

*Посвящается моей девушке, Хуэйи Нган (Ngan Huynh),
с благодарностью за любовь, большую поддержку
и воодушевление*

Содержание

Об авторе	11
О техническом консультанте	12
Благодарности	13
Предисловие	14
Глава 1. Введение в оптимизацию производительности iOS	15
Новая эра смартфонов	15
Почему важна производительность	15
Для кого предназначена эта книга	16
Стиль автора	16
Что вам потребуется	17
Как пользоваться этой книгой	18
Обзор книги	18
Загружаемые примеры	20
Обратная связь	20
Глава 2. Оценка производительности с использованием эмулятора и действующего устройства	21
Эмулятор и действующее устройство	22
Память и производительность	23
Инструменты	23
Основные инструменты	24
Инструмент Allocations (Распределение памяти)	25
Унаследованный код	29
Инструменты оценки производительности	34
Выводы	55
Упражнения	55

Глава 3. Увеличение производительности

UITableView	57
Предисловие к примерам	58
Повторный обзор инструментов	58
Первый пример	59
Второй пример	69
Чему учат данные примеры	73
Другие приемы	74
Сохранение значения высоты	74
Прозрачность	75
Отключение графических эффектов	76
Производительность редактирования/переупорядочивания	76
Выводы	77
Упражнения	79

Глава 4. Кеширование изображений и данных 80

Разница производительности обработки сети, файлов и памяти ...	80
Выявление узкого места	81
Введение в кеширование	83
Что такое кеширование?	83
Попадание в кеш	84
Промах кеша	84
Издержки извлечения	84
Издержки хранения	85
Недостоверность кеша	85
Политика вытеснения	86
Алгоритмы кеширования	87
Оценка кеша	94
Что следует кешировать	95
Где хранить изображения	95
Кеширование данных	100
Выводы	109
Упражнения	110

Глава 5. Оптимизация алгоритмов и структур данных 111

Первый пример	111
Теоретические вопросы тестирования производительности алгоритмов	114
Как измерить «большое O»	114
Сведения о реализации	116

«Большое О» известных алгоритмов.....	117
Практическая оценка.....	118
Структуры данных и алгоритмы	119
Структуры данных Сосоа Touch	120
Прочие структуры данных.....	132
Двоичное дерево	147
Граф.....	152
Прочие алгоритмы и подходы к решению проблем.....	159
Рекурсия	160
Парсер SAX/DOM для xml-парсинга	161
Выводы	163
Упражнения.....	164

Глава 6. Ускорение доступа к данным с использованием приемов многопоточного программирования

Что такое потоки и многопоточность	166
Терминология	168
Первый пример	169
Преимущества многопоточности	172
Создание многопоточных приложений	174
Создание потока	174
Настройка потока	182
Начало работы потока	183
Опасности, связанные с потоками	187
Синхронизация потоков	205
Альтернативы потокам	207
Инструмент Thread States (Состояние потоков)	211
Выводы	212
Упражнения.....	212

Глава 7. Оптимизация использования памяти

Краткий обзор.....	214
Старая политика владения объектом.....	214
Автоматическое освобождение объектов.....	214
Пул автоматически освобождаемых объектов	215
Автоматический подсчет ссылок	217
Как настроить проект в XCode	217
Политика автоматического подсчета ссылок.....	218
Новые модификаторы для автоматического подсчета ссылок ..	219
Свойства объектов.....	220
Дополнительные вопросы	221

Удерживающие циклические ссылки	221
Слабые ссылки	222
UIViewController	222
Процесс загрузки представления	223
Процесс выгрузки представления	224
Отображение и сокрытие представлений	225
Копирование объектов	226
Поверхностное и полное копирование	227
Реализация полного копирования	227
Интегрирование метода копирования в объект	229
Дополнительные возможности пула автоматически освобождаемых объектов	230
Пул автоматически освобождаемых объектов и стек	230
Пулы автоматически освобождаемых объектов и потоки	231
Влияние пулов автоматически освобождаемых объектов на производительность	231
Инструменты	232
Статический анализатор	232
Инструмент Leaks (Утечки памяти)	233
Объекты-зомби	233
Инструмент Allocations (Распределение памяти)	234
Уровни предупреждений о нехватке памяти	235
Выводы	235
Упражнения	235

Глава 8. Использование приемов многопоточного программирования и эффективного управления памятью в многозадачных приложениях	236
Что представляет собой многозадачность в iPhone	236
Жизненный цикл многозадачности	237
Преимущества и издержки многозадачности	244
Фоновые службы	245
Служба воспроизведения звука	245
Отображение экрана-заставки	247
Служба определения географического местоположения	247
Локальное уведомление	251
IP-телефония (VOIP)	252
Выполнение в фоновом режиме	252
На что обратить внимание при работе в фоновом режиме	255
Уведомления о системных изменениях	257
Совместимость с разными версиями iOS	258

Выводы	259
Упражнения.....	259
Глава 9. Улучшение производительности с использованием C/C++	260
Преимущества и недостатки	261
Основы программирования на C и C++.....	262
Программирование на C	262
Программирование на C++	275
Практический пример	282
SQLite.....	282
Интеграция C++ в приложение	284
Выводы	284
Упражнения.....	285
Глава 10. Сравнение проблем производительности устройств на базе Android и Windows Phone	286
Общий обзор.....	287
Тестирование в эмуляторе и на действующих устройствах	288
Эмулятор и действующие устройства.....	288
Оценка производительности	289
Производительность ListView	291
Android	292
Windows Phone	295
Кеширование данных	296
Android	297
Windows Phone	299
Структуры данных и алгоритмы	301
Многопоточность	302
Android	303
Windows Phone	306
Управление памятью	307
Android	308
Windows Phone	309
Многозадачность	311
Android	311
Поддержка программирования на C/C++	312
Выводы	313
Упражнения.....	314
Предметный указатель	315

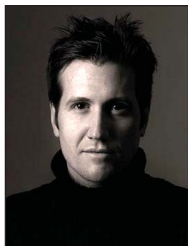


Об авторе



Во Ханг (Khang Vo) – программист и предприниматель, любит работать с новейшими технологиями и продуктами. Занимается разработками для платформы iOS с 2009 года. Ему нравится делиться своими идеями и обсуждать различные технические и организационные аспекты, которые помогают вывести потребителей на новый уровень. В основном он занимается созданием и продажей приложений в Apple App Store и Android Market. В настоящее время он также является студентом университета Карнеги-Меллона.

О техническом консультанте



Эван Койн Мэлони (Evan Coyne Maloney) учился программировать самостоятельно, после того как получил в наследство компьютер Apple IIe. Будучи подростком, в середине 1980-х он опубликовал операционную систему для линейки компьютеров Apple II, которая называлась FoscilDOS и была широко освещена журналами «Byte» и «A+». В 1994 году Эван начал заниматься разработкой программного обеспечения для Интернета и создал чат «KeepTalking» – первую браузерную чат-систему. В 1996-м Эван написал сетевую игру «DarkHorse», имитирующую политическую кампанию, для MSNBC.com. Во время президентской кампании, проходившей в том же году, было зафиксировано много миллионов человеко-часов, проведенных в ней, и она даже была использована на занятиях по политологии в высших учебных заведениях и колледжах. С 2001-го Эван посвятил себя торговле мобильным контентом. Он считается автором и создателем нескольких первых версий приложений для iPhone и iPad, удостоенных награды News Pro, учрежденной агентством «Reuters». В 2010-м Эван присоединился к «Gilt Groupe», где сейчас работает главным инженером высокопопулярных приложений iOS.



Благодарности

Автор выражает благодарность Стиву Энглину (Steve Anglin), предложившему идею создания этой книги и помогавшему в начале пути. Автор также благодарит Эвана Мэлони (Evan Maloney) за полезные предложения, касающиеся технической стороны книги. Эван во многом помог при ее написании.

Автор также признателен Тому Уэлшу (Tom Welsh) за помощь в создании простого и понятного читателю текста. Он предложил большое количество замечательных идей, которые привели книгу к этому окончательному варианту. Автор благодарен Корбину Коллинзу (Corbin Collins) за его четкие и исчерпывающие ответы на вопросы.

Большое спасибо разработчикам и всем, кто помогал справиться с техническими трудностями в процессе написания книги. Это позволило понять, чего не хватает разработчикам и как помочь им обрести необходимые знания и навыки.



Предисловие

Эта книга призвана помочь вам отточить навыки разработки для iOS в узкой области: оптимизации производительности. Она предназначена для тех, кто уже обладает базовыми навыками в разработке для платформы iOS и желает создавать лучшие приложения.

Одержимый вопросами производительности приложений, я потратил массу времени на изучение и применение различных приемов оптимизации приложений в различных платформах, таких как веб и смартфоны. Мне нравится дискутировать на эту тему с людьми. Проводя много времени на форумах и в сообществах iOS, таких как Stack Overflow и Apple Developer Forum, я понял, что большинство разработчиков iOS задают одни и те же вопросы о том, как улучшить производительность своих приложений. Я подумал, что было бы полезно собрать наиболее общие из этих вопросов в хорошо написанную и структурированную книгу, чтобы с легкостью можно было получить полное представление о проблемах оптимизации производительности приложений для платформы iOS. Такова была моя мотивация для написания этой книги, я сделал все возможное, чтобы охватить самые распространенные проблемы и ошибки, встречающиеся разработчикам.

Кроме того, я встречал и отмечал в своих заметках множество проблем, общих для платформ iOS, Android и Windows Phone. Последняя глава основана на этих заметках. Я думаю, что она действительно будет полезна тем, кто желает работать на этих трех платформах или перейти с одной платформы на другую.

При работе с острыми проблемами производительности полезно рассмотреть их с разных сторон и выработать компромисс между производительностью и сложностью реализации выбранного решения. Бывают такие незначительные проблемы, которые заставляют совершать множество ошибок, если заранее не знать возможных способов их решения. Я надеюсь, что эта книга поможет вам избежать таких ошибок и вместо этого потратить свое время на улучшение производительности и создание у пользователей лучших впечатлений о приложении.



Глава 1. Введение в оптимизацию производительности iOS

В данной главе приводится общая информация о книге, включая следующее:

- кому эта книга послужит лучше всего;
- вопросы, которые охватывает книга;
- общая структура и стиль книги.

Новая эра смартфонов

В настоящее время рынок iOS насчитывает сотни тысяч приложений и сотни миллионов пользователей. Все это делает его достаточно обширным как для компании, так и для отдельного разработчика. Этот рынок рос в течение многих лет и будет еще расти, наряду со спросом на интересные и мощные приложения. Имея хорошую задумку для нового приложения, необходимо продумать ее исполнение, включая создание хорошего пользовательского интерфейса. Исходя из уникальных технических характеристик *смартфонов*, высокая производительность – необходимое условие для любого приложения. Пользователи хотят видеть приложение, которое быстро откликается на их действия, мгновенно считывает данные и отображает их.

Почему важна производительность

Оптимизация производительности заключается не только в разработке оптимальных алгоритмов, структур данных и приемов эффективного управления памятью, но и в том, чтобы создать у человека ощущение, что приложение отвечает на любое его действие настолько быстро, насколько это возможно. Таким образом, оптимизация производительности приложений для iPhone играет важную роль. Пользователям необходимо чувство, что их команды воспринимаются и исполняются почти мгновенно. Что, если при нажатии кнопки

реакции пришлось бы ожидать в течение двух долгих секунд? Кого порадует такая производительность? Скорее всего, большинство пользователей были бы весьма расстроены.

Конечно, большую часть данных и их обработку можно было бы переместить в сеть с тысячами серверов, способных быстро обрабатывать и возвращать результат. Однако этого явно недостаточно. *Передача данных через сеть* – сложный процесс, и пользователям, скорее всего, придется несколько секунд ждать получения данных.

Затруднения в улучшении производительности своих продуктов испытывают и разработчики основных приложений, и разработчики игр.

Для кого предназначена эта книга

В основном данная книга написана для разработчиков начального и среднего уровней, которые уже владеют основами программирования для iPhone. Она для тех, кто стремится обеспечить высокую производительность и желает создавать инновационные, быстро реагирующие и готовые к распространению приложения. Даже опытные разработчики смогут найти что-то полезное для себя в этой книге.

Если вы намерены глубоко погрузиться в мир программирования приложений для смартфонов, эта книга даст достаточный объем знаний, чтобы можно было применить все, что вы знаете об iOS, к средам *Android* и *Windows Phone*.

Стиль автора

Я искренне верю, что принцип обучения на практике для программиста является лучшим способом приобрести необходимые навыки. Книга базируется именно на этом принципе. Я рассматриваю общие и частные вопросы с высоты моего двухгодичного опыта разработки для iPhone и более длительного периода разработки и преподавания *Java*. Проблемы, которые я поставлю перед вами, помогут избежать или исправить множество ошибок, связанных с производительностью, при разработке программ для iPhone. Я выбрал эти вопросы, исходя из опыта и исследования тем, наиболее популярных на форумах и в социальных сетях (таких как *Stack Overflow*). Я выделил общие подводные камни и дам вам советы, которые помогут их избежать.

Книга представляет собой совокупность основополагающих принципов, пояснительных иллюстраций и примеров исходного кода. Вместо предоставления конкретного инструмента для решения какой-либо проблемы я хочу предложить вам прочные навыки для каждодневного применения. Я использую разные подходы к изложению материала: иногда иллюстрация ценнее тысячи слов, некоторые идеи проще объяснить примером программного кода, а в некоторых случаях требуется та самая тысяча слов.

Один из лучших способов начать обучение – это разработать крутое приложение, которое бы вам очень нравилось. Такой практический опыт даст больше, чем некоторые теоретические и быстро забываемые примеры.

Совсем необязательно много знать о *фреймворке Cocoa Touch*, потому что я объясню основной синтаксис и классы, необходимые для улучшения производительности приложений. Каждая глава состоит из отдельной темы, при этом некоторые темы могут быть вам уже знакомы. Эту книгу можно также использовать как общий справочник. Как только у вас возникнет какая-то конкретная проблема, вы сможете отыскать описание и прочесть о ее решении.

Главы имеют простой формат: краткий обзор главы, за которым следуют разделы и подразделы. Каждая глава заканчивается выводами, помогающими обобщить полученные знания и напоминающими о самых важных вопросах. Затем следуют несколько простых практических упражнений, чтобы вы могли с удовольствием попрактиковать то, о чем только что узнали.

Что вам потребуется

Как разработчику для платформы iOS вам потребуется *Mac OS с установленным пакетом Xcode*. Бесплатная версия Xcode предоставляется зарегистрированным разработчикам для iOS, или ее можно загрузить непосредственно с сайта Apple Mac AppStore. Вам также понадобится экземпляр этой книги со всеми примерами программного кода, которые можно загрузить с сайта издательства «Apress». Примеры проектов были протестированы в среде разработки Xcode 4.2 с включенным режимом *автоматического подсчета ссылок* (Automatic Reference Counting – ARC), поэтому они будут запускаться в этой среде без всяких проблем.

Вы можете и должны опробовать каждый пример, чтобы лучше понять рассматриваемый вопрос. В книге также встречаются корот-

кие блоки программного кода, не привязанные ни к какому проекту, их тоже следует проработать.

Как пользоваться этой книгой

Хотя главы не совсем близки по смыслу, для более полного понимания проблем производительности и приемов оптимизации лучше читать от начала до конца. Главы могут в чем-то перекликаться между собой. Более поздние главы были написаны с учетом того, что вы уже прочли или знакомы с информацией из предыдущих глав.

Также рекомендуется читать каждую главу полностью, от начала до конца. Главы открываются коротким концептуальным введением в тему. Далее теория совмещается с практическими примерами, чтобы помочь овладеть темой полностью.

Вам следует внимательно читать заключительные разделы глав, поскольку они еще раз напоминают о ключевых вопросах, которые необходимо запомнить. Я также рекомендую выполнить все упражнения, поскольку это позволит закрепить полученные знания.

Обзор книги

Эта книга представляет собой хорошее сочетание базовых концепций и практических знаний, приемов и рекомендаций, которые помогут добиться успеха в конкурентном мире разработки приложений для iOS. Девять глав этой книги охватывают девять различных подходов к решению проблем производительности, возникающих при разработке приложений для iOS.

- ❑ *Глава 2.* Знакомит с рядом средств и инструментов, со способами их применения. Многие разработчики не используют данные средства просто потому, что не знают об их существовании.
- ❑ *Глава 3.* Как разработчику для iOS вам определенно придется пользоваться как простыми, так и сложными табличными представлениями (UITableView) для отображения списков данных или опций. Проблема архитектуры UITableView заключается в том, что как только вы начинаете ее настраивать, страдает эффективность прокрутки. Эта проблема обязательно возникнет, пусть и в слабой мере. В данной главе будут представлены несколько способов повысить эффективность прокрутки представлений UITableView.

- ❑ *Глава 4.* Может показаться, что большинство проблем с производительностью решаются при помощи облачных технологий и нескольких дополнительных серверов в системе. Даже если это и так, передача данных по сети всегда будет оставаться проблемой. Передача данных будет оставаться узким местом еще долгие годы. Вы должны знать, как кешировать данные в локальной памяти, в такой ограниченной среде, как в iOS.
- ❑ *Глава 5.* Структуры данных и алгоритмы, используемые в приложениях для iOS, сходны и в то же время отличны от структур данных и алгоритмов, применяемых в других средах. Фреймворк обеспечивает высокий уровень поддержки множества базовых структур данных, таких как массивы, множества и словари. В некоторых случаях можно просто поместить их в сеть, тогда как в других, особенно когда сбор и обработка данных выполняются для визуализации, все равно придется опираться на среду iOS.
- ❑ *Глава 6.* Под улучшением производительности также понимается уменьшение времени отклика приложения на действия пользователя. Это означает, что основной поток выполнения пользовательского интерфейса не должен блокироваться. Многопоточность может помочь увеличить не только скорость реакции, но и производительность приложения в целом. Многопоточность – сложная тема для любой платформы. Здесь она раскрывается рядом иллюстраций, примеров и доступными объяснениями.
- ❑ *Глава 7.* С появлением нового инструмента автоматического управления памятью разработчики теперь могут пользоваться его преимуществами, чтобы избежать таких типичных проблем с памятью, как утечка или нехватка памяти. Данная глава рассказывает, как лучше использовать память и когда следует загружать данные в память и выгружать их. Здесь также рассказывается о механизме автоматического подсчета ссылок (ARC), включенном в новый *набор средств разработки* (Software Development Kit – SDK), чтобы убедиться, что вы правильно понимаете и используете его.
- ❑ *Глава 8.* В версиях iOS 4 и выше все приложения могут пользоваться преимуществом многозадачности для создания благоприятных впечатлений у пользователей. В сущности, это даже не совсем многозадачность, а скорее механизм быстрого переключения между приложениями с некоторой специаль-

ной фоновой обработкой (приложения не могут выполняться в фоновом режиме). Данная глава поможет понять, какие компоненты будут поддерживаться iOS и какие задачи можно выполнять в фоновом режиме.


- *Глава 9.* Во многих приложениях для iPhone нет необходимости использовать программный код на C/C++ для реализации каких-то возможностей. Однако серьезной проблемой становятся ситуации, когда это действительно необходимо, особенно при интеграции с внешними библиотеками. Возможно, и не понадобится писать все приложение на C/C++, однако необходимо понимать, как эти языки работают, чтобы не допустить ошибок.
- *Глава 10.* На этом этапе ваше представление обо всем разнообразии аспектов производительности iPhone уже будет полностью сформировано. Вы наверняка задумаете в скором времени перенести свое приложение на устройства с Android или Windows Phone. Поэтому последняя глава посвящена сходству проблем производительности в iOS, Android и Windows Phone, что поможет сгладить практические нюансы при освоении новых платформ.

Загружаемые примеры

Примеры с исходным программным кодом можно загрузить на странице этой книги, находящейся на веб-сайте издательства «Apress» (www.apress.com), и опробовать их самостоятельно.

Обратная связь

Если у вас возникнут какие-либо вопросы, пожалуйста, напишите мне по электронной почте vodkhang@gmail.com или посетите мой веб-сайт <http://vodkhang.com>. Я буду рад обсудить проблемы производительности на платформе iPhone.



Глава 2. Оценка производительности с использованием эмулятора и действующего устройства

Данная глава посвящена следующим вопросам:

- различия между эмулятором и действующим устройством;
- влияние управления памятью на производительность приложения;
- инструменты и способы оценки производительности приложения, включая следующее:
 - базовые инструменты измерения производительности и расходования памяти;
 - улучшенные инструменты исследования различных аспектов управления памятью, таких как утечки или неоптимальное использование памяти;
 - улучшенные инструменты исследования различных аспектов, влияющих на производительность, таких как аккумулятор, необходимость загрузки файлов и отображение информации;
 - разделение программы на более мелкие части с целью упростить выявление узких мест в производительности.

Для улучшения производительности необходимо провести оценочное тестирование, чтобы увидеть, в чем заключается проблема. Чтобы провести тестирование с пользой, необходимо понимать причины, ведущие к замедленной работе программы или сегмента кода.

На начальном этапе придется принять два фундаментальных решения: что выбрать – эмулятор или действующее устройство и как сохранить баланс между эффективным использованием памяти и высокой производительностью программного кода.

Для начала необходимо понять разницу между эмулятором и действующим устройством.

Эмулятор и действующее устройство

Главная причина низкой производительности приложений для iPhone состоит в том, что они выполняются в среде с ограниченными ресурсами и невысокой скоростью обработки. *Эмулятор среды iPhone* работает гораздо быстрее настоящего устройства. В сущности, скорость работы *эмулятора* зависит от скорости работы компьютера, на котором он выполняется. В результате очень большим и неприятным сюрпризом может явиться тот факт, что программа действительно быстро выполняется в *эмуляторе* и гораздо медленнее – на *действующем устройстве*. Я встречал множество людей, которые винили сеть в плохой производительности приложения. Безусловно, это иногда верно. Однако в большинстве случаев производительность приложения может сильно страдать именно из-за особенностей реализации кода, а не из-за сетевых проблем. Поэтому тщательное тестирование и оценка производительности приложения основными инструментами в стандартных средах позволят получить полное представление о скорости работы приложения и его пользовательского интерфейса.

Чтобы продемонстрировать значительную разницу между эмулятором и действующим устройством, я протестировал программу в эмуляторе iPhone и на действующем устройстве iPhone. Результаты впечатляют:

- ❑ в эмуляторе iPhone вычисления занимают 0,5 сек;
- ❑ на действующем устройстве iPhone те же вычисления занимают 7 сек.

Программа была несложная: я сделал простой тест с двумя массивами по тысяче элементов в каждом. Программа проходит в цикле по обоим массивам, чтобы найти одно и то же число, и выводит слово «привет». В действительности вам может и не понадобится обработка тысячи элементов в массиве, или вы можете не использовать обработку массивов для поиска одинаковых чисел. Однако суть не в этом. Я выбрал именно эти действия, чтобы продемонстрировать, что действующее устройство iPhone намного медленнее эмулятора iPhone.

Отсюда следует вывод, к которому я буду еще не раз возвращаться в этой книге: приложение необходимо тестировать как в эмуляторе, так и на действующем устройстве. Так почему же не тестировать только на действующем устройстве? Потому, что эмулятор обладает следующими значительными преимуществами:

- ❑ тестирование в эмуляторе занимает меньше времени, что означает для разработчиков меньше временных потерь;
- ❑ эмулятор отлично подходит для выявления утечек памяти и проблем ее распределения.

Память и производительность

Память и производительность представляют собой разные вещи. Под памятью обычно понимается *оперативная память* (Random Access Memory – RAM) и, соответственно, сколько места вы использовали и сколько осталось свободным. Производительность обозначает, насколько быстро приложение выполняет определенную операцию.

Память может оказывать большое влияние на производительность. Когда устройство обладает большим объемом оперативной памяти, можно предварительно загрузить и кешировать в ней больше данных. Оперативная память обладает более высокой скоростью доступа, по сравнению с файловым и сетевым хранилищами. Предварительно загрузив и кешировав больше данных в оперативной памяти, во многих случаях можно значительно увеличить скорость программ. Например, для игрового приложения, требующего загрузки большого количества изображений, большой объем памяти является необходимым условием, потому что позволяет заранее загрузить картинки и отображать их по мере необходимости. Загрузка из оперативной памяти в 10 раз быстрее, чем загрузка из файловой системы.

Однако более эффективное использование памяти не всегда означает лучшую производительность. Некоторые приложения не требуют много памяти, поэтому, оптимизируя использование памяти, можно столкнуться с тем, что производительность уже не повышается. Обратная ситуация не лучше: приложение может занять всю память, чтобы добиться лучшей производительности, а затем ему памяти просто не хватит.

Таким образом, необходимо всегда тщательно оценивать как память, так и скорость выполнения, чтобы убедиться в наличии строгого баланса между использованием памяти и производительностью во время выполнения.

Инструменты

Инструменты делятся на следующие три основные категории:

- ❑ основные, помимо инструментов среды разработки Xcode;

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине
«Электронный универс»
e-Univers.ru