курс | development | 12 недель

iOS: разработка приложений с 0

r_d

iOS программа курса 20 занятий

1. Swift: начало
2. ООП: основы
3. Создание iOS-приложения в Xcode
4. Создание интерфейса iOS-приложения
5. Динамические интерфейсы, часть 1
6. Динамические интерфейсы, часть 2

iOS программа курса 20 занятий

7
Динамические интерфейсы, часть 3

8
Навигация в приложении, часть 1

9
Навигация в приложении, часть 2

10
Анимации в iOS

11
Работа с памятью в iOS

12
Многозадачность в iOS, часть 1

iOS программа курса 20 занятий

13

Многозадачность в iOS, часть 2

14

Дебаг iOS-приложения

15

Тестирование

16

Хранение данных в приложении

17

Работа с сетью в приложении

18

Сборка приложения

iOS программа курса 20 занятий

19

Современные архитектуры для iOS приложений

20

Защита курсовых проектов

Анимации в iOS

- Что такое CALayer
- Как iOS на самом деле отрисовывает контент на экране
- Способы работы с анимациями в iOS
- Demo 

r_d

Что такое CALayer

Объекты класса CALayer занимаются:

- менеджментом конкретно заданной прямоугольной области
- отрисовкой графического контента в этой области
- выполнением и управлением анимацией над этим контентом

Реализация Flat Navigation в приложении

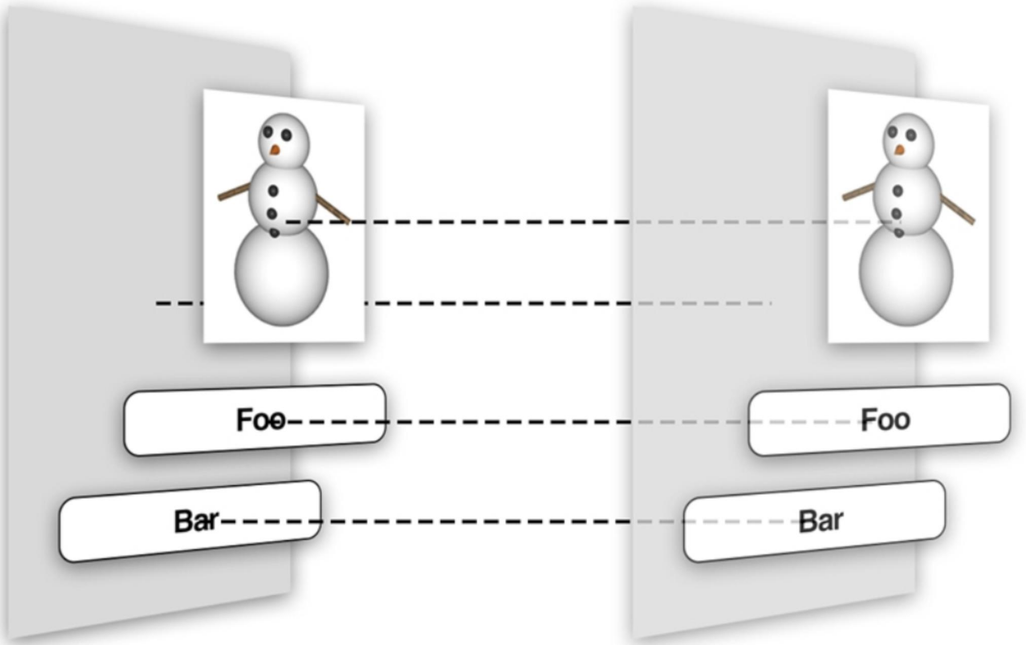
У каждого объекта класса UIView существует свойство, которое называется layer и его тип CALayer.

Задача объекта layer внутри объекта UIView — отрисовать весь контент, который должен находиться внутри прямоугольной области (после того, как система посчитала её месторасположение).

Подобно объектам класса UIView, все объекты класса CALayer образуют иерархию. Так у системы есть конкретная структура — дерево, у которого есть корневой объект (привет, UIWindow) и есть вложенные в него объекты sublayers.

Как iOS на самом деле отрисовывает контент на экране

- При добавлении дочерней view, под капотом происходит добавление дочернего слоя на родительский.
- Все изменения свойств frame, bounds, center, backgroundColor и многих прочих просто проксируются в CALayer. Иерархия объектов внутри view дублируется для её layer.



Как iOS на самом деле отрисовывает контент на экране

Таким образом между объектами UIView и CALayer происходит чёткое разделение ответственности — иерархия UIView ответственна за User Interactor, а иерархия CALayer за графическое представление на экране.

Кроме задачи по отрисовке контента, CALayer также поддерживает актуальную информацию для UIView про геометрию своего контента: позиция на экране (origin), размер (size) и трансформации. При изменении этих свойств система должна знать, в какой области ей перерисовать свой контент.

Как iOS на самом деле отрисовывает контент на экране

- Вся работа по управлению иерархией UIView выполняется на главном потоке (main thread), для чего используется только CPU, ядро процессора.
- В свою очередь, работа по управлению иерархией CALayer выполняется в отдельном потоке (background thread), и для отрисовки контента используются мощности GPU смартфона / планшета.

Как iOS на самом деле отрисовывает контент на экране

- Как и в случае с UIView, у CALayer есть набор классов-наследников, которые расширяют базовый функционал обычного свойства layer.
- Для решения более сложных задач существуют такие классы: CAShapeLayer, CATextLayer, CAGradientLayer.
- Каждый из классов позволяет решить ту или иную задачу эффективным способом, при этом выполняя всю отрисовку контента на GPU.

Как iOS на самом деле отрисовывает контент на экране

Важно: рисование контента происходит на отдельном потоке и выполняется на GPU. Но это если мы используем layers, которые так или иначе связаны с их views.

Но отрисовать контент на экране можно также и с помощью самой view, внутри специального метода draw(in:). Вызов этого метода происходит как и в случае с другими методами класса UIView на главном потоке, используя ресурсы CPU.

Важно знать, что интенсивное использование метода draw(in:) или слишком долгое выполнения кода внутри этого метода будет вызывать фризы (торможение) пользовательского интерфейса! Старайтесь по возможности всегда использовать CALayer.

Как iOS на самом деле отрисовывает контент на экране

Перед тем, как непосредственно перейти к обсуждению анимаций в iOS, хочу рассказать про ещё один важное свойство CALayer, про которое часто спрашивают на собеседованиях у разработчиков всех уровней и возрастов!

Каждый объект CALayer условно состоит из 2-х основных частей: (1) графический контент, который система должна отрисовать, и (2) набор параметров, которые описывают, как именно данный контент должен быть отрисован, его геометрию и трансформации (такие свойства, как opacity, transform, position, etc).

Как iOS на самом деле отрисовывает контент на экране

Когда CALayer анимируется, у объекта появляется 2 копии всех его параметров:

- свойство presentation() — хранит данные всех параметров CALayer с достоверной точностью относительно хода выполнения анимации.
- свойство model() — хранит конечные данные после окончания анимации.

Если нам надо узнать точное значение размеров объекта UIView во время выполнения анимации поворота на 360 градусов относительно своего центра, стоит использовать свойство presentation() у CALayer.

Способы работы с анимациями

Анимации в iOS можно разделить на 2 вида:

- implicit animations
- explicit animations

Способы работы с анимациями

Implicit animations — это те анимации, которые выполняются в блоке метода UIView.animate(…) для анимированных свойств UIView.

По-умолчанию у UIView для таких свойств, как frame, bounds, center, transform, alpha и backgroundColor анимации будут приниматься системой.

Способы работы с анимациями

Explicit animation — те анимации, для которых мы пишем код сами.

За создание и управление анимацией ответственен абстрактный класс CAAnimation. Если ему задать делегата и определить его методы, то можно будет отслеживать прогресс выполнения анимации.

Так же для explicit animations мы явно указываем такие параметры, как тайминг-функцию, длительность анимации, значения “от” и “до”.

Способы работы с анимациями

Мы можем использовать CAAnimation чтобы определить кастомную логику обработки для анимации, либо воспользоваться одним из существующих классов-наследников CAAnimation.

Фреймворк CoreAnimation уже содержит следующие классы:

- CABasicAnimation — обычная анимация, интерполирующая значения от начальной точки fromPoint до конечной toPoint.
- CAKeyFrameAnimation — анимация, интерполирует значения между двумя ключевыми кадрами, заданные с пом. массива value и keyTimes.
- CASpringAnimation — анимация пружины.

Способы работы с анимациями

Важно помнить, что анимации привязаны к жизненному циклу UIView, а также к application life cycle. Если приложение уйдет в background или удалится какой-то объект view из иерархии UIView, то и анимация пропадёт (анимации удалятся вместе с объектом).

Обратите внимание на свойство isRemovedOnCompletion — при установлении значения false конечные значения анимируемых свойств (для explicit animations) сохранятся в модельном представлении, свойство model().

Demo Time 

спасибо

задавайте вопросы