移动开发丛书

赠送iOS UI开发视频教程和源代码
共36堂课，播放时长超过13小时

源代码下载

iOS 开发实战

从零基础到上架App Store

· 第 3 版 ·

张益珲 编著

使用iOS 11+Xcode 9+Swift 4开发

清华大学出版社

移动开发丛书

iOS 开发实战

从零基础到上架App Store

· 第3版 ·

张益珲 编著

清华大学出版社
北京

内 容 简 介

本书结合编者多年 iOS 开发经验，使用 iOS 11+Xcode 9+Swift 4 开发环境，系统地介绍了 iOS 应用程序从开发到上架的全过程。从开发环境搭建、界面开发、传感器技术、布局与动画技术、网络和数据技术到打包与上传发布流程等，包含了 iOS 软件开发核心技术的方方面面。值得注意的是，本书针对每章的技术要点配备了来自工作实践的项目案例，读者可以边学边练，在编写代码中学习编程。

本书深入浅出，注重实战，案例丰富，非常适合快速上手 iOS 开发的新人，也很适合有一定编程基础但缺少开发 iOS 应用经验的开发人员，还可以作为大中专院校及培训机构的教学用书。

图书在版编目（CIP）数据

iOS 开发实战：从零基础到上架 App Store / 张益珲编著. —3 版. —北京：清华大学出版社，2018
（移动开发丛书）
ISBN 978-7-302-51195-3

Ⅰ. ①i…　Ⅱ. ①张…　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（2018）第 210616 号

责任编辑： 王金柱
封面设计： 王　翔
责任校对： 闫秀华
责任印制： 李红英

出版发行： 清华大学出版社
网　　址： http://www.tup.com.cn，http://www.wqbook.com
地　　址： 北京清华大学学研大厦 A 座　　**邮　　编：** 100084
社 总 机： 010-62770175　　**邮　　购：** 010-62786544
投稿与读者服务： 010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈： 010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者： 三河市国英印务有限公司
经　　销： 全国新华书店
开　　本： 190mm×260mm　　**印　　张：** 22　　**字　　数：** 563 千字
版　　次： 2016 年 8 月第 1 版　2018 年 10 月第 3 版　　**印　　次：** 2018 年 10 月第 1 次印刷
定　　价： 79.00 元

产品编号：077351-01

前　言

编写本书的目的

截止到本书出版，图书市场上已经有很多关于 iOS 开发教学的书籍，它们各有优势，比如有些书籍在某个技术面讲解的非常细致深入，有些书籍专注于 iOS 应用的性能优化，有些书籍则注重于开发语言的语法讲解等。然而如果你有一定的编程语言基础，并且没有任何完整的开发经验，想要学习 iOS 开发，本书依然是非常好的选择，本书为初学者定制，对基础知识点的讲解细致入微，并且配有非常详细的示例代码。本书的前两版，也得到了许多读者的认可，本版相比前两版扩充了一些新的技术点，并且采用最新的 iOS 11+Xcode 9+Swift 4 作为开发环境，以使读者了解 iOS 技术进展，掌握最新的开发技术。

开发一款完整的 iOS 软件是一个复杂的过程，开发者除了需要有编程语言的基础外，还需要对程序设计有宏观的把控。本书在编写时，定位的目标就是要帮助并无太多基础的读者快速上手 iOS 应用开发。从开发环境准备到程序界面开发，从网络与数据逻辑开发到动画与传感器技术，本书都有专门的章节进行介绍。如果说学习理论是一个枯燥的过程，那么在编程中的动手实践就是对理论学习成果的最好检验。本书中基本每一个模块都配备了实战项目，读者可以通过实战项目的练习，体会独立开发 App 应用软件的成就感。

本书主要内容

本书在结构上分 11 章，下面介绍各章的主要内容。

第 1 章是为学习应用开发做准备，介绍开发环境的搭建与开发工具的使用，这一章虽然为准备章节，但对初学者来说至关重要。

第 2 章介绍 iOS 开发中的一些基础 UI 控件，移动端应用一个很重要的特点就是要有绚丽的界面，应用程序的界面决定了用户使用这款应用程序的体验与心情，这一章向读者独立地介绍每个基础控件的用法，并通过一些综合实战来给读者提供使用这些控件的机会。

第 3 章在第 2 章的基础上，向读者介绍 iOS 开发中经常使用的更多高级控件的用法，同样也为读者提供了实战机会。

第 4 章主要讲解了 iOS 应用开发中的网络编程技术，由于网络编程的演示需要有网络数据支持，很多有关网络教学的文档书籍都只是讲授理论，却没有办法使读者切身地进行测试与练习。在编写本章时，特别注意了这个问题，除了讲授网络编程在 iOS 应用开发中的相关知识外，还教读者如何使用网上免费的 API 服务真正做出一款网络应用。

第 5 章主要讲解 iOS 应用程序开发中的音频与视频技术，这类技术在开发音频软件和视频软件中意义重大。

第 6 章将作为动画专题，向读者介绍 iOS 应用开发中的动画技术，章节设计由简到难，并且

都配有代码演示。

第 7 章将作为传感器专题，向读者介绍 iOS 开发中可以调用的设备传感器的相关知识。

第 8 章是界面布局专题，编写中参阅了很多 iOS 应用开发教材，其中都没有过多提到界面布局的相关知识，笔者认为这是一个十分大的弊端，界面布局技术是衡量一个开发者是否合格的重要指标，笔者相信读者学习 iOS 开发技术绝对不只是想简简单单地做出一个 Demo 自己玩，做出“产品”才是读者的真正目标，而一款成熟的产品一定是具有兼容性的，一定是优雅的。因此，本书特别将 iOS 界面布局技术作为一个单独的章节来向读者介绍。

第 9 章是数据持久化专题，本章将介绍有关 iOS 应用开发中的文件操作、数据库操作的相关知识。

通过前 9 章的学习，你已经具备独立开发一款 iOS 应用的基础能力，但是仅仅做出产品还不够，如何让自己的产品在市场发布，使用户可以下载使用也是开发者不得不去了解、学习的内容，第 10 章将完整地向读者介绍提交自己的应用到 App Store 的整个过程。

第 11 章是扩展章节，此章也是读者开发能力提升的一个章节，本章将介绍一些独立于前面章节，但在实际开发中也举足轻重的编程技术。

视频教学和源代码下载

读者可以从如下网址（注意区分数字与字母大小写）下载源代码和视频教学：

https://pan.baidu.com/s/19Z5WSYAqKXLpif-t9kkNMg

https://pan.baidu.com/s/1TaYi8cyW6zh_htMRjVOayA

也可以扫描下面的二维码获取网址。如果下载有问题，请联系电子邮箱 booksaga@126.com，邮件主题为“iOS 开发实战：从零基础到上架 App Store”。

如果你想获取更多关于 iOS 开发的视频教学，可以访问笔者的网络学院：http://edu.csdn.net/lecturer/1283。也可以访问笔者的技术博客: https://my.oschina.net/u/2340880。博客中有近 300 篇关于 iOS 开发的技术文章，这可以成为你的知识存储库。

对于本书前两版中出现的差错，衷心地向读者表示歉意。本版中对读者提出的问题都一一进行了修正，在终稿前，笔者花费了很多时间反复校稿，希望本版可以完美地出现在读者面前，然而世上完美之事少之又少，虽已尽力，仍不敢保证此书中再无差错。此书一经出版已成定局，如果你在阅读本书时发现任何差错或有任何疑问，都可以直接联系笔者，QQ：316045346。

最后，本书得以顺利完成和出版，要感谢清华大学出版社的王金柱编辑，感谢他在笔者写作过程中的指导与鼓励，可以将编程经验顺利地分享给读者，王金柱编辑是不可替代的桥梁。

张益珲（网名：珲少）

2018.7.22

目　录

第1章

开发准备

工欲善其事，必先利其器。在学习 iOS 移动开发之前，首先应该将开发环境配置完成并对所需要使用的开发工具进行了解与熟悉。本章首先向读者介绍 iOS 11 系统相比之前系统的一些新特性，使读者对目前主流的 iOS 系统有一个宏观上的了解，然后将一步步演示开发环境的搭建并介绍开发工具 Xcode 的常用功能。

通过本章的学习，读者能够掌握：

- 了解 iOS 11 的新特性和新功能。
- 申请免费的 Apple ID 账号。
- 使用 Xcode 开发工具创建 iOS 工程。
- 使用 Xcode 开发工具编写与调试程序。
- 熟悉 Xcode 工程结构。
- 编写第一个程序 Hello World。
- 使用 Git 工具进行版本管理。
- 使用 GitHub 代码托管平台。

1.1 iOS 11 新特性简述

随着 2017 年 iPhone X 的发布，iOS 系统也迎来了它的又一次重大升级。新版本 iOS 11 在用户交互维度新增了拖放操作，用户在 iOS 系统中可以将元素在不同界面间或不同 APP 间进行传递。在文件管理方面，iOS 11 提供了本地文档与 iCloud 云文档浏览器。除了一些细节上的调整和优化，iOS 11 开发框架中还新增了目前非常受欢迎的机器学习模型和 AR 开发框架。另外，Swift 语言也升级到了 4.0 版本，Swift 语言自从发布以来，就一直褒贬各半，首先其先进的现代编程语言特性的确有很强的安全性和编程效率，另一方面其频繁的更新和接口变动也给开发者的项目维护带来很

大的麻烦。从 Swift 语言的更新内容和趋势来分析，3.0 是一个分界点，Swift 3.0 对语言风格和 API 做了重构性质的升级，4.0 版本则只是完善与补充，并没有比较突出的修改，因此无论是学习还是开发项目，目前的 Swift 语言都是非常适合的。了解 iOS 11 的这些新特性与 Swift 4.0 的相关更新点，相信可以更好地帮助读者学习 iOS 应用程序开发。

1.1.1 新增拖放交互编程接口

在 Mac OS 软件开发时，拖拽交互是一种十分常用的交互方式，在 iOS 11 之前的系统中要实现拖拽交互往往比较困难。iOS 11 系统新引入了拖拽相关的 API，可以帮助开发者快速构建拖拽交互，在 iOS 11 系统中，使用这种 API 进行 APP 的开发为设计提供了一种全新维度的用户交互方式。

拖拽操作在 iPad 上是支持跨应用程序的，用户可以从一个应用中拖取项目，通过 Home 键回到主界面并打开另一个应用程序，然后将被拖拽的项目传递给这个应用程序中。在 iPhone 上，拖拽操作只支持当前应用程序内，用户可以将某个元素从一个界面拖拽到另一个界面，这种维度的操作可以使设计人员在设计产品时，有更大的灵活性。

对于拖拽操作，至少要有两个组件：一个组件作为拖拽源用来提供数据；另一个组件作为拖拽目的用来接收数据。当然，同一个组件既可以是拖拽源也可以是拖拽目的。

任意的 UIView 组件都可以作为拖拽源，让其成为拖拽源其实也十分简单，只需要 3 步：

步骤01 创建一个 UIDragInteraction 行为对象。

步骤02 设置 UIDragInteraction 对象的代理并实现相应方法。

步骤03 将 UIDragInteraction 对象添加到指定的 View 上。

最简单的可拖拽组件的创建示例代码如下：

```
    lazy var dragView = { ()->UIView in
        let view = UIView(frame: CGRect(x: 100, y: 100, width: 100, height: 100))
        view.backgroundColor = UIColor.red
        view.addInteraction(self.dragInteraction)
        return view;
    }()

    lazy var dragInteraction = { ()->UIDragInteraction in
        let dragInteraction = UIDragInteraction(delegate: self)
        dragInteraction.isEnabled = true
        return dragInteraction
    }()

    func dragInteraction(_ interaction: UIDragInteraction, itemsForBeginning session:
UIDragSession) -> [UIDragItem] {
        let provider = NSItemProvider(object: "Hello World" as NSItemProviderWriting)
        let item = UIDragItem(itemProvider: provider)
        return [item]
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        self.view.addSubview(self.dragView)
```

```
    }
```

拖拽源是数据的提供者，放置目的地就是数据的接收者。同样，对于任何自定义的 UIView 视图，我们也可以让其成为放置目的地，需要以下三步完成：

步骤01 创建一个 UIDropInteraction 行为对象。

步骤02 设置 UIDropInteraction 对象的代理并实现协议方法。

步骤03 将其添加到自定义的视图中。

例如，我们将自定义的 UILabel 组件用来显示拖拽的文案，代码如下：

```
    import UIKit
    class ViewController: UIViewController,UIDragInteractionDelegate,UIDropInteractionDelegate
{
        lazy var dragView = { ()->UIView in
            let view = UIView(frame: CGRect(x: 100, y: 100, width: 100, height: 100))
            view.backgroundColor = UIColor.red
            view.addInteraction(self.dragInteraction)
            return view;
        }()
        lazy var dragInteraction = { ()->UIDragInteraction in
            let dragInteraction = UIDragInteraction(delegate: self)
            dragInteraction.isEnabled = true
            return dragInteraction
        }()
        lazy var dropLabel = {()->UILabel in
            let label = UILabel(frame: CGRect(x: 10, y: 300, width: 300, height: 30))
            label.backgroundColor = UIColor.green
            label.isUserInteractionEnabled = true
            label.addInteraction(self.dropInteraction)
            return label
        }()
        lazy var dropInteraction = {()->UIDropInteraction in
            let dropInteraction = UIDropInteraction(delegate: self)
            return dropInteraction
        }()

        func dropInteraction(_ interaction: UIDropInteraction, canHandle session: UIDropSession)
-> Bool {
            return true
        }
        func dropInteraction(_ interaction: UIDropInteraction, sessionDidUpdate session:
UIDropSession) -> UIDropProposal {
                return UIDropProposal(operation: .copy)
        }
        func dropInteraction(_ interaction: UIDropInteraction, performDrop session: UIDropSession)
{
            let _ = session.loadObjects(ofClass: String.self) { (itemArray) in
                self.dropLabel.text = itemArray.first
            }
        }
        func dragInteraction(_ interaction: UIDragInteraction, itemsForBeginning session:
UIDragSession) -> [UIDragItem] {
            let provider = NSItemProvider(object: "Hello World" as NSItemProviderWriting)
```

```
        let item = UIDragItem(itemProvider: provider)
        return [item]
    }
    override func viewDidLoad() {
        super.viewDidLoad()
        self.view.addSubview(self.dragView)
        self.view.addSubview(self.dropLabel)
    }
}
```

上面的代码将我们自定义的拖拽源提供的 Hello World 拖放进了 UILabel 组件中。

1.1.2 其他新增功能

在 iOS 11 系统中新增了访问本地和 iCloud 文档的功能，开发者可以使用 UIDocumentBrowserViewController 和 UIDocumentBrowserTransitionController 这两个视图控制器来管理文档浏览。

新增了 MusicKit 开发框架，开发者可以在应用中访问完整的 Apple Music 音乐目录，并且提供了更多与 Apple 音乐程序的交互接口。

新增了 ARKit 开发框架，开发者结合摄像头可以更加容易地构建 AR 体验项目。

新增了用于检测人脸识别、条形码等视觉效果的开发框架。

新增了 CoreML 开发框架，开发者更便于将机器学习模型集成到应用中。

1.2 熟悉 iOS 开发环境

Xcode 是进行 iOS 应用开发必备的开发软件。Xcode 开发工具功能十分强大且简单易用，不需要过多的配置，下载并安装后，各种环境和模拟器即关联完毕，对于初学者来说门槛很低。

1.2.1 安装 Xcode 开发工具

由于 iOS 系统的封闭性，开发 iOS 软件的工具环境并不多，Xcode 是 Apple 公司自己开发的一套针对 OS、iOS、watchOS 和 tvOS 的开发环境，使用方便且功能十分强大。用户可以在 App Store 上免费获取 Xcode 开发工具。

首先需要申请个人的 AppleID。AppleID 是 Apple 会员的凭证，也是个人的信息管理凭证。申请个人的 AppleID 是免费的，登录 www.apple.com/cn Apple（中国）官方网站，在屏幕右上角的购物袋按钮中选择“登录”选项，如图 1-1 所示。