Broadview
www.broadview.com.cn

FK 疯狂软件教育 www.fkit.org 标准教材

疯狂iOS讲义

（提高篇）

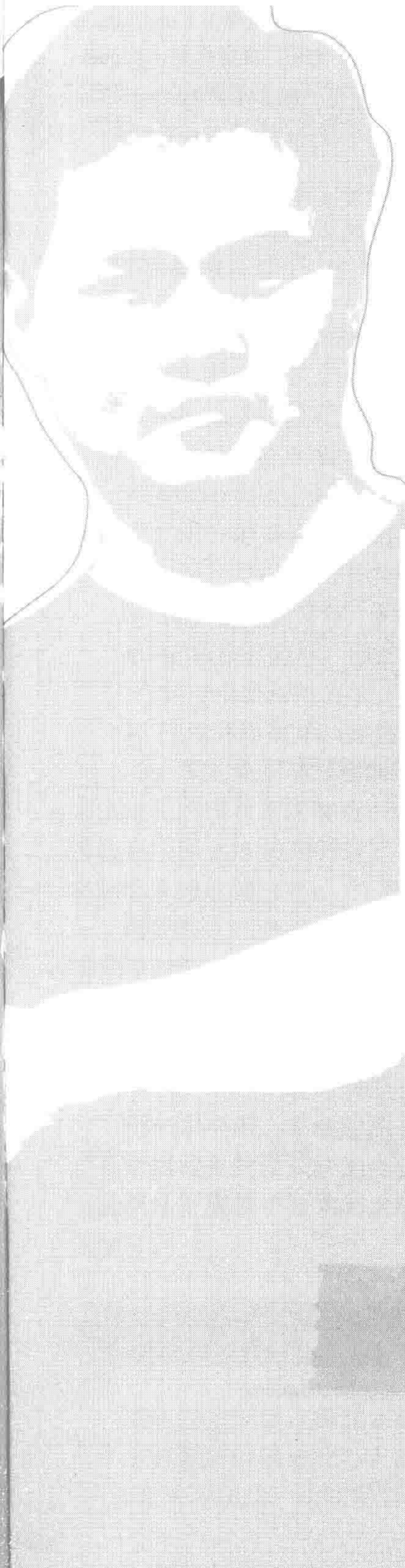
李刚 编著

疯狂源自梦想

技术成就辉煌

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn



疯狂iOS讲义

（提高篇）

李 刚 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书基于《疯狂 iOS 讲义（下）》升级而来，已真正升级成 Swift、Objective-C 双语讲解。本书所有案例全部提供 Swift、Objective-C 两个语言的版本。本书基于最新的 iOS 9.3、Xcode 7.3 平台，全面介绍 iOS 应用开发的高级知识。本书包含多点触摸与手势处理、应用国际化、数据存储、SQLite 数据库与 Core Data、iOS 多媒体开发、加速计和陀螺仪、多线程、网络编程、XMPP 即时通信、定位、地图、推送机制、iCloud 服务、HealthKit 框架等内容，本书最后还通过疯狂软件商城 App 整合介绍了 iOS App 的界面开发、自定义 UI 控件、手势处理、网络通信、本地数据存储、在线支付等功能的用法。读者在阅读本书之前，建议先掌握《疯狂 iOS 讲义（基础篇）》中的知识。

本书并不局限于介绍 iOS 编程的各种理论知识，而是从“项目驱动”的角度来讲授理论，全书一共包括近百个实例，这些示范性的实例既可帮助读者更好地理解各知识点在实际开发中的应用，也可供读者在实际开发时作为参考、拿来就用。如果读者在阅读本书时遇到技术问题，可以登录疯狂 Java 联盟（http://www.crazyit.org）发帖，笔者将会及时予以解答。

本书为所有打算深入掌握 iOS 编程的读者而编写，适合各种层次的 iOS 学习者和开发者阅读，也适合作为大学教育、培训机构的 iOS 教材。

图书在版编目（CIP）数据

疯狂 iOS 讲义. 提高篇 / 李刚编著. —北京：电子工业出版社，2016.8
ISBN 978-7-121-29700-7

Ⅰ. ①疯… Ⅱ. ①李… Ⅲ. ①移动终端－应用程序－程序设计 Ⅳ. ①TN929.53

中国版本图书馆 CIP 数据核字（2016）第 190887 号

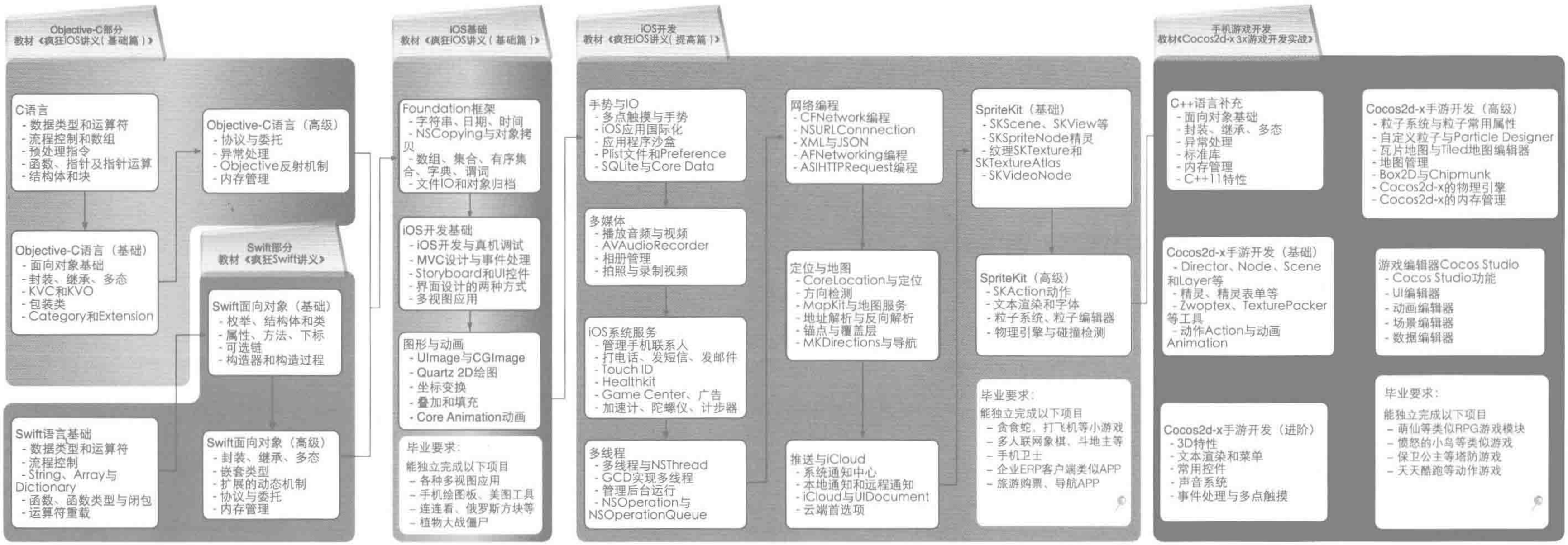
策划编辑：张月萍
责任编辑：葛　娜
印　　刷：三河市华成印务有限公司
装　　订：三河市华成印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×1092　1/16　　印张：36.75　　字数：1036 千字　　彩插：2
版　　次：2016 年 8 月第 1 版
印　　次：2016 年 8 月第 1 次印刷
定　　价：99.00 元（含光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819　faq@phei.com.cn。

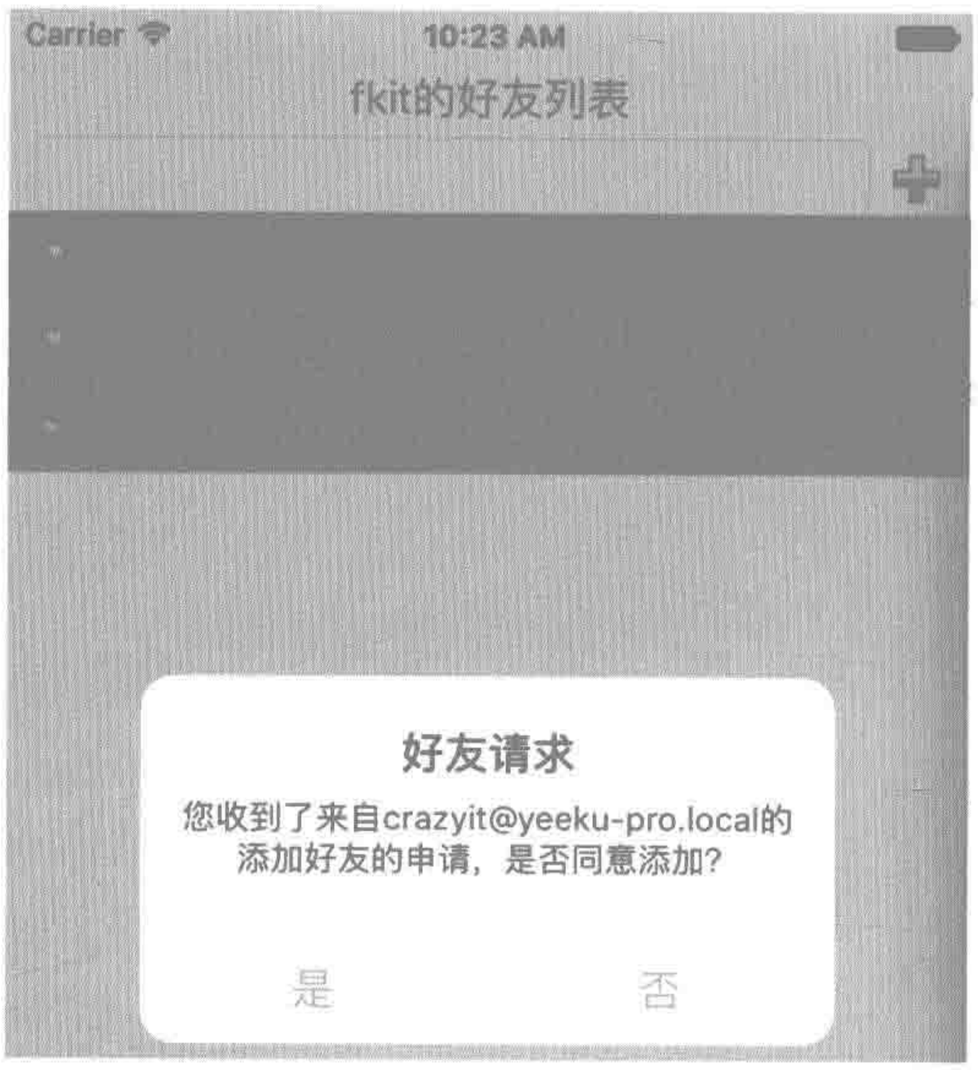
疯狂iOS学习路线图


Objective-C部分
教材《疯狂iOS讲义（基础篇）》
C语言
- 数据类型和运算符
- 流程控制和数组
- 预处理指令
- 函数、指针及指针运算
- 结构体和块
Objective-C语言（高级）
- 协议与委托
- 异常处理
- Objective反射机制
- 内存管理
Objective-C语言（基础）
- 面向对象基础
- 封装、继承、多态
- KVC和KVO
- 包装类
- Category和Extension
Swift部分
教材《疯狂Swift讲义》
Swift面向对象（基础）
- 枚举、结构体和类
- 属性、方法、下标
- 可选链
- 构造器和构造过程
Swift语言基础
- 数据类型和运算符
- 流程控制
- String、Array与Dictionary
- 函数、函数类型与闭包
- 运算符重载
Swift面向对象（高级）
- 封装、继承、多态
- 嵌套类型
- 扩展的动态机制
- 协议与委托
- 内存管理
iOS基础
教材《疯狂iOS讲义（基础篇）》
Foundation框架
- 字符串、日期、时间
- NSCopying与对象拷贝
- 数组、集合、有序集合、字典、谓词
- 文件IO和对象归档
iOS开发基础
- iOS开发与真机调试
- MVC设计与事件处理
- Storyboard和UI控件
- 界面设计的两种方式
- 多视图应用
图形与动画
- UIImage与CGImage
- Quartz 2D绘图
- 坐标变换
- 叠加和填充
- Core Animation动画
毕业要求：
能独立完成以下项目
– 各种多视图应用
– 手机绘图板、美图工具
– 连连看、俄罗斯方块等
– 植物大战僵尸
iOS开发
教材《疯狂iOS讲义（提高篇）》
手势与IO
- 多点触摸与手势
- iOS应用国际化
- 应用程序沙盒
- Plist文件和Preference
- SQLite与Core Data
多媒体
- 播放音频与视频
- AVAudioRecorder
- 相册管理
- 拍照与录制视频
iOS系统服务
- 管理手机联系人
- 打电话、发短信、发邮件
- Touch ID
- Healthkit
- Game Center、广告
- 加速计、陀螺仪、计步器
多线程
- 多线程与NSThread
- GCD实现多线程
- 管理后台运行
- NSOperation与NSOperationQueue
网络编程
- CFNetwork编程
- NSURLConnnection
- XML与JSON
- AFNetworking编程
- ASIHTTPRequest编程
定位与地图
- CoreLocation与定位
- 方向检测
- MapKit与地图服务
- 地址解析与反向解析
- 锚点与覆盖层
- MKDirections与导航
推送与iCloud
- 系统通知中心
- 本地通知和远程通知
- iCloud与UIDocument
- 云端首选项
SpriteKit（基础）
- SKScene、SKView等
- SKSpriteNode精灵
- 纹理SKTexture和SKTextureAtlas
- SKVideoNode
SpriteKit（高级）
- SKAction动作
- 文本渲染和字体
- 粒子系统、粒子编辑器
- 物理引擎与碰撞检测
毕业要求：
能独立完成以下项目
– 贪食蛇、打飞机等小游戏
– 多人联网象棋、斗地主等
– 手机卫士
– 企业ERP客户端类似APP
– 旅游购票、导航APP
手机游戏开发
教材《Cocos2d-x 3x游戏开发实战》
C++语言补充
- 面向对象基础
- 封装、继承、多态
- 异常处理
- 标准库
- 内存管理
- C++11特性
Cocos2d-x手游开发（基础）
- Director、Node、Scene和Layer等
- 精灵、精灵表单等
- Zwoptex、TexturePacker等工具
- 动作Action与动画Animation
Cocos2d-x手游开发（进阶）
- 3D特性
- 文本渲染和菜单
- 常用控件
- 声音系统
- 事件处理与多点触摸
Cocos2d-x手游开发（高级）
- 粒子系统与粒子常用属性
- 自定义粒子与Particle Designer
- 瓦片地图与Tiled地图编辑器
- 地图管理
- Box2D与Chipmunk
- Cocos2d-x的物理引擎
- Cocos2d-x的内存管理
游戏编辑器Cocos Studio
- Cocos Studio功能
- UI编辑器
- 动画编辑器
- 场景编辑器
- 数据编辑器
毕业要求：
能独立完成以下项目
– 萌仙等类似RPG游戏模块
– 愤怒的小鸟等类似游戏
– 保卫公主等塔防游戏
– 天天酷跑等动作游戏

精彩手游，尽在掌控

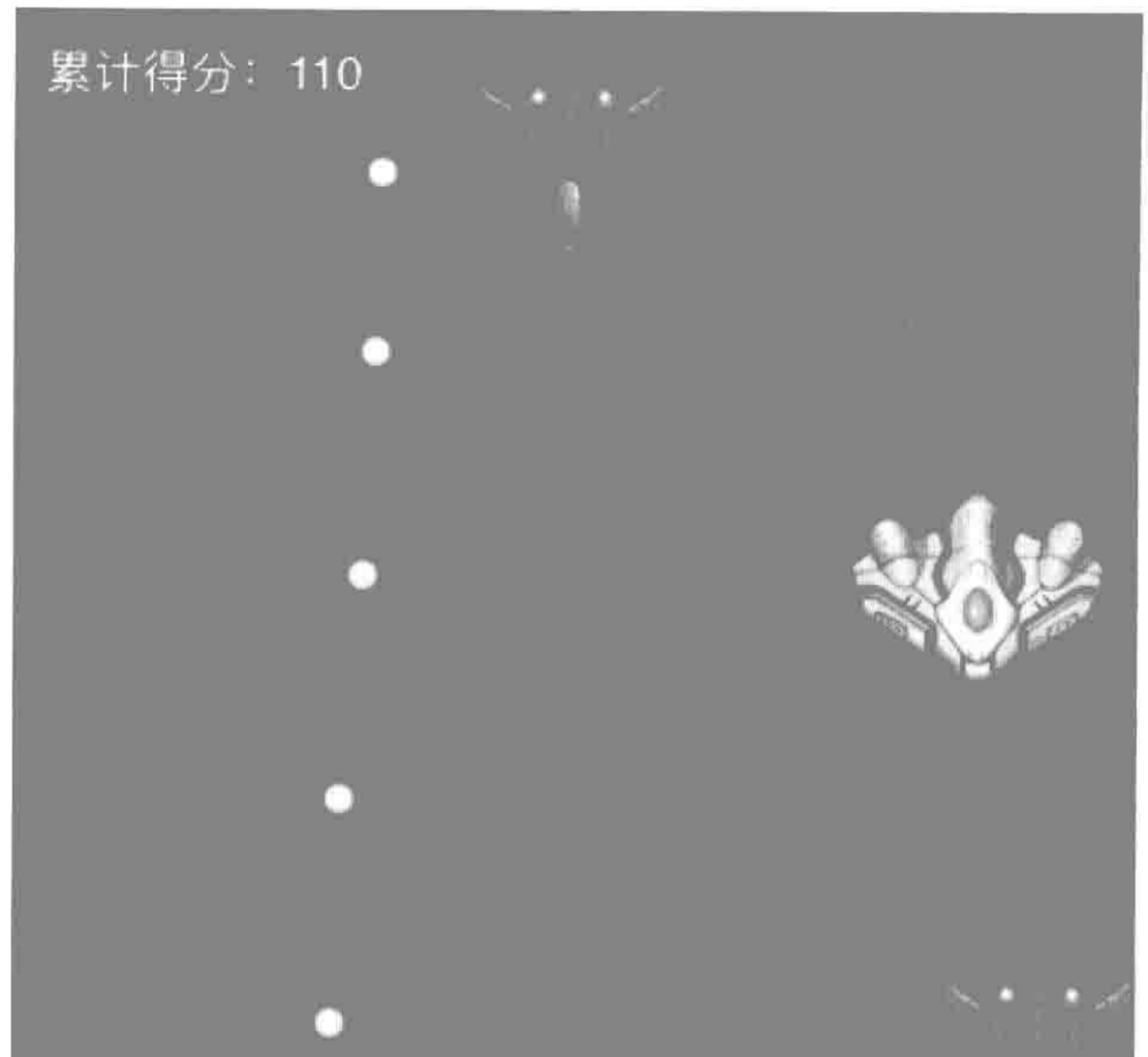
疯狂iOS讲义
（基础篇）（提高篇）

精彩案例



XMPP即时通信之好友请求

XMPP即时通信之聊天界面



打飞机

精彩手游，尽在掌控

疯狂iOS讲义（基础篇）（提高篇）

精彩案例

疯狂连连看1

疯狂连连看2

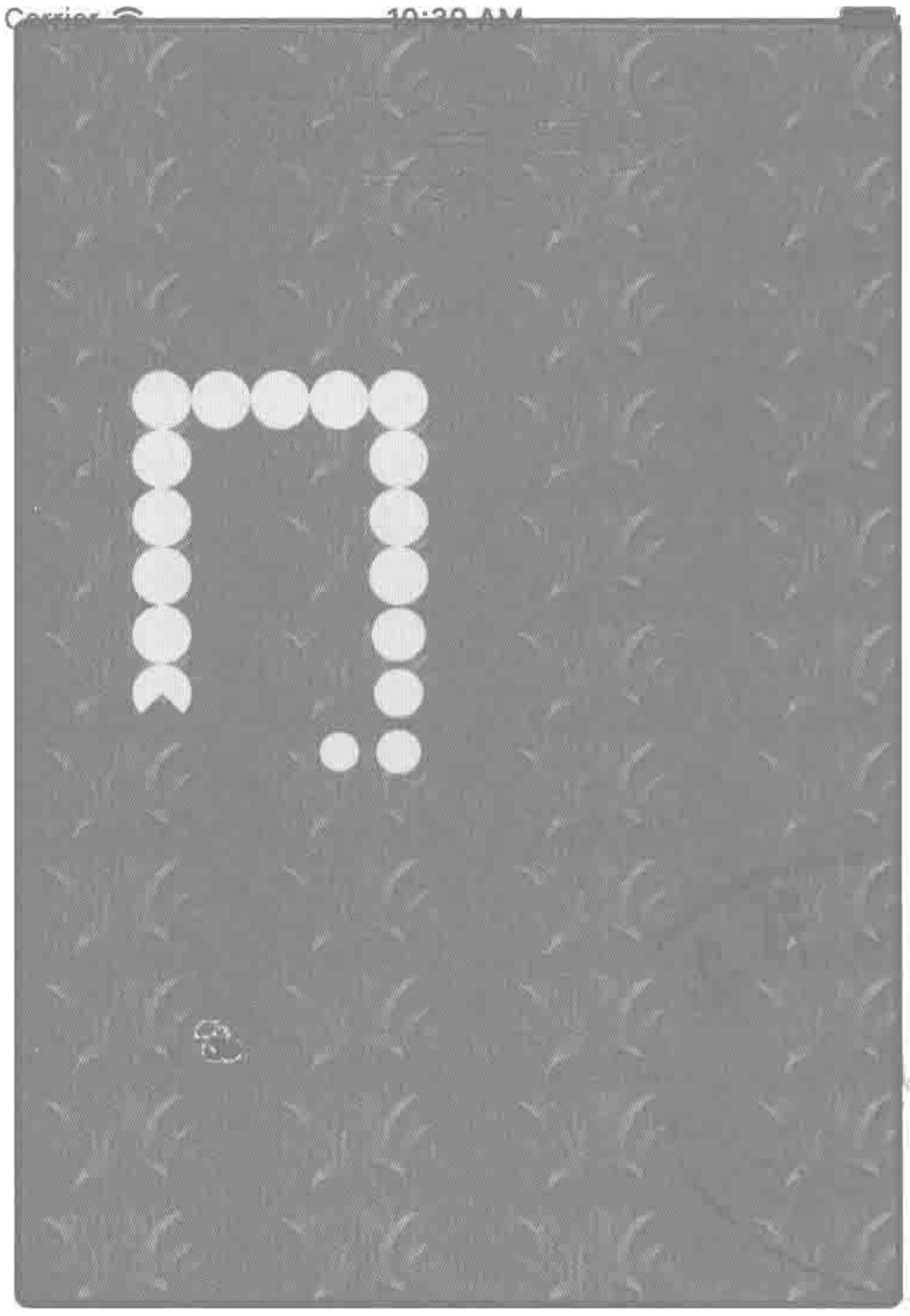
疯狂软件之确认订单

疯狂软件之物品分类

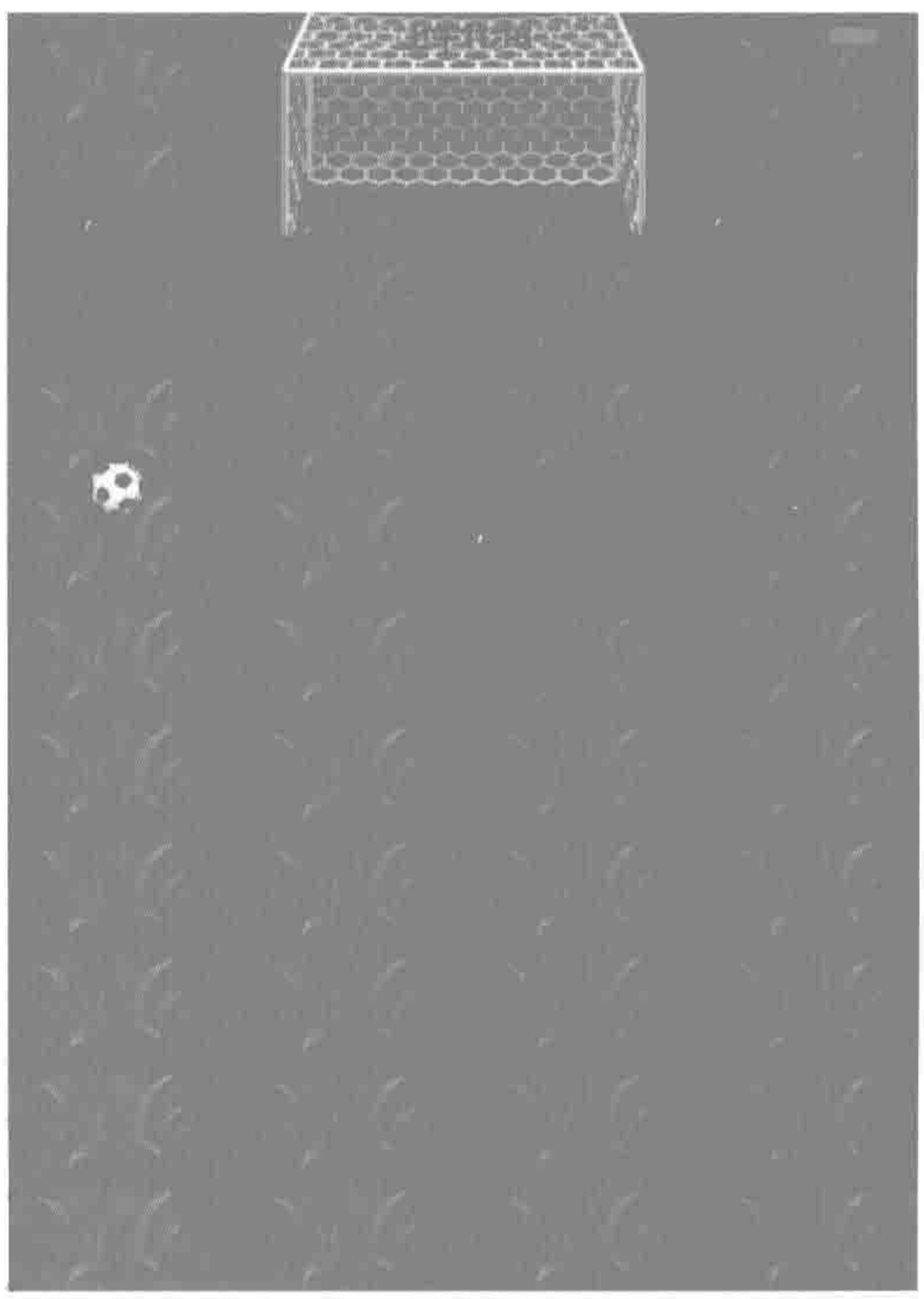
精彩手游，尽在掌控

疯狂iOS讲义（基础篇）（提高篇）

精彩案例



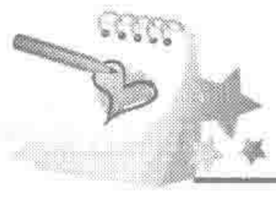
疯狂贪食蛇



怪物足球

天气预报首页

天气预报详情



前　言

本书基于《疯狂 iOS 讲义（下）》升级而来，已真正升级成 Swift、Objective-C 双语讲解。本书所有案例全部提供 Swift、Objective-C 两个语言的版本。读者通过本书可以同时掌握 Swift、Objective-C 两种语言的开发能力。

书中详细介绍了《贪食蛇》《打飞机》《天气预报》《XMPP 即时通信客户端》《疯狂软件商城 App》等大量精彩的 iOS App 的开发，本书不仅细致地介绍了开发这些应用所需的基础知识，而且真正带领读者从零开始，逐步完成这些应用。如果读者认真学完本书，并掌握了书中内容，将可以完全胜任 iOS 应用开发。

虽然本书的知识点足够深入、实用，但对于已经掌握《疯狂 iOS 讲义（基础篇）》的读者来说，本书的知识依然很容易看懂、理解，并很容易动手做出来。只要读者真正对书中的应用感兴趣，并愿意扎扎实实地动手实践，完全可以很好地学会、掌握书中这些知识点。

有些读者可能会被市面上某些薄薄的图书所吸引：他们以为阅读一本薄薄的图书更容易掌握这门技术。但实际上，无论你学还是不学，技术就在那里；无论作者写还是不写，技术就在那里。如果一本技术图书太过单薄，可能有以下三种情况：

- 技术本身就简单，只要简单介绍即可。
- 技术并不简单，但图书只介绍了该技术的皮毛，读者学起来很轻松，但实际开发时就感觉“书到用时方恨少”。
- 技术并不简单，该图书高度概括、提纲挈领地归纳该技术的要点。

iOS 开发绝不是三言两语就可以说清的一门技术，而《疯狂 iOS 讲义》并不打算只是提纲挈领地归纳该技术的要点，而是力求全面、细致地介绍 iOS 开发的相关内容，包括 Objective-C 语法介绍和 iOS 应用开发，尤其是 iOS 应用开发部分，本身包含的知识点确实比较多，读者只有全面掌握这些内容，才能真正在企业 iOS 开发岗位上游刃有余。

可能有人会觉得提纲挈领的书看上去压力没那么大，这其实是一个假象，如果仅凭某些提纲挈领的要点归纳就可以掌握一门开发技术，那只能说明其悟性很好（或有很强的技术功底），如果真是这样，那么看一本全面、细致的图书应该更易理解；反过来，如果觉得《疯狂 iOS 讲义》内容太多，不容易掌握，反而希望通过某些提纲挈领的要点归纳即可掌握这门开发技术，那更不现实——详细的书都不容易掌握，仅凭一些提纲挈领的要点归纳反而能学会？

写一本全面、系统的图书比写一本薄薄的书辛苦多了！如果只是写一本薄薄的书，可以随便介绍点常见的知识，加点例子就行了。但如果打算全面、系统地讲清楚一门技术，就需要高屋建瓴地把握整个技术的大纲，并理清各知识点之间的关联，然后还需要逐个介绍每个知识点的细节。

本书创作感言

多年过去以后，疯狂系列图书用事实证明：国内图书的质量，其实也可以很好，关键在于作者是否愿意付出全部心血去创作一本图书。《疯狂 Android 讲义》《疯狂 Java 讲义》《轻量级 Java EE 企业应用实战》这些图书在亚马逊、京东、当当等网店的销量排行榜的事实证明了国内原创

图书更适合中国开发者。

但总有些人说：还是国外的书更好——这里无意贬低国外引进的优秀图书。无论国外还是国内都可以产生优秀的图书，根据区域划分图书质量本身就是简单、粗暴的。国外作者固然创作了大量优秀的技术图书，但国内同样有一些作者在努力坚持。

“还是国外的书好”这句话之所以有人说，无非有两个原因：

- 从整体质量来看，国内原创的、优秀的图书数量确实比国外的优秀图书数量少——因此从统计学的角度来看，“还是国外的书好”这句话有一定的道理。
- 从心理学分析，说这句话的人往往是嫉妒：这些人觉得自己很行，但要么不能创作图书，要么创作的图书市场反映很差。这些人为了找回心理的平衡，不断地告诉自己：国内图书都很差，这样就可以把失败的原因归咎于所处的环境，同时又可打压周围的优秀者，从而获得心理安慰。

无论上面哪种原因，其实都应该成为国内程序员奋斗的动力。正因为国内原创的、优秀的图书数量较少，所以我们更应该努力，因为我们也是“国内程序员”的成员，提高国内软件行业的水平，提高国内原创、优秀图书的数量，国内每个软件从业人员责无旁贷。

心怀嫉妒的人，身边总是无处不在：在这些人口中，张三做的某某事很容易，李四做的某某事很容易，但这些人自己啥也不做。对于技术图书的作者而言，既不需要政府背景，也没有资金准入门槛，唯一的要求是掌握这门技术，并愿意努力介绍它——积极的人，应该通过努力把事情做好来证明自己，而不是通过语言否定别人来证明自己。

当然，如果有人坚持认为：国外图书具有更好的“思想”，那么他胜了——因为“思想”这东西，太难说清了，而本书只是一本介绍编程的工具书，并不是一本“形而上”的哲学书。如果读者想学习 Swift 或 Objective-C 编程，想在 iPhone、iPad 上开发应用，可以选择本书；如果读者想通过编程进而悟道、领悟人生哲学，应该放弃选择本书。

本书有什么特点

本书以《疯狂 iOS 讲义（基础篇）》为基础，因此本书不包含 Objective-C 语法、iOS 界面编程、图形图像处理等基础知识。

本书包含多点触摸与手势处理、应用国际化、数据存储、SQLite 数据库与 Core Data、iOS 多媒体开发、加速计和陀螺仪、多线程、网络编程、XMPP 即时通信、定位、地图、推送机制、iCloud 服务等内容，本书还穿插介绍了各种实用的 iOS App。

认真看完本书，把书中所有实例都练习一遍，本书带给你的只是 9 个字：“看得懂、学得会、做得出”。本书不能让你认识一堆新名词，只会让你学会实际的 iOS 应用开发。

本书还具有如下几个特点：

1. 知识全面，覆盖面广

不得不说，本书是市面上所有 iOS 图书中内容较为全面、体系庞大的。本书基础篇、提高篇全方位地涵盖了 Objective-C 语法、iOS 开发基础知识、iOS 开发高级知识等内容。这些知识将带领读者从零开始，逐步掌握 iOS 开发的基础，直至步入 iOS 开发殿堂。

2. 内容实际，实用性强

本书并不局限于枯燥的理论介绍，而是采用了“项目驱动”的方式来讲授知识点，全书近百个实例，几乎每个知识点都可找到对应的参考实例。还提供了《贪食蛇》《打飞机》《天气预

报》《XMPP 即时通信客户端》《疯狂软件商城 App》等真实的应用，具有极高的参考价值。

3. 讲解详细，上手容易

本书保持了“疯狂系列”的一贯风格：操作步骤详细、编程思路清晰、语言平实。只要读者有学习的决心和毅力，认真掌握本书基础篇、提高篇的知识，即可完全胜任实际企业中 iOS 开发者的要求。

不管怎样，只要读者在阅读本书时遇到知识上的问题，都可以登录疯狂 Java 联盟（http://www.crazyit.org）与广大学习者交流，笔者也会通过该平台与大家一起交流、学习。

本书写给谁看

如果你已经具备一定的 iOS 应用开发基础，或已经学完了《疯狂 iOS 讲义（基础篇）》一书，那么将非常适合阅读此书；如果你对 Objective-C、Swift 的掌握还不熟练，比如对 Xcode 安装、Objective-C、Swift 基本语法都不熟练，建议遵从学习规律，循序渐进，暂时不要购买、阅读此书。

2016-5-20

目 录 CONTENTS

CHAPTER

1

第 1 章
多点触摸与手势检测

本章要点

- 响应者链
- 响应触碰方法
- 使用 UITapGestureRecognizer 处理点击手势
- 使用 UIPinchGestureRecognizer 处理捏合手势
- 使用 UIRotationGestureRecognizer 处理旋转手势
- 使用 UISwipeGestureRecognizer 处理轻扫手势
- 使用 UIPanGestureRecognizer 处理拖动手势
- 使用 UILongPressGestureRecognizer 处理长按手势
- 自定义手势处理器的开发思路
- 创建和使用自定义手势处理器

多点触碰是 iPhone、iPad 设备的突出优点，有了这种多点触碰的支持，用户可以通过手指在屏幕上触碰、划过等动作来操作 iOS 应用。用户手指在屏幕上触碰、划过等行为，被统称为手势。iOS 应用通过检测用户手势，并针对用户手势做出响应，即可让 iOS 应用与用户手势交互。对于 iOS 应用来说，用户手势是一种非常重要的“输入方式”，尤其是在各种 iOS 游戏中，通过手势检测可让应用的操作方式更加多样化。

本章将会详细介绍 iOS 的手势处理器。本章将从响应者链、多点触碰等基础理论开始介绍，并通过丰富的示例介绍 iOS 内置的各种手势处理器。不仅如此，本章还会教读者如何开发自定义手势处理器。

1.1 响应者链

只要继承了 UIResponder 的对象就可作为事件的响应者，实际上 UIControl 继承了 UIView，UIView 又继承了 UIResponder，由此可见，所有的 UI 控件都可作为事件的响应者。

当用户与某个控件交互时，该控件将作为“第一响应者（First Responder）”，第一响应者将作为响应者链的开始，该事件首先被发送给第一响应者（也就是用户触摸屏幕的控件）。事件将沿着响应者链一直向下传播，直到被某个响应者处理。

通常来说，第一响应者都是 UIView 控件或 UIView 子类控件，当用户触摸该控件后，事件最先由该控件本身处理；如果该控件自身不处理事件，事件就会被传播到它对应的视图控制器（如果存在）；如果该视图控制器不处理该事件，事件就会传播到包含该控件的父 UIView 控件（如果存在）；如果该父 UIView 控件不处理该事件，事件就会传播到父 UIView 控件对应的视图控制器（如果存在）……直到顶层视图对应的视图控制器。

如果事件从第一响应者开始传播，一直传播到应用界面的顶层视图的视图控制器，该事件依然没有得到处理，接下来该事件就会传播到应用程序窗口（UIWindow 也继承了 UIView）对象；如果应用程序窗口也不处理该事件，该事件就会传播到 UIApplication（它也继承了 UIResponder）；如果该 UIApplication 还不处理该事件，该事件就会传播到应用程序委托对象（使用 Xcode 的项目模板生成的所有 iOS 项目中的应用程序委托类都继承了 UIResponder）。

事件响应者链的典型传播路线如下：

First Responder → First Responder 的视图控制器（如果有）→父容器（如果有）→父容器的视图控制器（如果有）→UIWindow→UIApplication→应用程序委托对象

如果某个事件经过上面完整的传播过程，依然没有被处理，那么该事件就会被丢弃。一般来说，上面响应者链的任何响应者处理该事件，该事件就会停止传播。

如果某个响应者“截获”了某个事件，那么该响应者要根据条件决定是否处理该事件。当响应者无法处理该事件时，则需要在处理方法中手动传递该事件。例如如下代码：

```
func handleTapEvent(event: UIEvent) {
    if(condition) {
        // 处理事件
    } else {
        self.nextResponder.handleTapEvent(event)
    }
}
```

上面粗体字代码手动调用了下一个响应者的 handleTapEvent:方法，这样即可手动将事件传播给下一个响应者。

1.2　响应触碰方法

如果希望自定义控件可以响应用户的触碰事件，则可以通过 UIResponder 的如下 4 个方法来实现。

- – touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event：当用户手指开始接触控件或窗口事件时激发该方法。
- – touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event：当用户手指在控件上移动时激发该方法。
- – touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event：当用户手指结束触碰控件时激发该方法。
- – touchesCancelled:(NSSet *)touches withEvent:(UIEvent *)event：当系统事件（比如内存低事件）中止了触碰事件时激发该方法。

如果希望用户在控件上按下手指激发相应的事件，则可通过重写如下方法来实现。

```
func touchesBegan(touches: Set<UITouch>, withEvent event: UIEvent?) {
    let fingersNum = touches.count
    let tapCount = touches.popFirst()?.tapCount
    ...
}
```

重写这些触碰事件处理方法时，第 1 个 Set 类型的参数代表了用户同时触碰控件的多个手指，如果该用户用 3 个手指触碰了该控件，那么该 touches 参数中将会包含 3 个 UITouch 对象。

UITouch 对象代表一个触碰事件，该对象提供了一个 tapCount 属性，该属性用于返回用户触碰屏幕的次数，比如用户只是轻轻地触碰屏幕一次，那么该属性将会返回 1。如果该属性返回 3，则代表用户快速触碰了 3 次屏幕。如果 touches 集合包含 3 个元素，而且 tapCount 属性返回 3，就代表用户使用 3 个手指触碰了 3 次控件。除此之外，UITouch 还提供了 locationInView: 方法来获取该触碰事件在 UIView 控件中的触碰位置。

至于获取 UITouch 事件之后程序应该针对该触碰事件进行何种处理，则完全取决于应用的业务需要。

除此之外，iOS 为手指触碰事件提供了手势处理器，通过手势处理器可使用一致的编程模式来处理各种触碰事件，而且编程更加简单，因此一般推荐用户使用手势处理器来处理用户触碰事件。

1.3　使用手势处理器（UIGestureRecognizer）

通过手势处理器处理用户触碰事件更加简单，而且无论处理哪种触碰手势，都可面向 UIGestureRecognizer 编程，UIGestureRecognizer 提供了如下子类。

- UITapGestureRecognizer：处理用户点击手势的手势处理器。
- UIPinchGestureRecognizer：处理用户捏合手势的手势处理器。
- UIRotationGestureRecognizer：处理用户旋转手势的手势处理器。
- UISwipeGestureRecognizer：处理用户滑动手势的手势处理器。
- UIPanGestureRecognizer：处理用户拖动手势的手势处理器。
- UIScreenEdgePanGestureRecognizer：处理用户在屏幕边缘的拖动手势的手势处理器。
- UILongPressGestureRecognizer：处理用户长按手势的手势处理器。

使用手势处理器处理用户触碰手势的编程步骤如下。