1 Основы

1.0. Введение

После выхода операционной системы iOS5 в программировании для iPhone, iPad и iPod touch многое стало другим. Вся среда времени исполнения и способы написания кода на языке Objective-C разительно изменились. Теперь в компиляторе LLVM появилась функция ARC (автоматический подсчет ссылок). С одной стороны, эта функция обеспечивает дополнительную гибкость при работе, с другой — среда времени исполнения становится более уязвимой. В этой главе мы коротко познакомимся с объектами и поговорим о том, как их можно использовать в современной среде времени исполнения языка Objective-C с применением автоматического подсчета ссылок.

Из названия Objective-C следует, что этот язык предназначен для управления объектами. Объекты — это, в сущности, контейнеры для всего того, чем вы управляете в программе, от чего-то совершенно простого, как точка в углу прямоугольника, до целых окон, содержащих всевозможные виджеты. В библиотеках Apple Cocoa к объектам относятся в том числе простые значения, например целые числа. *Объекты* определяются в соответствии с *классами*, поэтому два этих термина часто употребляются как синонимичные. Тем не менее класс — это не просто спецификация для определения объектов; каждый объект является *экземпляром* своего класса. Каждый класс — а следовательно, и объекты, создаваемые из этого класса, — это набор свойств, задач, методов, перечислений и многого другого. В объектно-ориентированном языке программирования классы могут наследовать свойства друг друга — принцип подобен тому, как человек может унаследовать определенные черты и стороны характера от своих родителей.

В Objective-C не допускается множественное наследование. Поэтому каждый класс является прямым потомком (как правило) другого класса.

Корневым классом большинства объектов в Objective-C является класс `NSObject`. Этот класс управляет возможностями среды исполнения, предлагаемыми в iOS. Поэтому любой класс, прямо или косвенно наследующий от `NSObject`, унаследует и эти возможности. Ниже в этой главе будет показано, что объекты, наследующие

от NSObject, могут пользоваться преимуществами характерной модели управления памятью, действующей в Objective-C.

1.1. Создание простого приложения для iOS в Xcode

Постановка задачи

Вы начали изучать программирование в системе iOS и хотите создать максимально простой проект iOS и приложение в среде Xcode.

Решение

Создадим новый проект iOS в Xcode, а затем запустим это приложение в эмуляторе iOS с помощью Command+Shift+R.

Обсуждение

Предполагается, что у вас есть компьютер Mac, на котором уже установлен инструментарий Xcode. Теперь вы хотите создать проект для iOS и запустить его в эмуляторе iOS. Этот процесс совершенно незамысловат.

1. Откройте Xcode, если еще не сделали этого.
2. Выберите в меню пункт File (Файл), в нем выберите New (Новый) и далее — New Project (Новый проект). На экране появится изображение примерно как на рис. 1.1.

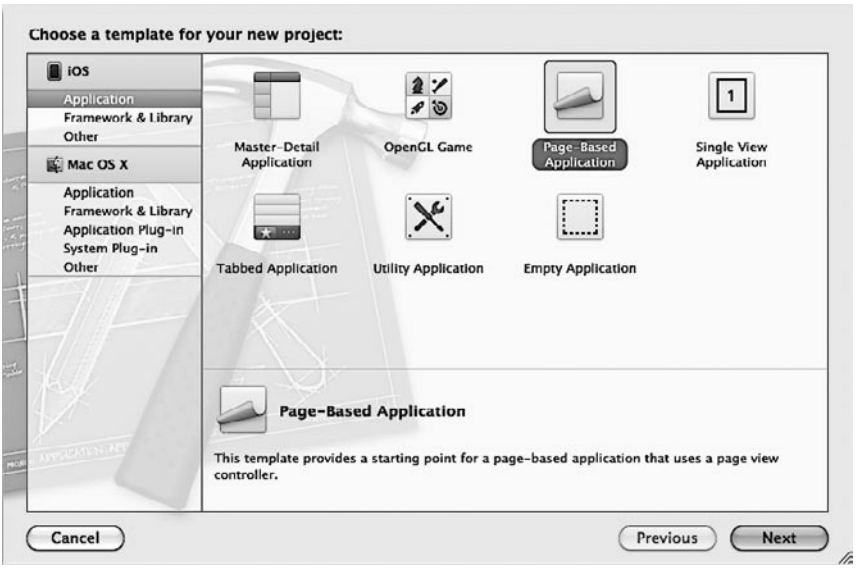

Рис. 1.1. Диалоговое окно нового проекта в Xcode

3. Слева в диалоговом окне создания нового проекта выберите подкатегорию Application (Приложение) в основной категории iOS. Затем справа щелкните на варианте Page-Based Application (Приложение с постраничной организацией) и нажмите кнопку Next (Далее).
4. Укажите название вашего продукта (Product Name) и идентификатор вашей компании (Company Identifier). Они уникально характеризуют ваш продукт, причем указывает, что этот продукт создан в вашей компании. Назовите ваш продукт Creating a Simple iOS App in Xcode (Создание простого приложения для iOS в Xcode). В качестве идентификатора компании обычно используется доменное имя, записанное в обратном порядке. Я работаю в компании Pixolity, поэтому в поле Company Identifier (Идентификатор компании) укажу `com.pixolity`, как показано на рис. 1.2. Остальные значения в окне оставьте такими, как на рис. 1.2, и нажмите кнопку Next (Далее).

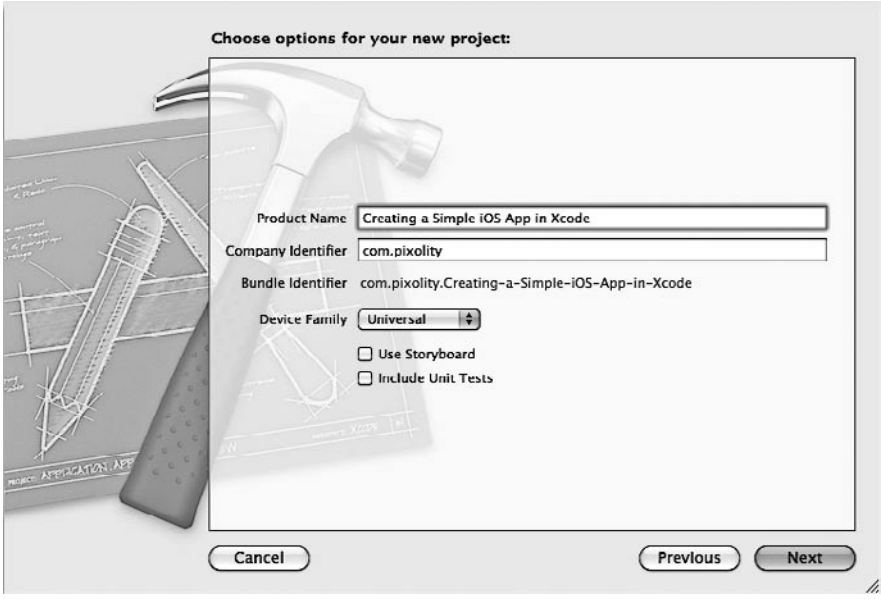

Рис. 1.2. Установка настроек нового проекта

5. Система предложит сохранить ваш новый проект на диске. Выберите желаемое местоположение проекта и нажмите кнопку Create (Создать) (рис. 1.3). Теперь Xcode создаст файлы вашего проекта и правильно их структурирует.
6. Теперь, перед запуском проекта, убедитесь, что к компьютеру не подключено ни одного устройства iPhone или iPad/iPod. Это необходимо, поскольку если к вашему Mac подключено такое устройство, то Xcode попытается запускать приложения именно на устройстве, а не на эмуляторе. А если вы не задали в подключенном устройстве конфигурацию, необходимую для разработки, то может включиться блокировка и запустить приложения вы не сможете.

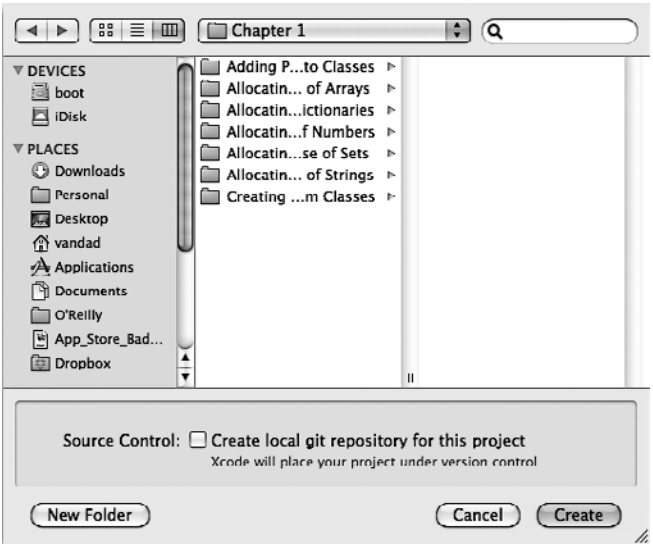

Рис. 1.3. Сохранение нового проекта на диске с помощью Xcode

7. В левом верхнем углу окна Xcode появится раскрывающееся меню. Убедитесь, что здесь выбран эмулятор iPhone или iPad. В данном примере у меня выбран эмулятор iPad (рис. 1.4).

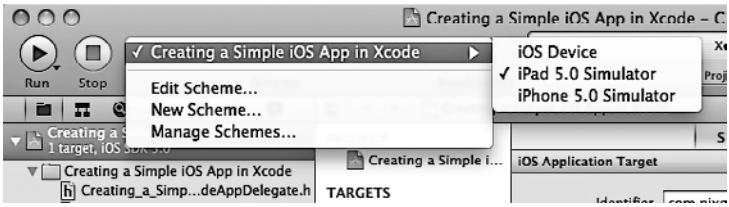

Рис. 1.4. Запуск приложения для iOS на эмуляторе iPad

8. Теперь, когда все готово, нажмите на клавиатуре Command+Shift+R либо просто перейдите в меню Product (Продукт) и выберите команду Run (Запустить) (рис. 1.5).

Ура! Вот и готово простое приложение, работающее в эмуляторе iOS. Как видите, существует несколько различных шаблонов для проектов iOS, предлагаемых на выбор. Вот список нескольких удобных шаблонов, которыми вы можете воспользоваться.

- *Приложение типа «Основной — детали» (Master-Detail)*. Такой шаблон проекта задает контроллер разделенного экрана. Контроллеры разделенного экрана рассматриваются в главе 2.
- *Приложение с постраничной организацией*. Такой шаблон позволяет создать пользовательский интерфейс iBooks, в котором пользователь может пролистывать

страницы, отрисовываемые приложением. Подробнее об этом интерфейсе мы поговорим в главе 2.

- *Пустое приложение*. Пустое приложение состоит из простейших компонентов, которые имеются в любом приложении iOS. Я часто пользуюсь этим шаблоном, чтобы мои приложения для iOS получались такими, какими я хочу их видеть, но при этом отпадает необходимость задавать в Xcode какую-либо предварительную конфигурацию.

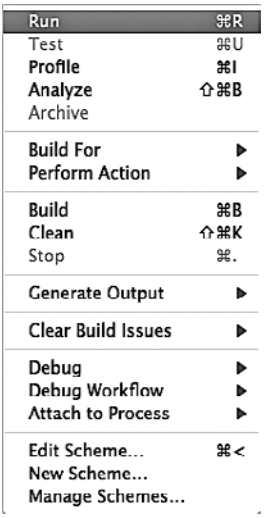

Рис. 1.5. Элемент меню Run (Запустить) в Xcode

1.2. Понятие конструктора интерфейса

Постановка задачи

Вы собираетесь приступить к проектированию пользовательского интерфейса для ваших приложений iOS, но хотите сэкономить время на написании кода.

Решение

Воспользуйтесь конструктором интерфейсов.

Обсуждение

Конструктор интерфейсов (Interface Builder, IB) интегрирован в Xcode и представляет собой инструмент для создания пользовательских интерфейсов приложений, которые пишутся для операционных систем Mac и iOS. Конструктор интерфейсов работает с файлами, имеющими расширение XIB, которые иногда называются NIB-

файлами, поскольку именно такое расширение подобные файлы имеют в продуктах Apple. В сущности, NIB-файл — это скомпилированная (двоичная) версия XIB-файла. В свою очередь, XIB — это XML-файл, управляемый конструктором интерфейсов. XIB-файлы пишутся в формате XML, чтобы их было удобнее использовать в системах контроля версий и с текстовыми инструментами.

Итак, начнем работу с конструктором интерфейсов. Для этого сначала создадим приложение для iOS, воспользовавшись шаблоном проекта iOS Single View Application (Приложение с единственным видом) в Xcode. Повторите шаги, описанные в разделе 1.1, но выберите не шаблон Page-Based Application (Приложение с постраничной организацией), как на рис. 1.1, а шаблон Single View Application (Приложение с единственным видом) и дойдите до последнего диалогового окна, где проект предлагается сохранить на диск. Я назвал проект Understanding Interface Builder (Понятие конструктора интерфейсов).

Убедитесь, что ваше приложение является универсальным (Universal), как показано на рис. 1.2.

После того как создан проект, первым делом нужно убедиться, что он будет работать на эмуляторе iPhone (рис. 1.6).

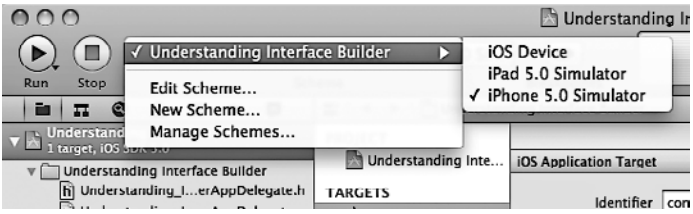

Рис. 1.6. Запуск приложения на эмуляторе iPhone

Теперь нажмите Command+Shift+R, чтобы запустить приложение. Вы увидите эмулятор iOS, в котором отображается пустое приложение (рис. 1.7).

Потом найдите в вашем проекте файл `Understanding_Interface_BuilderView-Controller_iPhone.xib` и щелкните на нем. Конструктор интерфейсов откроется в Xcode и отобразит тот пользовательский интерфейс, который вы начали создавать. Теперь, не закрывая конструктор интерфейсов, выберите из меню Xcode команду View (Вид), далее Utilities (Утилиты) и, наконец, Show Object Library (Отобразить библиотеку объектов) (рис. 1.8).

Теперь, если заглянете в библиотеку объектов, то увидите, что на выбор предлагается масса элементов, которые вы можете включить в свой интерфейс. В их числе — кнопки, переключатели типа «Вкл./выкл.», индикаторы протекания процессов, табличные виды и т. д. Пока просто перетащите на ваш интерфейс кнопку. Это совсем просто (рис. 1.9).

Прямо после этого выберите в меню Xcode команду File (Файл), а далее Save (Сохранить), чтобы убедиться, что ваш файл `Understanding_Interface_BuilderView-Controller_iPhone.xib` сохранен. Далее переходим к следующему шагу и запускаем наше приложение в эмуляторе iOS (рис. 1.10).