Objective-C 程序设计

基础教程

刘 冶◎著

- 全面解读Objective-C语言
- Mac OS X和iOS开发必备

吉林大学出版社
JILIN UNIVERSITY PRESS

Objective-C 程序设计基础教程

刘 冶◎著

吉林大学出版社
JILIN UNIVERSITY PRESS

图书在版编目（CIP）数据

Objective-C 程序设计基础教程/刘冶著. ——长春：吉林大学出版社，2011.6
ISBN 978-7-5601-7382-5

Ⅰ.①O… Ⅱ.①刘… Ⅲ.①C 语言—程序设计—教材 Ⅳ.①TP312

中国版本图书馆 CIP 数据核字(2011)第 108459 号

书　名：Objective-C 程序设计基础教程
作　者：刘冶 著

责任编辑、责任校对：刘冠红　魏丹丹　刘明明　　封面设计：三鼎甲
吉林大学出版社出版、发行　　北京紫瑞利印刷有限公司　印刷
开本：710×1000 毫米　1/16　　2011 年 6 月　第 1 版
印张：13.5　字数：250 千字　　2011 年 6 月　第 1 次印刷
ISBN 978-7-5601-7382-5　　定价：35.00 元

社址：长春市明德路 421 号　邮编：130021
发行部电话：0431-88499826
网址：http//www.jlup.com.cn
E-mail：jlup@mail.jlu.edu.cn

前　言

作为世界上最具创新能力的公司之一，苹果公司已经成为世界信息技术的领导者，其产品理念和工业设计一直保持着对业界广泛而深刻的影响。Objective-C 语言作为苹果软件产品创新的基石，凭借着其优秀的语言特性，广泛地受到开发人员的应用和认可。作为一门开发语言，Objective-C 已经在苹果软件应用开发中占据了主要地位，是苹果电脑 Mac OS X 操作系统的推荐开发语言，同时也是苹果公司的 iPhone、iPad 和 iPod Touch 等优秀产品第三方应用的指定开发语言。

随着苹果公司产品的影响力和市场占有率日益提高，Objective-C 语言在世界范围内的使用日益普及，在 2011 年 4 月份 TIOBE 网站的编程语言权威排名中，Objective-C 语言已经名列第 8，进入了前十名。作为一门面向对象的程序设计语言，Objective-C 语言的流行有多方面的原因，除了苹果公司在业界的影响力和优秀的产品推广外，还与 Objective-C 语言优秀的语言特性密切相关。

当前国内各大高校和信息技术企业从事 Objective-C 语言研究和开发的人员不断增多，市面上出现了不少国外优秀书籍的影印本和翻译本，但随着 Objective-C 语言的发展，关于 Objective-C 语言的知识需要与时俱进地更新。同时，随着国内更多的人想加入到 Objective-C 语言的研究和开发中来，而国外的优秀书籍大多讲解深入细致，介绍语言入门基础的书籍并不多，因此讲述 Objective-C 程序设计基础，引导更多的人快速地掌握 Objective-C 语言的基础知识和开发方法显得非常重要。

本书介绍 Objective-C 语言程序设计的基础，对 Objective-C 语言的语法结构和功能特性进行深入讲述，并在此基础上简单介绍了一些高级编程的内容，全书共分为八章。第一章到第三章主要介绍 Objective-C 语言的特性、C 语言基础和程序控制方法，第四章到第七章主要讲述 Objective-C 面向对象编程中的类和对象、继承和多态、分类和协议等功能特性，第八章讲述了常用的 Foundation 框架类型，同时讨论了一些高级编程的相关概念方法。

第一章　介绍 Objective-C 语言的特性，给出第一个使用该语言编写的

程序例子，并介绍苹果公司提供给开发者的 Foundation 框架，最后讲述在苹果电脑上使用 Xcode 工具进行程序设计的相关步骤和方法。

第二章　首先会讲述 C 语言的基本数据类型和运算符的使用方法，然后介绍数组与结构的定义和初始化，以及相关常用的方法，最后对函数与指针的定义和使用进行详细讲述。

第三章　重点介绍 Objective-C 程序控制结构中的选择结构和循环结构的实现方法，首先介绍选择结构中的 if 语句和 switch 语句的用法，然后讨论 while 语句、do-while 语句和 for 语句实现循环结构的方法，最后简要介绍 continue 语句和 break 语句在程序控制中的作用。

第四章　讲述的是面向对象编程的基础知识，首先会对面向对象编程进行介绍，并引入类和对象的概念，并讲述继承和多态，然后着重讲述 Objective-C 语言的面向对象特性，初步介绍 Objective-C 对 C 语言进行面向对象扩展的部分。

第五章　着重对面向对象编程中的类和对象进行讨论，首先会讲述类的创建和实例化，以及对象的使用方法，然后详细介绍类的接口部分和实现部分的模块化，并通过比较讲述类方法和对象方法的区别，最后简要介绍存储数据和访问限定的相关问题。

第六章　对 Objective-C 语言继承和多态的功能特性进行详细介绍，首先讲述对类继承的相关理论和实现形式，并对覆盖和重载进行比较研究，接着讨论继承中访问和初始化的问题，然后介绍 Objective-C 语言多态的特性和意义，并详细讲述动态绑定的使用方法。

第七章　重点讨论 Objective-C 语言中分类和协议两种特性，首先介绍分类的定义和使用方法，并尝试使用分类特性编写面向对象程序代码，然后介绍协议的定义和在类接口部分中的使用，同时讲述协议的具体实现方法，以及将正式协议与非正式协议进行对比。

第八章　着重讲述 Foundation 框架类型的使用方法，首先介绍 Foundation 框架中常用的类型，对可变类和不可变类的联系和区别进行讨论，接着分别对 Foundation 框架中的 NS 字符串类型、NS 数字类型、NS 数组类型和 NS 字典类型的声明和使用方法进行详细介绍。

本书还提供了附录和参考文献，可以供读者在了解掌握语言的过程中参考和深入学习。

与其他同类书籍相比，本书更强调理论和实践相结合的方法，注重对程序的认识和实现，提供了大量的程序例子供读者参考，书中还附有图片和表格使得叙述更加完整。本书力图通过知识体系的讲述，以及实际程序设计的操作，增加读者对 Objective-C 语言的兴趣，更好地提升读者对 Objective-C 语言的研究开发能力。

本书可以作为高等院校计算机科学与技术以及相关学科的教学研究资料和参考教程，也可以作为 Objective-C 语言程序开发人员的语言基础参考资料和培训教材。

编写一本程序设计语言教材并不是一件容易的事，本书的编著花费了近一年的时间，在本书的编写过程中，得到了许多前辈和朋友的大力支持和帮助，使得本书可以顺利完成。

感谢中山大学信息科学与技术学院计算机科学系的郭嵩山教授，他严谨的教学和研究作风深刻地影响着我，引导我从事计算机科学与技术相关领域的研究，并一直给予坚定的支持。

感谢中山大学信息科学与技术学院计算机科学系的纪庆革副教授，他在面向对象程序设计的相关理论，以及文献资料的参考方法上给了我重要的指导。

感谢张海光老师和姚肖华老师引导我进入计算机程序设计的领域，使我一直热爱并从事相关领域的学习和研究工作。

感谢我的前辈和好友，腾讯公司的林博、中山大学的徐振涛和悉尼大学的陈云龙，在本书的编写过程中提供的意见和建议，以及给予的支持与帮助。还有其他在本书编写过程中给予支持的朋友们，对于他们的辛勤付出，在此表示衷心的感谢。

最后还需要感谢我的家人一如既往的理解、支持和鼓励，使得我能有时间和动力，坚定地完成本书的编著工作。

由于作者水平有限，书中难免存在错误或疏漏之处，希望广大读者批评指正。作者的电子邮箱地址是 jourkliu@163.com，欢迎读者对书中的问题提出意见和建议。

刘冶

2011 年 5 月

于中山大学

目　录

第一章 绪 论

苹果公司是世界信息技术的领导者，其创新的产品理念和工业设计一直保持着对业界广泛而深刻的影响。Objective-C 语言作为苹果软件产品创新的基石，凭借着其优秀的语言特性，广泛地受到开发人员的应用和认可。作为一门开发语言，Objective-C 已经在苹果软件应用开发中占据了主要地位，这是苹果电脑 Mac OS X 操作系统的推荐开发语言，同时也是苹果公司的 iPhone、iPad 和 iPod Touch 等优秀产品第三方应用的指定开发语言。

本章将介绍 Objective-C 语言的特性，给出第一个使用该语言编写的程序例子，并介绍苹果公司提供给开发者的 Foundation 框架，最后讲述在苹果电脑上使用 Xcode 工具进行程序设计的相关步骤和方法。

第 1 节 Objective-C 语言

Objective-C 语言是著名的 C 语言的一个超集，对传统基于过程的 C 语言添加了面向对象的特性，使之成为优秀的面向对象开发语言之一。在 2011 年 4 月份 TIOBE 网站的编程语言权威排名中，Objective-C 语言已经名列第 8，进入了前十名，这是历年来最好的名次，也意味着 Objective-C 已经成为一门主流的开发语言。Objective-C 语言的迅速发展得益于其优秀的语言特性以及苹果公司的致力推广，读者可以从 TIOBE 官方网站 http://www.tiobe.com 获得详细且最新的编程语言排名信息。

1.1.1 Objective-C 的发展历程

Objective-C 语言是在 C 语言的基础上添加了 Smalltalk 特性的一门面向

对象程序设计的语言。下面将讲述 Objective-C 语言的发展史上的一些重要事件。

在20世纪80年代早期，Brad Cox 和 Tom Love 在他们名叫 StepStone 的公司创造了最初的 Objective-C 语言。

在 Steve Jobs 离开苹果公司之后，创建了另一家公司 NeXT。NeXT 公司在1988年从 StepStone 获得了 Objective-C 的授权。在这之后，NeXT 创建了自己的 Objective-C 编译器和库文件，这便是 NeXTstep 的用户接口和界面的基础。由于 NeXT 公司在业界的影响力限制，NeXTstep 并未能够广泛地推广开来。与此同时，著名的 GNU 项目开始了基于 OpenStep 标准的 NeXTStep 免费版本的开发，并命名为 GNUstep。

在1996年之后，随着 NeXT 公司被苹果公司收购，还有 Steve Jobs 回归到苹果公司，Objective-C 借助苹果公司在业界广泛而深刻的影响力，开始发挥重大的作用。Objective-C 语言被用在苹果的 Max OS X 操作系统的开发上，包括了基于 NeXT 的 Objective-C 语言标准，还有开发者工具和界面设计工具，后者就是现在苹果开发者所熟悉的 Xcode 和 Interface Builder。

Objective-C 语言在 C 语言的基础上加了一层特性，可以说是 C 语言的严格超集，所以 C 语言程序在 Objective-C 编译器上可以达到完全兼容。可以说，Objective-C 语言的所有非面向对象语法特征与 C 语言完全相同，而面向对象的实现方式则大部分参考了 Smalltalk 语言。

在2007年的10月份，苹果公司发布了 Max OS X 的新版本 10.5，并包含了新的 Objective-C 2.0 编译器。苹果公司在 Objective-C 2.0 中添加了许多新的特性，包括现代的垃圾回收机制，增强语法内容以及运行环境的64位支持。

1.1.2 Objective-C 的语言特性

Objective-C 是一门面向对象的程序设计语言，发行于1986年，由 Brad Cox 和 Tom Love 设计。Objective-C 属于弱类型语言，并支持动态类型。Objective-C 的设计过程中受到了 C 语言和 Smalltalk 语言的启发，并影响了之后 Java 语言和 Ruby 语言的设计。Objective-C 的另一个设计理念是跨平台跨操作系统，使得语言能支持在不同的平台上编译运行。虽然目前 Objective-C主要在 Max OS X 和 iOS 上面被广泛使用，但事实上 Objective-C

可以在任何支持 gcc 的平台上编译运行代码。

在 Objective-C 中引入具有特色的消息传递机制，这种机制经常被 C++程序员等价地理解为调用方法，但事实上这两者之间存在行为上的区别。在 Objective-C 中，类和消息并不是紧密地绑定在一起的，类可以调用不存在类里面的消息，程序对于类无法处理的消息会抛出异常。

Objective-C 语言中的另一个鲜明的特性是类的定义需要分为接口和实现两个代码模块，而且两个模块不能合并在一起。这种做法的好处是，接口可以放置在头文件中，而在代码文件中进行实现，结合 Objective-C 语言的语言特性，做到灵活的程序设计效果。

分类概念的提出，使得 Objective-C 语言适应了大型程序编写的要求。通常在大型程序的设计中，需要对一些类添加不同的特性以适应不同的应用需求。重复编写类的代码会使得程序变得冗长，同时导致了可用性变差，使用分类基本解决了这个问题。

由于 Objective-C 语言不支持多重继承，所以引入了协议的概念。熟悉 Java 语言的程序员会发现，这种协议十分类似 Java 语言中的接口概念，而事实上这两者是有一定区别的。Objective-C 提供了两种协议的定义方式，一种称为非正式协议，而另一种则是正式协议。在今后的相关章节中，我们会详细讨论协议的使用方法。

Objective-C 语言支持动态类型，还定义了 id 为泛型，使得程序设计的过程中避免了不停地进行强制类型转换。因为在理论上，强制类型转换并不总是可靠的，可能会因为类型无法匹配出现运行时的错误。

在 2007 年苹果公司发布的 Objective-C 2.0 版本中，加入了垃圾回收机制，同时引入了属性的概念，还支持快速枚举方法，这些新的语言特性使得 Objective-C 语言更加灵活方便。

1.1.3 第一个 Objective-C 程序

接下来将简要介绍第一个用 Objective-C 语言编写的程序，读者可能对语言和语法感到陌生和不适应，但是并不用感到慌张。通过后续的学习，能够深入了解 Objective-C 语言的特性，而这第一个程序，只是为了让初学者大致了解 Objective-C 语言程序的基本结构。

通常学习每一门语言，第一个例子就是打印出“Hello，world!”这个

句子，自然学习 Objective-C 也不例外，下面是使用 Objective-C 语言实现这个功能的代码：

```
//Prog_1 - 1. m
#import < Foundation/Foundation. h >

int main (int argc, const char * argv [])
{
  NSAutoreleasePool * pool = [ [NSAutoreleasePool alloc] init];
  NSLog (@"Hello, world!");
   [pool drain];
  return 0;
}
```

通过对比往往更容易理解，以下提供了用其他几种程序设计语言实现打印“Hello, world!”字符串功能的程序代码。

使用 C 语言的实现代码：

```
//Prog_1 - 2. c
#include < stdio. h >

int main (int argc, const char * argv [])
{
  printf ("Hello, world!");
  return 0;
}
```

使用 C + + 语言的实现代码：

```
//Prog_1 - 3. cpp
#include < iostream >

using namespace std;

int main (int argc, const char * argv [])
```

```
{
    cout << "Hello, world!";
    return 0;
}
```

使用 Java 语言的实现代码：

```
//Prog_1 -4. java
public class HelloWorld {
    public static void main (String [ ] args) {
        System. out. println ("Hello, world!");
    }
}
```

使用 Pascal 语言的实现代码：

```
//Prog_1 -5. pas
program HelloWorld (Input, Output);
begin
    write ('Hello, world!');
end.
```

1.1.4 分析第一个程序

为了让读者从感官上初步认识 Objective-C 程序的结果，理解一些简单的概念，下面对第一个 Objective-C 程序进行分析。在分析之前，我们再次给出第一个 Objective-C 程序的代码：

```
//Prog_1 -1. m
#import <Foundation/Foundation. h>

int main (int argc, const char * argv [ ])
{
    NSAutoreleasePool * pool = [ [NSAutoreleasePool alloc] init];
    NSLog (@"Hello, world!");
```

```
    [pool drain];
    return 0;
}
```

我们看到，程序的第一行由“//”引出一段内容，这段内容在程序中叫做注释，注释在 Objective-C 语言程序编译运行中并没有起到作用，只是为了便于阅读加上的。对于只有一行的注释内容，可以采用类似下面的写法：

```
//这里是注释的内容
```

对于多行的程序，可以使用“/*”表示开始，并用“*/”表示结束，如下所示：

```
/*
这里是一行注释的内容
这里是另一行注释的内容
…
*/
```

这里要提醒初学者，程序的注释虽然对程序的编译运行并不能起到任何作用，但这并不意味着注释不重要。相反，有时候程序中的注释往往是最重要的部分之一，因为程序员可能经过较长的时间之后，并不能清楚地回忆起当时编写的程序逻辑和含义，或者是开发较大型的软件时，需要多个人共享程序代码。总之，在很多情况下，编写程序需要作出必要但不冗长的注释，这样有利于提高程序的可读性。

接下来看看程序中的另一行代码：

```
#import < Foundation/Foundation. h >
```

这行代码是让编译器将系统的 Foundation. h 文件导入到程序中，代码的作用是为了使用程序中的一些系统给定的功能。代码中#import 引出本程序中所要包含的文件名称。应该注意，对于系统已有的文件，使用一对尖括号表示，例如 < Foundation/Foundation. h > ，而对于程序员自定义的文件名，使用一对双引号表示，例如"ImportFile. h"。这行程序引入的是系统提供的 Foundation 框架文件，关于这个框架的详细介绍在今后的内容中会进行讨论。

对 C 和 C + + 语言有所了解的读者可能会困惑 Objective-C 语言中的

#import和 C/C + +语言中的#include 存在的区别。作为 C 语言的一个超集，Objective-C 语言也是支持#include 导入文件的，但是需要注意的是，使用"#import"的时候，系统可以保证头文件只被包含一次，这样可以避免头文件在文件中被多次导入造成的混乱情况，所以使用 Objective-C 进行程序设计的时候，强烈推荐使用"#import"而不是"#include"命令。

紧接着是另一行程序代码：

```
int main (int argc, const char * argv [])
```

这行代码还引出一大段使用花括号括起来的内容。这是程序中的 main 函数，告诉系统程序的入口位置，并可以携带命令行参数。应该注意到，main 是一个特殊的名称，程序中只允许有一个这样的入口函数。

接下来花括号中就是 main 函数中执行的程序代码，如下：

```
NSAutoreleasePool * pool = [ [NSAutoreleasePool alloc] init];
NSLog (@"Hello, world!");
[pool drain];
return 0;
```

这段代码的第一行语句是在自动释放池中保留内存空间，关于内存空间的问题，本书中有专门的章节进行详细介绍。

第二行语句是调用"NSLog ()"函数输出字符串 “Hello, world!”，这里使用了系统定义的"NSLog ()"函数，这正是上面代码中为何使用"#import"语句带入系统库的原因。注意这里的字符串之前的符号"@"，这个符号表示之后的字符串是一个 Objective-C 中的 NSString 字符串对象。可以先简单地这么理解，Objective-C 语言作为 C 语言的一个超集，除了支持 C 语言本身的字符串之外，为了面向对象的需要，定义了 NSString 字符串对象，并且放在了 Foundation 框架中提供给开发者使用。如果在字符串之前加上了符号"@"，表明这是一个 NSString 字符串对象，而如果没有的话，则是一个普通的 C 类型字符串。

第三行语句是与之前第一行语句相关联的，作为是释放已经分配的内存池。

第四行是返回 main 函数的一个状态值，并表示程序结束，状态值 0 表示程序正常结束。

至此，读者应该对 Objective-C 语言程序的基本结构有基本的认识。

1.1.5 关于内容的一些说明

在本书进入详细的内容介绍之前，先做一些约定。书中的一部分，会通过引用参考文献引导读者进一步深入学习。书中的参考文献进行了编号，详细文献的规范描述可以在本书末尾的参考文献目录中查阅。关于苹果开发的基本问题，读者可以通过苹果公司的官方开发者网站 http：//developer. apple. com 获得信息。对于一些概念性的问题，也可以通过维基百科网站 http：//www. wikipedia. org 进行查询。

熟悉程序设计语言学习的过程的人会感到，理解冗长而繁复的理论描述可能远没有通过例子学习简单，本书力图通过丰富的例子，引导初学者快速学习 Objective-C 语言特性，并成为一个能够实践的开发者，而不是经过长时间学习仍然停留在掌握语法理论的基础上。

第 2 节　Foundation 框架

与 C + + 语言和 Java 语言的类库相似，Objective-C 语言引入了框架的概念，使得程序中可以动态地载入共享的资源。在 Max OS X 的程序开发中，苹果公司提供了著名的 Cocoa 框架，Cocoa 中包含了三个主要的框架，分别是 Foundation 框架、AppKit 框架和 Core Data 核心数据。而在 iOS 系统的应用开发中，苹果公司使用 UIKit 框架代替了 AppKit 框架提供用户界面类，并把 Foundation 框架和 UIKit 框架合称为 Cocoa Touch 框架。因此，在 Cocoa 框架和 Cocoa Touch 框架中，Foundation 框架的地位十分重要。特别是在命令行程序中，我们只需要使用 Foundation 框架进行编程。

1.2.1 Foundation 框架概述

在前面的第一个 Objective-C 程序中，我们可能已经很熟悉下面这个语句：

```
#import < Foundation/Foundation. h >
```

在这里，我们可以知道，通过"#import"命令，引入了系统提供的 Cocoa 框架中的 Foundation 框架，可以通过使用 Foundation 框架中提供的相关功

能来简化我们的程序设计。

在 Objective-C 中，引入框架的概念，是为了提高程序开发的简便性，利用框架提供了类、方法、函数和文档等信息，而 Foundation 框架正是各类框架的一个基础。

Foundation 框架提供了一些基本的对象，诸如 NSString 字符串对象和 NSNumber 数字对象，还提供了 NS 数组类型和 NS 字典类型，此外我们还可以使用 Foundation 框架的内存管理、文件系统处理和日期时间处理等功能。

苹果公司的官方开发者网站上提供了 Foundation Framework Reference，读者可以通过下面的链接查询关于 Foundation 框架的信息：

http：//developer. apple. com/library/mac/#documentation/Cocoa/Reference/Foundation/ObjC_classic/_ index. html

Foundation 框架的应用普遍且广泛，在今后的学习中，我们会专门在一些章节详细地讲述 Foundation 框架的一些特性和功能。

本书中的附录 A 中列出了 Foundation 框架中定义的类和协议名称。

1.2.2 Foundation 框架的应用

在 Foundation 框架中，应用广泛的是 NS 对象类型，包括 NSString 字符串和 NSNumber 数字，此外，Foundation 框架中还提供了 NS 数组类型和 NS 字典类型，以及相关的方法。虽然我们可以不使用 Foundation 框架编写 Objective-C 程序，但是在程序的设计开发中，使用苹果官方提供的 Foundation 框架能够提高我们的编写效率和准确性，还有利于代码的共享。在本书的第八章 NS 对象数据类型中，将详细讨论这些内容。

除了基本的对象类型，Foundation 框架还允许我们在文件系统中进行文件和目录的操作，Foundation 框架提供了相应的类和方法，使得我们可以方便地对文件和目录进行创建、读取、写入、更新和删除等操作，这些基本的文件和目录操作方便了程序编写者对文件系统的操作。

内存管理在 Objective-C 的程序设计中至关重要，贯穿着程序开发的始终，如何正确地创建和释放对象，防止内存泄露，这是一个关键的问题。在苹果提供的 Foundation 框架中，提供了内存管理的机制，方便程序员对内存的控制和预防内存泄露。