Что мы узнаем сегодня?

- UIView анимации
- CoreAnimation

Devteam

Зачем нужна анимация?

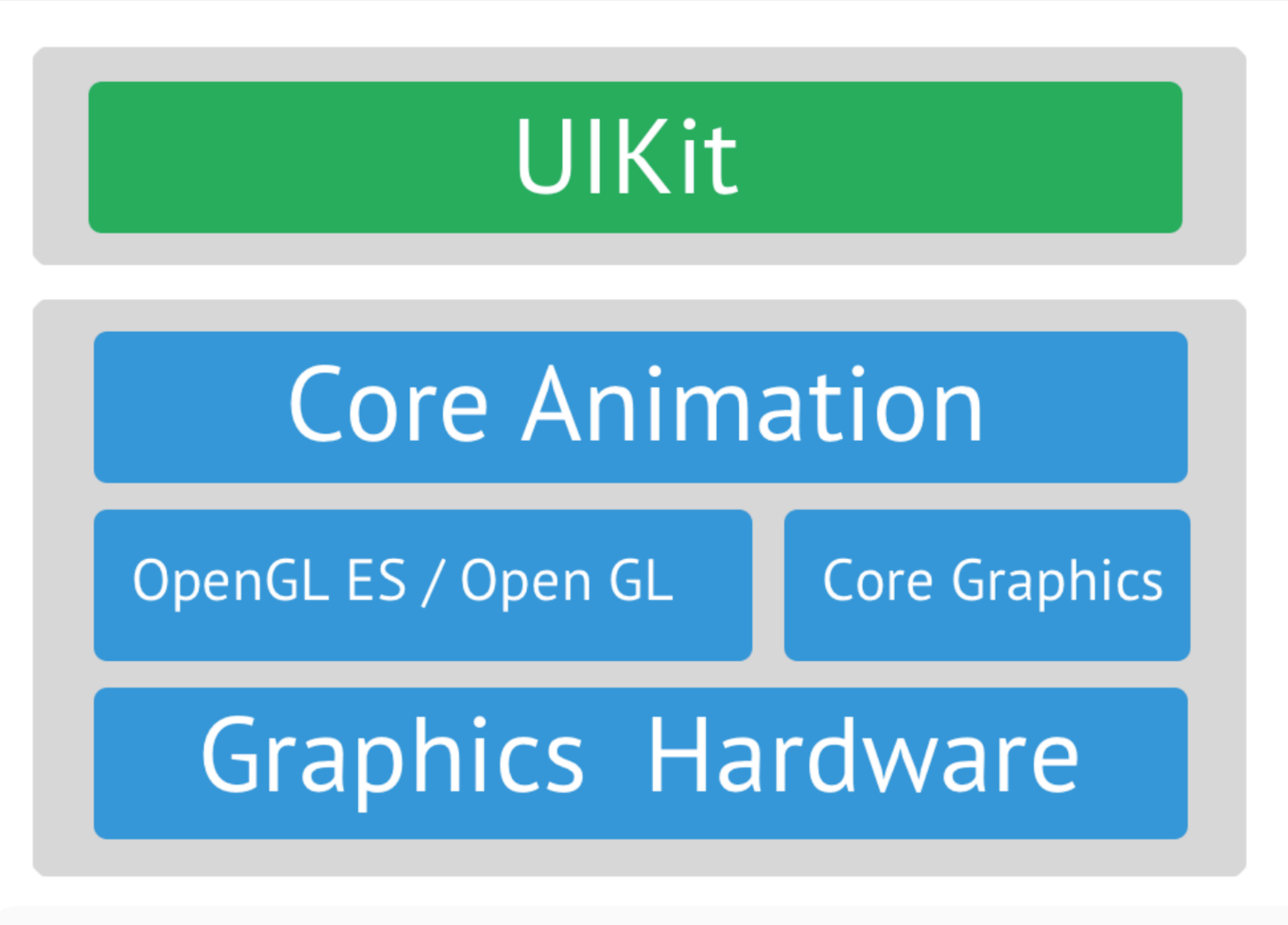
- привлечь внимание пользователя к конкретному объекту;
- показать ему, как и что нужно сделать в том или ином случае;
- показать логику перехода или иерархичность экранов, что помогает ориентироваться в приложении;
- убедить пользователя в том, что совершённое действие действительно совершено;
- разнообразить приложение, ведь без анимаций оно будет выглядеть очень сухо и пользоваться им будет неинтересно.

Core Animation

- фреймворк для работы с базовыми классами анимации: CABasicAnimation, CAKeyFrameAnimation, CATransition, CAAnimationGroup. Использование Core Animation полностью автоматизировано: не нужно создавать циклов и таймеров, чтобы сделать анимацию

Devteam

Core Animation



CALayer

- Объект, который управляет содержимым на основе изображений и позволяет выполнять анимацию для этого содержимого.

Devteam

Модели анимации

- Неявная модель анимации Core Animation предполагает, что все изменения в анимируемых свойствах слоя должны быть постепенными и асинхронными. Анимация будет происходить без эффектов, переходя от одного значения к другому.
- Явная модель анимации требует создания объекта анимации и постановки начальных и конечных значений и будет протекать плавно от одного значения к другому. Анимация не начнётся, пока не будет добавлена к слою.

Неявная анимация

Devteam

Timer

```
var timer : Timer?
timer = Timer.scheduledTimer(timeInterval: 1.0,
                                                target: self,
                                                selector: #selector(updateTimer),
                                                userInfo: nil,
                                                repeats: true)

timer?.invalidate()
timer = nil
```

Явная анимация

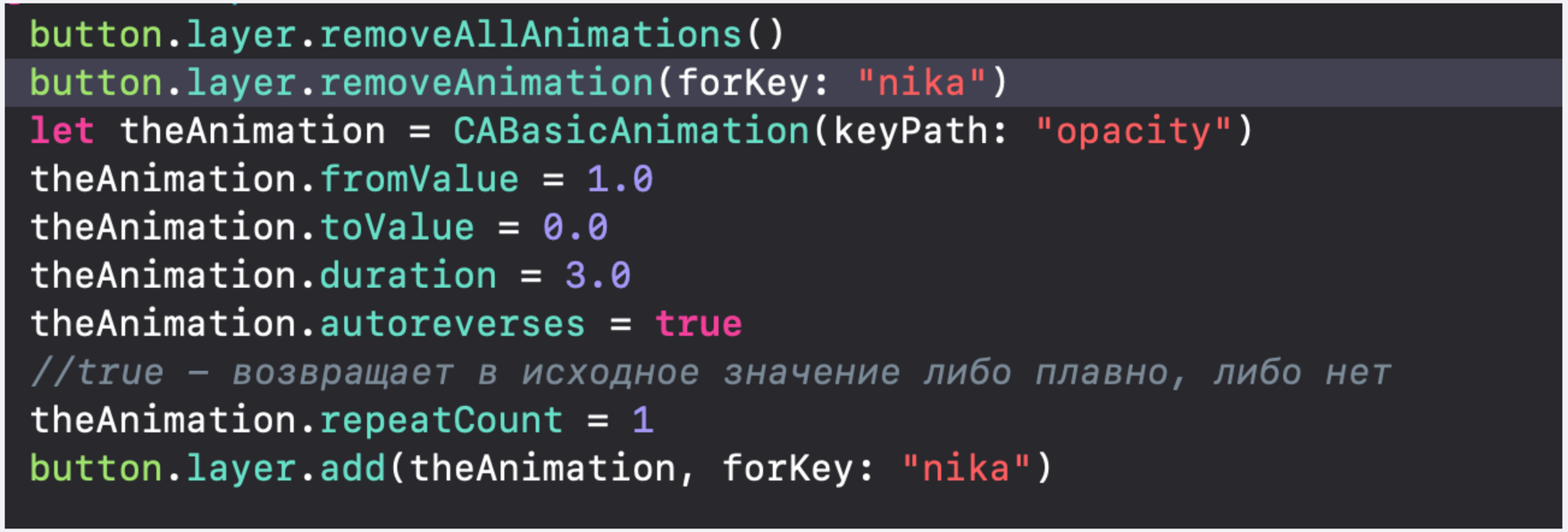
Devteam

Удаление анимаций

- theLayer.removeAnimationForKey(«animateOpacity»)
- theLayer.removeAllAnimations()

Devteam

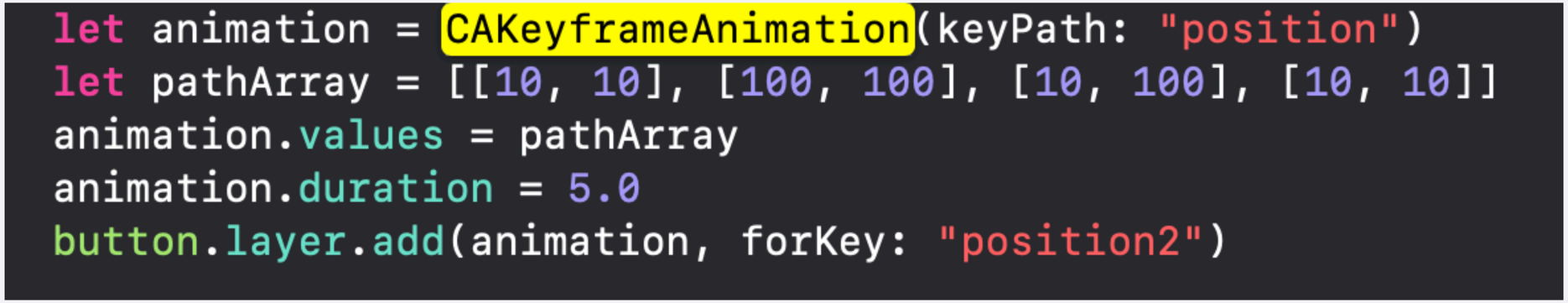
CABasicAnimation



```
button.layer.removeAllAnimations()
button.layer.removeAnimation(forKey: "nika")
let theAnimation = CABasicAnimation(keyPath: "opacity")
theAnimation.fromValue = 1.0
theAnimation.toValue = 0.0
theAnimation.duration = 3.0
theAnimation.autoreverses = true
//true - возвращает в исходное значение либо плавно, либо нет
theAnimation.repeatCount = 1
button.layer.add(theAnimation, forKey: "nika")
```

Devteam

CAKeyframeAnimation



```
let animation = CAKeyframeAnimation(keyPath: "position")
let pathArray = [[10, 10], [100, 100], [10, 100], [10, 10]]
animation.values = pathArray
animation.duration = 5.0
button.layer.add(animation, forKey: "position2")
```

CATransition

```
let transition = CATransition()
transition.duration = 1.5
transition.timingFunction = CAMediaTimingFunction(name:
    CAMediaTimingFunctionName.easeInEaseOut)
transition.type = CATransitionType.fade

navigationController?.view.layer.add(transition, forKey: "transition")

navigationController?.pushViewController(viewController, animated: false)
```

Devteam

CATransition.Type

- kCATransitionFade. Содержимое слоя исчезает, как только он становится видимым или невидимым;
- kCATransitionMoveIn. Содержимое слоя скользит поверх текущего контента. С этим типом используются простые подтипы transition.subtype;
- kCATransitionPush. Содержимое слоя выталкивает существующий контент. С этим типом используются простые подтипы transition.subtype;
- kCATransitionReveal. Содержание слоя раскрывается постепенно в направлении, указанном подтипом для перехода. С этим типом используются простые подтипы transition.subtype.

CATransition.Subtype

- kCATransitionFromRight. Представление начинается справа;
- kCATransitionFromLeft. Представление начинается слева;
- kCATransitionFromTop. Представление начинается сверху;
- kCATransitionFromBottom. Представление начинается снизу;

Devteam

CAAnimationGroup

```
let animation = CAAnimationGroup()
animation.animations = [theAnimation1, theAnimation2, theAnimation3]
animation.duration = theAnimation3.beginTime + theAnimation3.duration
button.layer.add(animation, forKey: "position3")
```

Devteam

Animations

- Изменения в нескольких свойствах view могут быть анимированными, то есть изменение свойства создает анимацию, начиная с текущего значения и заканчивая новым указанным значением.

Devteam

Animations

- frame
- bounds
- center
- transform
- alpha
- backgroundColor

Devteam

UIViewPropertyAnimator

- Чтобы анимировать ваши изменения, создайте объект UIViewPropertyAnimator и используйте его блок-обработчик для изменения значений свойств вашего представления.
- Класс UIViewPropertyAnimator позволяет указать продолжительность анимации.

UIViewPropertyAnimator

```
UIView.animate(withDuration: 0.3) {
    view.frame = view.frame.offsetBy(dx: 100, dy: 0)
}

let animator = UIViewPropertyAnimator(duration:0.3, curve: .linear) {
     view.frame = view.frame.offsetBy(dx:100, dy:0)
 }
animator.startAnimation()
```

Devteam

Домашнее задание

- Анимация любого процесса, например, роста дерева с яблоками

Devteam

Полезные ссылки

- https://developer.apple.com/documentation/quartzcore
- https://medium.com/@joncardasis/better-ios-animations-with-catransaction-72a7425673a6
- https://www.raywenderlich.com/113835-ios-timer-tutorial
- https://developer.apple.com/documentation/foundation/timer
- https://developer.apple.com/documentation/quartzcore/caanimationgroup
- https://www.raywenderlich.com/books/ios-animations-by-tutorials/v6.0/chapters/12-groups-advanced-timing

Devteam

Полезные ссылки

- https://developer.apple.com/documentation/quartzcore/cabasicanimation
- https://developer.apple.com/documentation/quartzcore/cakeyframeanimation
- https://developer.apple.com/documentation/quartzcore/catransition
- https://developer.apple.com/documentation/quartzcore/caanimation
- https://developer.apple.com/documentation/quartzcore/caanimationdelegate

Devteam

Обратная связь

Devteam

Вопросы?

Devteam

Спасибо 💚

Devteam