

Android UI设计领域的经典著作, Amazon五星级畅销书, 内容系统全面, 实战性极强, 包含大量最佳实践

不仅详细讲解了Android应用UI设计需要掌握的各项技术和特性, 还全面总结了Android应用UI设计的方法、技巧、模式以及反模式

移动开发



Smashing Android UI

Responsive User Interfaces and Design Patterns for Android Phones and Tablets

精彩绝伦的Android UI设计

响应式用户界面与设计模式

Juhani Lehtimäki 著

王东明◎译

机械工业出版社
China Machine Press

移动开发

Smashing Android UI

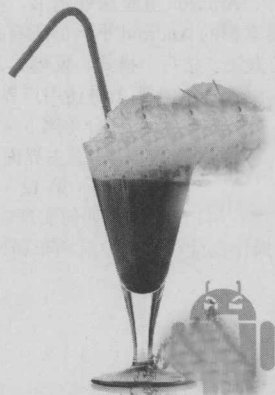
Responsive User Interfaces and Design Patterns for Android Phones and Tablets

精彩绝伦的Android UI设计

响应式用户界面与设计模式

Juhani Lehtimäki 著

王东明◎译



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

精彩绝伦的 Android UI 设计: 响应式用户界面与设计模式 / 莱特曼 (Lehtimäki, J.) 著; 王东明译. —北京: 机械工业出版社, 2013.8

(华章程序员书库)

书名原文: Smashing Android UI: Responsive User Interfaces and Design Patterns for Android Phones and Tablets

ISBN 978-7-111-43589-1

I. 精… II. ①莱… ②王… III. 移动终端—应用程序—程序设计 IV. TN929.53

中国版本图书馆 CIP 数据核字 (2013) 第 181195 号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书版权登记号: 图字: 01-2013-0625

Copyright © 2013 by John Wiley & Sons, Inc.

All Rights Reserved. This translation published under license. Authorized translation from the English language edition, entitled Smashing Android UI: Responsive User Interfaces and Design Patterns for Android Phones and Tablets, ISBN 978-1118387283, by Juhani Lehtimäki, Published by John Wiley & Sons. No part of this book may be reproduced in any form without the written permission of the original copyrights holder.

本书中文简体字版由约翰·威利父子公司授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

本书封底贴有 Wiley 防伪标签, 无标签者不得销售。

本书是 Android UI 设计领域的经典著作, Amazon 五星级畅销书。不仅从 Android 应用设计者的角度系统讲解了要从事 Android UI 设计必须要掌握的 Android 平台的所有技术和特性, 还从 Android 应用开发者的角度全面总结了 Android UI 设计的方法、技巧、模式、反模式, 以及如何实现响应式用户界面设计。本书共 21 章, 分为四部分。第一部分 (第 1 ~ 4 章) 讲述用户界面设计、用户界面设计的工具、移动设备和触摸设备的设计, 并探讨 Android 平台; 第二部分 (第 5 ~ 11 章) 介绍 Android 的应用架构和在线指南、Android 的意图系统、Android 应用的导航结构、主界面应用小部件、通知、物理按键、输入法和传感器设计, 以及平台用户界面组件设计; 第三部分 (第 12 ~ 16 章) 讨论 Android 资源的管理、Android 应用的布局、可缩放的图形、响应式设计, 以及如何实现响应式用户界面; 第四部分 (第 17 ~ 21 章) 阐述用户界面设计模式、用户操作设计模式、导航和布局设计模式、数据设计模式以及用户界面设计的反模式。

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 谢晓芳

北京市荣盛彩色印刷有限公司印刷

2013 年 10 月第 1 版第 1 次印刷

186mm × 240mm · 16.25 印张

标准书号: ISBN 978-7-111-43589-1

定 价: 59.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

译者序

目前市面上有很多关于 Android 的技术类书籍，还有各种各样的设计类书籍，而本书把与 Android 开发相关的技术与设计完美结合在一起，别具一格。因为绝大部分开发人员不熟悉设计，而多数设计人员又欠缺 Android 方面的开发知识，所以阅读本书，可以弥补各自的短板。

本书比较适合希望学习设计知识的 Android 开发人员以及希望了解 Android 技术知识的设计人员。因为本书涉及的技术非常基础，包含的代码也相当简单，所以设计人员在看本书的时候丝毫不会感到任何阻碍。通过阅读本书，设计人员能够了解 Android 平台的一些技术知识，通过对这些知识的学习，可以更好地实现针对 Android 平台的应用设计，不用照搬其他智能手机平台的设计，从而制作出具备浓郁 Android 风格的界面和交互设计。对于开发人员，熟悉符合 Android 的设计原则可以减少与设计人员的沟通成本，知道这样设计的优点，了解用户的使用习惯，从而可以在代码层面上更好地实现应用程序，进而创建用户易于使用的优秀应用程序。不过，正是因为兼顾了设计人员，所以本书涉及的技术知识都很浅显易懂，这些对于设计人员来说已经足够，但是对于开发人员还远远不够，所以我建议开发人员阅读完本书之后，再根据实际情况，找到相关知识点对应的其他技术书籍补充知识，这样才能更好地实现本书中的各类设计。

本书的结构非常科学、合理，首先讲解用户界面设计的入门知识，然后概览 Android 平台，接着又介绍 Android 界面的主要特性和各类 UI 组件，最后讨论 Android 设计模式和反模式。这种组织方法由浅入深，易于理解，相信读者可以从中获得一个很好的阅读体验。

为了行文流畅不夹杂过多的英文单词，译文中将绝大部分英语专业词汇翻译成中文，有一些没有统一译法的词汇大家看起来可能比较陌生，比如，将 intent 译为“意图”。所以在专业词汇第一次出现的时候，都加注了对应的英文，方便大家对照。

本书的翻译过程中得到了很多人的帮助，尤其要感谢仝鹏飞先生对部分译文提出了非常专业的建议。特别感谢谢晓芳编辑在本书翻译过程中的大力支持，对译者不明白的问题给予了耐心细致的解答并督促译者的翻译进度，此书才得以顺利出版。

鉴于译者水平有限，书中难免还有一些翻译不准确甚至译错的地方，恳请广大读者谅解，同时欢迎读者批评指正，共同进步。

王东明

也可以手动导航到对应的示例应用。



这个二维码扫描应用允许你扫描本书中的二维码，它将会自动把你带到对应的示例

资料来源：Scan 公司

应用兼容性

注意，并非全部示例程序都兼容所有 Android 版本。尽管有各种方法让应用兼容 Android 的老版本，但我还是打算仅使用 Android 的核心 API，以此保持示例代码的整洁和跨版本的兼容性。如果你正在使用旧版本的 Android 设备，请注意，你可能会看到一个界面告知你想要打开的这个示例应用不可用。

配套应用的源代码

配套应用是完全开源的，你可以下载并在自己的电脑上浏览配套应用的所有代码。在 github 中可以找到配套应用的源代码，链接地址：https://github.com/JuhaniLehtimaeki/smashing_android_ui_example_app/。

致谢

撰写本书时做了大量的工作，也收获了很多乐趣。如果没有女朋友默默支持，我无法完成这本书的创作。在过去的几个月中，她明白我为什么不得不坐在屋子里敲击键盘，而不是和她共同享受在一起的时光。非常感谢她的理解！

同时，感谢 Snapp TV 公司的领导和同事，让我自主安排工作事宜，允许我花费部分工

作时间撰写本书。特别感谢 Jasper Morgan 鼓励我在这本书上花费足够的时间，从而避免我在创作过程中压力过大。

同样感谢 Google+ 中优秀的 Android 社区。我喜欢阅读其中的文章和评论。很多问题的讨论内容都非常翔实，气氛友好，并且鼓励每个人都参与进来。非常感谢 Google 公司的 Android 开发人员，尤其是 Nick Butcher，他提倡积极参与到讨论中来，同时还鼓励撰写 Android 方面的文章。

感谢每一位阅读博客中的文章以及发表评论的读者，这是对我继续写下去莫大的鼓励，也让我从中学到了很多知识。感谢你关注 Android 博主，他们帮助在线 Android 社区信息日益丰富多彩。

如果没有 Android 社区中库的贡献者，那么构建 Android 应用将会非常困难。感谢每一位构建 Android 库并免费分发、开源给任何人使用的贡献者。他们正在做一项令人惊讶的工作，这项工作让每个人的生活变得更简单。

同时还要感谢 Google 公司，除了 Android 操作系统外，它还提供了诸多构建 Android 应用的工具。如果没有神奇的 Google Drive（文档）的帮助，它让我集中精力撰写，而不用掌握文字处理软件，我将无法完成本书的撰写。感谢 Herzoglich Bayerisches Brauhaus Tegernsee^① 给予我在深夜撰写本书的能量。

最后一点也同样重要，感谢 Wiley 让我撰写人生中的第一本书。感谢 Kezia Endsley、Craig Smith 以及 Sara Shlaer 在撰写过程中指导与帮助我。非常感谢 Sebastian Kaspari，他作为本书的技术编辑，确保了本书中所有的代码和示例都能正常工作并且浅显易懂。感谢 Kevin Cannon 在很多方面为我提供了设计人员的视角。

① 德语，巴伐利亚公爵啤酒屋特格尔恩。——译者注

目 录

译者序

前言

第一部分 Android 设计简介

第 1 章 用户界面设计和易用性入门·····2

- 1.1 考虑技术与设计·····2
- 1.2 理解心智模型·····3
- 1.3 为用户设计·····6
- 1.4 没有应用无所不精, 选择适合自己的领域·····9
- 1.5 你是专家, 用户不是设计师·····9
- 1.6 了解用户, 为普通人设计·····9
- 1.7 总结·····11

第 2 章 不要马上开始编程·····12

- 2.1 原型设计·····12
- 2.2 设计工具·····14
- 2.3 用户测试·····17
- 2.4 总结·····20

第 3 章 考虑为移动和触摸设备设计·····21

- 3.1 为移动而设计·····21
- 3.2 为触摸界面设计·····26
- 3.3 总结·····29

第 4 章 探索 Android 平台·····30

- 4.1 Android 平台的挑战·····30
- 4.2 Android 的版本·····36

4.3 Android 应用的发布·····38

4.4 开源的意义是什么·····40

4.5 总结·····41

第二部分 Android 平台的特性和 UI 组件

第 5 章 Android 应用架构和在线指南·····44

- 5.1 Android 应用程序结构概览·····44
- 5.2 Android 项目结构概览·····47
- 5.3 官方 Android 指南·····48
- 5.4 总结·····49

第 6 章 Android 的意图系统·····50

- 6.1 意图让应用程序相互协作·····50
- 6.2 使用社交网络并分享·····51
- 6.3 和浏览器一起工作·····52
- 6.4 Android 意图系统的工作原理·····53
- 6.5 创建自定义行为·····58
- 6.6 意图无处不在·····58
- 6.7 意图与第三方 API·····58
- 6.8 总结·····59

第 7 章 Android 应用的导航结构·····60

- 7.1 Android 导航系统的组件: 活动和意图·····60
- 7.2 Android 导航控件·····61
- 7.3 总结·····70

第 8 章 主界面应用小部件71	11.4 使用图标.....117
8.1 主界面应用小部件的使用.....71	11.5 使用动画和过渡效果.....121
8.2 更新应用小部件数据.....73	11.6 总结.....127
8.3 应用小部件布局和功能.....75	
8.4 实现应用小部件.....78	第三部分 可扩展的 Android 设计
8.5 总结.....80	
第 9 章 通知及告知用户81	第 12 章 管理 Android 资源130
9.1 Android 通知方法.....81	12.1 使用 Android 资源.....130
9.2 何时通知用户.....83	12.2 屏幕密度设计.....132
9.3 何时不要通知用户.....86	12.3 屏幕大小和形状的设计.....135
9.4 避免使用弹出框.....86	12.4 语言和地区设置.....137
9.5 从状态栏通知中获得更多信息.....87	12.5 设备控件的设计.....137
9.6 总结.....91	12.6 平台版本的设计.....138
第 10 章 物理按键、输入法和传感器的设计92	12.7 设备模式的设计.....138
10.1 触摸屏的设计.....92	12.8 总结.....138
10.2 手机物理按键的设计.....93	第 13 章 Android 应用布局139
10.3 软键盘的设计.....94	13.1 Android 布局策略.....139
10.4 物理键盘设计.....99	13.2 用 XML 文件和代码布局.....140
10.5 方向键和轨迹球设计.....99	13.3 布局的管理.....141
10.6 触控笔设计.....100	13.4 定义布局的尺寸.....149
10.7 声控设计.....101	13.5 滚动条.....149
10.8 外接键盘、鼠标和触摸板的设计.....101	13.6 z 轴的布局顺序.....150
10.9 智能附件的设计.....102	13.7 内边距和外边距.....150
10.10 传感器的设计.....102	13.8 导入与合并布局文件.....150
10.11 第二块屏幕的设计.....102	13.9 自定义布局.....151
10.12 总结.....103	13.10 Android 开发工具和用户界面生成器.....151
第 11 章 平台用户界面组件的设计104	13.11 调试布局.....152
11.1 使用用户界面小部件.....104	13.12 总结.....153
11.2 自定义用户界面小部件.....111	第 14 章 可缩放的图形154
11.3 调整排版.....113	14.1 九宫格图片.....154
	14.2 绘图 XML.....158

14.3	用代码绘制	163	18.4	使用下拉刷新设计模式	212
14.4	总结	166	18.5	使用滑动忽略的手势	215
第 15 章 超越可缩放——响应式设计		167	18.6	总结	217
15.1	Android 设备不仅仅只是手机	167	第 19 章 导航和布局设计模式		218
15.2	了解响应式设计	169	19.1	使用层叠图库	218
15.3	制作响应式用户界面的常见方法	176	19.2	使用仪表盘	220
15.4	总结	178	19.3	使用工作空间	223
第 16 章 实现响应式用户界面		179	19.4	使用分拆视图	226
16.1	片段简介	179	19.5	根据上下文扩展的设计模式	228
16.2	片段和活动结构	181	19.6	使用侧边导航	230
16.3	移植现存的应用	182	19.7	总结	232
16.4	示例程序	183	第 20 章 数据设计模式		233
16.5	总结	193	20.1	使用动态列表	233
第四部分 Android UI 设计模式			20.2	使用图片占位符设计模式	234
第 17 章 用户界面设计模式简介		196	20.3	非强制登录模式	236
17.1	用户界面设计模式	196	20.4	使用拖动重排手柄模式	238
17.2	使用 UI 设计模式的好处	197	20.5	总结	239
17.3	Android 设计指南中的设计模式	198	第 21 章 用户界面设计反模式		240
17.4	本书介绍的用户界面设计模式	198	21.1	避免使用启动界面	240
17.5	总结	199	21.2	避免使用引导界面	241
第 18 章 用户操作设计模式		200	21.3	避免使用确认窗口	242
18.1	使用操作栏模式	200	21.4	避免在界面上使用 Back 按钮	243
18.2	使用快捷操作设计模式	206	21.5	避免使用菜单按钮	244
18.3	使用操作抽屉设计模式	210	21.6	避免隐藏状态栏	244
			21.7	避免为快捷操作使用滑动覆盖层	245
			21.8	避免使用非 Android 的设计	246
			21.9	总结	247

第一部分

Android 设计简介

第1章 用户界面设计和易用性入门

一款应用（app）最重要的品质就是易用性（usability）。如果用户不知如何得到它，或者不清楚如何使用，那么这个应用做得再有特色也无济于事。在移动应用市场残酷的环境下，用户几乎总能找到替代品。如果一款应用不能让人感觉良好，或者用户不知道如何使用它的主要功能，那么用户极有可能卸载这款应用，不会给它第二次机会。

用户通过用户界面（UI）来操作和浏览你的应用，除此之外的一切元素，也都会为用户界面中反映出来。假如一款应用的杀手级特性提供了下一代云通信技术（或者应用包含任何其他一些了不起的功能），但是并不直观，那么你会面临花费了大把的时间，却构建了用户甚至都没有尝试过的一些功能的风险。

想要把用户界面做好，需要在设计上下工夫。本章介绍的一些概念和观点可以让读者更轻松地理解在设计上投资的重要性。本章还阐述设计过程中的一些重要概念并提供了一些想法，旨在让用户成为设计过程中不可或缺的一部分。

1.1 考虑技术与设计

用户界面设计不是一门精密科学，它也不会自动生成。它需要努力付出并投入资金。我职业生涯中的绝大多数时间都是在技术驱动的公司里度过，也就是所谓的开发者工作室。几乎在整个职业生涯应用的制作过程中，我都不得不争取要求加入设计依据。几乎所有雇用设计者或者让设计者参与进来的请求要么被回绝要么被严重误解。在一些公司里，认为设计者是画图标的人。在另一些公司里，认为设计者的存在就是浪费金钱和时间。在这些公司中，工程师或者产品团队经常自己做设计；有的时候设计源于偶然。尽管不同的项目会有各种各样的结局，但是始终都缺乏设计。产品在工程师眼中简单易用，但工程师无法理解为什么用户不能像他们所期待的那样使用这款产品。技术团队通常会这样说：“那个按钮明明就在那里，用户应该知道如何使用它。他们太笨了！”

我也曾经在另一种极端的公司工作过。所有的工作都是设计驱动的，只有设计者参与早期的客户见面会，选用的技术也没有参考团队的技术特点。设计工作用 Adobe Photoshop 和 InDesign 实现，并没有考虑到技术的可行性，这将把技术团队逼入绝境，他们无法完成设计者想要的设计。

两种极端都不可取。在这两种情况下，初衷与结果都将大相径庭。我也有幸参与了所有的工作都做得恰到好处的一些项目。与懂得软件工程并不是简单编程的设计者或者设计团队一起工作，并且拥有一支认可用户交互设计和视觉设计能够创造绝妙设计的开发团队。设计人员与开发人员密切协作所创造的价值远远大于他们各自创造的价值。

致开发人员——用户与你的思维方式并不一样！有人已经学习过用户界面相关的知识并做出了专业的设计。不要认为你做了相关的工作就不需要用户。用户知道如何操作你的用户界面证明不了什么，不会操作相同用户界面的用户也不能证明他们愚笨。用户不会用，说明用户界面设计得很糟糕。设计人员了解用户界面的结构，他们也懂得如何测试并验证这种设计是优秀的。信任你的设计师们，他们不会因为怀有恶意或者想让你更努力工作而提出用户界面修改意见，他们想要修改界面是因为这些修改可以让用户体验更好。

致设计人员——软件工程并非单单只是编程。事实上，编程仅仅占开发者的一小部分时间。制作成功的软件产品需要谨慎的计划、架构设计、对象关系设计、模块组件设计、数据库设计、可维护性计划、部署、质量保证等其他事宜。如果你知道怎样写脚本，那这将是一个很好的开端。但是写脚本并不是软件开发。基于 Photoshop 制作的用户界面示意图来制作一款合格的软件产品并不简单，这需要时间、计划，尤其是经验！相信你的开发人员，他们没有花费大量时间来制作一个界面不是因为他们懒惰。他们把时间花在计划上，是为了让程序运行更流畅也更容易维护。这样整个团队就可以一起调整完善用户界面，修改也不会令人不知所措。

1.2 理解心智模型

现在应该换个方向，从用户的角度考虑应用。当用户第一次看到你的应用或者当用户再次使用这个应用的时候，他们的脑海里会呈现出什么？本节介绍一个非常重要的概念——心智模型（mental model）。

心智模型是用户在大脑中对应用的功能形成的模型。事实上，人们面对所有事物的时候都会这么做。当我们学习如何开车的时候，我们会对车辆如何工作建立一种心智模型。这种模型并不一定要在技术上是正确的才能使司机受益。比如，实际上，在现代汽车中，方向盘没有直接与前轮相连，但这并不重要。我们还是可以假设它们连在一起。我们可以想象，当转动方向盘的时候，会带动一系列的齿轮旋转，最终让前轮转变方向。尽管这个模型在技术上做了简化，但是可以帮助我们理解和模拟在驾驶汽车时，车辆是如何工作的。在我们的心智模型中，转动方向盘与现实世界中转动方向盘的效果一致，所以在现实生活中按照此心智模型驾驶一辆汽车完全没有问题。我们认为自己明白了车辆如何工作，感到非常高兴。

人们在使用应用的时候，也认同一样的道理。对于用户来说，能让他们在脑海里模拟程序的功能，预测会在何时发生什么非常重要。心智模型是用户界面设计中极其重要的概念之一。用户界面设计必须支持用户的心智模型，同时应用要做到风格一致，功能可预测。

如果应用与用户的心智模型保持一致，那么这款应用将会易于使用。如果用户能非常轻

松地对这款应用建立心智模型,说明这款应用是直观的。用户界面设计中的每件事归根到底都与用户的心智模型有关,每个与用户体验有关的问题,都是由于此应用的用户心智模型与应用的实际表现不一致而导致的。

1.2.1 构建心智模型

在现实生活中,能够通过简单观察物体外表推断出物体的许多物理属性。你会知道它们有多重,知道通过什么方法使用它们,还知道用什么方法不奏效。对于实体来说,你可以通过亲身体验来确定它们如何工作。

当你使用软件时,会受到像素的影响。如何确定用户们知道哪些功能怎样使用?理论上,问题的答案非常简单,但是在实际中困难重重。你需要引导用户,用户界面必须富有逻辑性,并且包含界面中的各项功能是如何工作的视觉线索。

有很多工具可以用来解决这些问题。可以通过颜色、形状、纹理、3D 效果还有动画来达到引导的效果。对于任意一种工具,只有在正确的时机和正确的位置使用才能起到辅助效果,所以,这就需要用户界面设计的相关技能了。

让我们看两款 Android 应用。Google 的 Play Books 是一款电子书阅读器应用。用户在脑海中对如何阅读图书产生了一幅极富冲击力的画面,并且这款应用通过用手指滑过页面的变换动画帮助用户确定他们的心智模型。图 1-1 ~ 图 1-3 展示了变换动画的几帧。

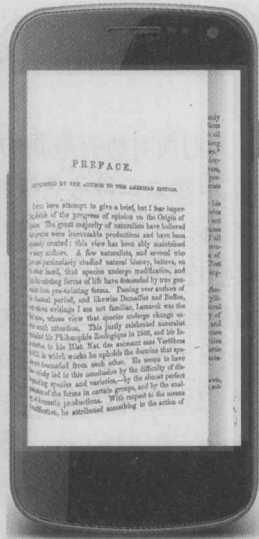


图 1-1 用户开始用手指滑过页面

资料来源: Google 公司

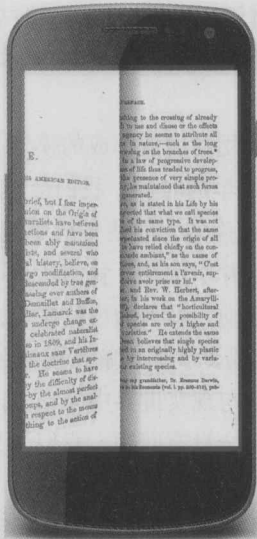


图 1-2 用户继续这个滑动的姿势

资料来源: Google 公司

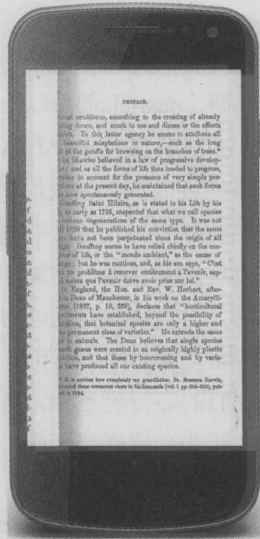


图 1-3 用户完成了滑动手势

资料来源: Google 公司

在另一个例子中(见图 1-4),这款 Gigbeat 应用为用户提供了细致的视觉线索,帮助用

户了解用户界面如何工作，所以用户形成了与产品一致的心智模型。在 Gigbeat 应用中的巡回演出界面上，你能看到乐队的部分简介图片。所选内容的旁边又显露出部分内容，这很容易让用户联想到如果想要看到更多内容，可以往那个方向移动。在一个触摸型的界面上，导航通常借助于滑动或者轻击来完成。

1.2.2 平台一致性和用户期望

现在面临的一个额外的巨大挑战是：每个用户都会形成与众不同的心智模型，而且他们所期待的内容也有所不同。部分用户对一些用户界面非常熟悉，而另一部分却从来没有见过这些用户界面。用户来自不同的文化环境并拥有各异的背景。他们在软件和软件平台方面具备不同的经验。比如，Mac 的一位终身用户与 Windows 的一位终身用户对于同一款用户界面的期望肯定是不一样的。

比如，Microsoft Windows 用户期望双击窗口的标题来最大化窗口。这个功能并不直观，但是用户已经学会使用这种功能并且现在很依赖这个功能。因此，维持一个平台内应用程序功能的一致性相当重要。

 **小贴士：** 用户期望是心智模型构建过程中非常重要的一部分。如果你的应用参考了所有平台的指南，那么将极大降低用户抵触应用的可能性。

让我们看几个有关用户界面约定的例子。通过约定可以让用户更轻松地操作应用，因为这些约定在 Android 平台上已经广泛使用。图 1-5 和图 1-6 展示了两款选项卡式的用户界面。与其他平台相比，Android 平台的选项卡拥有截然不同的外观和感觉。差异化的外观是件好事，因为 Android 平台上的选项卡与其他平台的选项卡在功能上也略有不同。在 Android 平台上，选项卡之间切换的主要方法是滑动手势，当然在选项卡上轻击也有同样的效果。基于平台的一致性，用户已经了解了这个功能，所以不需要你做额外的设计来解释它。

另外一个有助于应用设计的平台一致性例子是 Android 的菜单功能。如果应用拥有非常多的菜单选项以致超出了显示屏幕，可以使用 Android 操作栏扩展（overflow）菜单。扩展菜单使用三个垂直排列的点（见图 1-7 和图 1-8）。用户了解这个图标代表的意义，虽然对于非 Android 用户来说这个图标可能并不明显。在本书的后续部分，你将会更深入地了解不同的 Android 用户界面约定和组件。



图 1-4 应用 Gigbeat 通过细致的视觉线索让用户知道还有更多的内容

资料来源：Google 公司



图 1-5 Android 联系人应用拥有选项卡式用户界面，选项卡之间可以通过滑动手势导航

资料来源：Android



图 1-6 Android 拨号应用同样使用了类似的选项卡式用户界面，它允许使用滑动手势作为主要的导航方法

资料来源：Android

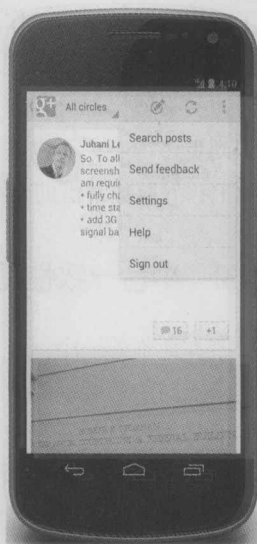


图 1-7 Google+ 应用使用扩展菜单来展示菜单项，这样就不会使菜单充斥整个屏幕

资料来源：Google 公司

1.3 为用户设计

制作合格的用户界面来支持用户心智模型的构建，需要将用户放在设计过程的中心。应用设计必须始于用户需求，并自始至终都要把重点放在用户身上。在软件行业中，以用户为中心的设计（User Centered Design, UCD）这个术语经常被提及，但它到底是什么意思呢？如何将用户置于设计工作的中心？

本节介绍以用户为中心的设计的一些基本概念，并讨论一些重要的术语。对于任何一个项目的成功来讲，以用户为中心的思想都相当重要。这里介绍的概念对于任何一个项目来说都适用。以用户为中心的设计这个主题非常庞大，市面上有很多书来阐述这个话题。所以您可以通过很多方式来加深对这个主题的理解。我希望这里介绍的内容能激发你对这个主题的兴趣，并且鼓励你从著作和网络中获取更多的信息和不同的观点。

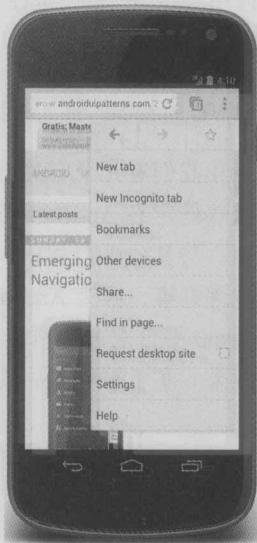


图 1-8 如果 Google 浏览器不使用扩展菜单，那么多菜单项将会塞满整个屏幕

资料来源：Google 公司

1.3.1 用户目标

用户并非单纯地想要使用你的应用，他们想借此应用完成某些事。他们有类似于“记得在明天下班的路上买牛奶”或者“买一台 Android 平板电脑”这样的目标。用户会根据应用帮助他们实现目标的情况来评价应用。如果你的用户界面能辅助他们实现目标，他们会很高兴。

我们从更深的层次审视用户目标。什么是用户目标以及什么不是用户目标？简而言之，一个用户目标就是用户想要完成的事，但是没有描述任何与应用相关的功能。比如，“用户想要保存一份文档”，这不是一个用户目标，用户从来不想保存任何东西。当用户想要再次使用文档的时候，他们希望文档还在那里。这个目标可以通过保存功能来实现，但这并不是唯一选择。

对于一些拥有技术背景的人来说，用户目标类似于用例（use case）。一个用例描述了用户单次使用应用时的场景。用户目标描述了用户为什么想要这么做。

表 1-1 展示了如何通过虚构的校园计划应用来确定用户目标的一些例子。表 1-1 包括糟糕的用户目标以及糟糕的原因，同时，也包括与情景一致并且正确格式化的用户目标。注意，好与坏的区别非常微妙，但是糟糕的用户目标可能会导致设计人员和开发人员提前思考产品特点和某些用户界面解决方案。

表 1-1 虚构的校园计划应用的用户目标案例

不是一个用户目标	对应的用户目标
用户想要保存一份文档（暗示功能）	过一会儿用户想要工作的时候，用户想继续操作刚才的那份文档
当收到新邮件时，用户想要获得通知（暗示功能）	用户想要知道新邮件何时到达
用户想要打开日历查看演讲时间（太简单）	用户想要知道指定的演讲何时开始
用户想要看到日历（太抽象）	包含多个用户目标
用户想通过搜索来查找一场演讲（描述功能）	也许包含多个用户目标，其中之一是用户想要知道某人是否参加了某场演讲
用户想要在日历中增加新的提醒（描述功能）	用户需要记住约会的开始时间

1.3.2 不要谈论特性

首先要做的就是改变谈论软件项目的语言和术语。你必须停止谈论产品特性并且开始谈论用户的需求。避免通过撞大运的方式完成用户界面设计。如果软件仍处于概念设计阶段，就开始谈论软件的特性和软件的工作原理，那你需要后退一步。首先需要满足用户的需求。注意避免类似这样的措辞，比如，“喂，我们应该在这里加个‘保存’按钮！”这句话可能造成的后果就是：没用的用户界面或者让用户感到迷惑的用户界面。首先要考虑用户需求！

1.3.3 确定用户目标

开始考虑应用特性和应用设计之前，你应该有个用户目标列表，这说起来容易，但是列

表从何而来?

用户目标应该源自用户调研、领域内的专业知识和对应用试图去解决问题的理解。用户调研有时候不容易实现或者不灵活,你不得不依赖更多的非正式方式。考虑通过用 Google Talk 和 Facebook 做一些专门的用户调研。

开始一个有价值讨论的前提是写下你认为最重要的用户目标。初始目标列表是极具价值的工具,尤其是当你正在构建一款面向消费者的应用时。一份完整的用户目标列表是对产品功能点非技术性的描述。不需要技术知识或者冗长的文档就可以很轻松地进行讨论。一个用户目标是一两句话或一个问题,例如,“这对你是否重要?”

为了创建用户目标列表,基于你的专业经验简单地记录你的思考过程非常重要。然后与消费者、领域专家、用户以及你觉得有帮助的任何人交流。接着你要重新组织、重写、重新排列目标列表的优先级。在这个过程的最后,你将会获得一些总结,描述通过使用你的应用,用户会完成什么目标,即使这个时候,一个产品特性都还没有开始设计。

1.3.4 从目标到特性

理解易用性重要的一点就是应用特性该如何与用户目标相关联。任何一款应用的每一个功能都应该为了满足用户的需求而设定。不考虑用户目标而草率地添加产品特性很容易产生令人困惑的用户界面,并且可能花费大量时间构建了将永远不会使用的特性。

理论上,一个应用的每个产品特性应该能追溯到一个用户目标(见图 1-9)。实际情况并不总是这样,但对于一个团队来说这应该是一个争取实现的目标。不考虑用户目标而添加的产品特性会导致应用功能蔓延。增加更多的产品特点并不能让产品变得更好,尤其是那些没有经过深思熟虑的产品特性。基本上任意一款办公产品套件都是功能蔓延的绝佳例子。多年来,办公产品套件不断地在已有的产品特性上堆砌新的产品特性,现在使用那些应用都需要经过专门训练。

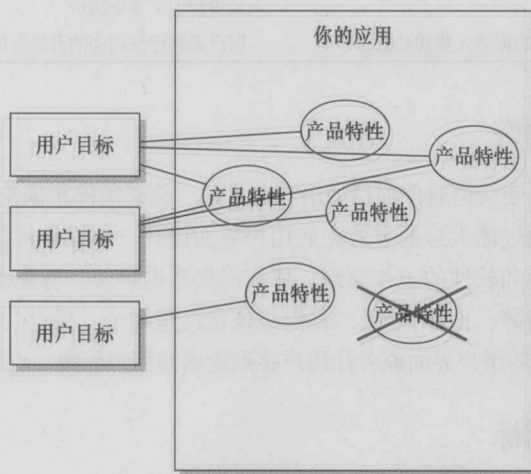


图 1-9 在应用开发的最后,任何一个产品特性都应该与用户目标相对应