Objective-C

Курс лекций и семинаров для студентов,
желающих научиться программировать под iPhone

Осень-Зима 2013

Лекция №5

Компоненты UI: UIView, иерархия UI элементов, Responder chain

Автор: Дмитрий Волков, iPhone Developer, Sibers

Sibers®

Что вы узнаете сегодня?

UIView:

1. Обзор
2. Размеры экрана
3. Система координат
4. Иерархия компонентов
5. Обработка touch-событий
6. Responder chain
7. UIControl, Target-action
8. Gesture recognizers

UIView

UIView — базовый класс UI объектов, с которыми взаимодействует пользователь.

- Те самые View из паттерна Model-View-Controller. Визуализируют ваши «модельные» данные.

- UIKit предоставляет обширный набор подклассов UIView для конкретных задач отображения данных. (Кнопки, текстовые поля и лейблы, компоненты для выбора даты / времени и прочее).

Размеры экрана

На данный момент устройства на платформе iOS поддерживают 3 размера экрана устройства:

- ***320 x 480 (640 x 960)*** — iPhone 4s и ниже. iPod touch 4 gen и ниже.
- ***320 x 568 (640 x 1136)*** — iPhone 5 и выше. iPod touch 5 gen.
- ***1024 x 768 (2048 x 1536)*** — iPad / iPad mini.

Начиная с iPhone 4, экраны устройств используют технологию Retina, что увеличивает разрешение экрана в 2 раза.

Размеры экрана, Points and Pixels

В iOS координатная система экрана разделена на 2 уровня:

- **Логическая** — измеряется в точках (points). Именно в этой системе и указываются размеры и положение UI элементов.
- **Физическая** — определяет, сколько пикселей соответствует одной точке (pixels per point).

Это означает, что на Retina и non-Retina устройствах размер UI элементов, определяемых кодом, не изменяется.

Класс ***UIScreen*** предоставляет свойство scale, которое возвращает количество пикселей в одной точке

Для Retina устройств scale = 2.0, для non-Retina — scale = 1.

Система координат

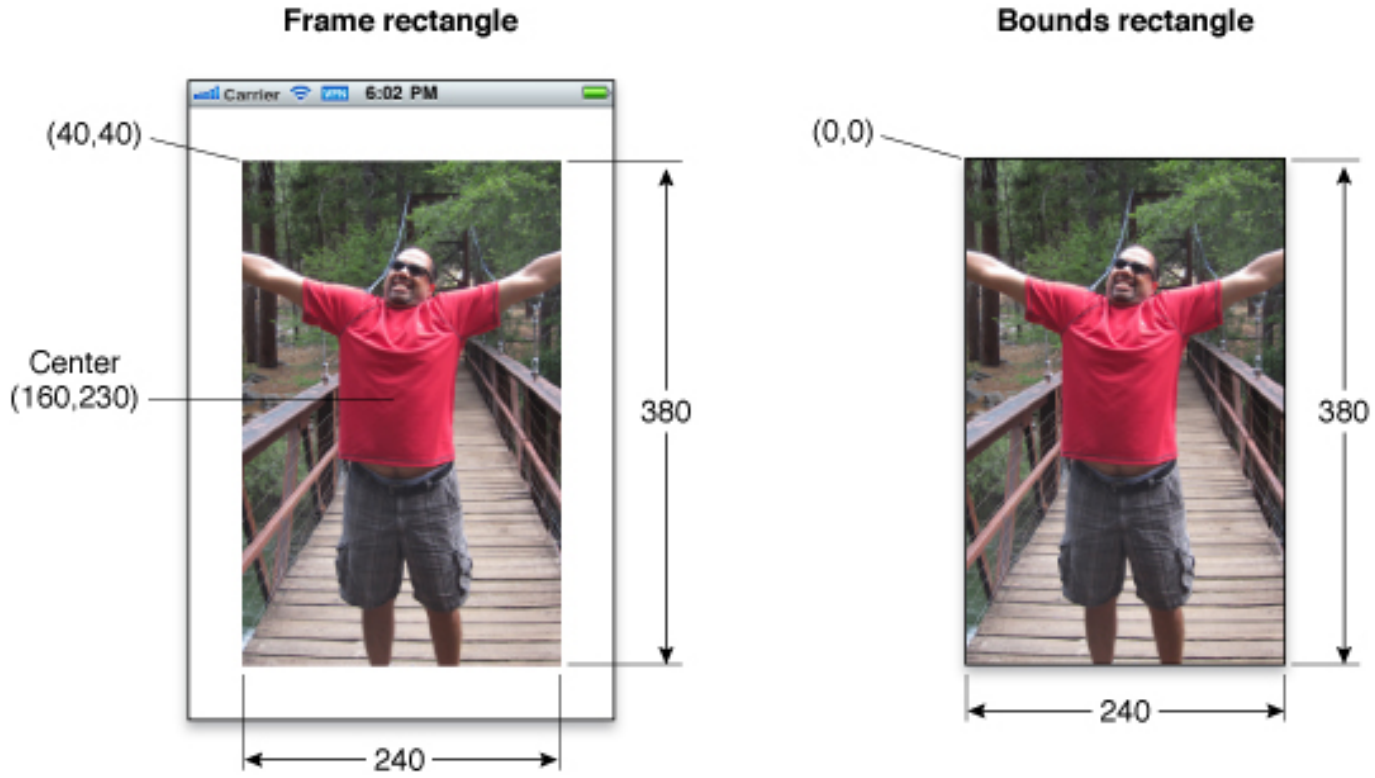
В UIKit началом координат является ***ВЕРХНИЙ ЛЕВЫЙ УГОЛ***.

- Тем временем, во фреймворке CoreGraphics, который отвечает за рисование контента, начало координат — *НИЖНИЙ ЛЕВЫЙ УГОЛ.*

Положение и размеры компонентов UI описываются 4мя свойствами:

- ***Frame***— задает положение элемента в координатах его контейнера.
- ***Bounds*** — собственная (локальная) система координат элемента.
- ***Center*** — центр элемента в координатах контейнера.
- ***Transform*** — афинная трансформация, примененная к элементу.

Система координат



Система координат, трансформации

Свойство ***bounds*** так же описывает видимую часть контента.

Свойство transform позволяет применять к элементам UI афинные трансформации:

- **Translation** (смещение)
- **Rotation** (вращение)
- **Scale** (масштабирование)
- Комбинации этих трансформаций

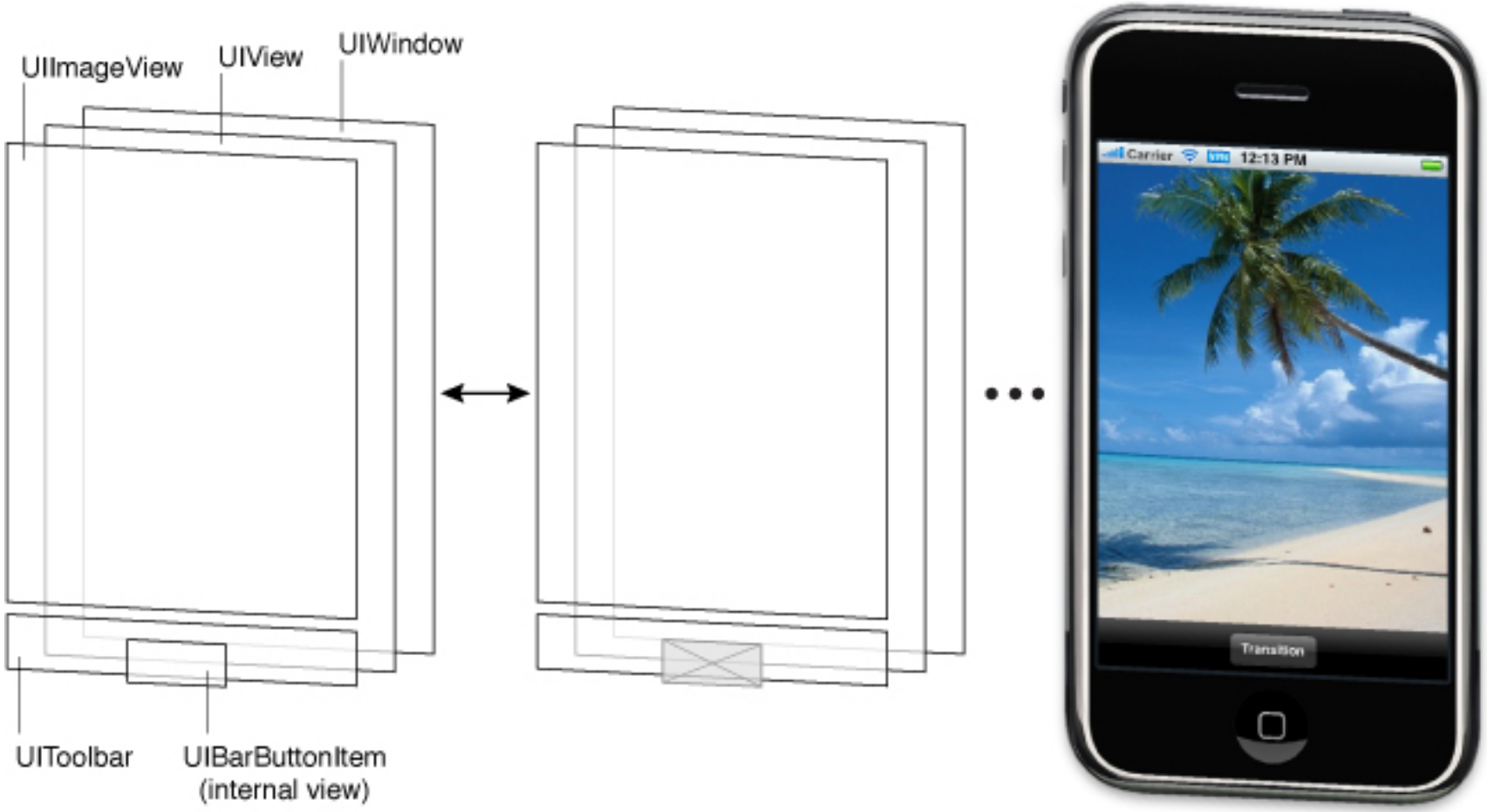
ВАЖНО: трансформации изменяют свойство frame, что делает его невалидным.

Система координат, трансформации

Unrotated

Rotated 45°

Иерархия компонентов



Иерархия компонентов

Кроме отображения контента, каждый UI-компонент так же может содержать другие компоненты, выстраивая таким образом более сложный и богатый UI.

Каждый наследеник UIView имеет следующие свойства, помогающие управлять иерархиями компонентов:

- ***subviews*** — объекты типа UIView, добавленные на текущий UIView
- ***superview*** — UIView-контейнер, в котором содержится UIView.

Иерархия компонентов

Для добавления, удаления и управления компонентами в иерархии, UIView определяет следующие методы:

- ***- (void) addSubview:(UIView*) view*** — добавляет UIView в набор subviews компонента-получателя сообщения.
- ***- (void) removeFromSuperview*** — удаляет компонент из своего контейнера.

Методы для управления порядком компонентов:

- ***- (void) bringSubviewToFront:(UIView*) view***
- ***- (void) sendSubviewToBack:(UIView*) view***
- ***- (void) insertSubview:(UIView*) view atIndex:(NSInteger) index***

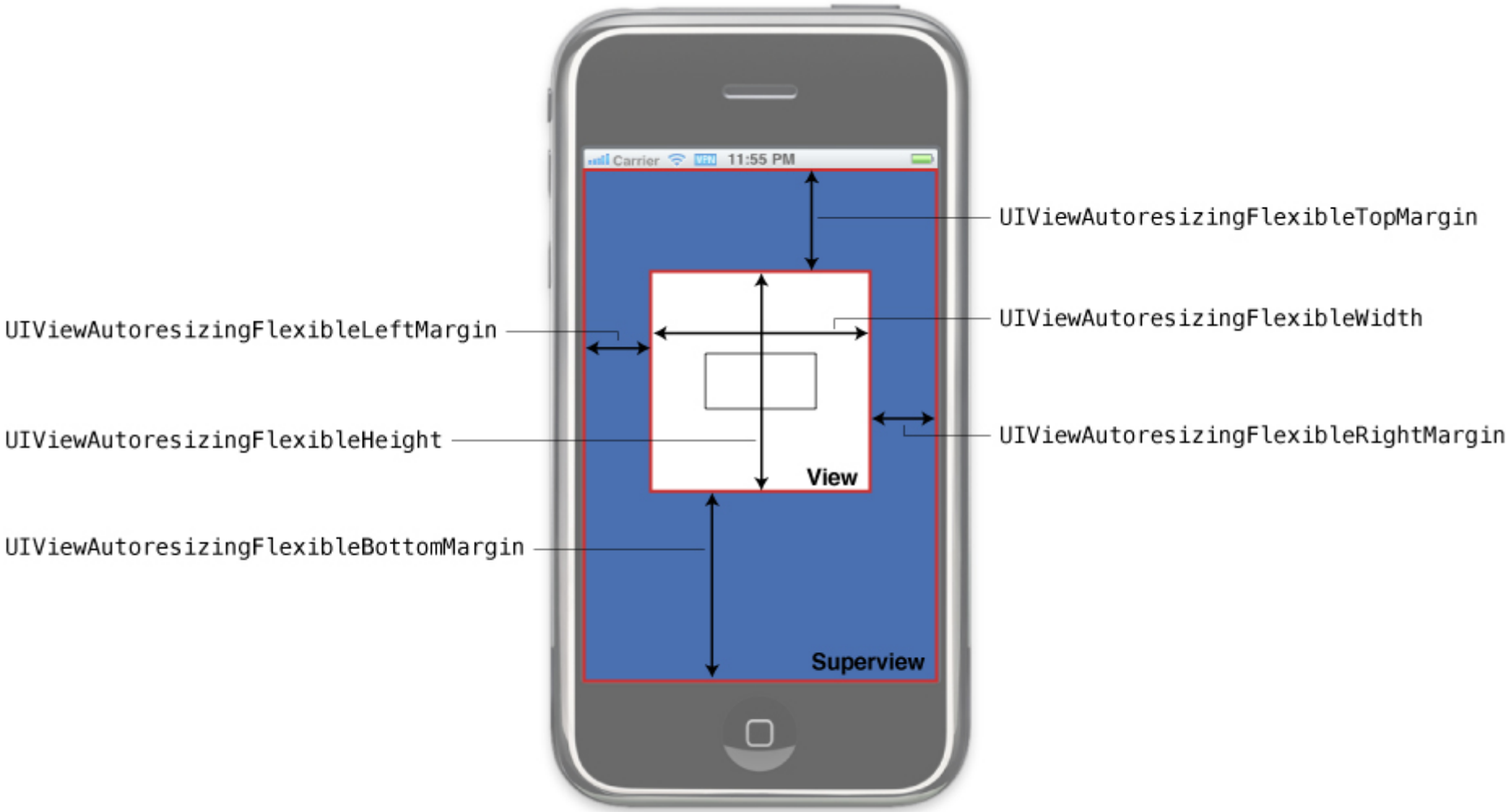
Иерархия компонентов

При изменении своих размеров, каждый объект UIView так же может изменить размеры и положение своих дочерних компонентов.

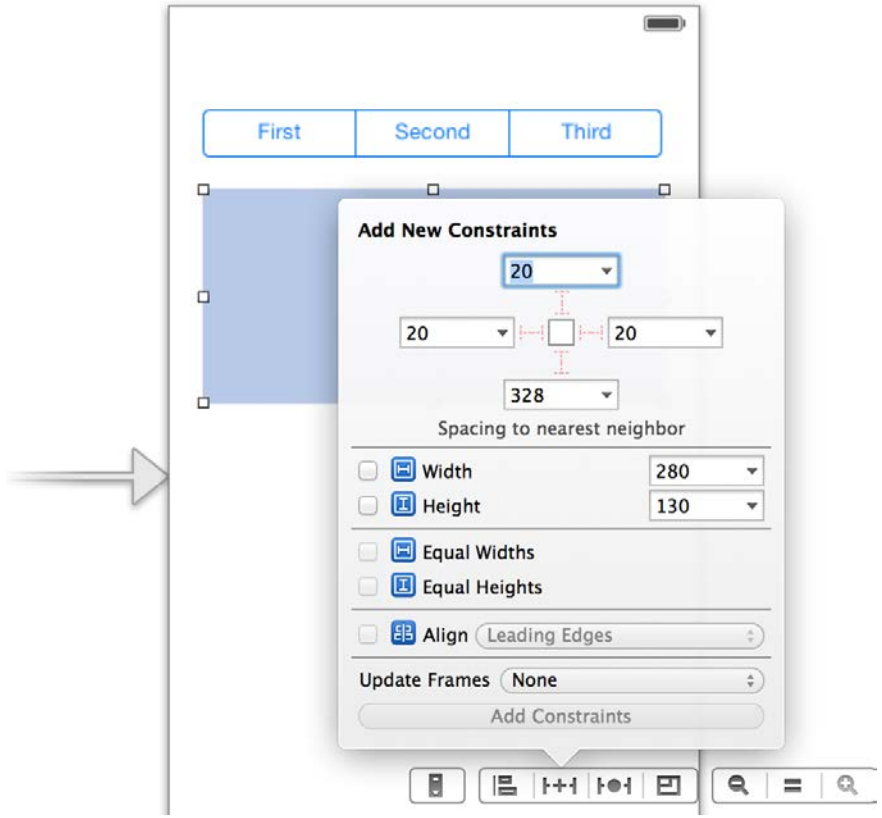
Для этого используются 3 подхода:

- ***Autoresizing masks*** — описывают относительное изменение размера и положения UIView относительно своего superview
- ***Autolayout*** — описывает ограничения, относительно которых UI-компоненты изменяют свой размер при изменении размера своего контейнера.
- Переопределение метода ***layoutSubviews*** и управление положением компонентов в коде.

Autoresizing



Autolayout


First
Second
Third
Add New Constraints
20
20
20
328
Spacing to nearest neighbor
Width
280
Height
130
Equal Widths
Equal Heights
Align
Leading Edges
Update Frames
None
Add Constraints

Иерархия компонентов

- Основой иерархии UI-компонентов iOS приложения является объект ***UIWindow***.
- ***UIWindow*** создается при запуске приложения. Далее, все UI-компоненты добавляются как дочерние на ***UIWindow***.
- Обычно, приложение содержит только один объект типа ***UIWindow***.

UIWindow занимается следующим:

- Отображает ваш контент
- Сообщает своим под-компонентам о событиях взаимодействия пользователя с UI
- Сообщает контроллерам о поворотах устройства

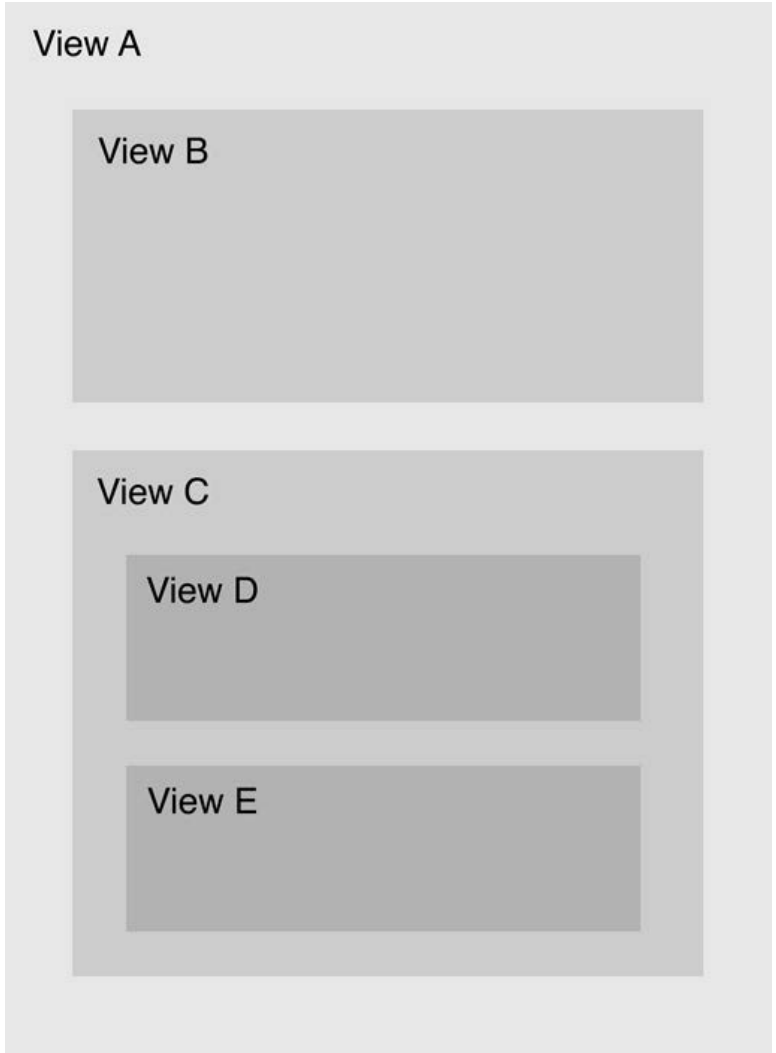
Обработка touch-событий

Для обработки взаимодействия пользователя с UI и внешних событий в iOS используется механизм Responder Chain.

Для touch-событий:

- ***UIWindow*** пытается передать touch в объект типа UIView, в котором он был вызван (hit-testing).
- Если UIView не может обработать данный touch (например, объект в данный момент невидим, либо отключил возможность взаимодействия с собой), touch передается в ***superview*** данного компонента, и так далее вверх, пока он не будет обработан.

Обработка touch-событий



Responder chain

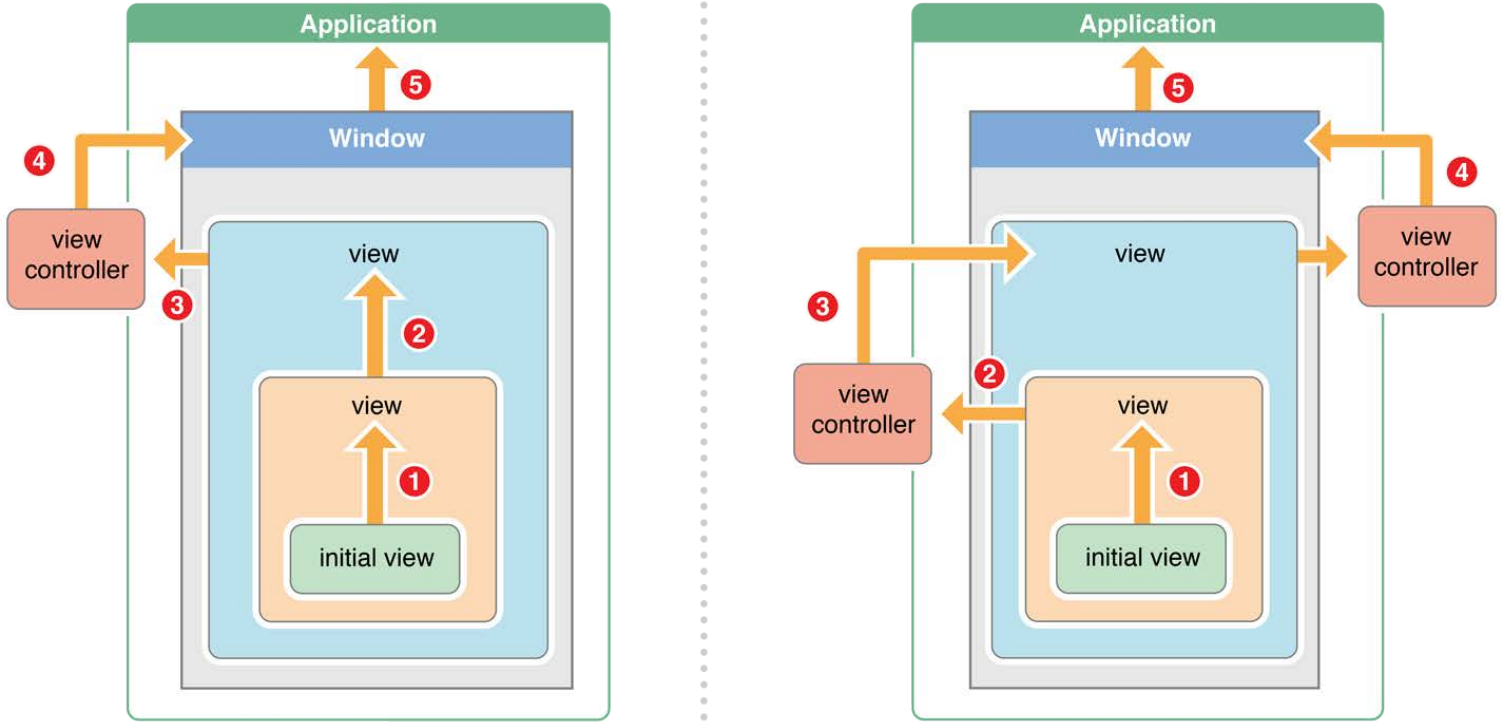
Классы ***UIApplication***, ***UIViewController*** и ***UIView*** наследуются от класса ***UIResponder***.

- ***UIResponder*** определяет порядок, в котором объекты обрабатывают события (touch-события, события от элементов UI (кнопки, слайдеры), изменение текста)
- ***UIResponder*** объявляет методы, которые позволяют объектам определять, кто первым будет отвечать и обрабатывать сообщения:

  becomeFirstResponder — объект-получатель сообщения будет первым получать все события, посылаемые системой.

  resignFirstResponder — объект-получатель отказывается от обработки сообщений первым.

Responder chain



UIControl: Target-action

Для обработки стандартных действий пользователя с UI (нажатие кнопки, изменение значения слайдера и тд) в UIKit / AppKit используется механизм target-action.

- UI-компоненты, которые могут посылать сообщения, наследуются от ***UIControl*** — класс-наследник UIView, конкретизирующий обработку действий пользователя и выполнения связанных с ними действий.
- Так же позволяют задать внешний вид компонента при определенном состоянии (выбран, нажат)

UIControl, Target-action.

```
//допустим, мы создаем кнопку в методе viewDidLoad

UIButton* button = [UIButton new];

[button addTarget:self action:@selector(buttonPressed:)
forControlEvents:UIControlEventTouchDown];


....


- (void) buttonPressed:(id) sender

{

}
```

Gesture recognizers

UIGestureRecognizer — специальный класс, позволяющий объектам UIView обрабатывать touch-события.

- UIKit предоставляет набор готовых подклассов UIGestureRecognizer для самых распространенных жестов: tap, long-tap, pinch, rotate

UIGestureRecognizer могут работать в 2х режимах: дискретном и непрерывном.

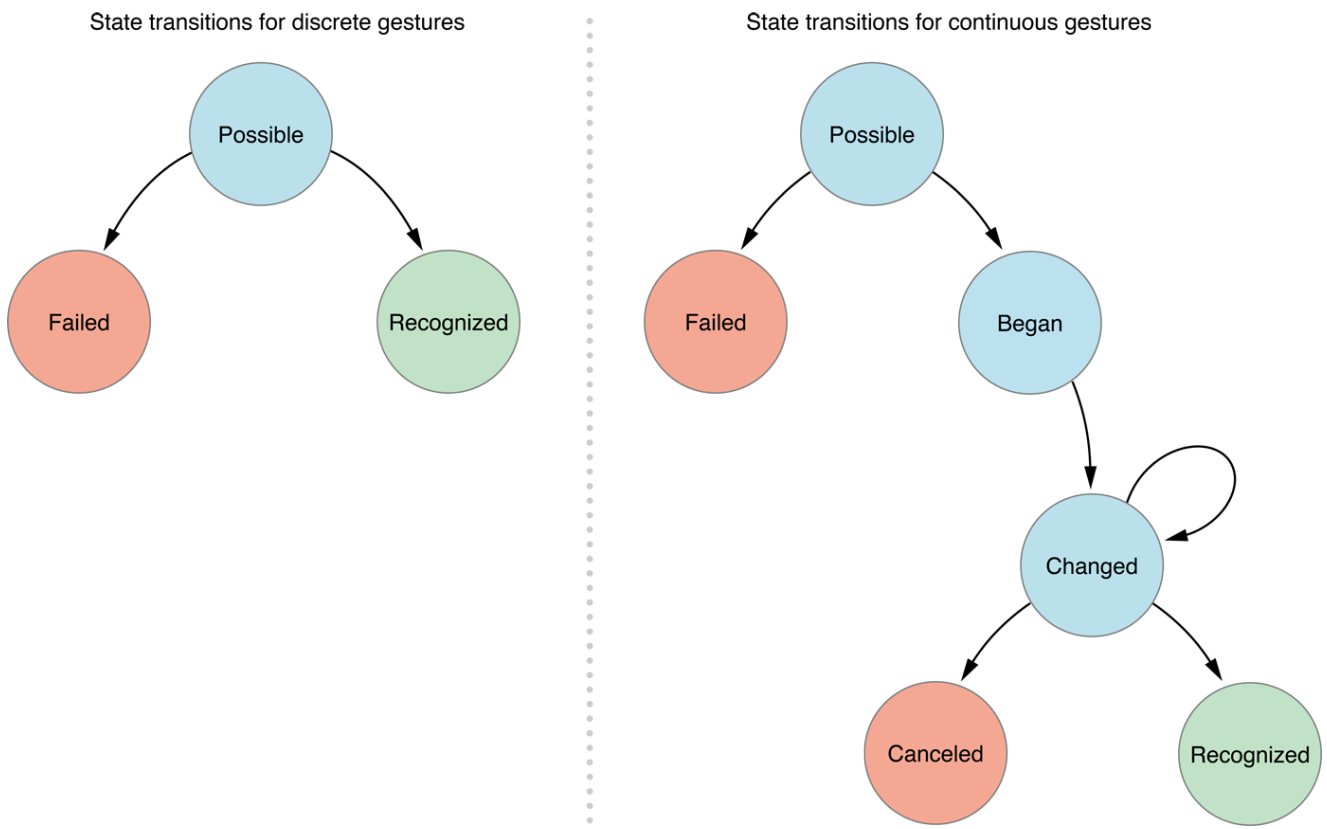
- Используют такой-же механизм target-action, как и UIControl.

Gesture Recognizers

```
//предположим, что мы снова в методе viewDidLoad
UITapGestureRecognizer* recognizer = [[UITapGestureRecognizer alloc]
initWithTarget:self action:@selector(viewTapped:)];
//UIGestureRecognizer добавляется к объектам UIView
[self.view addGestureRecognizer:recognizer];
...

- (void) viewTapped:(UIGestureRecognizer*) recognizer
{
...
}
```

Gesture Recognizers



Заключение

- UIView — основной элемент пользовательского интерфейса.
- iOS предоставляет множество наследников UIView для различных UI use case'ов.
- События между элементами UI передаются посредством паттерна Responder Chain.
- UIKit так же предоставляет стандартные классы-распознователи жестов — UIGestureRecognizer и его наследники.

В следующей лекции:

UIViewConotroller: Life cycle, View management, Containment

Спасибо за внимание!

Жду ваших вопросов.