004.738.5

К ВОПРОСУ ОРГАНИЗАЦИЯ ХРАНЕНИЯ ДАННЫХ В МОБИЛЬНОМ ПРИЛОЖЕНИИ

В.А. АТРОЩЕНКО, М.В. РУДЕНКО, Р.А. ДЬЯЧЕНКО, Р.Х. БАГДАСАРЯН

Кубанский государственный технологический университет,
350072, Российская Федерация, г. Краснодар, ул. Московская, 2;
электронная почта: rafael_555@mail.ru

Представлен краткий обзор работы хранилища данных для мобильных устройств на основе технологии Core Data. Рассмотрена схема организации хранения объектов в БД и следующие принципы работы: среды управляемых объектов, управления объектами и контекстом, выборки данных в Core Data, работы постоянного координатора хранилища, управляемых объектов и управляемой объектной модели. В выводе приведены примеры целесообразного и нецелесообразного использований данной технологии при организации хранения данных для мобильных устройств.
Ключевые слова: хранилище данных, мобильные приложения, базы данных.

Организация хранения данных в мобильном приложении

В мобильной операционной системе iOS в качестве стандартного хранилища данных используется технология Core Data. Core Data не является реляционной базой данных или системой управления реляционными базами данных (СУБД).

Core Data это фреймворк для построения объектно-графового хранилища. Он позволяет создать реляционную модель «сущность – атрибут» для сериализации в XML, бинарное или SQLite хранилища.

Core Data обеспечивает инфраструктуру для управления изменениями, сохранения объектов и извлечения их из хранилища. Эта технология может использовать SQLite в качестве одного из видов постоянного хранилища. При этом технология Core Data позволяет использовать вместо файловой системы только хранилище в памяти мобильного приложения, без сохранения данных в файл [1].

Core Data позволяет создавать сложные приложения исключительно с использованием инструментов моделирования данных Xcode и Interface Builder без написания кода. Но при необходимости программистом могут быть внесены изменения в логику управления БД.

Core Data хорошо интегрируется с Cocoa Bindings и использует те же технологии, их совместное использование позволяет значительно сократить объем кода, но можно использовать Core Data без Cocoa Bindings.

Cocoa Bindings позволяют разработчику сосредоточиться на описании связей между объектами, вместо того, чтобы детально описывать поведение программы. Cocoa Bindings (связывание) представляет собой набор технологий, которые позволяют использовать в своих приложениях для полной реализации парадигмы Model-View-Controller, где Model инкапсулирует данные приложения, View отображает и редактирует данные, а Controller является посредником между ними. Cocoa Bindings сокращает код, зависимый от модели, вида-представления и контроллера, поддерживает несколько способов просмотра данных и автоматически синхронизирует виды-представления, когда модели меняются. Cocoa Bindings обеспечивает расширенные контроллеры и протоколы для модели и вида-представления, принимая и дополняя классы Foundation и Application Kit [2].

Схема работы Core Data представлена на рисунке. При разработке мобильного приложения с Core Data необходимо отказаться от работы с хранилищем данных напрямую. Особенностью такого подхода является возможность безболезненно сменить тип хранилища (например XML файл, на SQLite), не меняя написанный код.

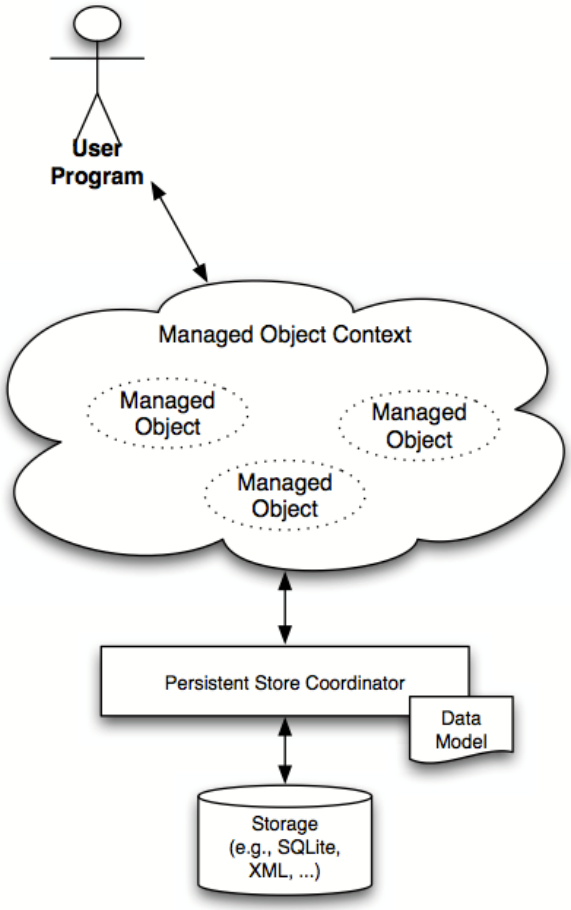

Схема работы Core Data

Объекты, которые находятся под управлением Core Data, должны наследовать методы и свойства класса NSManagedObject.

Среда, в которой находятся объекты, называется Managed Object Context (среда управляемых объектов). Она позволяет следить не только за тем, в каком состоянии находится объект, с которым происходит работа, но и за состояниями связанных объектов (объектов, которые зависимы от данного и от которых зависим он сам).

Экземпляр класса NSManagedObjectContext предоставляет ту самую среду для объектов, объект данного типа должен быть доступен в мобильном приложении всегда [3]. Обычно экземпляр класса NSManagedObjectContext является свойством делегата (структура данных, указывающая на методы) мобильного приложения. Без среды, без экземпляра класса NSManagedObjectContext невозможно работать с Core Data.

Управление объектами и контекстом в Core Data

Когда производится выборка объектов из постоянного хранилища, происходит перенос временных копий объектов в оперативную память, где они образуют объектный граф. После этого можно производить изменение этих объектов. Но эти изменения не будут записываться в постоянное хранилище.

Модели объектов, которые связываются в рамках Core Data, называются управляемыми объектами. Все управляемые объекты должны быть зарегистрированы в управляемом объекте контекста. Для добавления объектов к графу или удаления объектов из графа используют контекст. Контекст отслеживает все изменения как отдельных атрибутов объектов, так и связей между объектами. Отслеживание изменений контекста позволяет обеспечить поддержку отмены и повтора (undo, redo), а также гарантирует, что, если произошло изменение отношений между объектами, целостность графа объектов сохраняется.

Контекст позволяет выдать разрешение на сохранение произведенных изменений. Если данное разрешение получено, то все изменения будут записаны в постоянное хранилище.

Возможно использование более одного управляемого объекта контекста в приложении. Для каждого объекта в постоянном хранилище может быть не более одного соответствующего управляемого объекта, связанного с данным контекстом, т. е. данный объект в постоянном хранилище может быть изменен более чем в одном контексте одновременно. Каждый контекст имеет свой управляемый объект, соответствующий исходному объекту, и каждый управляемый объект может быть отредактирован независимо друг от друга.

Выборка данных в Core Data

Для получения данных с помощью управляемого объекта контекста необходимо создать запрос выборки. Запрос должен содержать предикат, определяющий условия, которым объекты должны соответствовать, и массив объектов дескрипторов, который определяет порядок выборки объектов.

После этого запрос необходимо отправить на выборку управляемому

объекту контекста, который возвращает объекты, соответствующие запросу, из источников данных, связанных с постоянным хранилищем [4]. Так как все управляемые объекты должны быть зарегистрированы в управляемом объекте контекста, то объекты, возвращаемые из выборки, будут автоматически зарегистрированы в контексте, используемом для извлечения. Для каждого объекта в постоянном хранилище может быть не более одного соответствующего управляемого объекта, связанного с данным контекстом. Если контекст уже содержит управляемый объект для объекта, возвращенного из выборки, то существующий управляемый объект возвращается в выборку результатов.

Когда происходит выборка множества объектов из постоянного хранилища, то загружаются только те объекты, которые соответствуют запросу. Если будет необходимо использование объекта с отношением, то необходимые данные загружаются автоматически. Если прекращается использование объекта, то он будет выгружен из оперативной памяти.

Постоянный координатор хранилища

Core Data является посредником между объектами приложения и внешними хранилищами данных и выполняет роль стека сохранения данных. В верхней части стека находится управляемый объект контекста, в нижней части стека — постоянное хранилище объекта. Между управляемым объектом контекста и постоянным хранилищем объекта есть постоянный координатор хранилища.

Координатор предназначен для представления программного слоя для управляемого контекстом объекта таким образом, что группа постоянных хранилищ выглядит как единое совокупное хранилище. Управляемый объект контекста может создать граф объектов на основе объединения всех хранилищ данных координатора. Координатор может быть связан только с одной управляемой объектной моделью.

Постоянные хранилища

Заданный объект постоянного хранилища связан с одним файлом или другим внешним хранилищем данных и отвечает за соответствие между данными, которые хранятся и соответствующих объектах в управляемом контексте объекта. Большинство других взаимодействий с Core Data выполняются через управляемый объект контекста.

В исходном коде мобильного приложения не должны быть прописаны пути к файлу постоянного хранилища данных. Core Data обеспечивает поддержку нескольких форматов файлов. Разработчик может выбрать необходимый формат в зависимости от потребностей приложения. Если на каком-то этапе необходимо выбрать другой формат файла, архитектура приложения остается неизменной [5]. Например, если первоначальная реализация может запрашивать только записи из локальной файловой системы, а затем, на более позднем этапе, будет добавлена поддержка нового типа удаленного постоянного хранилища, приложение сможет использовать новый тип без внесения изменений в код.

Управляемые объекты и управляемая объектная модель

Для управления графом объектов и поддержки постоянного хранилища объектов Core Data необходим файл с управляемой объектной моделью. Управляемая объектная модель представляет собой схему, которая описывает структуру хранилища. Создание управляемой объектной модели выполняется графически с использованием встроенных в Xcode инструментов для управления Core Data.

Управляемые объекты должны быть экземплярами NSManagedObject или подкласса NSManagedObject. NSManagedObject может представлять любую сущность. Он использует закрытое внутреннее хранилище, чтобы описать свои свойства и поведение, требуемое от управляемого объекта. Управляемый объект имеет ссылку на объект описания сущности, экземпляром которой он является [6]. Это относится к объекту описания метаданных, включая имя сущности, которую он представляет, и информацию о ее атрибутах и связях.

ВЫВОДЫ

В мобильном приложении для реализации хранения данных целесообразно использовать фреймворк Core Data в следующих случаях:

1. Использование Core Data позволяет уменьшить количество кода, написанного для поддержки модели слоя приложения, на 50–70%, измеряемое в строках кода;

2. Core Data использует информацию, содержащуюся в модели, и выполняет функции, как правило, не работающие на уровне приложений в коде;

3. Core Data обеспечивает достойный уровень безопасности и обработки ошибок;

4. Core Data предлагает лучшую масштабируемость при работе с памятью относительно любого конкурирующего решения.

Использование данной технологии позволит увеличить скорость работы и разработки мобильного приложения за исключением следующих случаев:

1. Если планируется использовать очень небольшой объем данных; в этом случае проще воспользоваться для хранения данных объектами коллекций – массивами или словарями и сохранять их в .plist файлы;

2. Если используется кросс-платформенная архитектура или требуется доступ к строго определенному формату файла с данными (хранилищу), например SQLite;

3. Если используются базы данных клиент-серверной архитектуры, например MySQL или PostgreSQL.

ЛИТЕРАТУРА

1. **Mark D., Bucanek J.** Learn C on the Mac For OS X and iOS. Apress, 2013. P. 256.

2. **Campbell M.** Objective-C Recipes A Problem-Solution Approach. Apress, 2012. P. 324.

3. **Mark D., Nutting J., LaMarche J., Olsson F.** Beginning iOS 6 Development Exploring the iOS SDK. Apress, 2012. P. 452.

4. **Mark D., Horovitz A., Kim K., LaMarche J.** More iOS 6 Development Further Explorations of the iOS SDK. Apress, 2012. P. 552.

5. **Grnlund H-E., Francis C., Grimes S.** iOS 6 Recipes A Problem-Solution Approach. Apress, 2013. 696 p.

6. **Keur C., Hillegass A., Conway J.** iOS Programming: The Big Nerd Ranch Guide. Apress, 2014. P. 456.

Поступила 24.04.14 г.

ON THE QUESTION OF THE ORGANIZATION OF DATA STORAGE IN A MOBILE APPLICATION

V.A. ATROSHCHENKO, M.V. RUDENKO, R.A. DYACHENKO, R.KH. BAGDASARYAN

Kuban State Technological University,
2, Moskovskaya st., Krasnodar, Russian Federation, 350072; e-mail: rafael_555@mail.ru

This article provides a brief overview of data storage for mobile devices based on technology Core Data. The scheme of the organization of the storage object to the database, and the following principles: protection of managed objects, facilities management and context data sample in Core Data, Resident Coordinator of storage managed objects and managed object model. In the output examples expedient and inappropriate use of this technology for the organization of data storage for mobile devices.
Key words: data storage, mobile applications, databases.