料到人们会在不适合AR的环境中使用你的应用：人们可能会在没有足够空间移动或没有大的平坦表面区域的位置打开你的应用。所以要预先向用户明确传达我们的要求和期望，并且预料到这些具有挑战性的场景，可以考虑提供可用于不同环境的各种功能。

注意用户的舒适度：长时间地将设备保持一定的距离或角度可能会导致用户疲劳。所以要考虑人们在使用你的应用时需要怎样手持设备，并找到一种让用户拥有愉快体验的解决方法。例如，默认情况下，你可以将对象放置在距离较近的位置，这样就没必要再移动设备使它更靠近对象。当一个游戏面临短暂的停机时间时，它可以短暂地保持一定的优先级。

如果你的应用鼓励用户发生动作，请逐步介绍：当用户一进入游戏中时，没必要直接给用户来一发虚拟炮弹。先要给他们时间先适应，然后再逐步鼓励动作的发生。

注意用户的安全：如果其他人或物体在附近，大幅度的移动可能会产生危险。你需要考虑让你的应用安全运行的方法。一个游戏不鼓励大幅度和突然的动作。

使用音频和触觉反馈来增强沉浸式体验：声音效果或碰撞感是确认虚拟物体与物理表面或其他虚拟物体接触的好方法。在虚拟现实游戏中，背景音乐可以帮助用户沉浸在虚拟世界中。更多相关指南，请参阅 [Audio](#) 和 [Haptic Feedback](#)。



尽可能在上下文中提供提示：例如，在物体周围放置3D旋转指示器比叠加在图层上的文本指令更直观。但在表面检测之前，或者用户没有响应上下文提示，文本覆盖提示需要被授权。

考虑引导用户找到屏外虚拟物体：有时可能很难找到位于屏幕外的物体。

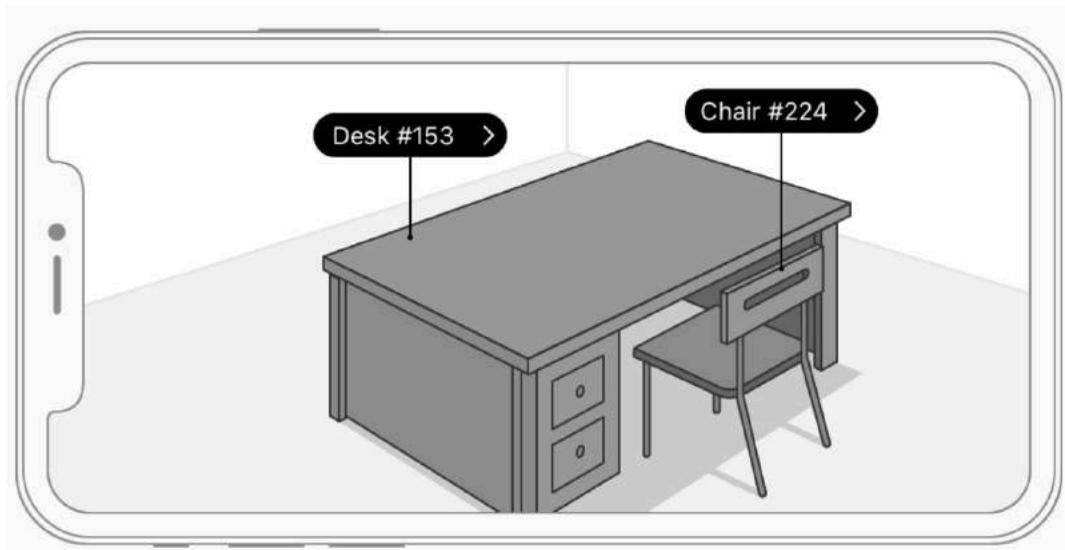
如果用户看起来无法找到屏外对象，请考虑提供视觉或听觉提示。例如，如果某个物体偏向屏幕外的左侧，则可以在屏幕左侧显示一个指示器，以便用户将摄像头对准该方向。

如果你一定要显示引导文字，请使用简单易懂的术语：AR是一种先进的概念，可能会对某些用户造成困扰。为了使其便于理解，请避免提及像 ARKit、世界检测技术、和跟踪技术这样的开发者导向的技术性术语。而使用大多数人都会理解的常用术语。

应该这样做	不应该这样做
无法找到表面，尝试移动到侧面或重新定位手机。	无法找到平面，调整追踪。
点击一个位置以放置（要放置的对象的名称）。	点击平面来锚定一个物体。
尝试调亮灯光并四处走动。	特征不足。
缓慢移动你的手机。	检测到过度动作。

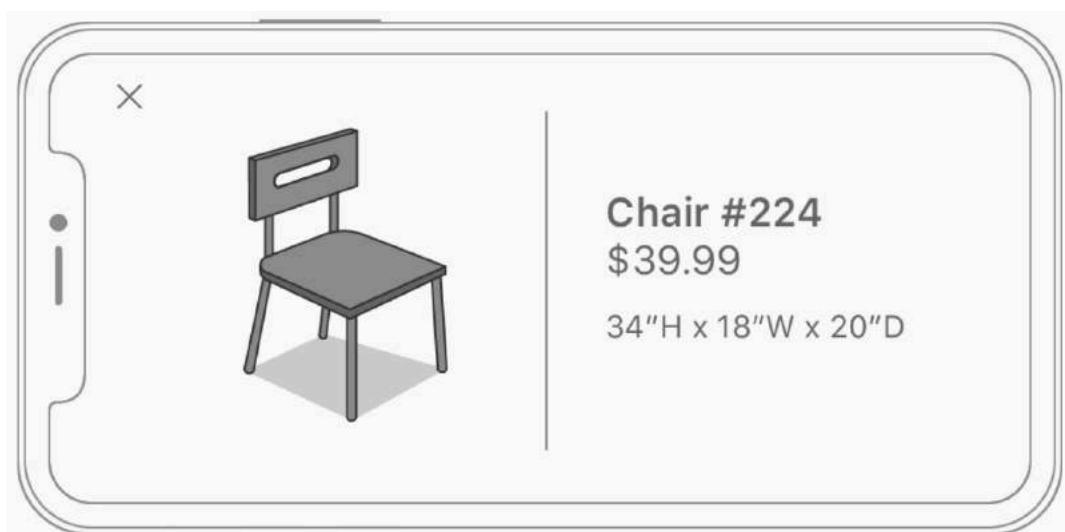
确保重要的文本可读：显示用于标签，注释和说明的文本，它就像是贴在到手机屏幕上的而不是虚拟空间内的。无论被标记的对象距离用户多远，文本

都应面向用户并以相同的大小显示。

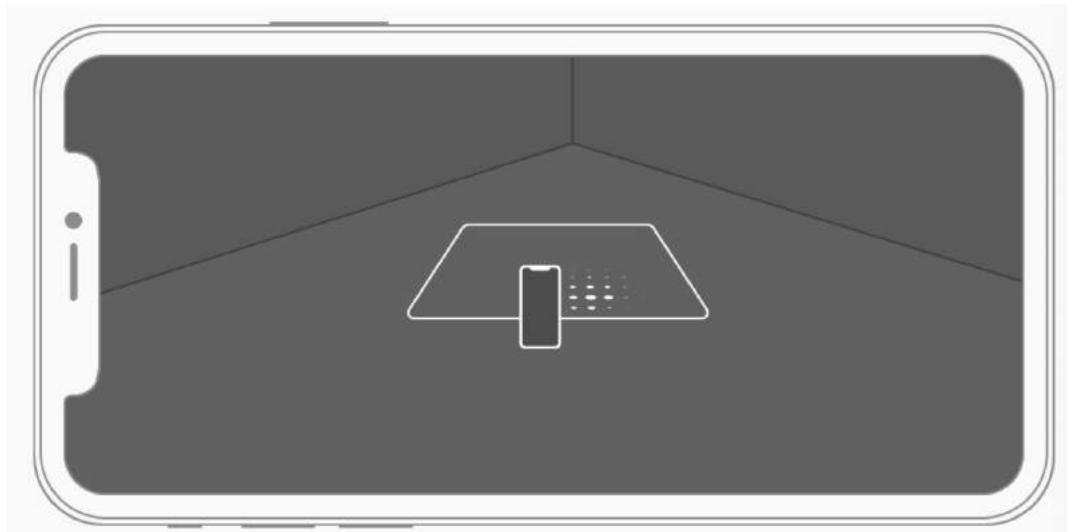


示出重要的文本是否有更多的未显示信息：使用符合您应用视觉风格的标识，告诉用户他们可以点按获取更多信息。比如说：Measure 应用中测量值旁边向右的小箭头。

将文本信息保持在最低限度：只显示用户需要的信息；再次比如 Measure 应用中，只会显示测量值和单元类型。



考虑在屏幕上显示其他信息：固定在屏幕上的信息能够随时随地方便用户阅读，用户随处移动也不会丢失可读性。此外，屏幕空间可用于展示其他内容。比如 Measure，屏幕上可以显示更加详细的测量信息，以及在弹出窗中提供复制测量信息的按钮。



使用视觉化的提示，让用户知道该如何初始化表面检测。

提示是否在进行初始化表面检测并让用户看到：每次您的应用进入 AR 模式，就会进行环境评估和表面检测的初始化进程。表面检测的时长受诸多因素影响而有所不同。如果初始化时间过长，请告诉用户此 APP 正在检测表面，并使用与应用风格一致的视觉元素来提醒用户：慢慢地扫视周围环境可以加速初始化。

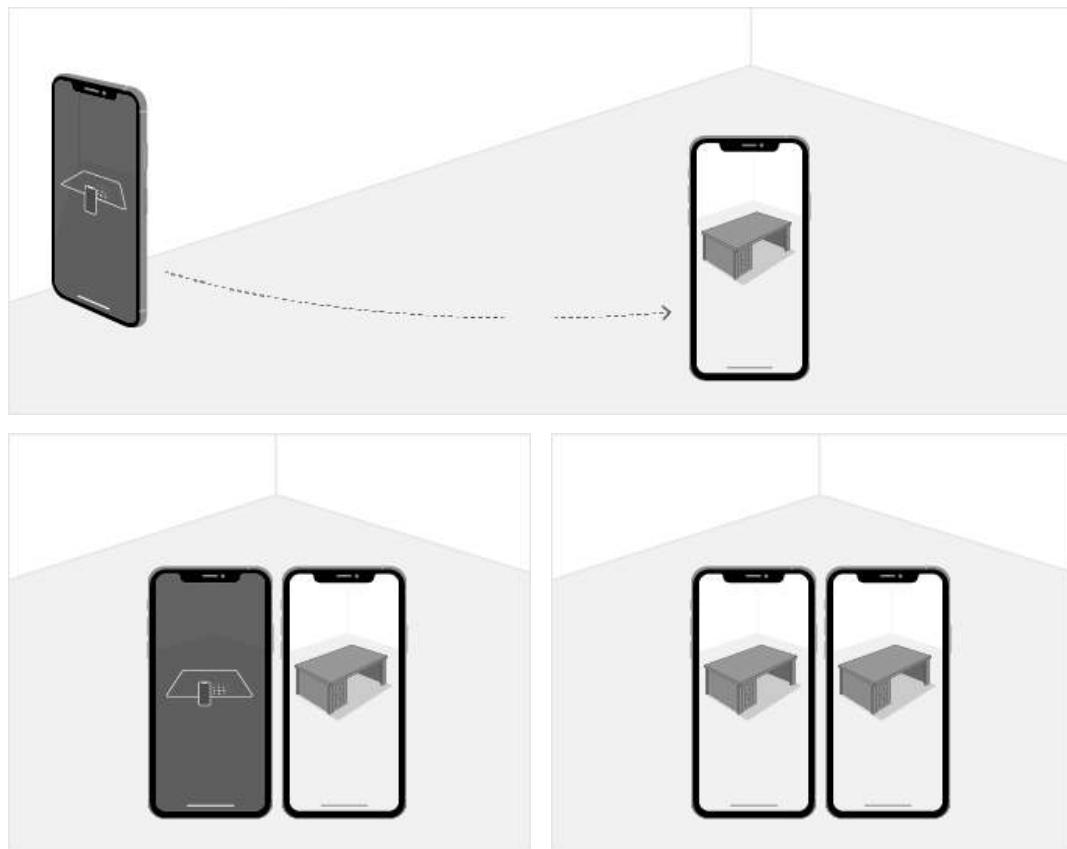
多用户 AR 应用的设计

无论在同一地点或是不同地点，您都通过共享表面，共享的物理对象（例如玩具）或共享图像来构建多用户 AR 体验。比如说，SwiftShot 允许位于同

一位置的两个玩家协作击倒同一表面上的虚拟块。

同时在主机与客机的同一屏幕上显示「主持」和「加入」共享 AR 的选项：

在虚拟地图构建和渲染完毕之前，不要像用户展示画面。



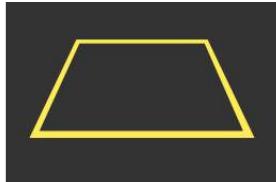
通过与主机共享同一表面同一角度的视图来加入多人 AR。

使用共享表面或物理对象加入多人 AR 时，鼓励受邀请的一方站在主机附近并将设备指向相似的方向：从同一位置扫描可以更快地检测共享表面或物理对象。当有多个设备指向同一个虚拟对象时，可以考虑使用一个视觉图像来代表这些设备。

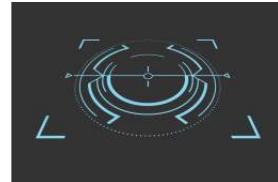
放置虚拟物体



表面检测指示器



物体位置指示器



App特定指示器

帮助人们了解何时找到表面并放置物体：视觉指示器能很好地展示表面定位模式已激活。例如，屏幕中心的梯形标线可以帮助用户推断出他们应该找到一个水平或垂直的平面。一旦定位了一个表面，指示器就应该在外观上发生变化，以表示现在可以放置物体。如果指示器的方向与被检测表面对齐，它可以帮助用户预测被放置的物体将如何对齐。要把视觉指示器设计成像是你的 App 体验的一部分。

当用户放置物体时适当地作出响应：在表面检测过程中精度逐渐提高（在很短的时间内）。如果用户点击屏幕放置物体，请立即使用当前可用的信息进行放置。然后等到表面检测完成，再对物体的位置进行细微调整。如果物体放置在被检测表面的边界之外，请将物体轻轻推回到表面上。

不要尝试将物体与检测到的表面边缘精确对齐：在 AR 中，表面边界是近似值，它可能随着用户周围环境被进一步分析而发生改变。

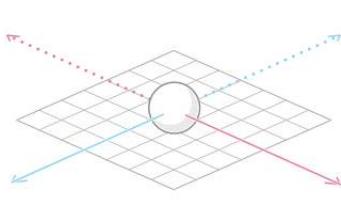
用户与虚拟对象的交互



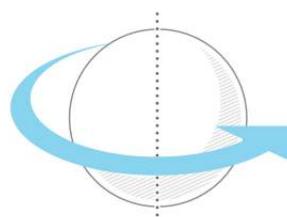
偏向直接对物体进行操作而非使用单独的控件：当用户可以触摸屏幕上的物体并直接与其进行交互时，这种方式代入感更强并且更加直观，所以不会让用户与屏幕不同部分的单独控件进行交互。但请记住，当用户在移动时，直接操作会比较困难。

允许用户使用标准、熟悉的手势直接与虚拟物体进行交互：例如，考虑支持用于移动对象的单指拖动手势，以及用于旋转对象的双指旋转手势。旋转通常应该发生在物体所在的表面。例如，放置在水平表面上的物体通常围绕物体的纵轴旋转。有关相关指导，请参阅 [Gestures](#)。

通常需要保持简洁的交互：触摸手势本质上是二维的，但是AR体验涉及真实世界的三个维度。考虑以下方法来简化与虚拟物体的用户交互。



放置对象的二维表面有限的移动



对象转动的有限的单轴

在合理范围内接近可交互虚拟物体时响应手势：用户难以精确地触摸小、薄或放置在一定距离的物体上的特定区域。当你的应用在可交互物体附近检测到手势时，最好假设用户想要影响该物体。

考虑被用户驱动的物体缩放是否必要：当物体（如玩具或游戏角色）不具有固有尺寸并且用户希望看到它更大或更小时，缩放通常是合适的。对于类似于一件家具这样相对于真实世界而言尺寸有限的物体，如果物品设置为精确的尺寸，则缩放无关紧要。缩放不是调整物体距离的补救措施。例如，放大物体使它看起来更加接近，但实际上物体仍然很远。

警惕潜在的手势冲突：例如，双指捏合手势与两指旋转手势非常相似。如果你实现了这样两个类似的手势，一定要测试你的应用，并确保它们被正确解读。

确保虚拟物体移动平稳：当用户调整物体大小，旋转它们或将它们移动到新位置时，不应该出现跳动。

探索更多有魅力的互动方式：手势不是人们与AR中的虚拟物体交互的唯一方式。你的应用可以使用其他因素（如动作和接近度）将内容带入生活。例如，一个游戏角色会在用户走向它时转头看着用户。

对用户环境中的图像做出反应

你可以使用用户环境中的已知图像来触发虚拟内容，从而增强AR体验。你的应用提供一组2D参考图像，ARKit指示何时何地在用户环境中可以检测到这些图像。例如，应用可能会识别一张科幻电影海报，然后从海报中出现虚拟飞船并在四周飞行；一家零售店的应用可以通过识别放置在门两侧的海

报，让商店的前门出现虚拟角色。

设计并显示参考图像以优化检测：当你提供参考图像时，可以指定你希望在用户环境中这些图像的物理尺寸。提供更精确的尺寸测量有助于ARKit更快地检测图像，并提高对其真实世界位置估算精度。具有高对比度和显著细节的平面矩形图像的检测性能和精度是最好的。避免尝试检测出现在反射或曲面的真实世界表面上的图像。

仅将检测到的图像用作显示虚拟内容的参考框架：ARKit不会跟踪被检测图像位置或方向的更改。因此，如果您尝试精确地放置虚拟内容，就像在画中将胡须放在脸上一样，内容可能不会保留在原位。

限制一次使用的参考图像的数量：当ARKit在用户环境中查找25个或更少的不同图像时，图像检测性能效果最佳。如果你的应用需要超过25个参考图像，则可以根据上下文更改一组活动参考图像。例如，博物馆指南应用可以使用核心位置来确定用户当前所在地是博物馆的哪一部分，然后仅查看该区域中显示的图像。

有关更多开发人员指南，参见 [在AR体验中的图像识别](#)。

处理中断

避免不必要的中断AR体验：当AR未激活时，ARKit无法跟踪设备的位置和方向。避免中断的一种方法是让人们在体验中调整物体和设置。例如，如果用户将他们正在考虑购买的椅子放入起居室，并且该椅子可以采用不同的面料，则允许他们在不退出AR的情况下更换面料。

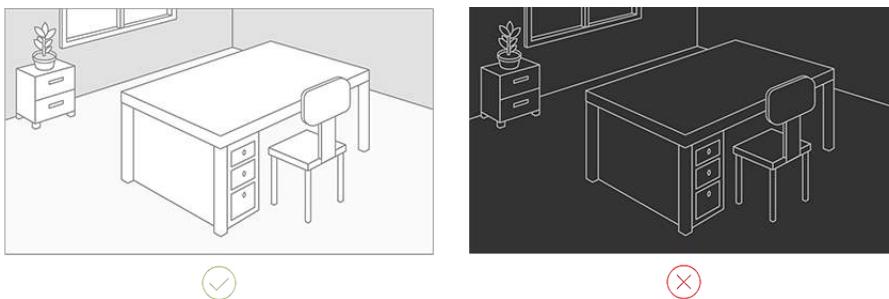
使用重定位从其他中断恢复：在中断期间，ARKit无法跟踪设备的位置和方向，例如用户暂时切换到其他应用或接听电话。中断之后，先前放置的虚拟物体可能出现在错误的真实世界位置。当你启用重定位时，ARKit会尝试恢复将这些虚拟物体回到其原始真实世界位置所需的信息。这个过程要求用户将他们的设备放置在与中断前一致的位置和方向。有关开发人员指南，请参阅 [ARSessionObserver](#)。

考虑隐藏先前放置的虚拟物体，直到重新定位完成：在重新定位期间，ARKit试图将其以前的状态与用户环境的新观察结果进行协调。在此过程完成之前，虚拟物体的位置可能有误差。

允许用户取消重定位：如果用户无法确定设备的位置和方向并定位在中断前的位置附近，则重新定位会无限期地继续下去，但不会成功。你需要引导用户成功恢复进程，或提供重置按钮和其它方式，以便用户在重定位不成功时重新启动AR体验。

处理问题

如果没有达到用户预期，允许用户重新设置体验：不要强迫用户等待条件的改善或者陷入物体如何放置的困境。给他们一种重新开始的方式，看看是否有更好的结果。



如果发生问题，建议进行修复：对用户环境和表面检测的分析可能由于各种原因而失败或花费太长时间，例如光线不足，表面过度反射，表面没有足够的细节，或者摄像头过度运动。如果你的应用收到这些问题的通知，请提供解决这些问题的建议。

问题	可行的建议
检测到的特征不足	尝试增加光线和四处走动
检测到过度运动	尝试缓慢移动你的手机
表面检测花费时间太长	尝试四处走动，增加光线，确保你的手机指向一个有足够纹理的表面

仅在有能力的设备上提供AR功能：如果你的应用的主要用途是增强现实，请仅将你的应用提供给支持ARKit的设备。如果你的应用提供AR作为辅助功能（如包含产品照片并允许在AR中查看某些产品的家具目录）；如果用户尝试在不支持的设备上进入AR，避免显示错误。如果设备不支持ARKit，首先不要提供可选的AR功能。有关开发人员指南，请参阅 [信息属性列表项关键参考](#) 的 [UI必需的设备功能](#) 中的 `arkit key` 以及 [ARConfiguration](#) 的 `isSupported`。

AR字形

应用可以在启动基于ARKit的体验的控件中显示AR字形。你可以在参考资料中下载该字形。



按预期使用AR字形：该字形应严格用于启动基于ARKit的体验。切勿改变字形（除调整其大小和颜色外），或将其用于其他目的，或将其与不使用ARKit创建的AR体验结合使用。

保持最小的净空间：AR字形周围需要的最小净空间数值为字形高度的10%。不要让其他元素以任何方式占用此空间或遮挡字形。



AR 徽章

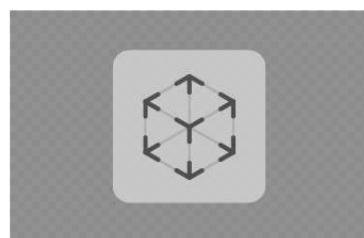
包含产品集合或其他对象的应用可以使用徽章来标识特定项目，并使用ARKit在AR中查看。例如，百货商店的应用可能会使用徽章来标记家具，以便用户在购买前在家中预览家具。



按预期使用AR徽章，不要改变它们：你可以在 [Resources](#) 中下载AR徽章，可用折叠和展开形式。这些图像专门用于识别可以在AR中查看的产品或其他对象。切勿改变徽章及其颜色，或将它们用于其他目的，或将其与不使用 ARKit 创建的AR体验结合使用。



AR标志



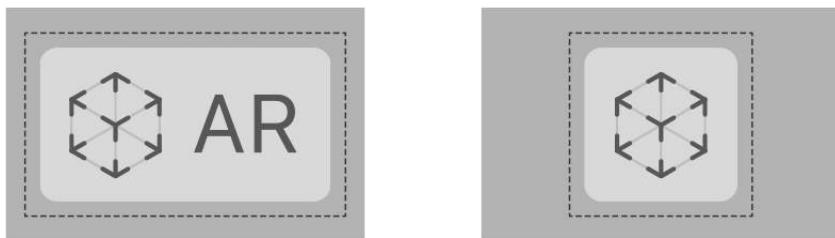
只有象形的AR标志

AR徽章比字形徽章更可取。一般情况下，如果空间有限且不能容纳AR徽章，请使用带字形徽章。两个徽章都能以默认尺寸正常工作。

只有当你的应用同时包含可以在AR中查看的对象和不能查看的对象时，才使用徽章：如果你的应用中的所有对象都可以在AR中查看，则徽章是多余且不必要的。

保持徽章位置一致和清晰：显示在对象照片的一个角落时，徽章看起来最好。始终将其放置在同一个角落，并确保它足够大以便清晰可见（但不会太大以至于遮挡照片中的重要细节）。

保持最小的清晰空间：AR徽章周围所需的最小空间为徽章高度的10%。其他元素不应侵占此空间或以任何方式遮挡徽章。



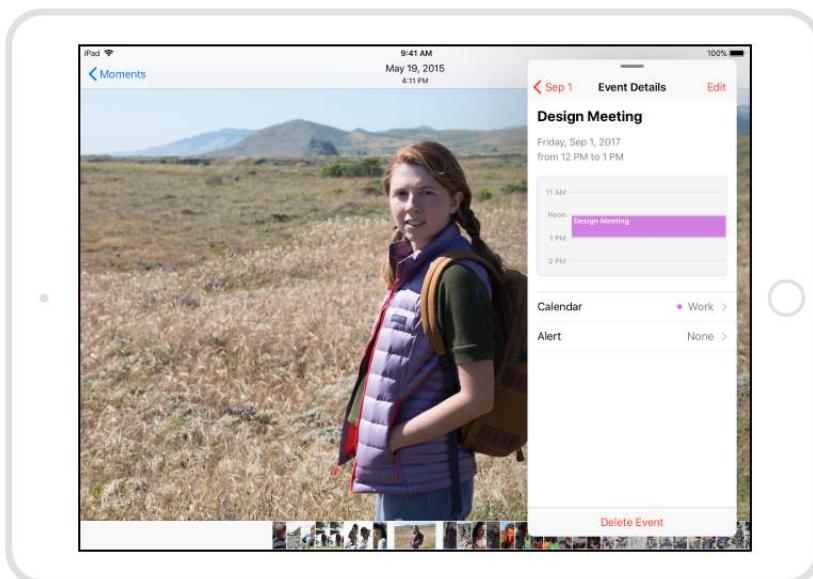
了解更多

有关开发人员指南，参见 [ARKit](#)。

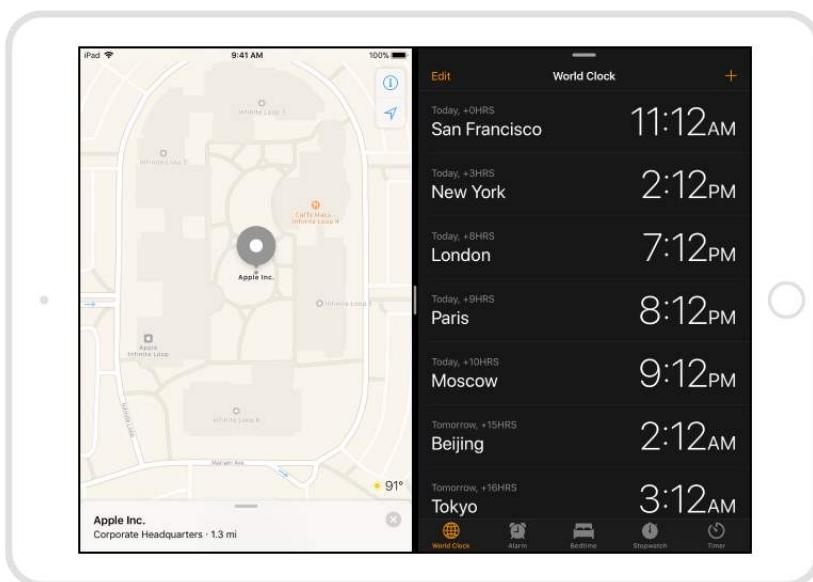
4.2 多任务处理

多任务处理功能可以让你通过iOS设备上的多任务界面，或者在 iPad 上使用多点触控手势，随时从一个 App 快速切换到另一个 App。在iPad上，多任务处理功能还可以让你在“幻灯片”、“分割视图”或“画中画”模式下同时使用两个 App。在屏幕右侧向左滑动可以进入“幻灯片”模式，它能让你在不离开当前 App 的情况下暂时性地使用第二个 App，例如在使用 Safari 时快速查看你的邮件收件箱。“分割视图”模式可以让你同时使用两个并排的 App，而“画中画”模式可让你在使用另一个 App 时观看视频。

设计出一个能够在多任务环境中良好运行的 App 取决于你的 App 可以与设备上的其他 App 和谐共存。这意味着你的 App 不应该占用太多 CPU、内存、显示屏幕或者其他系统资源。它应该很好地应对来自其他 App 的突发干扰和声音，能快速流畅地在显示界面和后台来回切换，并且在后台运行时表现得可靠稳定。



幻灯片



分割视图



画中画

做好随时中断的准备，并时刻准备着恢复：你的 App 能在任何时候被中断。当中断发生时，你的 App 应该快速精准地保存当前状态，这样当用户返回时，就可以无缝地从他们离开的地方继续使用。

更多开发细则请参阅 [App Programming Guide for iOS](#) 中的 [Preserving Your App's Visual Appearance Across Launches](#) 部分。

确保你的界面能够适应双倍行高的状态栏：一些进程中的电话、录音、共享功能在屏幕顶部会显示额外的一个状态栏。在一些没有考虑该情况的App中，这个增加的高度会遮挡其他界面元素或者把它们挤下去。你需要在这些情况下测试你的 App，确保你的界面能够应对自如，并且仍然保持美观。

需要格外注意和应对暂停操作：如果你的 App 是一个游戏或者媒体播放应用，请确保你的用户在切换到其他App时也不会错过任何内容。当他们切换回来时，让他们从上次离开的地方继续使用。

合理处理来自应用外的声音：有时你的 App 的音频可能会被来自其他App 或是系统的声音打断。比如，来电铃声或是被 Siri 打开的音乐播放列表。当这些情况发生时，你的 App 应该以用户预期的方式处理。对于重要的音频干扰，比如音乐播放、广播或有声读物，你的 App 应该立刻暂停其音频。对于短暂的干扰，比如 GPS 导航通知，你的 App 应该暂时降低音量或是先暂停音频等待干扰结束，再继续播放。

了解更多指导，请参阅 [Audio](#)。

在后台完成用户发起的任务：当用户开启了一个任务，即使离开了 App 他们也希望任务能够继续完成。如果你的 App 正在执行一个不需要用户额外输入的任务，请在 App 回到前台前，在后台完成它。

有效控制通知次数：无论你的 App 在前台、后台或是完全没有在运行，它都能在特定的时间给用户推送通知。你可以使用通知来传达重要讯息，但是要避免用户被过多通知烦扰。比如，当你的 App 在后台时，不要每完成一个任务就给用户发送一个通知。相反的，可以让用户通过返回你的 App 来查看任务的完成情况。了解更多指导，请参阅 [Notifications](#)。

了解关于iPad特定的开发细则，请参阅 [Adopting Multitasking Enhancements on iPad](#)。

4.3 iPad 上的多窗口

在 iOS13 和更高的版本中，iPad 应用能够支持多窗口。举个例子，在一个可以创建文档的 iPad 应用里，用户可以同时打开多个文档的窗口。



NOTE

如果您想让自己 iPad 应用的 Mac 版本也支持多窗口，您必须让应用支持 iPad 上的多窗口。详情请见 [iPad Apps for Mac](#)。

用户可以通过多种方式打开一个新窗口，比如说：

- 将应用在 Dock 栏中的图标拖拽到屏幕的一侧，从现有窗口中选择一个打开，或新建一个窗口。
- 将对象拖拽到屏幕的一侧，并放置在支持该操作的目标之上。
- 长按主屏或 Dock 中应用的图标呼出情景菜单，点击「显示所有窗口」，再点击添加按钮。
- 长按一个对象呼出情景菜单，选择在新窗口中预览对象。

一般来说，iPad 应用使用两种类型的窗口。主窗口包含多个可操作对象；通常，用户倾向于持续与主窗口交互。辅助窗口包含单个对象以及与之关联的操作；在关闭窗口之前，人们往往只与辅助窗口交互一次。比如说，在 Mail 中，主窗口显示邮箱，而单个邮件则显示在辅助窗口中。

让主窗口和辅助窗口都支持多窗口功能：由于主窗口通常包含层级更高的对象，多主窗口能够显示列表内的不同区域，用户因此能够看到更多内容。比如人们可能希望一个主要的窗口显示收件箱，另一个显示草稿。多个辅助窗口允许人们轻松查看或处理多个项目，例如多个邮件。

确保辅助窗口本身有其用处：辅助窗口应该提供给用户一种额外的视角来使用应用的内容和功能，而不要仅仅在辅助窗口中显示主窗口里已有内容相关的选项或工具。

在辅助窗口中使用「完成」或「关闭」按钮：由于辅助窗口显示的是一次性的任务或对象，用户希望任务完成时这个窗口可以关闭。不要使用「返回」按钮退出辅助窗口，「返回」应当帮助用户在同一个窗口内退回上一视图。

4.4 通知

无论设备是锁屏状态还是正在使用中，App 都能随时利用通知功能来提供及时和重要的信息。比如，通知可能会在以下几种情况发生时出现：新消息到来时、一个事件将要发生时、有新数据可获取时或是某些状态发生改变时。用户可以在锁屏上、屏幕顶部（使用设备时），以及通知中心（通过从屏幕顶部边缘下滑呼出）看到通知栏。每个通知都包含 App 的名称、一个 App 图标以及一条消息。通知也可能伴随着声音提示，以及 App 图标上小红点角标的出现和更新。



提示：通知可以是本地或是远程的。本地通知由同一个设备发出和接收。一个待办事项 App 可以使用本地通知来提醒用户一个将要到来的会议或是到期日。远程通知，也叫做推送通知，来自一个服务器。一个多玩家游戏就可以使用远程通知让每个玩家知道什么时候轮到他们。

通知表现

每个 App 的通知行为都可以在设置里单独管理。只要是支持通知功能的 App，你可以完全地启用或禁用这个功能。你也可以设置通知是否在通知中心和锁屏上可见，是否在 App 图标上出现角标，以及选择以下一种通知样式：

横幅：当设备在使用时在屏幕顶部出现几秒，然后消失。

提醒框：当设备在使用时在屏幕顶部出现，直到被手动关闭。

在未锁屏时点击通知，或是在锁屏时滑动这项通知，都可以实现以下几种行为：结束通知、把它从通知中心移除，打开发送通知的应用并展示相关的内容。比如，在未锁屏的设备上点击一条新的邮件通知，就会打开邮箱并且显示新的信息。

在未锁屏时，上滑通知或让它消失，或让它关闭通知，也可将它从通知中心移除。

使用 3D Touch 在一个通知上按压，或是在未锁屏时下滑通知面板，就能打开扩展的详情视图。这个视图支持自定义并且包含最多四个操作按钮。比如，一个待办事项 App 可以推送一个含有详情视图的任务通知，上面有可以推迟任务和标记为已完成的按钮；一个日历事件的通知提供了“小睡”功能来短暂推迟事件的闹铃。



NOTE: 用户在第一次使用 App 的时候，会被要求明确地选择是否接收来自该 App 的通知。如果有人选择不接收通知，他们同样也能通过访问“设置”来选择接收。

设计一个很棒的通知体验

提供有用的通知：用户打开通知是为了快速获得最新消息，所以你的重点是提供有价值的信息。通知显示需要使用完整的句子，句首英文字母大写，合适的标点符号，并且不要删减你的信息——必要时系统会自动处理。当通知被关闭时用户很难再记住它们，所以要避免在通知中引导用户打开你的

App，进入指定页面然后点击指定按钮来完成一些任务。

即使用户没有作出回应，也不要为同一件事情发送多个通知：用户只有在方便时才会处理通知。如果你为同一件事发送了多个通知，并且填满了通知中心，那么用户就很可能关闭所有来自你的 App 的通知。

不要在通知里包含你的 App 的名字和图标：系统会自动在每条通知的顶部显示这些信息。

提供描述性文字，以便在通知预览被隐藏时显示：根据用户的设置，出于隐私考虑通知预览可能会被隐藏。在这种情况下，仅显示你的 App 图标和通用描述（通知是默认描述）。为了向用户提供足够的内容，你的 App 应该提供能简洁描述通知内容的自定义文本，如朋友请求，新评论，提醒或发货开发者指南，请参阅 [hiddenPreviewsBodyPlaceholder](#)。

提供声音以辅助你的通知提醒：当用户没有盯着屏幕时，声音是一个引起他们注意的绝佳方式。当一个待办事项 App 开始执行重要任务时可能就会发出一个提示音。你的 App 可以使用自定义声音或是系统的提示音来达到效果。如果你使用自定义的声音，请确保它是简短、与众不同并且制作精良的。详情请参阅 [Local and Remote Notification Programming Guide 中的 Preparing Custom Alert Sounds](#) 部分。需要注意的是用户可以随时地关闭通知提示音。他们也可以开启伴随着声音的振动——这只能被手动开启，而不是通过你的 App 程序自动启用。

考虑提供一个详情视图：一个通知的详情视图提供了关于该通知的更多信息，并且允许他们在不离开当前环境的情况下执行快捷的操作。这个视图应该包含有用、易识别的信息，让人感觉就是一个你的 App 的自然延伸。

它可以包含图片、视频以及其它内容，它还能在显示时动态更新。比如，一个拼车 App 就能够在该窗口显示一张地图，并标出一辆车正在朝着你当前的位置驶来。

提供直观、有用的操作：一个通知的详情视图能最多包含四个操作按钮。这些按钮用来执行常用、省时的任务，而不用打开你的 App。使用简短、英文首字母大写的名称明确地描述该操作的含义。一个通知的详情视图还能在屏幕上呼出一个键盘用来收集执行操作需要的信息。比如，一个通讯 App 可以允许用户直接在新消息通知的详情视图上回复。

避免展示破坏性的操作：要在通知详情视图里展示破坏性操作之前请仔细考量。如果你必须展示它们，确保用户拥有足够的上下文信息，以避免出现意外后果。破坏性的操作应该以红字呈现。

角标

角标是用来补充说明通知，而不能用来表示重要的信息：需要注意 App 的角标可以被关闭。如果你的 App 依赖于通过角标来传达重要信息，就等于你在冒着用户会错过这些信息的风险。

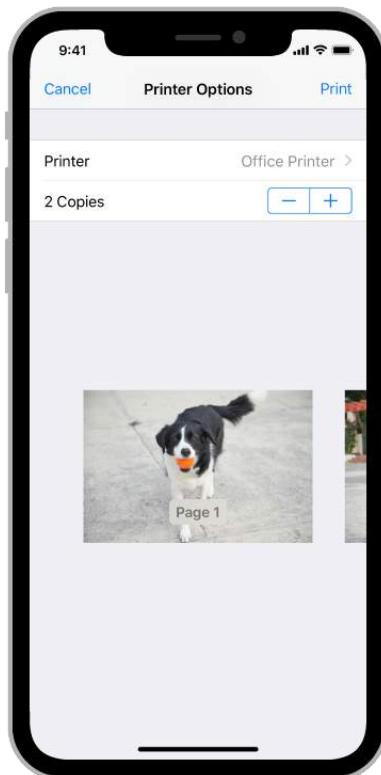
角标仅用于通知：角标不应该用于显示其他类型的数字信息，例如空气质量，日期，股票价格或天气。

保持角标实时更新：当收到对应的消息时要立即更新你的 App 的角标数字。用户只有在看到确切提示之后才会进入你的App查看。请注意，将角标上的数字清零意味着同时在通知中心移除所有相关的通知栏。

开发者指南, 请参阅 [Local and Remote Notification Programming Guide](#)。

4.5 打印

你的 App 可以利用系统自带的 AirPrint 技术来使用兼容的打印机实现图片、PDF 以及其它内容的无线打印。当用户在有 AirPrint 功能的应用内浏览可打印的内容时, 他们可以通过点击导航栏或是工具栏上的操作按钮打开选项面板, 然后再点击打印按钮来打开打印视图。这个视图提供了一个可用打印机的列表以及一些自定义选项, 比如打印的份数、页面范围, 并且提供了一个开始打印的按钮。



让打印选项易于发现: 如果你的 App 有一个工具栏或是导航栏, 请使用系

统提供的操作按钮来打印。用户对这个按钮更加熟悉，并且在其它应用中也是用它来打印。如果你的 App 没有工具栏或是导航栏，那么设计一个自定义的打印按钮来代替。

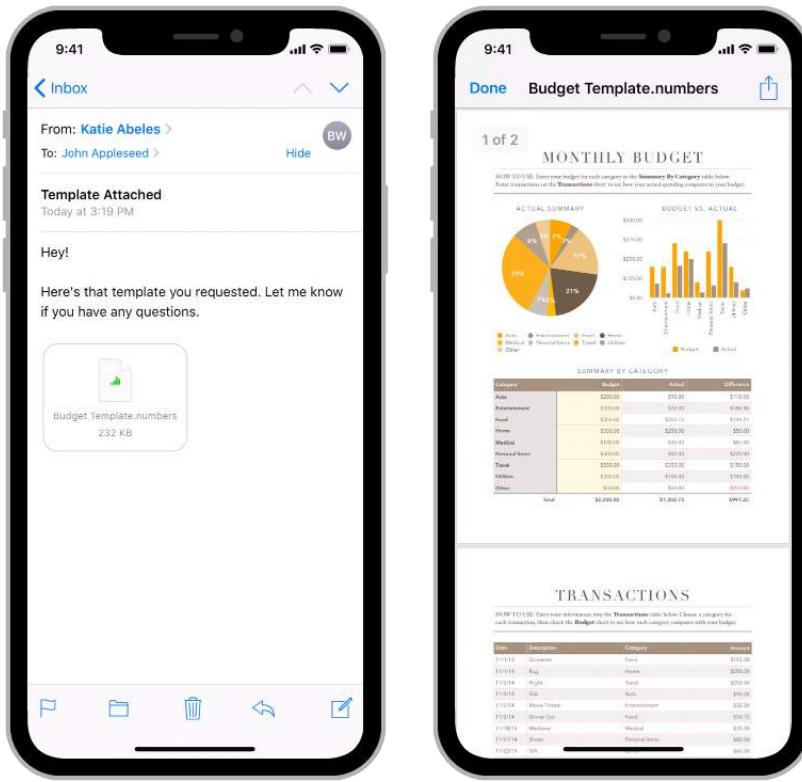
只在可以打印的情况下才允许打印：如果在你的屏幕上没有任何内容或是没有可用的打印机，那么在用户点击操作按钮后禁用打印按钮。如果你的 App 使用自定义的打印按钮，在无法打印时让其不可点击或是隐藏它。

提供有用的打印选项：思考用户在打印 你的内容时会想要指定哪些选项。可以考虑选择页面范围和打印份数的选项。也可以启用附加的选项，比如双面打印，如果这样有意义并且打印机也支持的话。

开发者指南，请参阅 [Drawing and Printing Guide for iOS](#) 和 [UIPrintInteractionController](#)。

4.6 快速查看

在你的app中，快速查看功能可以让用户预览 Keynote、Numbers、Pages、PDF 文档、图片以及其它类型的文件，即使你的应用并不支持这些文件格式。也可以使用该功能来查看邮件的附件。在下载附件之后，邮件信息中会显示附件的图标和文件名。点击图标就能预览附件。



在当前环境下合理地展现预览窗口：在 iPhone 上，如果你的 App 有导航栏，让预览视图下移留出位置给导航栏，就和你的 App 其它层级的视图一样。在 iPad 或是没有导航栏的 App 内，用全屏有导航栏的模态视图中打开预览视图。通过以上两种方法，导航栏就能提供退出快速查看状态的按钮，以及预览特定的一些按钮，比如分享和标记这样的操作。如果你的 App 包含一个工具栏，那么预览特定的按钮就会在工具栏出现而不是导航栏。

开发者指南，请参阅 [Document Interaction Programming Topics for iOS](#) 和 [Quick Look](#)。

4.7 评级和评价

评级和评价可以帮助用户在考虑是否尝试 App 时做出合理的决定。良好的

评级和积极的评论意味着你的 App 能获得更多下载，用户反馈可以让你深入了解你的 App 在现实世界的使用情况，从而更好地帮助你实现未来的开发工作。

提供良好的整体体验是提升评级和鼓励积极评价的最佳方法，但在适当的时候要求提供反馈也很重要。在请求用户对你的 App 进行评级时，请牢记这些注意事项：

在用户与你的 App 有一定程度互动后再请求用户对其进行评级：例如，在达到某个游戏等级或完成某个生产任务时提示用户。绝对不要在用户第一次使用或者进行新手引导时请求评级。给用户充足的时间，让他们对你的 App 有自己的意见。

不要中断用户：特别是当他们在执行紧急或压力很大的任务时。寻找合适的时机发送评级请求。

不要变得让用户讨厌：重复的评级提示可能会刺激，甚至可能会让用户对你的 App 产生负面情绪。评级请求至少间隔一周或两周，并在用户与你的 App 有进一步互动后可以再次提示。

系统评级和审查提示

系统为 App 提供了一致且不干扰的方式来请求用户进行评级和评论。要使用此功能，你只要识别你的用户在体验中反馈意见的位置。如果用户尚未提供反馈，并且你的 App 最近也没有提出请求，那么系统将显示一个应用内提示，请求用户进行评级和评论。用户可以点击提供反馈或关闭提示。（在“设置”中，用户还可以选择不接收所有已安装应用的评级请求提示。）在

365天的周期内，系统自动限制每个App的提示显示次数为三次。



最好使用系统提供的提示：系统的评级提示提供了一个熟悉，有效的体验，旨在使用用户的影响最小。

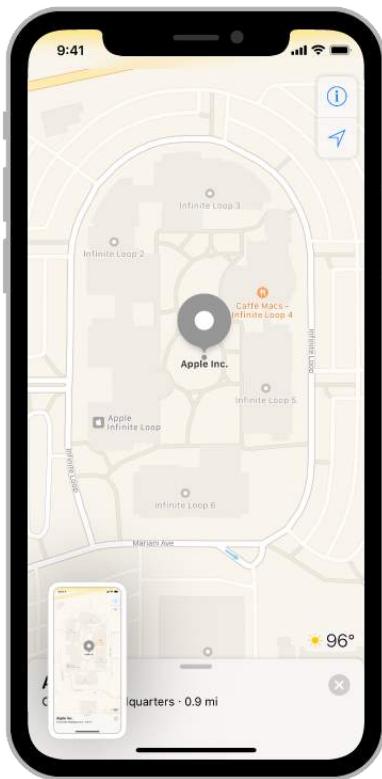
不要使用其他控件来请求反馈：由于系统限制了评级提示出现的频率，所以尝试用控件请求反馈可能会导致不显示评级提示。

开发者指南，请参阅 [StoreKit 中的 SKStoreReviewController](#)。

提示：回复用户评论是一个与用户沟通，解决问题，并可能提高App评级的好方法。更多操作指南，请参阅 [Responding to Reviews on the App Store](#)。

4.8 截图

用户可以通过截取屏幕截图来捕获屏幕上显示的内容。从 iOS 11 开始，屏幕截图在被截取后会在屏幕底部以预览形式简要显示。用户可以将预览图滑动到屏幕边缘来关闭它（如果用户没有任何操作，预览窗口会在几秒钟后自动关闭），点击预览窗口即可快速访问即时标记和共享工具。屏幕截图被保存到“照片”中的“截图”专辑中。



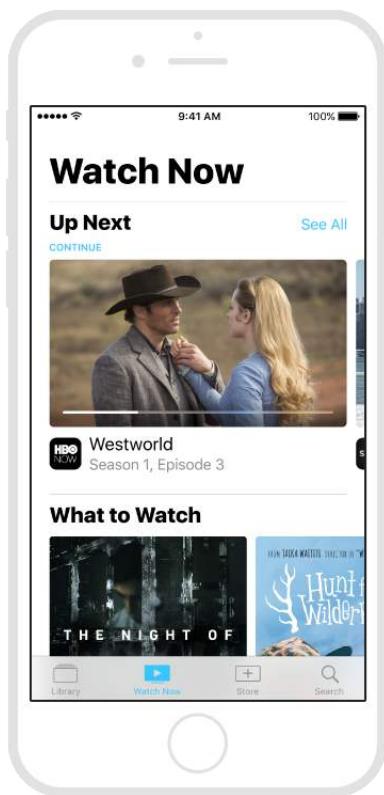
截取本地屏幕时，请勿更改 App 的界面：系统提供的屏幕截图功能在截取截图时提供了足够的界面和功能。自定义截图启动界面的更改和功能是重复的，不必要的，可能会令人困惑。如果你的 App 适用该功能，当其他用户使用该设备截取截图时，可以提醒用户。

4.9 电视提供商

你的 App 可以与 [TV App](#) 和 [single sign-on](#) 进行交互，为用户提供高度方便和一致的娱乐体验。

TV App 交互

TV App 可以全面访问系统的收藏，最近播放的，以及推荐的电影和电视节目。



观看 Westworld 在 HBO NOW 的订阅。HBONOW® 只能通过美国和某些美国地区的合作伙伴进行访问。某些限制申请, ©2016 Home Box Office , Inc. 保留所有权利。HBO®, HBONOW® 以及相关渠道和服务商标均为Home Box Office,Inc.的财产。通过订阅或认证可以观看 STARZ 上

的 Power。Power 为 ©2016 Starz Entertainment , LLC 版权所有。

开始并恢复播放

当用户请求在 App 中重放内容时，TV App 会自动打开你移动端的 App，并向你的移动设备发送通知。

确保平滑过渡到你的App：转换到移动设备上的 App 时，TV App 会变成黑色，且不会显示启动屏幕。移动端的App应该在开始播放或恢复内容之前立即呈现黑屏，保持视觉连续性。

立即显示预期内容：用户期望在你的 App 转换完成后，所选内容就开始播放（从 App 的黑屏直接跳到播放内容）。避免提供闪屏，详细屏幕，介绍动画或任何其他花费时间的障碍。这在恢复播放时尤为重要。

不要询问用户是否要恢复播放：如果可以恢复播放，则应自动执行此操作而无需提示确认。

确保为正确的用户播放内容：如果你的 App 支持多个用户配置文件，TV App 可以在发出播放请求时指定配置文件。开始播放之前，你的 App 应自动切换至这个设定。如果播放请求没有指定配置文件，则请求用户在播放开始之前选择一个，以便将来可以直接提取此信息。

载入内容

如果加载需要两秒以上，请考虑显示一个带有动态加载图标的黑色屏幕，且不没有其他内容。

尽可能避免使用加载屏幕：如果您的内容快速加载，则不需要加载屏幕。

尽快开始播放：如果需要加载屏幕，加载足够的内容之后再开始播放，并继续在后台加载剩余的内容。

设计黑色加载屏幕：在内容播放的过渡期间，可能会显示加载屏幕。为了与 TV App 的淡入淡出功能相融合，所以应该显示黑色屏幕。

加载屏幕上的内容最小化：如果你一定要让加载屏幕上包含品牌或图像，请尽量保持黑色背景并无缝衔接播放。

退出播放

退出播放后，用户仍然在你的 App 中，而不是返回到 TV App。不要让用户迷失方向。

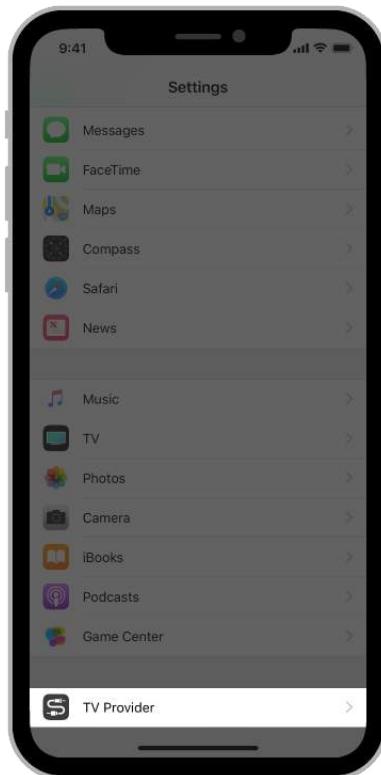
显示一个有相关内容的屏幕：退出播放时，保留用户正在观看内容的屏幕视图，并包含一个恢复播放的选项。如果详细信息屏幕不可用，请显示一个包含用户正在观看的内容或你的 App 主菜单的菜单。

准备即时退出：在收到回放通知后尽快准备退出屏幕，以便用户即使在播放开始后立即退出，也可以看到退出画面。

单点登录

许多流行的电视提供商允许用户在系统级别登录他们的帐户，消除了逐个应

用验证身份的需要。如果你的 App 需要电视提供商进行身份验证，请使用此功能提供最有效的入门体验。



当用户在系统级别登录时，避免显示退出选项：如果你的 App 必须包含退出选项，则调用该选项，将用户指向“设置”>“电视提供商”以退出其帐户。

不要通过调整隐私控制来指示用户退出：“设置”>“隐私”中的电视提供商控制不是退出机制。这些设置可以让用户管理可访问其电视提供商帐户的 App。

第五大章

视觉设计

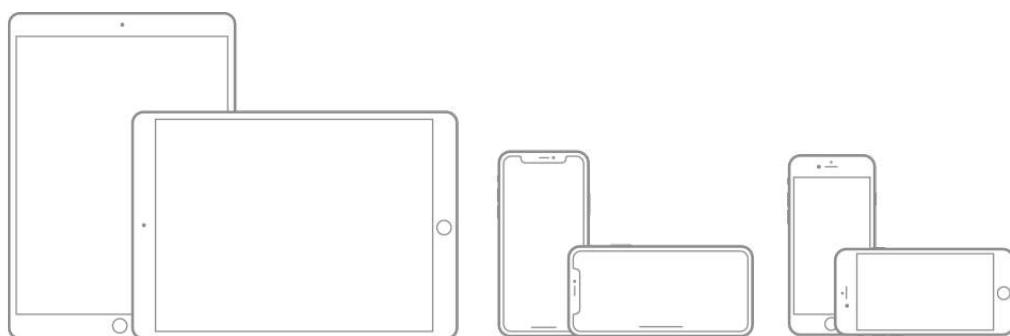
Visual Design

5.1 适应性和布局

人们总是希望他们的所有设备，随时随地都能使用他们喜欢的 App。在 iOS 中，界面元素和布局能够自动改变形状和大小以匹配不同的设备，比如在 iPad 中多任务操作时、分屏模式时以及屏幕旋转时的显示。因此，你必须设计一个适应性强的界面，在任何环境中都能提供良好的（用户）体验。

设备屏幕尺寸和方向

iOS 设备有各种不同的屏幕尺寸，可以横屏或者竖屏使用。



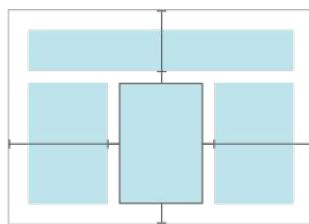
设备	竖屏尺寸	横屏尺寸
12.9" iPad Pro	2048px × 2732px	2732px × 2048px
10.5" iPad Pro	1668px × 2224px	2224px × 1668px
9.7" iPad	1536px × 2048px	2048px × 1536px
7.9" iPad mini 4	1536px × 2048px	2048px × 1536px
iPhone X	1125px × 2436px	2436px × 1125px
iPhone 8 Plus	1242px × 2208px	2208px × 1242px
iPhone 8	750px × 1334px	1334px × 750px
iPhone 7 Plus	1242px × 2208px	2208px × 1242px
iPhone 7	750px × 1334px	1334px × 750px
iPhone 6s Plus	1242px × 2208px	2208px × 1242px
iPhone 6s	750px × 1334px	1334px × 750px
iPhone SE	640px × 1136px	1136px × 640px

想要更多了解屏幕分辨率如何影响App中的显示效果，请参阅 [Image Size and Resolution](#).

自动布局

自动布局是构建自适应界面的开发工具。你可以定义 App 的内容规则。例如，您可以约束一个按钮，使其总是水平居中，并且位于图像下方8点，而不管可用的屏幕空间如何。

当检测到某些场景变化（称为特征）时，自动布局将根据特定约束自动调整布局。你可以将 APP 设置为动态适应特征，包括：

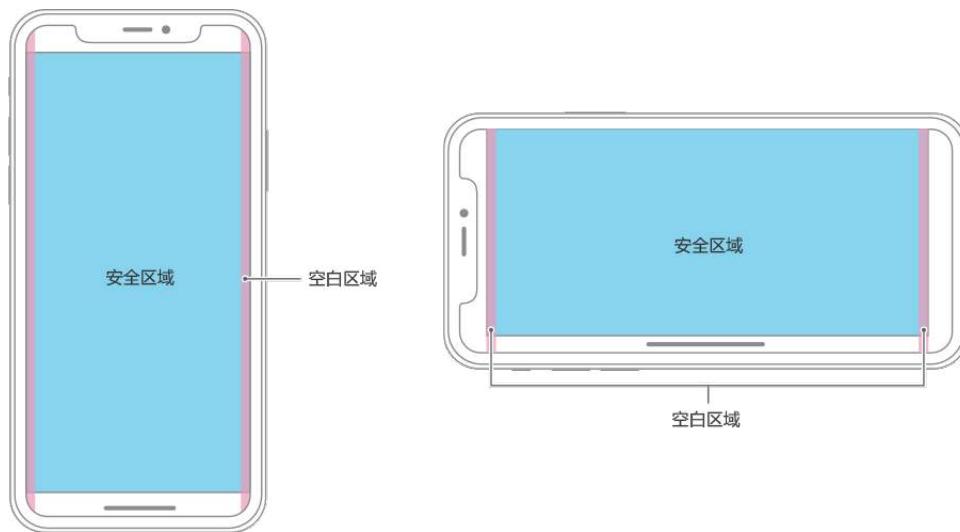


- 不同的设备屏幕尺寸、分辨率和颜色（显示）模式(sRGB/P3) ([screen sizes, resolutions, and color gamuts \(sRGB/P3\)](#))
- 不同的设备方向（横屏 / 竖屏）
- 拆分视图（[Split view](#)）
- 在iPad上的多任务处理模式（[Multitasking](#)）
- 动态文本的大小变化（[Dynamic Type](#)）
- 场景定制的国际化特征（布局方向的从左到右或者从右到左，日期/时间/数字格式，字体变化，文本长度）
- 系统特性可用性（[3D Touch](#)）

开发者指南，请参阅，[Auto Layout Guide](#) 和 [UITraitCollection](#)。

布局引导和（显示）安全区

布局引导定义了在屏幕上实际上并没有显示的矩形区域，但对内容的定位、对齐和间距提供了帮助。该系统包含预定义的布局引导，可以方便地在内容周围应用标准的边缘，并限制文本的宽度以达到最佳的可读性。你还可以自定义布局引导。



遵守 UI Kit 定义的安全区域和布局边界：这些布局引导可以根据设备和（显示）内容进行适当的嵌入，安全区域还可以防止内容对状态栏、导航条、工具栏和标签栏的影响。标准系统提供的视图自动采用安全区域布局引导。

了解更多开发者的指南，请参

阅 [UILayoutGuide](#), [layoutMarginsGuide](#), [readableContentGuide](#), 和 [safeAreaLayoutGuide](#)。

尺寸类型

尺寸类型是根据它们的大小自动的分配内容区域。该系统定义了两个尺寸种类，常规的(表示可拉伸的空间)和固定的(表示固定长度的空间)，描述视图的高度和宽度。视图可以拥有任何尺寸类型的组合：

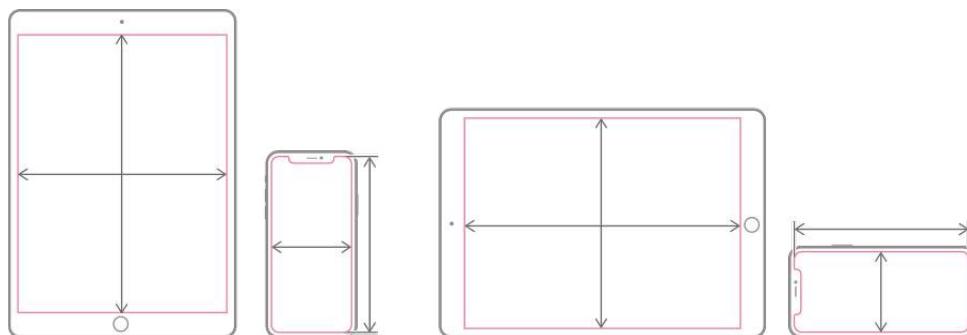
- 常规的宽度，常规的高度
- 紧凑的宽度，紧凑的高度
- 常规的宽度，紧凑的高度
- 紧凑的宽度，常规的高度

与其他环境变化一样，iOS 基于内容区域的尺寸是动态地进行布局调整的。

例如，当垂直尺寸种类从固定高度变为常规高度时，可能是因为用户将设备从横屏旋转到竖屏，标签栏可能会更高。

设备尺寸类型

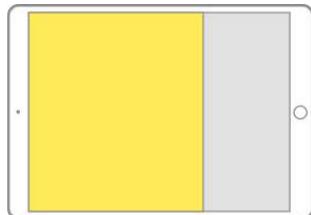
基于屏幕尺寸，不同尺寸类型的组合应用于不同设备的全屏体验。



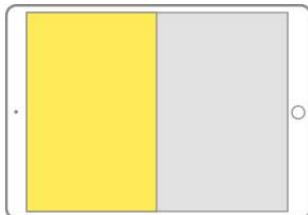
设备	竖屏方向	横屏方向
12.9" iPad Pro	常规的宽度, 常规的高度	常规的宽度, 常规的高度
10.5" iPad Pro	常规的宽度, 常规的高度	常规的宽度, 常规的高度
9.7" iPad	常规的宽度, 常规的高度	常规的宽度, 常规的高度
7.9" iPad mini 4	常规的宽度, 常规的高度	常规的宽度, 常规的高度
iPhone X	紧凑的宽度, 常规的高度	紧凑的宽度, 紧凑的高度
iPhone 8 Plus	紧凑的宽度, 常规的高度	常规的宽度, 紧凑的高度
iPhone 8	紧凑的宽度, 常规的高度	紧凑的宽度, 紧凑的高度
iPhone 7 Plus	紧凑的宽度, 常规的高度	常规的宽度, 紧凑的高度
iPhone 7	紧凑的宽度, 常规的高度	紧凑的宽度, 紧凑的高度
iPhone 6s Plus	紧凑的宽度, 常规的高度	常规的宽度, 紧凑的高度
iPhone 6s	紧凑的宽度, 常规的高度	紧凑的宽度, 紧凑的高度
iPhone SE	紧凑的宽度, 常规的高度	紧凑的宽度, 紧凑的高度

多任务的尺寸类型

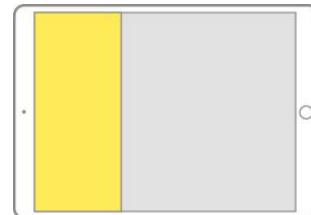
在 iPad 上，当你的 App 在进行 [多任务](#) 处理时，尺寸类型也同样适用。



2/3拆分视图



1/2拆分视图



1/3拆分视图

设备	模式	竖屏方向	横屏方向
12.9" iPad Pro	2/3 拆分视图	紧凑的宽度, 常规的高度	常规的宽度, 常规的高度
	1/2 拆分视图	紧凑的宽度, 常规的高度	常规的宽度, 常规的高度
	1/3 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
10.5" iPad Pro	2/3 拆分视图	紧凑的宽度, 常规的高度	常规的宽度, 常规的高度
	1/2 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
	1/3 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
9.7" iPad	2/3 拆分视图	紧凑的宽度, 常规的高度	常规的宽度, 常规的高度
	1/2 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
	1/3 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
7.9" iPad mini 4	2/3 拆分视图	紧凑的宽度, 常规的高度	常规的宽度, 常规的高度
	1/2 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度
	1/3 拆分视图	紧凑的宽度, 常规的高度	紧凑的宽度, 常规的高度

布局注意事项

确保重要内容在默认大小下清晰可读：除非用户选择调整大小，否则不应该让用户横向滑动才能阅读重要的文字信息，或者放大才能看清重要的图片。

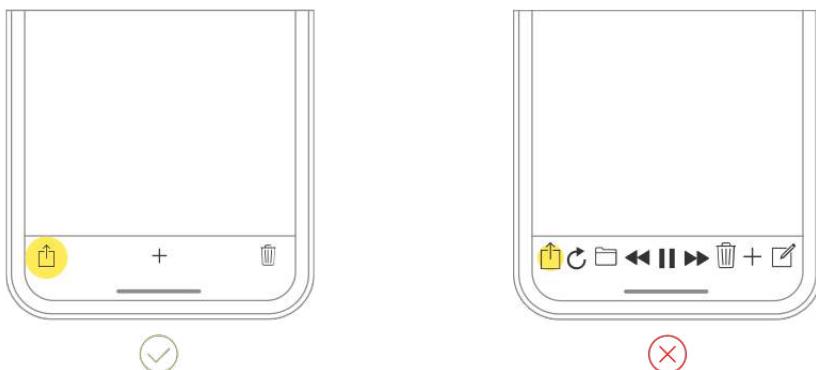
在整个 App 中保持一致的视觉外观：一般来说，具有相似功能的元素应该看起来类似。

利用视觉权重和平衡来表示重要性：大的对象比小的对象更能抓住人的眼球，显得更重要。大的对象也更容易点击，这在容易分散注意力的使用环境中（如厨房或健身房）使用时这点尤为重要。一般来说，处于从左往右的阅读环境时把首要的对象放在屏幕的上半部分并且偏左的位置。

利用对齐来方便浏览，并且表达结构和层级：对齐会让 App 看起来整齐有次序，当页面滑动时有助用户聚焦，更容易找到信息。缩进和对齐还可以表明多组内容之间的关系。

尽可能同时支持纵向视图和横向视图：对齐使应用程序看起来整洁有序，帮助人们在滚动页面时集中注意力，这样就更容易查找信息。缩进和对齐还可以暗示一组内容之间的相关性。

准备好应对文本大小的改变：当用户在设置里选择了不同的文本大小，他们总是希望大部分的 App 都能适配。为了适应某些文本大小的改变，你可能需要调整布局。了解更多请参阅 [Typography](#)。



为可交互元素踢狗足够的触发区域：尽可能为所有控件提供 44ptx44pt 的最小可点击区域。



在多种设备上预览您的 APP: 你可以使用模拟器 (Xcode 中自带) 来预览您的 APP，检查是否出现裁切或其他布局上的问题。如果您的应用支持横屏模式，请确保您的布局无论在设备向左还是向右旋转的横屏下都能看起来良好。全面屏手机不支持页面上下倒置。一些功能，比如广色域的图像，还是在实际设备上预览比较好。

在较大的设备上显示文本时，应在两边留足边距以确保可读性：这些边距让每一行文字保持在一个让人读起来舒适的长度。

根据情景自动适应

环境变化时保持当前内容的焦点不变：内容是你的最高优先级。让焦点随着

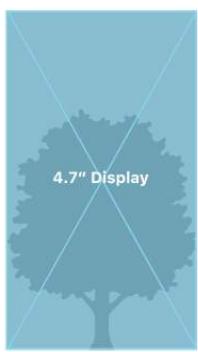
环境变化而发生改变是会让用户困惑的，感觉当前的 App 失控了。

避免无缘由的布局变动：即使用户旋转了设备，也不代表整体的布局需要变换：比如，如果你的 App 在竖屏模式展示了一网格的图片，那么在横屏模式你没必要展示同样的图片。相反地，你只需要简单地调整网格的尺寸就行了。尽量在任何环境下都能维持一样的体验。

如果你的 App 只支持一种模式，那么请提供两种屏幕模式的变量：如果你的 App 只能在一种模式下运行，那么确保它能够支持该模式的两种方向变化是十分重要的。比如，如果你的 App 只在横屏模式运行，那么无论 Home 键在左边还是右边，应用都该能正常使用。如果设备被旋转 180 度，那么你的 App 内容也该同时旋转 180 度。反之，当用户拿错设备方向时，你的 App 没有自动旋转，那么他们就会很自然地知道应该旋转设备。你无需告诉他们该如何纠正。

根据当前使用内容来规定相应的旋转方向：比如，一个需要用户旋转设备来控制角色移动的游戏，就不会在游戏中改变横竖屏的方向。但是，它可以根据当前设备的旋转方向来展示菜单和引导步骤。

确保你的应用在 iPad 上也能运行：用户很高兴能够灵活地在任一类型的 iOS 设备上运行你的应用。即使你希望大多数人在 iPhone 上使用你的应用，但其界面元素在 iPad 上仍需保持可见并且实用。如果应用的某些功能需要特定于 iPhone 的硬件（如 3D Touch），请考虑在 iPad 上隐藏或禁用这些功能，并让用户使用应用的其他功能。



4.7英寸设备全屏图像



在 iPhone X 上被裁切的部分



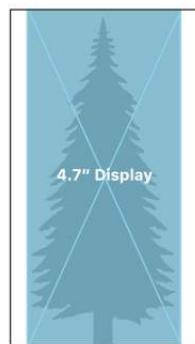
iPhone X 上的信箱模式



iPhone X 全屏图像



在4.7英寸设备上被裁切的部分



4.7英寸设备上的邮筒模式

复用现有的图像时，请注意不同屏幕的长宽比差异：不同的屏幕尺寸可能拥有不同的长宽比，导致您的图像被信箱模式或邮筒模式剪裁。确保那些重要的视觉内容在所有尺寸的屏幕下都还完整的在视图里。

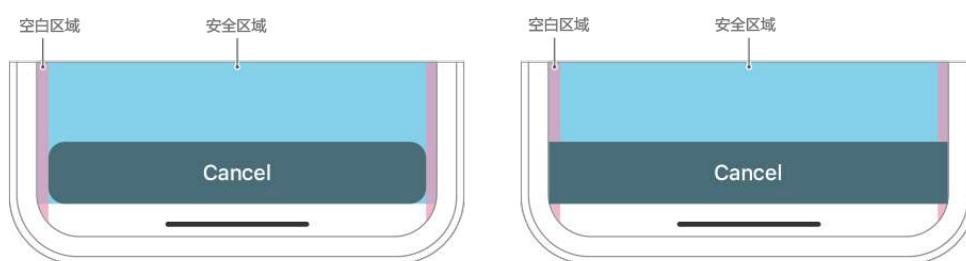
为全面屏体验做设计

将视觉元素充满屏幕：确保背景扩展到显示屏的边缘，并且是垂直可滚动的布局，如列表视图和照片精选页面，可以一直延伸到底部。

不要将交互式控件放置在屏幕底部和角落：如果放置在屏幕底部，由于用户要使用显示屏底部的滑动手势来访问主屏幕和切换 App 程序，这些手势可能会使你在此区域中设置的自定义手势失效。如果放置在屏幕上方的两个角落，则用户不能够轻易触及，从而给操作带来不便。



嵌入的重要内容要防止被裁切：一般来说，内容应该是居中对称的，这样它从任何方向看起来效果都很好，不会被圆角、设备的传感器外壳截断或被用于访问主屏幕的指示器遮挡住。为获得最佳效果，请使用系统自带的界面元素和响应式布局来创建你的界面。所有 App 都应遵循 UIKit 定义的安全区域和布局边距，这些区域可以根据设备和上下文进行适当的填充。安全区域还可以防止状态栏、导航栏、工具栏和标签栏被内容覆盖。



插入全宽按钮：需要注意全宽按钮两侧的标准UIKit边距，因为延伸到屏幕

边缘的按钮可能看起来不像一个按钮。出现在屏幕底部的全宽按钮看起来最好，它具有圆角并与安全区域的底部对齐，这也确保它不会与Home键发生冲突。

不要隐藏或使关键的显示特性太醒目：不要在屏幕顶部和底部放置黑色区块来隐藏设备的圆角、传感器外壳和用于访问主屏幕的指示器。不要使用诸如括号，边框，形状或引导文字等视觉装饰来使这些区域更加醒目。

注意状态栏的高度：在 iPhone X 上，状态栏比其他 iPhone 要高。如果你在自己的 App 中自定义了一个位于状态栏下方的固定元素，则必须更新你的 App 才能根据用户的设备动态地定位该元素。 (译者注)

请注意，当后台任务如录音和定位，处于激活状态时，iPhone X 上的状态栏高度不变。

如果你的 App 目前隐藏了状态栏，在 iPhone X 上需要重新考虑这一设定：状态栏会占据屏幕的一块区域从而导致 App 无法充分利用这部分空间，但由于状态栏可能会显示对用户有用的信息，并且相比于4.7英寸的 iPhone，iPhone X 的显示屏在竖向上能显示更多内容，因此，应当重新考虑是否要隐藏 iPhone X 上的状态栏。

允许自动隐藏访问主屏幕的指示器：当启用自动隐藏时，如果用户几秒钟内没有触摸屏幕，指示灯将被隐藏。当用户再次触摸屏幕时，指示灯会再次出现。这种行为只适用于被动观看体验吗，如播放视频或照片幻灯片。

请参阅 [适应性和布局](#)。

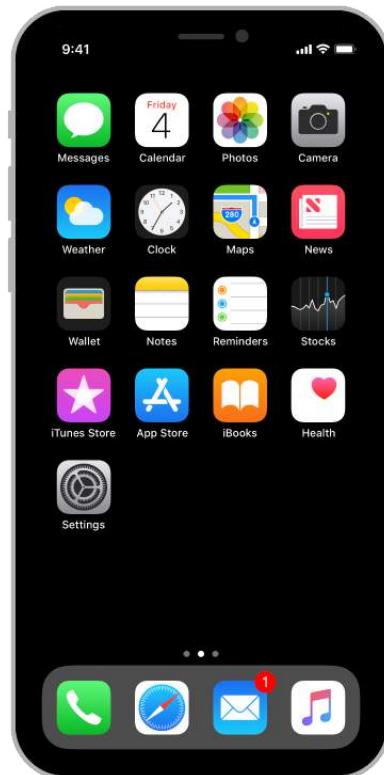
注：例如考虑到本来状态栏是 20pt，考虑到它的占用空间你自定义了一个距离顶部 20pt 的 Banner，那么因为在 iPhone X 的状态栏更高可能就会导致原本正好的模块边缘出现挤占

其他关于布局的注意事项

确保您的网站在边缘视图中看起来依然很棒。参阅 webkit.org 中的为 iPhone X 设计网站。

5.2 动效

贯穿于 iOS 系统的优美、精细的动画在用户和屏幕屏幕内容之间建立了一种视觉上的联系。当动画被合理利用时，它能够表达状态、提供反馈、加强直接操纵感，并且视觉化呈现用户的操作结果。



明智且审慎地使用动画和动效：不要为了使用动画而动画。过度或是无理由的动画会让用户有不连贯和错乱的感觉，尤其是在那些不能提供沉浸式体验的 App 中。iOS 经常使用动效，比如在主屏和其它地方使用了视差效果，来建立用户的沉浸感受。这些效果有助于增强理解和提升愉悦感，但是滥用它们就会让一个 App 变得难以控制。如果你想使用动效，一定要提前进行用户测试以保证它们真的能不辱使命。

力求真实性和可信性：用户可以接受艺术创造，但是当动效没有意义或是违背了物理定律时，用户就会感到混乱。打个比方，如果用户通过在屏幕顶部下滑出一个视图，那么他们应该也能通过上滑将该视图关闭。

使用连贯的动画：一个熟悉并流畅的体验能一直让用户参与其中。用户已经习惯了贯穿于 iOS 系统的精细动画，比如平稳的过渡、横竖屏之间的流畅转换和基于物理现实的滚动。除非你在创造一个沉浸式体验，比如游戏，不然自定义动画都应该和系统动画相符。

让动效可选：当在辅助功能偏好设置中启用减少动作的选项时，App 应尽可能减少或消除应用程序动画。

5.3 品牌化

成功的品牌化不仅是单纯地在应用中添加品牌元素。优秀的 App 通过优雅别致的文字、颜色和图片来营造独特的品牌辨识度。提供足够多的品牌元素让用户感觉是处在你的 App 中，但不要因为过度使用而造成干扰。



融入精妙的、不唐突的品牌元素：用户使用你的 App 是获得娱乐、得到信息或是做某些事情的，而不是为了观看一个广告。要想达到最好的体验，请巧妙地将品牌融于 App 设计中。让 App 图标的颜色贯穿整个 App 设计，不失为一种很好的专属使用界面。

不要让品牌化阻碍了优秀的 App 设计：首先，让你的 App 像是一个 iOS App。保证它是直观的、易于导航的、易用的并且以内容为中心的。当你的 App 在其它平台也适用，不要为了保持品牌的一致性而牺牲了设计的质量。

内容比品牌化更重要：在屏幕顶部放置一个除了展示品牌元素以外没有任何用途的头部栏，就意味着牺牲了用来浏览内容的空间。取而代之的，考虑采用低侵入性的方式来实现品牌化，比如使用自定义的配色方案和字体，或是巧妙地自定义背景。

克制住想要在应用中到处展示 logo 的诱惑：避免在 App 中到处展示 logo，除非它是品牌化中必不可少的一部分。这点在导航栏中尤其重要，因为提供一个标题比 logo 更加有用。

遵循 Apple 的商标准则：Apple 的商标不能在你的 App 名字或是图像中出现。请参阅 [Apple Trademark List](#) 和 [Guidelines for Using Apple Trademarks](#)。

注意：App Store 为突出你的品牌提供了更多的机会。了解相关指导，请参阅 [App Store Marketing Guidelines](#)。

5.4 色彩

色彩能够很好地赋予页面活力，提供视觉上的连续性；表达不同的状态信息，对用户的操作提供反馈；并将可视化的数据提供给用户。如果您想为自己的应用挑选单独的色彩或一整套色彩组，如果您希望让它们同时在浅色或深色模式中都能看起来良好，挑选之前请参考我们的系统配色方案。

明智地使用颜色进行沟通：慎地使用色彩，那么色彩可以更有效得强调重要信息，吸引用户的注意力。例如，如果在应用中其他一些不是那么关键的地方使用红色，那么提醒用户重要信息的红色三角将变得不那么显眼。

在 App 内使用互补的颜色：你的 App 内的颜色应该协调，不会互相冲突和干扰。如果你的 App 风格的基础色调是柔和的，那么使用一系列与之协调的柔和色调。

一般来说，选择与你的 App Logo 相符的颜色地色板：巧妙地使用颜色是一个传达品牌的好办法。

在 App 中统一使用一种关键色来显示可交互性：在备忘录中，可交互的元素是黄色的。在日历中，可交互的元素是红色的。如果你定义了一种关键色用于表达可交互性，那么你要保证其它颜色不会与之冲突。

提供两种版本的关键色，以确保它们在浅色和深色模式下都很好看：如果您使用系统色彩作为关键色，那么在浅色和深色模式下它们会自动切换来保证高对比度。

注意图像和透明度会影响周围元素的颜色：图像的变化有时可以改变附近的颜色，以保持视觉连续性，并防止界面元素变得太强烈或者不能引起用户的注意。例如，地图在使用地图模式时显示浅色方案，但在激活卫星模式时切换到深色方案。当放置在透明元素后面时，或当应用于透明元素（例如工具栏）时，颜色也可能会不同。

在多种光线条件下测试您 App 颜色：光线在室内和室外、房间氛围、不同的时间、气候等条件下会发生明显地变化。您的 App 在实际使用时看到的颜色不会一直和你在电脑上看到的颜色相同。您应该在不同的光线条件下进行预览，来观察颜色的真实表现，比如在晴朗的户外。必要时，应当调整颜色以求在大多数的使用场景下提供最好的视觉体验。

考虑 True Tone 对颜色的影响：True Tone 显示屏利用了环境光传感器来自动调整白色参考点，以适应当前环境下的光线情况。专注于阅读、照片、视频和游戏的 App 可以通过调整白平衡来强化或弱化 True Tone 的效果。开发者请参阅 [UIWhitePointAdaptiveStyle](#)。

请注意不同国家和文化对您使用的颜色会有不同的认知：例如，在某些文化中，红色代表着危险。但在其他文化中，红色具有积极的内涵。确保应用中的颜色传达的是同一个意思。

不要使用低对比度颜色，使用户难以看到内容：比如说，色盲用户可能无法分辨一些色彩组合，对比度不足导致了图标和文字与背景混在一起，用户难

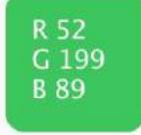
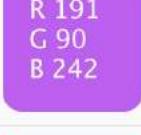
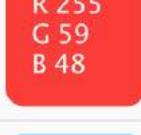
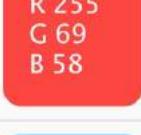
以阅读内容。详情请参阅 [Color and Contrast](#)。

系统色彩

iOS 提供了一系列系统颜色，可自动适应用户对无障碍设置做出的更改，如增加对比度和降低透明度。系统颜色在浅色和深色背景以及浅色和深色模式下都可以单独和组合使用，而且效果很棒。

不要把系统颜色的值写死在您的 APP 中：下方提供的系统颜色的值仅仅只作为设计时的参考，实际使用的色彩应当根据不同的环境，随着版本的变更而有所变动。**始终使用 API 来引用系统色**，开发者请参阅 [UIColor](#)。

默认

Light	Dark	Name	API
 R 0 G 122 B 255	 R 10 G 132 B 255	Blue	systemBlue
 R 142 G 142 B 147	 R 152 G 152 B 157	Gray	systemGray
 R 52 G 199 B 89	 R 48 G 209 B 88	Green	systemGreen
 R 88 G 86 B 214	 R 94 G 92 B 230	Indigo	systemIndigo
 R 255 G 149 B 0	 R 255 G 159 B 10	Orange	systemOrange
 R 255 G 45 B 85	 R 255 G 55 B 95	Pink	systemPink
 R 175 G 82 B 222	 R 191 G 90 B 242	Purple	systemPurple
 R 255 G 59 B 48	 R 255 G 69 B 58	Red	systemRed
 R 90 G 200 B 250	 R 100 G 210 B 255	Teal	systemTeal
 R 255 G 204 B 0	 R 255 G 214 B 10	Yellow	systemYellow

无障碍

Light	Dark	Name	API
 R 0 G 64 B 221	 R 64 G 156 B 255	Blue	systemBlue
 R 105 G 105 B 110	 R 152 G 152 B 157	Gray	systemGray
 R 36 G 138 B 61	 R 48 G 219 B 91	Green	systemGreen
 R 54 G 52 B 163	 R 125 G 122 B 255	Indigo	systemIndigo
 R 201 G 52 B 0	 R 255 G 179 B 64	Orange	systemOrange
 R 211 G 15 B 69	 R 255 G 100 B 130	Pink	systemPink
 R 137 G 68 B 171	 R 218 G 143 B 255	Purple	systemPurple
 R 215 G 0 B 21	 R 255 G 105 B 97	Red	systemRed
 R 0 G 113 B 164	 R 112 G 215 B 255	Teal	systemTeal
 R 160 G 90 B 0	 R 255 G 212 B 38	Yellow	systemYellow

iOS13 也引入了六种不透明的灰色，在少数透明色不好使的情况下可以使用它们。比如说，类似网格中横线和竖线这种相交的元素，使用不透明色显然更好。一般来说，使用语义化的描述来定义系统色彩。

默认

Light	Dark	Name	API
 R 142 G 142 B 147	 R 142 G 142 B 147	Gray	<code>systemGray</code>
 R 174 G 174 B 178	 R 99 G 99 B 102	Gray (2)	<code>systemGray2</code>
 R 199 G 199 B 204	 R 72 G 72 B 74	Gray (3)	<code>systemGray3</code>
 R 209 G 209 B 214	 R 58 G 58 B 60	Gray (4)	<code>systemGray4</code>
 R 229 G 229 B 234	 R 44 G 44 B 46	Gray (5)	<code>systemGray5</code>
 R 242 G 242 B 247	 R 28 G 28 B 30	Gray (6)	<code>systemGray6</code>

无障碍

Light	Dark	Name	API
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 108 G 108 B 112</div>	<div style="background-color: #333; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 174 G 174 B 178</div>	Gray	<code>systemGray</code>
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 142 G 142 B 147</div>	<div style="background-color: #333; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 124 G 124 B 128</div>	Gray (2)	<code>systemGray2</code>
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 174 G 174 B 178</div>	<div style="background-color: #333; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 84 G 84 B 86</div>	Gray (3)	<code>systemGray3</code>
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 188 G 188 B 192</div>	<div style="background-color: #333; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 68 G 68 B 70</div>	Gray (4)	<code>systemGray4</code>
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 216 G 216 B 220</div>	<div style="background-color: #333; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 54 G 54 B 56</div>	Gray (5)	<code>systemGray5</code>
<div style="background-color: #e0e0e0; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 235 G 235 B 240</div>	<div style="background-color: black; width: 40px; height: 40px; display: flex; align-items: center; justify-content: center;">R 36 G 36 B 38</div>	Gray (6)	<code>systemGray6</code>

动态的系统色

除了关键色彩外，iOS 还提供了用语义定义的系统色，它们可以被用于背景区域和前景内容，例如标签，分隔线和填充色。这些颜色自动适应浅色和深色模式。

iOS 提供了两套背景色彩——默认和成组，每套包含一级、二级、三级色彩来帮助您构建信息的层级。一般来说，界面中有成组的列表内容的情况下，

使用成组背景色；否则使用默认的系统背景色。

对任意一组背景色，您可以使用以下的方法来暗示层级结构：

- 整体背景的视图使用一级色彩；
- 背景中的元素或组使用二级色彩；
- 二级元素或组中的元素或组使用三级色彩；

针对前景内容，iOS 定义了以下色彩：

颜色	应用于	API
字段	包含基础内容的文字段	Label
次级字段	包含次级内容的文字段	secondaryLabel
三级字段	包含三级内容的文字段	tertiaryLabel
四级字段	包含四级内容的文字段	quaternaryLabel
占位符字段	控件或输入框中的占位符	placeholderText
分隔符	底层内容可见的分隔符	separator
不透明分隔符	底层内容不可见的分隔符	opaqueSeparator
链接	有链接作用的字段	link

不要重新自定义系统色的语义含义：为了提供给用户一致的体验，并且确保您的界面在任何情境下看起来都很棒，按预期使用动态的系统色彩。

请不要尝试复制动态系统色：动态系统色可能会根据大环境的变化，随着版本的迭代而变更。与其为了搭配系统色自定义创建一堆色彩，不如戳用动态系统色。

苹果的意思是，他们的色彩今后可能还会改来改去，所以使用 API 应用色彩，而不是把色值写在应用里。

5.5 深色模式

在 iOS13 以及更高版本中，用户可以选择一种称为深色模式的系统级黑暗主题。在深色模式中，系统在所有视图、菜单、控件中采用了一种更黑的色板，并且它使用了一种更为醒目的色彩让前景内容在背景中脱颖而出。深色模式支持所有无障碍特性。

用户可以将深色模式设定为他们默认的界面风格，也可以设定当环境光线变暗时自动切换至深色模式，这些都能在「设置」中设置。

专注于内容：深色模式将焦点放在界面的内容区域，让内容脱颖而出，而其他 UI 元素隐于黑暗。

在浅色和深色模式中审视您的设计：您需要了解您的设计在两种模式下分别有怎样的表现，如有必要请调整您的设计，来适应每种模式。在一种模式下看起来不错的设计，在另一种模式下未必如此。

当调整辅助功能中的透明度和对比度时，确保您的内容在深色模式下仍能保持舒适和清晰：在深色模式下，您需要在「增强对比度」和「降低透明度」打开的情况下测试您的内容，任意单个打开和两个全开都需要测试。您可能因此发现一些黑色文字在黑色模式下显示并不清晰，也会发现深色模式下打开增强对比度可能会导致黑色字段与黑色背景的对比度反而降低了。尽管视力较好的用户仍然可以阅读低对比度的文字，但是这些文字对视障用户来说并不友好。

色彩

深色模式的色板包含了更暗的背景色和更亮的前景色。为了确保对比度的同时保持两种模式下的一致性，这些色彩都经过了我们的精心挑选。

使用适应当前外观的色彩：语义化的色彩，比如「分隔符」，能够自动适应当前外观。但当你使用自定义色彩时，将色彩资源集添加到应用的资源目录中，并指定浅色和深色模式使用哪一个，以便它可以适应当前的外观模式。

保证在深浅模式下颜色都能拥有足够的对比度：使用系统定义的色彩能够确保前后景内容拥有合适的对比度。若使用自定义色彩，目标对比度为7：1，尤其是小字段。详情参阅[动态系统色](#)。

白色背景需要稍微柔和一些：若您必须在深色模式下使用白色的内容背景，请使用稍暗一些的白色，防止纯白在深色环境下形成背景发光的感觉。

相关规范请参阅[色彩章节](#)。

图片、图标以及符号的色彩

新系统引入了[SF Symbols](#)，一种能够自动适应深色模式的视觉需要，并在深浅模式下都经过了优化的全彩图像。

尽可能使用 SF Symbols：当您使用动态色彩对其进行着色或添加更多状态时，SF Symbols 在两种外观模式下看起来都很棒。

必要时为浅色和深色模式分别设计图形：在浅色模式下使用线性的图形，在深色模式下可能用实心、填充的会更好一些。

确保全彩图像和图标看起来很好：如果在两种模式下看起来都不错，那请使用相同的资源库；但若资源库只在一种模式下看起来正常，那修改资源库或者创建一个独立的浅色/深色资源库。使用资源目录将您的资源库（们）合并为一个单独的、被命名的图像。

文字色彩

醒目的色彩能让您的文字段在深色背景下保持对比度。

标签（文字段）请使用系统提供的标签色：四种层级的标签色能够自适应浅色和深色外观。相关规范请参阅 [Typography](#)。

使用系统定义的视图绘制文本框和文本视图：系统视图和控件是您 APP 的文本在两种背景上看起来正常，它们能够自动调整为暗色或是亮色。如果文本能够用系统视图呈现，那么尽量不要自己绘制视图。开发者请参阅 [UITextField](#) 和 [UITextView](#)。

5.6 材质

iOS 提供的材质（或背景模糊）可以创建半透明效果，这能够让用户感觉到深度。在视图或控件的背景上使用材质效果能够避免分散前景内容的吸引力。为了达到这个效果，材质允许背景的色彩信息穿透前景视图，同时为了保证清晰度，添加了背景模糊。

如果您使用系统定义的材质，您的元素在任何环境中都能有不错的表现，因

为材质能够根据浅色和深色模式自动调整。

TIP

材质需要在构建应用的 View 时使用 Visual Effect View 的 API，开发者请参阅 [UIVisualEffectView](#)。

遵循系统使用的材质：尽可能用自定义的材质视图与系统自带的材质视图进行比较，确保相似功能的视图使用相同的材质。

利用系统提供的文字色、填充色、图形色和分隔符色：系统提供的色彩能够自适应半透明背景，而使得效果看起来依然很棒。详情参阅[动态系统色](#)。

尽可能在材质上使用 SF Symbols：当您使用动态系统色为符号着色、或为其添加醒目色效果时，符号在任何背景上都能看起来很棒。相反，全彩图像与背景相比就可能对比度不足，而且在半透明背景上看起来不是很合适。详情请参阅[SF Symbols](#)。

系统定义的材质与醒目色

iOS 定义了您能够在特定区域的几种材质，以控制背景外观和前景内容之间视觉独立。系统提供的材质包含浅色和深色的样式，这些样式在绝大多数背景上都表现良好。

为了在内容卡片中使用，iOS13 定义了四种具有不同半透明度的材质（每种材质还包括一个深色版本）：

- [SystemUltraThinMaterial](#)
- [SystemThinMaterial](#)
- [SystemMaterial](#) (默认)

- [SystemThickMaterial](#)

在选择材质时考虑对比度以及视觉独立性：当您选择材质以及应用在其上的醒目色的效果时，并没有绝对的规范。鉴于这需要您自己作出决定，考虑以下方面：

- 越厚的的材质能够为精细的文本或其他元素提供约好的对比度
- 越透明的材质越能够在视觉上提醒用户背景上存在的内容，以保持情景之间的联系

iOS 还为文字标签、填充色以及分隔符定义了专门用在各种材质上的醒目色。醒目色通过采样背景色彩以及调整饱和度值来使 UI 元素变亮或变暗。醒目的 UI 元素与材质融为一体，更好地提升了半透明效果的观感。

文字标签与填充色各提供了几组不同级别的醒目色，分隔符只有一种。级别的名称代表了元素与背景之间对比度的相对值：默认的级别拥有最高的对比度，而四级 (如果有的话) 则对比度最低。

除了四级，你可在任意材质上为文字标签使用以下的醒目色的值。不推荐在薄 (thin) 和超薄 (Ultrathin) 的材质上面使用四级醒目色，那样对比度太低。

- [Label](#) (默认)
- [secondaryLabel](#)
- [tertiaryLabel](#)
- [quaternaryLabel](#)

你可以再任意材质上为填充块使用以下的醒目色：

- [Fill](#) (默认)

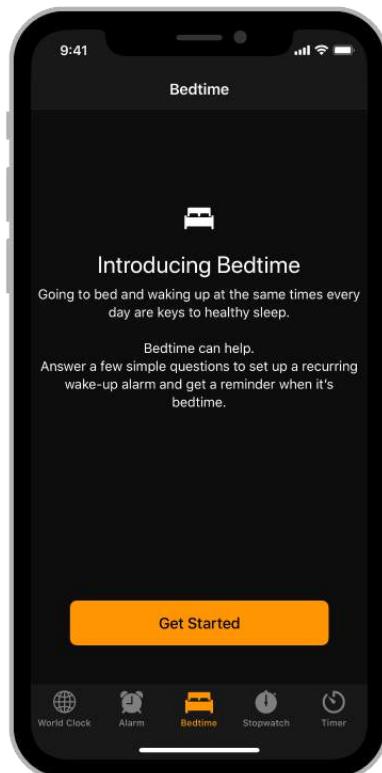
- [secondaryFill](#)
- [tertiaryFill](#)

iOS 为分隔符定义了一个单独的、默认的醒目色值：[separator](#)，它能在任意材质上使用。

选择醒目色的效果时一定要遵从语义：不要混用这些效果，比如不要把文字标签的效果用在分隔符上。

5.7 术语

App 中的每一字都是与你的用户对话的一部分。用此对话来帮助他们在你的 App 中感到舒适。



使用熟悉的、可以理解的单词和短语：技术性词语可能会令人生畏。避免人们可能不理解的缩写和技术术语。利用你对用户的了解来确定某些词或短语是否合适。一般来说，对每个人都有吸引力的 App 应该避开高技术性的语言。这类语言可能适用于面向更高级的用户或技术人群。

保持界面文本清晰简洁：人们快速而容易地吸收简短的、直接的文本，不喜欢被强迫阅读冗长的文章来完成一项任务。找出最重要的信息，简明扼要地表达出来，并把它呈现在显眼的位置，这样人们就不必为了找到他们想要的东西而阅读太多，或者不知道下一步该做什么。

适当的表明交互元素：人们应该能够一眼看出元素的作用。当标记按钮和其他交互元素时，使用动作动词（action verbs），例如连接、发送和添加。

避免那些听起来可能会显得屈尊俯就的语言：避免使用“我们”，“我们的”，“我”和“我的”（例如“我们的教程”和“我的训练”）。它们有时会被理解为侮辱或傲慢。

尽量用一种非正式的、友好的语气：一种非正式的、平易近人的风格，与你在午餐时与人交谈的方式相似。偶尔使用缩写，直接用“你”和“你的”来对用户说话。

要小心的使用幽默感：记住，人们很可能多次在你的界面中阅读文本，而一开始看起来很聪明的东西可能会随着时间的推移而变得令人厌烦。同时要记住，一种文化中的幽默不一定能很好地转化为其他文化的幽默。

使用相关的、一致的语言和图像：一定要确保样式对当前的上下文而言是合适的。例如，如果有人使用 iPad，就不要显示 iPhone 的提示或图形。使用与平台一致的语言。你可以在触摸屏上点击、快速滑动、滑动、缩放和拖

拽。你按下物理按键和对 3D Touch 做出反应的按钮。你还可以旋转并摇动设备。**要准确表达日期：**使用像 “今天” 和 “明天” 这样的友好术语是合适的，但是如果您不考虑当前的语言环境，这些术语可能会令人困惑或不准确。考虑一个在午夜前开始的活动。在一个时区，活动可能会在今天开始。在另一个时区，同样的事件可能在昨天开始。一般来说，日期应该反映出观看事件的人的时区。然而，在某些情况下，例如在飞行跟踪 App 中，可以更清楚地显示航班的起始日期和时区。

5.8 排版

San Francisco (SF) 是 iOS 系统的默认英文字体：这种字型的字体经过了优化，使你的文本具有无与伦比的可读性、清晰性和一致性。点击[这里](#)下载 San Francisco 系列字体。

(23pt)	John Appleseed
(22pt)	John Appleseed
(21pt)	John Appleseed
SF Pro Display (20pt)	<u>John Appleseed</u>
SF Pro Text (19pt)	John Appleseed
(18pt)	John Appleseed
(17pt)	John Appleseed
(16pt)	John Appleseed

iOS 使用 San Francisco 字体作为拉丁、希腊和西里尔字母的系统字体，以及其他的各种字体。

强调重要信息：使用字重、字体大小和颜色来突出 App 中最重要的信息。

如果可以的话，只使用一种字体：混合几种不同的字体可能会让你的 App

显得支离破碎和草率。考虑使用一种字体，只使用少量的字体的变体和大小

尽可能使用内置的文本样式：内置的文本样式允许您以视觉上不同的方式来

表达内容，同时保持最优的易读性。这些样式是基于系统字体的，允许你利

用关键的排版特性，比如动态类型，它自动调整跟踪，并为每种字体大小提

供引导。

iOS 包含以下文本样式			
大标题	三级标题	插图编号	一级说明
一级标题	内容提要	小标题	二级说明
二级标题	正文	脚注（补充说明）	

开发者指南，请参阅 [UIFontTextStyle](#)。

确保字体是清晰的：在 iOS 上支持自定义字体，但通常不利于阅读。除非你的应用对定制字体有迫切需求，比如为品牌设计或创造一种沉浸式游戏体验，否则最好还是坚持使用系统字体。如果你使用自定义字体，请确保它易于阅读，即使是很小的字体。

实现自定义字体的辅助功能：系统字体会自动对辅助功能做出反应，比如加粗字体和放大文本。使用自定义字体的App应检查是否启用辅助功能并实现相同的行为。

请参阅 [Accessibility](#)。

动态类型大小

San Francisco 字体的设计的大号字体和小号字体都很清晰。动态类型通过让读者选择他们喜欢的文本大小来提供额外的灵活性。

在响应文本大小的变化时，优先考虑内容：并不是所有的内容都同样重要。

当用户选择更大的尺寸时，他们想要让他们所关心的内容更容易阅读;他们并不总是希望屏幕上的每一个字都变大。

很小的尺寸

类型	字重	磅值	行距	字距
大标题	常规	31pt	38pt	12pt
一级标题	常规	25pt	31pt	14pt
二级标题	常规	19pt	24pt	-26pt
三级标题	常规	17pt	22pt	-24pt
内容提要	半粗体	14pt	19pt	-11pt
正文	常规	14pt	19pt	-11pt
插图编号	常规	13pt	18pt	-6pt
副标题	常规	12pt	16pt	0pt
脚注	常规	12pt	16pt	0pt
一级说明文字	常规	11pt	13pt	6pt
二级说明文字	常规	11pt	13pt	6pt

小尺寸

类型	字重	磅值	行距	字距
大标题	常规	32pt	39pt	12pt
一级标题	常规	26pt	32pt	14pt
二级标题	常规	20pt	25pt	19pt
三级标题	常规	18pt	23pt	-25pt
内容提要	半粗体	15pt	20pt	-16pt

类型	字重	磅值	行距	字距
正文	常规	15pt	20pt	-16pt
插图编号	常规	14pt	19pt	-11pt
副标题	常规	13pt	18pt	-6pt
脚注	常规	12pt	16pt	0pt
一级说明文字	常规	11pt	13pt	6pt
二级说明文字	常规	11pt	13pt	6pt

中等尺寸

类型	字重	磅值	行距	字距
Large Title	常规	33pt	40pt	11pt
Title 1	常规	27pt	33pt	13pt
Title 2	常规	21pt	26pt	17pt
Title 3	常规	19pt	24pt	-26pt
Headline	半粗体	16pt	21pt	-20pt
Body	常规	16pt	21pt	-20pt
Callout	常规	15pt	20pt	-16pt
Subhead	常规	14pt	19pt	-11pt
Footnote	常规	12pt	16pt	0pt
Caption 1	常规	11pt	13pt	6pt
Caption 2	常规	11pt	13pt	6pt

大尺寸 (默认的)

类型	字重	磅值	行距	字距
大标题	常规	34pt	41pt	11pt
一级标题	常规	28pt	34pt	13pt
二级标题	常规	22pt	28pt	16pt
三级标题	常规	20pt	25pt	19pt

类型	字重	磅值	行距	字距
内容提要	半粗体	17pt	22pt	-24pt
正文	常规	17pt	22pt	-24pt
插图编号	常规	16pt	21pt	-20pt
副标题	常规	15pt	20pt	-16pt
脚注	常规	13pt	18pt	-6pt
一级说明文字	常规	12pt	16pt	0pt
二级说明文字	常规	11pt	13pt	6pt

非常大的尺寸

类型	字重	磅值	行距	字距
大标题	常规	36pt	43pt	11pt
一级标题	常规	30pt	37pt	12pt
二级标题	常规	24pt	30pt	35pt
三级标题	常规	22pt	28pt	16pt
内容提要	半粗体	19pt	24pt	-26pt
正文	常规	19pt	24pt	-26pt
插图编号	常规	18pt	23pt	-25pt
副标题	常规	17pt	22pt	-24pt
脚注	常规	15pt	20pt	-16pt
一级说明文字	常规	14pt	19pt	-11pt
二级说明文字	常规	13pt	18pt	-6pt

特别大的尺寸

类型	字重	磅值	行距	字距
大标题	常规	40pt	48pt	10pt
一级标题	常规	34pt	41pt	11pt
二级标题	常规	28pt	34pt	13pt

类型	字重	磅值	行距	字距
三级标题	常规	26pt	32pt	14pt
内容提要	半粗体	23pt	28pt	16pt
正文	常规	23pt	28pt	16pt
插图编号	常规	22pt	27pt	16pt
副标题	常规	21pt	26pt	17pt
脚注	常规	19pt	24pt	-26pt
一级说明文字	常规	18pt	23pt	-25pt
二级说明文字	常规	17pt	22pt	-24pt

在 [Resources](#) 中下载一个动态类型大小的表格。

更大的可访问类型尺寸

除了标准的动态类型大小之外，系统还为具有应用需求的用户提供了一些更大的类型。

AX1

类型	字重	磅值	行距	字距
大标题	常规	44pt	52pt	9pt
一级标题	常规	38pt	46pt	11pt
二级标题	常规	34pt	41pt	11pt
三级标题	常规	31pt	38pt	12pt
内容提要	半粗体	28pt	34pt	13pt
正文	常规	28pt	34pt	13pt
插图编号	常规	26pt	32pt	14pt
副标题	常规	25pt	31pt	14pt
脚注	常规	23pt	29pt	16pt
一级说明文字	常规	22pt	28pt	16pt

类型	字重	磅值	行距	字距
二级说明文字	常规	20pt	25pt	19pt

AX2

类型	字重	磅值	行距	字距
大标题	常规	48pt	57pt	8pt
一级标题	常规	43pt	51pt	10pt
二级标题	常规	39pt	47pt	10pt
三级标题	常规	37pt	44pt	11pt
内容提要	半粗体	33pt	40pt	12pt
正文	常规	33pt	40pt	11pt
插图编号	常规	32pt	39pt	12pt
副标题	常规	30pt	37pt	12pt
脚注	常规	27pt	33pt	13pt
一级说明文字	常规	26pt	32pt	14pt
二级说明文字	常规	24pt	30pt	15pt

AX3

类型	字重	磅值	行距	字距
大标题	常规	52pt	61pt	7pt
一级标题	常规	48pt	57pt	8pt
二级标题	常规	44pt	52pt	9pt
三级标题	常规	43pt	51pt	10pt
内容提要	半粗体	40pt	48pt	10pt
正文	常规	40pt	48pt	10pt
插图编号	常规	38pt	46pt	11pt
副标题	常规	36pt	43pt	11pt
脚注	常规	33pt	40pt	11pt

类型	字重	磅值	行距	字距
一级说明文字	常规	32pt	39pt	12pt
二级说明文字	常规	29pt	35pt	13pt

AX4

类型	字重	磅值	行距	字距
大标题	常规	56pt	66pt	5pt
一级标题	常规	53pt	62pt	6pt
二级标题	常规	50pt	59pt	7pt
三级标题	常规	49pt	58pt	8pt
内容提要	半粗体	47pt	56pt	9pt
正文	常规	47pt	56pt	9pt
插图编号	常规	44pt	52pt	9pt
副标题	常规	42pt	50pt	10pt
脚注	常规	38pt	46pt	11pt
一级说明文字	常规	37pt	44pt	11pt
二级说明文字	常规	34pt	41pt	11pt

AX5

类型	字重	磅值	行距	字距
大标题	常规	60pt	70pt	4pt
一级标题	常规	58pt	68pt	5pt
二级标题	常规	56pt	66pt	5pt
三级标题	常规	55pt	65pt	6pt
内容提要	半粗体	53pt	62pt	6pt
正文	常规	53pt	62pt	6pt
插图编号	常规	51pt	60pt	7pt
副标题	常规	49pt	58pt	8pt

类型	字重	磅值	行距	字距
脚注	常规	44pt	52pt	9pt
一级说明文字	常规	43pt	51pt	10pt
二级说明文字	常规	40pt	48pt	10pt

不是所有 App 都用 1/1000em 来表示字间距。磅值大小基于 @2x 的图像分辨率为144 ppi, @3x的图像分辨率为216 ppi。

字体用法和字距

在界面原型中使用正确的字体和字形变体：使用 SF Pro 文本来显示文本19磅或更小的文本，以及在文本20磅或更大的文本中显示文本。当你在标准控件中使用 San Francisco 的文本，比如按钮和标注，iOS 会自动根据磅值和用户的可访问性设置应用最合适的变体，并适当地调整字距。

在 iOS 10 中 San Francisco 字体的变体是 SF UI Text 和 SF UI Display。

SF Pro Text

144ppi下两倍率 字体磅值	字距
6pt	41pt
8pt	26pt
9pt	19pt
10pt	12pt
11pt	6pt
12pt	0pt
13pt	-6pt
14pt	-11pt
15pt	-16pt
16pt	-20pt

144ppi下两倍率 字体磅值	字距
17pt	-24pt
18pt	-25pt
19pt	-26pt

SF Pro Display

144ppi下两倍率 字体磅值	字距
20pt	19pt
21pt	17pt
22pt	16pt
24pt	15pt
25pt	14pt
27pt	13pt
30pt	12pt
33pt	11pt
40pt	10pt
44pt	9pt
48pt	8pt
50pt	7pt
53pt	6pt
56pt	5pt
60pt	4pt
65pt	3pt
69pt	2pt
79pt + up	0pt

不是所有 App 都用 1/1000em 来表示字间距。磅值大小基于 @2x 的图像分辨率为 144ppi, @3x的图像分辨率为 216ppi。

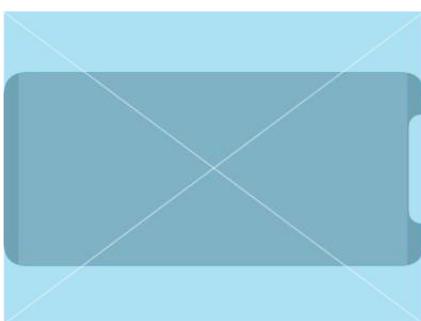
5.9 视频

系统提供的视频播放器拥有两种显示模式：全屏显示（宽高全填充）和适应屏幕显示（宽/高适应）。默认的，系统会根据视频的宽高比来选择显示模式，而用户能够在播放期间任意在两种模式之间切换。开发者指南，请参阅 [AVPlayerViewController](#)。

全屏显示模式：视频缩放来填充屏幕，这可能会对边缘进行裁切。这是宽视频（2:1 到 2.4:1）的默认显示模式。开发者请参阅 [resizeAspectFill](#)。

适应屏幕显示模式：能够在屏幕中看到完整的视频，这会出现信箱黑边或者邮筒黑边。这是标准视频（4:3、16:9 以及任和达到 2:1）和超宽视频（任何超过 2.4:1）的默认使用模式。开发者指南，请参阅 [resizeAspect](#)。

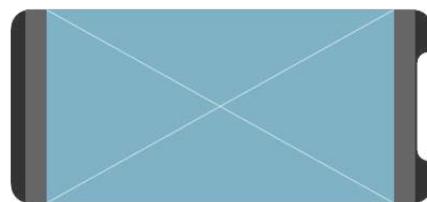
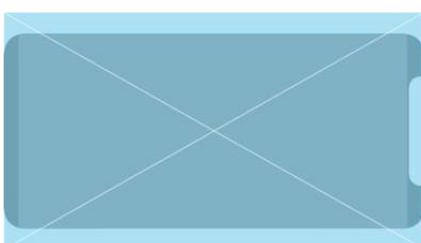
以iPhone Xs 的显示模式为例



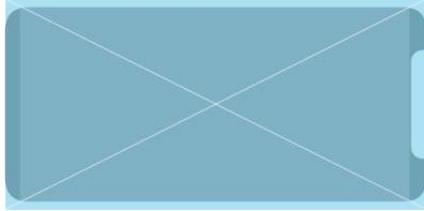
4: 3 视频中全屏视图模式下



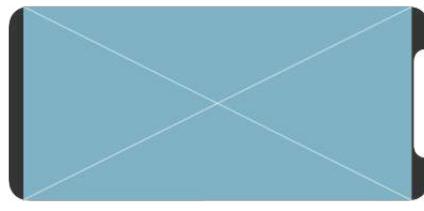
4: 3 视频中默认自适应屏幕模式下



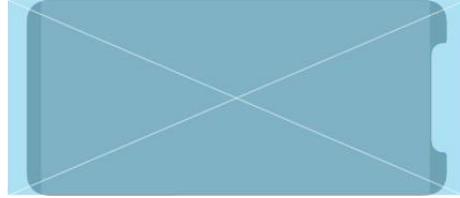
16: 9 视频中全屏视图模式下



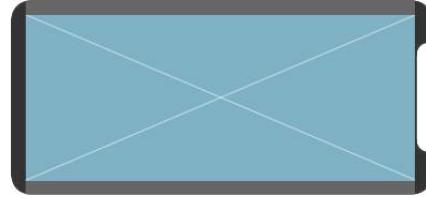
16: 9 视频中默认自适应屏幕模式下



2: 1 视频中全屏视图模式下



2: 1 视频中默认自适应屏幕模式下



21: 9 视频中全屏视图模式下



21: 9 视频中默认自适应屏幕模式下

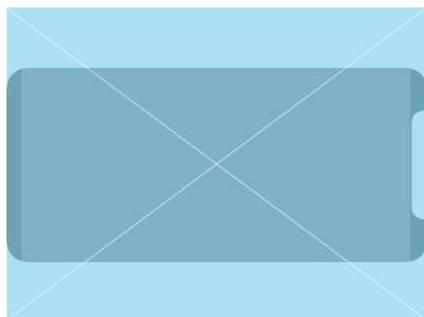
AVKit safe area

Video

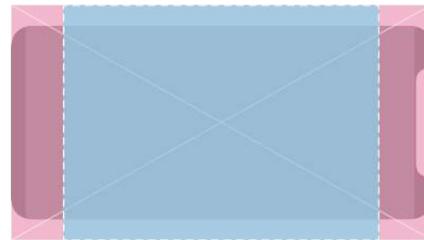
确保自定义视频播放器是用户所期待的：管我们的目标是在全面屏设备上播放视频时默认填充显示，但是如果填充显示会导致裁剪过多，那么应缩放视频适应屏幕。您还应该允许人们根据个人喜好在两者之间切换。开发者请参阅 [AVPlayerLayer](#)。

以iPhone Xs 的嵌入衬边为例

4: 3 视频嵌入衬边的结果

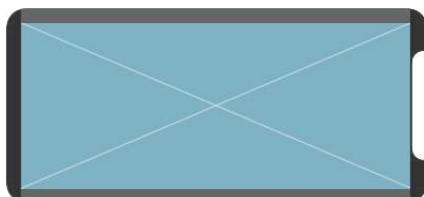


4:3视频在全屏视图模式下

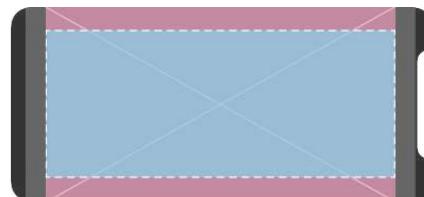


全屏显示模式下嵌入衬边

21: 9 视频嵌入衬边的结果



适应屏幕显示模式



适应屏幕显示模式下嵌入衬边

● AVKit safe area● Video● Embedded padding

始终以原始宽高比播放视频：当视频内容本身为了达到某一个宽高比就已经使用了嵌入式信箱衬边或邮筒衬边时，iOS 无法根据用户选择的显示模式正确缩放视频。嵌入在帧里的衬边使视频在全屏模式和适应屏幕模式下变得更小。此外这还会让视频在边缘或非全屏播放的情况下出现显示错误，比如在 iPad 的[画中画](#)模式。

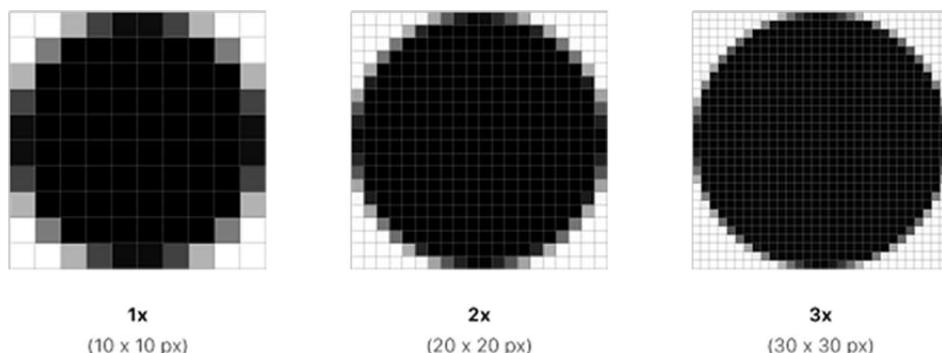
第六大章

图标和图像

Icons and Images

—

6.1 图像尺寸和分辨率



iOS 用于在屏幕上放置内容的坐标系统是以点 (pt) 为基础的。该坐标系以点为基本测量单位，这些点映射到显示器中则以像素显示。在标准分辨率屏幕上，一个点等于一个像素 ($1\text{pt} = 1\text{px}$)。因为高分辨率屏幕的像素密度更高，所以在真实世界里同等面积的屏幕中就包含更多的像素，即一点中包含更多像素 ($1\text{pt} = 2\text{px}$ 、 $1\text{pt}=3\text{px}$)。因此，高分辨率屏幕需要具备更高像素的图像。

为了支持所有的 iOS 设备，你需要为设计的内容提供高分辨率的图像：基

于不同的设备，将每个图像中的像素数量乘以特定比例系数来进行适配。标准分辨率图像的比例系数为 1.0，这种图像被称为 @1x 图像。高分辨率图像的比例系数为 2.0 或 3.0，被称为 @2x 或 @3x 图像。假设你有一个标准的分辨率 @1x 图像，例如 100px × 100px，那么，该图像的@ 2x 版本将是 200px × 200px，@3x 版本将是 300px × 300px。

设备	比例系数
iPhone X, iPhone 8 Plus, iPhone 7 Plus, and iPhone 6s Plus	@3x
所有其他高分辨率的 iOS 设备	@2x

设计高分辨率设计图

使用 8px × 8px 的网格系统：网格系统可以让线条和图像内容在所有尺寸上保持清晰，无需太多的修饰和锐化。将图形边界对齐到网格上，以减少按比例缩小图像时出现的半像素和内容模糊的情况。

以适当的形式进行创作：通常，位图/光栅的切图使用去交错的 PNG 格式文件（PNG 导出的设置）。选择 PNG 格式，是因为它支持透明度，并且因为是无损格式，不会在压缩过程中丢失图形的细节和色彩。对于需要阴影、纹理和高光效果的复杂切图来说，PNG 格式是一个很好的选择。而照片内容（如广告图、照片）请使用 JPEG 格式，因为它的压缩算法可以比无损格式节省更多的存储空间，且较难识别到压缩的痕迹。所以，写实的 App 图标最适合使用 PNG 格式。其它需要进行高分辨率缩放的扁平、矢量切图则使用 PDF 格式。

对于不需要使用 PNG-24 的切图，只需要使用 PNG-8 格式即可：因为 8

位色彩可以减少文件大小，并且不会降低图像质量。注意：PNG-8 不适用于照片类型。

优化 JPEG 文件，以找到存储容量和质量之间的平衡：大多数 JPEG 文件可以被压缩，而不会造成明显的图像失真，即使是少量的压缩也可以为磁盘节省大量空间。在每个 JPEG 图像上进行压缩设置，找出可接受失真范围内的最佳参数。

提供图像和图标的替代文本标签：替代文本标签在屏幕上看不到，但是它们帮助“VoiceOver”（APPLE 公司推出的一种语音辅助程序）描述屏幕上的内容，便于视觉障碍的人“看到”屏幕上的内容。

6.2 应用图标

每个 App 都需要一个美观的，令人印象深刻的图标，能在苹果商店和主屏幕上夺人眼球。用户轻撇图标的瞬间，是你的第一个机会，用来传达你的 App 的用途。图标也贯穿于整个系统中，例如在设置和搜索结果中。



拥抱简约：寻找个单一的元素能够表现你的 App 的精髓，然后通过一个简单并且独特的形状表达该元素。谨慎地添加细节部分。如果图标的内容或形状过于复杂，则细节难以辨别，特别是在较小的尺寸中。

提供一个单独的焦点：为图标设计一个单独的，集中的焦点，这种图标能立

即捕获用户的注意力并明确的代表你的 App。

设计一个可识别的图标：用户不应该只有通过分析图标才能弄清楚它代表什么。例如，邮件 App 的图标使用了一个信封，因为它普遍与邮件联系在一起。花时间来设计一个美观迷人且精炼的抽象图标，从而艺术化地传达你的 App 的目的。

保持背景简单，避免透明度：确保你的图标是不透明的，不要杂乱的背景。

使用一个简单的背景，这样它就不会过度影响周围的其它图标。你没有必要将整个图标填满内容。

只有当 logo 全部或部分由文字组成时，才在图标使用文字：在主屏幕时，一个 App 的名称会在图标之下显示。不要包含没有意义的文字重复说明名称或是告诉用户该如何使用你的 App，比如“Watch”或“Play”。如果你的设计包含了一些文字，那么请强调文字与你的 App 提供的实际内容相关。

图标中不要包括照片，屏幕截图或界面元素：摄影细节在小尺寸上很难看出。屏幕截图对于 App 图标来说太复杂了，通常不利于传达 App 的目的。图标中的界面元素会令人误解和困惑。

不要使用苹果硬件产品的副本：苹果产品受版权保护，无法在你的图标或图像中被二次创作。一般来说，避免复用设备的图形，因为硬件设计往往会频繁更新迭代，而且会使你的图标看起来过时。

不要在界面到处放置 App 图标：在 App 里发现一个图标被用于多种目的会让人困惑。反之，考虑使用图标的色彩方案。请参阅 [Color](#)。

在不同的壁纸环境下测试你的图标：你不能预期用户会为他们的主屏幕选择

什么样的壁纸，所以不要只是在一种深色和一种浅色的背景上测试你的图标。而是观察它在不同的背景上如何表现。在有动态背景的真实设备上试用它，因为背景会随着设备移动而改变视角。

保持图标的四角是方形的：系统会自动覆盖一个遮罩层让图标变成圆角。

App 图标属性

所有App图标应符合以下规格标准。

属性	值
格式	PNG
颜色空间	sRGB 或者 P3 (参阅颜色管理)
图层	扁平无透明度
分辨率	不同的分辨率。参阅图像大小和分辨率
形状	正方形且无圆角

App 图标大小

每个 App 必须提供一大一小两个图标，小图标会出现在主屏幕，并且当你的 App 被安装后会被系统使用，大图标会被用在苹果商店中。



设备或环境	图标大小
iPhone	180px × 180px (60pt × 60pt @3x)
	120px × 120px (60pt × 60pt @2x)
iPad Pro	167px × 167px (83.5pt × 83.5pt @2x)
iPad, iPad mini	152px × 152px (76pt × 76pt @2x)
App Store	1024px × 1024px (1024pt × 1024pt @1x)

为不同的设备提供不同大小的图标：确保你的 App 图标在你支持的所有设备上，看起来都很棒。

使用 App Store 图标模拟您的小图标：虽然 App Store 图标的使用方式与小型图标不同，但它仍然是你的 App 图标。大图标一般都和小图标外观相匹配，但是可以稍微丰富一些，更有细节，因为不会有视觉效果叠加在它上面。

Spotlight、设置和通知图标

每个 App 都应提供一个小图标，在 Spotlight (快速索引的功能) 搜索，如果关键词与 App 名称相符，iOS 会展示该图标。同时，需要设置的 App 同

样应该提供一个小图标用于在系统内置的设置 App 中展示。两个图标都应
该清晰标识你的 App——理想情况下，它们应该与 App 图标相符。如果你
不能提供这些图标，iOS 可能会缩小你的主图标，以便在这些场景中显示。



设备	Spotlight图标大小
iPhone	120px×120px (40pt×40pt @ 3x)
	80px×80px (40pt×40pt @ 2x)
iPad Pro, iPad, iPad mini	80px×80px (40pt×40pt @ 2x)

设备	设置图标大小
iPhone	87px×87px (29pt×29pt @ 3x)
	58px×58px (29pt×29pt @ 2x)
iPad Pro, iPad, iPad mini	58px×58px (29pt×29pt @ 2x)

设备	通知图标大小
iPhone	60px×60px (20pt×20pt @ 3x)
	40px×40px (20pt×20pt @ 2x)
iPad Pro, iPad, iPad mini	40px×40px (20pt×20pt @ 2x)

不要在“设置”图标上添加叠加层或边框：iOS 会自动为所有图标添加1像
素描边，以确保它们很好地呈现在白色的背景上。

提示：如果你的 App 能创建自定义文档，则不需要设计文档图标，因为 iOS 会使用你的 App 图标自动创建文档图标。

用户可选的 App 图标

对于某些 App，定制功能可以产生用户黏性并增强用户体验：如果定制功能可以为你的 App 提供价值，你可以在App 中嵌入的一组预设的图标，用户可以从中选择一个备用 App 图标。例如，运动 App 可能会为不同的球队提供图标，或者具有明暗模式的 App 可能会提供相应的明暗图标。请注意，你的 App 图标只能根据用户的要求进行更改，系统始终向用户提供此类更改的确认信息。

为所有需要的尺寸，提供视觉一致的替代图标：就像你的 App 主图标一样，每个备用图标均要提供不同尺寸的图像集合。当用户选择备用图标时，相应尺寸的图标将替代原先主图标，Spotlight 和系统其他位置的主图标。比如说，为了确保系统中的备用图标始终保持一致，用户不应该在主屏幕上看到图标的一个版本，而在“设置”中则看到完全不同的版本。请按照 App 主图标的尺寸大小，为备用 App 图标提供相同的尺寸（App Store图标除外）。请参阅 [App Icon Sizes](#)。

开发者指南，请参阅 [UIApplication](#) 中的 `setAlternateIconName` 方法。

注意：替代应用图标需要通过 [App Review](#) 进行审核，并且必须遵守 [App Store评估指南](#)。

6.3 自定义图标 (iOS12 及更早的版本)

在 iOS13 或更高版本中，尽量使用 [SF Symbols](#) 来表示应用中的任务和模式。如果您的应用程序在 iOS12 或更早的版本上运行，或者您需要创建自定义位图图标，请按照以下指导进行操作。

创作简单、辨识度高的设计：太多的细节会让图标看起来难以理解且不具可读性。努力实现一种简单的通用设计，大多数人都会快速识别，不会反感。最好的图标设计，使用与 App 内容和作用直接相关的、广为人们知悉的视觉比喻。



设计象形符号的图标：象形符号，也称为模板图像，是具有透明度，抗锯齿功能的单色图像，并且图像边缘没有投影。符号外观根据 App 内容和用户交互来设计，包括着色，亮度和活力。各种标准界面元素支持符号，包括导航栏，标签栏，工具栏和主屏幕快捷操作。

准备比例系数为 @2x 的符号切图，并保存为 PDF 格式：因为 PDF 是一种支持高分辨率缩放的矢量格式，所以通常而言，在你的 App 中提供一个@2x 版本就足够了，并且这个版本支持扩展其他分辨率。

保持图标之间一致连贯：无论你只使用自定义图标或是混合使用自定义图标和系统图标，在 App 中的所有图标都应该在大小、细节程度、透视和描边粗细上保持一致。



确保图标清晰可辨：一般来说，纯色图标往往比轮廓图标更清晰。如果图标

必须包含线条，请与其他图标和 App 的版式协调好比重。



使用颜色来表示选中和取消的状态：避免在两个不同的图标设计之间切换，如纯色版本和轮廓版本。

避免在图标中加入文字：如果你需要展示文本，请在标签下面直接加上标题，并且适当调节位置。

不要复用 Apple 硬件产品的图形：Apple 产品受版权保护，不能在你的图标或是图片中被二次创作。一般来说，避免复用设备的图形，因为硬件设计频繁地更新换代，这会导致你的图标看起来过时。

提供图标的替代文本标签：替代文本标签在屏幕上看不到，但是它们帮助 VoiceOver (APPLE公司推出的一种语音辅助程序) 描述屏幕上的内容，便于视觉障碍的人“看到”屏幕上的内容。

自定义图标尺寸

首先，你的 App 的图标系列应该在视觉上保持大小一致。如果个别图标设计的比重不同，则某些图标可能需要略大于其他图标才能保持视觉一致。



导航栏和工具栏图标尺寸

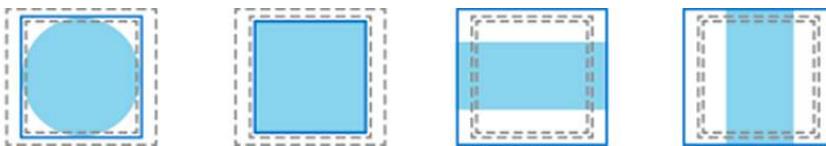
设计自定义导航栏和工具栏图标时，请参考以下尺寸指导，如果为了保持平

衡，可以进行适当调整。

目标尺寸	最大尺寸
75px × 75px (25pt × 25pt @3x)	83px × 83px (27.67pt × 27.67pt @3x)
50px × 50px (25pt × 25pt @2x)	56px × 56px (28pt × 28pt @2x)

标签栏图标尺寸

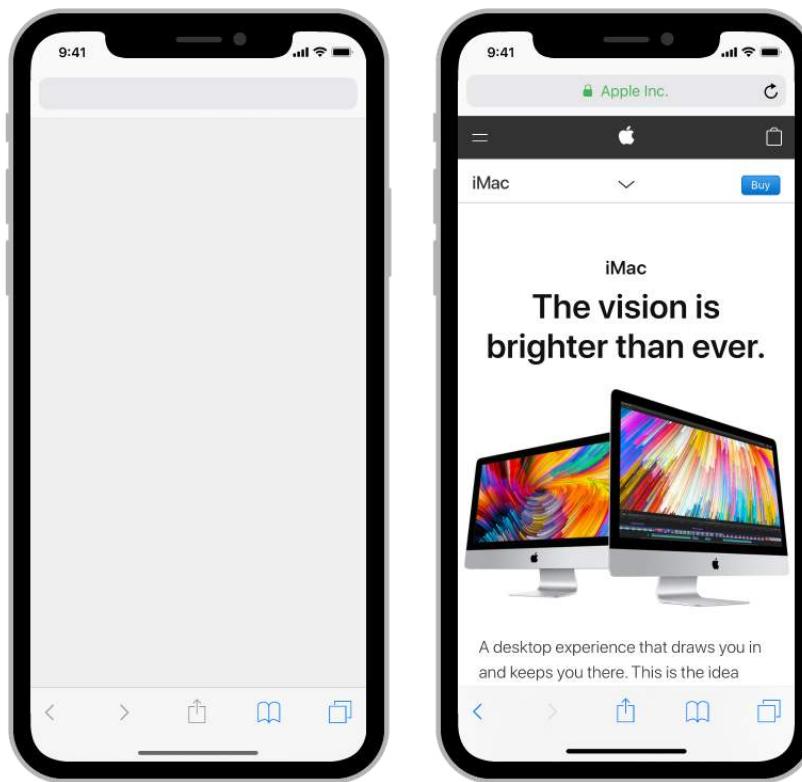
在纵向上，底部栏图标显示在其标题上方。在横向上，图标和标题并排出现。根据设备和方向，系统会显示常规或紧凑的底部栏。你的 App 应该包含两种尺寸的自定义底部栏图标。



属性	常规标签栏	紧凑型标签栏
目标宽度和高度（圆形符号）	75px × 75px (25pt × 25pt @3x)	54px × 54px (18pt × 18pt @3x)
	50px × 50px (25pt × 25pt @2x)	36px × 36px (18pt × 18pt @2x)
目标宽度和高度（正方符号）	69px × 69px (23pt × 23pt @3x)	51px × 51px (17pt × 17pt @3x)
	46px × 46px (23pt × 23pt @2x)	34px × 34px (17pt × 17pt @2x)
目标宽度（宽符号）	93px (31pt @3x)	69px (23pt @3x)
	62px (31pt @2x)	46px (23pt @2x)
目标高度（高符号）	84px (28pt @3x)	60px (20pt @3x)
	56px (28pt @2x)	40px (20pt @2x)

6.4 启动画面

启动画面出现在 App 刚启动的时候。随后，启动画面会很快被 App 的首屏取代，让人感觉你的 App 是快速响应的。启动画面不是一个炫技的时机，它只是为了增强用户体验，让用户觉得你的 App 能够快速启动并且立即被使用。每个 App 都应该提供一个启动画面。



因为设备屏幕大小不同，启动画面的大小也有所差异。为了适应这个需要，你可以通过 Xcode 故事板或是一组静态图片的形式，为你的 App 所支持的设备提供启动画面。因为 Xcode 故事板灵活性高且易于适配，所以推荐采用该形式。你可以使用一个单独的故事板来管理你的所有启动画面。了解关于可适配界面的开发细节，请参阅 [Auto Layout Guide](#)。

设计一个与你的 App 首屏几乎相似的启动画面：如果你的启动画面包含了

与首屏看起来不同的元素，那么在启动画面过渡至 App 首屏时，用户会经历一次不愉快的跳转体验。

避免在启动画面中包含文本：因为启动画面是静态的，任何展示的文本都不能被点击。

淡化启动：用户通常会在不同的 App 之间频繁切换，所以请设计一个启动画面，能够让 App 的启动不易被人察觉。

不要打广告：启动画面不是一个宣传品牌的时机。不要设计一个类似开屏广告或是介绍窗口的登录体验。不要在启动画面包涵 logo 或是其它品牌元素，除非它们是你的 App 首屏的静态元素。

静态启动画面图片

最好使用 Xcode 故事板制作启动画面，但必要时你也可以提供一组静态图片。根据设备创建不同尺寸的静态图片，并确保包涵状态栏区域。

设备	竖屏尺寸	横屏尺寸
12.9" iPad Pro	2048px × 2732px	2732px × 2048px
10.5" iPad Pro	1668px × 2224px	2224px × 1668px
9.7" iPad	1536px × 2048px	2048px × 1536px
7.9" iPad mini 4	1536px × 2048px	2048px × 1536px
iPhone X	1125px × 2436px	2436px × 1125px
iPhone 8 Plus	1242px × 2208px	2208px × 1242px
iPhone 8	750px × 1334px	1334px × 750px
iPhone 7 Plus	1242px × 2208px	2208px × 1242px
iPhone 7	750px × 1334px	1334px × 750px
iPhone 6s Plus	1242px × 2208px	2208px × 1242px

设备	竖屏尺寸	横屏尺寸
iPhone 6s	750px × 1334px	1334px × 750px
iPhone SE	640px × 1136px	1136px × 640px

6.5 系统图标

该系统提供内置的图标，代表各种用例中常见任务和内容类型。

- [导航栏和工具栏图标](#)
- [标签栏图标](#)
- [主屏幕快速操作图标](#)

尽可能使用这些内置的图标，因为用户对这些图标是熟悉的。

按照预期使用系统图标： 每个系统提供的图像具有特定的、众所周知的意义。为了避免混淆用户，必须根据其含义和推荐用法使用每张图像。

提供图标的替代文字标签： 替代文字标签在屏幕上看不到，但是他们让 VoiceOver 可以描述屏幕上的内容，使视觉障碍的人更方便导航。

如果找不到符合需求的系统提供的设计，请设计自定义图标： 设计自己的图标比错误使用系统提供的图像更好。查看 [Custom Icons](#)。

导航栏和工具栏图标

在 [导航栏](#) 和 [工具栏](#) 中使用以下图标。

了解更多开发者指南，请参阅 [UIBarButtonItemSystemItem](#)。

提示：你可以使用文本而不是图标来表示导航栏或工具栏中的项目。 例如，日历在工具栏中使用“今天”，“日历”和“收件箱”。 您还可以使用固定的空格元素来提供导航和工具栏图标之间的填充。

图标	名称	含义	API
	行动	显示包含在当前上下文中有用的共享扩展，操作扩展和任务（如“复制”，“收藏夹”或“查找”）的模态视图。	action
	添加	创建一个新项目。	add
	书签	显示特定应用的书签。	bookmarks
	相机	拍摄照片或视频，或者显示照片库。	camera
Cancel	取消	关闭当前视图或结束编辑模式而不保存更改。	cancel
	撰写	在编辑模式下打开新视图。	compose
Done	完成	完成保存状态并关闭当前视图，或退出编辑模式。	done
Edit	编辑	在当前页面进入编辑模式。	edit
	快进	媒体播放或幻灯片快进。	fastForward

图标	名称	含义	API
	整理	将项目移动到新的目的地，如文件夹。	organize
	暂停	暂停媒体播放或幻灯片。暂停时始终存储当前位置，以便播放在以后恢复。	pause
	播放	开始或恢复媒体播放或幻灯片。	play
Redo	重做	重做上次撤消的操作。	redo
	刷新内容	刷新内容。请谨慎使用此图标，因为你的应用程式会自动刷新内容。	refresh
	回复	将项目发送或路由到另一个人或位置。	reply
	后退	通过媒体播放或幻灯片向后移动。	rewind
Save	保存	保存当前状态。	save
	搜索	显示搜索字段。	search
	停止	停止媒体播放或幻灯片。	stop
	垃圾桶	删除当前或所选项目。	trash
Undo	撤销	撤消上一个操作。	undo

标签栏图标

在标签栏中使用以下图标 `tab bars`。有关开发者的指南，请参阅 [UITabBarSystemItem](#)。

图标	名称	含义	API
	书签	显示App专用书签。	bookmarks
	通讯录	显示用户的联系人。	contacts
	下载	显示活跃或最近的下载。	downloads
	收藏	显示用户最喜欢的项目。	favorites
	精选	展示应用程序特色的 内容。	featured
	历史	显示最近的操作或活 动。	history
	更多	显示其他标签栏项 目。	more
	最新	最近在特定时段内访 问的内容或项目。	mostRecent
	浏览最多	最受欢迎的项目。	mostViewed
	搜索	进入搜索模式。	search
	最高评分	显示最高评分的项 目。	topRated

主屏幕快速操作图标

在主屏幕快速操作菜单中 [home screen quick action](#) 使用以下图标。有关开发者指南，请参阅 [UIApplicationShortcutIconType](#)。

图标	名称	含义	API
	添加	创建一个新项目。	add
	闹钟	设置或播放一个闹钟。	alarm
	音频	表示或调节音频。	audio
	书签	创建书签或显示书签。	bookmark
	拍照	拍照片。	capturePhoto
	摄像	拍视频。	captureVideo
	云	表示，显示或打开一个云的服务。	cloud
	撰写	撰写新的可编辑内容。	compose
	确认	表示一个操作完成。	confirmation
	通讯录	选择或显示一条通讯录。	contact
	日期	显示日历或事件，或执行一个相关操作。	date
	收藏	表示或标记一个收藏的项目。	favorite

图标	名称	含义	API
	家	指示或显示主屏幕。 指示, 显示或按路线前往家庭住址	home
	邀请	表示或显示邀请。	invitation
	位置	表示位置的概念或访问当前的地理位置。	location
	喜爱	表示或标记一个喜爱的项目。	love
	邮件	创建邮件消息。	mail
	标记位置	表示, 显示或保存地理位置。	markLocation
	信息	新建一条信息或者表示使用信息工具。	message
	暂停	暂停播放媒体, 总是存储当前进度, 这样就可以继续播放。	pause
	播放	开始或继续播放媒体。	play
	禁止	表示某对象被禁止。	prohibit
	搜索	进入搜索模式。	search
	分享	与他人分享内容或者分享至社交媒体。	share
	随机播放	表示或打开随机播放模式。	shuffle

图标	名称	含义	API
	任务	表示一个未完成任务或者是完成任务时标记。	<code>task</code>
	任务完成	表示一个完成任务或者是一个未完成任务标记。	<code>taskCompleted</code>
	时间	表示或显示一个时钟或计时器。	<code>time</code>
	更新	更新项目。	<code>update</code>

第七大章

UI栏

Bars

7.1 导航栏

导航栏出现在屏幕顶部的状态栏下方，并可以通过一系列分层屏幕进行导航。当显示新屏幕时，通常标有前一屏幕的后退按钮出现在栏的左侧。有时，导航栏的右侧包含一个控件，如编辑或完成按钮，用于管理活动视图中的内容。在拆分视图中，导航栏可能会出现在拆分视图的单个窗格中。导航栏是半透明的，可能具有背景色调，并且可以在屏幕上键入屏幕时隐藏，发生手势或视图调整大小。



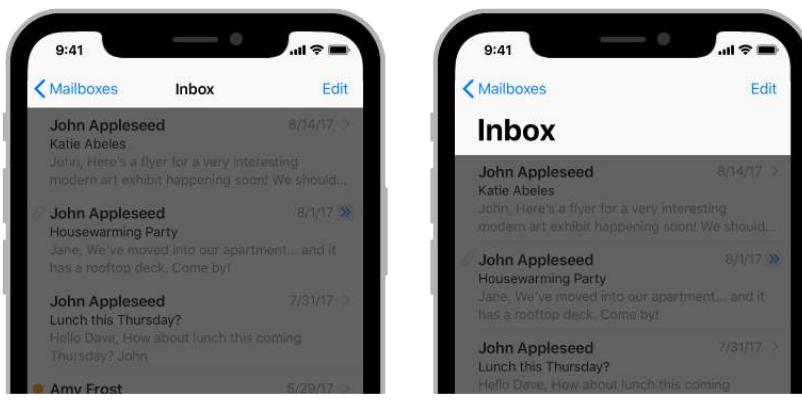
考虑在显示全屏内容时暂时隐藏导航栏：当您想关注内容时，导航栏可能会分散注意力。暂时隐藏栏目，提供更身临其境的体验。查看全屏照片时，照片会隐藏导航栏和其他界面元素。如果App要隐藏导航栏，让用户使用简单的手势（如轻按）来还原它。

开发者指南，请参阅 [UINavigationBar](#)。

建议：当不需要导航或使用多个控件来管理内容时，请使用工具栏。请参阅 [Toolbars](#)。

导航栏标题

考虑在导航栏中显示当前视图的标题：在大多数情况下，标题可帮助人们了解他们正在查看的内容。但是，如果导航栏标题是多余的，可以将标题留空。例如，Notes 不会为当前注释标题，因为第一行内容提供了所需的所有上下文。



当需要特别强调上下文时，请使用较大的标题：在一些 App 中，放大加粗的大标题可以帮助人们浏览和搜索。例如，在标签中，大标题可以分辨活动

选项，并在用户滚动到顶部时通知用户。音乐使用大标题来区分专辑，艺术家，播放列表和收音机等内容区域。当用户开始滚动内容时，大标题转换为标准标题。大标题在所有 App 中都没有实际意义，不应与内容竞争。虽然 Clock 应用程序具有标签式布局，但是不需要大标题，因为每个标签都有一个不同的，可识别的区域。

开发者指南，请参阅 [prefersLargeTitles](#)。

导航栏控件

避免使导航栏的控件太多：通常，导航栏不应包含页面的当前标题、后退按钮和管理视图的控件。如果您在导航栏中使用分段控件，则该栏不应包含标题或除分段控件之外的任何控件。

使用标准后退按钮：人们知道标准的后退按钮可以让他们通过信息层次来回溯步骤。但是，如果使用自定义后退按钮，请确保它仍然像后退按钮，反应敏捷，与界面的其余部分相匹配，并始终贯穿你的 App。如果用自定义图像替换系统提供的返回按钮，也可以提供自定义遮罩图像。iOS 在使用此遮罩时，可以在转换期间为按钮标题设置动画。

不要使用面包屑式的多段路径：后退按钮总是执行单个操作 - 返回到上一个屏幕。如果你认为在没有当前屏幕的完整路径的情况下，人们可能会迷失方向，请考虑对 App 的层次结构进行梳理。

给文本按钮足够的空间：如果您的导航栏包含多个文本按钮，那些按钮的文本可能会一起出现，使按钮无法分隔。通过在按钮之间插入固定空间项来添加分隔。开发者指南，请参阅 [UIBarButtonItem](#) 中

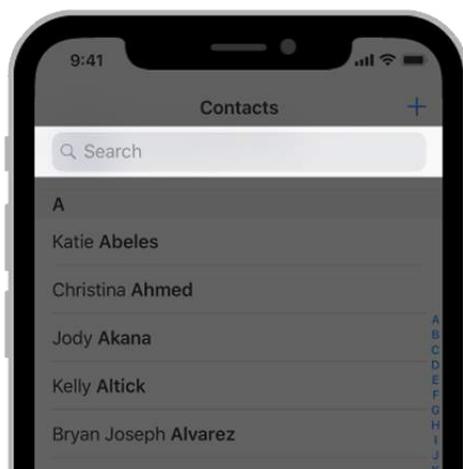
的 `UIBarButtonSystemItemFixedSpace` 常量值。

考虑在导航栏中使用分段控件来简化应用程序的信息层次结构：如果您在导航栏中使用分段控件，请仅在层次结构的顶层执行此操作，并确保在较低级别选择准确的后退按钮。有关其他指导，请参阅 [Segmented Controls](#)。



7.2 搜索栏

人们使用搜索栏来搜索大量的数据。有两种样式的搜索栏 - 突出（默认）和最小。联系人采用突出的风格，其中包括明显的有色背景。照片使用最小的风格，这有利于与周围的界面更协调。搜索栏默认是半透明的，但可以使其不透明。适用时，可以将搜索栏配置为自动隐藏导航栏。



搜索栏包含单个搜索字段，可以包含占位符文本和清除按钮：除了搜索字段

之外，搜索栏还可以包含用于退出搜索的“取消”按钮。

让用户通过搜索栏而不是文本框去搜索：文本框不具备搜索栏所拥有的外观

特征。

根据搜索的重要性，选择搜索栏样式：如果搜索是应用程序中的主要功能，

请使用默认的突出显示样式的搜索栏。如果搜索不频繁，请使用最小样式

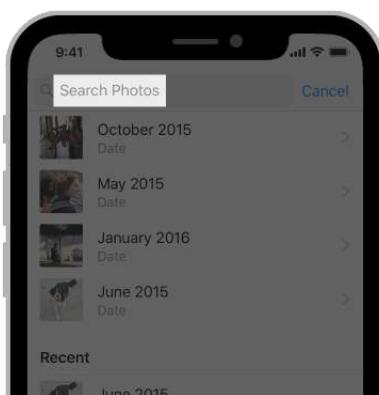
如有必要，请在搜索栏中提供提示和搜索范例：搜索字段可以包含占位符文

本，例如“搜索服装，鞋子和配件”或简称“搜索”，提醒您正在搜索。

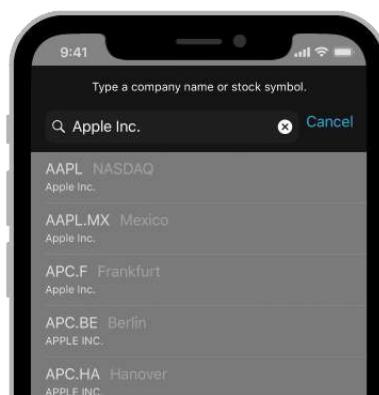
也可以在搜索栏的正上方显示一个简洁的，含有适当标点符号的一行句子，

以提供指导。例如，股票在搜索字段上方使用文字，让人们知道他们可以

输入公司名称或股票代码。



Placeholder text



Introductory text

考虑在搜索栏下方提供有用的快捷方式和其他内容：使用搜索栏下的区域来

帮助人们更快地获得内容。例如，Safari可以在您点击搜索后立即显示您的

书签。选择一个可以直接进入，而不用输入任何搜索字词。股票在您输入

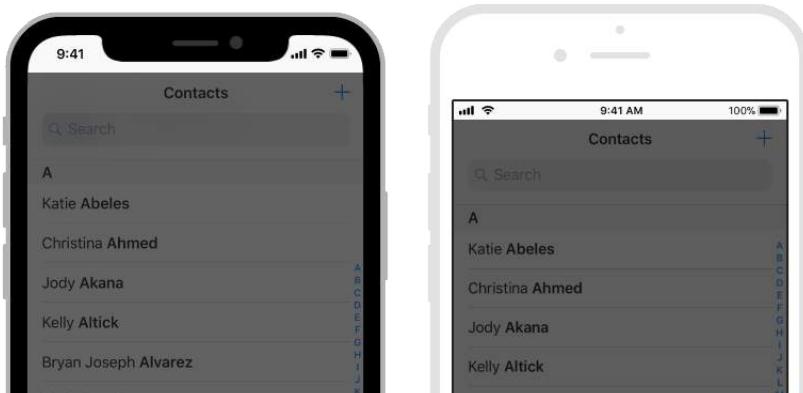
搜索字段时显示结果列表。随时点击一个，而不再输入任何字符。

开发者指南，请参阅 [UISearchController](#) 和 [UISearchBar](#)。

7.3 状态栏

状态栏出现在屏幕的上边缘，并显示有关设备当前状态的有用信息，如时间，蜂窝数据，网络状态和电池电量。状态栏中显示的实际信息因设备和系统配置而异。

使用系统提供的状态栏：人们期望状态栏在系统上保持一致。不要用自定义状态栏替换它。



Light status bar



Dark status bar

使用你的设计来协调状态栏样式：状态栏的文本和指示器的视觉样式是白或的，可以设置到 App 的所有页面，或是单独为不同的页面设置。黑色的状态栏在浅色内容之上运行良好，而白色状态栏可以在深色内容之上运行。

隐藏状态栏下的内容：默认情况下，状态栏的背景是透明的，允许下方的内容显示。保持状态栏的可读性，并不意味着它背后的内容是互动的。有几种常用的技巧：

- 在你的 App 中使用导航栏，它会自动显示状态栏背景，并确保内容不会显示在状态栏下。
- 在状态栏后面显示自定义图像，如渐变或纯色。
- 对状态栏后面的内容应用背景模糊效果（[UIBlurEffect](#)）。

考虑在全屏显示时暂时隐藏状态栏：当用户试图关注某个媒体时，状态栏可能会分散注意力。暂时隐藏这些元素，提供更身临其境的体验。例如，照片 App 在用户浏览全屏照片时隐藏状态栏和其他界面元素。



避免永久隐藏状态栏：没有状态栏，人们必须离开你的应用程序来检查时间，或者看看他们是否有 Wi-Fi 连接。让人们通过简单，可发现的手势重新显示隐藏的状态栏。在照片中浏览全屏照片时，只需点击一次即可显示状态栏。

使用状态栏来表示显示网络状态：当 App 使用网络时，特别是长时间的操作，显示网络状态栏，以便人们知道活动正在进行中。

请参阅 [Network Activity Indicators](#)。

开发者指南，请参阅 [UIApplication](#) 中的 [UIStatusBarStyle](#) 常量和 [UIViewController](#) 中的 [preferredStatusBarStyle](#)。

7.4 标签栏

屏幕底部会显示一个标签栏，并提供在应用的不同部分之间快速切换的功能。标签栏是半透明的，可能具有背景色调，在所有屏幕方向上保持相同的高度，并在显示键盘时隐藏。标签栏可以包含任意数量的选项，但可见选项的数量根据设备大小和方向而有所不同。如果某些选项由于水平空间有限而无法显示，那么最后一个可见选项将成为“更多”选项，显示列表中的其他选项卡。**一般来说，使用标签栏来梳理各 App 级别的信息：**标签栏是简化信息层次结构的一种很好的方法，可以一次访问多个对等信息类别或模式。



标签栏只用于导航，标签栏按钮不应用于执行操作：如果您需要提供对当前视图中元素的操作，请改用工具栏。请参阅 [Toolbars](#)。

避免太多的标签：每个附加标签都会减少可点击区域，并增加应用程序的复杂性，从而更难查找信息。虽然“更多”选项卡可以显示额外的选项卡，但这本身就需要额外的选项，而且空间不足。仅包含基本选项卡，并使用信息层次结构所需的最小选项数。太少的标签也可能是一个问题，因为它会使界面断开连接。一般来说，在 iPhone 上使用三到五个标签。iPad 上可以再多几个。

当其功能不可用时，不要删除或禁用选项卡：如果选项卡在某些情况下可用，但在其他情况下不可用，App 界面就会变得不稳定和不可预测。确保始终启用所有选项，并解释选项的内容不可用的原因。例如，如果 iOS 设备上没有歌曲，“音乐”应用中的“我的音乐”选项卡会解释如何下载歌曲。

始终在附件视图中切换上下文：为了保持界面可预测，选项卡应直接附加到选项栏中，而不影响屏幕上其他位置的视图。例如，选择分割视图左侧的选项卡不应导致分割视图的右侧突然更改。在 popover 中选择一个选项卡不应该导致弹出窗口后面的视图更改。

确保标签栏图标在视觉上保持一致和平衡：该系统为常见用例提供了一系列预定义的图标。请参阅 [System Icons > Tab Bar Icons](#)。您也可以创建自己的图标。查看 [Custom Icons](#)。

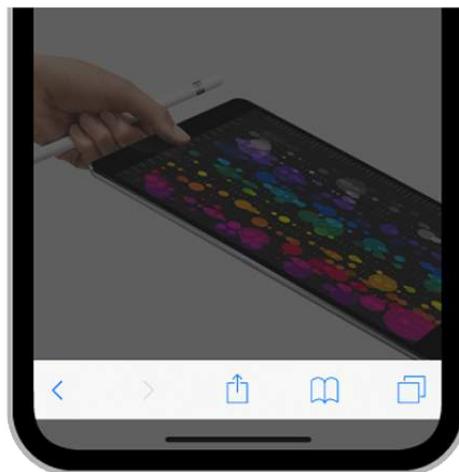
使用角标这种不引人注目的提醒方式：您可以在选项卡上显示徽章 - 包含白色文本和数字或感叹号的红色椭圆，以指示新信息与该视图或模式相关联。

开发者指南，请参阅 [UITabBar](#)。

提示：了解标签栏和工具栏之间的区别很重要，因为这两种类型都显示在 App 屏幕的底部。标签栏可让用户在应用程序的不同部分之间快速切换，例如 Clock 应用程序中的“闹钟”，“秒表”和“计时器”选项卡。工具栏包含用于执行与当前上下文相关的操作按钮，例如创建项目，删除项目，添加注释或拍摄照片。请参阅 [Toolbars](#)。标签栏和工具栏从不会出现在同一个视图中。

7.5 工具栏

工具栏出现在 App 屏幕底部，包含了执行当前视图或包含内容相关操作的按钮。工具栏是半透明的，也可能会有背景色，并且通常在用户不太需要它们时被隐藏。比如，在 Safari 中，当你滚动页面表明你再阅读时，工具栏就藏起来了。当你在屏幕底部点击时，工具栏又会再次出现。当前屏幕有键盘时，工具栏也会被隐藏。



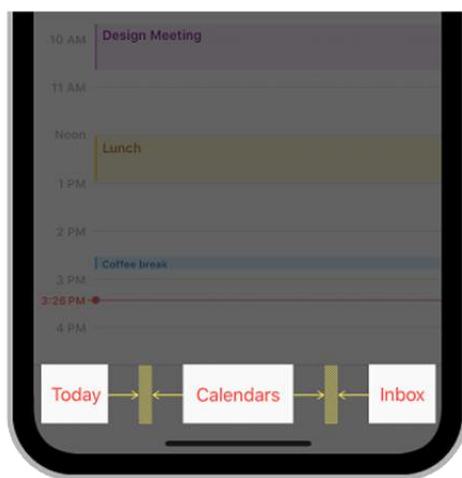
提供相关的工具栏按钮：工具栏必须包含在当前环境下有意义的常用操作命令。

考虑图标或文字按钮是否适合你的 App：当你需要多于三个工具栏按钮时，使用图标会比较合适。当你只有三个或是更少的按钮时，文字按钮有时候看起来更加清楚。比如，在日历 App 中，文本就被当作按钮使用，因为图标可能会令人迷惑。因为使用了文本，“Inbox”按钮还能显示所有的日程表和受邀活动数量。

避免在工具栏中使用分段控件：分段控件让用户切换内容，而工具栏更针对于当前屏幕。如果你需要提供切换内容的方法，请考虑使用标签栏。详情

见 [Tab Bars](#)。

给文本标题按钮足够的空间：如果你的工具栏包含多个按钮，那些按钮的文本可能会挤在一起，使按钮无法区分。通过在按钮之间插入固定间距使其分离。更多开发者指南，请参阅 [UIBarButtonItem](#) 中的 [UIBarButtonItemSystemItemFixedSpace](#) 的恒定值。



开发者指南，请参阅 [UIToolbar](#)。

提示：理解标签栏和工具栏之间的不同十分重要，因为这两种栏都是出现在 App 屏幕的底部。工具栏包含了执行当前视图相关操作的按钮，比如创建项、删除项、添加注释或是拍照。标签栏让用户在 App 的不同部分间快速切换，比如，时钟 App 中的“闹钟”、“秒表”、“计时器”。请参阅 [Tab Bars](#)。标签栏和工具栏决不会在同一个视图内同时出现。

八大章

内容视图

Views

8.1 动作菜单

动作菜单是一种特定的警告样式，它出现在对控件或操作的响应中，并呈现一组与当前窗口相关的两个或多个选择。用户启动任务或者执行潜在的破坏性操作之前弹出动作菜单请求确认。在较小的屏幕上，动作菜单从屏幕的底部向上滑出，在较大的屏幕上，动作菜单会以弹出窗口的形式呈现出来。



提供一个取消按钮可以增加操作的明确性：取消按钮能够让用户在放弃一项任务时对自己的操作更加肯定，并且动作菜单中的取消按钮应该总是位于屏幕的最底部。

突出破坏性的选择项：使用红色突出用来执行破坏性或危险动作的按钮，并在动作菜单的顶部显示这些按钮。

避免在动作菜单中出现滚动情况：如果一个动作菜单中存在太多选项，用户就必须滚动查看所有选项，而滚动需要更多的时间来做出选择，且滚动过程中也容易误触到其他按钮。

开发者指南，请参阅 [UIAlertController](#) 里的 [UIAlertControllerStyleActionSheet](#)。

8.2 活动视图

一个活动就是一个针对当前窗口中内容的服务操作，例如“复制”、“收藏”或“查找”，点击启动之后可以立即执行一个服务操作，也可以先请求更多的信息然后再执行该操作。活动在活动窗口中启动，活动窗口可以以表单或者弹出窗口的形式出现，这取决于设备的种类和屏幕的方向。使用活动可以让用户访问 App 中可执行的自定义服务或者任务。



系统内置了许多事件，包括“打印”，“信息”和“Airplay”。这些任务总是出现在事件窗口的顶部，而且位置固定，所以你不需要为这些默认操作创建新的自定义事件。事件窗口还显示其他 App 的共享和操作扩展程序，详情请参阅 [Sharing and Actions](#)。

为你的自定义事件设计一个简单的图像模板：图像模板通过蒙版的方式创建事件的图标。创建图形模板时你需做到：

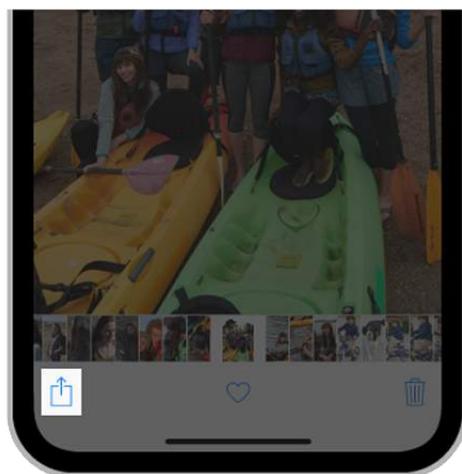
- 使用透明度适当的纯黑色或者纯白色。
- 使用抗锯齿处理。
- 不要包含阴影。
- 图像模板应该在一个 70×70 像素的区域内居中显示。

为你的服务创建一个简练的事件标题：标题显示在事件窗口中事件图标的下

方，通常来说简短的标题效果最好，如果标题过长，iOS 会先缩小字号，仍然太长的话就会截断它。通常来说，尽量避免将你的公司或产品名称包含在标题中。

确保活动是可以对当前窗口中的内容进行操作的：虽然事件窗口中系统内置事件的位置是固定的，但如果它们不适用于你的 App，则可以将其移除。例如，为了防止用户打印图像，你可以移除“**打印**”活动。另外，你还可以控制自定义服务出现的时间。

使用操作按钮来显示事件窗口：用户已经习惯了点击操作按钮来使用系统内置的服务，所以为了避免用户产生混淆不要提供其它操作方式。

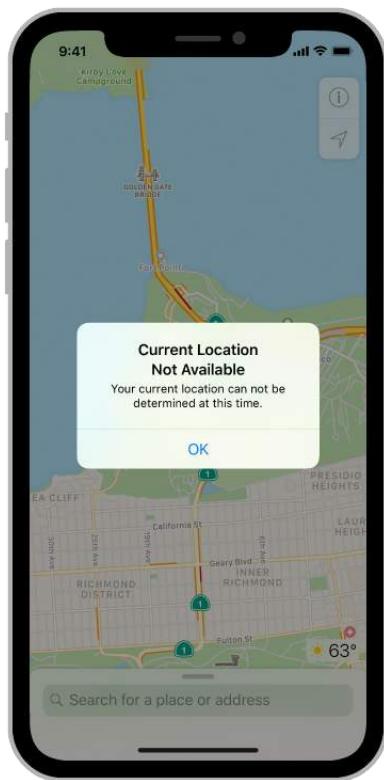


开发者指南，请参阅 [UIActivityViewController](#) 和 [UIActivity](#)。

8.3 警告框

警告框用来传达与你的 App 或设备状态相关的重要信息，并且通常需要得到用户的反馈。警告框的内容包括标题、描述消息（可选）、一个或多个按钮以及输入框（可选）。除了这些元素之外，警告框的外观样式是不可更改

的。



尽量避免使用警告框：警告框十分影响用户的操作体验，所以只应该在重要的场景下使用，比如确认购买和破坏性操作（例如“删除”），或者通知用户有关 App 和设备的问题（警告、预警之类的问题）。警告框的不常出现有助于提高用户的重视程度，请确保每个警告都提供关键的信息和有效的选择。

测试两个方向下警告框的显示情况：在横屏模式和竖屏模式下，警告框的显示可能会不同，需要优化警告框内的文本信息，使其不需要滚动就可以在任何方向上被轻松阅读。

开发者指南，请参阅 [UIAlertController](#)。

标题和描述消息

使用一些简短的描述性标题：让用户看到的文字越少越好，能够用标题表达清楚的就不要增加额外的描述信息。可以使用疑问句或简短的陈述句来让标题传递更准确的信息。尽量将标题控制在一行以内，如果标题是完整的句子，使用句子模式的大小写和适当的标点符号来完成标题。（译者注）如果标题是句子片段，请不要使用表示结束的标点符号。

如果一定要增加描述信息，请使用完整的短句：尽量将句子控制在一、两行以内，使用句子模式的大小写和适当的标点符号。

避免出现指责、判断、侮辱的语气：用户知道警告框弹出是出现了问题和危险的情况。只要语气友好，直截了当的传达消极的消息会比表意模糊的积极消息更好。避免出现“你”，“你的”，“我”，“我的”这类词语，因为有时候会被认为是具有生疏和趾高气昂的态度。

避免对警告按钮做出解释：如果警告框的文本和按钮标题内容传达的信息足够明确，就不需要解释按钮的作用。如果一定要对用户提供指引，请使用“tap”（点击）这个词，在引用按钮时保持大写，并且不要在引中包含按钮的标题。

警告按钮

通常使用包含两个按钮的警告弹窗：两个按钮的警告弹窗，可以使用户更容易地在两个选择中做出决定。一个按钮通常用于通知，它无法做出更多的选择。三个或以上的按钮会使选择变得复杂，并且需要滚动，会造成很不好的用户体验，这种情况下可以考虑使用动作菜单，请查阅 [Action Sheets](#)。

注：除此之外，标题文本的大小写都应该采用标题模式。句子模式是一个句子中的第一个字母大写；标题模式是每个单词的首字母都大

创建简洁的、逻辑清晰的按钮标题：按钮标题最好由一、两个描述操作结果的词语组成。与所有按钮标题一样，使用标题模式的大小写（英文环境下，每个单词的首字母大写），不要出现表示结束的标点符号。尽量使用与警告标题和描述信息直接相关的动词或动词短语，例如“查看全部”、“回复”或者“忽略”。使用“确认”进行明确的表达，避免使用“是”和“否”。

(注：在具有破坏性的选项中为了防止误操作，会刻意把顺序对调)

把按钮放在用户预期的地方：通常来说，右边的按钮是用户最有可能点击的按钮，取消按钮应该总是放在左边（译者注）。

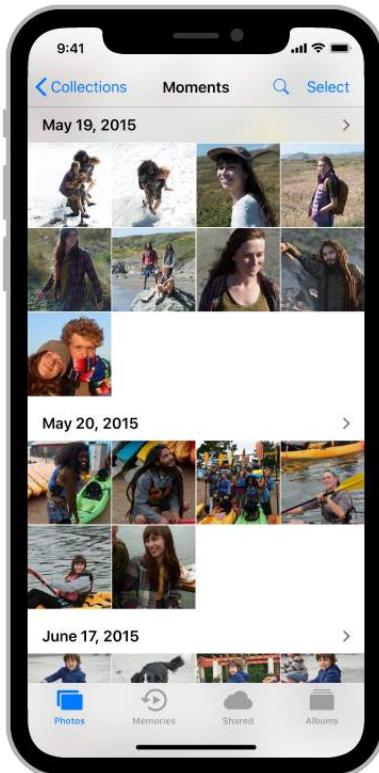
创建合适的按钮标签：表达取消操作的按钮都应该叫做“取消”（原文中是 Cancel，在中文环境里可以更灵活的运用来匹配操作类型，如退出、关闭等）。

强调会产生破坏性操作的按钮：如果警告弹窗的按钮点击后会导致破坏性的操作，例如“删除”操作，那么应该将其设计为具有破坏性的样式以适配系统。开发者指南，请参阅 [UIAlertAction](#) 里的 [UIAlertActionStyleDestructive](#)。除此之外，需要设置一个取消按钮，并让其处于默认选择的状态，这样用户可以选择安全的退出破坏性的操作。

用户可以通过回到主屏幕来取消警告窗口：通过返回屏幕主页并退出 App 的操作（如按 HOME 键）也可以关闭弹出弹窗，和点击警告窗口中的“取消”按钮功能相同。当用户再次返回该 App 时，警告弹窗就消失了。如果你的警告框里没有取消按钮，当用户需要退出 App 时，必须在 App 的代码中执行取消操作。

8.4 精选

精选页面可以有序地管理一系列内容，例如一组照片，是可定制且在高度视觉化的布局中进行呈现。由于精选页面并不是一个严格的线性布局，因此它特别适合显示那些尺寸不一的图像。通常，精选页面更适合展示照片内容。为了能够直观的区分类目，可以选择性地实现背景和其他装饰性视图。



精选支持丰富的交互和动画。默认情况下，你可以通过点击来进行选择，通过长按进行编辑，还可以使用滑动操作来滚动视图。如果你的 App 有其它需求，你可以添加更多的手势来自定义这些操作。在精选页面内，“插入”、“删除”、“重新排序”等操作都支持动画效果，同样，你也可以自定义照片演示的幻灯片效果（批量选择照片后在事件弹窗中可以启用幻灯片）。

在标准行或网格布局够用时避免使用全新的设计方式：

- 精选页面应该提高用户体验，而不是让页面本身成为用户关注的焦点

- 让用户更容易地找到所需内容，因为过于复杂的操作会让用户丧失兴趣。
- 应该在内容周围保持适当的留白，使布局更整洁防止相互叠加。

考虑使用表格排列的方式而不是文本集合：在可滚动列表中查看和消化内容会更简单有效。 (译者注)

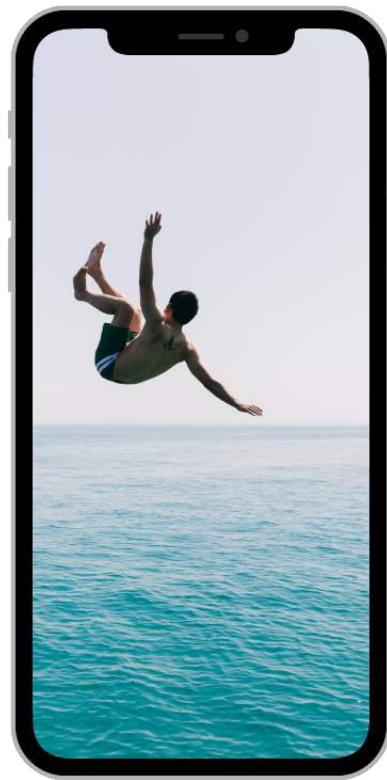
请谨慎使用动态化布局：精选页面的布局可以随时更改。如果你决定要动态地调整精选页面的布局，请确保其中的变化是有意义而且容易被用户理解的，缺乏动机的布局变化会给用户带来不符合预期且难以使用的印象。如果由于布局的变化而导致用户丢失当前的关注点或所处的场境，用户会觉得你的 App 已经失控了。

开发者指南，请参阅 [UICollectionView](#)。

注：原文应该意指在精选的页面中，排列内容时图文混排优于只排列标题文本的形式。

8.5 图像视图

图像视图可以在透明或不透明的背景中显示单图或动画序列图。在图像窗口中，图像可以被拉伸、缩放来调整到合适的大小或者固定在一个特定的位置上。默认情况下，图像窗口不支持用户的交互行为。



尽可能确保动画序列中的所有图像都是相同尺寸：理想情况下，图像应该预设为当前窗口大小，这样系统就不需要对其进行任何缩放。如果系统必须对图像进行缩放操作，那么当所有图像的大小和形状都相同时，最容易实现所需的结果。

开发者指南，请参阅 [UIImageView](#)。

注意：如果一张图片被认为是色彩丢失的样本，那么给图片着色的同时必须要为它提供图片窗口。请参阅 [Custom Icons](#)。

开发者指南，请参阅

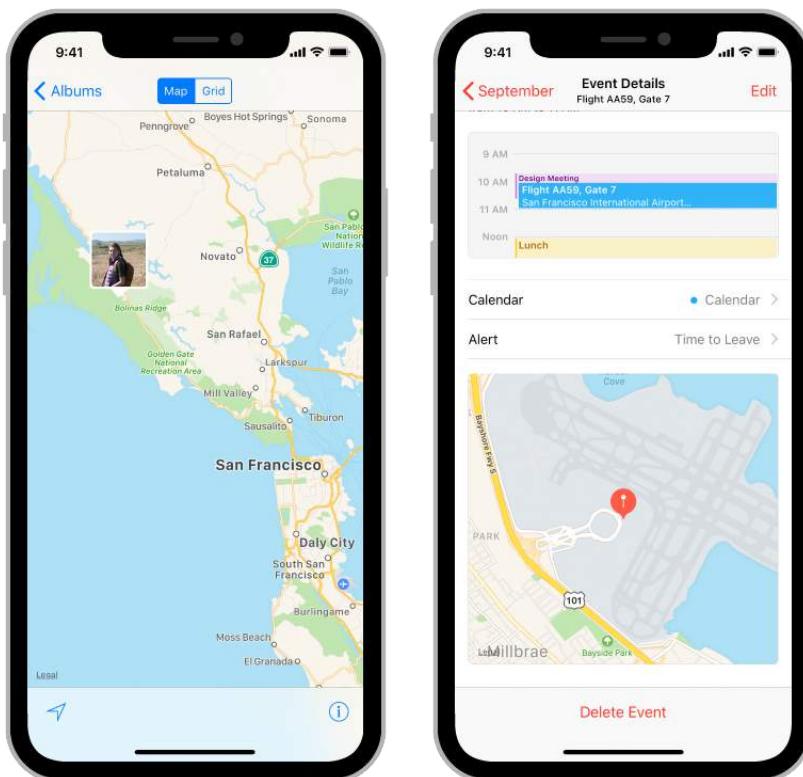
[UIImage](#)

里

的 [UIImageRenderingModeAlwaysTemplate](#)。

8.6 地图

你可以在 App 中使用地图窗口来呈现地理数据，并支持系统内置应用“地图”中提供的大部分功能。使用地图窗口可以显示标准地图、卫星图像，或者两者兼而有之。它可以显示备注（图钉）和叠加图层，支持缩放和平移。如果你的 App 支持路线功能，比如在一个记录运动路线的 App 中，你可以使用地图窗口来显示该路线。



通常，允许用户和地图进行交互：用户习惯于使用手势与系统内置的地图 App 进行交互，因此他们也希望使用同样的方式与你 App 中的地图进行交互。

使用用户心里预期的图钉颜色：地图图钉用来显示用户在地图上感兴趣的地点。因为用户已经熟悉了“地图”App 里图钉的颜色，所以最好避免在你的 App 里重新定义这些颜色的含义：红色表示目的地，绿色表示起始点，

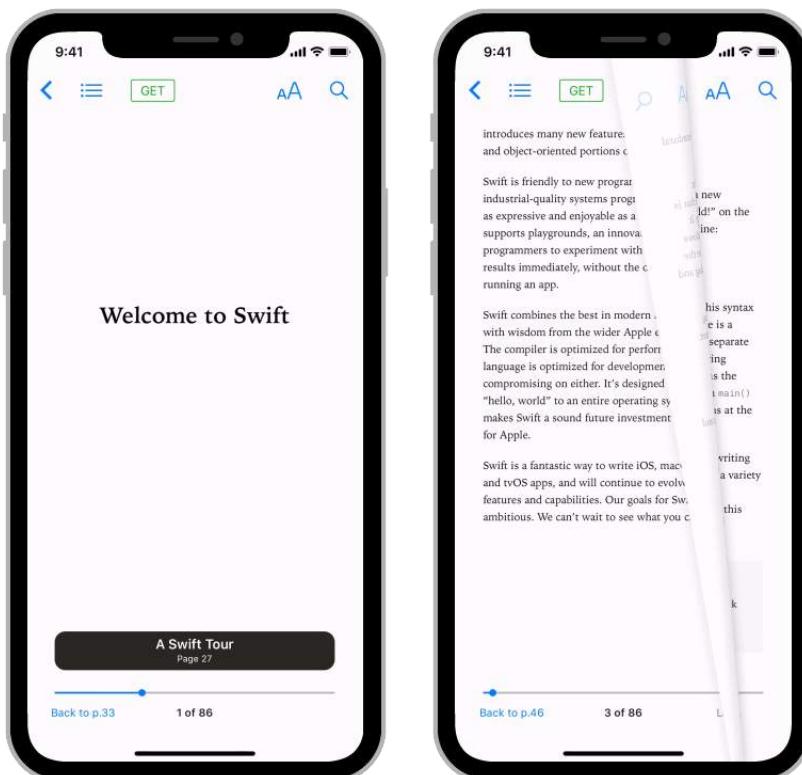
紫色用来标识用户指定的地点。

开发者指南，请参阅 [MapKit](#)。

8.7 翻页

页面视图控制器可以实现翻页时的线性导航，用于文档、书籍、记事本或日历中。页面视图控制器有两种实现页面间切换的方式：

- 滚动过渡：没有特定的样式，页面流畅地从一个页面滚动到下一个页面。
- 翻页过渡：当你在屏幕上滑动的时候，页面就会弯曲，和阅读纸质书时的翻页效果一样。



滚动过渡翻页过渡

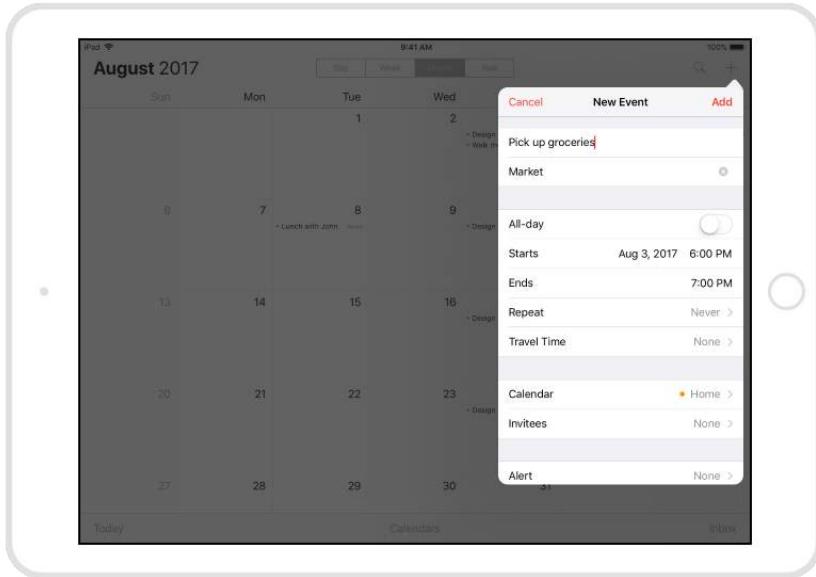
依序切换页面的方式：页面视图控制器只能实现相邻页面之间的跳转切换，如果用户需要在你的 App 中切换到任意的页面，那么就需要自定义一个控件来实现这个功能。

开发者指南，请参阅 [UIPageViewController](#)。

8.8 弹出框

弹出框是在用户点击一个控件或者屏幕上的一个区域后，出现在屏幕其他内容之上的临时窗口。通常来说，弹出窗口包含一个指向它出现位置的箭头。弹窗可以是非原生的也可以是系统原生样式的：对于非原生的弹出窗框，通过点击屏幕上的其他部分或弹出框上的按钮关闭它；对于系统原生样式的弹出框，使用其中的“取消”按钮或其他按钮来关闭。

弹出框适合出现在较大的屏幕上，它可以包含各种各样的元素，如导航栏、工具栏、标签栏、表格、精选窗口、图像、地图和自定义窗口。在弹出框出现且没有被取消时，通常不能操作其它视图的内容。使用弹出框显示与当前屏幕内容相关的选项或信息，例如，当你点击“事件”按钮时，许多 iPad 上的 App 会显示一个共享选项的弹出窗口。



尽量不要在 iPhone 上显示弹出窗口：通常，弹出窗口应该在 iPad 上的 App 中使用。在 iPhone 的 App 中，由于屏幕大小有限，请在全屏视图中显示信息。有关指南，请参阅 [Modality](#)。

请使用“关闭”按钮进行确认和引导：弹出框中的关闭按钮应该传达出明确的定义，比如“取消”表示不保存且退出，“完成”表示保存并退出。通常，弹出框不再需要显示时就应该自动关闭。在大多数情况下，当用户在弹出框区域外点击或在弹出框中选择一个项目时，弹出窗口就应该关闭。如果存在多个选择项，弹出框应该保持打开状态，直到有人点击取消按钮或者在弹出窗口区域外点击将其关闭。

请在非原生弹框关闭时自动保存用户已操作的内容：用户会不小心点击到弹出框外的区域而取消非原生弹出框，所以只有当用户明确点击取消按钮时才不保存操作内容直接退出。

弹出框应该在屏幕的合适位置出现：弹出框的箭头应该直接指向使其浮现的控件位置。弹出框无法在屏幕上进行拖拽，所以弹出框不应该遮挡屏幕上需要被用户看到的信息。弹出框也不应该遮挡住点击后会浮现弹出框的控件。

确保同一时间内屏幕上只有一个弹出框：显示多个弹出框会使界面显得混乱，给用户造成困扰。不要让弹出框有级联或层级关系，这样会使一个弹出框从另一个弹出。如果你需要显示一个新的弹出窗口，请关闭打开的第一个弹出窗口。

不要在弹出框出现时显示另一个窗口：只有警告窗口能够出现在弹出框的上方。

如果允许，请让用户仅通过一次点击就可以关闭一个弹出框接着打开另一个弹出框：当几个不同的栏目按钮都能打开弹出窗口时，这个操作会减少用户的额外点击。

避免将弹出框显示的太大：弹出框不应该占据整个屏幕，它的大小应该恰好足够显示其中的内容，并指向其出现的位置。请注意，系统可能会调整弹出框的尺寸来确保其良好适应屏幕。

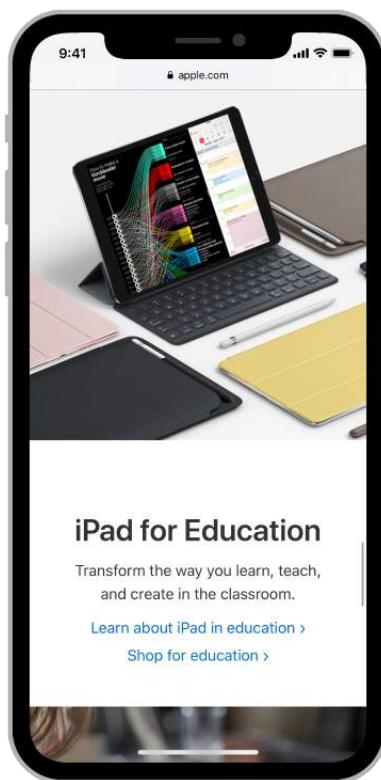
请确保自定义的弹出框让用户理解这就是弹出框：虽然你可以自定义弹出框的许多视觉效果，但尽量避免设计出一个用户难以理解的弹出框。使用标准控件和视图的弹出框看上去最容易被用户所接受。

确保弹出框的大小改变过程有一个平滑的过渡：部分弹出框提供了相同内容的精简和扩展视图两种样式，如果要调整弹出框的大小，请避免让用户感觉是一个新的弹出框取代了原先的弹出框。

开发者指南，请参阅 [UIPopoverPresentationController](#)。

8.9 滚动视图

滚动视图能让用户浏览比屏幕上可见区域里更多的内容，比如文档中的文本或者收集的图像。当用户滑动、快速滑动、拖拽、点击和缩放时，滚动视图会跟随这个手势以一种自然流畅的方式显示或缩放内容。滚动视图本身没有什么样式，但是当用户与之交互时，会有一个滚动条出现。滚动视图还可以在分页模式下进行操作（iPad 的分页模式），此时会滚动显示出一个全新的内容页面，而不是在当前的页面中进行移动。



支持适当的缩放操作：如果缩放操作对你的 App 有用，就可以让用户通过双指开合或者双击来进行放大和缩小，如果你的 App 支持了缩放，你需要设定一个最大和最小的缩放范围，例如：允许用户放大文字直到一个字符填满屏幕，虽然这对大多数 App 来说是什么意义的。

可以考虑在分页模式的滚动视图中使用页码控件：页码控件可以让用户知道当前浏览到什么位置了，还可以显示有多少可用的页面或内容块。如果滚动

视图上显示了一个页码控件，最好禁用和页码控件同一方向的滚动条，以避免误操作。关于页码控件使用的更多信息，请查阅 [Page Controls](#)。

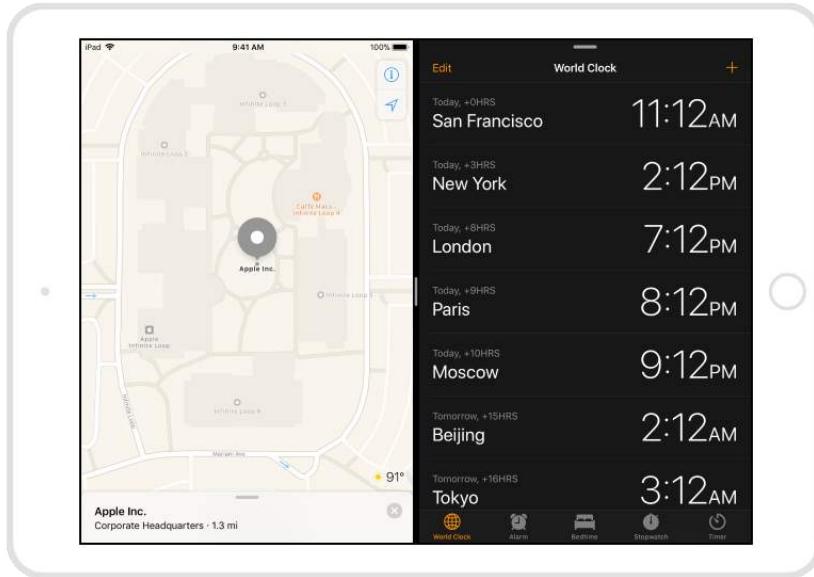
不要在一个滚动视图里置入另一个滚动视图：这样做会出现失控和意料之外的情况。

通常，一次只显示一个滚动视图：用户经常在滚动时做出大幅度的滑动手势，而且很容易误触到同一屏幕上相邻的另一个滚动视图。如果你需要在一个屏幕上放置两个滚动视图，可以考虑让它们朝不同的方向滚动，这样一个手势就不能同时影响两个视图，例如，当 iPhone 在竖屏状态时，“股票” App 中显示的股价是横向滚动的，而公司的特定信息是竖向滚动的。

开发者指南，请参阅 [UIScrollView](#)。

8.10 分屏视图

分屏视图可以管理两个并列面板的显示，包括主面板中作为主要信息的展示而在副面板中展示其关联信息。每个面板中都可以包含导航栏、工具栏、标签栏、表格、精选、图像、地图和自定义窗口等多种元素。分屏视图通常应用于筛选信息的操作：在主面板中显示一个筛选类别列表，将筛选后的结果显示在副面板中。如果你的 App 有需要，主窗口可以覆盖在副面板之上，主面板在不使用时也可以在屏幕上隐藏，当设备处于竖屏状态时，这一点非常有用，因为能够在副面板中显示更多的内容。



请根据内容来选择一个合适的分屏视图布局：默认情况下，分屏视图将屏幕的三分之一用于主面板，三分之二用于副面板。屏幕也可以分成两半，根据你的内容选择合适的分割，但是请确保窗口看起来是平衡的。避免创建比主面板更窄的副面板。

突出显示主面板中当前选择的选项：虽然副面板中的内容可以更改，但它应该始终对应于主面板里选中的选项，这有助于用户理解两个面板之间的关系

通常，只在分屏视图的一侧显示导航：在分屏视图的两个中都放置导航会让用户失去方面板向感，难以辨别两个面板之间的关系。

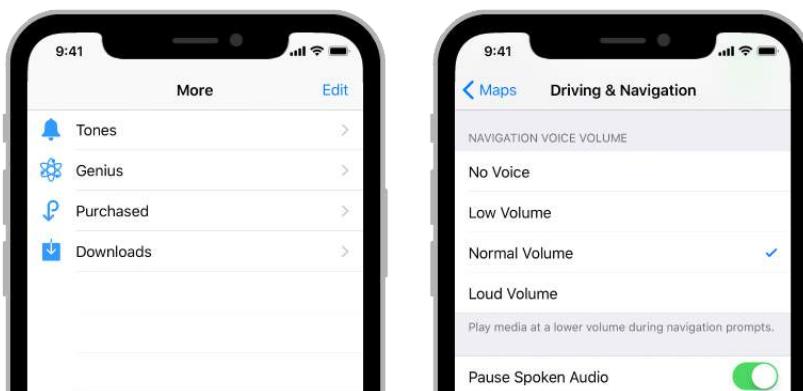
提供多种方式来唤醒隐藏的主面板：默认情况下只会显示一个副面板，因此需要提供一个按钮（通常位于导航栏）用来呼出主面板。除非你的 App 使用滑动手势来执行其他功能，否则应该支持用户从屏幕的边缘向屏幕内滑动来唤出主面板。

开发者指南，请参阅 [UISplitViewController](#)。

8.11 表元素

表元素以一个可滚动的单列列表进行展示，其中的行可以对其进行相应的分组。表元素能够以列表的形式明确有效地显示大量或简略的信息。通常，表元素是基于文本格式的最佳形式，表元素常常作为一种导航工具出现在分屏视图的一侧，而与之相关的内容显示在另一侧，关于分屏窗口请查阅 [Split Views](#)。

iOS 定义了两种表格样式：平铺式和分组式。



平铺式（左）：行可以被分为若干带标签的段落，而表格的右侧可以出现垂直排列的索引符号（译者注：即从字母 A 到 Z 的字母索引条，如通讯录的联系人列表页）。页眉会在这一节的首个条目前显示，页脚会在最后一个条目后显示。

分组式（右）：行会被分成多个分组显示，每组都能够以页眉开始，页脚结束。这种类型的表格至少包含一个分组，每个分组中至少包含一行列表，分组型列表中没有索引条。

考虑表元素的宽度：过窄的表元素会导致文本被截断或者挤到一起，用户很

难快速扫浏览这些内容。过宽的表格也很难让用户快速浏览，需要用内容来填充部分空间。

尽快呈现表元素中的内容：不要等表格相关的所有数据都加载完成才一次性显示出来，可以先在屏幕中显示文本内容，如图像等较为复杂的数据则加载完成后再显示出来。这种技术可以让用户尽快看到有用的信息，同时也增加了 App 交互响应效率。有时，在新数据加载出来前，显示旧数据也是有用的。

请在内容加载时给予用户反馈：如果表元素中的数据需要一定时间来加载，请显示一个进度条或动态旋转指示器，让用户知道 App 还在运行，且在执行数据加载中。

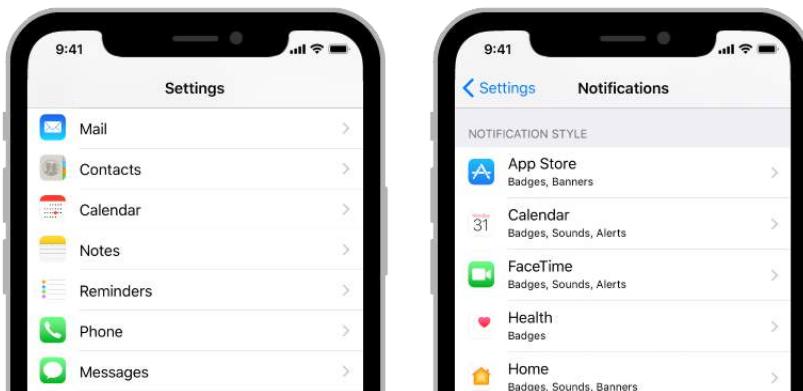
保持最新的表元素内容：定时更新（后台读取）并准备表元素中的最新数据，但不要改变屏幕当前的显示情况。用户在滚动内容时才能提前做好准备，并将新的内容添加到表的开头或结尾。有些 App 在加载完新数据时会出现一个提示，并提供了某操作可以跳转到新数据的位置。在 App 中包含一个刷新控件也是不错的主意，这样用户可以在任何时候手动刷新，关于刷新控件的更多信息请查阅 [Refresh Content Controls](#)。

避免在包含右对齐方式的表元素中使用索引：索引是通过大幅度的滑动手势来控制的，当使用手势操作时，如果其他交互元素（例如详情指示器）停留在附近，可能系统会发生对操作的误判而导致启动了其他元素。

开发者指南，请参阅 [UITableView](#)。

表行

在表元素中使用标准的单元格样式定义显示的内容。



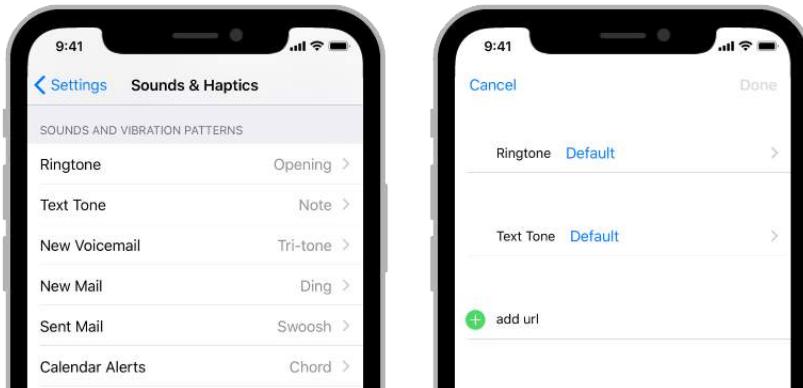
基本型(默认) (左)：包含一个在行左端的图片（可选）和一个与图片右边左对齐的文字标题，适用于显示不需要补充信息的项目。

开发者指南，请参阅 [UITableViewCell](#) 里的 [UITableViewCellStyleDefault](#)

副标题型 (右)：包含一个左对齐的文字标题（一行）和在下方同样左对齐的副标题，这些相似的行在一个表中可以得到很好的展示，因为副标题可以用来区分不同的行。

开发者指南，请参阅 [UITableViewCell](#) 里的

[UITableViewCellStyleSubtitle](#)。



右提示(Value1)。左对齐的标题与右对齐的副标题在同一行中显示。

请参阅 [UITableViewCell](#) 里的 [UITableViewCellStyleValue1](#)。

左提示(Value2)。 右对齐的标题后面跟着一个左对齐的副标题，并在同一行中显示。

请参阅 [UITableViewCell](#) 里的 [UITableViewCellStyleValue2](#)。

所有表元素的标准单元格样式都允许使用图形控件，例如复选标记或详情指示器，当然，增加这些元素后会减少标题和副标题的使用空间。

请使用简洁的文本，以避免被截断：被截断的单词和短语让用户难以阅读和理解。所有类型的表元素单元格都会自动截断文本，但不同的单元格样式和截断发生的位置会产生问题。

你可以为“删除”按钮自定义一个标题：如果你自定义的标题可以让用户更清晰地理解该操作的含义，你可以替换系统默认的“删除”文本。前提是这一行支持“删除”操作。

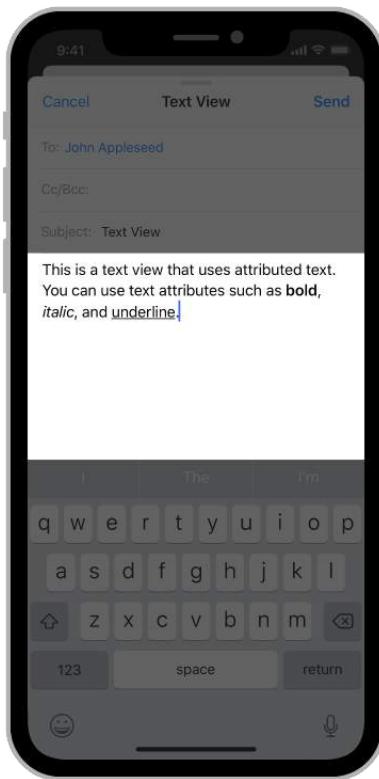
在用户的选中操作中给予反馈：当用户在点击某个可选中的项目时，他们会希望这个行应该有短暂的高亮，点击过后，用户期望出现一个新的视图或者有什么其它改变，例如出现一个勾选标记，以表明这个项目已经被选中或激活。

为非标准的表元素行设计一个自定义单元格样式：标准样式适用于各种常见场景，但有些内容或整个 App 设计可能需要一个高度自定义的表元素外观，想了解如何创建自己的单元格，请参阅 [Table View Programming Guide for iOS](#) 里的 [Customizing Cells](#)。

开发者指南，请参阅 [UITableViewCell](#)。

8.12 文本视图

文本视图可以容纳并显示文本内容。文本视图的高度不限，当内容超出视图时可以滚动查看。默认情况下，文本视图中的内容是以左对齐的黑色系统字体显示。文本视图支持编辑，当用户在视图中点击时，会唤出键盘。



保持文本的可读性：虽然你在文本中可以使用各种字体、颜色和对齐方法，但是保持文本内容的可读性是非常重要的。最好在文本视图中支持动态文本，这样文本视图在用户改变字体大小后依然会易于阅读。你还应该通过启用可访问性选项，如粗体文字来测试内容。

唤起合适的键盘类型：根据不同类型内容的输入，iOS 提供了几种不同的键盘类型。为了简化内容输入，在文本视图的编辑过程中，唤起的键盘应该匹

配相应字段的类型（例如输入手机号只唤起数字键盘）。关于键盘类型的

完整列表，请参阅 [UITextFieldTraits](#) 里的 [UIKeyboardType](#)。

开发者指南，请参阅 [UITextView](#)。

8.13 网页视图

网页视图可以加载并显示丰富的网页内容，可以直接在你的 App 中嵌入

HTML 和网页。“邮箱”App 在邮件中就使用了 web 视图来展示。



需要时可以使用向前和返回的导航按钮：web 视图支持向前和返回操作，但

默认情况下这个操作是被禁止的。如果用户需要使用你的 web 视图访问较

多页面，请提供相应的控件来启用向前和返回的操作。

避免使用 web 视图来构建 web 浏览器：使用 web 窗口可以让用户不用离

开当前 App 中的内容就能简单地访问一个网页，但 Safari 才是用户在 iOS 上浏览网页的主要方式。在你的 App 中复制 Safari 的功能是没有必要也不被鼓励的。

开发者指南，请参阅 [WKWebView](#)。

第九大章

控件

Controls

9.1 按钮

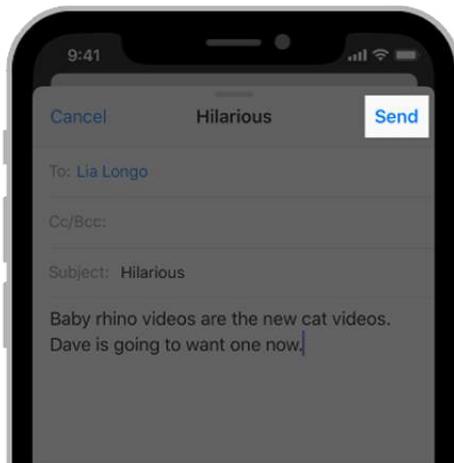
按钮用于启动 App 特定的操作，能自定义按钮背景，并且常常包含标题或者图标。

系统提供了许多预定义的按钮样式，可用于大多数使用场景。你也可以设计自定义按钮。

开发者指南，请参阅 [UIButton](#)。

系统按钮

系统按钮通常显示在导航栏和工具栏中，但在任何地方都可使用。



标题要使用动词：一个表明动作的标题说明按钮可交互的，并且会让用户知道点击后发生什么。

(英文) 标题要首字母大写：除了冠词、并列连词以及，四个或四个以下字母的介词之外的每个词都要大写首字母。

标题要短：超长的文本会让用户界面看起来很挤，并且可能在小屏幕上显示时被截断。

仅在必要时添加描边或背景：默认情况下，系统按钮没有描边或背景。

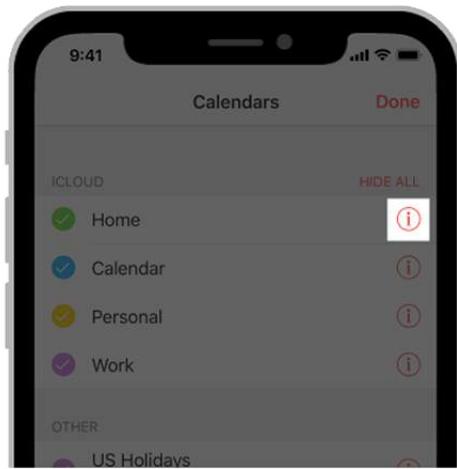
然而，在某些使用情景中，需要使用描边或背景来表示交互性。在“电话”App中，描边的数字键加强了通话的传统形式感，“呼叫”按钮的绿色背景也很吸人眼球，便于点击。

开发者指南，请参阅[UIButton](#)下的[UIButtonTypeSystem](#)。

详情按钮

详情按钮会打开一个窗口——通常是模态窗口——包含屏幕上特定项目的

相关附加信息或功能。尽管你可以在任何一种窗口里使用它们，但详情按钮通常用在表格中，来获取关于某特定行的信息。



在表格中恰当使用详情按钮：当详情按钮显示在表格行中，点击该按钮将显示附加信息。点击其他地方，会选择该行，或者发起 App 定义行为（译者注：即点击该行后的进行的操作）。

如果你想让人们点击整行来查看附加详情，请勿使用详情按钮。而是使用细节披露控件，其显示为^形或v形。详情查看[UITableViewCell](#)下的[UITableViewCellAccessoryType](#)。

开发者指南，请参阅[UIButton](#)下的[UIButtonTypeDetailDisclosure](#)。

信息按钮

信息按钮会显示有关 App 的配置详情，有时会在翻转当前视图后在背面显示。



信息按钮有深色和浅色两种风格。选择最适合你 App 设计的风格，才不会让用户感觉困惑。

了解更多开发者指南，请参阅 [U I B u t t o n](#) 的 [UIButtonTypeInfoLight](#) 和 [UIButtonTypeInfoLight](#) 按钮类型。

添加联系人按钮

用户可以点击添加联系人按钮来浏览现有联系人列表，并选择一个来插入文本字段或其他窗口。例如，在邮件中，您可以点击邮件的“收件人”字段中的“添加联系人”按钮，从联系人列表中选择收件人。



除了添加联系人按钮之外，还允许键盘输入： 添加联系人按钮只是一种新方法，而并没有取代打字输入联系人信息。该按钮为添加现有联系人提供了捷径，也允许人们用键盘输入联系人信息。

开发者指南，请参阅 [UIButton 的 UIButtonTypeContactAdd](#)。

9.2 情景菜单

在 iOS13 和更高的系统版本中，您可以利用情景菜单来为屏幕上显示的元素提供额外的功能选项，而不会是界面变得混乱。



情景菜单跟 Peek & Pop 有点类似，但是有两个主要的不同：

- 情景菜单支持所有运行着 iOS13 的设备；而 Peek & Pop 只能在支持 3D touch 的设备上使用。
- 情景菜单立即显示情景相关的命令；Peek & Pop 还需要上划才能显示这些命令。

为了呼出情景菜单，用户可以使用系统定义的长按手势或 3D Touch (3D Touch 可以更快的呼出情景菜单)。打开时，情景菜单显示项目的预览视图并列出对其起作用的命令。用户可以选择一条命令或将项目拖动到另一个区域，窗口或应用中。

在一致的地方使用情景菜单：若您在一些地方使用情景菜单，而在另一些地方却没有，用户会疑惑道理哪里才能用这功能，并怀疑是你的 APP 出现了问题。

只包含该项目最常用的命令：例如，在邮箱信息的情景菜单中，提供「回复」和「移除信件」的命令才是有意义的，提供格式相关的或收件箱的命令就不合理。而且，列出过多的命令会对用户形成负担。

如果过于复杂则使用子菜单：子菜单是一个情景菜单项，它显示了与父级命令逻辑相关的二级菜单。为子菜单提供能够直观描述其内容的标题，这样人们不用打开也能理解子菜单的功能。简洁、明确动作的标题还能够让用户跳过当前情境下不需要的子菜单。

子菜单只能有一个层级：尽管子菜单能够缩短情景菜单的长度，精简用户可执行的命令，但是如果子菜单超过了一个层级依然会是体验变得更复杂，并难以导航。

将最常用的选项放在菜单顶部：当用户打开情景菜单时，他们的注意力会集中在菜单的顶部区域。将最常用的选项放在顶部有助于用户快速找到自己需要的。

使用分隔符对相关菜单项进行分组：创建视觉上的组能够帮助用户快速扫描菜单。比如你可使用分隔符将编辑相关的组与分享相关的组区分开。通常，不要在情景菜单中分出三个以上的组。

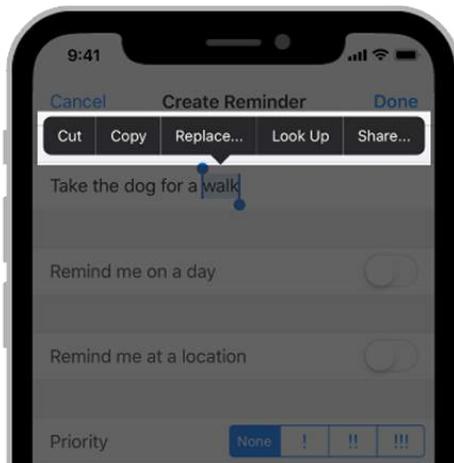
不要对同一项目同时使用情景菜单和编辑菜单：否则将会对用户造成困扰，也会增加系统判断用户意图的难度。相关信息请参阅 [编辑菜单](#)。

不要再提供打开项目的按钮：用户可以直接点击预览打开项目，再提供一个直接描述为「打开」的按钮多此一举。

开发者请参阅 [UIContextMenuInteraction](#)。

9.3 编辑菜单

用户可以在文本字段，文本窗口，网页窗口或图像窗口中触摸并按住或双击某个元素（注：例如文本字段中的文字），来选中内容并显示编辑选项，如“复制”和“粘贴”。



为当前情景显示适当的命令：默认情况下，选项包括“剪切”，“复制”，“粘贴”，“选择”，“全选”和“删除”命令，每个命令都可以禁用。

如果没有选中，菜单不应显示选项，例如“复制”或“剪切”。同样，如果已经选中了，菜单就不应该具有“选择”这个选项了。

如有必要，适当调整编辑选项的位置：默认情况下，菜单位于插入点或选择内容的上方或下方，具体取决于可用空间，并包含指向插入点或选择内容的指针（译者注：即编辑菜单上下指向内容的实心黑三角，用于指示内容和菜单的相对位置关系）。

虽然你无法更改菜单的形状，但可以调整它的位置来防止它覆盖重要内容或部分界面。

不要使用其他功能相同的控件作为编辑菜单：提供多种启动操作的方法会导致用户体验不一致，并引起混乱和困惑。例如，如果你的App让用户使用菜单来复制内容，就不要使用复制按钮。

让可能有用的非编辑文本也能被选择和拷贝：人们经常想给邮件、笔记或者网页搜索添加静态内容，比如一个图像标签或者社交媒体状态

不要将编辑菜单的选项添加到按钮中：如果你这样做，人们会使用按钮，而不去打开编辑选项。

让编辑操作可以撤销：在动作执行前，菜单不会请求确认。因为有的人在执行操作之后改变主意，要确保一直都有撤销和重复功能。

用自定义命令扩展编辑选项：你可以通过提供附加的App操作来增加选项。和标准命令类似，自定义命令应该在选中的文本或对象上操作。

在系统提供的命令之后显示自定义命令：不要让系统命令和自定义命令杂乱分布，系统命令对用户来说更为实用并且使用频率更高。

让自定义命令的数量尽可能少：太多选择会让用户感到困惑。

自定义命令名字要短：命令的名字应该是动词，或者简洁描述将执行动作的动词短语。

(英文中)除了冠词、并列连词、以及四个字母以下的介词外，都要首字母大写。

开发者指南，请参阅 [Text Programming Guide for iOS](#) 下的 [Copy, Cut, and Paste Operations](#)，以及[UIViewControllerAnimated](#)。

9.4 标注

标注描述了屏幕上的界面元素或者提供短信息。尽管用户不能编辑标注，他们有时能拷贝标注的内容。标注能显示任意数量的静态文本，但还是精简为好。



让标注明确易读：标注可以包含简单朴素的或风格化的文本。哪怕你调整标注的样式或者使用自定义字体，也要保证可读性。最好采用动态文本，这样当用户在设备上改变文本大小时，标注依然容易阅读。详见 [Dynamic Type](#)

你也应在无障碍选项打开时测试标注，比如粗体文本。详见 [Accessibility](#)。

文本详情请参阅 [String Programming Guide](#)；创建样式文本详见 [Attributed String Programming Guide](#)，关于标注开发指南详见 [UILabel](#)。

9.5 页面指示器

页面指示器将页面序列扁平化，用于显示当前页面的位置。指示器是一连串的小点，用来代表按顺序排列的可用页面。

实心点代表位于当前页面。视觉上，这些点是等距离的，但如果屏幕上出现太多点的话就会被切掉。用户可以点击页面指示器的首端或者尾端来访问下一页或前一页，但是他们不能点击特定的点到特定的页面。（译者注）。页

面导航总是按顺序出现，通常把页面向另一边滑动。



注：即用户可以通过点击指示器第一个点及之前的某个位置来向前翻一页，向后翻一页类推，但是若当前处于第一个页面，点击第三个点，会向后翻到第二个页面而不是跳转到第三个页面

不同层级的页面之间不要使用页面指示器：页面控制不显示页面之间的联

系，也不显示每个点对应哪个页面。这种控件是为同等级的页面设计的。

不要显示太多页面：超过十个点就很难一眼数清，如果超过20个点再去按顺序浏览这些页面就很耗时间。如果你的App需要显示超过20个同等级页面，考虑使用不同的方式：比如表格，能实现非顺序的导航。

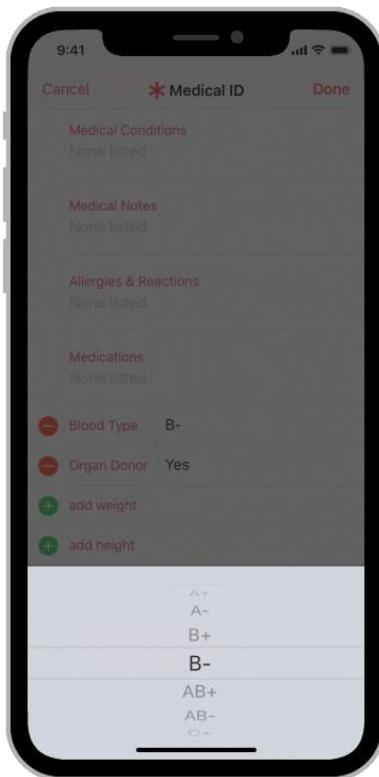
在页面底部居中放置页面控件：页面控件应该居中放在页面内容末尾和屏幕底部之间。如此一来控件可见且不遮挡页面内容。

开发者指南，请参阅 [UIPageControl](#)。

9.6 选择器

选择器包含一个或多个由不同值组成的滚动列表，每个都可以被单独选中——该值在窗口中间的黑色文本中做凸显提示。选择器通常显示在屏幕底端，或者在用户编辑字段或点击菜单时显示在弹出框里。选择器也可以内联

显示，比如在“Calendar”的事件中编辑日期时（开始日期和结束日期使用同一个选择器形式）。选择器的高度大概为单个值行高的五倍。选择器的宽度可与屏幕同宽或者与封闭窗口等宽，具体取决于设备和使用情景。



使用易推测且逻辑排列的值：当滚动列表静止时，选择器中的很多值可能被隐藏。最好用户能推测到这些隐藏值是什么，比如按字母顺序排列的国家，便于快速浏览列表。

避免切换屏幕来显示选择器：当显示在当前使用情境中且低于或者紧挨正在编辑的字段时，选择器能更好地发挥作用。

对于多数值列表，请使用表格来替换选择器：通过选择器导航一个长长的列表可能会很乏味。表格可以调整高度，而且能包含索引，滚动可以更快速。

开发者指南，请参阅 [UIPickerView](#)。

日期选择器

日期选择器是用于选择特定的日期、时间的高效界面。它也提供显示倒计时的界面。



日期选择器有四个模式，每个代表不同的可选择值。

- 日期。显示年、月、日。
- 时间。显示时、分、上午下午（可选）
- 日期和时间。显示日期、时、分、上午下午（可选）
- 倒计时。显示时和分，最大显示23小时59分钟。

日期选择器中显示的确切值和它们的顺序取决于用户的语言环境：当对特定时刻有需求时，考虑提供不同的时间间隔。默认情况下，一个分钟列表包含

60个值（0-59）。你可以选择增加分钟间隔的值，只要它能被60整除即可。例如，你可能想要刻钟的间隔（即只有 0,15,30,45 四个选项）。

开发者指南，请参阅 [UIDatePicker](#)。

9.7 进程指示器

不要让用户盯着静态屏幕等待你的App加载内容或者处理冗长的数据。 使用活动指示器和进度条，让用户知道你的App没有停顿，并告诉他们大概还要等多久。

另见 [Loading](#)。

滚动圈

当执行不可量化的任务（例如加载或同步复杂数据）时，滚动圈就会派上用场。任务完成时滚动圈消失。滚动圈是不可交互的。



进度条的使用要优先于滚动圈：如果操作等待是可量化的，请使用进度条而非滚动圈，以便用户可以更好地了解发生的情况和所需时长。

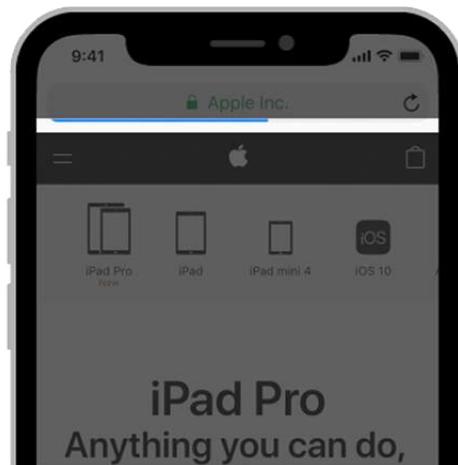
让滚动圈保持运行：人们认为滚动圈停止也就意味着进程停止。保持运行，让用户知道活动仍在进行中。

如适用，可在等待任务完成时提供有效信息：在滚动圈上方添加标注以提供附件信息。不要使用表意不清的术语，如“加载中”或“验证中”，因为它门没什么实际意义。

开发者指南，请参阅 [UIActivityIndicatorView](#)。

进度条

进度条的轨道会从左到右逐渐填充的方式来显示任务进度。虽然进度条通常伴有取消对应操作的按钮，但其本身是不可交互的。



始终准确显示进度：不要只为了使App看上去在运行而显示不准确的进度信息。进度条只用于可量化任务。非量化任务请使用滚动圈。

使用进度条时对任务持续时间做出明确规定：进度条非常适合显示任务状

态，尤其在它可以传达任务所需用时的时候。

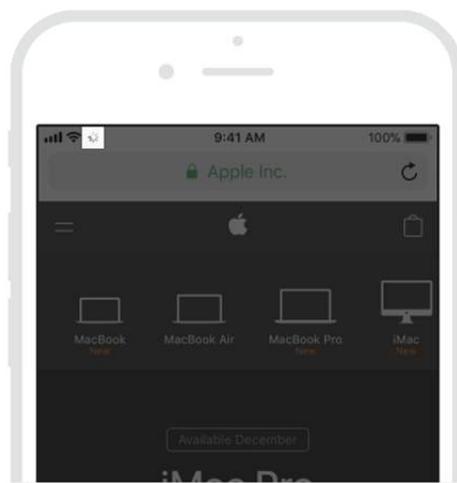
在导航栏和工具栏中隐藏未填充部分：默认情况下，进度条的轨道包括已填
充和未填充部分。当在导航栏或工具栏中使用时，例如表示页面加载时，应
当设置隐藏进度条轨道的未填充部分。

考虑自定义进度条的外观以匹配App：进度条的外观可以根据App的设计
进行调整。例如，您可以指定轨道和填充的自定义色调或图像。

开发者指南，请参阅 [UIProgressView](#)。

网络活动指示器

在状态栏为全屏宽度的设备上，开始联网时，网络活动指示器会在屏幕顶部
的状态栏中旋转。当联网成功时该指示器消失。它看起来就滚动圈，也是不
可交互的。



仅在持续时间超过几秒的网络操作时显示该指示器：快速网络操作时不需
要显示该指示器，因为在用户注意到它的存在或意识到它的传达意图之前，

它可能已经消失了。

另见 [Status Bars](#)。

开发者指南，请参阅 [UI Application](#) 下的 [networkActivityIndicatorVisible](#)。

9.8 刷新内容控件

通常在表格窗口中，刷新控件用来手动即时加载内容，而不用等待下一次自动更新内容。刷新控件是一种专门的活动指示器，默认情况下隐藏，当在视图中下拉重新加载时会出现；例如，在邮件中，您可以将Inbox消息列表下拉，以查看最新消息。



自动更新内容：尽管人们喜欢能够即时刷新内容，但他们也期望定期自动刷新。不要让用户自己负责所有更新。定期更新数据，保持数据更新。

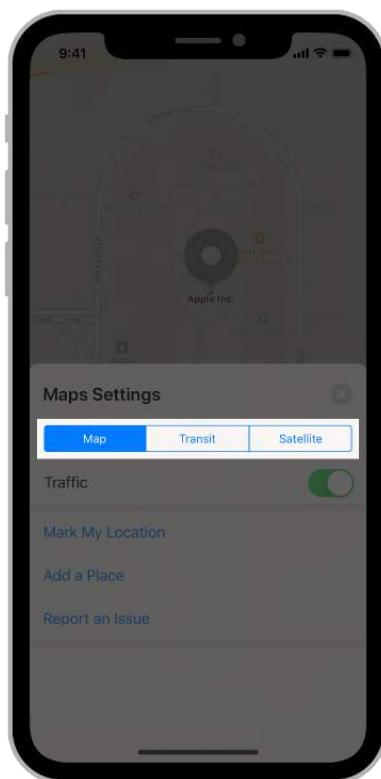
如果有意义可增加一个短标题：刷新控件可以包含标题。在大多数情况下没有必要，因为控件的动画表明内容在加载。如果一定要加一个标题，不要使

用它来解释如何刷新。而是提供刷新的内容的有用信息。例如，在播客中的更新控制中，用标题告诉人们最后一条播客是什么时候更新的。

了解更多开发指南，请参阅 [UIRefreshControl](#)。

9.9 分栏控件

分栏控件是两个或多栏的线性集合，每栏都充当一个互斥按钮。在控件中，所有的栏都是等宽的。像按钮一样，栏也可以包含文本或图像。分栏控件通常用来显示不同的窗口。例如，在地图中，分栏控件可以让你在地图、传输和卫星视图之间切换。



限制栏的数量以提高可用性：更宽的栏更容易打开。在iPhone上，分栏控制内的数量应小于等于5个。

尽量保持栏内容大小一致：因为所有栏都是相同的宽度，如果某些栏内有内容而其他的没有，就会显得不协调。

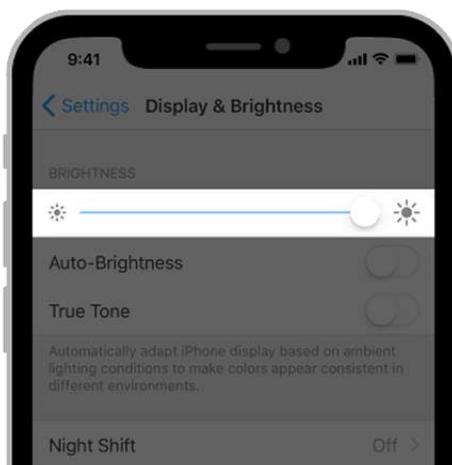
在分栏控制中不要既有文本又有图像：虽然单个栏可以包含文本或图像，但将两者混合在一个控件中可能导致连接中断和混乱界面。

在自定义分栏控件中适当地插入内容：如果更改了分栏控件的背景外观，请确保内容看起来仍然协调，并且不会出现对齐错误。

开发者指南，请参阅 [UISegmentedControl](#)。

9.10 滑块

滑块是一条可以用拇指划滑动的水平轨迹，你可以用你的手指在最小值和最大值之间滑动，以调节屏幕亮度级别或在媒体播放期间的位置等。当一个滑块的值发生变化时，在最小值和拇指之间的部分就会填充上颜色。可以选择显示滑块的左右图标，说明最小值和最大值的含义。



如果有需要，自定义滑块的外观：一个滑块的外观，包括轨迹颜色，拇指滑

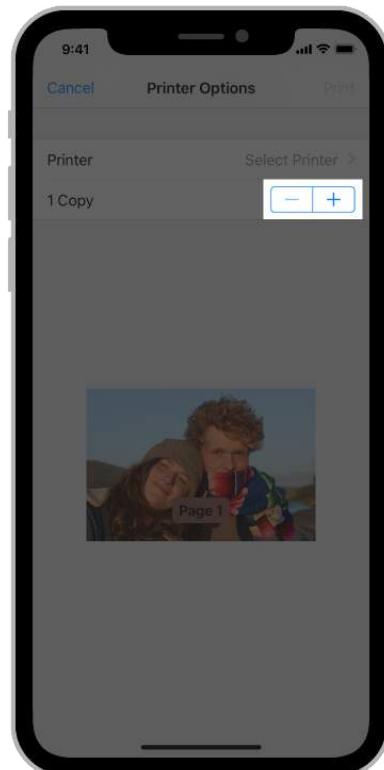
动的图像和左右图标，都可以根据你App的设计风格和想表达的意图进行调整。例如，一个调整图像大小的滑块，可以在左边放个小的图像图标，右边放大的图像图标。

不要使用滑块来调整音量：如果你需要在应用程序中提供音量控制功能，那就使用一个音量窗口，它是可定制的，包括音量滑块和改变音频主动输出设备。要了解如何做出音量视图，请参见 [MPVolumeView](#)。

开发者指南，请参阅 [UISlider](#)。

9.11 分档器

分档器分两部分控制，用于增加或减少量值。默认情况下，分档器的一边显示加号，另一端显示减号。如果需要，符号可以用自定义图像替换。



让分档器控制的值有明显改变：分档器本身并不显示任何值，所以要确保人

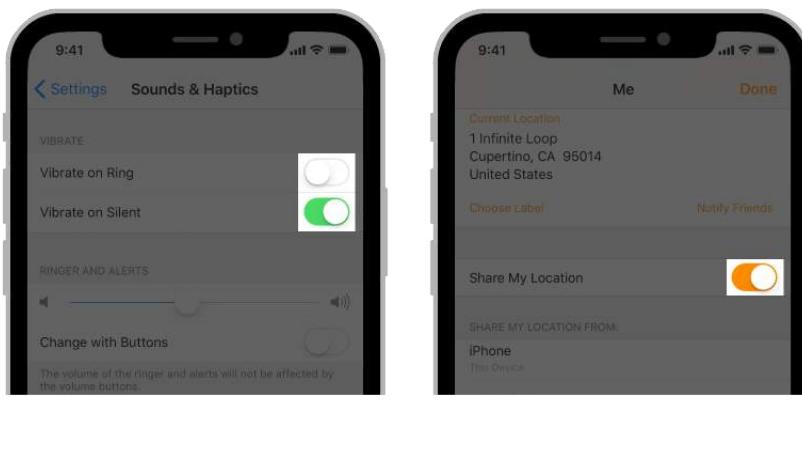
们知道使用分档器时改变的是什么值。

当可能有较大的价值变化时，不要使用分档器：分档器适合做需要一些点击的小改动。例如，在一个打印屏幕上，可以使用一个分档器来设置副本的数量，因为人们很少改变这个设置。但是，使用分档器来选择一个页面范围是没有意义的，因为即使是一个合理的页面范围也需要大量的点击。

开发者指南，请参阅 [UIStepper](#)。

9.12 开关

开关是一个可视控件，用来实现两个相互排斥的状态，比如开和关之间的切换。



可以调整一个开关的外观，以匹配你的App的风格：如果它在你的App中运行良好，改变它的开关状态颜色，或者使用自定义的图像来代表开关。

仅在表行中使用开关：开关被用在表元素中，比如在一个可以开关的设置列表中。如果你需要一个工具栏或导航栏上的类似功能，用按钮代替，并提供

两个不同的图标来显示状态。

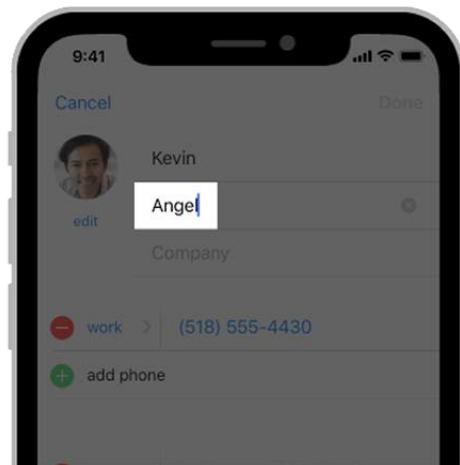
不要使用标注来描述开关的值：开关要么打开，要么关闭。描述这些状态的标签是冗余的，并且会使界面很凌乱。

可以使用开关来管理相关联的界面元素的有效性：开关通常会影响界面上的其他功能。例如，在设置中启用飞行模式，可以禁用某些其他设置，比如移动信号和个人热点。在设置> Wi-Fi 时禁用 Wi-Fi，会导致可用网络和其他选项消失。

开发者指南，请参阅 [UISwitch](#)。

9.13 文本框

文本框是单行、固定高度字段，通常带有圆角，当用户点击它时，它会自动弹出一个键盘。使用文本框来获取少量信息，如电子邮件地址。



在文本框中显示小提示，可以有助于沟通：文本框中没有其他文本时，可以使用占位符，如“电子邮件”或“密码”。当占位符文本描述充分时，不要

使用单独的标注来描述文本框。

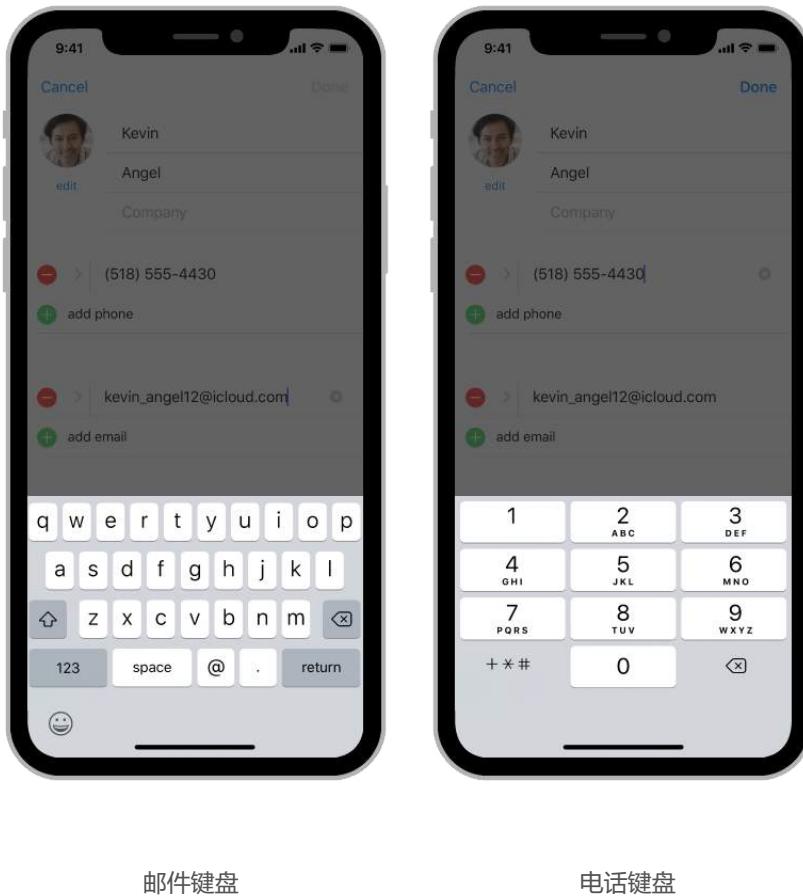
需要的时候，在文本框的右端添加一个清晰的按钮：有了这个按钮，就可以通过点击它清除文本框的内容，不需要继续点击删除键。

需要时候使用保密文本框：当你的应用程序询问敏感数据(比如密码)时，一定要使用保密文本框。

文本框中可以使用图像和按钮来解释目的和提供功能：您可以在文本框的左侧或右侧显示自定义图像，或者可以添加一个系统提供的按钮，例如书签按钮。一般来说，文本框的左端来表示字段的目的，右端显示额外的特性，例如书签。

开发者指南，请参阅 [UITextField](#)。

键盘



显示适当的键盘类型：iOS提供了几种不同的键盘类型，每一种都旨在促进不同类型的输入。为了简化数据输入，在编辑文本字段时显示的键盘应该适合于字段中的内容类型。例如，如果你的应用程序要求输入一个电子邮件地址，它应该弹出电子邮件地址键盘。对于可用的键盘类型的完整列表，请参见 [UITextField Trait](#) 的 [UIKeyboardType](#) 常量。

开发者指南，请参阅 [Custom Keyboards](#)。

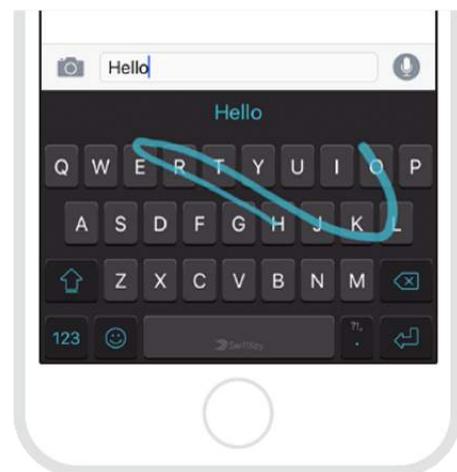
第十大章

扩展

Extensions

10.1 自定义键盘

键盘扩展是指，用定制键盘代替标准键盘。定制的键盘在设置 App 中启用，在“通用”下的>键盘。一旦启用，除了编辑安全文本字段和电话号码字段外，在任何 App 的文本输入中都可以使用该键盘。人们可以启用多个定制键盘，并随时在它们之间切换。



确保你真的需要一个定制的键盘：当你想要看到独特的键盘功能时，可以使用自定义键盘，例如输入文本的一种新颖方式，或者使用 iOS 不支持的语言类型。如果你只想在你的 App 中使用自定义键盘，那么可以考虑创建自定

义输入视图。跳转 [Custom Input Views](#)。

提供一个简单明了的方法在键盘之间切换：人们知道标准 iOS 键盘上的全球键，当你打开多个键盘时，它取代了表情符号，可以快速切换到其他键盘。用户会期望在你的键盘上有类似的跳转按钮。请注意，当你安装多个键盘时，用“地球键”取代表情键。

不要复制系统提供的键盘功能：在 iPhone X 上，表情符号/地球键和听写键会自动出现在键盘下方——甚至是在使用定制键盘的时候。你的 App 不能影响这些按键，所以避免在你的键盘上重复这些内容，造成混乱。

考虑在你的 App 中提供一个键盘教程：用户习惯了标准键盘，学习新键盘就会需要时间。通过 App 的使用说明让键盘上手的更快，而不是只在键盘上操作。告诉用户如何启用你的键盘，在文本输入时激活它，使用它，然后切换回标准键盘。

开发者指南，请参阅 [Custom Keyboard](#) 以及 [App Extension Programming Guide](#)。

自定义输入视图

自定义输入视图用自定义键盘代替标准键盘，但只在 App 中使用，而不是在整个系统中使用。使用自定义输入视图提供一种独特而有效的数据输入方法。例如，在编辑电子表格时实现了数字的自定义输入视图。



使功能更明显：自定义输入视图应该和整个 App 融为一体。数据条目应该清晰直观，不需要额外的指令。

在打字时播放标准键盘敲击的声音：用户在敲击键盘上的键位时，可以提供声音反馈。在你的输入页面中点击键盘也应该产生这种声音。注意，这个声音只能在可见的自定义输入视图中使用，而且人们可以在设置>声音中可以禁用声音。开发者指南，请参阅 [UIDevice](#) 的 `playInputClick` 方法。

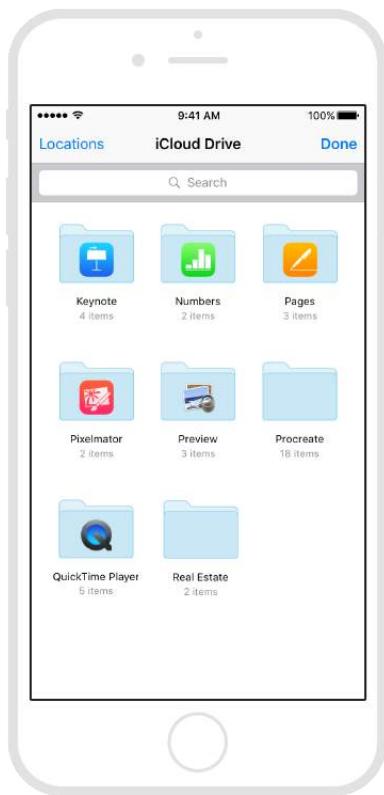
有需要的画可以提供一个输入附件视图：一些 App 实现了额外的自定义输入的附件视图，它出现在键盘上方。在数字中，附件视图帮助人们可以标准化输入或自定义计算。



开发者指南，请参阅在 [Text Programming Guide for iOS](#) 中可以看到 [Custom Data Input Views](#)。

10.2 文件应用

文件应用实现了一个自定义的入口，用于导入、导出、打开或将文档在 App 中移动。当文件加载时，它的入口显示在导航栏的模态视图中。



当用户打开或导入文件时，只显示与上下文相关的文档和信息：当有人使用你的扩展文件来打开或导入文档时，只显示适合当前上下文的文档。例如，如果一个 PDF App 加载你的扩展文件时，则只列出可打开或导入的 PDF 文件。一定要列出其他可能有用的信息，比如修改日期、大小，以及文档是本地的还是远程的。

当导出和移动文档时，让用户选好一个目的地：除非 App 将文档存储在单个目录中，否则让用户导航到目录层中的特定目标。考虑提供一种方法来添加新的子目录。

不要提供自定义导航栏：你的扩展文件在一个已经包含导航条的模态视图中。提供第二个导航栏是令人困惑的，并占用了你的内容空间。

开发者指南，请参阅 [Document Provider](#) 和 [App Extension Programming Guide](#)。

10.3 主屏幕快捷操作

主屏幕快捷操作是一种便捷的行为，可以通过在主屏幕上使用 3D Touch 来执行有用的、特定于 App 操作。用你的手指在 App 图标上施加一点压力——比你在普通点击和长按上使用的要多——然后看到可用的快捷行动列表。点击一条列表激活它。例如，邮件的快捷操作就包括：你可以直接进入收件箱或 VIP 邮箱、发起搜索、创建一条新的信息。主屏幕快速操作都有一个标题，左边或右边的图标（取决于你的 App 在主屏幕上的位置），以及一个可选的副标题。标题和副标题总是左对齐。你的 App 甚至可以在新信息可用时动态更新它。例如，消息的快捷操作就有打开最近的消息。



提示：一些App在他们的快捷操作列表上显示一个小部件。例如，股票快捷操作列表显示在当前股票值的小部件之下。考虑实现一个小部件，以提供更有吸引力的体验。对于小部件设计指导方针，请参阅 [Home Screen Quick Action Widgets](#)。

为用户最关注的、重要的任务创建快捷操作：例如，地图允许人们在他们当前的位置进行附近搜索，或者在不打开地图 App 的情况下就可以回家。每个 App 都应该至少能提供一个有用的快捷操作；快捷操作总共可以有4个。

避免把快捷操作当成导航：如果在你的 App 中访问重要的区域是困难的或耗时的，首先要优化你的导航，让用户可以更快访问。接下来，使用快捷操作，让你能够完成有用的创造性任务。

避免对快捷操作做未知的改变：动态快捷操作是保持操作相关性的好方法。

例如，根据 App 中的位置、最近的活动或者设置的更改，实时更新和修改快捷操作。然而，快捷操作不应以不可预测或无序的方式改变。

为每一个快捷操作提供一个简洁的标题：操作的标题要可以传达行动的结果；例如，“回家”，“创建新的联系人”和“新消息”。如果你需要提供更多的上下文信息，也可以提供一个副标题。邮件使用副标题显示收件箱和 VIP 文件夹中是否有未读邮件。不要在标题或副标题中包含你的 App 名称或任何外部信息，保持文本短以避免被缩略，并在编写时考虑文本本地化。

不要使用快捷操作来通知用户：人们希望通过其他方式收到 App 的通知。

参阅 [Notifications](#)。

为每一个快捷操作提供一个可识别的图标：尽量使用熟悉的系统图标。跳转 [Quick Action Icons](#)。如果你想设计自己的图标，下载快捷操作图标模板（[Quick Action Icon Template](#)），并遵循定制图标的指导方针（[Custom Icons](#)）。

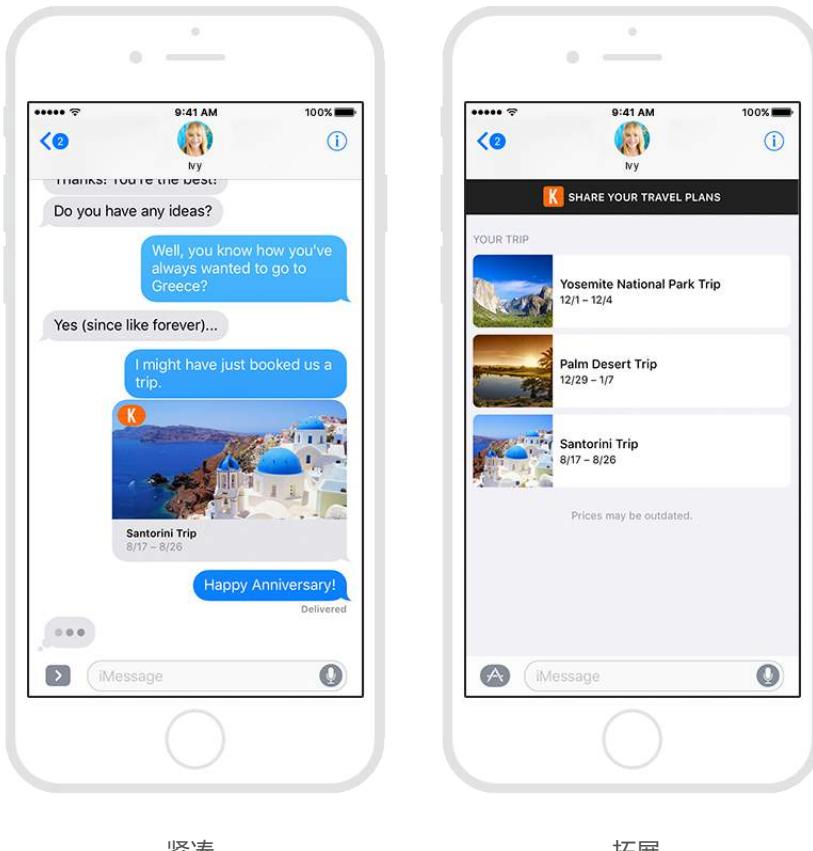
不要用表情符号代替图标：Emojis 不能正确地与右对齐的文本对齐。另外，emojis 是彩色的，而快速的动作图标是单色的。

10.4 消息

[iMessage apps](#) 和贴纸包（[Sticker packs](#)）是一种信息扩展，可以让人们与朋友分享内容，交流表达方式，分享协作体验。

iMessage App

iMessage App在消息会话的过程中提供交互式体验。iMessage App 可以让人们分享数据和多媒体文件，在共享任务上进行协作，并互相玩游戏。



紧凑

拓展

设计一个直观的界面：如果你的 iMessage App 提供静态内容，比如文本、照片或视频，那么人们就可以很容易地浏览和在对话中选择要加入的信息。如果你的App是交互式的，比如多人游戏或多人点餐，确保它的功能是有用的和可以理解的。

提供集中的内容：为了便于使用，每个消息传递得应该只有一个焦点。例如，不要试图设计一个结合了贴纸和拼车的 App。

提供一个有趣的，合作的体验：iMessage App通常在两个或更多的人之间快速、非正式的对话中使用。好好利用这个环境，通过分享、编辑或增加内

容来鼓励用户参与。如果他们都在使用这个 App，那么消息就会不断地被更新。

突出有趣的 iOS App 的内容：通过在消息中显示它的内容来扩展 iOS 应用的功能。想想人们可能想要分享什么样的信息，以及他们如何在有趣的对话中与你的应用互动。

要注意内容的放置，以避免被隐藏：你的 App 的内容是在带有圆角的信息气泡中显示的，所以不要在角落里放置重要的信息。

区分压缩和扩展视图：你的 App 出现在一个紧凑的视图下面。这一视图应该能直观地将你的 App 与其他 App 区分开来，并提供对经常使用的功能的跳转访问。人们也可以在扩展的视图中打开你的 App，以访问高级功能，或者同时看到更多的选项。注意，在紧凑视图中不允许水平滚动。

只允许在扩展视图中进行文本编辑：紧凑的视图与键盘的高度大致相同。为了确保用户能够看到他们编辑的内容，只允许在扩展视图中输入文本。

贴纸

贴纸提供了一种有趣的、吸引人的方式，让人们在没有打字或使用表情符号的对话中表达自己。贴纸是一种图像或动画，可以发送或放置在信息、照片和其他贴纸上，以增加重点和交流情感。



紧凑

拓展

设计表现力：人们使用贴纸来直观地传达情绪和反应。努力传达与人在情感层面上的联系。考虑将图像、词汇和短语添加到对话中添加新的维度。

让全球的用户都能用：消息传递是一种通用的通信形式。要瞄准具有国际吸引力的贴纸。

使用描述性的图像名称或提供替代的文本标签：虽然它们在屏幕上看不到，但图像名称和替代文本标签让配音人员可以通过声音来描述贴纸，让视觉障碍的人更容易导航。

通过动画添加活力：虽然贴纸可以是静态图像，但动画贴纸是在对话中传递能量的好方法。一定要使用足够高的帧率来保持动画流畅性。

测试位置的可能性：用户可以在对话的部分内容上缩放、旋转和放置贴纸。

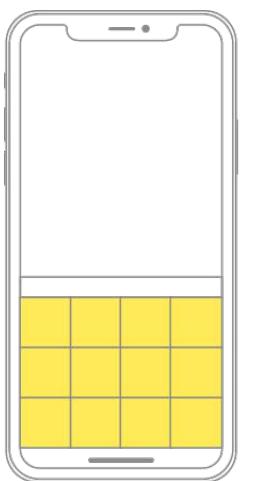
确保你的贴纸在不同的颜色和图像上是清晰的，不管它们的位置或大小。

考虑使用充满活力的颜色和透明度：鲜艳的色彩为谈话增添了丰富的体验。

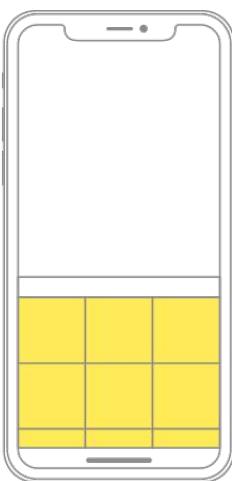
允许在信息、照片和其他贴纸上放置贴纸。

贴纸的大小

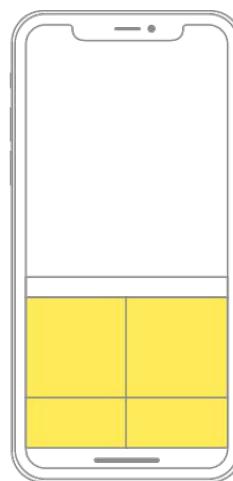
消息支持三种标签大小，基于网格的浏览器中显示。选择适合你的内容的尺寸，并将你所有的贴纸统一规格。



小型 300px × 300px



标准型 408px × 408px



大型 618px × 618px

适当大小的贴纸：尽管在必要的时候，消息会缩小过大的贴纸，但还是要提供高质量的贴纸。

注意文件大小的限制：为了提高效率，每个单独的贴纸不得超过 500KB。

请注意，Xcode 使用一个 24 位调色板保存 PNG 动画，这可能会导致比预期更大。

标签格式

消息支持以下文件格式的贴纸:

格式	推荐	支持透明	支持动画
PNG	•	• 8-bit transparency	
APNG	•	• 8-bit transparency	•
GIF		• single color transparency	•
JPEG			

App 和贴纸包图标

就像 iOS App 一样, iMessage App 和贴纸包都需要可识别的应用图标。

保持背景简单并提供一个重点: 设计一个以单一的、具有中心点的图标, 以吸引眼球。包括一个简单的背景, 它不会盖住其他图标。

保持图标是圆角的: 系统应用了自动圆角的模板。

提供不同大小的图标: 你的图标出现在 App Store、消息、通知和设置中。

为了确保你的图标在任何环境下, 任何设备上都看起来都很棒, 你可以在以下尺寸上找到匹配的尺寸:

	@2x	@3x
iMessage app icons	148px × 110px	—
	134px × 100px	—
	120px × 90px	180px × 135px

	@2x	@3x
	64px × 48px	96px × 72px
	54px × 40px	81px × 60px
Settings icons	58px × 58px	87px × 87px
App Store (prior to iOS 10)		1024px × 1024px

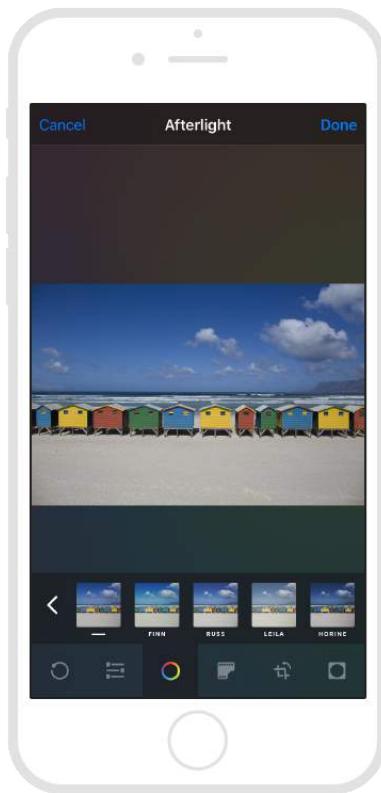
对于设计优秀的 App 图标的额外指导，其中大部分可以应用于 iMessage App 和贴纸包图标，参见 [App Icon](#)。

下载 iMessageApp 和贴纸包图标 Photoshop 模板，见 [Resources](#)。

开发者指南，请参阅 [The Messages Framework](#)。

10.5 照片编辑

照片编辑允许人们通过过滤器等工具来修改照片和视频。照片编辑会在照片 App 中保存为新的文件，并保留原始版本。



要访问照片编辑，照片必须处于编辑模式下：在编辑模式下，点击工具栏上的扩展图标显示可用的操作菜单。选择一个在导航栏的模态视图中显示扩展的接口。删除此视图确认并保存编辑，或取消它并返回到照片 App。

确认取消编辑：编辑照片或视频可能很耗时。如果有人点击取消按钮，不要立即放弃他们的更改。请他们确认他们真的想取消，并通知他们任何编辑将在取消后丢失。如果没有编辑，则没有必要显示此确认。

不要提供自定义导航栏：你的照片编辑会在一个已经包含导航条的模态视图中。提供第二个导航栏是令人困惑的，并占用了正在编辑的内容空间。

让人们预览编辑：如果用户看不到照片编辑后的样子，就不知道对它是否满意。让人们看到他们努力的结果，然后关闭照片编辑，并返回到照片 App。

使用你的 App 图标为你的照片编辑：这让人们相信，编辑实际上是由你的 App 提供的。

开发者指南, 请参阅 [Photo Editing](#) 和 [App Extension Programming Guide](#)。

10.6 共享和动作

“共享”提供了一种便捷的方式，可以通过 App、社交媒体帐户和其他服务共享当前上下文中的信息。“动作”允许人们发起专属于内容的任务，比如“添加书签”、“复制链接”或“保存图像”。人们通过点击 App 中的动作按钮来访问共享和动作的扩展视图。动作视图只显示与当前上下文相关的扩展。例如，在编辑视频时，你不会看到文本操作按钮。在活动视图中，共享功能被列与其中。



启用一个单一的、集中的任务：扩展功能并不是一个迷你应用。它执行与当

前上下文相关的小项任务。

设计熟悉的界面：对于共享功能，系统提供的组合视图是用户熟悉的，并在整个系统中提供了一致的共享体验。尽可能的使用它。对于动作扩展，包含你的 App 名称或设计的界面，让它在你 App 中看上去更自然。

简化并限制交互：最好的扩展允许人们在几个步骤中执行任务。例如，共享扩展可能会立即将图像发布到一个社交媒体帐户。请只在必要时提供共享操作。

避免在你的扩展之上放置模态视图：默认情况下，在模态视图中显示扩展。虽然警告是必要的，但要避免额外的模态视图。

使用你的主App来表示冗长操作的进展：活动视图应该在启动共享或操作之后立即消失。要在后台继续进行耗时的任务，你的主 App 应该提供一些方法来检查这些任务的状态。不要使用通知。当任务完成时，人们不希望看到通知，如果有特殊问题再通知他们。

使用动作扩展图标的模板图像：模板图像使用蒙版来创建图标。使用黑色和白色具有适当的透明度和反锯齿，并且不包括投影。模板图像应该集中在一个区域测量约 70 px×70 px。

有关额外的指导，请参见 [Activity Views](#)。开发者指南，请参阅 [Share](#) 和 [Action App Extension Programming Guide](#)。

提示：分享扩展会自动地使用你的 App 图标，逐步开启你的 App 所提供的扩展功能。

10.7 小部件

小部件是显示少量及时、有用信息或 App 特定功能的扩展。例如，新闻小部件显示了头条新闻。日历提供了两个小工具，一个显示了今天的事件，另一个显示了接下来的内容。Notes 允许你预览最近的笔记，并快速创建新的笔记、提醒、照片和绘图。小部件是可定制的，可以包含按钮、文本、自定义布局和图像等。

当你在主屏幕上使用 3D Touch 对 App 图标施加压力时，小部件出现在快捷操作列表之上。人们还将他们关心的小部件添加到搜索屏幕上，通过在主屏幕和锁屏的右滑来进行访问。你的目标应该是设计一个用户想要添加到搜索屏幕的小部件。



搜索界面的小部件



主界面中快捷操作功能里的小部件

设计很棒的授权体验：人们使用小部件来获得简洁的更新和执行非常简单的任务，所以提供适当数量的信息和交互性是很重要的。只要有可能，就提供可以单击就完成的任务。在小部件中不支持平移和滚动。

快速显示内容：人们花很少的时间查看小部件，不需要等待内容加载。在本地缓存信息，以便在获取更新时始终显示最近的信息。

提供足够的边距：避免将内容扩展到小部件的边缘。一般来说，在每个边缘和内容之间至少提供几个像素的空白。使用小部件顶部的 App 图标进行对齐指导。当内容与图标的中心对齐时会给用户提供舒适的体验。如果你的 App 提供了一个网格样式的布局，确保你在网格项目之间提供足够和相等的填充。如果可能的话，将图标和按钮的网格限制为每行4个。

适应性强：小部件的宽度因设备和方向而异。小部件显示的高度和信息取决于它是否紧缩或扩展(不是所有的小部件都支持扩展)。紧缩的小部件是大约两个半表行的高度。扩展的小部件最好不要超过屏幕的高度。快捷操作列表只显示处于紧缩状态的小部件。紧缩小部件可以显示基本的信息。扩展小部件可以显示额外的信息，以增强主信息。例如，天气小部件在紧缩时显示了当前的天气状况，但在扩展时增加了每小时的预测。

避免定制小部件的背景：系统提供的毛玻璃是为一致性和易读性而设计的。尽可能的使用它。不要使用照片作为背景，因为它可能与锁屏和主屏墙纸相冲突。

一般来说，使用黑色或深灰色的系统字体为文本：系统字体是为易读性而设计的，黑色的颜色与标准的小部件背景就很好。

让人们跳转到你的 App 去做复杂的操作：你的小部件应该独立于你的 App

运行，但是，如果人们偶尔需要做的事情，比你的小部件提供的多，那么就要让用户轻易的跳转到 App 中。不要设计一个“打开 App”的按钮，但是根本没有打开你的 App。例如，在日历小部件中，你可以点击一个事件，在日历 App 中打开它，但不要使用你的小部件打开其他 App。

给小部件起个好名字：一个 App 图标和标题出现在每个小部件的内容之上。一般来说，小部件的名称应该与 App 的名称相匹配。如果你的 App 提供了多个小工具，可以考虑使用你的 App 名称为主要的小部件命名，其他的小部件就用清晰的、简洁的名字。如果你使用自定义标题，可以考虑用你的 App 的名称命名它。例如，用于显示映射附近位置的小部件的标题为“附近的地图”，让用户相信这个小部件实际上是由附近位置的 App 提供的。

让人们知道进行身份验证可以获得额外的功能：如果你的小部件在用户登录后提供了额外的功能，请确保人们知道这一点。例如，预订 App 可能会包含一条信息，当人们没有登录的时候，上面写着“登录 App 查看你的预订”。

为快捷操作界面选择一个小部件：如果你的 App 有多个小部件，你可以选择一个在快捷操作中出现，当有人用 3D Touch 对你的 App 图标施加压力的时候打开。（例如支付宝快捷操作界面下方的付款等内容）

开发者指南，请参阅 [App Extension Programming Guide](#)。

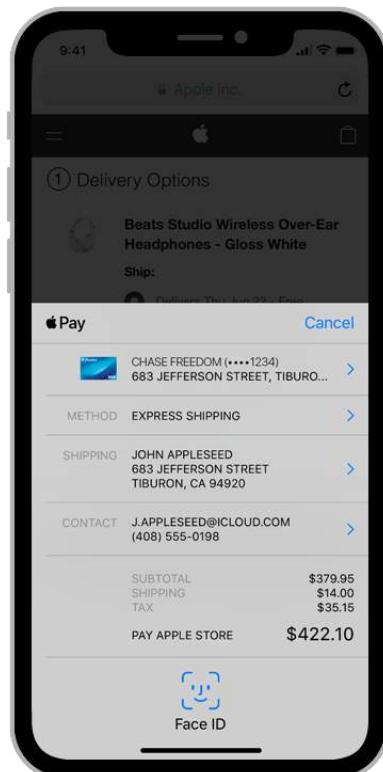
第十一章

术语

Technologies

11.1 Apple 支付

在 iOS 和 WatchOS 系统中，Apple 支付是一种安全便捷地支付实物商品、服务以及捐赠的方式。通过使用设备上的安全存储凭证，用户自主授权支付，并提供送货和联系的信息。



Apple 支付 App 会在任何支付选项显示的地方，展示 Apple 支付标识和一个支付按钮，当用户点击按钮时会显示一张支付表。在付款过程中，支付表会显示与 Apple 支付绑定的信用卡或储蓄卡，购买金额(包括税费)、送货选项和联系信息。用户可以进行必要的调整，然后支付、完成购买。

开发者指南，请参阅 [Apple Pay Programming Guide](#) 和 [PassKit > Apple Pay](#)。

网站同样可以接受 Apple Pay。对于专门针对网站的设计指导，请参阅 [Apple Pay on the Web Human Interface Guidelines](#)。

提示：能够理解 Apple 支付和 App 内购买之间的区别很重要。Apple 支付可以用来购买日用品、衣服和电器等实物商品；支付俱乐部会员、酒店预订、赛事门票等服务；以及各种捐款。而使用 App Store 内购是用来购买虚拟商品，如你 App 中的优质内容，电子读物的订阅。

请参阅 [In-App Purchase](#)。

按钮

该系统采用了几种按钮样式以供 Apple 支付 App。开发者指南，请参阅 [PKPaymentButtonStyle](#)。

Apple 支付或用 Apple 支付按钮购买

当用户想要发起一项支付活动时，点击支付按钮，就可以看到商品详情页或是购物车页面，如果已启动 Apple Pay，用户在点击支付按钮后，就应该显

示一张能协助用户完成付款过程的支付单。如果未启动 Apple Pay，在这种情况下，用户点击支付按钮就应该可以启动一个添加卡片的过程——最好展示一个“设置 Apple 支付按钮”。



设置 Apple 支付按钮

当设备支持 Apple 支付但权限没有设置好的时候，请考虑在付款页面上显示按钮。点击按钮应该可以启动添加卡片的过程。在用户完成此步骤之后，他们将会返回到付款过程完成交易。尽管其他 Apple 支付按钮在未给权限状态下进行的是相同的步骤，“设置 Apple 支付权限”的按钮应该最显眼。这个按钮也可以显示在一个像用户信息页面这种没有付款操作的界面。



用 Apple 支付按钮捐赠

经过批准的非营利组织（[Approved nonprofits](#)）可以通过这个按钮来推进捐赠项目。在运行旧系统的设备上不支持用 Apple 支付按钮捐赠，而是显示 Apple 支付按钮。



用 Apple 支付按钮付款

当用户在一家实体商店购买时，银行和信用卡发行商可以使用这个支付按钮，快速地让人们在钱包 App 中使用他们的银行卡。

Pay with Apple Pay

风格

你拥有多种 Apple 支付按钮的展现方式。

黑色：使用白色或浅色的背景能够充分地进行对比。不要使用黑色或深色的背景。



有边框的白色：使用白色或浅色背景不能充分地对比。不要使用黑色或深色的背景。



白色：使用黑色或彩色背景能够充分地对比。



大小和位置

保持最小宽度: 所有 Apple 支付按钮的最小宽度为32pt(32px @1x, 64px @2x)。



保持最小间隔: 一个 Apple 支付按钮需要的最小间隔是按钮高度的1/10。

其他内容如图形和文本，不应该覆盖这一空间。



重点突出 Apple 支付按钮: Apple 支付按钮与其他支付按钮一样大或更大。理想情况下,滚动不应该能看到 Apple Pay 按钮。

关于添加到购物车按钮，始终固定 Apple Pay 按钮位置: 将 Apple Pay 按钮放置在一个添加到购物车按钮的右方或上方。

Apple Pay 标识

在以一种类似的方式显示其他支付选项时，Apple 支付标识图形用来说明 Apple 支付是一种可使用的支付选项。在[这里](#)下载 Apple 支付标识图形及使用指南。



关于 Apple 支付的文本

你可以用纯文本来宣传 Apple 支付，并指出 Apple 支付是一种支付方式。

Apple公司的商标上用大写字母写 Apple Pay：用一个大写的A和P，其他字母用小写，来写这两个单词。只有在必须要符合一个已建立的界面排版样式时，例如在一个所有文本都是大写的 App 中，才会全部用大写形式显示 Apple 支付。请参阅 [Apple Trademark List](#)。

永远不要用 Apple 商标来代替“Apple”这个名词：在美国，Apple 支付第一次出现在正文文本中使用了已注册的商标标志(®)。当 Apple 支付在结帐过程中出现时，不要包含注册商标的标志。

	范例文本
	Purchase with Apple Pay
	Purchase with Apple Pay®
	Purchase with ApplePay
	Purchase with ⚡ Pay
	Purchase with APPLE PAY

支付文本应该和你 App 中字体大小和尺寸相一致，不要模仿Apple的排版：相反，使用与App风格一致的文本。

不要翻译 Apple Pay：使用英文的Apple商标，即使它们出现在非英语文本中。

在推广你的 App 使用 Apple Pay 时，遵循 Apple store 中的指导指南：

在为你的 App 推广 Apple 支付之前，请参考 [App Store Marketing Guidelines](#)。

在所有支持它的设备上提供 Apple Pay

在所有设备上显示 Apple Pay 按钮：如果该设备不支持 Apple Pay，不要将 Apple 支付作为支付选项。

只使用 Apple 提供的 API 来显示 Apple Pay 按钮：与按钮图形不同，API 所生成的按钮总是具有正确的外观，并自动地进行本地化。永远不要创建自定义支付按钮。

不要禁用或隐藏 Apple Pay 按钮：如果 Apple Pay 按钮还不能被使用，比如当一个产品尺寸或颜色没有被选中时，在用户点击支付按钮之后，委婉地指出问题所在。

用 Apple Pay 这个标记只是来表示 Apple Pay 是可接受的：Apple pay 这个标记并不会促成支付，不要使用它作为付款按钮或是像按钮一样放置它。

开发者指南，请参阅 [Apple Pay Programming Guide](#) 和 [PassKit > Apple Pay](#)。

简化结算流程

人们喜欢用 Apple Pay 来快速方便地购物。提供一份付款单，让他们立即授权付款并完成交易。

可以的话，默认使用 Apple pay 支付：如果用户的 Apple Pay 是打开的，并有意愿使用它的，考虑将 Apple Pay 按钮作为第一个或唯一的支付选项，显示它比其他选项更大，或者使用一条线将它与其他选项区分开来。

在产品细节页面上，用 Apple Pay 按钮加速单个商品的购买：除了提供购物车外，还可以考虑在产品细节页面上放置 Apple Pay 按钮，这样用户就可以快速购买一个单独的商品。以这种方式启动的购买应该只针对一个单独的项，并且应该排除已经存在于用户购物车中的任何项。如果用户的购物车包含从单个产品页面直接购买的商品，则在购买完成后从购物车中删除该商品。

通过快速结帐，加快多个商品的购买：考虑提供快速显示支付表单的快速结账功能，允许用户使用单一的发货方式和目的地快速购买多个商品。

确保一个顺利的结帐过程：收集必要的信息，比如颜色和尺寸选项，在用户到达 Apple Pay 按钮之前，如果需要额外的信息，可能是因为用户忘记了选择一个合适的选项来指出问题并帮助用户纠正错误。使用高亮显示或警告文本来提示丢失的信息，并自动导航到有问题的字段，这样用户就可以快速更正并完成他们的购买。

在结帐前收集可选的信息：没有办法在付款单上输入数据，所以要收集任何可选的信息，比如优惠码、赎回码、礼物信息和交货指令。

在显示付款单之前，汇总多个商品的运输速度和目的地：付款单可以让人们选择一个单一的运输速度和整个订单的目的地。如果你的客户可以根据订单选择不同的送货速度和目的地。在 Apple Pay 开始之前，收集这些细节。

最好用来自 Apple Pay 的信息：假设 Apple Pay 的信息是完整的，并且是最新的。即使你的 App 已经有了联系人、物流和支付信息，你也可以考虑在付款过程中从 Apple Pay 中获取最新的信息，以避免多余的输入操作。

显示订单确认或感谢页面：在确认支付之后，使用订单确认页面来感谢用户

购买，提供关于订单何时发货的详细信息，并说明如何检查其物流状态。

把 Apple Pay 列在确认页面是没有必要的。但是，如果你已经列出了它，在处理交易的账户的最后四位数字之后，或者作为单独的笔记，比如：“1234(Apple Pay)或者“用 Apple Pay 已支付”。

在购买前不要求创建账户：如果你想让人们注册一个账户，请他们在订单确认页面上这样做。使用付款表在付款时提供的信息，提前填写已知的用户信息。

定制支付表

你可以根据完成交易所需的信息来定制付款表的内容。

只提供必要的信息。如果支付表包含了无关的信息，人们可能会感到困惑：

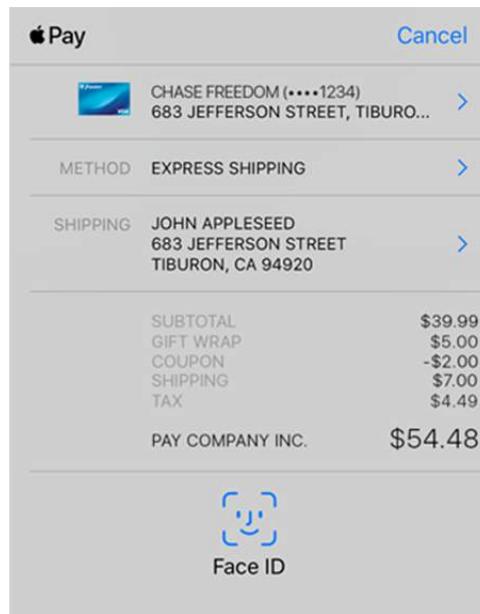
例如，如果购买是一个礼品卡，可以通过电子方式发送，那么就应该要求填写电子邮件地址，而不是发货地址。在这个场景中显示请求发送地址的请求可能会给人造成困惑，让用户以为有实体商品。

让人们在付款单上选择送货方式：在许可的范围内，可以显示清楚的描述、成本，以及每个可用选项的估计交付日期。

将解释额外的费用、折扣和待处理的费用都显示在一行列表上：一行包括一个标注和成本。不要使用行项来显示产品的列表，这些产品构成了购买的商品。

把附加的捐赠作为单独的项目：如果你的 App 允许人们添加 [approved nonprofit](#)，那么就清楚明了的单独列出捐赠的金额。有关指导，请参阅

Accepting Donations。

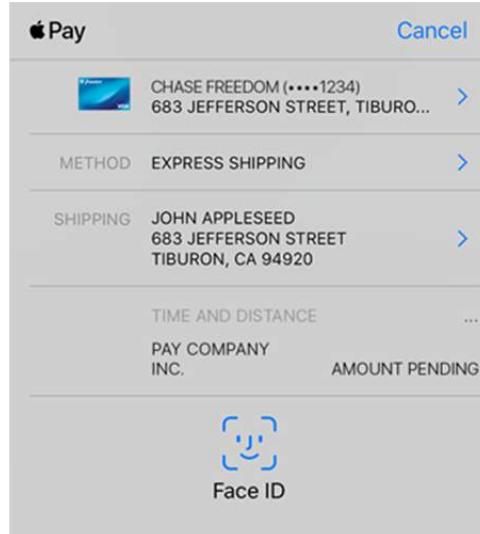


保持项目行短小精悍：让行列表的信息更具体，并且容易理解。尽量让行列表单占一行。

与“PAY”在一行的情况下，提供一个商业名称：当用户在银行或信用卡账单上查找费用时，使用相同的商业名称。这让用户放心，付款将会向正确的方向发展。如果你的App是一个中介，而不是支付的终端商家，那么你可以用以下的格式来表示：

PAY [END_MERCHANT_NAME] (VIA [YOUR_APP_NAME])

清楚地显示在支付授权后可能产生的额外费用。在一些 App 中，总成本可能在付款时不会显示：例如，一辆基于距离或时间的汽车行驶里程可能会在结账后改变。客户可能希望在产品交付后添加一个提示。在这样的情况下，在付款单上提供一个清晰的解释，以及一个被标记为“待处理金额”的数额。如果你预先授权某个金额，也要确保支付表准确地反映了这一信息。



委婉地处理付款错误：如果在付款过程中出现错误，那就帮助他们快速解决问题，从而完成交易。参阅 [Error Handling](#)。

实体店购买

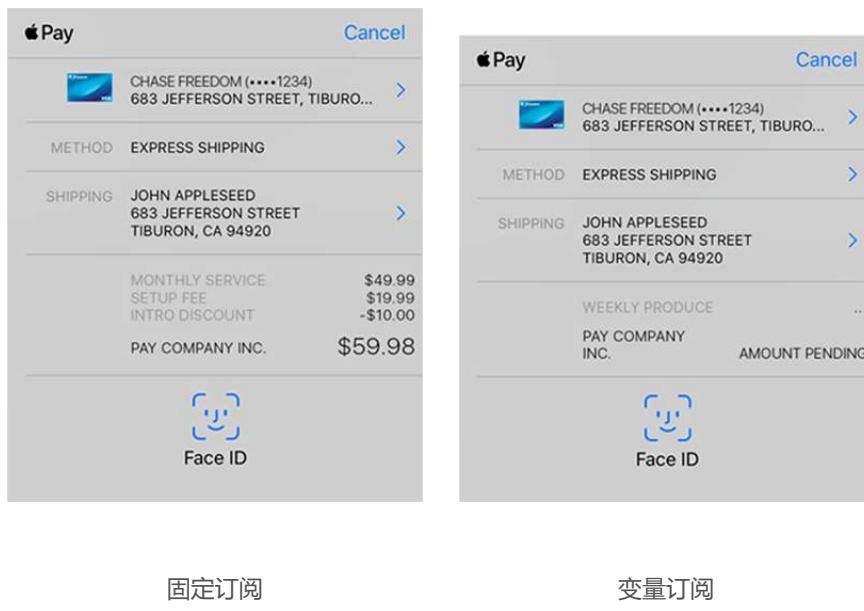
如果可以的话，可以允许人们在实体店购买商品。

在显示付款单之前，要收集一些个人信息：人们在支付单上是不能选择地点、日期或时长的，所以在按下 Apple Pay 之前，先收集这些个人信息。

在付款单的物流信息区域，显示店内的取货时间：将物流标注更改为取快递，并显示商店名称、地址和电话号码。在订单确认或感谢页面中提供这些信息。

支持订阅

你的 App 可以使用 Apple Pay 来请求用户连续订阅的授权。可能是固定的数额，例如每月的电影票，或者是一个变量，例如每周的商品订单。最初的授权还包括折扣和额外费用。



固定订阅

变量订阅

在显示支付单之前说清订阅细节：在要求用户授权订阅之前，请确保他们完全了解收费的频率和其他服务条款。

在明细页面上提供支付频率、折扣和额外的预付费用的详细信息：来提醒用户他们在授权什么。

明确目前的付款总额：确保用户在授权的时候知道确切的账单金额。

只有在更改订阅时，导致的额外费用的情况下，才显示支付单：当用户更改订阅时，如果成本降低或是保持不变，则不需重新授权。

接受捐赠

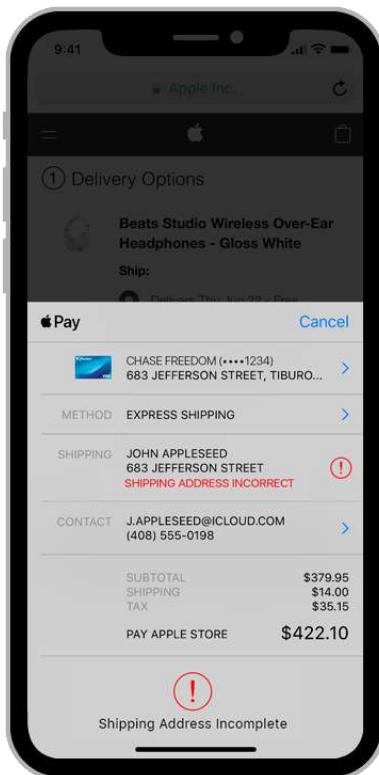
Approved nonprofits 可以使用 Apple Pay 来接受捐赠。

使用明细单来捐款：在付款表单上提醒用户他们正在捐赠，例如，捐款50美元。

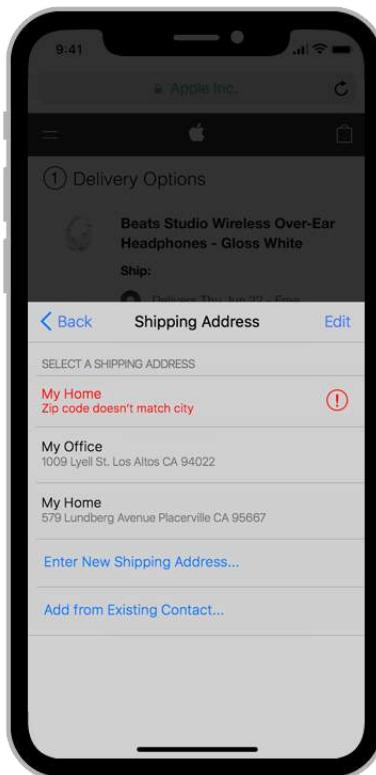
通过提供预定义的捐赠金额来简化结帐流程：你可以在捐款过程中减少捐款步骤，比如直接提供具体金额：25美元，50美元，100美元。一定要包含其他金额选项，用户可以修改捐赠金额。

数据验证

在结帐出现问题时，提供可执行的、可操作的指导，这样用户就可以快速解决问题。你的 App 可以在付款表单页面让用户更改某些信息，以及进行身份验证。检查数据输入问题，并提供清晰的消息。



付款单错误消息



自定义错误信息的视图

信息的视图

数据无效时，系统提供的错误消息将注意力转移到支付表单上的相关的红色字段。用户可以点击一个字段查看细节并解决问题。为用户点击问题字段时自动跳出问题信息。

开 发 者 指 南 , 请 参 阅 [PKPaymentAuthorizationViewControllerDelegates](#) 。

提示：出于隐私方面的考虑，在用户试图交易之前，你的 App 对数据的访问权限是有限的。在授权之前，只有卡片类型和一个修订过的送货地址是可访问的，因此向用户报告问题是非常重要的，但是在一定程度上，你的 App 也应该尝试验证可用的信息并在授权之前报告问题。

避免强制用户遵从你的业务逻辑：设计数据验证过程，智能到可以忽略无关的数据，并自动推断丢失的数据，例如，如果你的 App 需要一个五位数的邮政编码，但是用户输入的是zip+4代码，那么你就忽略错误数字，而不是请求修正。以多种格式接受状态名称(用户应该不管输入加利福尼亚州或 CA，都不会提示错误)。以多种格式处理电话号码(带横线和不带横线的，带或者不带国家代码)。

向系统提供准确的状态报告：当出现问题时，你的 App 必须准确地指出问题的类型，这样系统才能在付款表单上显示错误消息，这是通过使用适当的状态码与错误消息一起完成的。开发者指南，请参阅 [PKPaymentError](#) 。

在数据无效或格式不正确的情况下，简洁地描述问题：引用相关字段并准确地指出，如果用户输入的是无效的邮政编码，而不是显示“地址是无效的”，则显示一个特定的消息，比如“邮政编码不匹配城市”。如果发货地

址是不可用的，说明为什么“不能使用这个状态”。使用带有短句，不要使用结束标点符号。保持简短的信息。

当需要的字段为空时通知用户。请将注意力集中到一块空白的字段上。一个简短的描述性信息，如“Zin代码是必需的”

11.2 增强现实

ARKit是苹果公司的增强现实（AR）技术，提供身临其境，引人入胜的体验，将虚拟物体与真实世界无缝融合。AR应用通过设备的相机在屏幕上呈现了生动的物理世界视图。三维虚拟物体叠加在这个视图上，营造它们实际存在的感觉。用户可以重新调整设备的方向，以便从不同的角度探索对象，尽可能符合现实经验，可以使用手势和移动与对象进行交互。

设计引人入胜的体验

使用整个显示器：投入尽可能多的屏幕来查看和探索物理世界和应用中的虚拟对象。避免让控件和信息扰乱屏幕，增加沉浸式体验。

在放置逼真的物体时创造令人信服的错觉：并非所有AR体验都需要真实的虚拟对象。那些确实应该存在于物理环境中的物体，为获得最佳效果，需要设计具有逼真纹理的细腻的3D素材。然后使用ARKit提供的信息将物体定位在检测到的真实世界表面上，并适当缩放对象，在虚拟物体上提供环境光照条件，在真实世界表面上投射虚拟物体阴影，并在相机位置改变时更新视觉效果。

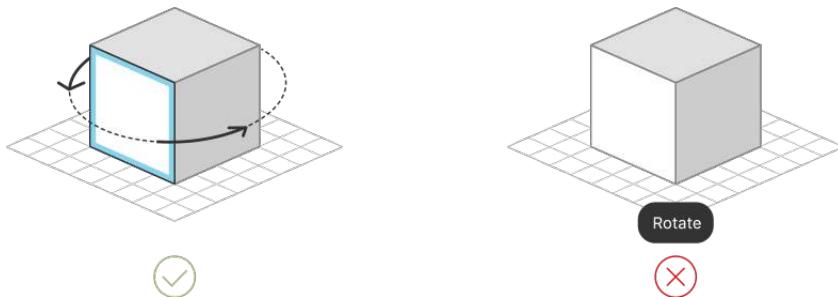
预料到人们会在不适合AR的环境中使用你的应用：人们可能会在没有足够空间移动或没有大的平坦表面区域的位置打开你的应用。所以要预先向用户明确传达我们的要求和期望，并且预料到这些具有挑战性的场景，可以考虑提供可用于不同环境的各种功能。

注意用户的舒适度：长时间地将设备保持一定的距离或角度可能会导致用户疲劳。所以要考虑人们在使用你的应用时需要怎样手持设备，并找到一种让用户拥有愉快体验的解决方法。例如，默认情况下，你可以将对象放置在距离较近的位置，这样就没必要再移动设备使它更靠近对象。当一个游戏面临短暂的停机时间时，它可以短暂地保持一定的优先级。

如果你的应用鼓励用户发生动作，请逐步介绍：当用户一进入游戏中时，没必要直接给用户来一发虚拟炮弹。先要给他们时间先适应，然后再逐步鼓励动作的发生。

注意用户的安全：如果其他人或物体在附近，大幅度的移动可能会产生危险。你需要考虑让你的应用安全运行的方法。一个游戏不鼓励大幅度和突然的动作。

使用音频和触觉反馈来增强沉浸式体验：声音效果或碰撞感是确认虚拟物体与物理表面或其他虚拟物体接触的好方法。在虚拟现实游戏中，背景音乐可以帮助用户沉浸在虚拟世界中。更多相关指南，请参阅 [Audio](#) 和 [Haptic Feedback](#)。



尽可能在上下文中提供提示：例如，在物体周围放置3D旋转指示器比叠加在图层上的文本指令更直观。但在表面检测之前，或者用户没有响应上下文提示，文本覆盖提示需要被授权。

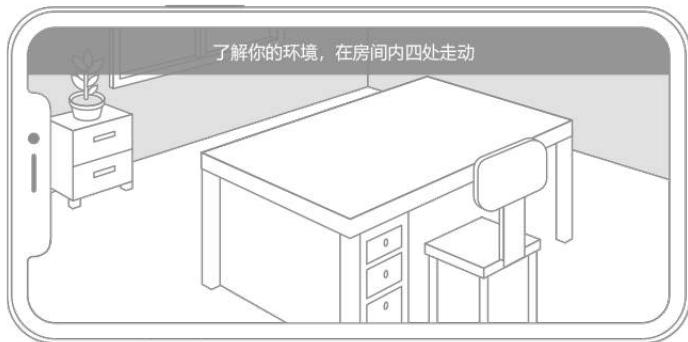
考虑引导用户找到屏外虚拟物体：有时可能很难找到位于屏幕外的物体。

如果用户看起来无法找到屏外对象，请考虑提供视觉或听觉提示。例如，如果某个物体偏向屏幕外的左侧，则可以在屏幕左侧显示一个指示器，以便用户将摄像头对准该方向。

如果你一定要显示引导文字，请使用简单易懂的术语：AR是一种先进的概念，可能会对某些用户造成困扰。为了使其便于理解，请避免提及像 ARKit、世界检测技术、和跟踪技术这样的开发者导向的技术性术语。而使用大多数人都会理解的常用术语。

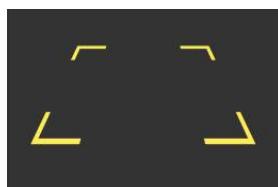
常用术语	专业术语
无法找到表面，尝试移动到侧面或重新定位手机。	无法找到平面，调整追踪。
点击一个位置以放置（要放置的对象的名称）。	点击平面来锚定一个物体。
尝试调亮灯光并四处走动。	特征不足。
缓慢移动你的手机。	检测到过度动作。

进入增强现实

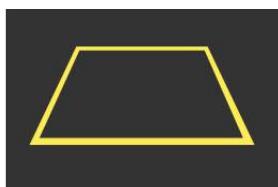


表明设备初始化和表面检测的进程并使用户参与：每次你的应用进入AR时，都会发生初始化过程，在此过程中，你的应用将评估环境并检测表面。表面检测时间可能因许多因素而异。为了减少可能的混淆，请表明你的应用正在尝试检测表面，并通过缓慢扫描周围环境来鼓励人们加速此过程。

放置虚拟物体



表面检测指示器



物体位置指示器



App特定指示器

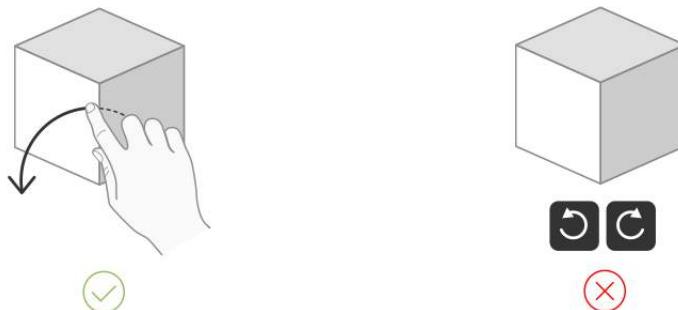
帮助人们了解何时找到表面并放置物体：视觉指示器能很好地展示表面定位模式已激活。例如，屏幕中心的梯形标线可以帮助用户推断出他们应该找到一个水平或垂直的平面。一旦定位了一个表面，指示器就应该在外观上发生

变化，以表示现在可以放置物体。如果指示器的方向与被检测表面对齐，它可以帮助用户预测被放置的物体将如何对齐。要把视觉指示器设计成像是你的 App 体验的一部分。

当用户放置物体时适当地作出响应：在表面检测过程中精度逐渐提高（在很短的时间内）。如果用户点击屏幕放置物体，请立即使用当前可用的信息进行放置。然后等到表面检测完成，再对物体的位置进行细微调整。如果物体放置在被检测表面的边界之外，请将物体轻轻推回到表面上。

不要尝试将物体与检测到的表面边缘精确对齐：在 AR 中，表面边界是近似值，它可能随着用户周围环境被进一步分析而发生改变。

用户与虚拟对象的交互

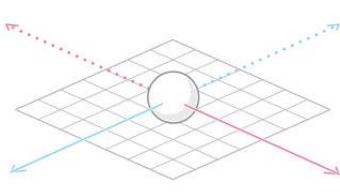


偏向直接对物体进行操作而非使用单独的控件：当用户可以触摸屏幕上的物体并直接与其进行交互时，这种方式代入感更强并且更加直观，所以不会让用户与屏幕不同部分的单独控件进行交互。但请记住，当用户在移动时，直接操作会比较困难。

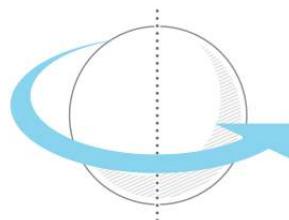
允许用户使用标准、熟悉的手势直接与虚拟物体进行交互：例如，考虑支持

用于移动对象的单指拖动手势，以及用于旋转对象的双指旋转手势。旋转通常应该发生在物体所在的表面。例如，放置在水平表面上的物体通常围绕物体的纵轴旋转。有关相关指导，请参阅 [Gestures](#)。

通常需要保持简洁的交互：触摸手势本质上是二维的，但是AR体验涉及真实世界的三个维度。考虑以下方法来简化与虚拟物体的用户交互。



放置对象的二维表面有限的移动



对象转动的有限的单轴

在合理范围内接近可交互虚拟物体时响应手势：用户难以精确地触摸小、薄或放置在一定距离的物体上的特定区域。当你的应用在可交互物体附近检测到手势时，最好假设用户想要影响该物体。

考虑被用户驱动的物体缩放是否必要：当物体（如玩具或游戏角色）不具有固有尺寸并且用户希望看到它更大或更小时，缩放通常是合适的。对于类似于一件家具这样相对于真实世界而言尺寸有限的物体，如果物品设置为精确的尺寸，则缩放无关紧要。缩放不是调整物体距离的补救措施。例如，放大物体使它看起来更加接近，但实际上物体仍然很远。

警惕潜在的手势冲突：例如，双指捏合手势与两指旋转手势非常相似。如果你实现了这样两个类似的手势，一定要测试你的应用，并确保它们被正确解读。

确保虚拟物体移动平稳：当用户调整物体大小，旋转它们或将它们移动到新

位置时，不应该出现跳动。

探索更多有魅力的互动方式：手势不是人们与AR中的虚拟物体交互的唯一方式。你的应用可以使用其他因素（如动作和接近度）将内容带入生活。例如，一个游戏角色会在用户走向它时转头看着用户。

对用户环境中的图像做出反应

你可以使用用户环境中的已知图像来触发虚拟内容，从而增强AR体验。你的应用提供一组2D参考图像，ARKit指示何时何地在用户环境中可以检测到这些图像。例如，应用可能会识别一张科幻电影海报，然后从海报中出现虚拟飞船并在四周飞行；一家零售店的应用可以通过识别放置在门两侧的海报，让商店的前门出现虚拟角色。

设计并显示参考图像以优化检测：当你提供参考图像时，可以指定你希望在用户环境中这些图像的物理尺寸。提供更精确的尺寸测量有助于ARKit更快地检测图像，并提高对其真实世界位置估算精度。具有高对比度和显著细节的平面矩形图像的检测性能和精度是最好的。避免尝试检测出现在反射或曲面的真实世界表面上的图像。

仅将检测到的图像用作显示虚拟内容的参考框架：ARKit不会跟踪被检测图像位置或方向的更改。因此，如果您尝试精确地放置虚拟内容，就像在画中将胡须放在脸上一样，内容可能不会保留在原位。

限制一次使用的参考图像的数量：当ARKit在用户环境中查找25个或更少的不同图像时，图像检测性能效果最佳。如果你的应用需要超过25个参考图像，则可以根据上下文更改一组活动参考图像。例如，博物馆指南应用可以

使用核心位置来确定用户当前所在地是博物馆的哪一部分，然后仅查看该区域中显示的图像。

有关更多开发人员指南，参见 [在AR体验中的图像识别](#)。

处理中断

避免不必要的中断AR体验：当AR未激活时，ARKit无法跟踪设备的位置和方向。避免中断的一种方法是让人们在体验中调整物体和设置。例如，如果用户将他们正在考虑购买的椅子放入起居室，并且该椅子可以采用不同的面料，则允许他们在不退出AR的情况下更换面料。

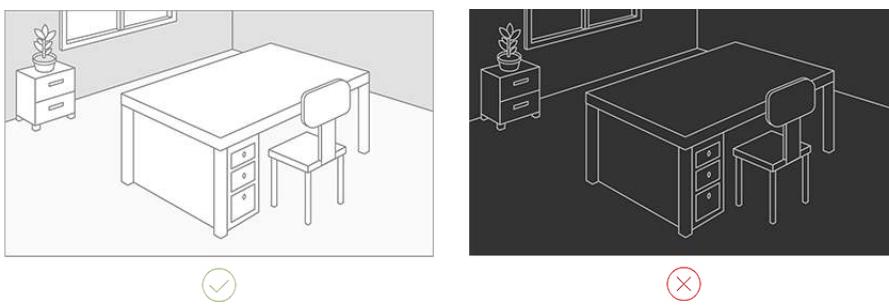
使用重定位从其他中断恢复：在中断期间，ARKit无法跟踪设备的位置和方向，例如用户暂时切换到其他应用或接听电话。中断之后，先前放置的虚拟物体可能出现在错误的真实世界位置。当你启用重定位时，ARKit会尝试恢复将这些虚拟物体回到其原始真实世界位置所需的信息。这个过程要求用户将他们的设备放置在与中断前一致的位置和方向。有关开发人员指南，请参阅 [ARSessionObserver](#)。

考虑隐藏先前放置的虚拟物体，直到重新定位完成：在重新定位期间，ARKit试图将其以前的状态与用户环境的新观察结果进行协调。在此过程完成之前，虚拟物体的位置可能有误差。

允许用户取消重定位：如果用户无法确定设备的位置和方向并定位在中断前的位置附近，则重新定位会无限期地继续下去，但不会成功。你需要引导用户成功恢复进程，或提供重置按钮和其它方式，以便用户在重定位不成功时重新启动AR体验。

处理问题

如果没有达到用户预期，允许用户重新设置体验：不要强迫用户等待条件的改善或者陷入物体如何放置的困境。给他们一种重新开始的方式，看看是否有更好的结果。



如果发生问题，建议进行修复：对用户环境和表面检测的分析可能由于各种原因而失败或花费太长时间，例如光线不足，表面过度反射，表面没有足够的细节，或者摄像头过度运动。如果你的应用收到这些问题的通知，请提供解决这些问题的建议。

问题	可行的建议
检测到的特征不足	尝试增加光线和四处走动
检测到过度运动	尝试缓慢移动你的手机
表面检测花费时间太长	尝试四处走动，增加光线，确保你的手机指向一个有足够纹理的表面

仅在有能力的设备上提供AR功能：如果你的应用的主要用途是增强现实，请仅将你的应用提供给支持ARKit的设备。如果你的应用提供AR作为辅助功能（如包含产品照片并允许在AR中查看某些产品的家具目录）；如果用户尝试在不支持的设备上进入AR，避免显示错误。如果设备不支持ARKit，首

先不要提供可选的AR功能。有关开发人员指南，请参阅 [信息属性列表项关键参考](#) 的 [UI必需的设备功能](#) 中的 `arkit key` 以及 `ARConfiguration` 的 `isSupported`。

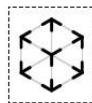
AR字形

应用可以在启动基于ARKit的体验的控件中显示AR字形。你可以在参考资料中下载该字形。



按预期使用AR字形：该字形应严格用于启动基于ARKit的体验。切勿改变字形（除调整其大小和颜色外），或将其用于其他目的，或将其与不使用ARKit创建的AR体验结合使用。

保持最小的净空间：AR字形周围需要的最小净空间数值为字形高度的10%。不要让其他元素以任何方式占用此空间或遮挡字形。



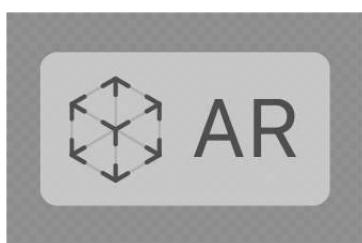
AR 徽章

包含产品集合或其他对象的应用可以使用徽章来标识特定项目，并使用ARKit 在 AR 中查看。例如，百货商店的应用可能会使用徽章来标记家具，

以便用户在购买前在家中预览家具。



按预期使用AR徽章，不要改变它们：你可以在 [Resources](#) 中下载AR徽章，可用折叠和展开形式。这些图像专门用于识别可以在AR中查看的产品或其他对象。切勿改变徽章及其颜色，或将它们用于其他目的，或将其与不使用 ARKit 创建的AR体验结合使用。



AR标志



只有象形的AR标志

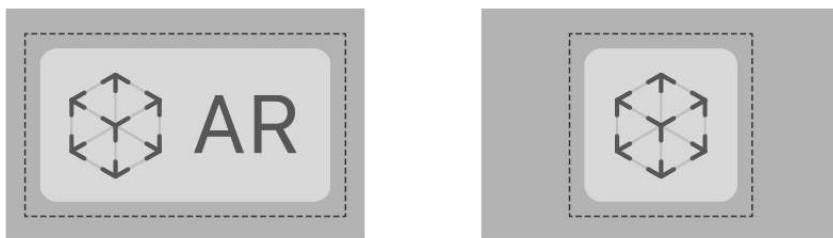
AR徽章比字形徽章更可取。一般情况下，如果空间有限且不能容纳AR徽章，请使用带字形徽章。两个徽章都能以默认尺寸正常工作。

只有当你的应用同时包含可以在AR中查看的对象和不能查看的对象时，才使用徽章：如果你的应用中的所有对象都可以在AR中查看，则徽章是多余

且不必要的。

保持徽章位置一致和清晰：显示在对象照片的一个角落时，徽章看起来最好。 始终将其放置在同一个角落，并确保它足够大以便清晰可见（但不会太大以至于遮挡照片中的重要细节）。

保持最小的清晰空间：AR徽章周围所需的最小空间为徽章高度的10%。 其他元素不应侵占此空间或以任何方式遮挡徽章。



了解更多

有关开发人员指南，参见 [ARKit](#)。

11.3 GameKit

GameKit 提供了用来创建优秀社交游戏的特性。游戏中心可以发布得分与成就，显示排行榜，添加好友，发现新游戏，以及更多。

不要设计一个自定义的登录界面：如果用户尚未在其设备上登录，系统会自动提示用户登录支持 GameKit 的游戏。设计一个自定义的登录界面就是重复的工作，而且会让预期看到标准界面的人感到困惑。

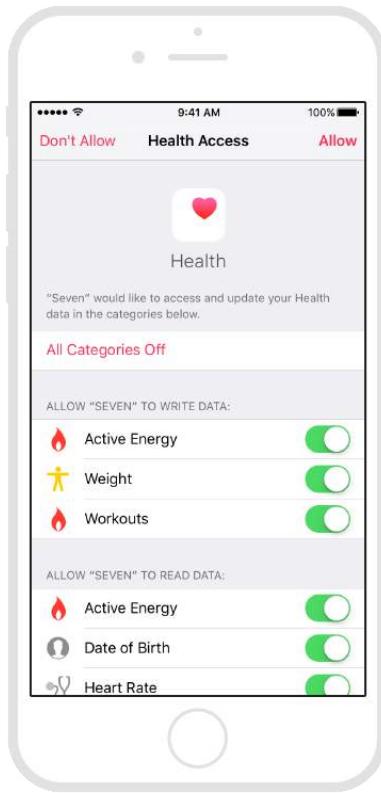
一般都要使用标准的 GameKit 接口：在极少数情况下，对于一个游戏来说包含一个GameKit 提供屏的自定义版本是有意义的，但这样做会让人感到困惑。标准的 GameKit屏旨在为所有 Apple 平台上的大型游戏社区提供熟悉的体验。

让人们关掉语音聊天：有些人可能不希望在开始游戏时自动开启语音聊天，而且每个人都喜欢在某些情况下关闭语音聊天的功能。

详情请参阅 [GameKit](#)。

11.4 HealthKit

Health App 是 iOS 中健康和健身数据的中心存储库。有了用户的许可，使用 HealthKit 构建的 App 可以与 Health App 进行互接，以访问和共享信息。例如，用户可能会允许一个营养 App 检索体重和活动数据，这样 App 可以定义卡路里消耗目标和制定饮食建议。营养 App 还可以将实际消耗的卡路里量发送给 Health App，包括在全球发展指标中，并与其他 App 共享。



请求获得健康数据的有效理由：HealthKit 是被设计用于健康和健身的 App。人们不会信任那些请求访问不必要或看似无关的私人健康数据的 App

仅当需要时请求访问健康数据：例如，当你填写一个减肥清单时，请求访问体重信息是有意义的，而不是在App启动后立即请求。请求情景中的健康数据有助于传达App的意图。

使用标准页面来表明意图：当被要求获取健康数据时，人们期望看到系统提供的许可页面。给这个屏幕添加简短但富有描述性的信息，来表明为什么需要这些信息，以及授予访问权限的益处。避免直接复制系统许可页面的内容

仅通过系统的隐私设置来管理健康数据共享：人们希望在 “>隐私” 中设置是否允许健康信息的访问。不要在 App 中添加影响健康数据流程的多余页面，这样做会导致混乱和低效。

不要使用 Health App 图标、图片或截屏：和所有的 Apple 图片一样，这

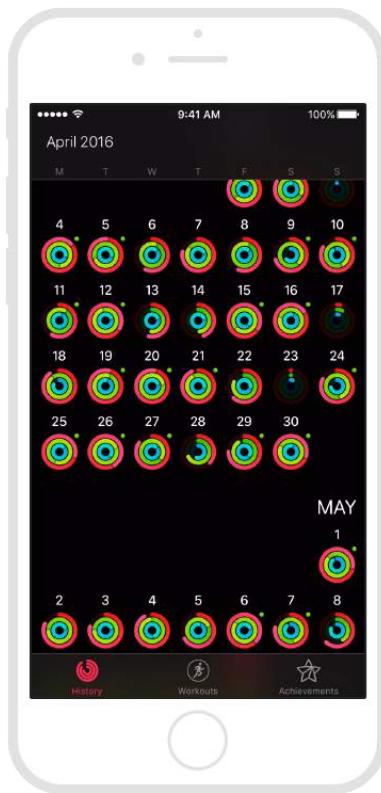
些设计都是有版权的，并且不应该出现在你的 App 或是营销活动中。你可以在 App 中显示运动环元素来展示行动，锻炼与站立的进度。参阅 [Activity Rings](#)。

不要使用“HealthKit”这个术语：HealthKit 是用于获取 Health App 中存储数据的开发框架。如果你需要解释App如何使用这些数据，请使用 Health App 这个说法。例如,你可能会说你的 App “向 Health App 存储数据”或“用 Health App 的数据。”

详情请参阅 [HealthKit](#)。

运动环

App 可以通过显示运动环来增强健康功能。运动环显示身体的移动、锻炼和站立方面的进程。这个元素包含三个环，颜色、含义和 Activity App 中提供的相匹配。



运动环只用于移动、锻炼和站立的信息：运动环旨在这些特定领域呈现一致的信息。不要试图为了其他目的去复制或修改运动环。不要用运动环显示其他种类的信息。不要使用其他环状元素显示移动、锻炼和站立进程。

一个运动环只显示一个人的活动进程：不要使用运动环来显示多人的数据，并确保其显示某人的进展是明显的，比如使用标签、照片或头像。

不要将运动环用来装饰：运动环应该为人们提供信息，而不是用来点缀你的 App 的。不要在标签或背景图形中显示运动环。

不要将运动环用于推广：运动环仅用于在 App 中显示活动进程。不要将运动环用于 App 的图标或营销材料。

不要改变运动环和背景颜色：为了一致的用户体验，不管它们出现在什么情境下，运动环的视觉外观必须保持一致。不要通过使用滤镜、改变颜色或修改透明度来改变运动环的外观或背景。相反，将周边界面设计与环相融合。

例如，将环扩在一个圆中。总是将环缩放得当，这样他们就不会显得唐突或是脱离环境。

保持运动环的边距：运动环元素必须不小于环间距的最小的外边距。永远不要让其他的元素裁剪，阻碍，或侵犯边距或环本身。要在圆内显示运动环元素，调整封闭视图的角半径，而不是使用圆形的遮罩。

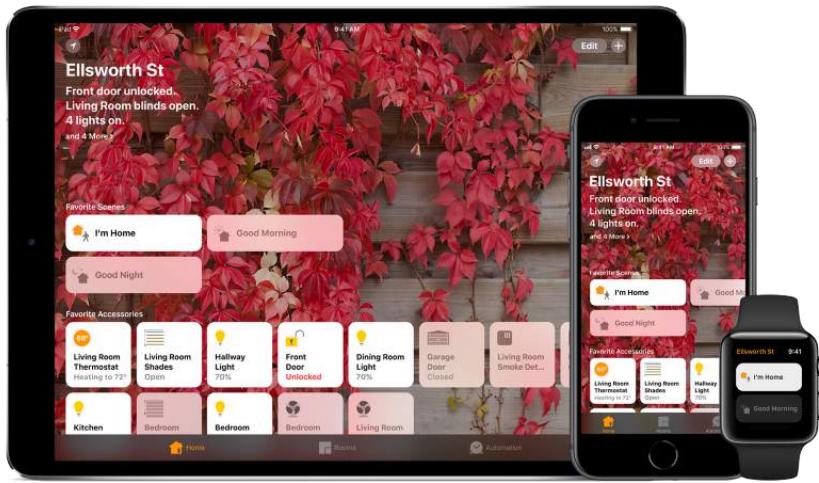
将运动环和其他环状元素区分开：混合不同的环样式会导致视觉混乱的界面。如果你必须包括其他环，请使用填充，线条，或标签将其与运动环分离开来。颜色和比例也可以帮助提供视觉分离。

仅在活动通知中提供 App 专属信息：系统已经提交移动、锻炼和站立进度更新。不要重复同样的信息，也不要在 App 的通知中显示运动环元素，可以在通知中提及活动进度，但是请确保这样做对 App 来说是唯一途径，并且不会复制系统提供的相同信息。

开发者指南，请参阅 [HKActivityRingView](#)。

11.5 HomeKit

HomeKit 允许用户使用 Siri 或是 the Home App、iPhone、iPad 和 Apple Watch 来安全地控制家中的连接配件。在 iOS 中，Home App 还可以让用户管理和安装配件。你的 iOS、tvOS 或 watchOS App 也可以与 HomeKit 互相协调，以提供自定义或品牌家庭的自动化体验。



HomeKit 术语

家庭自动化这种词语可能是吓人的。为了让它更容易亲近，所有的 HomeKit App 都应该使用人们理解的友好的语言。

房屋

在 HomeKit 中，home 代表实体房间、办公室或与用户相关的其他位置。



房间

A room 代表一个家中的实体房间。房间没有具体的大小或位置之类的特征。它们只是名字，比如卧室或办公室。房间可以根据他们在家里的实际位置来组织和控制。房间还能发出像“Siri，打开卧室灯”这样的语音指令。



区域

一个区域代表一个家庭的特定区域，如楼上或楼下。区域允许对房间进行分组，组织和控制，通常彼此接近。区域还允许语音指令如“Siri，打开楼上所有的灯”。



配件，服务，特性和动作

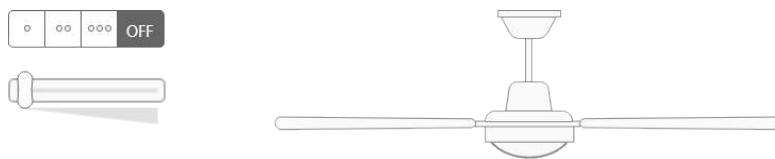
配件是一个物理的、连接的 HomeKit 物件，如吊扇、灯或照相机。人们在他们的家里给房间添加配件。附件与 iCloud 通信或服务器接收指令，响应

这些指令，并提供状态更新。这种通信直接作为一个集线器通过苹果电视或 iPad 等 iOS 设备在家中运行。

配件的可控特性，如所连接上的灯，被称为服务：有些配件提供多种服务。例如，连接的车库门可以让你单独控制灯和门，或者连接的出口可以让你分别控制顶部的出口和底部的出口。在 App 中实际上并不使用“服务”这个词。相反，他们使用描述服务的术语，如厨房灯和厨房风扇。

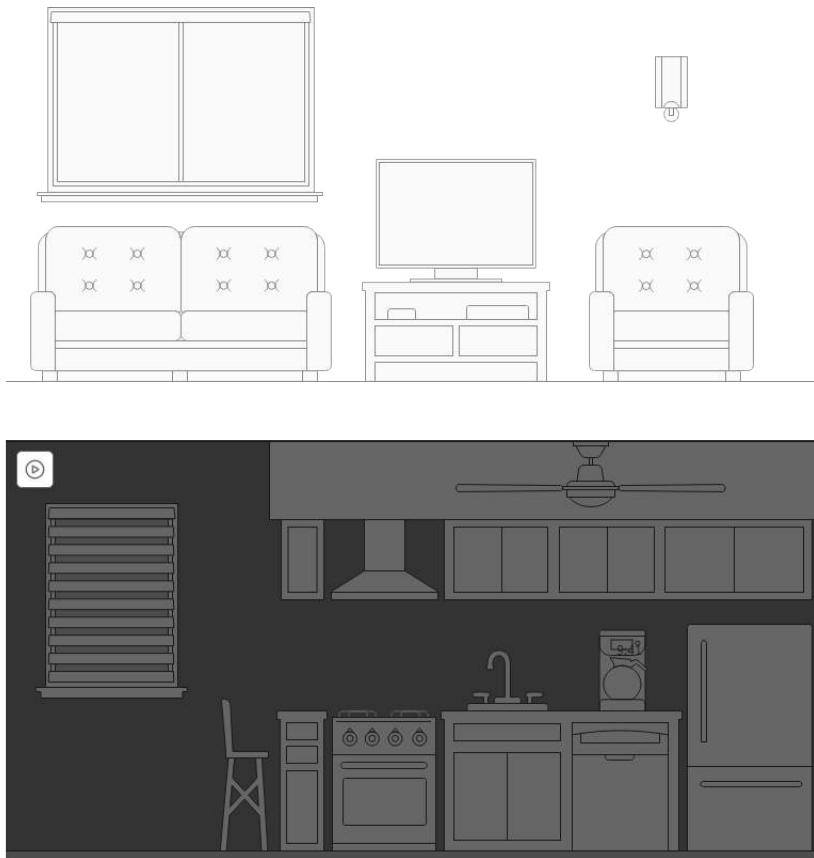
特性是服务的可控制属性：例如，在吊扇上，风扇服务可能有速度(关闭、低、中、高)的特性，而光服务可能具有亮度特性。应用程序实际上并没有使用“特性”这个词。相反，他们使用描述服务的术语，比如速度和亮度。

行动是改变服务特性的行为：例如，调整风扇的速度或亮度。动作是通过用户启动的并且通过自动化。



场景

场景是一组用于在一个或多个附件中控制一个或多个服务的操作。例如，有人可能会创造一个电影的时间场景，让客厅里的灯光变暗，把灯光调暗，或者是一个早上的场景，打开灯，调暗色调，然后开始在厨房里煮咖啡。



自动化

自动化会导致配件在某些情况下做出反应，例如，当你的位置发生变化时，某一特定的时间发生了，另一个配件打开或关闭，或者传感器检测到一些东西。例如，可以使用自动化，在日落时打开你的房子灯，或者当你拉上车窗。





提供一个舒适的体验

通过 HomeKit，App可以帮助人们：

- 设置和管理家庭、房间、区域和场景
- 查找、添加、删除和与配件交互
- 给信任的家人和朋友获得控制配件的权限
- 使用 Siri 通过语音指令控制配件
- 检查配件的状态
- 自动化常见的配件行为和设置

简化配件的设置和交互

让配件设置尽可能快速、直接和自动化：理想情况下，App自动检测到新的配件，并把它们放在显眼的位置。不要强迫人们在添加附件之前创建一个帐号。

在设置之后提示用户将配件分配给房间：配置新配件后，立即提醒用户将配件添加到一个房间。如果家庭或房间还没被创建，提供创建的方法。如果家里只有一个房间，建议在房间里添加配件，同时提供方法来增加另一个房间

确保配件容易识别：在安装过程中，当用户与配件交互时，以及在之后的调整设置时，配件应该是可识别的。例如，如果一个起居室里有多盏灯，那就提供区分每个灯的方法。在这种情境中，你可以提供一个物理闪烁灯的控件来识别它并且让人们给每个灯的服务取一个独特的名字，如台灯或吊灯。

提供多种方法来定位配件：人们想快速的与配件交互，比如在走回家的时候解锁家门。提供通过名称、种类或者在家中的位置来过滤、搜索配件的方法。考虑提供最喜欢的配件服务的面板。

准确报告配件状态，帮助用户解决问题：如果不能连接到配件，就不要猜测它的状态。例如，如果一个锁连不上，不要基于上次状态说它锁上了或是没锁。相反，表明这锁当前无法连接，提出可能的原因，并提出解决问题的建议。

支持免提交互

声控和自动交互使控制配件变得容易。

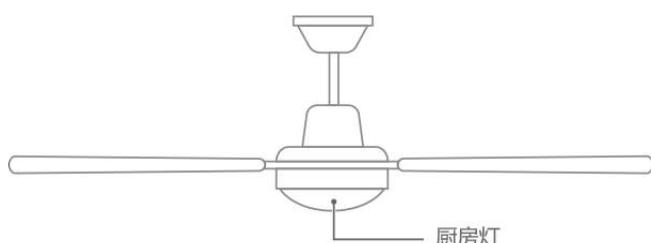
在安装过程中，引导用户使用Siri能理解的命令：Siri 通过一个简单的陈述就可以很容易地执行复杂的操作。Siri 能自动识别 HomeKit 中的家庭、房间、区域、服务和场景的名称。在整个 App 的适当时候，显示与 Siri 一起使用的短语示例，比如“你可以说” Siri，把房子设置成电影模式。“以激活这个配置。”

当有多个家庭的时候，设立一个主要的家庭：Siri 需要知道哪个家是当说“打开我的客厅灯”的目标。“如果只有一个家，那么这个家就成了所有命令的目标。”如果一个人有多套房子，而没有配置主家，那么当前的房子，就是用户此刻所在的那一个。

使自动化配置直观：HomeKit 配件可以基于条件关系自动控制，如时间、位置和其他配件的行为。例如，当车库门打开时，厨房的灯可以设置为在车库门打开时和日落后开。设置这样的条件关系会让人感到困惑，所以使这个过程尽可能简单。设计清晰的界面，并使用可以反映人们说话方式的说法。

让你的App平易近人

在你的 App 中使用友好的对话术语：避免使用那些可能引起混乱或沮丧的缩写和术语。使用标准的 HomeKit 术语(家庭、房间、区域、配件、自动化)来提供一致的和可接近的体验。



在涉及服务和特性时使用日常用语：对于服务，使用厨房灯和厨房风扇等术语。对于特性，使用像亮度和速度这样的术语。

✓	✓	✗	✗
Kitchen Light	Kitchen Fan	Fan 2 - Ceiling Light Service	Fan 2 - Ceiling Fan Service

使用适当的调控让用户改变特性：例如，使用开关，让人们打开或关闭服务

提供良好的默认值：每个用户的家庭设置都是不同的。在 home 配置期间，提供广泛适用的通用默认值。例如，你的 App 可能假设所有用户都有一个他们想要命名的主家庭。默认的名字应该是简洁的，众所周知的单词或短语

允许自定义：确保家庭、房间、区域、配件、服务和自动化名称是可定义的。例如，有人可能想要重命名一个连接的插座或切换到台灯，如果这是插座或开关控件的话。切勿对服务名称使用公司名称或型号。

处理名称冲突：Siri 要求家庭、房间、区域、服务和场景名称都是独一无二的。当出现名称冲突时，清晰地解释这个问题，并提供有意义的容易记住的替代方案。

配件注意事项

在设计 HomeKit App 时，考虑特定配件的用户体验。



HomeKit 相机

HomeKit App 可以显示静态图像或来自HomeKit IP 相机的视频流。

不要遮住相机图像：用一些有用的功能来补充相机的内容是很好的，比如提醒人们注意潜在的有趣活动。然而，要避免用其他内容覆盖相机的图像内容

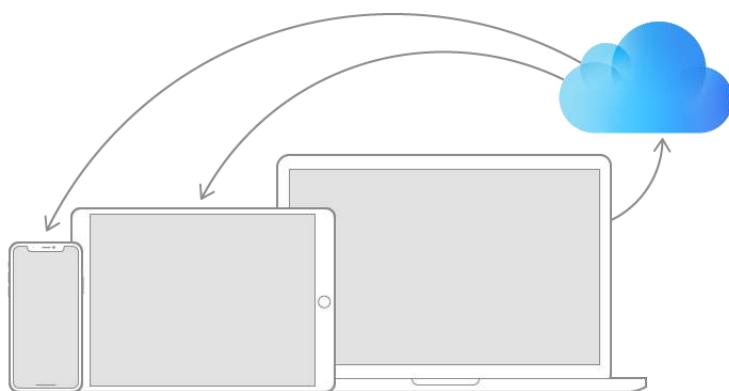
只有在相机支持双向音频的情况下，才能显示麦克风按钮：一个不能使用的麦克风按钮是浪费空间的并且会让用户感到困惑。

了解更多：

开发者指南，请参阅 [HomeKit Developer Guide](#) 和 [HomeKit](#)。如果你是 MFi 的接受方，请访问 [MFi portal](#)。

11.6 iCloud

iCloud 是一项服务，可以让用户无需手动同步即可从任何设备无缝地访问他们所关心的内容——照片、视频、文档等。iCloud 的一个基本特性是通用。用户不需要知道内容当前保存在哪里，可以假设访问的总是最新版本。



用 iCloud 轻松使用 App：用户在设置中启用 iCloud，并期望 App 自动使用。如果你认为用户想要选择在 App 中使用 iCloud，那么请在 App 第一

次打开时显示一个简单的选项，以便为所有数据提供是否使用 iCloud 的选择。

避免询问在 iCloud 中是否保留文档：大多数人希望他们所有的内容在 iCloud 中可用，也不想管理个人文档的存储。考虑 App 如何处理和公开内容，并尝试自动执行更多的文件管理任务。

尽可能保持内容最新：在支持 iCloud 的应用程序中，让用户总是能访问最新的内容。但是，你需要在设备存储和带宽限制方面平衡这一体验。如果 App 使用的是非常大的文档，那么最好让用户掌握下载更新内容的时间。如果 App 恰好属于这个类别，请设计一种方法来表示 iCloud 中提供了更新版本的文档。文档更新时，如果下载时间超过几秒钟，请提供下载反馈。

尊重 iCloud 存储空间：iCloud 是人们付费使用的有限资源。使用 iCloud 来存储人们创建和了解的信息，避免将其用于 App 资源或可重新生成的内容。即使 App 没有使用 iCloud 支持，记住 iCloud 备份包括每个 App 文档文件夹的内容。为了避免占用太多的空间，请谨慎选择你放置在“文档”文件夹中的内容。

当 iCloud 无法使用时，请确保 App 运行正常：如果有人手动禁用 iCloud 或打开飞机模式，则不需要显示警报通知 iCloud 不可用。他们已经知道了。但是，用不太引人瞩目的方式让他们知道，iCloud 访问恢复之前，它们所做的更改在其他设备上不可用仍然是很有帮助的。

在 iCloud 中保留 App 状态信息：除了存储文件和其他文件，你可以使用 iCloud 存储首选项和有关 App 状态的信息。例如，杂志 App 可能会存储最后一次查看的页面，因此当 App 在另一台设备上打开时，用户可以从他们可以

继续从上次停止的页面进行阅读。如果你使用 iCloud 存储为首选项,请确保用户想要把这些设置应用到所有的设备上。例如,一些设置在工作中比在家里更有用。

警告删除文档的后果: 当用户在启用 iCloud 的 App 中删除文档时,该文档也会从 iCloud 和所有其他设备中删除。在执行删除之前会显示警告并请求确认。

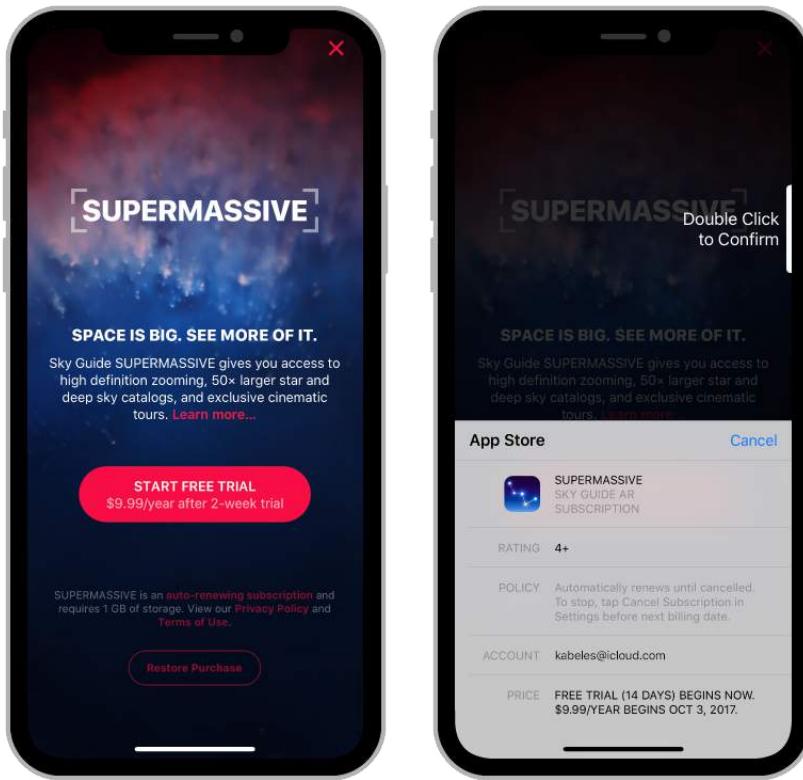
解决冲突要迅速而简单: 尽可能尝试自动检测和解决版本冲突。如果无法完成此操作,请显示一个不引人注目的通知,以便轻松地区分并在冲突的版本之间进行选择。冲突解决应该尽可能早地发生,这样时间就不会被浪费在错误的版本上了。

搜索结果中包含 iCloud 内容: 拥有 iCloud 账号的用户认为他们的内容是普遍可用的,并且希望搜索结果能反映这个特点。

开发者指南,请参阅 [iCloud Design Guide](#) 和 [CloudKit Quick Start](#)。

11.7 内购

App 内购可以让用户通过 App 内的自定义商店安全地支付数字产品。App 通常使用内购,让用户使用高级功能,订阅定期内容,并在游戏中购买新的级别或武器等虚拟物品。



设计一体化购物体验：用户不应该认为他们在浏览和购买数字产品时进入了一个不同的 App。在 App 中，展示产品并用宾至如归的方式处理交易体验

使用简单、简洁的产品名称和描述：最好能让用户能够快速浏览一组产品并找到他们感兴趣的东西。使用不会被截断或遮挡的直白的语言和标题。

使用默认的确认表：当有人启动内购时，系统会显示一个确认单，以避免意外购买。不要尝试修改此表。

应用内购买会收集你在 App Store 注册产品的付款信息。你的 App 必须真实的展示商品，解锁功能，或者下载内容所需要的任何功能。商业指南和开发者指南，请查阅 [In-App Purchase for Developers](#) 和 [In-App Purchase Programming Guide](#)。了解 App 中可以以及不可以售卖物品的详细信息，以及应用内的购买使用要求和限制，请查看 [Apple Developer Program license agreement](#)。

提示：了解内购和 Apple Pay 的区别非常重要。使用内购购买虚拟商品，比如 App 的优质内容，以及数字内容的订阅。使用 Apple Pay 销售杂货、衣服和家用电器等实物商品。还可以用 Apple Pay 支付俱乐部会员、酒店预订、赛事门票等服务。相关指导，请参阅 [Apple Pay](#)。

11.8 Live Photos

Live Photos 用声音和动作的交互体验来抓住珍爱的记忆，为传统静态照片增添活力。在启用这个功能后，相机 App 捕捉到附加的内容，包括拍照前后的音频和额外动画。只要轻按 Live Photo 就能看到它又播放起来。



对所有帧进行调整：如果 App 允许人们对 Live Photo 应用效果或调整，确保这些修改被用在了整张照片上。如果不支持这个功能，给用户转换成静态

照片的选项。

保持Live Photo内容完整：对于人们来说，在所有App中使用相同的视觉处理和交互模式，用一致的方式体验 Live photos 是很重要的。不要分解 Live Photo，或单独展示它的画面和音频。

实现优秀的照片分享体验：如果 App 支持照片共享，让用户在决定分享之前预览 Live Photos 的所有内容。总是提供将 Live Photos 作为传统照片分享的选项。

当一张 Live Photo 正在下载时或是可播放时，要明确提示：在下载过程中显示进度指示器，并在下载完成时提供一些指示。

在不支持 Live Photos 的环境中，将其显示为传统照片：不要尝试在受支持的环境中复制Live Photos体验。相反，显示传统的静态照片。

使 Live Photos 与静态照片轻松区分：识别 Live photos 的最好方式是通过移动提示。请注意，没有内置的实时照片运动效果，例如当你在照片应用的全屏浏览器中滑动照片时出现的效果。任何类似的动作效果都必须是自定义设计和执行的。在无法移动的情况下，请在照片上方显示系统提供的标志。这个标志可以显示为带阴影的叠加层，或者是不带阴影的纯色。如过 Live Photos 作为传统照片显示，也可以使用标志变体。切勿包含可以被解释为视频播放按钮的回放按钮。



保持标志位置一致：如果你要显示标志，把它放在每张照片的相同位置。通常，标志在照片的边角上看起来最好。

Live Photos 支持可在 iOS 9.1 和更高版本中使用。

11.9 ResearchKit

科研App让用户无处不在的参与重要的医学研究。ResearchKit框架提供了预先设计页面和转换，使得其易于设计和构建一个引入入胜的定制研究App。详情请查看 researchkit.org。

这些指导方针仅供参考，不构成法律建议。与律师联系，以获得有关研究App开发和任何适用法律的建议。

创建新员工培训经验

当第一次打开一个科研App时，人们会遇到一系列的页面，介绍他们参加研究，确定他们的参与资格，请求允许继续进行研究，在适当的时候，允许他们访问个人数据。这些页面在完成后通常不会重新访问，因此清晰度至关重要。

始终有正确的顺序显示引导页面。



1. 工作介绍

提供通知并采取行动的介绍：清楚地描述你的研究主题和目的。允许现有的参与者快速登录并继续进行正在进行的研究。



2. 确定资格

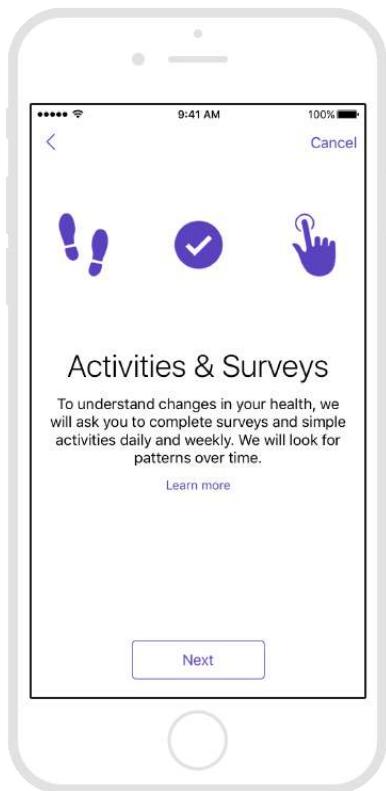
尽快确定资格：如果用户不符合这项研究，就不需要进入同意区域了。只在研究有必要时才显示资格要求。使用简单明了的语言描述需求，并使其易于输入信息。



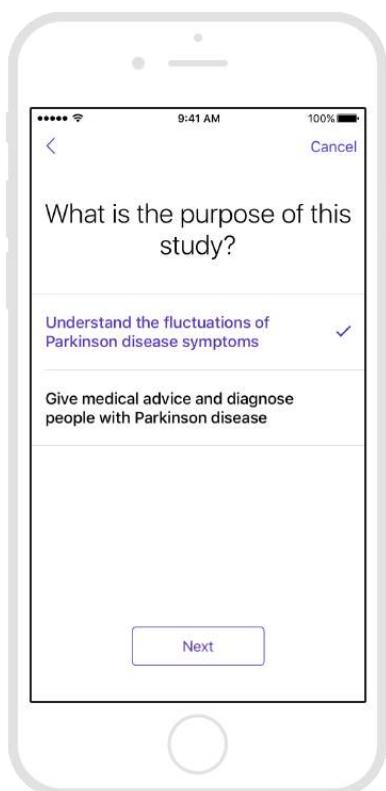
3. 获得知情同意书

在获得参与者的同意之前，确保他们了解你的研究：ResearchKit 帮助你使同意过程简洁明了，同时允许你将任何法律要求或机构审查委员会或道德审查委员会设定的要求纳入同意书。确保 App 遵守包含同意书要求的 App store 指南。通常，同意书区域解释了研究的工作原理，确保参与者理解研究和他们的责任，并得到参与者的同意。

将冗长的同意书形式分解成易于理解的几个区域：每个区域都可以涵盖研究的某一方面，例如数据收集、数据使用、潜在收益、可能的风险、时间承诺、如何退出等等。对于每个区域，使用简单明了的语言提供高度概括。必要的话，提供一个更详细的解释区域，让参与者可以通过点击 Learn More 按钮阅读。参与者在同意参加之前，应该查看完整的同意书。



如果要使其有意义，可以提供测试参与者理解的测试：在你可能提问题时，参与者将被要求获得同意。

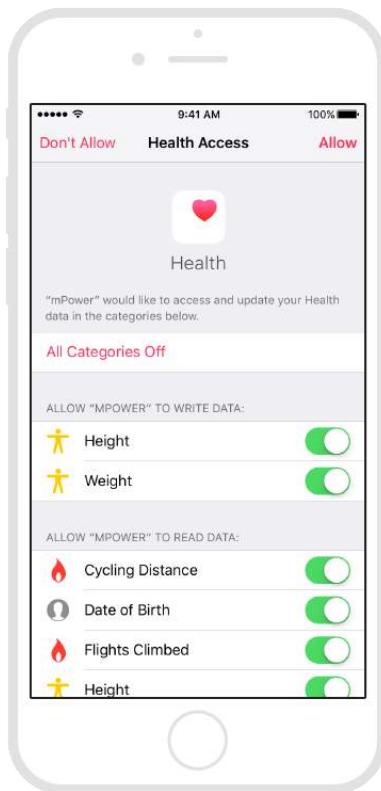


如果合适的话，获得参与者的同意书和一些联络信息：在同意参加这项研究后，参与者会收到一个确认对话框，在他们提供签名和合同详情之后。大多数研究App会给参与者用邮件发放一份PDF版本的同意书，以供保存。



4. 请求访问数据的权限

获得访问参与方设备或数据的权限，并发送通知：清楚地解释为什么你的研究App需要访问位置、健康或其他数据，而不要求对你的研究不重要的数据访问。如果你的App需要它，也要请求得到允许去发送通知到参与者的设备上。



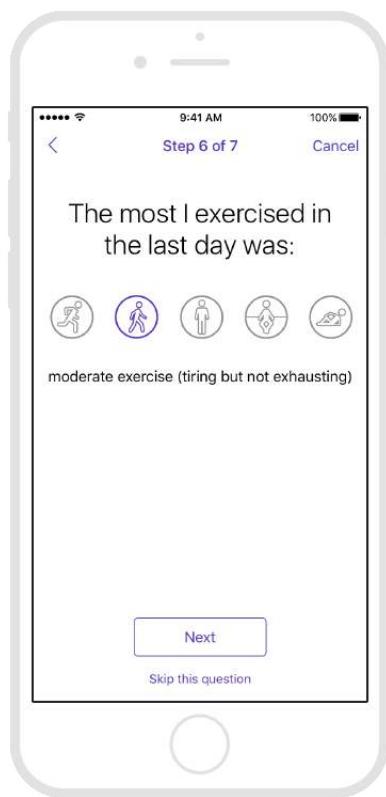
进行研究

为了从参与者那里获取输入，你的研究可能会使用调查，活动任务，或者两者兼而有之。根据你研究的架构，参与者可能和每个区域进行多次或一次的互动。

创建让参与者专注并吸引人的调研：ResearchKit提供了许多可自定义的屏幕，你可以在你的调研中使用，并且让它易于提出要求不同类型答案的问题，例如true或false、多项选择、日期和时间、滑动的刻度和开放式的文本输入。当你使用ResearchKit屏幕来设计一个调查时，遵循以下的指导方针来提供优秀的体验：

- 告诉参与者有多少问题，以及调查需要多长时间。

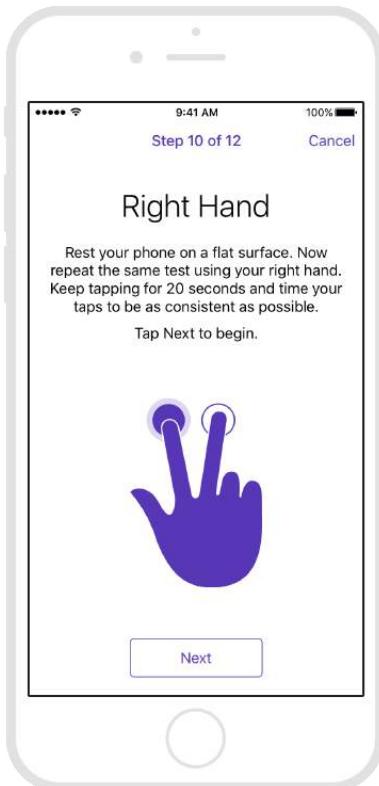
- 每个问题使用一个屏幕。
- 显示参与者的调研进程。
- 尽可能地缩短调研时间。几项短调研往往比一项长调研效果更好。
- 对于需要额外解释的问题，对问题使用标准字体和对答案使微小一点的解释性文本字体。
- 当调研完成时告诉参与者。



让活动任务变得易于理解：活动任务要求参与者参与到活动中，比如对着麦克风说话，在屏幕上点击，行走，或者进行记忆测试。遵循以下指导原则来鼓励参与者执行一项活动任务，并给予他们最大的成功机会：

- 用清晰简单的语言来描述如何完成任务。

- 解释清楚要求，例如这个任务必须在特定的时间或特定的环境下执行
- 当任务完成时确保参与者被告知

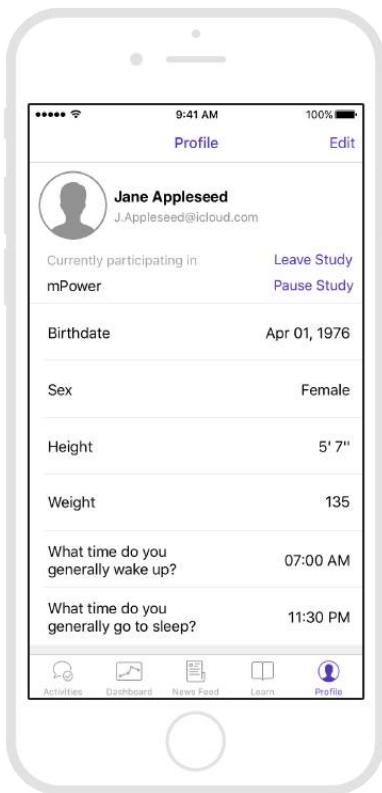


管理个人信息并提供鼓励

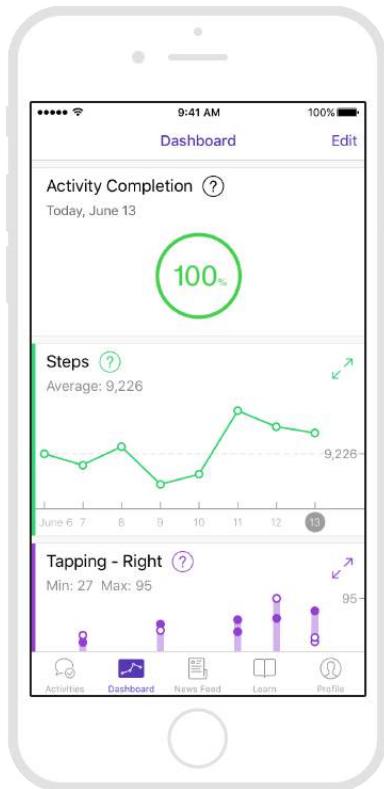
ResearchKit提供了一个档案屏，你能用它来供参与者在你的研究App中管理个人信息。同时设计一个可以激励人们的自定义屏幕，并为他们提供跟踪研究进展的方法。理想情况下，两个屏幕在任何时候都可以在你的App中访问

使用档案帮助参与者管理与你的研究有关的个人数据：档案屏可以让人们编辑在学习过程中可能改变的数据，比如体重或睡眠习惯，并提醒他们即将进行的活动。档案页面。还可以提供一种简单的方法来离开研究和查看重要信

息，例如同意文档和隐私政策。



使用图表来显示进度并激励参与者继续：如果适合你的研究，使用图表来提供鼓励的反馈，比如每天的进度，每周的评估，具体活动的结果，甚至是比較参与者与其他在研究中的参与者的結果。



11.10 社交媒体

人们期待能够快速访问他们喜欢的社交媒体账户，不管当前的情景是什么。iOS使得在App中整合的社交媒体互动变得更容易了，包括分享内容和显示活动反馈。



不要求用户登录：iOS采用单点登录的模式来访问社交媒体账户，这样你就可以获得访问账户的授权，而无需身份验证。如果用户没有登录帐户，则可以显示系统提供的界面，让他们这样做。

使共享更容易：许多人都可以通过共享扩展，方便地将内容从任何地方发布到他们喜欢的社交媒体服务上。这些扩展通过点击共享按钮(通常是在工具栏中)来访问，以显示可用服务的列表。选择一项服务，然后显示合成视图。可能的话，在显示出编辑界面之前，先用内容填充此视图。例如，一个照片App可以附加动态照片，旅游App可以添加当前位置，或者一个烹饪App能插入一个收藏喜爱菜谱的URL。

有关在App中实现社交媒体交互的信息，请参见 [Social](#)。要了解如何设计共享额外资讯，你可以在使用其他应用可以将内容发布到你的社交媒体服务，请查看 [Sharing and Actions](#)。

11.11 钱包

用户使用Wallet App来查看和管理各种通行证，现在呢是数码显示的，之前这些信息都会被印刷在小纸片或塑料上，并保存在一个实体钱包里。通行证被用于现实世界中的活动，例如登机，进入场地，或获得折扣。钱包还可以储存Apple Pay中所使用的信用卡、借记卡和存储卡。



通行证的类型

PassKit框架使 App 更容易访问、创建、发送和更新通行证。这个框架使用了几个模板，每个模板为特定类型的通行证定义布局和区域。

风格	使用
登机牌	火车票、登机牌和其他类型的交通通行证。每个通行证通常对应单一且特定的起始点和结束点的单程旅行。
优惠券	优惠券，特价优惠和其他折扣。
活动门票	参加音乐会、电影、戏剧、体育赛事或其他活动的门票。通常情况下，每个通行证都对应一个特定的事件，但是你也可以为几件事用一张通行证，比如季票。
商店卡	商店的会员卡、折扣卡、积分卡和礼品卡。如果帐户关联的商店卡有余额，那么卡里通常显示当前的余额。
通行证	任何其他类型的通行证，如健身卡或存包存衣的通行证

设计优秀的通行证

设计在所有设备上看起来都很棒的通行证：注意通行证在不同的设备上看起来不一样。例如，条形图和缩略图不会显示在Apple Watch上。不要将基本信息放在某些设备中可能无法使用的元素上。请注意，即使你不创建手表专用App，人们也可以把你的通行证添加到他们的Apple Watch上。

尽可能避免重新生成现有的实体通行证：Wallet与iOS的通用计美学相匹配，建立信任和熟悉。不要复制实体物品的外观，而是利用这个机会设计一个遵循Wallet形式和功能的简洁通行证。

避免使用纯白色背景：当通行证背景是逼真的、实心填充或是颜色强烈跳跃的图片时，它看起来是最好的。在设计背景时，确保背景不会影响内容的可读性。

使用通行证区域来显示文本：每个通行证样式定义的字段，使你的通行证统

一外观和使用辅助功能。避免在图像中嵌入文本，因为不是所有的图像都显示在所有设备上。避免使用让文本难以阅读的自定义字体。

避免使用特定于设备的语言：你无法预测用户会使用哪些设备来查看你的通行证，所以不要在特定的设备上使用可能没有意义的语言。例如，告诉人们“滑动查看”的文本在Apple Watch上显示时是没有意义的。

仔细挑选通行证正面的信息：人们希望浏览一遍，然后迅速得到他们所需要的信息，所以通行证的正面应该是整洁的，便于阅读。如果你认为人们可能需要额外的信息，把它放在通行证的背面。只是要注意，Apple Watch的通行证不包括后视图。

为公司名称使用logo文本字段：该字段中的文本在所有通行证中以一致的字体呈现。为了避免与Wallet中的其他通行证发生冲突，可以考虑在这里输入文本，而不是使用自定义字体。

适当的更新通行证：即使实体通行证通常不会改变，数字通行证也可以被更新以反映真实世界的事件。例如，航空公司的登机牌可以更新，以反映航班延误。

尽可能使用矩形条形码：正方形的条形码在两边创建空槽，可以垂直地包围上面和下面的字段。由于通行证的布局，一个长方形的条形码，比如pdf417风格的条形码通常看起来更好。

优化图像的性能：人们经常通过电子邮件或Safari接受通行证。为了尽可能快地下载文件，使用看起来很好的最小的图像文件。

开发者指南，请参阅 [Wallet Developer Guide](#) 和 [PassKit](#)。

第十二大章

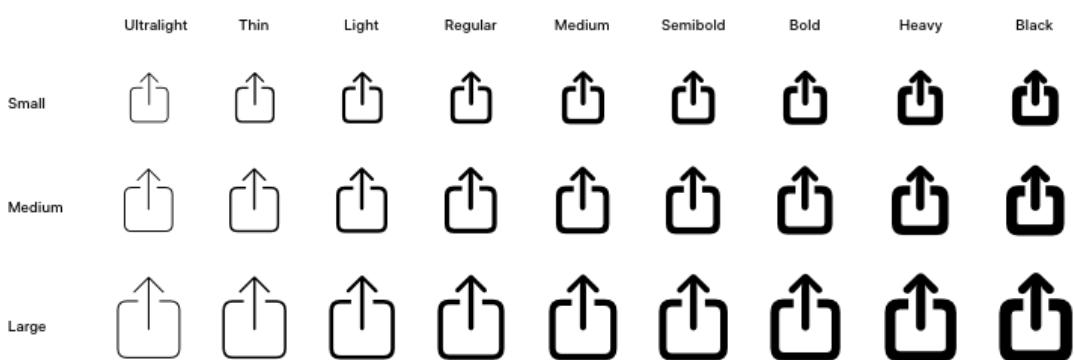
其他

One more thing

注：已独立成页的内容，或被删除的旧规范内容。

12.1 SF Symbols

SF Symbols 提供了一整套超过 1500 个高度一致的、可自定义的符号，您可以在您的 APP 中使用它们。Apple 设计的 SF Symbols 能够与 San Francisco 系统字体无缝集成，因此符号可自动与所有字重和大小的文本进行视觉上垂直对齐。SF Symbols 有多种字重和尺寸可供选择，让您的设计适应性更强。



你可以在适配了 iOS13、watchOS6、tvOS13 或对应更高版本系统的 APP 中使用 SF Symbols。下载 [SF Symbols](#) 应用可以浏览完整的 SF Symbols

系列。

SF Symbols 有九种从极细到黑色的字重，每种字重对应 SF 系统字体。这种对应关系使您可以在符号和相邻文本之间实现精确的权重匹配，同时灵活支持不同字号和不同情境。

每个符号又有三种尺寸：小、中、大。尺寸是相对于 SF 系统字体的大写字高定义的。通过指定尺寸，您可以调整符号与相邻文本之间的大小比例，而与相同字号的字体保持一致的字重。

您可以使用 SF Symbols 表示各种特定位置内容的任务和类型，例如导航栏，工具栏，标签栏，情境菜单和主屏快捷菜单。在整个应用程序的其余部分，您也可以在任何可以使用图像的地方使用它们。

创建自定义的符号

如果您想使用一些 SF Symbols 没有提供的符号，您可以自己创建。SF Symbols 应用允许您将符号导出为模板，它可以重复使用并基于矢量文件格式。要创建自定义符号，请挑选一个与您想设计的类似的符号并导出为模板，使用 Sketch 或 Illustrator 这类矢量编辑工具修改模板。像使用原始模板文件一样在应用中使用新的模板。有一些符号无法自定义，列表请参阅下方原样取用的符号。

以模板为准：创建与系统提供符号一致的自定义符号，无论是在在细节水平、视觉重量、对齐、位置还是透视方面都必须如此。力图设计一个满足以下条件的符号：

- 简洁
- 识别度高
- 无恶意
- 明确表达它所代表的行为或内容

为了适配各种文本设置，请按照应用所需的字重和比例创建自定义符号：有时候系统需要启用粗体文本和动态字体，所以您需要创建所有尺寸下的常规、中等、半粗和粗体符号。如果您的应用需要使用其他字重和尺寸，那么创建对应的符号。

不要使用苹果产品相关符号的副本：苹果产品的版权受到保护，您无法在您自定义的符号中使用它们相关的符号。**为自定义符号提供备用文本标签：**备用文本标签用户不可见，但是它们能够让 VoiceOver 播报屏幕上的内容，让视障人士更容易导航。

原样取用符号

某些符号仅可用于标注 Apple 的技术，无法导出为自定义模板。下面列出的符号应被视为系统提供的图像，如 Xcode 和 Apple SDK 许可协议 ([点击查看](#)) 中所定义的那样，并受其中规定的条款和条件的约束。

注：这些符号不能在未经许可的情况下在个人 APP 中随意指代非官方指定含义或元素。

符号	名称	只能指代苹果的...
	pencil.tip	标记功能
	pencil.tip.crop.circle	标记功能
	pencil.tip.crop.circle.badge_MINUS	标记功能
	book	图书

符号	名称	只能指代苹果的...
	book.fill	图书
	icloud	iCloud 服务
	icloud.fill	iCloud 服务
	icloud.circle	iCloud 服务
	icloud.circle.fill	iCloud 服务
	icloud.slash	iCloud 服务
	icloud.slash.fill	iCloud 服务
	exclamationmark.icloud	iCloud 服务
	exclamationmark.icloud.fill	iCloud 服务
	xmark.icloud	iCloud 服务
	xmark.icloud.fill	iCloud 服务
	link.icloud	iCloud 服务
	link.icloud.fill	iCloud 服务
	bolt.horizontal.icloud	iCloud 服务
	bolt.horizontal.icloud.fill	iCloud 服务
	person.icloud	iCloud 服务
	person.icloud.fill	iCloud 服务
	lock.icloud	iCloud 服务
	lock.icloud.fill	iCloud 服务
	arrow.clockwise.icloud	iCloud 服务

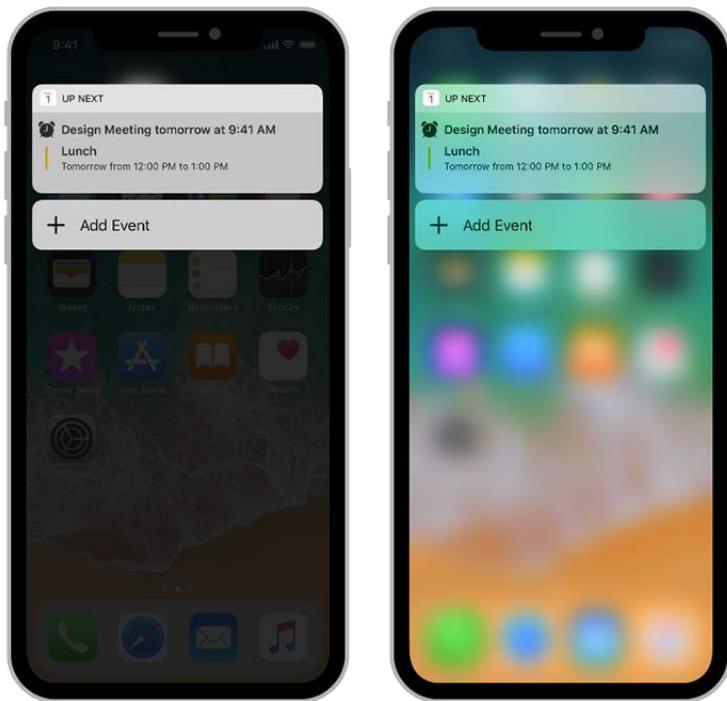
符号	名称	只能指代苹果的...
	arrow.clockwise.icloud.fill	iCloud 服务
	arrow.counterclockwise.icloud	iCloud 服务
	arrow.counterclockwise.icloud.fill	iCloud 服务
	icloud.and.arrow.down	iCloud 服务
	icloud.and.arrow.down.fill	iCloud 服务
	icloud.and.arrow.up	iCloud 服务
	icloud.and.arrow.up.fill	iCloud 服务
	phone	电话应用
	phone.fill	电话应用
	phone.circle	电话应用
	phone.circle.fill	电话应用
	phone.arrow.up.right	电话应用
	phone.arrow.up.right.fill	电话应用
	phone.arrow.down.left	电话应用
	phone.arrow.down.left.fill	电话应用
	phone.arrow.right	电话应用
	phone.arrow.right.fill	电话应用
	phone.badge.plus	电话应用
	phone.badge.plus.fill	电话应用

符号	名称	只能指代苹果的...
📞	phone.down	电话应用
📞	phone.down.fill	电话应用
📞	phone.down.circle	电话应用
📞	phone.down.circle.fill	电话应用
📠	teletype	电传打字机功能
📠	realtimetext	实时信息功能
🎥	video	FaceTime 通话应用
🎥	video.fill	FaceTime 通话应用
🎥	video.circle	FaceTime 通话应用
🎥	video.circle.fill	FaceTime 通话应用
☒	video.slash	FaceTime 通话应用
☒	video.slash.fill	FaceTime 通话应用
+🎥	video.badge.plus	FaceTime 通话应用
+🎥	video.badge.plus.fill	FaceTime 通话应用
↗️	arrow.up.right.video	FaceTime 通话应用
↗️	arrow.up.right.video.fill	FaceTime 通话应用
↖️	arrow.down.left.video	FaceTime 通话应用
↖️	arrow.down.left.video.fill	FaceTime 通话应用
❔	questionmark.video	FaceTime 通话应用
❔	questionmark.video.fill	FaceTime 通话应用

符号	名称	只能指代苹果的...
	questionmark.video.rtl	FaceTime 通话应用
	questionmark.video.fill.rtl	FaceTime 通话应用
	envelope	邮箱应用
	envelope.fill	邮箱应用
	envelope.circle	邮箱应用
	envelope.circle.fill	邮箱应用
	envelope.open	邮箱应用
	envelope.open.fill	邮箱应用
	envelope.badge	邮箱应用
	envelope.badge.fill	邮箱应用
	safari	Safari 浏览器
	safari.fill	Safari 浏览器
	airplayvideo	AirPlay
	airplayaudio	AirPlay
	arkit	ARkit
	livephoto	动态照片
	livephoto.slash	动态照片
	livephoto.play	动态照片
	faceid	Face ID

12.2 辅助功能

iOS 提供了大量的辅助功能来帮助视力障碍、听力障碍以及其他残疾人群体。大部分以 UIKit 为基础的 App 都有容易上手的特点，这样就可以让更多人在使用 App 时拥有更平等、良好的用户体验。



降低透明度



VoiceOver



按钮形状

注：电视字幕

为图片、图标和界面元素提供可选择的文本标签：文本备注在屏幕上是不可见的，但是能被 VoiceOver 读出来，让视障人士可以轻松导航。

相应辅助功能的偏好设置：如果你的 App 使用了 UIKit，那么用户界面、文本、界面元素就会自动调整至你的偏好设置，例如自动加粗并且放大文字（译者注）。当“减弱动态效果”的开关被打开，你的 App 也会适时检查并响应辅助功能的偏好设置。并且 App 所采用的自定义字体要和系统字体相匹配

注：iOS 设置中的字号设置

测试App的辅助功能：除了文字和动效的变化，辅助选项还能改变对比度、反转颜色、降低透明度等。当用户启用辅助功能时，可以观察到 App 在开启辅助功能后是如何表现和运作的。

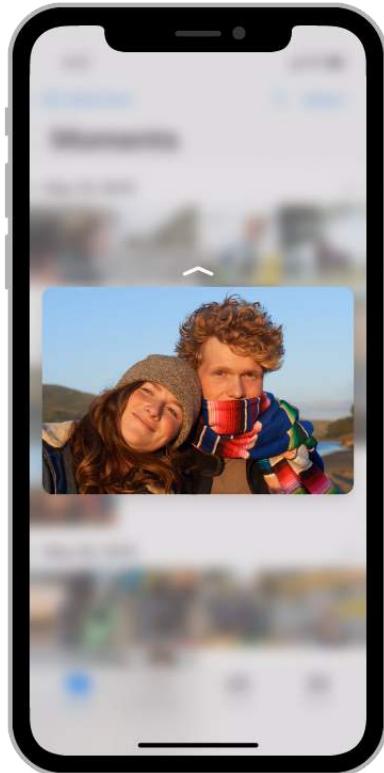
包含隐藏式字幕（注）和口述影像：隐藏式字幕有助于听障以及听觉不灵敏的用户，观看视频中的对话和其它音频内容。口述影像为视力受损的用户提供了重要视频的口头解说。

使用足够的色彩对比度：App 中的色彩对比度不足会导致内容难以阅读。例如，图标和文本可能会与背景混淆。在线颜色对比度计算器可以帮助您准确分析应用中的颜色对比度，以确保其符合最佳标准。尽管 7:1 是首选，不过还是要争取 4.5:1 的最低对比度，因为它符合更严格的可访问性标准。有关其他设计指南，请参阅 [颜色](#)。

有关更多信息，请参阅 [iOS 辅助功能](#) 和 [iOS 辅助功能编程指南](#)。

12.3 轻按和重压

轻按可以让用户使用 3D Touch 预览一项内容，例如页面、链接或者文件，预览窗口处于页面内容的最前端。只需用手指轻按就能预览，而抬起手指就能退出预览。如想要查看更多信息，只需重压，直到预览窗口弹出并充满整个屏幕即可。在一些预览窗口中，你可以向上滑动来显示相关的操作按钮。例如，在浏览 Safari 中的网页时，你可以向上滑动屏幕，实现在“新标签页中打开”，“加入阅读列表”和“拷贝”这三个操作。



轻按可以提供实时的、内容丰富的预览效果：在理想的情况下，预览窗口是为了给当前的页面提供补充信息，或者帮助你决定是否打开该页面。例如，先预览一下邮件中的链接，再决定是否在Safari中打开或与朋友 共享此链接。轻触还可以 App 在可被点击的列表中，通过预览补充信息来决定是否选择该行。

设计足够大的轻按视图：需要设计足够大的预览视图，使手指不会遮挡住内容。并且用户通过视图包含的信息来决定，是否应该重压打开完整项目。

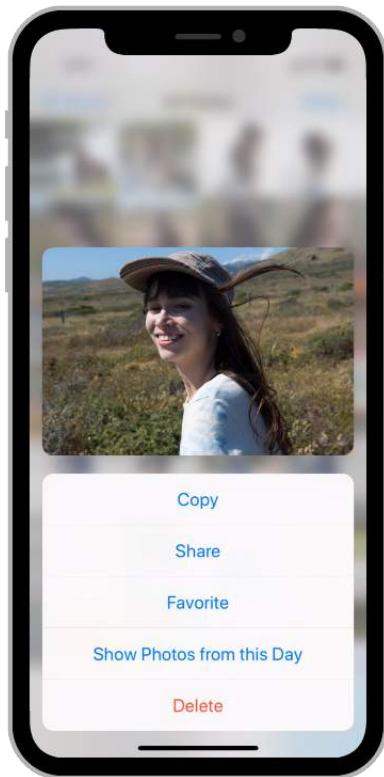
轻按和重压要保证场景的一致性：如果你在相同类型的元素中有些地方支持轻按和重压，在其它元素中不支持（译者注：比如同一个列表中第一个选项支持 3D Touch 功能，而第二个选项却不支持），用户就会对在什么地方可以使用这个功能产生混乱，或者认为你的 App 还是他们的设备发生了故障。**避免在轻按视图中显示按钮式元素：**如果这时用户抬起手指来点击面板中看起来像按钮的元素，那么轻按就会消失。

允许每个轻按视图都能被重压：尽管轻按视图能提供给人用户所需的大部分信息，但如果他们决定直接离开当前页面，直达该页面时，那么轻按也应该能转换成重压。重压展示出的内容也应该和点击该页面时一样。

不要让同一元素具备轻按和编辑菜单两个功能：当一个项目都启用两个功能时，不但会让用户感到困惑，也会让系统难以判断用户的意图。

了解更多的指导，请参阅 [Edit Menus](#)。

适时提供操作按钮：并不是每个轻按都需要操作按钮，但这是一个为常见任务提供便捷入口的好方法。如果你的 App 已经为该项目提供了自定义的触控操作，那么在最好在轻按中也加入同样的操作。



避免加上一个操作按钮来打开轻按界面：用户一般都通过更用力的按压来打开他们轻按的项目。所以，没有必要再提供一个明显的打开按钮。

不要让轻按成为唯一能执行动作的操作：并不是所有设备都支持轻按和重压，甚至有的用户会关闭 3D Touch 功能。你的 App 应该考虑到其它触发操作的方式。比如，你的 App 也可以通过点击和长按时出现的轻按视图。

12.4 Siri

你的 App 可以与 Siri 进行交互。用户可以发送语音指令让你的 App 执行某些任务。Siri 负责语言处理和语义分析来把语音请求转换成你的 App 能够处理的操作指令。你的 App 应该负责定义它所支持的功能、验证收到的信息、为 Siri 提供需要它展示的信息以及采取行动。来自你的 App 的回应信

息会由 Siri 说出来并且呈现在 Siri 界面。如果允许的话，你的 App 可以提供自定义的内容让 Siri 来展现。比如一个健身 App，可能会提供自定义的锻炼信息。



力求一个无需触屏或注视屏幕的声控体验：用户在使用 Siri 时不会经常盯着屏幕看。他们可能会通过耳机、汽车或是穿过房间来与 Siri 互动。尽可能地让用户在无需解锁屏幕的情况下也能完成任务。

不要试图模仿或是操控 Siri : 你的 App 不要去模仿 Siri，也不要尝试复制由 Siri 提供的功能或是提供一个看似来自于 Apple 公司的回复。

传播的内容要适当：不要包含可能令人反感或低俗的内容。

不要打广告：属于你的 App 的 Siri 体验绝对不要包含广告、营销或是 App 内购买的推销。

支持的交互

提供以下服务的 iOS App 可以与 Siri 进行交互。

服务	可支持的Siri交互行为
音、视频通话	发起呼叫 搜索通话记录
CarPlay 交互	激活并保存驱动程序的设置 更改汽车的音频资源 更改汽车的室内温度 更改汽车的除霜设置 更改汽车的座椅设置 切换汽车的广播电台
健身活动	开始, 暂停, 恢复, 结束和取消锻炼
列表和笔记	创建待办事项列表和项目 搜索待办事项列表和项目 标记完成的待办事项。 根据日期, 时间或位置创建提醒 创建笔记 搜索笔记 修改笔记
消息	发送消息 读取收到的消息 搜索消息
支付	向他人支付 要求付款 支付帐单 搜索账单 搜索并查看帐户信息, 包括余额, 积分和里程 转账
照片管理	搜索照片 在 App 中显示照片
骑行预订	预订自行车 提供骑行状态信息
车辆交互	激活危险警报灯或鸣喇叭 锁定和解锁车门 检查当前燃油或功率水平
视觉代码	显示视觉代码, 如QR码或条形码 (二维码)

回应用户

快速应答并且减少交互操作：用户使用 Siri 是为了方便，所以不要让他们等待回应。你的 App 需要呈现高效且集中的选项来减少需要额外提示的情况

将用户直接带到指定内容：从 Siri 切换至你的 App，应该直接跳转用户想要去的地方。不要显示中间画面或是信息，阻碍跳转过程或是拖慢用户。

保证是相关的、精确的以及合适的：你的 App 的回应必须和用户当前的请求相关且符合用户期望。比如，用户向 Siri 发送指令，要求你的 App 发送一条信息，那么你的 App 就应该就严格执行该操作，不要有额外的操作。

在收到关于金钱交易的请求时，将最安全、价格最低的选项设为默认值：绝对不要欺骗用户或歪曲信息。对于涉及不同价位的购买，不要默认选择最贵的。当用户决定付钱时，并不知情的情况下，不要收取额外的费用。

设计自定义界面

确保你的自定义界面与 Siri 很好的融合：你可以使用 App 特有的颜色、象征性图像或是其它设计元素来表达品牌风格，但是任何自定义界面元素必须让人感觉它们仍然适合于 Siri。除非你的 App 需要一个完全独立的自定义界面，否则就会将你的内容整合进 Siri 的默认界面中。

提供足够的边缘：避免将内容扩展到界面的边缘，除非它的内容能够像地图一样自然地延伸出屏幕。一般在界面的每个边缘和内容之间提供至少几个像素的边距。使用界面顶部的 App 图标进行对齐指导。当内容与图表中心对齐时往往会觉得比较和谐。

最小化界面的高度：理想情况下，你的界面不应高于屏幕高度，因此用户可

以在不滚动的情况下查看所有内容。

不要创建交互式的界面：你的界面需要禁用手势（但是用户可以进行点击，当你的 App 或其他事件在 Siri 界面显示时），以避免显示出交互式的图像或形状。

不要在自定义界面包含你的 App 名字和图标：系统会自动展示这些信息。

提高准确度

如果可以的话，定义自定义词汇：帮助 Siri 通过定义人们可能在请求中实际使用的特定术语，以了解更多关于你的 App 执行的操作，例如帐户名称，联系人姓名，照片标签，相册名称，乘车选项和训练名称。这些术语对你的 App 来说应该是特定的、唯一的。不要包含其他 App 名称、与其他应用程序明显相关的术语、不当的语言或保留短语，例如“嘿 Siri”。请注意，Siri 会使用你定义的术语解决用户请求，但不能保证辨认成功。

考虑可替代的 App 名称：如果用户改变 App 名称的发音，你可以提供备用名称列表，以便 Siri 能够更灵活准确地定位 App。例如，UnicornChat 可以将 Unicorn 定义为备用 App 名称。不要将其他 App 名称列为你的 App 的备用名称。

提供例句：为 Siri 提供例句，当用户点击 Siri 界面的帮助按钮时这些例句就会被展现在指南里。使用这些例句引导用户如何以最简单高效的方式，通过 Siri 来使用你的 App。

开发者指南，请参阅 [Intent Phrases](#)。

更多：相关设计指导，请参阅 [watchOS > Siri](#)。开发者指南，请参阅 [SiriKit](#)。