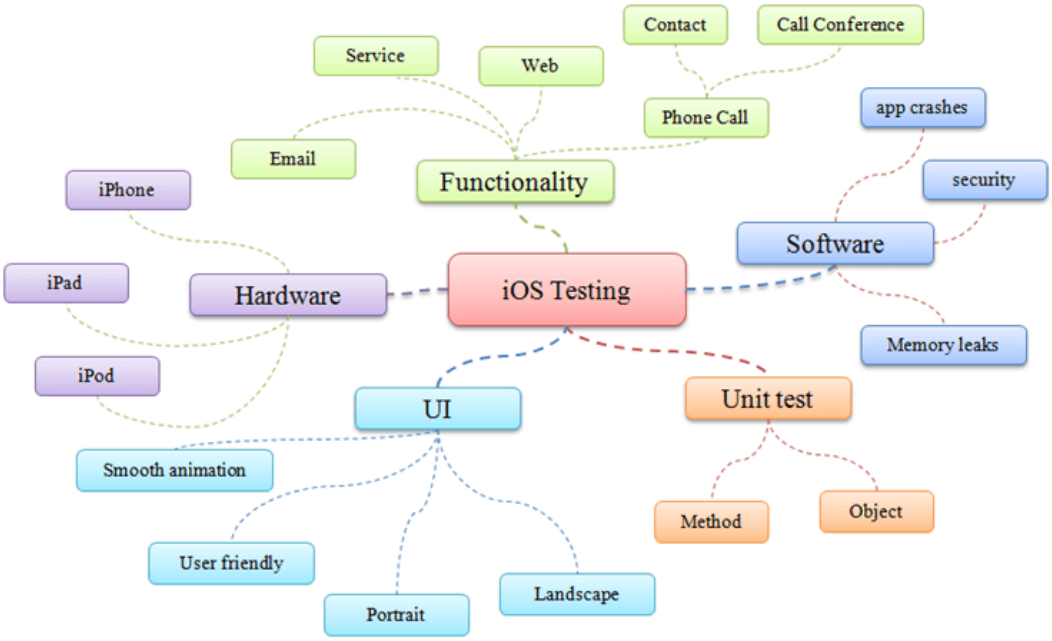
Особенности тестирования iOS-приложений

Почему тестирование iOS?

Впервые выпущенная 29 июня 2007 года, iOS — это название **платформы Apple** для мобильных приложений. В отличие от Android, Apple **не** лицензирует iOS для установки на оборудование сторонних производителей. Приложения iOS и iOS устанавливаются **только** на устройствах Apple. Ваше приложение должно быть совместимо с четырьмя типами устройств и версиями iOS:

iOS тестирование MindMap

На рисунке ниже - iOS Testing MindMap - показывает все элементы, которые тестер должен учитывать при проведении тестирования на iOS.



Контрольный список тестирования приложений iOS

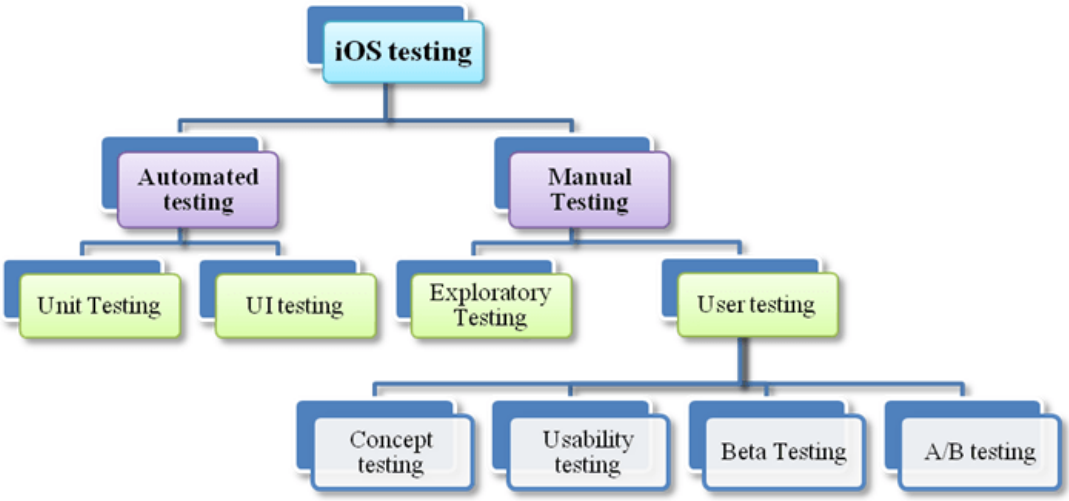
Этот контрольный список специально разработан для проверки характеристик мобильных приложений iOS. Очевидно, что он тестирует только общие характеристики приложения, а не его функциональность.

- Проверьте время установки приложения на устройство. Убедитесь, что приложение установлено в течение приемлемого времени.
- После того, как приложение установлено, проверьте, есть ли у приложения значок и имя приложения. Кроме того, убедитесь, что и значок, и имя говорят сами за себя, отражая основной смысл приложения.
- Запустите приложение и проверьте, отображается ли заставка.
- Проверьте время ожидания заставки и время, необходимое для загрузки начального экрана. Главный экран приложения должен загрузиться в течение приемлемого времени. Если загрузка главного экрана занимает больше времени, у пользователя появляется больше шансов выйти или даже удалить само приложение. Также проверьте, как содержимое загружается на главном экране.
- Основная функция приложения должна быть очевидна сразу. Это должно говорить само за себя.
- Проверьте, поддерживает ли приложение как альбомную, так и портретную ориентацию. Если это так, проверьте приложение в обоих направлениях. Пользовательский интерфейс приложения должен быть настроен соответствующим образом.

- Без подключения к интернету, запустите приложение. Убедитесь, что приложение ведет себя так, как задумано / желательно. Существует вероятность того, что приложение может произойти сбой при запуске или может просто отобразить пустой экран.
- Если приложение использует службы определения местоположения, проверьте, отображается ли предупреждение о разрешении местоположения или нет. Это предупреждение должно быть предложено пользователю только один раз.
- Если приложение отправляет push-уведомления, проверьте, отображается или нет уведомление о разрешении push-уведомлений. Это предупреждение также должно быть предложено пользователю только один раз.
- Запустите приложение, выйдите из него и перезапустите. Проверьте, работает ли приложение как задумано / желательно
- Закройте приложение, нажав кнопку «Домой» на устройстве, и снова откройте приложение. Проверьте, работает ли приложение как задумано / желательно.
- После установки проверьте, есть ли приложение в списке настроек iPhone.
- После того, как приложение запущено, проверьте, можно ли найти приложение в «App Store». Будет поддерживаемая версия ОС для приложения. Поэтому убедитесь, что приложение можно найти в «App Store» устройства с поддерживаемой версией ОС. Кроме того, приложение не должно быть указано в неподдерживаемой версии ОС устройства «App Store».
- Проверьте, не переходит ли приложение в спящий режим при работе в фоновом режиме, чтобы предотвратить разрядку аккумулятора.
- Если производительность приложения низкая или когда содержимое загружается, проверьте, имеется ли значок состояния выполнения («Загрузка…»), предпочтительно с определенным сообщением.
- Найдите приложение по названию в строке поиска устройства. Проверьте, есть ли приложение в списке
- Проверьте, не изменяется ли в приложении внешний вид кнопок, выполняющих стандартные действия (например, «Обновить», «Организовать», «Корзина», «Ответить», «Назад» и т. Д.).
- Проверьте, не используются ли стандартные кнопки для других функций, тогда как они обычно используются для

Стратегия тестирования iOS

На рисунке ниже представлены некоторые общие **типы стратегии тестирования** iOS.



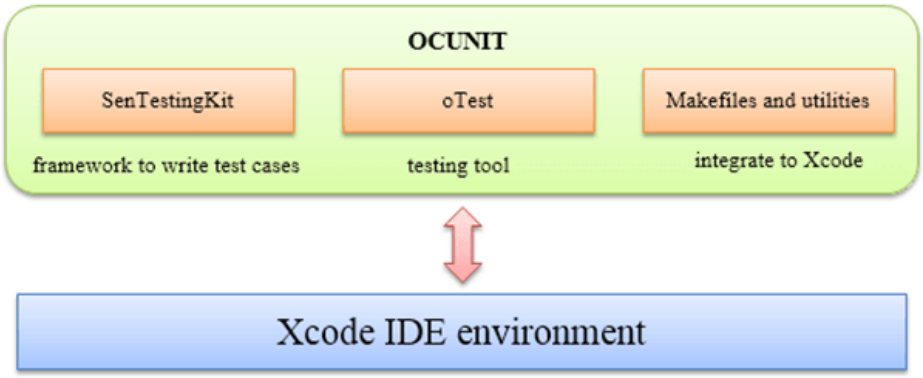
Автоматизированное тестирование

Автоматизированное тестирование — это большинство преимуществ тестирования iOS. Это позволяет быстро обнаруживать ошибки и проблемы с производительностью. Преимущества автоматического тестирования, как показано ниже:

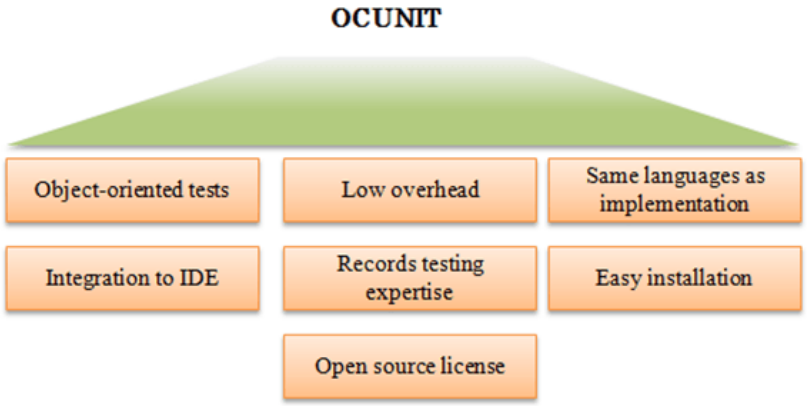
- Автоматическое тестирование может выполняться на нескольких устройствах, что экономит ваше время.
- Автоматическое тестирование может быть нацелено на SDK. Вы можете запустить тест на разных версиях SDK
- Автоматизированное тестирование повышает продуктивность тестирования и снижает затраты на разработку программного обеспечения.

- Существует множество платформ тестирования с открытым исходным кодом, поддерживающих автоматическое тестирование на iOS.

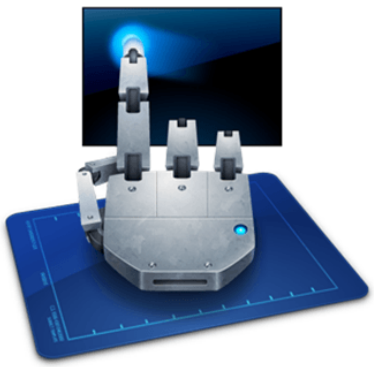
Модульное тестирование с OCUnit **(Unit testing with OCUnit)**
Когда был выпущен оригинальный iOS SDK, в нем отсутствовали возможности модульного тестирования. Поэтому Apple вернула решение для модульного тестирования OCUnit в iOS SDK версии 2.2.
OCUnit - это среда тестирования C-Objective в Mac OS. Самыми большими преимуществами платформы OCUnit являются тесная интеграция со средой разработки XCode, как показано ниже.)



Некоторые из преимуществ OCUnit показаны на рисунке ниже.



Тестирование пользовательского интерфейса с помощью UIAutomation (UI Testing with UIAutomation)

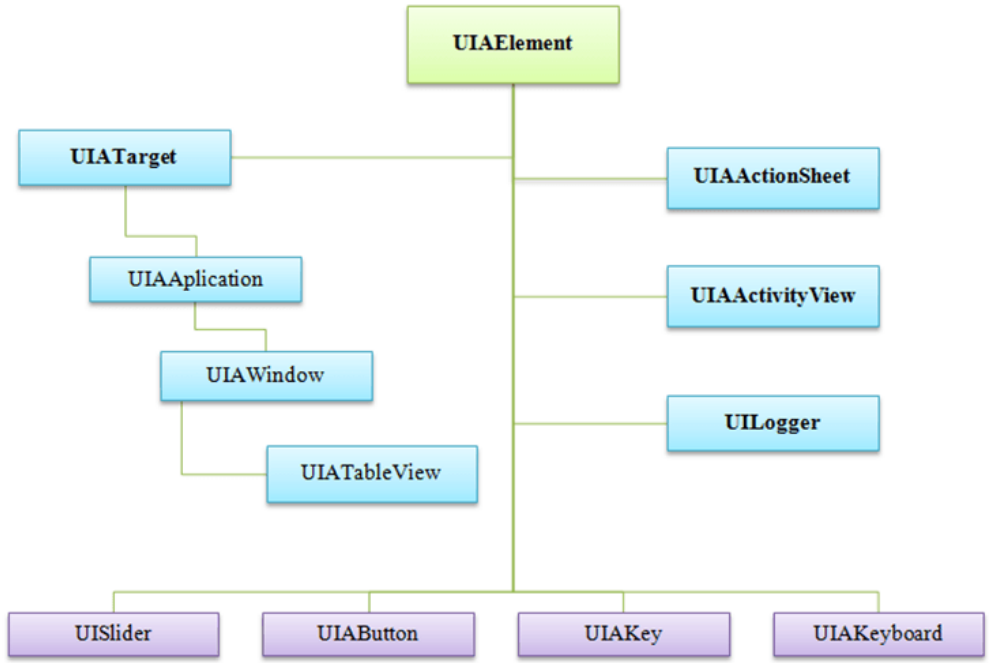


UI Automation - это библиотека JavaScript, предоставленная Apple Inc, которую можно использовать для выполнения автоматизированного теста на реальных устройствах и в iOS Simulator. Этот фреймворк добавлен в iOS SDK4.0. Используя UI Automation, вы можете автоматизировать тестирование приложения не только на симуляторе, но и на реальном устройстве. UIAutomation дает вам следующие преимущества: Снижение затрат на ручное тестирование Используйте меньше памяти для выполнения всех ваших тестов Упростите процедуру тестирования пользовательского интерфейса (просто нажмите одну или три кнопки и запустите все свои тестовые наборы) Инструмент UIAutomation работает из скриптов, написанных на JavaScript. Имитирует пользовательские события в целевом приложении iOS.

Минусы против плюсов UI Automation (*UIAutomation Cons vs. Pros)*

	Pros - ЗА	Cons - ПРОТИВ
1.	Хорошая поддержка жестов и вращения	Это не с открытым исходным кодом, меньше поддержки со стороны разработчика
2.	Может запускать тесты UIAutomation на устройстве, а не только на симуляторе.	Не очень хорошо интегрируется с другими инструментами
3.	Разработанный на JavaScript, это популярный язык программирования.	

Рисунок ниже представляет некоторые общие классы в рамках UIAutomation.



- Класс **UIAElement** является суперклассом для всех элементов пользовательского интерфейса в контексте автоматизации
- Класс **UIATarget** представляет высокоуровневые элементы пользовательского интерфейса тестируемой системы.
- Класс **UIALogger** предоставляет информацию о тестах и ошибках в функциях поиска
- Класс **UIAActivityView** обеспечивает доступ к представлениям активности в вашем приложении и управление ими.
- Класс **UIAActionSheet** позволяет получить доступ и управлять листами действий в вашем приложении.
- Пользователь **Действие события**
  - Класс UISlider
  - Класс UIAButton
  - Класс UIAKey
  - UIAКлавиатура

Другие автоматизированные рамки тестирования

- MonkeyTalk : инструмент для автоматического тестирования приложений iOS, Android, HTML5 и Adobe. Это интегрированная среда для управления и запуска тестовых пакетов.
- Фрэнк : рамки автоматизированного **приемочного тестирования** для iPhone и iPad
- KIF : **интегрированная** среда **тестирования** iOS . Это позволяет легко автоматизировать приложения для iOS, используя атрибуты доступности, которые ОС делает доступными для людей с нарушениями зрения.

Ручное тестирование

Исследовательское тестирование

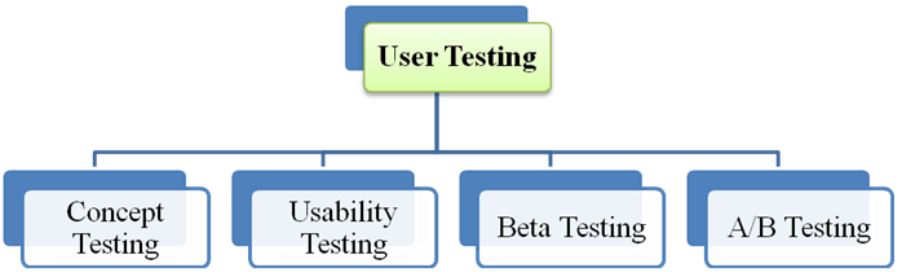
Это тестирование без официального плана тестирования. Исследовательское тестирование — это недорогой метод тестирования, но он может пропустить потенциальные ошибки в вашем приложении iOS.

Исследовательское тестирование против и за

	Pros	Cons
1.	Требуется меньше подготовки, рано выявлять серьезные ошибки.	Требует высокого мастерства тестера
2.	Не нужно, чтобы план тестирования ускорял обнаружение ошибок.	Тестовый охват низкий. Это не гарантирует, что все ваши требования проверены.
3.	Большинство ошибок обнаруживаются на ранних этапах своего рода предварительных испытаний	Отсутствие документации по тестированию

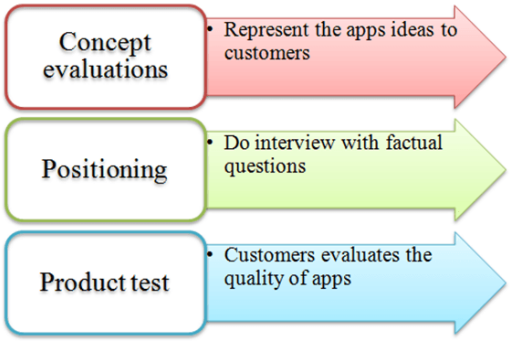
Тестирование пользователя

Пользовательское тестирование — это тип ручного тестирования на iOS. Целью этого тестирования является создание более качественных приложений, а не только приложений без **ошибок**. На рисунке ниже показаны четыре типа пользовательского тестирования.



Концептуальное тестирование

Оцените реакцию пользователя на идею приложения перед выпуском на рынок. Процедуры концептуального тестирования на iOS описаны ниже

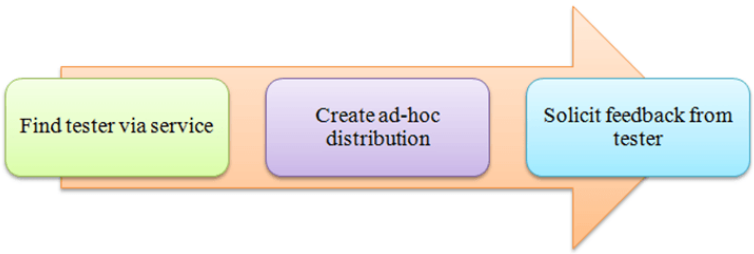


Юзабилити-тестирование

Юзабилити-тестирование — это тест на удобство использования вашего iOS-приложения. В тестировании iOS тест на юзабилити можно было **записать,** чтобы запомнить или поделиться с другими.
Существует несколько инструментов поддержки юзабилити-тестирования на iOS.
Magitest , простое iOS-тестирование юзабилити для сайтов и приложений.
Delight.io , этот инструмент может фиксировать реальное взаимодействие с пользователем в ваших приложениях для iOS.

Бета-тестирование

Бета-тестирование — это **интеграционное тестирование** с реальными данными для получения окончательной обратной связи с пользователями. Чтобы распространять свои приложения для бета-тестирования, вы должны выполнить следующие шаги.



— **Предварительное условие** : если вы проходите бета-тестирование окончательного кандидата на выпуск, обязательно проверьте приложение, прежде чем распространять его среди тестировщиков.
— **Найти тестера через сервис** : вы собираете идентификаторы устройств у тестеров и добавляете их в Member Center
— **Создание специального распределения** : специальное распространение позволяет тестировщику запускать ваше приложение на своем устройстве без необходимости Xcode. Этот шаг включает в себя 2 подэтапа

- Создать сертификаты распространения
- Создайте специальные профили обеспечения

— **Запрашивайте отзывы от тестера:** Тестер проводит тестирование и отправляет вам отчеты об ошибках. После того, как ваше приложение выпущено, вы можете получать отчеты из iTunes connect.

A / B тестирование

A / B-тестирование — один из самых мощных способов **оценки эффективности вашего приложения для iOS**. Он использует **рандомизированные эксперименты** с двумя устройствами, A и B.

A / B тестирование включает в себя три основных этапа

- **Настройка теста**: подготовлены 2 версии вашего приложения для iOS (A & B) и метрика теста
- **Тест**: Тестируйте 2 версии приложений iOS одновременно на устройствах.
- **Анализ**: измерьте и выберите лучшую версию для выпуска

Лучшие практики для A / B-тестирования

- Определите **цель** вашего теста. Любой тест бесполезен без цели.
- **Смотрите, как** конечные пользователи используют ваше приложение в первый раз
- Запускайте только **один** тест на обновление. Это экономит ваше время при проведении тестирования
- **Контролируйте** свой тест тщательно. Вы можете узнать опыт из своего теста, контролируя его.

Тестирование iOS Лучшая практика

Вот несколько советов, которые вы должны знать при организации тестирования вашего приложения iOS

1. Протестируйте приложение на **реальном устройстве,** чтобы узнать о производительности
2. **Улучшите** свои методы тестирования, потому что традиционных методов тестирования больше не достаточно, чтобы охватить все тесты по тестированию iOS
3. Использование **консольного журнала** для тестирования iOS-приложения. Эта функция iOS включает информацию о каждом приложении на устройстве.
4. **Документируйте** ошибки приложения, используя **встроенную экранную** команду. Это помогает разработчику понять, как возникают ошибки.
5. **Отчеты о сбоях** полезны при тестировании вашего приложения. Они могут обнаруживать сбои и регистрировать детали, чтобы вы могли легко исследовать ошибки.

МИФЫ о тестировании iOS

В этом разделе рассматриваются несколько популярных мифов и реалий тестирования iOS

1)Тестирование приложения на iOS и Android — это одно и то же.

iOS и **Android** — две платформы, разработанные Apple Inc и Google. Они совершенно разные. Тестовые среды Etc, тестовые среды, языки программирования.

2)Тестового приложения на iOS Simulator достаточно.

iOS Simulator недостаточно силен для тестирования приложения. Потому что iOS Simulator имеет некоторые ограничения:

- Аппаратные ограничения (камера, микрофонный вход, датчик)
- Пользовательский интерфейс вашего приложения может работать быстрее и плавнее, чем на устройстве
- Ограничения API
- Некоторые фреймворки не поддерживаются (Media Player, Store Kit, Message UI ..)

3)Каждый будет загружать мои приложения в магазине приложений, потому что он имеет много функций

Чем больше функций в вашем приложении, тем больше ошибок вы можете получить. Ни один пользователь не загрузит ваше приложение, если оно все еще имеет много дефектов.

https://coderlessons.com/tutorials/kachestvo-programmnogo-obespecheniia/mobilnoe-testirovanie/14-testirovanie-ios-prilozhenii