

Айк Харазян

Язык **Swift**

Простые типы данных в Swift
Операторы и типы коллекций
Условные конструкции и циклы
Функции и замыкания
Перечисления и кортежи
Классы и структуры
ООП в Swift
Расширения и протоколы
Обобщенные типы



Айк Харазян

с а м о у ч и т е л ь

Язык Swift

Санкт-Петербург

«БХВ-Петербург»

2016

УДК 004.438 Swift
ББК 32.973.26-018.1
Х20

Харазян А. А.

Х20 Язык Swift. Самоучитель. — СПб.: БХВ-Петербург, 2016. — 176 с.: ил. — (Самоучитель)

ISBN 978-5-9775-3572-4

Книга предназначена для самостоятельного изучения Swift — нового языка программирования для iOS и OS X. Описана версия Swift 2.0. Материал построен по принципу от более легкого к сложному, изложение сопровождается большим количеством листингов кода, для тестирования и отладки используется новая среда быстрой разработки Playground. Объяснены основы Swift, синтаксис языка и его особенности. Описаны типы данных, условные выражения, циклы, массивы, функции, кортежи, базовые операторы и другие стандартные конструкции. Кратко даны основы объектно-ориентированного программирования. Подробно рассмотрены более сложные или специфические для Swift конструкции: перечисления, замыкания, опциональные типы, классы, структуры, встроенные и обобщенные типы, расширения, протоколы, расширенные операторы и др.

Для программистов

УДК 004.438 Swift
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Екатерина Капальгина</i>
Редактор	<i>Григорий Добин</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн обложки	<i>Марины Дамбиевой</i>

Подписано в печать 31.07.15.

Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 14,19.

Тираж 1000 экз. Заказ №

"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

Первая Академическая типография "Наука"
199034, Санкт-Петербург, 9 линия, 12/28

Оглавление

Введение	9
Для кого предназначена эта книга?	10
Какова польза от книги?	10
Структура книги.....	10
Тестируйте листинги кода!	11
Программа Apple для разработчиков.....	11
ЧАСТЬ I. ОСНОВЫ SWIFT	13
Глава 1. Swift: как он появился и с чего начать?	15
1.1. Как появился Swift?	15
1.2. Что нужно, чтобы начать разрабатывать на Swift?	16
1.3. Playground.....	16
1.4. Как создать новый документ Playground?	16
Глава 2. Особенности синтаксиса Swift	20
2.1. Swift — это C-подобный язык	20
2.2. Отсутствие заголовочных файлов	20
2.3. Точки с запятой ставить не обязательно.....	20
2.4. Набор символов	21
Выводы	21
Глава 3. Простые типы данных, переменные и константы	22
3.1. Переменные и константы	22
3.2. Вывод информации в консоль	24
3.3. Комментарии	24
3.4. Статическая типизация и вывод типов	25
3.5. Явное указание типов	26
3.6. Литералы	27
3.7. Числовые типы.....	27
3.7.1. Целые числа	27
3.7.2. Числа с плавающей точкой.....	28
3.7.3. Способы записи числовых типов	28
3.7.4. Преобразование числовых типов	29

3.8. Строки и символы.....	30
3.8.1. Конкатенация строк.....	30
3.8.2. Преобразование в строку.....	31
3.8.3. Интерполяция строк.....	31
3.9. Логические типы.....	31
3.10. Псевдонимы типов.....	31
Выводы.....	32
Глава 4. Базовые операторы.....	33
4.1. Оператор присваивания.....	34
4.2. Арифметические операторы.....	34
4.3. Составные операторы присваивания.....	34
4.4. Операторы инкремента и декремента.....	35
4.5. Операторы унарного минуса и унарного плюса.....	36
4.6. Операторы сравнения.....	36
4.7. Тернарный условный оператор.....	37
4.8. Операторы диапазона.....	37
4.9. Логические операторы.....	38
4.9.1. Логическое НЕ.....	38
4.9.2. Логическое И.....	38
4.9.3. Логическое ИЛИ.....	38
Выводы.....	38
Глава 5. Типы коллекций.....	40
5.1. Массивы.....	40
5.1.1. Объявление массива.....	40
5.1.2. Получение доступа к элементам массива.....	42
5.1.3. Добавление элементов в массив.....	43
5.1.4. Изменение элементов массива.....	44
5.1.5. Удаление элементов из массива.....	45
5.1.6. Итерация по массиву.....	46
5.2. Множества.....	46
5.2.1. Объявление множеств.....	47
5.2.2. Работа с множествами.....	47
5.2.3. Сочетание и сравнение множеств.....	49
5.3. Словари.....	50
5.3.1. Объявление словаря.....	50
5.3.2. Получение доступа к элементам словаря.....	51
5.3.3. Добавление элементов в словарь.....	51
5.3.4. Изменение элементов словаря.....	51
5.3.5. Удаление элементов из словаря.....	52
5.3.6. Итерация по словарю.....	52
Выводы.....	53
Глава 6. Ветвление потока.....	55
6.1. Условия.....	55
6.1.1. Условный оператор <i>if</i>	55
6.1.2. Оператор <i>switch</i>	57

6.2. Циклы	61
6.2.1. Циклы <i>for</i>	61
Стандартный цикл <i>for</i>	61
Цикл <i>for-in</i>	62
6.2.2. Цикл <i>while</i>	63
6.3. Управление потоком цикла.....	64
6.4. Оператор <i>guard</i>	65
6.5. Проверка на доступность API.....	66
Выводы	66
Глава 7. Функции	67
7.1. Объявление функции	67
7.2. Параметры	68
7.2.1. Внешние имена параметров	69
7.2.2. Параметры со значением по умолчанию.....	70
7.2.3. Сквозные параметры.....	70
7.2.4. Функции с переменным числом параметров	71
7.3. Возвращаемое значение функции	72
7.3.1. Функции с несколькими возвращаемыми значениями	73
7.4. Функции — объекты первого класса	75
7.4.1. Функции, принимающие параметры в виде функции	75
7.4.2. Функции, возвращающие функцию	76
7.4.3. Вложенные функции	77
Выводы	77
ЧАСТЬ II. УГЛУБЛЕННОЕ ИЗУЧЕНИЕ SWIFT	79
Глава 8. Опциональные типы	81
8.1. Опциональная привязка	83
8.2. Принудительное извлечение	83
8.3. Неявное извлечение.....	84
8.4. Опциональное сцепление	85
Выводы	85
Глава 9. Кортежи	86
9.1. Объявление кортежа	86
9.2. Получение доступа к элементам кортежа.....	86
9.2.1. Использование индекса элемента	86
9.2.2. Разложение кортежа.....	87
9.3. Именованное элементов кортежа.....	87
9.4. Использование кортежей	88
9.4.1. Массовое присвоение.....	88
9.4.2. В циклах <i>for-in</i>	88
9.4.3. В качестве возвращаемого значения для функций	88
9.5. Опциональный кортеж	89
Выводы	90
Глава 10. Замыкания.....	91
10.1. Сокращенные имена параметров замыкания	94

10.2. Операторы-функции	94
10.3. Последующее замыкание	94
Выводы	96
Глава 11. Перечисления	97
11.1. Объявление перечислений	98
11.2. Перечисления и оператор <i>switch</i>	98
11.3. Связанные значения	99
11.4. Исходные значения	100
11.5. Вложенные перечисления	102
Выводы	103
Глава 12. Классы	104
12.1. Свойства, методы и объекты класса	104
12.2. Объявление классов	105
12.3. Свойства класса	105
12.3.1. Ленивые свойства	107
12.3.2. Вычисляемые свойства	108
12.3.3. Наблюдатели свойств	111
12.3.4. Вычисляемые переменные и наблюдатели для переменных	111
12.3.5. Свойства типа	112
12.4. Инициализаторы	113
12.4.1. Инициализатор по умолчанию	113
12.4.2. Инициализаторы с параметрами	114
12.4.3. Локальные и внешние имена параметров инициализатора	116
12.4.4. Проваливающиеся инициализаторы	117
12.4.5. Деинициализаторы	118
12.5. Методы	118
12.5.1. Создание методов	118
12.5.2. Методы типа	120
12.6. Индексаторы	121
12.6.1. Синтаксис индексаторов	121
12.6.2. Многомерные индексаторы	121
Выводы	124
Глава 13. Наследование	126
13.1. Переопределение	127
13.2. Наследование инициализаторов	127
13.3. Переопределение инициализаторов	131
13.4. Назначенные и удобные инициализаторы	132
13.5. Необходимые инициализаторы	134
Выводы	135
Глава 14. Автоматический подсчет ссылок	136
14.1. Принципы работы автоматического подсчета ссылок	136
14.2. Циклы сильных ссылок внутри объектов классов	137
14.3. Решение проблемы циклов сильных ссылок между объектами классов	138
14.3.1. Слабые ссылки	139
14.3.2. Ссылки без владельца	139
Выводы	140

Глава 15. Структуры	141
15.1. Типы-значения и ссылочные типы	141
15.2. Оператор идентичности	141
15.3. Свойства структур	142
15.4. Свойства типа для структур	143
15.5. Методы структур	144
15.6. Методы типа для структур	144
15.7. Инициализаторы структур	145
Выводы	145
Глава 16. Проверка типов и приведение типов	146
16.1. Проверка типов	146
16.2. Приведение типов	148
16.3. Проверка типов <i>Any</i> и <i>AnyObject</i>	150
16.3.1. Тип <i>AnyObject</i>	150
16.3.2. Тип <i>Any</i>	151
Выводы	152
Глава 17. Расширения	153
17.1. Расширение свойств	153
17.2. Расширение методов	154
17.3. Расширение инициализаторов	155
Выводы	155
Глава 18. Протоколы	156
18.1. Объявление протокола	156
18.2. Требования для свойств	156
18.3. Требования для методов	158
18.4. Требования для инициализаторов	160
18.5. Протоколы как типы	160
18.6. Соответствие протоколу через расширение	161
18.7. Наследование протоколов	161
18.8. Протоколы только для классов	161
18.9. Сочетание протоколов	162
18.10. Проверка объекта на соответствие протоколу	162
18.11. Расширения протоколов	162
Выводы	163
Глава 19. Обобщенные типы	164
19.1. Обобщенные функции	164
19.2. Обобщенные типы	165
19.3. Ограничения типов	166
Выводы	166
Глава 20. Обработка ошибок	167
Выводы	168
Глава 21. Расширенные операторы	169
21.1. Оператор объединения по нулевому указателю	169

21.2. Операторы с переполнением	169
21.2.1. Переполнение значения	170
21.2.2. Потеря значения	170
21.3. Перегрузка операторов.....	170
21.4. Побитовые операторы	171
21.4.1. Побитовый оператор <i>NOT</i>	171
21.4.2. Побитовый оператор <i>AND</i>	171
21.4.3. Побитовый оператор <i>OR</i>	172
21.4.4. Побитовый оператор <i>XOR</i>	172
21.4.5. Побитовые операторы левого и правого сдвига	172
Выводы	172
Заключение.....	173
Изучайте фреймворки Apple.....	173
Вступайте в Apple’s Developer Program.....	173
Вперед, к новым высотам!	173

Введение

2 июня 2014 года на ежегодной конференции разработчиков WWDC компания Apple представила новый язык программирования Swift. С его помощью можно создавать приложения для iOS и OS X с еще большей легкостью, чем ранее. Swift сочетает в себе производительность и эффективность компилируемых языков программирования с простотой и интерактивностью популярных скриптовых языков. Он был вдохновлен такими языками программирования, как JavaScript, Python и Ruby.

При создании языка Swift разработчики руководствовались тремя парадигмами:

- быстрота;
- современность;
- надежность.

По словам Apple, некоторые алгоритмы, написанные на Swift, выполняются в разы быстрее, чем те же алгоритмы на Python. Этим она утверждает, что код, написанный на Swift, не только лаконичен и легко читаем, но и может конкурировать по скорости с мощными скриптовыми языками. Современность языка демонстрируют особенности, перенятые из скриптовых языков программирования, — такие как интерактивность и простота восприятия. Разработчики очень тщательно подошли к реализации возможностей, которые помогают противостоять появлению ошибок, тем самым делая код более надежным. Например, переменные всегда должны быть объявлены перед их использованием, а массивы и целые числа проверяются на перегрузку. Память очищается автоматически, а технология опциональных типов позволяет исключить возможные ошибки с отсутствием значения.

Еще одной важной особенностью Swift являются песочницы Xcode Playground, которые позволяют тестировать код и видеть результат его выполнения в реальном времени. До их появления приходилось экспериментировать на кусках кода внутри целых проектов.

Для кого предназначена эта книга?

Необходимость изучения нового языка программирования может сильно пугать новичков. Но эта книга прекрасно подойдет даже тем, кто никогда не использовал языки программирования C, C++ или Objective-C. И хотя книга не рассчитана на тех, кто знакомится с языками программирования впервые, но, все же, в начале каждой главы даны краткие объяснения основ программирования, касающиеся освещаемой темы.

Опытных же разработчиков в первую очередь заинтересуют особенности и новшества языка Swift, которым уделено особое внимание, а также материалы второй части книги по углубленному изучению программирования на Swift.

Изложение каждой новой темы в книге сопровождается подробными примерами на языке Swift. А сами главы построены по принципу последовательного изучения материала от более легкого к более сложному. Это облегчает поиск нужных глав для опытных разработчиков и помогает новичкам изучать Swift постепенно.

Какова польза от книги?

В книге рассмотрены все особенности языка Swift, которые в будущем помогут вам при разработке приложений и игр под iOS и OS X. Изучив материал книги, вы сможете полностью разбираться в коде, написанном на Swift, а также самостоятельно разрабатывать алгоритмы на этом языке. Знания, изложенные в книге, лежат в основе фреймворков Cocoa и Cocoa Touch, служащих для разработки приложений и игр под iPhone, iPad, Mac и т. д. Так что, изучив техники языка программирования Swift, вы смело можете приступить к изучению API и фреймворков и разработке приложений для iOS и OS X.

Структура книги

Книга состоит из двух больших частей.

- **Часть I. Основы Swift.** Первая часть книги знакомит читателя с тем, как реализованы в Swift стандартные структуры вроде циклов, условий и массивов. Она будет полезна для начинающих разработчиков, поскольку расскажет им об основах программирования. А опытным разработчикам эта часть даст понять, какие особенности есть в простых структурах Swift в отличие от других языков.
- **Часть II. Углубленное изучение Swift.** Вторая часть книги описывает более сложные структуры и понятия, которые не встречаются в других языках программирования. Она будет полезна для опытных разработчиков, которые хотят углубиться в разработку на Swift более сложных приложений. Для новичков же важно сначала изучить главы из первой части, и только затем приступить ко второй.

Тестируйте листинги кода!

Каждая глава книги сопровождается большим количеством листингов кода. Они позволяют лучше понять некоторые алгоритмы и операции. Мы рекомендуем вам по ходу изучения материала набивать предлагаемые программные коды непосредственно вручную. Так вы легче запомните синтаксис языка Swift, поскольку будете учиться на своих ошибках.

А помогут вам в этом уже упомянутые песочницы Playground — новый тип документа Xcode, который появился в его шестой версии, вместе с языком Swift. Playground представляет собой самостоятельный файл, куда можно вписать несколько строк кода, которые будут тут же выполнены с выводом полученного результата. Более подробно о Playground рассказано в *главе 1*.

Программа Apple для разработчиков

Чтобы получать последнее программное обеспечение для разработчиков и иметь возможность публиковать свои приложения в магазине цифровой дистрибуции AppStore, вы, вероятно, захотите стать членом Apple Developer Programs. Найти подробную информацию о привилегиях, предоставляемых вступившим в эту программу разработчикам, вы можете на странице сайта Apple по адресу: <https://developer.apple.com/programs>.



ЧАСТЬ I

Основы Swift

В первой части этой книги мы познакомимся с простыми элементами Swift, которые встречаются в большинстве языков программирования. Но сначала мы узнаем, как Swift появился на свет, чем разработчики пользовались до него и с чего следует начать с ним знакомство.

- ❑ Глава 1. Swift: как он появился и с чего начать?
- ❑ Глава 2. Особенности синтаксиса Swift.
- ❑ Глава 3. Простые типы данных, переменные и константы.
- ❑ Глава 4. Базовые операторы.
- ❑ Глава 5. Типы коллекций.
- ❑ Глава 6. Ветвление потока.
- ❑ Глава 7. Функции.

ГЛАВА 1



Swift: как он появился и с чего начать?

1.1. Как появился Swift?

Язык для человека является универсальным средством коммуникации и обмена информацией. С его помощью мы можем высказать наши мысли друзьям, членам семьи или коллегам по работе. И компьютерные языки программирования в этом плане также не являются исключением. Принцип выражения своих мыслей в них тот же. Только на этот раз мы высказываем их компьютеру.

Языки программирования, подобно человеческим языкам, существуют в разных формах и со временем эволюционируют и совершенствуются. Главной задачей языков программирования изначально являлась необходимость взаимодействовать с инструкциями компьютера, чтобы задать набор определенных действий, которые мы хотим от него получить.

Опытные пионеры компьютерной эпохи знают, что центральный процессор компьютера понимает только набор инструкций, состоящий из нулей и единиц. Чтобы разработчики могли разрабатывать свой код на более понятном им языке, чем нули и единицы, существуют программы-компиляторы, которые переводят код, написанный на определенном языке программирования, в инструкции для процессора, состоящие из тех самых нулей и единиц.

Компромисс между возможностями языка программирования и его производительностью обеспечили такие языки программирования, как C и C++. Но пока языки C и C++ распространялись на все большее количество платформ, Apple решила пойти иным путем и разработала язык Objective-C. Построенный на основе C, Objective-C сочетал в себе мощь широко используемого языка программирования с объектно-ориентированными методологиями. На протяжении многих лет Objective-C являлся основой для разработки программ для компьютеров Macintosh и iOS-устройств.

Однако, несмотря на производительность и простоту, Objective-C носил с собой весь багаж, наследованный от C. Для разработчиков, которые разбираются в C, это не вызывает трудностей. Но новички жаловались, что Objective-C труден для понимания и разработки.