

Занятие №3



# Организационная часть

---



- Отметиться!
- На прошлом занятии
  - База для написания приложений
- **Сегодня**
  - MVC и другие архитектурные паттерны
- Оставить отзыв (после занятия)

# Вопросы к аудитории

---



- Кто скачал из репозитория код примера с прошлого занятия?
- Кто смотрел пример, переписанный на Objective-C?
- Кто начал работу над своим проектом?
- У кого нет темы?
- Кто ещё не разбился на группы?
- Вопросы?

# Домашние задания



1. **ДЗ 0 (срок 08.10.2017 12:00) - максимум 5 баллов**
  - Разбиться на группы, придумать тему, создать репозиторий
2. **ДЗ 1 (срок 15.10.2017 12:00) - максимум 5 баллов**
  - Базовая функциональность
3. **ДЗ 2 (срок 05.11.2017 12:00) - максимум 10 баллов**
  - Использование списков элементов
4. **ДЗ 3 (срок 26.11.2017 12:00) - максимум 15 баллов**
  - Работа с сетью и локальной базой данных
5. **ДЗ 4 (срок 06.12.2017 12:00) - максимум 10 баллов**
  - Работа с дополнительными API
    - карта
    - камера / фотогалерея

# Текущая ситуация с проектами



- Заявлено в посте <https://park.mail.ru/blog/topic/view/9810/>
  - 16 проектов
  - из них в 8 проектах заявлен только 1 человек, лучше объединиться, можно обсудить
    - на портале
    - в ВК
    - в группе в Телеграме
- По факту <https://docs.google.com/spreadsheets/d/1PLrj37keBHd890paL53zwwtusYeVC9SWb61l71CaBa4/edit#gid=0>
  - 11 GitHub репозиторий
  - из них 1 не открывается, 1 пустой, 2 - хороший readme, 1 - стандартный проект
- Заявлено:
  - 32 участника (из 64)
- По факту
  - 22 GitHub аккаунта пользователей
- Сегодня последняя возможность очно согласовать тему/команду (без потери баллов)



- Повторение (мать учения)
- Про Objective-C
  - посмотрим прошлый пример, переписанный на нём
- MVC
  - Model
  - View
  - Controller
- Другие архитектурные паттерны (кратко, подробнее в лекции «Углублённые темы» 13 декабря)

# Objective-C



- Нет строгой типизации
  - тип `id`
  - вызвано тем, что это надстройка над C
    - структуры, указатели на функции
- Runtime
- Отправка сообщения объекту vs. Вызов метода

# Повторение, вопросы?



- Жизненный цикл приложения (состояния приложения)
- Жизненный цикл контроллера
- Контейнеры
  - UITabBarController
  - UINavigationController
- UIView + наследники (label, button...)
- UIScrollView + наследники (table view, collection view) - на следующей лекции
- Auto layout
- Создание UI с помощью
  - Interface Builder
  - кодом
- Segue vs. работа с Navigation Controller напрямую



## Model-View-Controller

- Разделение зон ответственности
- Даёт возможность переиспользования кода

- сами данные
- обеспечение доступа к данным
- логика для изменения данных
- Ещё называют реализацией бизнес-логики приложения
- Не зависит от UI
- Чаще всего потомок NSObject (в Objective-C) или базового класса базы данных

# View

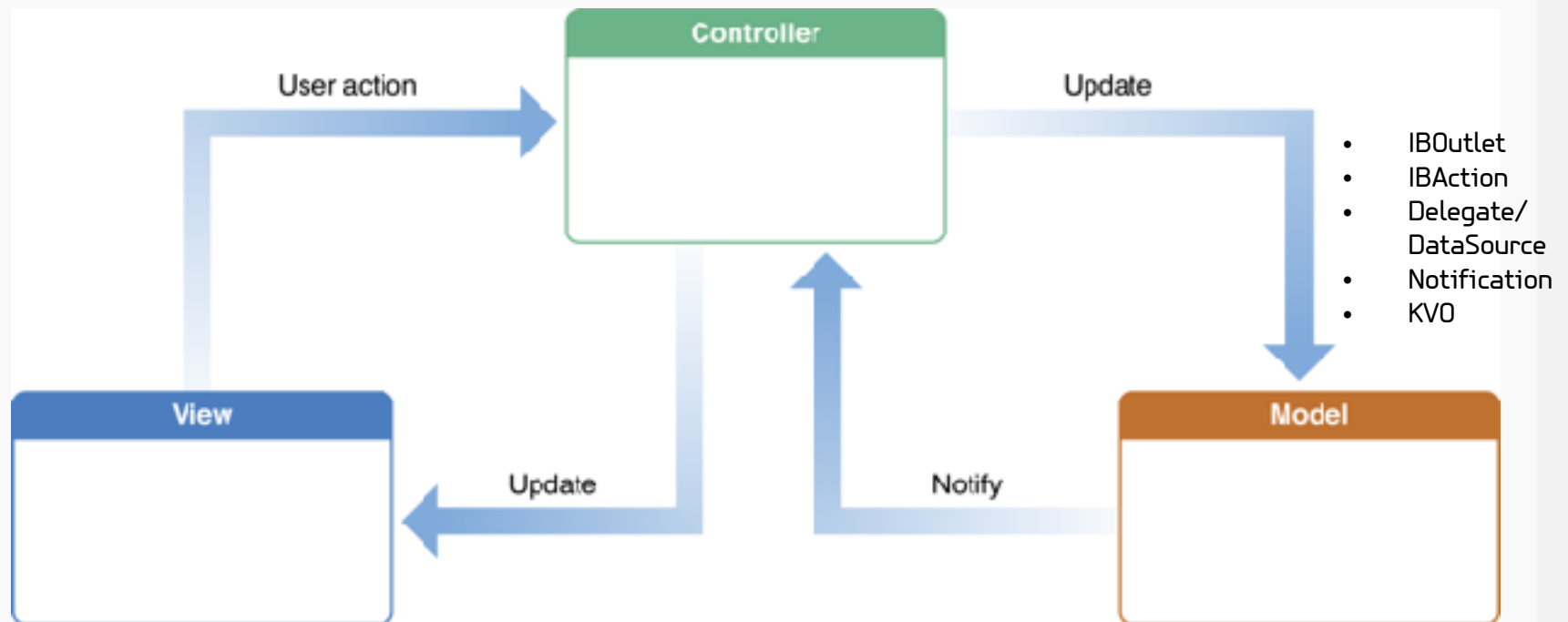


- Стандартные элементы (view, label и т.п.)
- Показывают данные
- Общаются с пользователем
- Ничего не знают о модели

Координирует взаимодействие между View и Model  
(посредник)

- Заполняет view данными
- Обработывает события от view
- Передаёт данные в модель
- Берёт данные из модели (модель уведомляет об изменении данных)

# Схема MVC



<https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html>

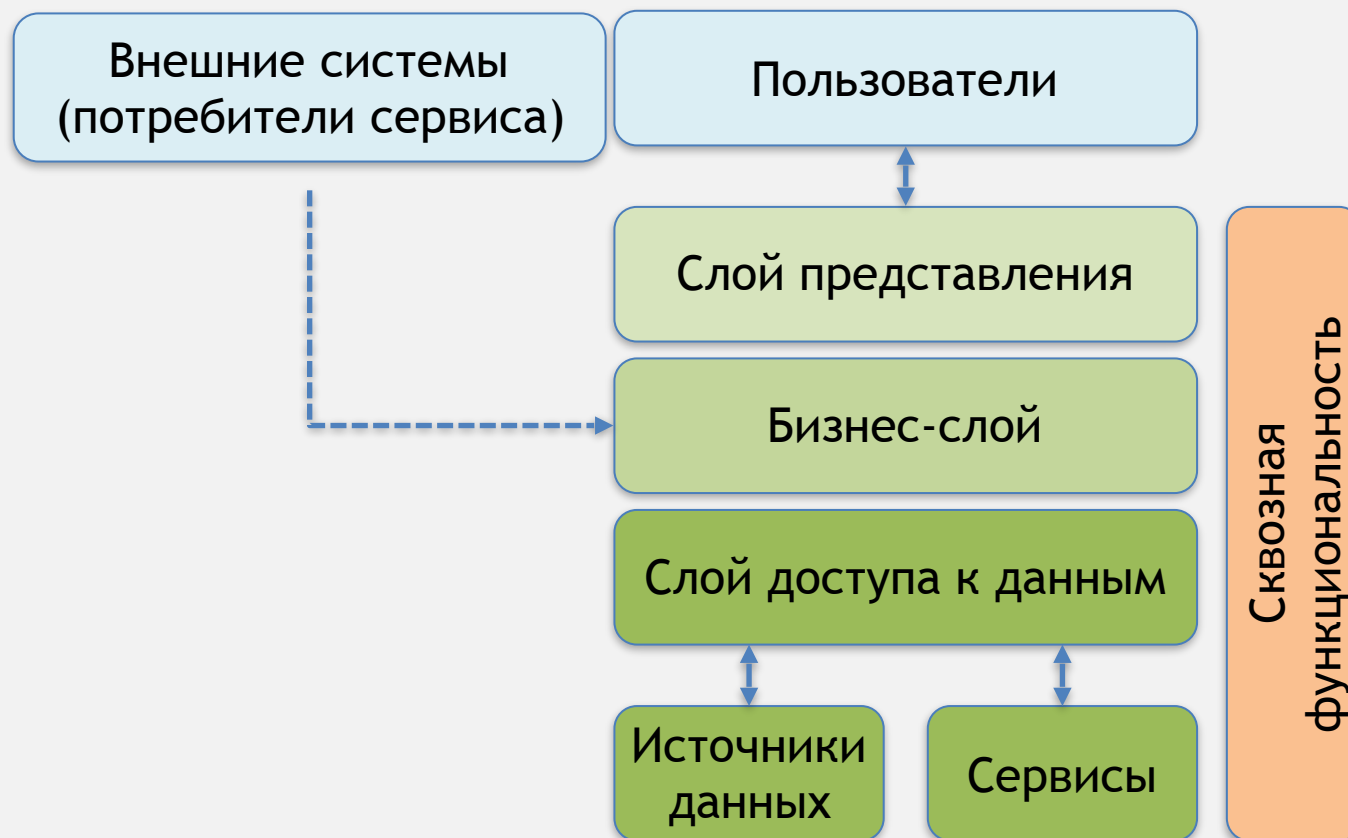
# Типовые составляющие приложения

---



1. Пользовательский интерфейс (UI)
2. Бизнес-логика
3. Сетевое взаимодействие
4. Хранение/кеширование данных

# Layered архитектура приложения



# Способы коммуникации между слоями

---



1. Делегат
2. Блоки
3. Notifications
4. KVO
5. IBOutlet / IBAction (между View и Controller)



# Основные архитектурные паттерны



1. MVC
2. MVP
  - похож на MVC, но
    - presenter не занимается layout
    - view controller относится к view
3. MVVM
  - похож на MVP, но связь не view и model, а view и view model (биндинг)
4. VIPER
  - ещё большее разделение обязанностей
    - view - UI
    - presenter - бизнес-логика, связанная с UI (всё инициализирует в модуле)
    - interactor - бизнес-логика, связанная с данными
    - router - переходы между модулями
    - entity - объекты данных

<https://habrahabr.ru/company/badoo/blog/281162/>



- Про «тяжёлые» view controller'ы  
<https://www.objc.io/issues/1-view-controllers/>
- Про структуру проекта в Xcode:  
<https://habrahabr.ru/post/261907/>
- «Архитектурный дизайн мобильных приложений»  
часть 1: <https://habrahabr.ru/company/redmadrobot/blog/246551/>  
часть 2: <https://habrahabr.ru/company/redmadrobot/blog/251337/>
- Про архитектурные паттерны:  
<https://habrahabr.ru/company/badoo/blog/281162/>