UIView, UIControl, собственные UIView

Занятие №10

Дмитрий Тараев

План лекции

- Отметиться (чекин)?
- На прошлой лекции шла речь о БД и работе с сетью
- Для чего нужны view?
- Поговорим о:
  - **UIView**
    - кастомные view
    - UIControl
    - auto layout
  - **UIGestureRecognizer**
- Вопросы по проектам

Замечания по проектам

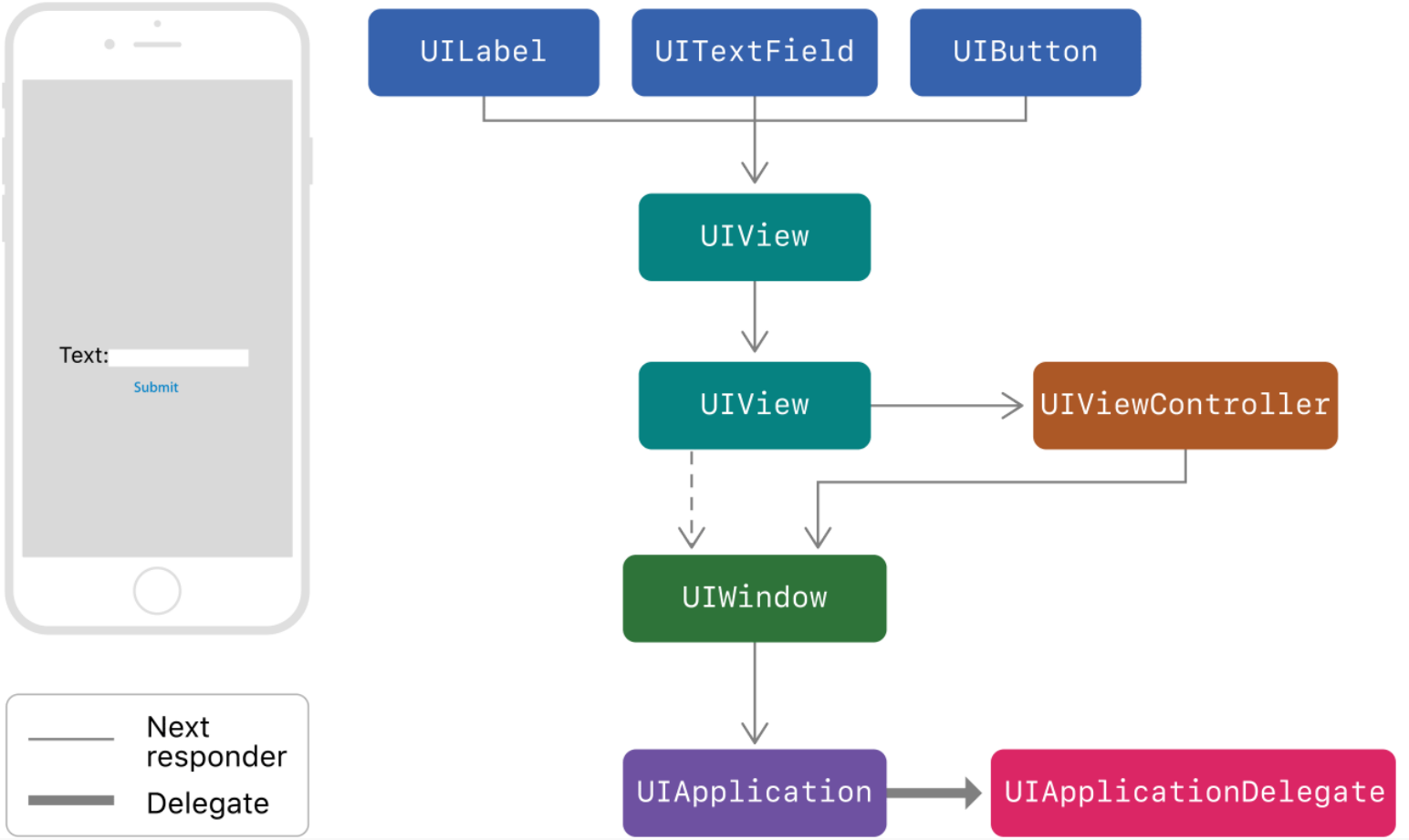
- Если есть аутентификация/авторизация
  - проверять в AppDelegate сессию, если нужно показывать окно авторизации
- Основным должен быть не экран авторизации!
- Не делать всю логику переходов в сторибордах (если она сложная)
  - Особенно не делать обратные переходы!
- Проект должен собираться без ошибок
  - Можно проверить: клонировать репозиторий в другое место

UIView

- Базовый класс UIKit («кирпичи» из которых делается интерфейс)
- Потомок UIResponder'а
    - обрабатывает события:
        - касания (touches)
        - движения (motions), например, встряхивание

- Другие потомки UIResponder:
    - UIViewContoller
    - UIApplication (центральная точка входа и управления приложением), используем
        - UIApplicationDelegate
        - UIApplication - один из примеров синглтона

Responder Chain



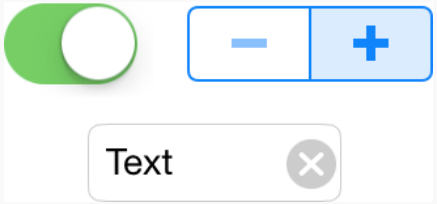
https://developer.apple.com/documentation/uikit/understanding_event_handling_responders_and_the_responder_chain

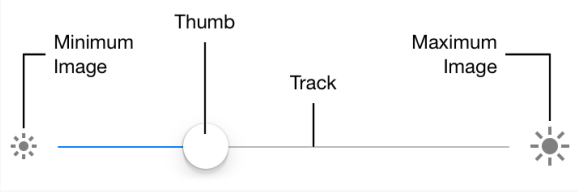
UIView (потомки)

- UIControl
  - UIButton, UITextField и др.
- UIWindow
  - редко нужен в iOS, но может пригодиться, если нужно что-то показать поверх status bar'а
- UILabel
- списки
  - UIScrollView,
  - UITableView,
  - UICollectionView,
  - UITextView
- UIImageView
- MKMapView
- WKWebView

UIControl

- UIControl
  - UIButton
  - UITextField
  - UISwitch
  - UISegmentedControl
  - UISlider
  - UIProgress
- Добавляет к UIView
  - механизм target/action
  - протокол (UITextFieldDelegate и пр.)





https://developer.apple.com/reference/uikit/uicontrol
https://developer.apple.com/reference/uikit/uislider

UIView (основные понятия)

- один superview
- много subview
- addSubview: (у родительской view)
  - добавляет к нашему view subview
- removeFromSuperview
  - удаляет наш view из его superview

- анимации
  - class func **animate**(**withDuration** duration: TimeInterval,
    delay: TimeInterval,
    options: UIView.AnimationOptions = [],
    animations: @escaping () -> Void,
    completion: ((Bool) -> Void)? = nil)

UIView (полезные свойства)

- CGFloat alpha
  - прозрачность (от 0 до 1)
- Bool isOpaque (непрозрачный)
  - true/false
- Bool isHidden (невидимый)
- Bool userInteractionEnabled (отключенный)
- Bool clipsToBounds (обрезать по границам)

- Bool translatesAutoresizingMaskIntoConstraints
  (при использовании auto layout кодом)

Распознавание жестов

- Вручную
  - touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?)touchesEnded:withEvent:
  - touchesMoved...
  - touchesCancelled...
- Использование UIGestureRecognizer (точнее его потомков)
  - кодом
  - с помощью Interface Builder

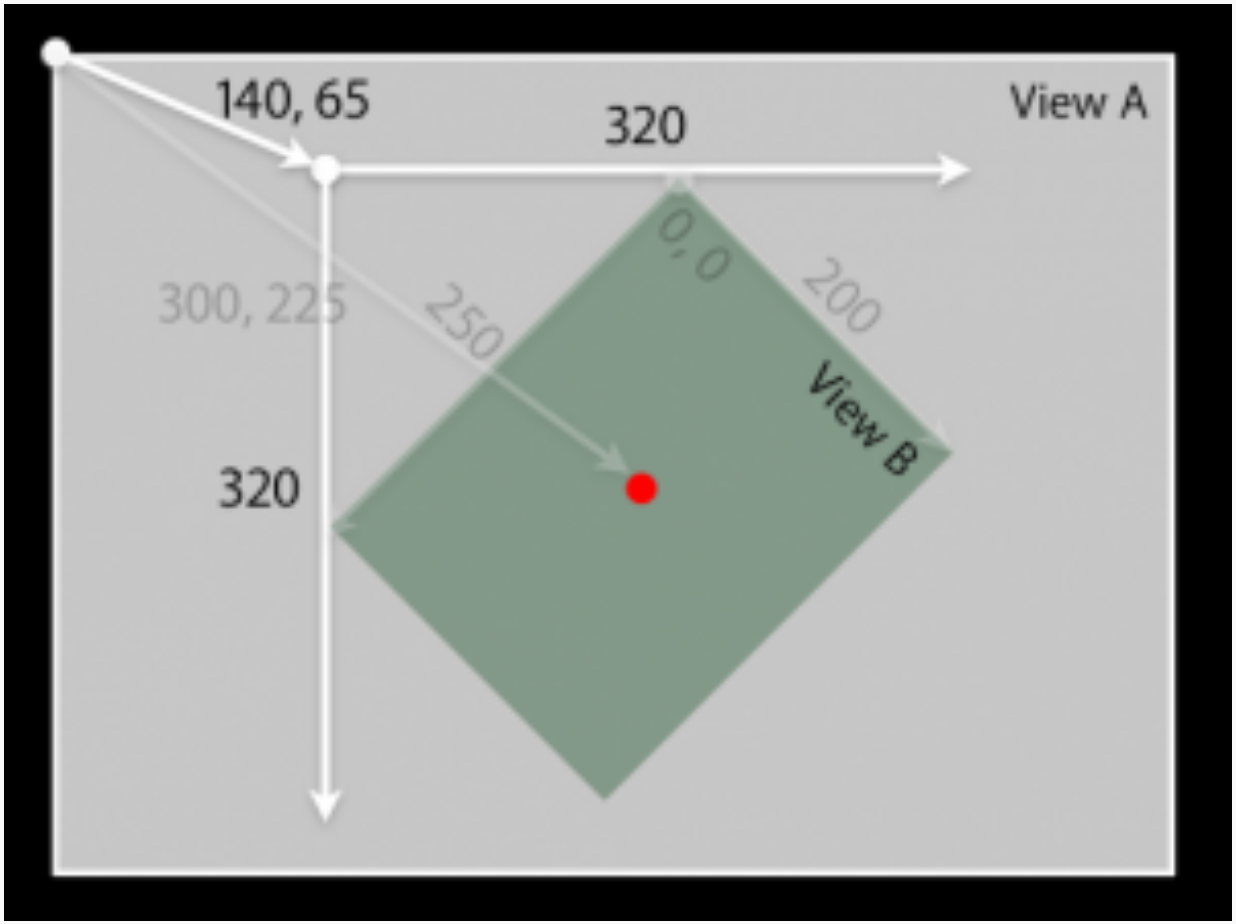
Пример распознавания жеста:


```
.............

let gestureRecognizer =

   UIGestureRecognizer(target: self, action: #selector(viewTapped(_:)))

view.addGestureRecognizer(gestureRecognizer)

.............

@objc func viewTapped(_ sender: UIGestureRecognizer) {
    print("viewTapped: \(gestureRecognizer.view)")
}
```

UIGestureRecognizer - основные виды

- tap
- swipe
- pan
- long press
- pinch

UIView (bounds / frame / center)



Курс Стэнфорда по iOS «Developing iOS 7 Apps for iPhone and iPad», слайды к лекции 7 (слайд 8)

Кастомные UIView

- Кастомизация
  - init(frame:) (могут отличаться у наследников UIView, например, у UITableViewCell)
    - не подходит для случая Interface Builder'а
  - awakeFromNib()
    - для view, созданного через Interface Builder
  - draw(_:)
    - если нужно во view что-то нарисовать

Кастомные UIView: -drawRect:

- В UIView можно рисовать:
  - **UIBezierPath**

```
let path = UIBezierPath()
```

    - moveToPoint:,
    - addLineToPoint
  - **Core Graphics**
    - контекст CGGetCurrentContext()
      - экран
      - pdf
      - принтер
    - функции для рисования

- Используется CPU, а не GPU

Пример drawRect:

```
1.  - (void)drawRect:(CGRect)rect {
    // Получаем указатель на контекст
2.  CGContextRef context = UIGraphicsGetCurrentContext();
    // Очищаем контекст
3.  CGContextClearRect(context, rect);

4.  CGContextSetRGBFillColor(context, 0, 255, 0, 1);
5.  CGContextFillRect(context, CGRectMake(10, 10, 150, 150));
6.  }
```

```
1.  - (void)drawRect:(CGRect)rect {

2.  UIBezierPath *path = [UIBezierPath bezierPath];

3.  [[UIColor greenColor] setFill];
4.  [path moveToPoint:CGPointMake(10.0, 10.0)];

5.  [path addLineToPoint:CGPointMake(160.0, 10.0)];
6.  [path addLineToPoint:CGPointMake(160.0, 160.0)];
7.  [path addLineToPoint:CGPointMake(10.0, 160.0)];
8.  [path closePath];

9.  [path fill];
```

```
[UIBezierPath bezierPathWithRoundedRect:CGRectMake(10.0, 10.0, 150.0, 150.0) cornerRadius:0.0];
```

Auto Layout

- Система линейных неравенств
- Constraints (ограничения)
  - основной класс NSLayoutConstraint
- Можно задать 3,5 (!) способами:
  - Interface Builder
  - Код (NSLayoutConstraint)
  - Код (visual format)
  - разные библиотеки (PureLayout)

Пример Auto Layout


```
UIView *subview = [[UIView alloc] init];

 subview.translatesAutoresizingMaskIntoConstraints = NO;
 [self.view addSubview:subview];

 NSArray *constraints = [NSLayoutConstraint constraintsWithVisualFormat:@"|-[subview]-|"
                                                                 options:0 metrics:nil
                                                                   views:@{@"subview": subview}];
 [self.view addConstraints:constraints];
 constraints = [NSLayoutConstraint constraintsWithVisualFormat:@"V:|-50-[subview]-50-|"
                                                                 options:0 metrics:nil
                                                                   views:@{@"subview": subview}];

 [self.view addConstraints:constraints];
```

```
 NSLayoutConstraint *constraint = [NSLayoutConstraint constraintWithItem:subview
                                                               attribute:NSLayoutAttributeLeft
                                                               relatedBy:NSLayoutRelationEqual
                                                                  toItem:self.view
                                                               attribute:NSLayoutAttributeLeft
                                                              multiplier:1.0 constant:50.0];
 [self.view addConstraint:constraint];


то же для

 NSLayoutAttributeRight
 NSLayoutAttributeTop
 NSLayoutAttributeBottom
```

CALayer

- Часть framework'а Core Animation
- Создаётся UIView (strong ссылка)
- нет обработки событий (не наследуется от UIResponder)
- нужен для сложных анимаций
- Нам может быть полезно:
  - cornerRadius
  - borderColor
  - shadowPath

Демо

МГТУ им. Н.Э. Баумана ТЕХНОПАРК Mail.Ru Group

- UIView / UIControl
- распознавание жестов
- Auto Layout

Заключение

- **UIView**
  - кастомные view
  - UIControl
  - auto layout
- **UIGestureRecognizer**

- **Следующая лекция**
  - Углубленные темы

- **Отзыв** (важно!)

- **Вопросы?**
  - по лекции
  - по домашним заданиям

Полезные ссылки

- **Apple Human Interface Guidelines (представления Apple о «прекрасном» - дизайне мобильного приложения, библия для мобильных дизайнеров, но разработчик тоже должен знать):**
  https://developer.apple.com/ios/human-interface-guidelines/

- **Подробно о кастомных control'ах, а не только view:**
  https://www.objc.io/issues/3-views/custom-controls/

- **Подробнее о visual format (для auto layout):**
  https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/AutolayoutPG/VisualFormatLanguage.html

- **Ссылка на упомянутый CocoaPods:** https://cocoapods.org

- **IBDesignable / IBInspectable (упомянуты в теме лекции, но не вошли) - вещь редкая, для фанатов Interface Builder'а:**
  http://nshipster.com/ibinspectable-ibdesignable/
  И вообще NSHipster - отличный ресурс