РАЗРАБОТКА РАСШИРЯЕМОЙ ИНФРАСТРУКТУРЫ ГРУПП ПРИЛОЖЕНИЙ ПОД ПЛАТФОРМУ IOS

К. А. Гонта, кафедра системного программирования СПбГУ, gontakseniya@gmail.com

Руководитель: Т. А. Брыксин, доцент кафедры системного программирования СПбГУ, timofey.bryksin@gmail.com

Аннотация

При iOS-разработке приложения, созданные одним разработчиком, можно объединять в группы – App Groups. В данной статье описывается архитектура и реализация расширяемой инфраструктуры групп iOS-приложений [1], которая обладает следующей функциональностью: возможность навигации между приложениями одной группы [4], возможность установки доступных пользователю приложений из этой группы, агрегация информации из всех приложений и отображение её в понятном пользователю виде.

Введение

На рынке существуют IT-компании, которые оказывают услуги по автоматизации бизнес-процессов (разрабатывают корпоративные порталы, системы электронного документооборота и т.д.). При необходимости они могут внедрять мобильные решения, в том числе для платформы iOS [2]. С ростом числа таких приложений на одном устройстве у пользователей появляется потребность в единой инфраструктуре для работы с ними, например, для удобной навигации между приложениями одного производителя.

Отдельно стоит рассмотреть вопрос установки новых приложений на устройства. Одним из популярных в России способов распространения приложений в пределах одной компании является поднятие Web-сервера, где пользователи авторизуются и скачивают доступные им приложения. Механизм установки возможно сделать для пользователей удобнее и прозрачнее. Одним из таких способов может стать приложение, которое сможет подключаться к серверу и запрашивать, какие ещё приложения доступны для скачивания этому пользователю, а затем давать возможность устанавливать их. Данное приложение помимо установки новых должно

также отображать уже существующие на устройстве приложения и некоторую дополнительную информацию о них, например, количество уведомлений.

Целью данной работы является создание единой инфраструктуры групп приложений от одного разработчика, которая бы позволяла навигироваться между этими приложениями, агрегировать из них информацию и устанавливать доступные для скачивания приложения.При решении поставленной задачи необходимо было учесть, что на устройстве могут со временем появляться новые приложения от этого разработчика, поэтому список доступных приложений должен быть динамически формируемый, а не статический. К тому же, у Apple достаточно жёсткая политика конфиденциальности, которая не позволяет получать из одного приложения сведения о другом, поэтому каждое приложение из группы должно “добровольно” предоставлять информацию о себе в некое общее хранилище.

Обзор

App Groups

Компания Apple заботится о конфиденциальности данных своих клиентов, поэтому в iOS нет возможности из одного приложения напрямую получать данные другого приложения. Однако, приложениям нередко требуется обмен информаций и для этого придуманы несколько механизмов. Приложения, созданные одним разработчиком, можно объединить в группу (App Group) и тогда они все они будут иметь доступ к общим настройкам, куда они смогут записывать информацию и откуда получать её при необходимости. Важно отметить, что приложения от различных разработчиков невозможно объединить в одну группу.

NSUserDefaults

Ресурс, разделяемый группой приложений, называется NSUserDefaults. Каждый экземпляр инициализируется со своим собственным идентификатором группы, например, "com.example.domain.MyApplication".

Экземпляр настроек хранится на устройстве до тех пор, пока установлено хотя бы одно приложение из группы.

URL Scheme

Для открытия одного приложения из другого существует механизм URL схем. Так, например, если на iPhone установлено приложение Instagram, то при введении “instagram://” в строке поиска Safari откроется соответствующее приложение. Для того, чтобы приложение можно было вызвать подобным образом, у него должна быть задана URL схема. К сожалению, это невозможно сделать из кода и приходиться прибегать к ручному заданию схемы через XCode, но для каждого приложения схему необходимо задать всего один раз.

Framework + Bundle

При разработке библиотеки главная цель — получить её в таком формате, который требует минимум усилий и знаний при интеграции. В iOS фреймворк [6] – это директория, которая содержит библиотеку и ресурсы (заголовочные файлы, изображения, READ.ME файлы). При этом, всё, что понадобится пользователю для интеграции — перетащить фреймворк к себе в проект.

К тому же есть потребность запускать код библиотеки на всех возможных для iOS архитектурах процессоров. На данный момент актуальны 3 архитектуры для устройств (armv7, armv7s, arm64) и 2 для

симуляторов (i386, x86_64). Для этого были собраны библиотеки для всех конфигураций оборудования, под которым будет работать приложение.

В iOS bundle – это особый вид папки, содержимое которой следует определённой структуре. Помимо скомпилированной библиотеки появилась необходимость распространять её вместе с ресурсами: файлы с описаниями фрагментов пользовательских интерфейсов , заголовочные файлы и т.д. Фреймворк может содержать в себе папку с ресурсами, но Xcode её проигнорирует.

Чтобы иметь возможность добавить ресурсы в проект, нужно распространять вместе с фреймворком либо отдельный bundle с ресурсами, либо символическую ссылку на ресурсы, расположенные внутри самого фреймворка.

Все ресурсы были размещены в bundle, а он помещён в фреймворк.

Архитектура

На Рис. 1 представлена архитектура решения для платформы iOS. Основой взаимодействия приложений между собой является библиотека, которая должна быть встроена в каждое приложение группы. Также по центру обозначено общее хранилище, к которому имеют доступ все приложения одной группы.

Приложение с правой стороны условно называется базовым, так как в нём демонстрируется основная функциональность, которую необходимо было реализовать в данной работе: возможность навигации между приложениями, агрегация данных из приложений одной группы, взаимодействие с веб-сервером и возможность установки приложений на устройство.

Легко заметить, что в базовое приложение встроена такая же библиотека, как во все приложения группы. Кроме того оно занимается взаимодействием с web-сервером приложений.

Среди элементов архитектуры присутствует виджет [5]. Виджетами называют расширения приложений, которые отображают информацию в Центре Уведомлений на экране “Сегодня” и, следовательно, призваны показывать ту информацию, которая важна в текущий момент. Когда пользователь открывает “Сегодня”, то он ожидает, что интересующая его информация будет мгновенно доступна. Виджет становится доступным после того, как пользователь установит приложение, содержащее виджет. Из Рис. 1 видно, что для базового приложения предполагается наличие

расширения для отображения его в Центре Уведомлений. Подробное устройство каждой из компонент рассмотрено ниже.

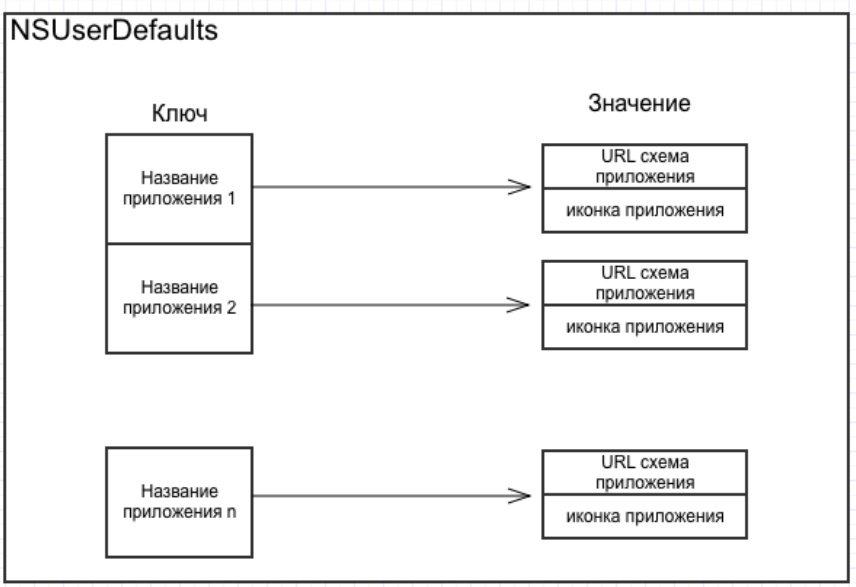
Рис. 1. Общая архитектура предлагаемого решения

Для начала рассмотрим хранилище данных. В нём содержатся сведения о всех приложениях текущего разработчика, установленных на устройстве, его структура представлена на Рис. 2. Для каждого приложения есть поля для записи его названия, иконки и URL схемы, по которой его можно вызывать. При необходимости можно увеличить количество полей информации о каждом приложении.

iOS-библиотека

Далее рассмотрим устройство и назначение iOS-библиотеки. В ней есть класс, отвечающий за загрузку данных. При установке приложения на устройство сущность “Загрузчик данных” сохраняет информацию о нём в NSUserDefaults. Также есть класс CollectionViewController, который в виде коллекции ячеек отображает все установленные на устройстве приложения и при нажатии на ячейку перенаправляет в соответствующее приложение.

Для получения информации об установленных приложениях



используется тот же “Загручик данных”, который загружает всю имеющуюся информацию из NSUserDefaults. При удалении приложения с устройства запись о нём не стирается из общего хранилища, поэтому необходимо проверять, что ни одно из приложений, перечисленных там, не было удалено. Для этого существует функция CanOpenUrl, которая

Рис. 2. Устройство общего хранилища данных

проверяет, установлено ли на устройство приложение с такой-то схемой.

Базовое iOS-приложение

В базовое приложение включена библиотека со всей функциональностью, то есть с отображением всех установленных на телефоне приложений и навигации между ними. Помимо этого добавлена агрегация однотипных данных, а именно, количество уведомлений (при помощи увеличения количество полей информации о каждом приложении в общем хранилище NSUserDefaults). Эта цифра отображается в стандартном для iOS виде: в красном кружке в верхнем правом углу иконки.

Кроме того, у базового приложения есть возможность предлагать

пользователю устанавливать отсутствующие приложения и скачивать их установочные файлы. В рамках работы был написан веб-сервер, на котором хранится информация о том, какому пользователю доступны для скачивания какие приложения. При помощи POST запроса на сервер в зашифрованном виде через безопасный канал отправляется логин и пароль пользователя, и в ответ приходит JSON файл со ссылками на все приложения, которые доступны пользователю. Если какое-то приложение доступно, но не установлено, то пользователю предлагается его установить. По нажатию пользователь переходит по ссылке, где лежит установочный файл, и дальше iOS сама занимается установкой.

Заключение

В ходе работы была реализована расширяемая инфраструктура групп iOS-приложений, которая удовлетворяет всем необходимым требованиям, а именно: возможность навигации между приложениями одной группы, возможность установки доступных пользователю приложений из этой группы, агрегация информации из всех приложений и отображение её в понятном пользователю виде.

Литература

1. Swift 2.2. Основы разработки приложений под iOS и OS X, Усов Василий
2. iOS. Разработка приложений для iPhone, iPad и iPod, Нахавандипур, Вандад
3. Swift: разработка приложений в среде Xcode для iPhone и iPad с использованием iOS SDK, Марк Дэйв
4. https://developer.apple.com/library/content/documentation/Carbon/Conceptual/LaunchServicesConcepts/LSCIntro/LSCIntro.html#//apple_ref/doc/uid/TP30000999
5. https://developer.apple.com/library/content/documentation/General/Conceptual/ExtensibilityPG/
6. https://www.raywenderlich.com/65964/create-a-framework-for-ios