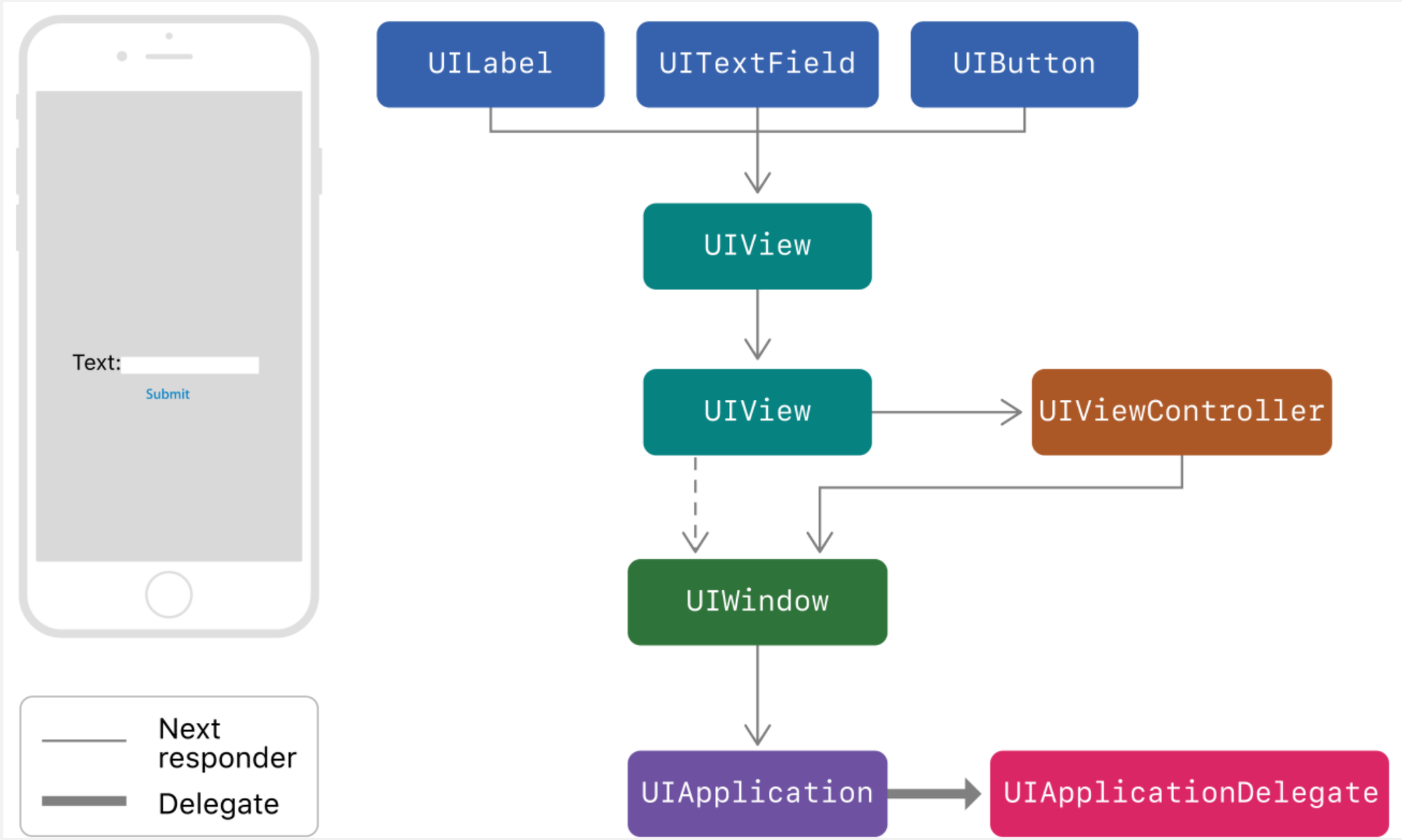
UIView

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

- Другие потомки UIResponder:
  - UIViewController
  - UIApplication (центральная точка входа и управления приложением), используем
    - UIApplicationDelegate
    - UIApplication

Responder Chain



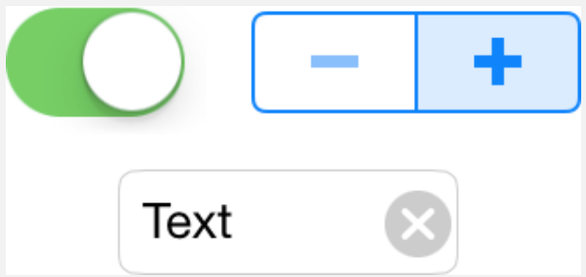
https://developer.apple.com/documentation/uikit/understanding_event_handling_responders_and_the_responder_chain

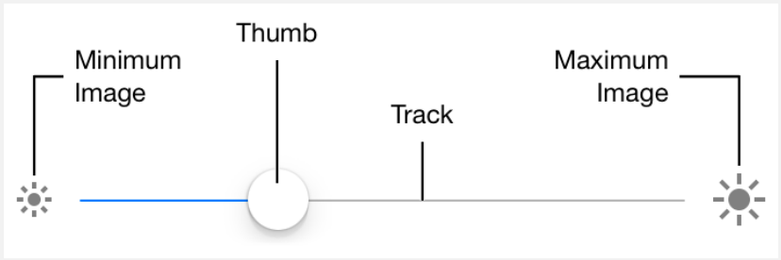
UIView (потомки)

- 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

UIGestureRecognizer - основные виды

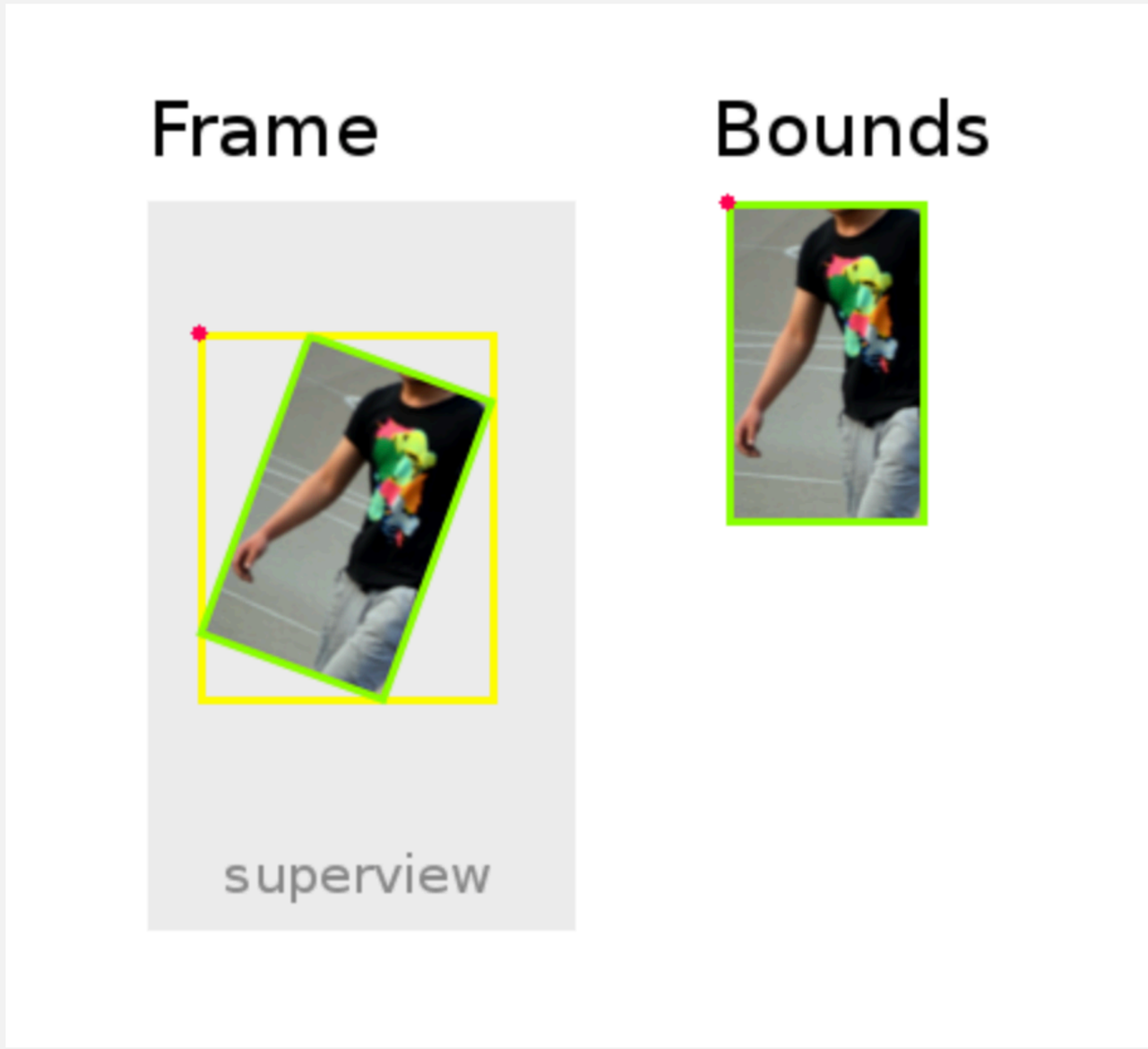
- tap
- swipe
- pan
- long press
- pinch
- rotation

UIView (bounds / frame)

- bounds - положение и размер в собственной системе координат
- frame - положение и размер в системе координат родительской вью

- Лучше менять bounds и center, а frame пусть считается сам

UIView (bounds / frame)



Кастомные UIView

- Кастомизация
  - init(frame:) (могут отличаться у наследников UIView, например, у UITableViewCell)
    - не подходит для случая Interface Builder’a
  - 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()
    - 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 разными способами:
  - Interface Builder
  - Код (NSLayoutConstraint)
  - Код (visual format)
  - Библиотеки (PureLayout)

Пример Auto Layout

```objc
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];
```

```objc
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
```

Layout Anchors


```
// Creating constraints using NSLayoutConstraint
NSLayoutConstraint(item: subview,
                   attribute: .leading,
                   relatedBy: .equal,
                   toItem: view,
                   attribute: .leadingMargin,
                   multiplier: 1.0,
                   constant: 0.0).isActive = true

NSLayoutConstraint(item: subview,
                   attribute: .trailing,
                   relatedBy: .equal,
                   toItem: view,
                   attribute: .trailingMargin,
                   multiplier: 1.0,
                   constant: 0.0).isActive = true


// Creating the same constraints using Layout Anchors
let margins = view.layoutMarginsGuide

subview.leadingAnchor.constraint(equalTo: margins.leadingAnchor).isActive = true
subview.trailingAnchor.constraint(equalTo: margins.trailingAnchor).isActive = true
```

CALayer

- Часть framework’a Core Animation
- Создаётся UIView (strong ссылка)
- нет обработки событий (не наследуется от UIResponder)
- нужен для сложных анимаций
- Пригодится:
  - cornerRadius
  - borderColor, borderWidth
  - shadowPath

CAShapeLayer

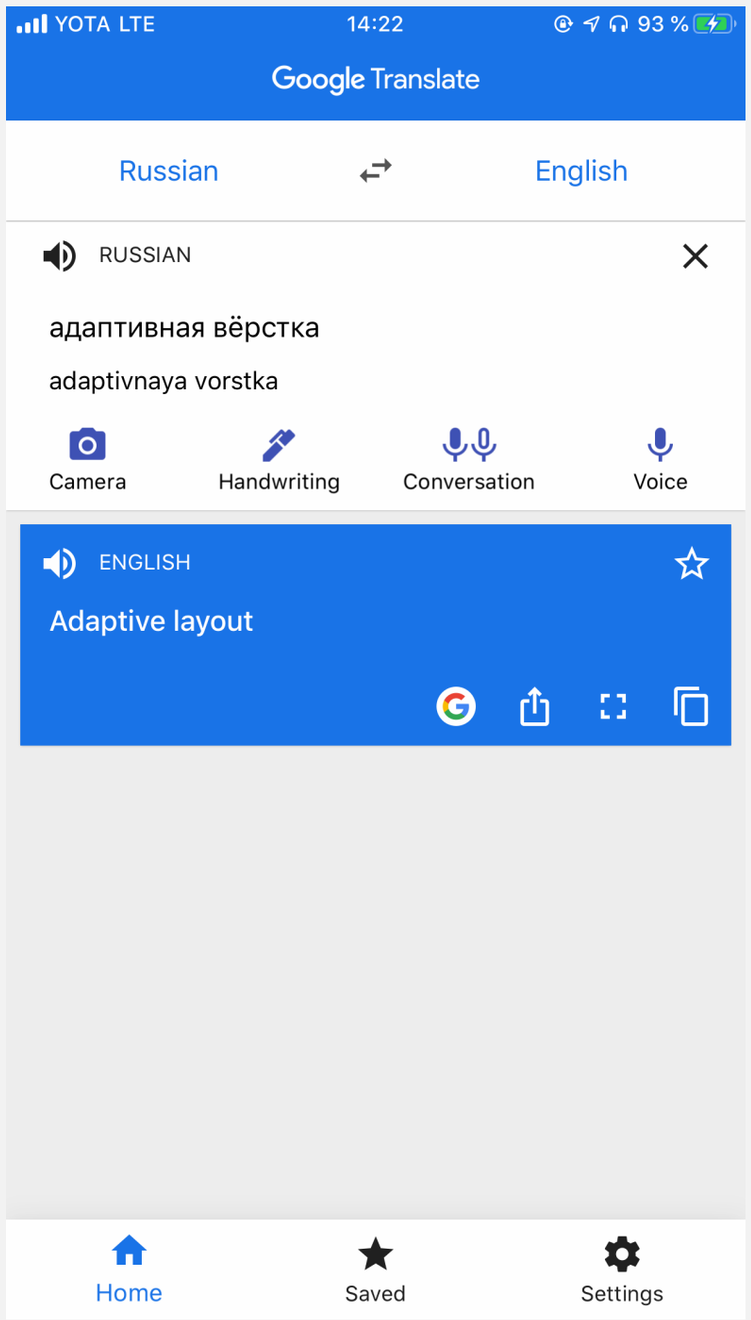
- Более низкий уровень по сравнению с UIView
- Используем CGColor (Core Graphics)

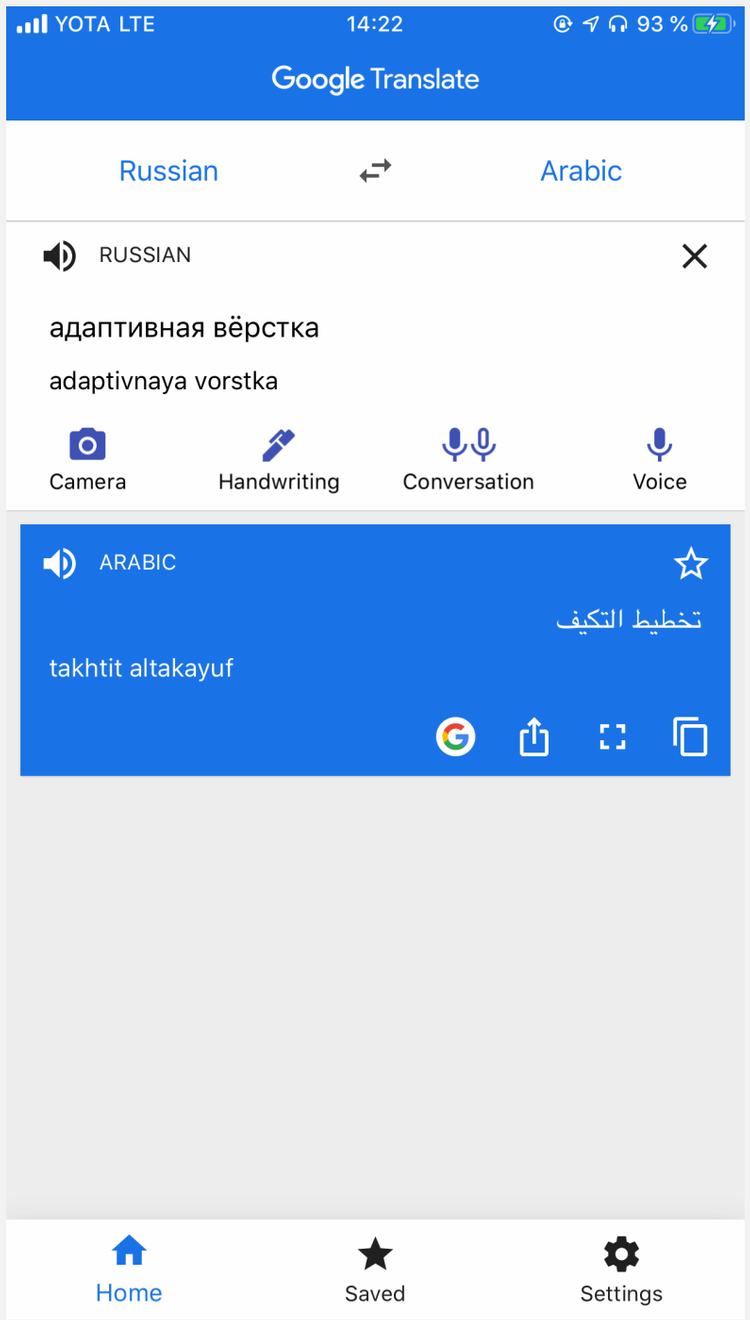
```
let layer = CAShapeLayer()
layer.path = UIBezierPath(roundedRect: CGRect(x: 64, y: 64, width: 160,
height: 160), cornerRadius: 50).cgPath
layer.fillColor = UIColor.red.cgColor
view.layer.addSublayer(layer)
```

Adaptivity

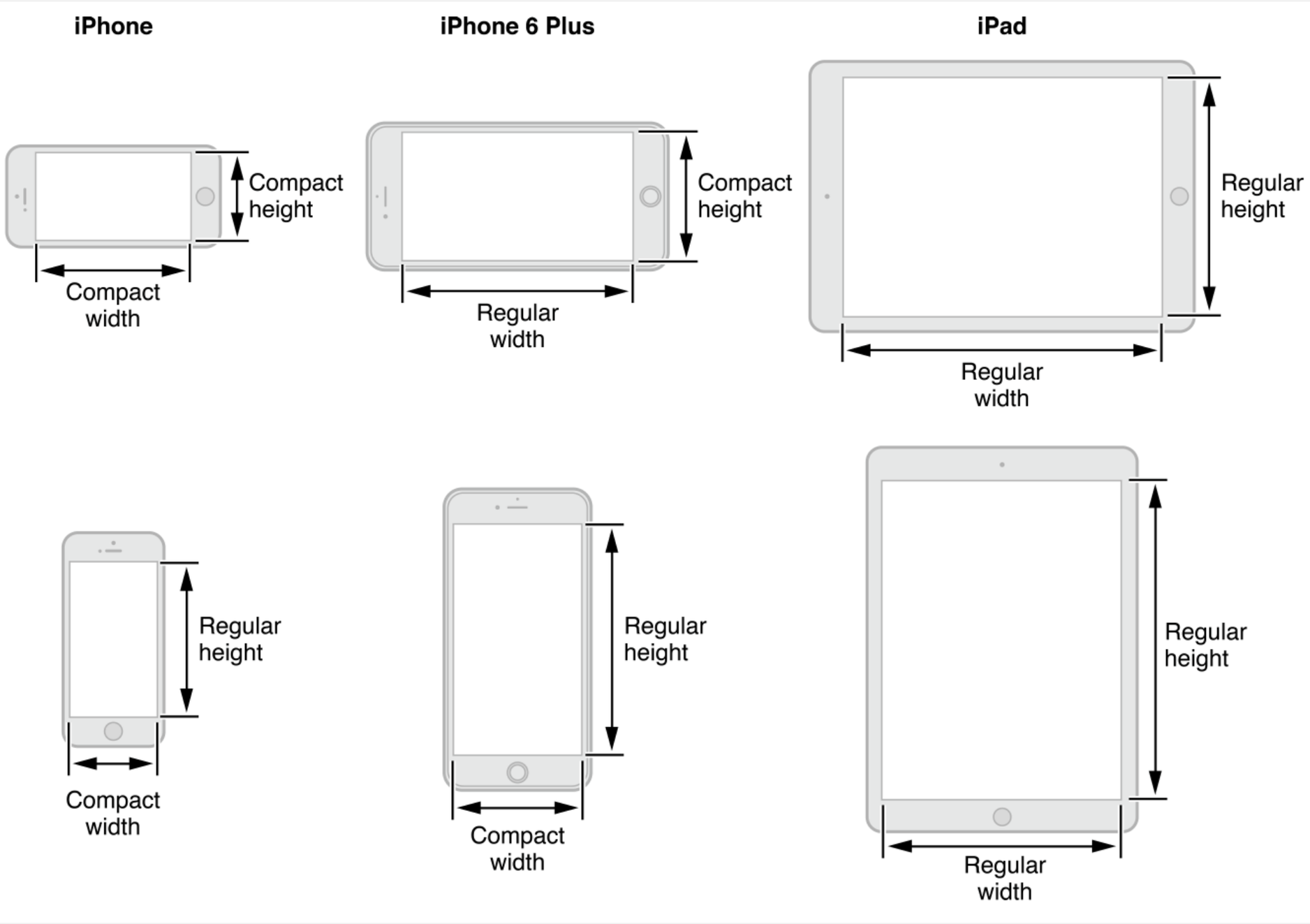
- Разные размеры экранов
- Разные ориентации (вертикальная, горизонтальная)
- Split View (iPad)
- Меняющийся текст
- Layout direction в зависимости от страны

Adaptivity


YOTA LTE
14:22
93 %
Google Translate
Russian
English
RUSSIAN
адаптивная вёрстка
adaptivnaya vorstka
Camera
Handwriting
Conversation
Voice
ENGLISH
Adaptive layout
Home
Saved
Settings


YOTA LTE
14:22
93 %
Google Translate
Russian
Arabic
RUSSIAN
адаптивная вёрстка
adaptivnaya vorstka
Camera
Handwriting
Conversation
Voice
ARABIC
تخطيط التكيف
takhtit altakayuf
Home
Saved
Settings

Size Classes



Size Classes

- Доступно в Storyboard
- В зависимости от size class можно:
  - добавлять/удалять constraint
  - добавлять/удалять view
  - менять атрибуты (например, шрифт)

- В коде

```
if (self.view.traitCollection.horizontalSizeClass == .compact) {
    // do something
}
```

Ссылки

- https://www.raywenderlich.com/443-auto-layout-tutorial-in-ios-11-getting-started - Autolayout
- https://developer.apple.com/design/human-interface-guidelines/ios/visual-design/adaptivity-and-layout/ - Adaptivity
- https://developer.apple.com/documentation/uikit/nslayoutanchor - Anchors
- https://developer.apple.com/ios/human-interface-guidelines/ - Apple Human Interface Guidelines
- https://www.objc.io/issues/3-views/custom-controls/ - о кастомных control'ах
- http://nshipster.com/ibinspectable-ibdesignable/ - IBDesignable / IBInspectable (редко используется, для фанатов Interface Builder'а)
- https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/AutolayoutPG/VisualFormatLanguage.html - visual format (для auto layout)
- https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/AutolayoutPG/Size-ClassSpecificLayout.html - size classes