

The  
Pragmatic  
Programmers

TURING

图灵交互设计丛书

第2版

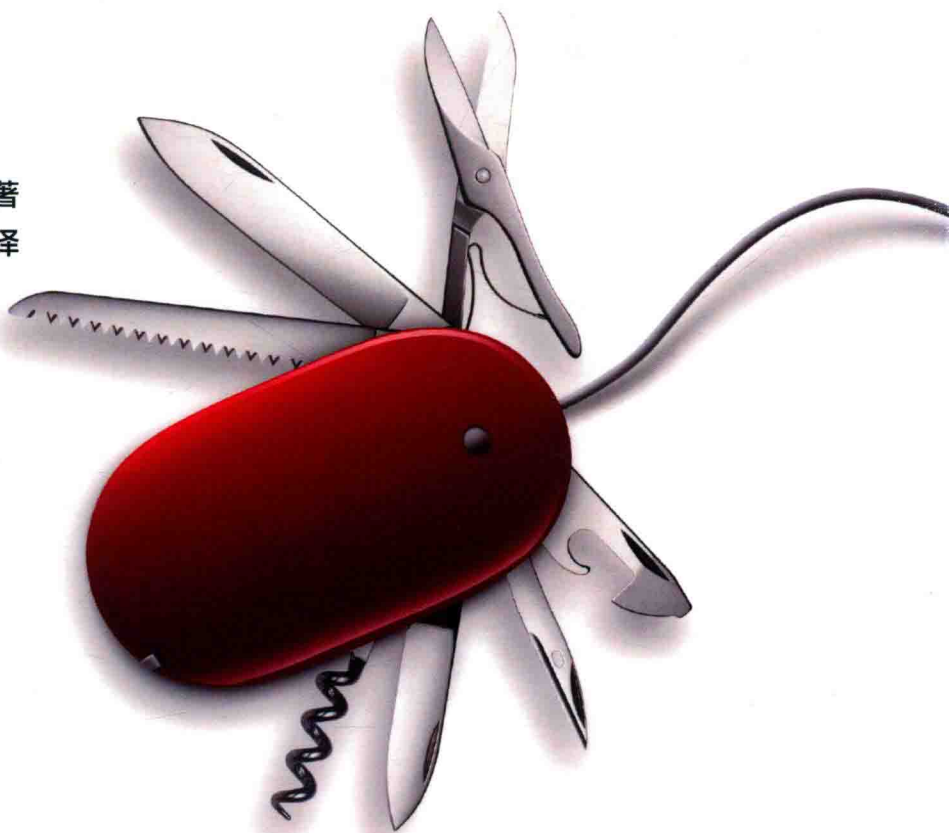
# 亲爱的界面

## 让用户乐于使用、爱不释手

用户体验设计百科全书

[瑞士] Lukas Mathis 著

杨文梁 译



中国工信出版集团



人民邮电出版社  
POSTS & TELECOM PRESS

TURING

图灵程序设计丛书

# Designed for Use

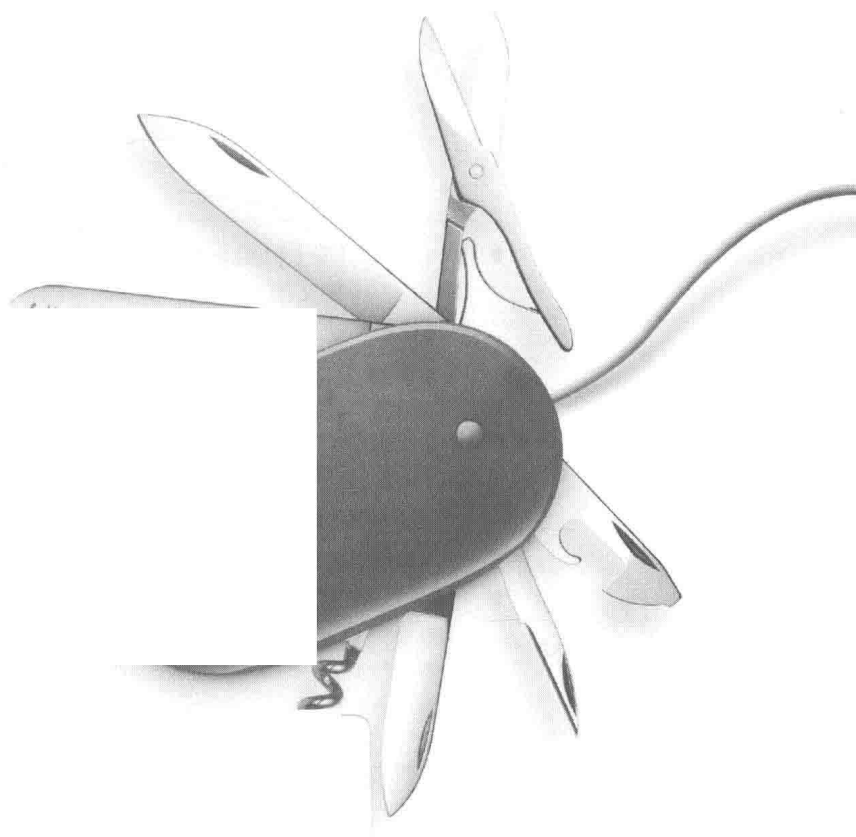
Create Usable Interfaces for Applications and the Web, Second Edition

# 亲爱的界面 (第2版)

## 让用户乐于使用、爱不释手

[瑞士] Lukas Mathis 著

杨文梁 译



人民邮电出版社

北 京

## 图书在版编目 (C I P) 数据

亲爱的界面：让用户乐于使用、爱不释手：第2版 /  
(瑞士) 卢卡斯·马西斯 (Lukas Mathis) 著；杨文梁译.  
— 北京：人民邮电出版社，2018.1  
(图灵交互设计丛书)  
ISBN 978-7-115-47166-6

I. ①亲… II. ①卢… ②杨… III. ①人机界面—程  
序设计 IV. ①TP311.1

中国版本图书馆CIP数据核字(2017)第268898号

## 内 容 提 要

本书主要介绍如何设计出具有可用性的应用程序和网站，不但让用户乐于使用，而且令其爱不释手。可用性是设计大厦的基石，作者详细介绍了如何将可用性融入设计、测试及开发的各个流程，如何优化设计过程、把握设计重点、提高设计效率。另外，作者在本书中给出了大量宝贵建议，传授了用户界面的设计与评估技巧，提供了独具匠心的设计视角。

本书适用于设计人员、开发人员、产品经理、专业软件开发工程师和用户界面设计师。

- 
- ◆ 著 [瑞士] Lukas Mathis
  - 译 杨文梁
  - 责任编辑 岳新欣
  - 执行编辑 张 憬
  - 责任印制 彭志环
  - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
  - 邮编 100164 电子邮件 315@ptpress.com.cn
  - 网址 <http://www.ptpress.com.cn>
  - 大厂聚鑫印刷有限责任公司印刷
  - ◆ 开本：800×1000 1/16
  - 印张：15.75
  - 字数：372千字 2018年1月第1版
  - 印数：1—3 500册 2018年1月河北第1次印刷
  - 著作权合同登记号 图字：01-2017-0674号
- 

定价：59.00元

读者服务热线：(010)51095186转600 印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京东工商广登字 20170147 号

站在巨人的肩上  
**Standing on Shoulders of Giants**



iTuring.cn

# 版权声明

Copyright © 2016 The Pragmatic Programmers, LLC. Original English language edition, entitled *Designed for Use, Second Edition*.

Simplified Chinese-language edition copyright © 2018 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 The Pragmatic Programmers, LLC 授权人民邮电出版社独家出版。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。

版权所有，侵权必究。

献给 Regula 和 Werner。

# 前言

本书适合软件经理、设计师以及开发工程师阅读。本书并不关心具体的管理、视觉设计或者程序代码，而关注更为重要的用户，也就是使用产品的人。

没有用户使用，产品再好也毫无意义。这就好比你创造了世界上最美丽、最结实、最优雅的画笔，但是只要没人用它来绘画，你的工作就是徒劳的。

本书将帮助你设计用户乐于使用的网站和应用。

本书包含两类章节，一类关乎技术，另一类关乎理念。技术类章节将阐述具体的技术，如故事板、可用性测试、纸面原型。你可以在设计产品的过程中使用这些技术，让产品对用户更加友好。可以说，这类章节介绍的是具体的设计工具。

理念类章节则宏观地探讨观点和概念，比如：如何编写可用的文本，设计的实际效果如何，何时使用动画，等等。这类章节涵盖了在设计过程中需要考虑的事情。

## 技术类章节



技术类章节的章首有小齿轮图标。

所有的技术章节都使用相同的基本提纲。并非每种技术都适用于所有的情况，所以对于每种技术，都提前标注了应用场景，随后才描述它的定义和使用方法。本书将设计一个虚拟的应用程序，一些技术章节的最后会附加此技术在这个程序中的具体应用。

众所周知，Twitter<sup>①</sup>的应用是我们这一代最经典的“Hello World”示例应用程序。因此，我们将在技术类章节中设计一个Twitter应用。为了增加趣味性，我们不会单纯设计一个普通的Twitter应用。这个应用程序的目标用户是需要为公司Twitter账号更新内容的人。我们将这个虚拟的应用程序命名为：BizTwit。

你可以将技术类章节当作攻略指南，可以从头到尾完整阅读一遍，当然也可以研读某个特定

---

<sup>①</sup> 这里解释一下，因为你有可能不知道Twitter是什么。（也许你是在2053年读到了这本书，没准那个时候人类已经可以像控制计算机一样控制大脑，以此进行交流，所以不知道Twitter很正常。）Twitter（<http://twitter.com>）是红极一时的网络服务，人们使用它发布短消息，还能订阅他人的消息。

的话题。为了方便阅读，这些章节都短小精悍、简明扼要，同时会指出本书中、其他书籍中或互联网上出现的更多信息。

### 如何使用本书中出现的网址

本书包含很多网址，其中一些还很长。如果阅读的是纸质书，那么你会觉得将这些长地址键入浏览器中很麻烦。为了简化操作，我创建了一个站点，<http://designedforuse.net>，罗列出了书中出现的所有长地址。这样你就不用逐字键入这些长地址了，访问<http://designedforuse.net>，点击相应链接即可。

## 理念类章节

技术类章节自始至终都在介绍具体的技术和实践方法，而理念类章节则比较抽象。理念类章节主要介绍概念，这些概念更像是灵感的源泉，而不是严格的规则。某些理念类章节也会涉及技术类章节的知识，但是关注点却是那些更基础的概念，比如：什么样的设计才切实可行？怎样使用动画才最有效？什么是模式？从视频游戏中可以学到什么？

### I

理念类章节的章首有灯泡图标。

这些理念不一定适用于你的项目，因为在某种程度上，人总是难以捉摸。你的产品使用起来和你期望的效果并不一定一致，而且有时会违背你的经验法则。

人的行为通常出乎意料，要说明这一点，让我们来看一个真实的案例。假定你研究的方向是公共健康及安全。你将如何着手？鉴于每年有几万骑自行车的人在车祸中受伤，自行车行车安全是个不错的切入点。

研究显示，安全头盔可以帮助骑车的人免于车祸伤害。因此，让骑车的人戴上头盔应该可以减少受伤者的数量，这样自然就提高了公众的健康和安全指数。预期的结果是显而易见的：人遭遇自行车车祸时，头盔可以避免骑车的人受伤，所以戴了自行车头盔的人不会受伤。结论：应当强制骑车的人戴头盔。

这些年来，一系列要求佩戴自行车头盔的法案出台。然而，这些法案并没有起到预期的效果。

2009年，澳大利亚麦考瑞大学精算学系的Piet de Jong发表了一篇名为“强制实施的自行车头盔法案对健康的影响”（The Health Impact of Mandatory Bicycle Helmet Laws）<sup>①</sup>的论文，评估了这些法案的影响。文中指出，人们根本就不喜欢戴自行车头盔，所以被强制要求戴头盔才能骑车时，很多人彻底放弃了使用自行车。

<sup>①</sup> 你可以在<http://ssrn.com/abstract=1368064>阅读此论文。

这一结果让de Jong得出以下结论：自行车头盔法案实际上对公众安全产生了负面影响。没错，这些法案是避免了一些伤害，但是对于那些完全放弃使用自行车（取而代之开汽车）的人来说，他们所遭受的健康影响全然是负面的。

归根结底，问题在于这些法案在实施之前没有经过检验。人们在法案影响下的行为完全超出了法案制定者的预期。

用户界面的设计也经常会遇到类似的情况。设计的改版经常并未如你所愿，有时甚至与你的预期完全相反。

阅读本书中的理念和规则的时候，希望你能谨记这一点。你可以尽力提出一个可用的解决方案。在设计用户界面的时候，你可以遵循所有规则，作出看似明显可用的决策。但即使这样，用户仍然会出乎意料地误解你的用户界面，也有可能在你的网站中迷失方向，或者做出一些不可预测的、看上去不合逻辑的举动，不能完成那些最简单、最显而易见的操作等。这些情况无一不让你心生惊奇。

永远不要妄想遵循可用性规则的产品就一定拥有很高的可用性。当你设计用户界面时，符合那些常识即可，但无需依赖它们。要了解这些规则，必要之时，也要打破它们。说这些，并不是要你照猫画虎，完全按我说的去做，重点是给你提供一些思路与灵感，之后检验你的设计。

## 本书结构

我将典型的设计过程分为三个阶段：调研、设计、实现。本书章节即大致遵循此顺序组织的。

### 调研

跳过这一阶段直接开始产品设计（开发人员甚至可以直接开始写代码）是很诱人的。在某些场景中，这样做没问题，但是通常来说最好做一些前期调研，比如：产品的目标用户是谁？解决哪些痛点问题？

### 设计

思考如何解决用户的问题。设计解决方案，然后在写代码之前测试它们。在纸上修改错误远比在代码中修复bug容易得多。

从设计的角度来说，这一阶段可能是开发过程中最重要的一环。因此，这部分内容在书中也占据了最大的篇幅。

### 实现

创建项目并持续测试。测试的具体内容包括：你早期的假定是否正确？你的设计是否起作用？程序运行时用户如何与之交互？实现的效果如何？你的产品有怎样的容错机制以及如何处理真实数据？产品表现如何？

我主要凭感觉决定将理念类章节置于何处，你会觉得那是恰到好处的，而且与技术类章节完美结合。

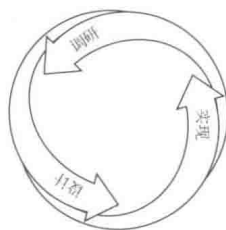
与此同时，每个技术章节都会先给出如下时间线。



这个时间线可以帮助你了解这种技术在哪个阶段最重要或者应用最广泛。上面所示的示例时间线就指出了，某种技术通常在产品开发过程中实现阶段的最开始使用。然而，很多技术适用于开发过程的不同阶段。时间线可以很好地对它们进行归类，当然这不是严格意义上的规定。

以上描述使典型的开发过程看起来像是线性的：从调研到设计再到实现。但实际上，设计过程通常是迭代的。产品开发的流程看上去更像一个圆圈。

严格来讲，在产品的生命周期内，开发是一系列的线性迭代过程，我引入的线性时间线概念应该更容易理解。



## 补充一点

在我们开始前，需要说明的是，本书有自己的主页：<http://www.pragprog.com/titles/lmuse2>。主页内提供了论坛以及勘误表。当然，在我写下这段话时，勘误表还是空的。但是当你访问的时候，没准就不是空的了。

好，介绍完主页，我们开始进入正题吧。

## 电子书

扫描如下二维码，即可购买本书电子版。



# 致 谢

有很多人帮助我完成了这本书。我将尽自己所能将大家都列出来，如有疏漏，请接受我最诚挚的歉意，我将赠送一盒瑞士巧克力。

至于错误，本书中所有的错误当然都是我造成的，下面要提到的这些热心的人不应该为这些错误买单。

首先，我想衷心感谢我的编辑Jill Steinberg，是他将我语无伦次的表述修改得流畅可读。我还要感谢另一位编辑，无与伦比的Michael Swaine，是他编辑了本书的第2版（就是你现在手里拿的这本）。

我还想感谢我的技术编辑，他们为本书校正了很多错误，并且提供了大量令人称奇的反馈意见和想法，他们是（排名不分先后）——

Evernote的设计师Keith Lang。

设计教育公司UX Launchpad（<http://uxlaunchpad.com>）的联合创始人Jon Bell。

诗人、短篇小说家Charlotte Simmonds。你可以找到更多有关其作品*The World's Fastest Flower*的信息（<http://vup.victoria.ac.nz/worlds-fastest-flower-the/>）。

为Mac、iPhone以及iPad开发软件的工程师Duncan Wilcox。欢迎关注他的博客：<http://duncanwilcox.com>。

旧金山的澳大利亚籍交互设计师Chris Clark，别名Clarko。欢迎关注他的主页：<http://twitter.com/clarko>。

专注于教育学的心理学硕士Sharmila Egger。在编辑本书的同时，她还帮我做了调研。

常驻企业的平面和UI设计师Max Steenbergen，他为豪华游艇上所用的软件设计UI和图形元素。欢迎关注他的博客<http://maxsteenbergen.com>，你还可以通过Twitter联系他：@maxsteenbergen。

瑞士数字咨询和品牌公司atedo（<http://www.atedo.name>）的联合创始人兼创意总监David Naef。

为银行开发风险管理工具的专家开发工程师Sibylle Aregger（可惜她没开博客）。

Numcom Software AG的Appway高级项目经理Michael Trummer。你可以在<http://ch.linkedin>。

com/in/trummer找到他，也可以在<http://twitter.com/DrummeratWork>读到他关于商业应用开发的一些想法。

Appway用户体验顾问Adriana Ispas。她还参与组织了苏黎世前端会议。她的Twitter地址是<http://twitter.com/aPepperNote>。

西雅图软件设计师、作家以及教师William Van Hecke。欢迎关注他的主页：<http://metalbat.com>。

感觉自己没什么了不起的UX设计师兼玉米煎饼爱好者Brian Plaum。欢迎关注他的主页：[www.brianplaum.com](http://www.brianplaum.com)。

我还想感谢Chris Pruett（欢迎关注他的主页：<http://replicaisland.blogspot.com>），他准许我在第29章使用他的《小绿人历险记》游戏中的死亡热图。还有Clayton Miller（欢迎关注他的主页：<http://relayton.net>），他允许我在第22章中使用他的“延迟被动确认”模型。

来自<http://zuendung.ch>的Amadé Fries拍摄了第26章中的两张汽车图片，并且很爽快地让我在本书中使用。

我要感谢Lynx的联合设计师RJ Mical，他回答了我关于Lynx电子游戏控制台开发的问题（在第1章中提到过）。欢迎关注他的个人主页：<http://www.mical.org>。

Bohemian Coding的Pieter Omvlee跟我谈过他从DrawIt中删除功能的经历，为我撰写第28章帮了大忙。Bohemian Coding的网址是<http://www.bohemiancoding.com>。

我要感谢Amanda Kiefer帮我为本书做了心理学调研。

还有很多人允许我引用他们的著作或者博文，帮助我顺利完成了本书，我要感谢他们所有人。

还有很多人允许我在本书的插图中使用他们的Twitter图片、账号名称以及推文。他们的Twitter账号（按字母顺序）分别是：7WP、aidanhornsby、AlphabUX、CoryCalifornia、designdare、fetjuel、gauravmishr、ienjoy、jonbell、larryv、lorentey、louije、maxsteenbergen、neave、oliverw、shawnblanc、thibautsailly、timeboxed以及workjon。感谢你们！

最后，我想感谢Damaris Kiefer。她为本书拍摄了很多照片，还在其中一些照片中出境了。

# 目 录

## 第一部分 调研

### 第 1 章 用户调研.....2

要想设计适合用户的产品，你需要了解用户。有一些工具可以协助你完成这项任务，有些则会误导你。你要学会辨别它们。

### 第 2 章 功能不是需求.....6

对于如何解决问题，用户常常自有一套具体的想法。你不能一味听取意见建议，而要探究隐藏在背后的真正需求。

### 第 3 章 工作影子和情境访谈.....9

要解决问题，先了解问题。要开发一个能够帮助他人的应用，先查明用户需要何种帮助。本章将介绍完成这项工作的常用技巧。

### 第 4 章 用户角色模型.....14

当你完成调研并且了解了产品用户，就需要将这些信息导入设计工具中。Personas 正是这样一个工具，它可以帮助你在设计过程中运用用户调研的结果。

### 第 5 章 以活动为中心的设计.....19

有时，关注人们要做什么，有什么目标，比关注他们是谁更有意义。你要关注具体情境，并学会如何用设计解决问题。

### 第 6 章 开始编写文档.....22

不要小看使用手册。尽早考虑为应用撰写文档，可以使设计思路更加清晰。如果应用很难描述，那很可能代表它很难用。

## 第 7 章 文案的可用性.....26

尽管我们使用的都是高分辨率的图形设备，文案仍然是与用户交互的最重要一环。确保文案准确为用户获得优质的体验铺平了道路。如果文案有误，再优秀的图形设计也不能挽救你的产品。

## 第 8 章 用户界面设计的层级.....36

要使用层级嵌套的 UI 元素来构建用户界面，比如视窗应该包含头部、尾部以及其他区域，按钮要置于工具条中，图标列表放在同一视图中。对你的用户来说，视觉的层级隐含着 UI 元素的关系。正确的层级关系可以让用户更快理解产品，而不会不知所措。

## 第 9 章 卡片分类.....40

卡片分类可以帮助你了解用户认为产品的功能怎样排列才最合理。换句话说，它可以告诉你，怎么组织产品才对用户最有益。

## 第 10 章 创建可用的层级结构.....47

信息架构的主要任务是对产品中的内容进行组织和标识，这并不简单。本章将提供一些指导，协助你着手进行信息架构。

## 第 11 章 心智模型.....51

尽可能让产品的实际效果和用户的预期保持一致，是产品实现高可用性的关键。

# 第二部分 设计

## 第 12 章 保持开放的心态.....66

不要过快地倾心于某些设计思想。你总能找到更好的解决方案。保持开放的心态，通常比过早决定使用并完善某个设计方案要好。你无需担心扼杀了已有的好方案。

## 第 13 章 草图和原型.....70

将设计原型用纸或者专用的程序画出来，有助于在付诸开发之前明确设计方向。流程图、故事板、草图、线框图、实物模型可以协助你深入思考应用的设计，设想用户如何使用它，以及用户如何与他人交流、协作。

## 第 14 章 纸面原型测试.....77

无需等到产品运行的时候才开始可用性测试。即使你只有几张草图都没问题，这些足够了。

## 第 15 章 写实主义.....88

有真实感的用户界面看上去很棒，但有时候，添加过多的细节反而会在本质上损害产品，让其变得难用。

## 第 16 章 自然用户界面.....96

自然用户界面摒弃了图形用户界面的传统规则，它的交互规则是基于真实世界的。正因如此，界面会更加易懂易用。本章将阐明如何做到这一点。

## 第 17 章 费茨法则.....102

区域越大，越容易点击和触摸，但是，区域是不是可以无限大呢？

## 第 18 章 动画.....108

动画可以帮助用户理解程序如何工作，并且极大地提升用户体验。本章将阐述何时使用动画，以及怎样设计动画。

## 第 19 章 一致性.....116

缺乏一致性通常是用户界面中广受用户诟病的一点。何为一致性？对于某些场景而言，缺乏一致性是否反而会产生有利的影响？

## 第 20 章 可发现性.....120

对于用户来说，找不到某个功能与不存在这个功能是一回事。开发功能需要工作量和金钱，所以你要确保用户能发现它们。

## 第 21 章 不要打断用户.....125

打断用户是粗鲁的，计算机也不该这样做。这样不礼貌且不必要。你要学会如何避免打断用户。

## 第 22 章 使用撤销代替打断 ..... 129

最佳用户体验的一个因素是：让用户信任你的应用。也就是说，允许他们犯错误，同时也要从容、快捷、安全地处理他们不可避免的错误。

## 第 23 章 模式 ..... 132

拿起的瞬间，一支铅笔变成了一个油漆罐——你当然不可能遇到这种事。这是因为现实世界不存在模型，铅笔就是铅笔。然而在虚拟世界中，事物可以有各种不同的形态，这就是个问题了。

## 第 24 章 避免使用首选项 ..... 138

需要做选择的时候，最简单的办法就是将问题抛给用户。然而你的用户不是设计师，你才是。不要将设计选型的问题推卸给用户。

## 第 25 章 层级结构、空间、时间，以及我们认知世界的方式 ..... 143

计算机擅长在层级系统中存储数据，人类则倾向于通过空间和时间认识世界。大多数产品强制人类像计算机一样处理事物，而不是让计算机像人类想象的那样工作。让你的产品表现得更像人类的一员，而不是冷冰冰的数字系统，这样对用户更加友好。

## 第 26 章 速度 ..... 150

应用程序的速度听上去属于技术细节，与设计无关。但是，快速施展性能、进行响应对用户体验是大有裨益的，当然，有时（极少情况），过快也会产生负面影响。

## 第 27 章 避免加入过多新功能 ..... 155

开发新功能满足用户的需求通常会让设计师和开发工程师很有成就感，但这不是最优的选择。慎之又慎地增加新功能可以使产品体验更佳、寿命更长久。

## 第 28 章 移除功能 ..... 162

再资深的设计师也会犯错，比如新增了功能，结果发现不但没效果而且是个累赘。但是移除这些功能又很痛苦。本章将教你如何减轻这样的痛苦。

## 第 29 章 向电子游戏学习 ..... 166

游戏常常让人愉悦，应用程序也当如此。程序设计可以从游戏设计中借鉴很多经验。但是，要小心，有时候游戏表里不一，很容易让你误入歧途。

## 第三部分 实现

### 第 30 章 后端设计 .....178

设计师不关注具体的代码实现，但是需要参与后台的设计。如果后台的设计和前端用户体验不一致，那程序肯定不会正常运行。

### 第 31 章 打游击式的可用性测试 .....181

可用性测试可以花销极少，但不可能没有花销。有没有可能以更少的投入得到想要的测试数据呢？回答是肯定的，我们能！

### 第 32 章 首次运行体验 .....185

设计软件时，我们考虑的都是司空见惯的场景。我们将样本数据填入线框图，将文字填入样机。但是，如果用户是第一次打开这个软件，它还是空的，这时候会发生什么呢？

### 第 33 章 可用性测试 .....189

第 14 章提到过可用性测试。现在，是时候对正在运行的程序进行实实在在的测试了。本章将阐述相关流程。

### 第 34 章 面对面测试 .....196

大家都熟悉的面对面可用性测试可以查明软件中存在的可用性问题。本章将阐述如何进行这种测试。

### 第 35 章 远程测试 .....202

没有条件或者很难邀请别人进行可用性测试的时候，也不要烦恼，因为无需邀请他们到现场，远程进行测试也是可行的。

### 第 36 章 测试中常见的错误做法 .....210

可用性测试即使做得很差也大有帮助。当然，测试做得越好，结果也就越棒。你需要学着怎么避免那些常见的错误。

### 第 37 章 用户错误即设计错误 .....213

发现用户不会使用产品，最不理智的做法就是责怪他们，这毫无意义。用户所犯的错误归根结底都源自设计纰漏。本章将阐述如何设计足够稳健的程序，包容用户的错误。