

Objective-C

КАРМАННЫЙ СПРАВОЧНИК

Objective-C

PHRASEBOOK

SECOND EDITION

David Chisnall

DEVELOPER'S
LIBRARY


Addison
Wesley

www.informit.com/developers

Objective-C

КАРМАННЫЙ СПРАВОЧНИК

ВТОРОЕ ИЗДАНИЕ

Дэвид Чиснолл



Москва • Санкт-Петербург • Киев
2012

ББК 32.973.26-018.2.75

Ч-67

УДК 681.3.07

Издательский дом “Вильямс”

Зав. редакцией *С.Н. Тригуб*

Перевод с английского и редакция *И.В. Берштейна*

По общим вопросам обращайтесь в Издательский дом “Вильямс”
по адресу:

info@williamspublishing.com, <http://www.williamspublishing.com>

Чиснолл, Дэвид.

Ч-67 Objective-C. Карманный справочник, 2-е изд. : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2012. — 352 с. : ил. — Парал. тит. англ.

ISBN 978-5-8459-1777-5 (рус.)

ББК 32.973.26-018.2.75

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения издательства Addison-Wesley Publishing Company, Inc.

Authorized translation from the English language edition published by Addison-Wesley Publishing Company, Inc., Copyright © 2012

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise.

Russian language edition published by Williams Publishing House according to the Agreement with R&I Enterprises International, Copyright © 2012

Научно-популярное издание

Дэвид Чиснолл

Objective-C. Карманный справочник 2-е издание

Литературный редактор *И.А. Попова*

Верстка *О.В. Мишутина*

Художественный редактор *Е.П. Дынный*

Корректор *Л.А. Гордиенко*

Подписано в печать 17.05.2012. Формат 70x100/32.

Гарнитура Times. Печать офсетная.

Усл. печ. л. 11,0. Уч.-изд. л. 12,9.

Тираж 1500 экз. Заказ № 0000.

Первая Академическая типография “Наука”

199034, Санкт-Петербург, 9-я линия, 12/28

ООО “И. Д. Вильямс”, 127055, г. Москва, ул. Лесная, д. 43, стр. 1

ISBN 978-5-8459-1777-5 (рус.)

ISBN 978-0-321-81375-6 (англ.)

© Издательский дом “Вильямс”, 2012

© Pearson Education, Inc., 2012

Оглавление

Введение	13
Глава 1. Основные принципы Objective-C	19
Глава 2. Азбука Objective-C	35
Глава 3. Управление памятью	75
Глава 4. Шаблоны, распространенные в Objective-C	105
Глава 5. Числа	127
Глава 6. Манипулирование символьными строками	141
Глава 7. Работа с коллекциями	163
Глава 8. Даты и моменты времени	183
Глава 9. Работа со списками свойств	193
Глава 10. Взаимодействие с окружением	209
Глава 11. Доступ к значениям по ключам	219
Глава 12. Обработка ошибок	231
Глава 13. Доступ к файлам и каталогам	247
Глава 14. Потоки.....	263
Глава 15. Блоки и Grand Central Dispatch	277
Глава 16. Уведомления	289
Глава 17. Доступ к сети	299
Глава 18. Отладка программ на Objective-C	313
Глава 19. Динамическая библиотека Objective-C	325
Предметный указатель	341

Содержание

Об авторе.....	13
Благодарности.....	14
Введение.....	15
О версиях книги.....	16
От издательства.....	17
Глава 1. Основные принципы Objective-C.....	19
Понятие объектной модели.....	20
О двух системах типов.....	22
Родство C и Objective-C.....	23
Язык и библиотека.....	24
История развития Objective-C.....	26
Межплатформенная поддержка.....	30
Компилирование программ на Objective-C.....	32
Глава 2. Азбука Objective-C.....	35
Объявление типов Objective-C.....	36
Отправка сообщений.....	39
Представление о селекторах.....	43
Объявление классов.....	45
Применение протоколов.....	50
Добавление методов в класс.....	51
Применение неформальных протоколов.....	54

Синтезирование методов с помощью объявляемых свойств	55
Представление об аргументах <code>self</code> , <code>_cmd</code> и <code>super</code>	61
Представление об указателе <code>isa</code>	64
Инициализация классов	67
Чтение кодировок типов	69
Применение блоков	71
Глава 3. Управление памятью	75
Сохранение и освобождение объектов из памяти	76
Присваивание переменным экземпляра	77
Автоматический подсчет ссылок	79
Возврат объектов по аргументам-указателям	81
Исключение циклов сохранения	83
Переход к механизму ARC	85
Автоматически освобождаемые пулы	88
Применение автоматически освобождаемых конструкторов	90
Автоматическое освобождение объектов в методах доступа	91
Поддержка автоматической “сборки мусора”	92
Взаимодействие с кодом C	95
Представление о разрушении объектов	97
Применение слабых ссылок	99
Выделение просмотренной памяти	102
Глава 4. Шаблоны, распространенные в Objective-C	105
Поддержка шаблона создания объектов в два этапа	105
Копирование объектов	107

Архивирование объектов	109
Создание выделенных инициализаторов	112
Реализация шаблона-одиночки	114
Делегирование.....	117
Предоставление фасадов.....	118
Создание кластеров классов.....	120
Применение циклов исполнения.....	124
Глава 5. Числа.....	127
Сохранение чисел в коллекциях	129
Выполнение десятичных арифметических операций.....	132
Преобразование символьных строк в числа.....	135
Чтение чисел из символьных строк.....	137
Глава 6. Манипулирование символьными строками	141
Создание строковых констант	142
Сравнение символьных строк.....	143
Посимвольная обработка строк	146
Преобразование кодировок символьных строк.....	149
Обрезка символьных строк.....	151
Разделение символьных строк	153
Копирование символьных строк	154
Построение символьных строк по шаблонам	156
Сопоставление с шаблоном в символьных строках....	159
Сохранение форматированного текста	161
Глава 7. Работа с коллекциями	163
Применение массивов.....	164
Манипулирование индексами	166

Сохранение неупорядоченных групп объектов	168
Создание словаря	170
Циклическое обращение к коллекции	171
Поиск объекта в коллекции	175
Подклассификация коллекций	177
Сохранение объектов Objective-C в коллекциях C++	180
Глава 8. Даты и моменты времени	183
Обнаружение текущей даты	184
Преобразование дат для отображения	185
Расчет истекшего времени	188
Извлечение дат из символьных строк	189
Получение событий от таймера	190
Глава 9. Работа со списками свойств	193
Сохранение коллекций в списках свойств	194
Чтение данных из списков свойств	196
Преобразование форматов списков свойств	199
Применение формата JSON	200
Сохранение пользовательских настроек по умолчанию	202
Сохранение произвольных объектов в пользовательских настройках по умолчанию	206
Глава 10. Взаимодействие с окружением	209
Получение переменных окружения	209
Извлечение аргументов из командной строки	211
Доступ к региональным параметрам пользователя	213
Поддержка внезапного завершения процесса	214

Глава 11. Доступ к значениям по ключам.....	219
Доступ к значениям по ключу	220
Обеспечение совместимости с механизмом KVC	221
Представление о путях к ключам	224
Наблюдение за ключами.....	226
Обеспечение совместимости с механизмом KVO	228
Глава 12. Обработка ошибок	231
Отличия в обработке исключений во время выполнения.....	232
Генерирование и перехват исключений	235
Применение объектов исключений.....	237
Применение единой модели исключений	239
Управление памятью при обработке исключений ..	240
Передача делегатов ошибок.....	243
Возврат значений ошибок	244
Применение класса NSError	245
Глава 13. Доступ к файлам и каталогам	247
Чтение файла	248
Перемещение и копирование файлов	250
Получение атрибутов файлов.....	252
Манипулирование путями к файлам	253
Выяснение факта существования файла или каталога	254
Работа с пакетами.....	256
Поиск файлов в системе	259
Глава 14. Потoki.....	263
Создание потоков	263
Управление приоритетностью потоков	265

Содержание	11
Синхронизация потоков	267
Сохранение данных, характерных для потоков.....	269
Ожидание по условию.....	272
Глава 15. Блоки и Grand Central Dispatch.....	277
Привязка переменных к блокам	278
Управление памятью при использовании блоков	282
Выполнение действий в фоновом режиме	285
Организация специальных очередей работ.....	287
Глава 16. Уведомления	289
Запрашивание уведомлений	290
Отправка уведомлений.....	291
Постановка уведомлений в очередь.....	292
Обмен уведомлениями между приложениями	294
Глава 17. Доступ к сети	299
Заключение сокетов C в оболочку.....	299
Установление связи с серверами	302
Общий доступ к объектам по сети.....	304
Поиск одноранговых узлов в сети.....	306
Загрузка данных по веб-адресу.....	309
Глава 18. Отладка программ на Objective-C.....	313
Инспектирование объектов	313
Выявление ошибок управления памятью	315
Наблюдение за исключениями	318
Утверждение исключений	320
Вывод отладочных сообщений.....	322

Глава 19. Динамическая библиотека Objective-C....	325
Отправка сообщений по имени	326
Поиск классов по имени	327
Проверка реагирования объекта на сообщение.....	329
Пересылка сообщений	330
Поиск классов	332
Инспектирование классов.....	334
Создание новых классов	336
Добавление переменных экземпляра.....	338
Предметный указатель	341

Об авторе

Дэвид Чиснолл является независимым писателем и консультантом. Повышая свое образование с целью получить ученую степень доктора философских наук, он принял участие в основании проекта *Étoile*, предназначенного для разработки открытой настольной среды поверх GNUStep — свободно доступной реализации прикладных программных интерфейсов OpenStep и Cocoa. Он является активным участником проекта GNUStep, первоначальным автором и сопроводителем динамической библиотеки GNUStep Objective-C 2, а также связанной с этим поддержкой компилятора Clang.

Получив ученую степень доктора философских наук, Дэвид занимался некоторое время академической наукой, изучая историю языков программирования. Но в конечном итоге отошел от академической науки, осознав, что есть и другие прекрасные места для приложения его знаний, где не требуется много заниматься бумажной работой. Поэтому он периодически принимает участие в различных проектах по моделированию семантики динамических языков программирования.

Дэвид хорошо знаком со всеми особенностями Objective-C благодаря работе над проектами, в которых используется этот язык программирования, а также над реализацией самого языка. Ему пришлось работать над реализацией и других языков программирования, включая диалекты Smalltalk и JavaScript в качестве надстройки над динамической библиотекой Objective-C, что дает возможность совместно использовать код всех этих языков без промежуточного преобразования.

В свободное от написания книг и программ время Дэвид любит танцевать аргентинское танго и кубинскую сальсу, играть в бадминтон, метать летающие тарелки и вкусно готовить.

Благодарности

Что касается работы над этой книгой, то мне, как ее автору, хотелось бы прежде всего поблагодарить Николаса Роуда (Nicolas Road). Я приобрел свой первый Мак почти одновременно с началом работы над диссертацией на соискание ученой степени доктора философских наук и собирался воспользоваться им для программирования на Java, не желая изучать оригинальный язык данной платформы. В начале работы над диссертацией мне пришлось сотрудничать с Николасом Роудом, активным участником проекта GNUStep. Именно он убедил меня в том, что язык Objective-C и среда Сосоа пригодны не только для Маков, и поэтому заслуживают изучения. И он оказался совершенно прав: сопутствующие интегрированные среды в значительной степени упрощают разработку прикладных программ на Objective-C.

Далее мне хотелось бы выразить благодарность Фреду Кайферу (Fred Keifer). Фред сопровождает реализацию GNUStep в интегрированной среде разработки AppKit. Он выполнил невероятно скрупулезное (до педантизма) научное рецензирование этой книги, обнаружив в ней ряд неразъясненных мест. Если книгу будет приятно читать, то в этом огромная заслуга Фреда.

И наконец, хотелось бы поблагодарить всех, кто воплотил рукопись этой книги в настоящее издание, и в особенности Марка Тэйбера (Mark Taber), предложившего мне первым написать книгу по Objective-C.

Введение

Блез Паскаль однажды написал: “У меня не было времени на короткое письмо, и поэтому я написал длинное”. Этот карманный справочник — самая маленькая книга, которую мне довелось написать, и не скажу, что я с легкостью уместил в ее лаконичном формате все, что мне хотелось бы сказать на данную тему.

Когда Марк Тэйбер предложил мне написать эту книгу, я поначалу не был уверен в том, что это именно то, чем я хотел бы заняться. Ведь фразеологический словарь для родного языка — это перечень кратких идиом для тех, кто хотел бы одним-двумя меткими предложениями выразить важную мысль. Фразеологический словарь для языка программирования должен выполнять аналогичную роль.

Эту книгу не следует рассматривать в качестве исчерпывающего справочного пособия по языку Objective-C. Компания Apple подготовила такое пособие, доступное по адресу <http://developer.apple.com>. Ее не следует рассматривать и в качестве подробного учебного пособия по языку Objective-C, и, в отличие от другой моей книги на данную тему, *Cocoa Programming Developer's Handbook*, вы не найдете в ней примеров исходного кода законченных программ, но множество очень коротких примеров идиом, которыми, надо надеяться, вы сможете успешно пользоваться в самых разных ситуациях программирования на Objective-C.

Если вам доводилось в какой-нибудь книге по программированию встречать примеры кода, который на самом деле оказывался неработоспособным, то вы согласитесь

с тем, что в таких случаях чувствуешь себя попросту обманутым. В этой книге приведены листинги с двумя видами кода. Код на белом фоне предназначен для иллюстрации какой-нибудь простой идеи. Этот код может зависеть от некоторого предполагаемого контекста и не должен рассматриваться в качестве работоспособных и практически полезных примеров.

Большинство примеров кода в этой книге приведены в виде листингов на сером фоне. В нижней части каждого из таких примеров указано имя исходного файла, из которого взят данный листинг. Исходные файлы можно загрузить с веб-страницы этой книги на веб-сайте издательства InformIT по адресу <http://www.informit.com/title/0321743628>.

Во время работы над первым изданием этой книги я написал и протестировал все примеры на платформе Mac OS X. Отослав черновой вариант текста на редактирование, я протестировал их в среде GNUstep и был приятно удивлен тем, что практически все они работали. Ко времени выхода книги в свет эти примеры стали работоспособными полностью. Во втором издании книги рассматривается ряд средств, которые поддерживаются только на платформе Mac OS X 10.7 или iOS 5. Все приведенные здесь примеры также работают в среде GNUstep. Как и прежде, я подготовил все примеры в Mac OS X, не делая никаких уступок для GNUstep (если не считать уступкой тестирование примеров в данной среде).

О версиях книги

Настоящая книга написана в Vim (свободном режимном текстовом редакторе, созданном на основе более старого редактора vi) с использованием семантической разметки. В нем были сгенерированы три разные версии книги. Две из них создаются с помощью программы pdf_latex. Если

вы читаете напечатанный текст книги или его электронный вариант в формате PDF, значит, имеете дело с одной из этих двух версий. Единственное различие между ними состоит в том, что печатная версия содержит метки обрезки страниц на печатной машине.

Третья версия представляет собой XHTML-документ, предназначенный для электронного издания в формате ePub. Она создается в среде EtoileText, которая сначала анализирует разметку в стиле LaTeX для построения древовидной структуры, а затем выполняет некоторые преобразования для обработки перекрестных ссылок и индексирования и, наконец, генерирует XHTML-документ. Весь необходимый для этого код написан на языке Objective-C.

Если у вас есть доступ как к текстовому, так и к электронному варианту книги, то вы можете заметить, что листинги примеров программ в формате издания ePub выглядят немного лучше. Дело в том, что в среде EtoileText для выделения синтаксиса используется пакет сервисных программ SourceCodeKit, еще одна среда типа Йтоилй, в которой для разметки листингов исходного кода служит часть Clang — современного компилятора Objective-C. Это означает, что места, занимаемые листингами в тексте книги, аннотированы точно такими же семантическими типами, которые воспринимает компилятор. Он, например, может отличать вызов функции от реализации макрокоманды.

Весь код, необходимый для выполнения описанного выше форматирования текста книги, вы найдете в хранилище подчиненных версий Etoile по адресу <http://svn.gna.org/viewcvs/etoile/trunk/Etoile/>.

От издательства

Вы, читатель этой книги, и есть главный ее критик и комментатор. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать

лучше и что еще вы хотели бы увидеть изданным нами. Нам интересно услышать и любые другие замечания, которые вам хотелось бы высказать в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо, либо просто посетить наш веб-сайт и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится или нет вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию последующих книг.

Наши координаты:

E-mail: info@williamspublishing.com

Веб: <http://www.williamspublishing.com>

Адреса для писем из:

России: 127055, Москва, ул. Лесная, д. 45, стр.1

Украины: 03150, Киев, а/я 152

Основные принципы Objective-C

Для того чтобы овладеть языком Objective-C, необходимо усвоить основные принципы его построения. В отличие от языков программирования C++, D или Java, которые разработаны как новые C-подобные языки, язык Objective-C является гибридным языком. Он представляет собой чистое надмножество языка C, а это означает, что каждая действительная программа на C является также действительной программой на Objective-C. А кроме того, в нем допускается использовать некоторые Smalltalk-подобные синтаксические конструкции и семантику.

Один из разработчиков языка Objective-C, Том Лав (Tom Love), описал синтаксис квадратных скобок как “указательный столб”, напоминающий о том, что вы покинули область языка C и вступили на “землю объектов”. Первоначальная идея создания Objective-C зародилась в результате поиска подходящего способа для компоновки библиотек C, который способствовал бы слабой связности компонентов.

Одним из фундаментальных проектных решений в Objective-C послужило в отрицание какого бы то ни было “волшебства”. Все детали реализации должны быть открыты для программирующего. В отличие от C++, где подробности таблицы виртуальных функций скрыты от программирующего, Objective-C позволяет просматривать и модифицировать все, что касается объектов и классов.

В старых динамических библиотеках классы Objective-C были представлены структурами C, объявленными открытыми (`public`). Их можно было модифицировать непосредственно или даже создавать новые классы и регистрировать их средствами исполняющей системы. А в новых библиотеках эти структуры объявлены закрытыми (`private`), и для манипулирования ими как непрозрачными типами предусмотрен набор открытых функций.

Понятие объектной модели

В Objective-C используется Smalltalk-подобная объектная модель. Если у вас имеется некоторый опыт программирования на Java, то вам будет нетрудно усвоить Objective-C. Если же вы писали программы на таком языке семейства Simula, как C++, то перейти на Objective-C вам, возможно, будет труднее.

Алан Кей (Alan Kay) описывал идею объектов как простое упражнение в упрощении. Решая некоторую задачу, имеет смысл разделить ее на составные (более простые) части. Простейшее устройство, способное выполнять часть программы, может в то же время выполнять всю программу в целом, и таким устройством является компьютер. Объекты в представлении Алана Кея являются простыми моделями компьютеров, которые связываются друг с другом, обмениваясь сообщениями.

Точно так же ведут себя и объекты в Objective-C и его родителе — языке Smalltalk. Они представляют собой изолированные части программы, которые передают друг другу сообщения. Обычно эти сообщения доставляются синхронно и поэтому действуют подобно вызовам функций, но важно понимать, чем они отличаются.

Сообщения относятся к более высокому уровню абстракции, чем вызовы функций. Вызов функции организуется очень просто. Во многих архитектурах для этого доста-