



# Вводное занятие

Лекция №1



Артур Сардарян, Павел Тихонов, Павел Носов

# Артур Сардарян



- Выпускник МГТУ им. Баумана (БМТ1)
- Окончил курс iOS в Технопарке
- Тимлид одной из iOS команд Юлы
- Опыт разработки 2D-игр в SpriteKit (just a little bit)
- Бизнес-приложения на аутсорсе
- Писал на Python чат-ботов, бэкенд, делал админку

Telegram: @sard\_art

# Организационная часть



- Отметиться — важно
- О чем пойдет речь в сегодняшнем занятии
  - О себе
  - Цели курса
  - Структура курса
  - Что нужно выполнить за семестр
  - Требования к проекту
  - Примеры проектов
  - Как будет построен процесс
  - iOS
  - Git
  - Coding (если успеем)
- Оставить отзыв (после занятия)

# Что нужно выполнить в течение курса



- 3 теста (5, 5, 10 баллов)
- 2 рубежных контроля (5, 10 баллов)
- Защита проекта (65 баллов)

# Баллы и оценки



Количество баллов	Итоговая оценка
0-49	Неудовлетворительно
50-69	Удовлетворительно
70-80	Хорошо
81-100	Отлично

# Часто задаваемые вопросы



- Что делать, если у меня в это время лекции/семинары/лабы/тренировки/надо забрать ребенка из садика/ и тд?
- Что делать, если у меня нет Mac?
- Будет ли SwiftUI?
- Будет ли Objective-C?
- Еще вопросы?

# Требования



- Необходим MacBook с установленным Xcode
- Виртуалка, хакинтош
- Базовое знание Git
- Знание ООП
- iPhone/iPad, иначе симулятор

# Проекты с прошлых выпусков



- Приложение для сервиса каршеринга
- Приложение для записи в салоны красоты
- Путеводитель по МГТУ
- Помощник по упражнениям для фитнеса
- Органайзер для студентов, с расписанием, домашним заданием и пр.
- Приложение с кулинарными рецептами
- Клиент для электронной очереди (eQueue)
- Клиент для GitHub (GitLit)
- Приложение для изучения английских слов (RemineMe)
- Приложение-гид по городам со списком достопримечательностей
- Лента Instagram в виде ленты VK
- Приложение для чтения книг
- Агрегатора распродаж одежды и обуви
- Приложение для выбора фильма
- Приложение для контроля расходов
- и многое другое

# Оценивание проекта



- качество проработки деталей,
- качество кода,
- стабильность работы,
- качество презентации проекта на защите

# Оценивание проекта



- Проект должен собираться и запускаться без ошибок
- На всякий случай пробовать заново клонировать из репозитория и удостовериться, что проект собирается и запускается
- Тестовый пользователь с тестовыми данными
- Функциональность
- Дизайн
- Навигация

# Входные данные



- Есть знания/опыт
  - опыт iOS разработки на Swift/Objective-C
  - опыт Android разработки
  - опыт в других языках C/C++/Python/Java/JS/PHP
  - опыт в Pascal
- Мало или нет опыта программирования вообще

# Ожидания от курса: цитаты

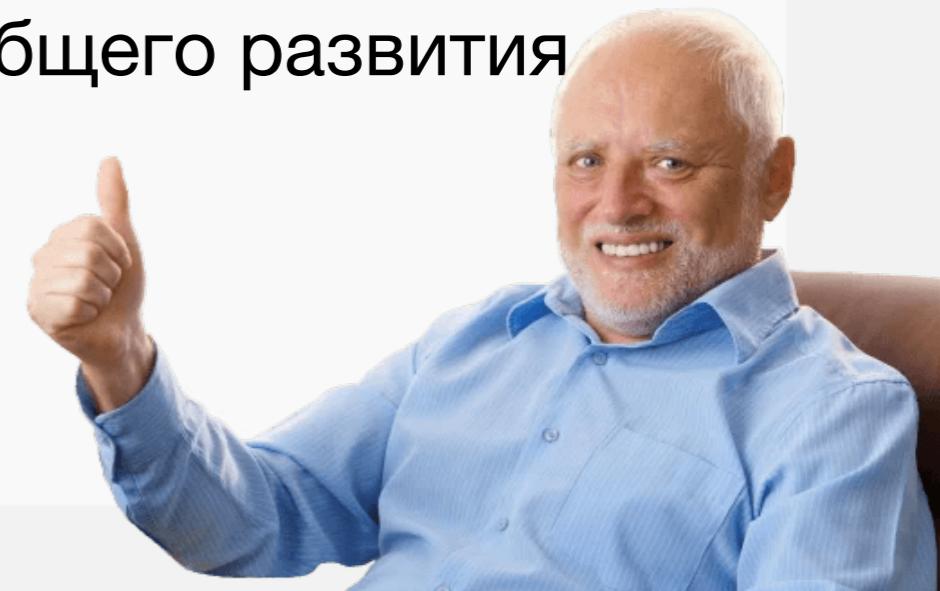


- Глядя на некоторые приложения, хочется многозначительно посмотреть в глаза тому, кто их создал и сделать самому в сто раз лучше, но я не умею. Этому и хочу научиться!
- Я хочу научиться разрабатывать приложения на swift, получить опыт работы в команде и общение с настоящими ios-разработчиками
- Пониманию архитектуры VIPER, разработке адаптивного интерфейса для разных layout'ов девайсов. SwiftUI.
- Хочу научиться разрабатывать приложения и игры для устройств ios! Узнать как работает вся система изнутри
- Разработать приложение, осуществляющее обмен данными с удаленным сервером, где они будут храниться и обрабатываться.

# Ожидания от курса: цитаты



- Разрабатывать приложения под ios для управления беспилотником
- Чему то неистово крутому
- Я хочу научится всему чему меня возможно обучить
- Честно признаюсь я допустил ошибку и хотел подать на курс "разработка приложений на android", но поскольку я опоздал сменить решение хотел бы обучиться на данном направлении и потому скорее как для общего развития
- хочу праграммировать на йзыке поскаль
- Зарабатывать деньги)



# Ожидания от курса: реалии



- командная работа
- работа над реальным проектом
- упорядочение обрывочных знаний
- углубление знаний
- Не будет:
  - разработка игр
  - SwiftUI

# Выбор проекта



- уже есть проект
- есть только идея
- нет даже идеи
  - посмотреть открытые API
  - присоединиться к существующей команде

# Команда до 4 человек



# Backend

---



- Firebase Realtime Database, Firestore, Storage
- Handmade (собственными руками)
- Public API

# Требования к проекту



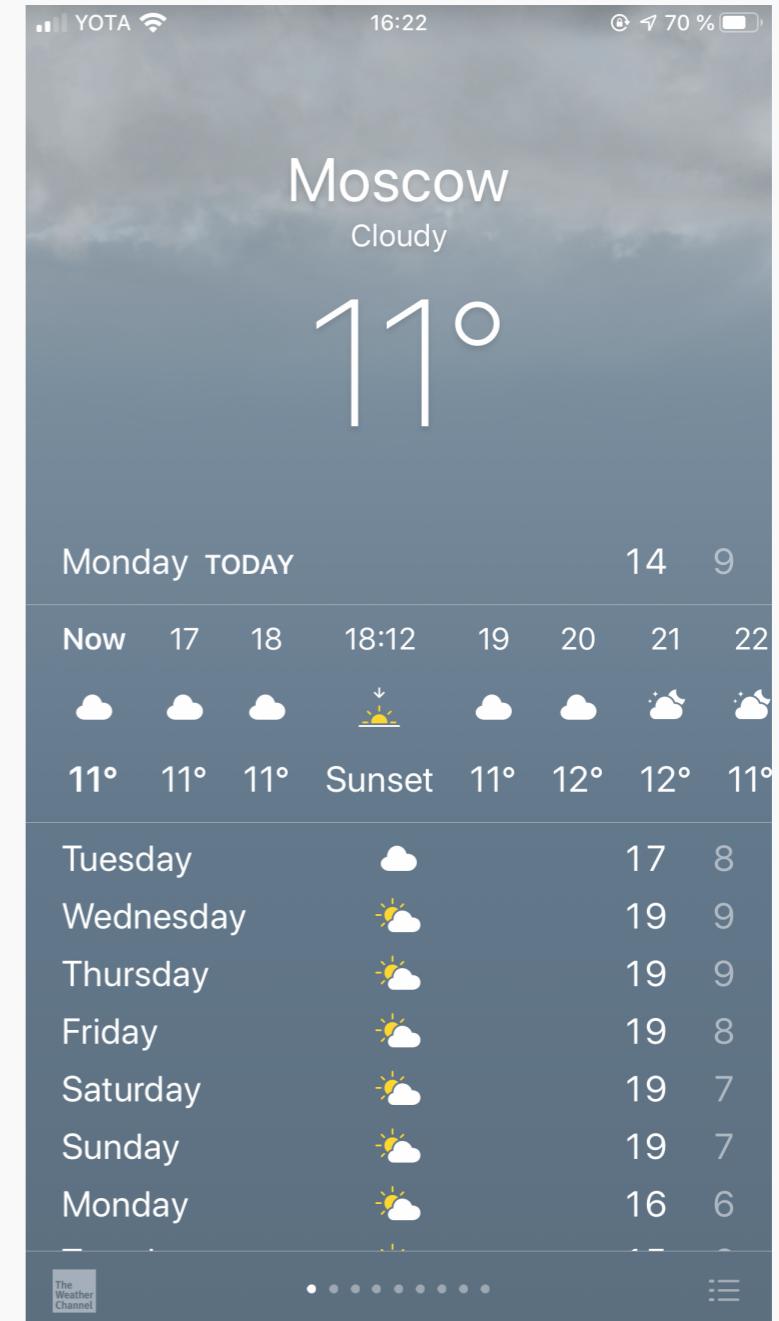
- отдельным экраном должна быть авторизация в сервис (там где требуется)
- основной экран должен содержать табличное представление данных (UITableView / UICollectionView)
- минимальное требование для представления отдельного элемента - картинка, название, описание
- 



# Требования к проекту



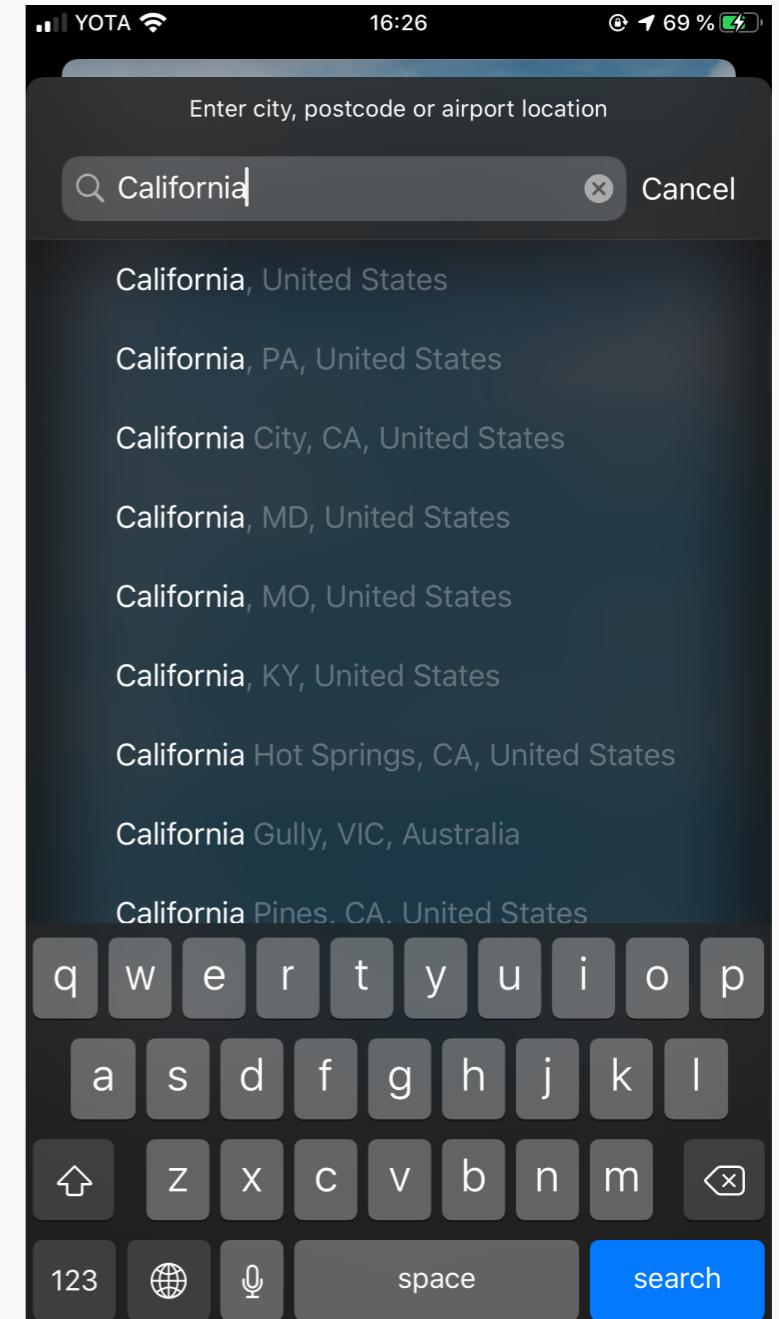
- дополнительный экран, открывающийся по нажатию на элемент в основном, должен содержать детальную информацию по выбранному элементу



# Требования к проекту



- Возможность добавлять данные в коллекцию

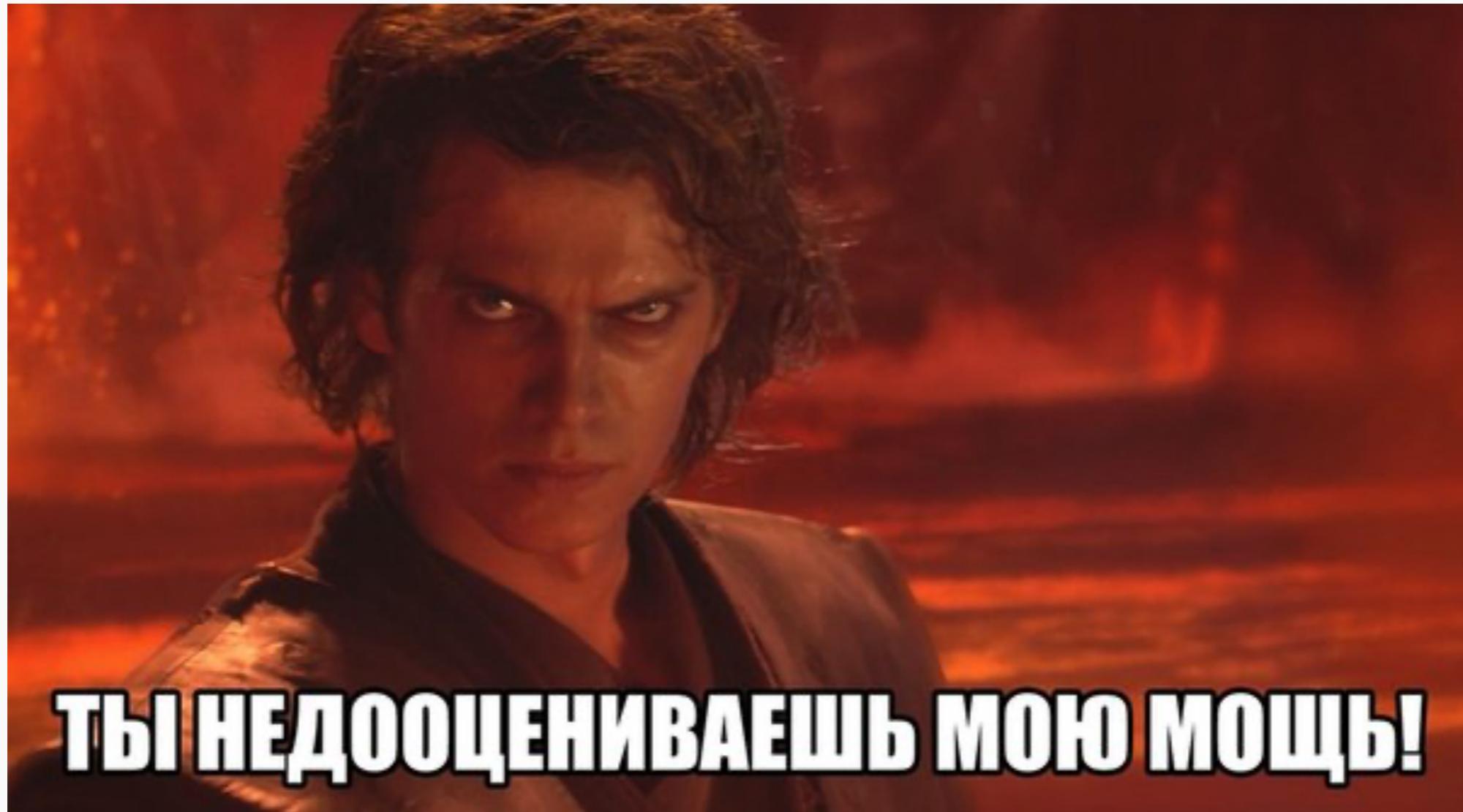


# Требования к проекту



- Обработка ошибок
- Сетевые запросы не блокируют основной поток
- Архитектура (желательно)
- Хранение данных в базе (крайне желательно)
- Оффлайн работа (будет круто)
- UI/Unit тесты (если очень хочется)

# Не переборщите с амбициями



Ваши идеи?

# Ссылки



- Репозиторий - <https://github.com/sardart/technopark-ios>
- Общий чат - [https://t.me/joinchat/EmNHgR1yAaTXqw\\_35WN5IQ](https://t.me/joinchat/EmNHgR1yAaTXqw_35WN5IQ)
- Блог курса - <https://park.mail.ru/blog/view/24/>



- Выпускник МГУПИ (МИРЭА)
- Больше 8 лет промышленной разработки приложений
- Тимлид продуктовой команды iOS в “Юла”
- Бизнес-приложения на аутсорсе
- Разрабатываю Android приложения just for fun

# iPhone – начало пути



29.06.2007 г. –  
первый показ iPhone  
и релиз iPhone OS.



# Конкуренты



Windows mobile



Palm OS



Symbian



BlackBerry



# Главный конкурент



# iOS 1.0



# Пользовательский интерфейс iOS



# iOS 2



# iOS 3 & iPhone 3GS



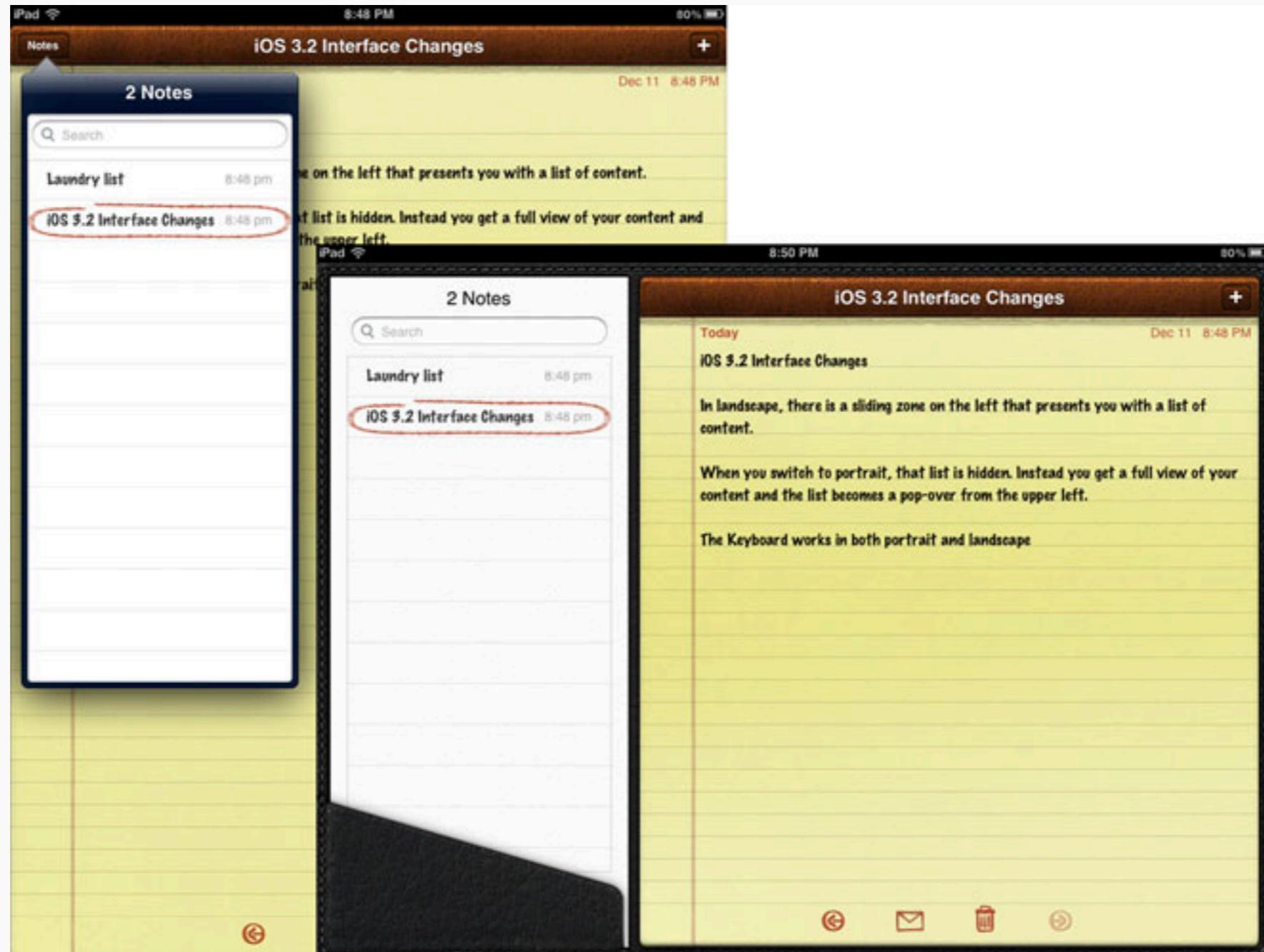
Июнь 2009 Apple выпустила iPhone 3GS & iOS 3



# iOS 3.2: iPad заходит в чат



# Нововведения iOS 3.2



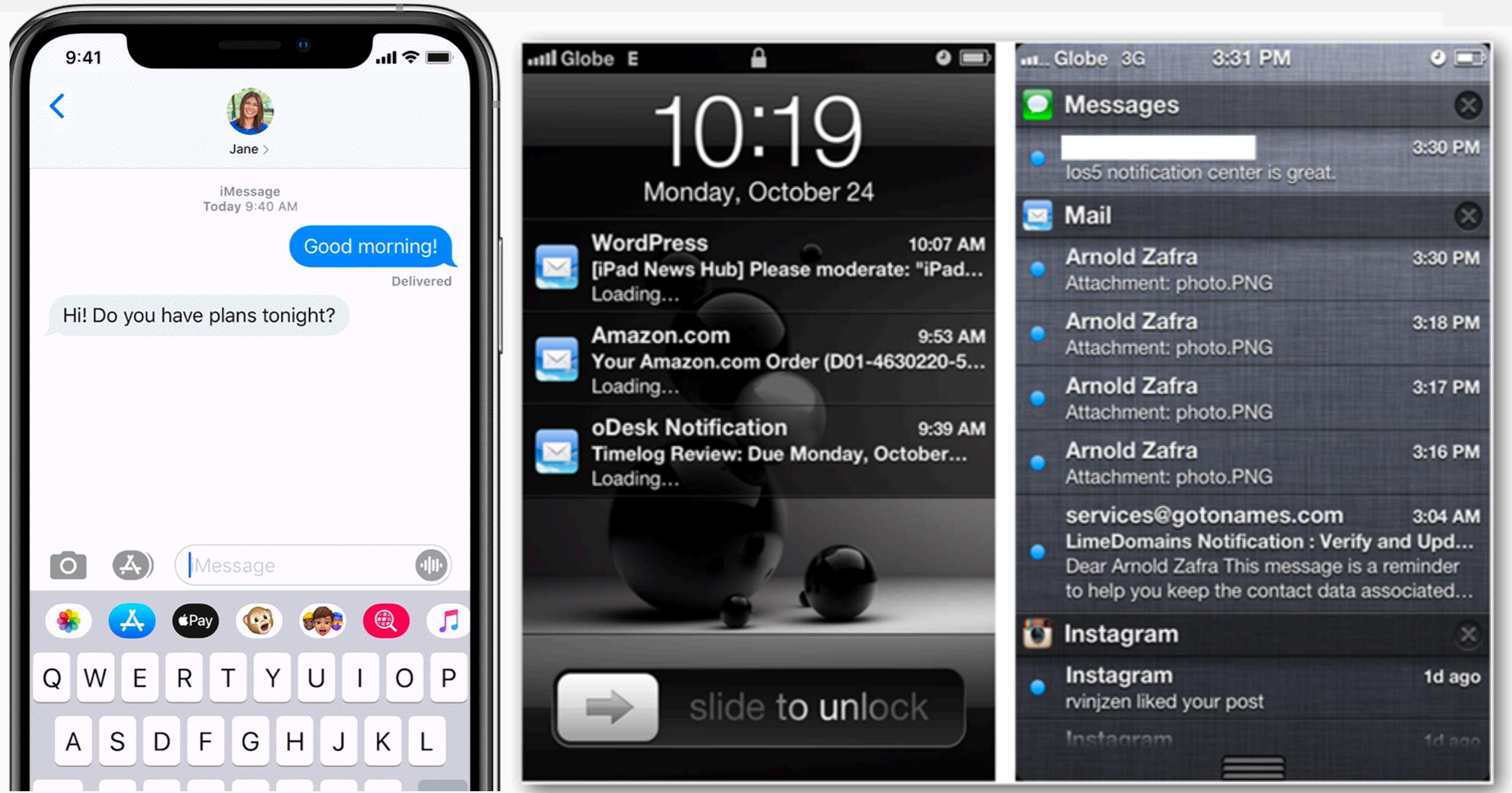
# iOS 4 - Многозадачность



# iOS 5 - Siri



# iOS 5



# iOS 6



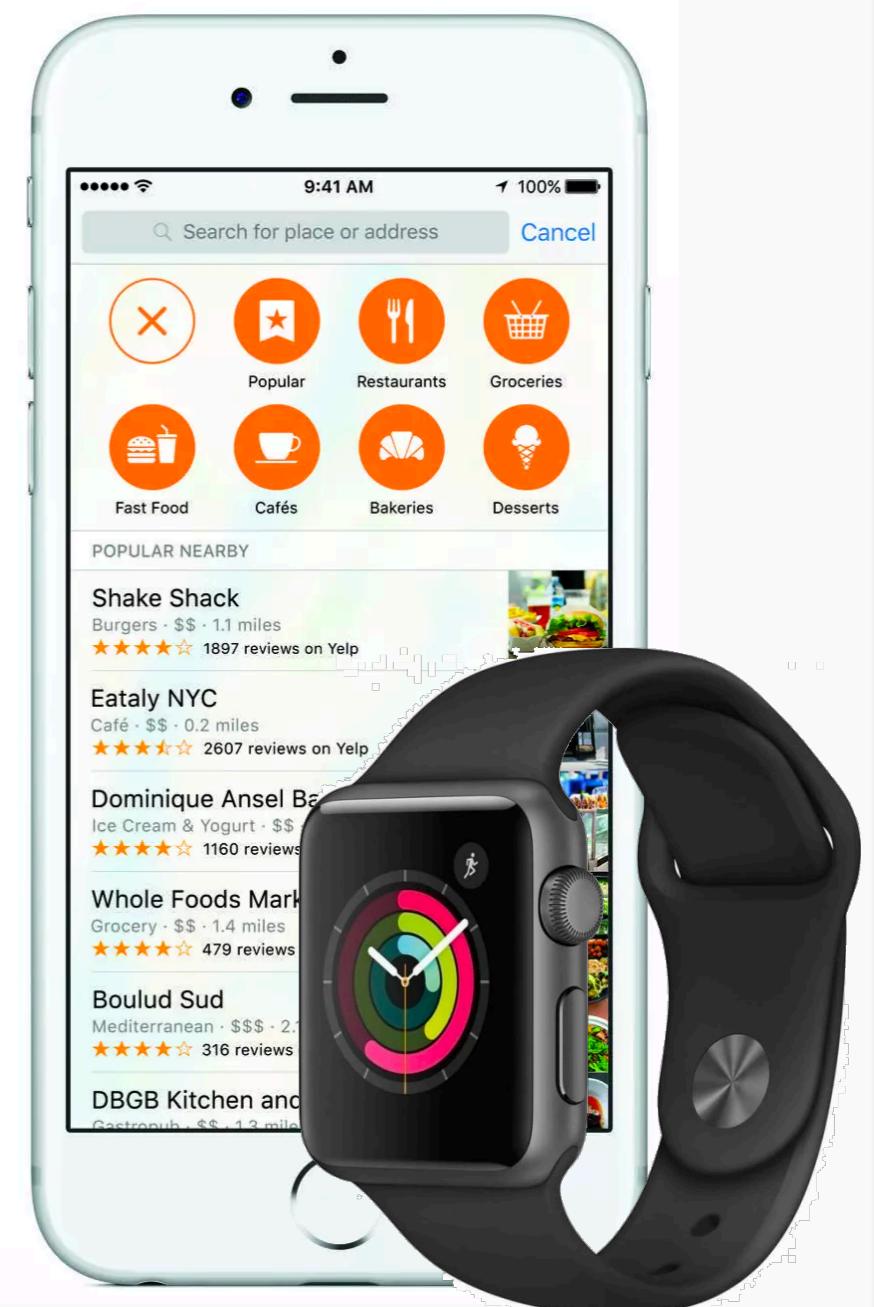
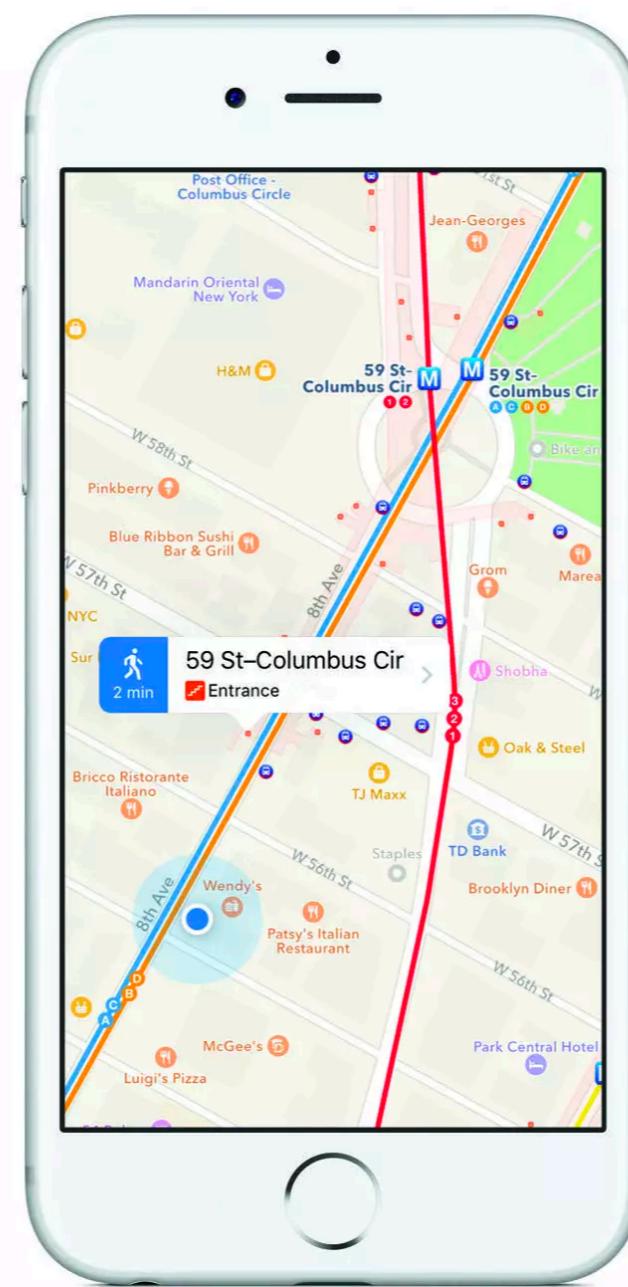
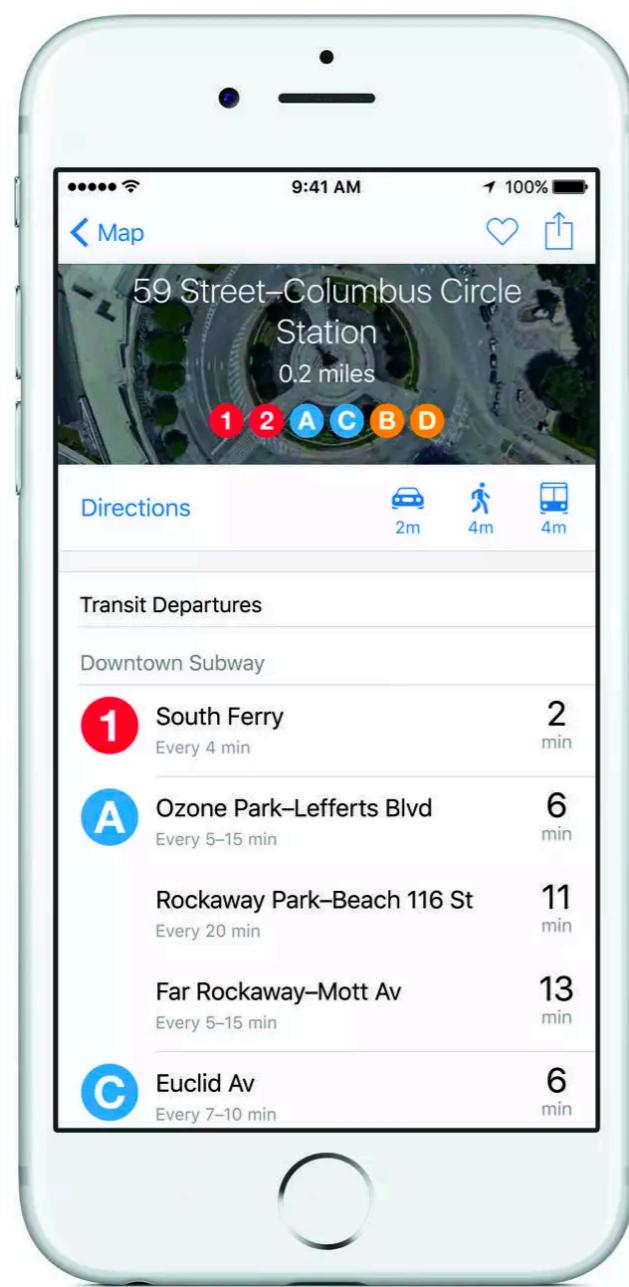
# iOS 7



# iOS 8



# iOS 9



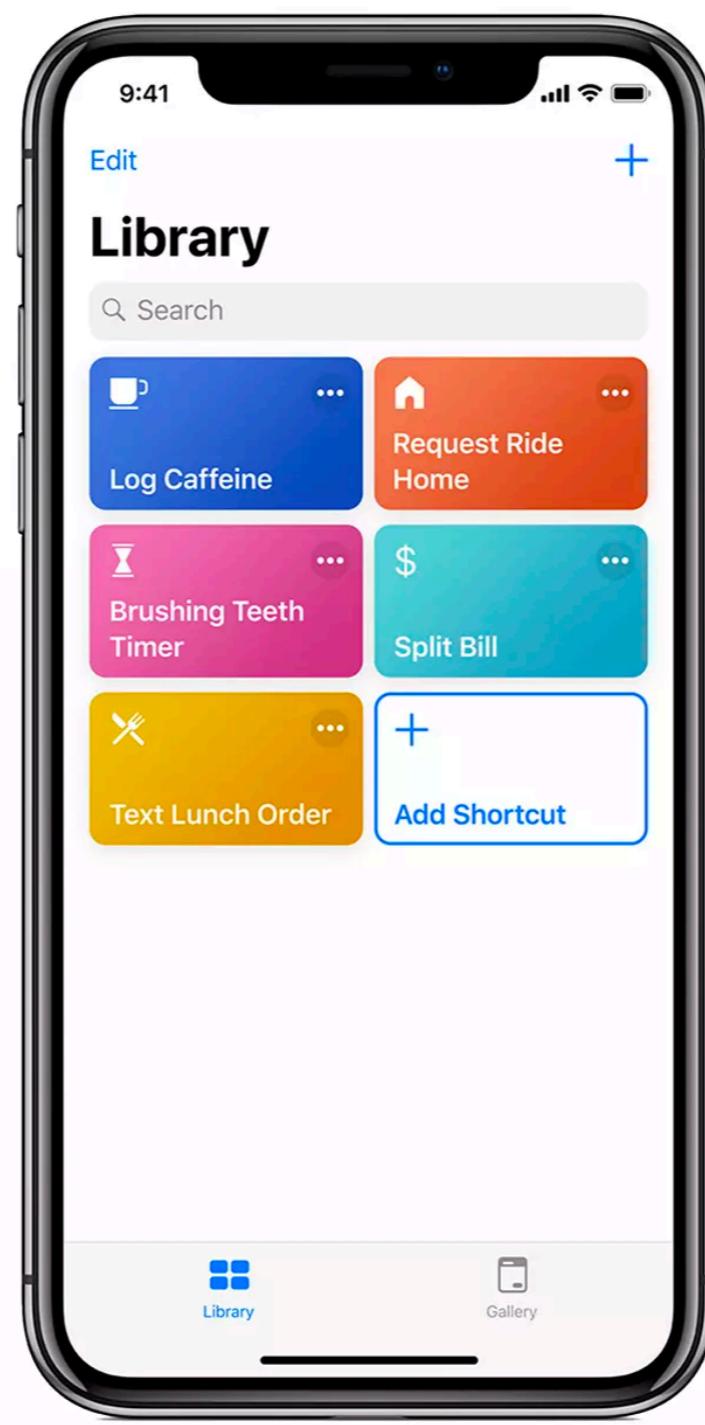
# iOS 10



# iOS 11



# iOS 12



# iOS 13



# iOS 14



# Постскриптум



Стив Джобс

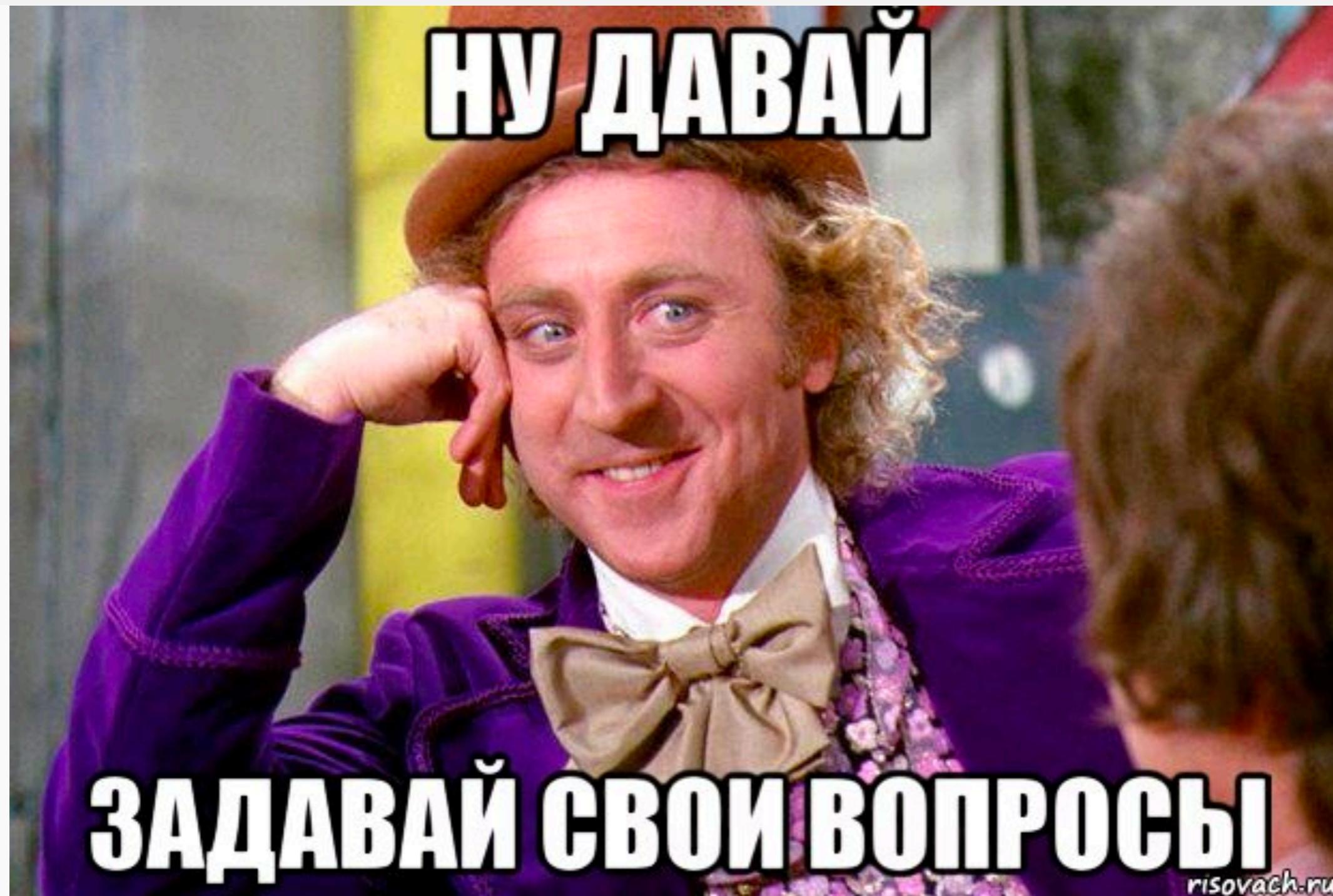


Джони Айв



Тим кук





# Павел Тихонов



- Выпускник МИЭМ (МИЭМ НИУ ВШЭ)
- Руководитель команды iOS в проекте Юла
- Опыт разработки iOS-приложений более 8 лет
- Бизнес-приложения на аутсорсе

Telegram: @pash3r

# Git – что это?



Кросплатформенная распределенная  
система контроля версий

# Зачем использовать git?



- Позволяет следить за историей изменений и изменять ее
- Для командной работы над проектом
- Код доступен в любой системе, где установлен git \*
- Переиспользование кодовой базы между проектами

# Начало работы с git



1. Убедиться, что git установлен

```
git --version
```

Гайд по установке: <https://git-scm.com/book/ru/v2/Введение-Установка-Git>

2. Добавить репозиторий к проекту

```
git init
```

# Staging



**Staging area** – место (логическое), где находятся файлы, добавленные для коммита

Добавить файл в stage:

```
git add demo.txt
```

Добавить несколько файлов в stage:

```
git add file1 file2 ... fileN
```

Добавить все изменения в stage:

```
git add .
```

# Статус файлов



```
git status
```

```
Pash3r ➤ ~/Desktop/gitExamples ➤ ⚡ master •+ ➤ git status
```

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: demo.txt

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: demo1.txt

```
Pash3r ➤ ~/Desktop/gitExamples ➤ ⚡ master •+ ➤
```

# Committing



**Committing** – процесс добавления файлов (изменений) в локальный репозиторий

Коммит с редактированием в vim:

```
git commit
```

Коммит с сообщением без редактирования в vim:

```
git commit -m "Сообщение коммита"
```

# История коммитов



Вывести историю коммитов:

```
git log
```

На экране появится список, состоящий из даты, автора, хеша и сообщения коммита

# Branch

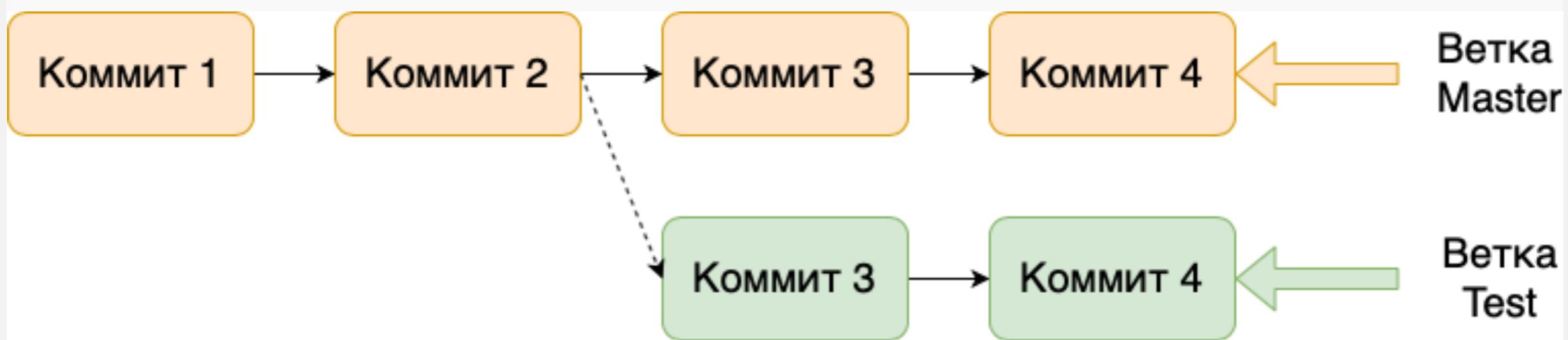


**Branch** – указатель на коммит в репозитории

**Зачем?**

Позволяет параллельно вести разработку

# Схема ветвления



Ветка  
Master

Ветка  
Test

# Branch



Вывод списка локальных веток:

```
git branch
```

Создать новую ветку локально:

```
git branch [branch_name]
```

Переключение между ветками:

```
git checkout [branch_name]
```

# Merge



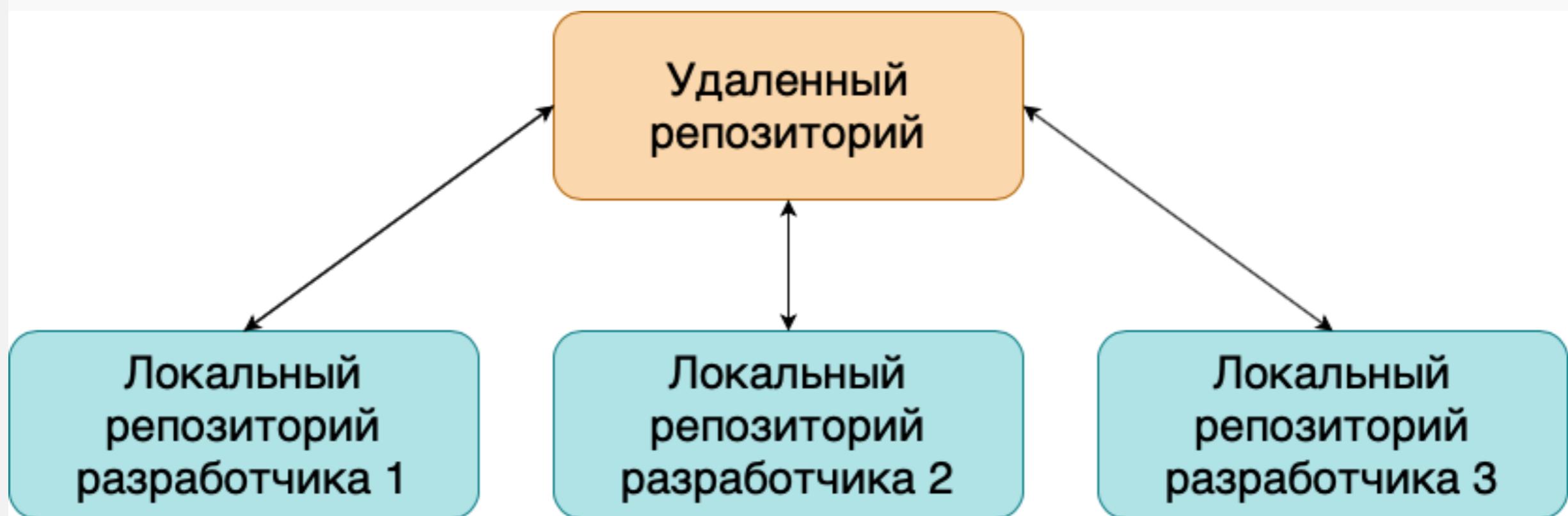
**Merge** – объединение текущей ветки с указанной

```
git merge [branch_name]
```

**Зачем?**

Для объединения кода из разных веток, чтобы  
сохранить целостность

# Удаленный репозиторий (remote)



# Создание удаленного репозитория



Удаленный репозиторий на GitHub



# Добавление удаленного репозитория



Добавление удаленного репозитория в git:

```
git remote add origin [repo_url]
```

Просмотр списка репозиториев:

```
git remote -v
```

# Push и Pull



**Push** – отправка коммитов из локального репозитория в удаленный:

```
git push origin [branch_name]
```

**Pull** – скачивание коммитов из удаленного репозитория:

```
git pull origin [branch_name]
```

# Скачивание удаленного репозитория



Скачать к себе уже существующий репозиторий:

```
git clone [repository_url]
```

# GUI-клиенты (популярные)



- GitHub Desktop
- Atlassian SourceTree
- GitKraken



# Материалы



1. Официальная книга по git на русском языке:  
<https://git-scm.com/book/ru/v2>
2. GitHub: <https://github.com>
3. GitHub Desktop: <https://desktop.github.com>
4. Atlassian SourceTree: <https://www.sourcetreeapp.com>
5. GitKraken: <https://www.gitkraken.com>