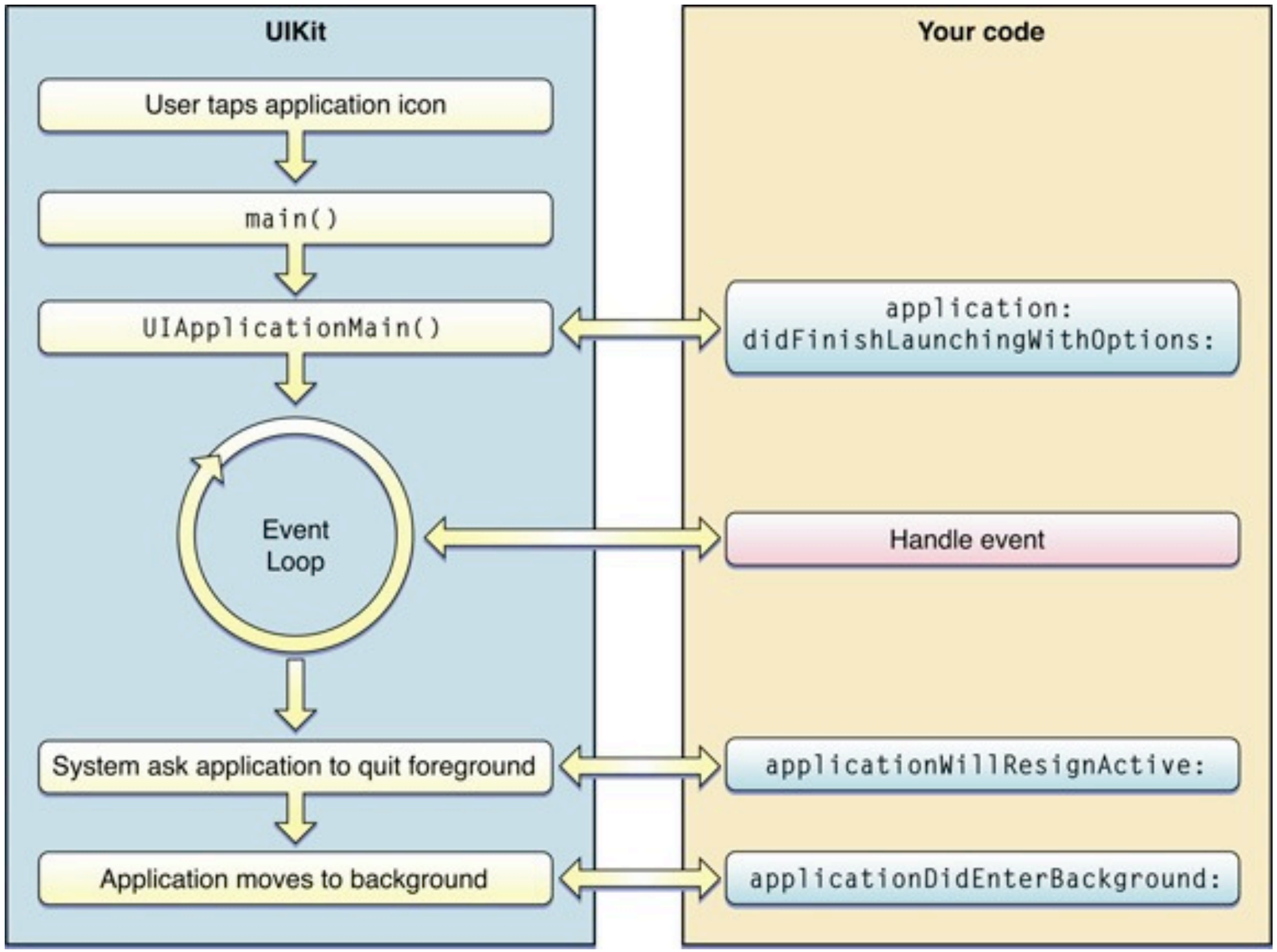
CS 646 iPad/iPhone Application Development
Fall Semester, 2014
Doc 9 Textfields, Keyboard & Some Graphics
Oct 2, 2014

App life cycle



Thursday, October 2, 14

diagram from "iPhone Application Programming Guide"

Starting points for your app code

App Delegate

```
- (BOOL)application:(UIApplication *)application
        didFinishLaunchingWithOptions:(NSDictionary *) launchOptions
```

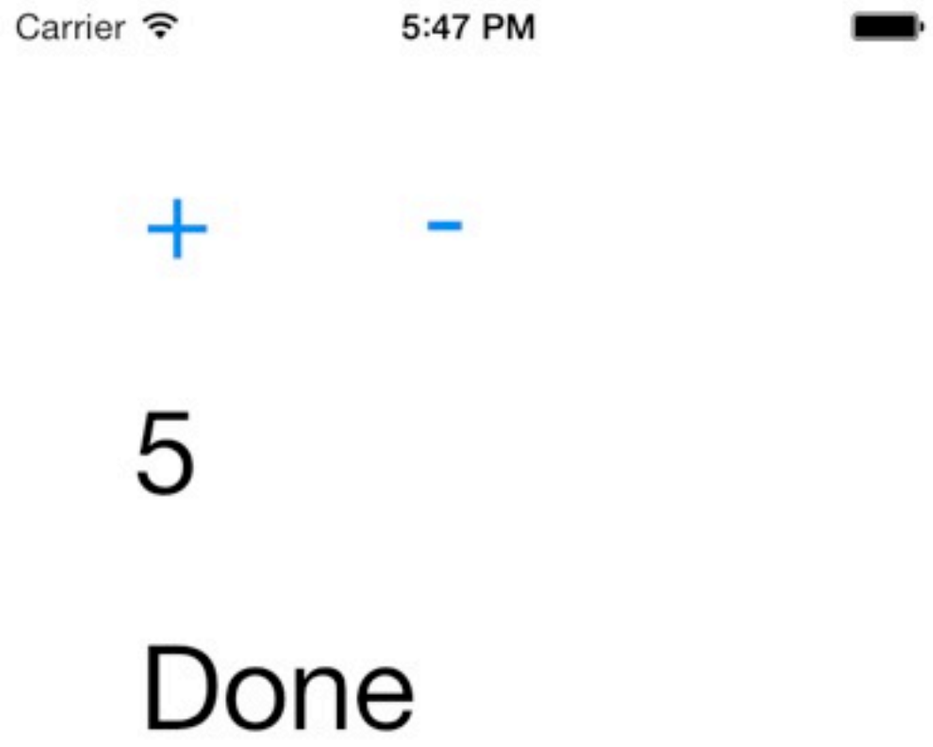
Controller

```
- (void) viewDidLoad
```

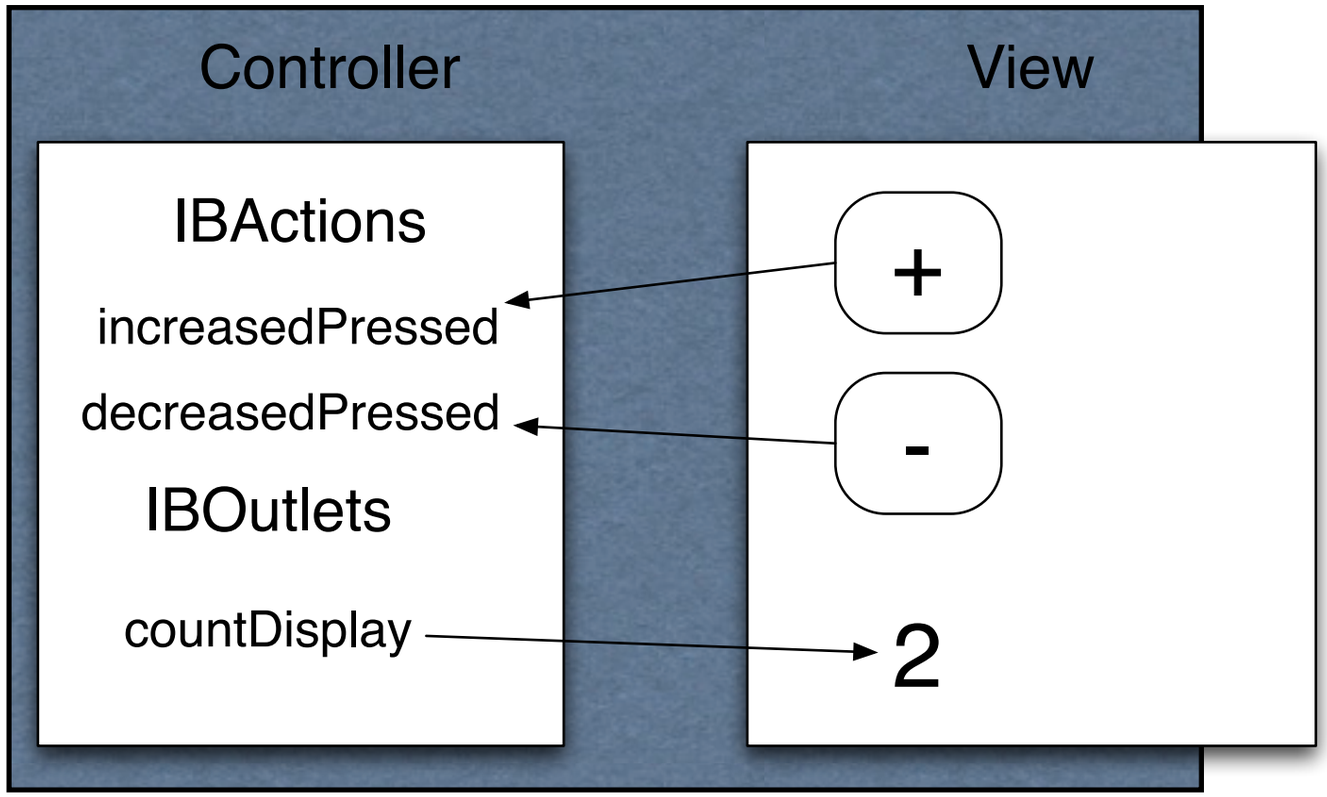

There are more starting points, but we will cover them later.

Creating Views in code

When count >=3
Create & display "Done"



Declaring connections in Code




```
@interface CounterViewController : UIViewController {
      int * count;
}
@property (nonatomic, retain) IBOutlet UILabel * countDisplay;
@property (strong, nonatomic) UILabel * doneLabel;

- (IBAction) increasePressed;
- (IBAction) decreasePressed;
@end
```

CounterViewController.m

```
- (IBAction)increasedPressed {
    count++;
    [self updateDisplay];
    if (count >= 3) {
            [self displayDoneLabel];
      }
}
```

```
- (void) displayDoneLabel {
      if (doneLabel == nil) {
            [self createDoneLabel];
      }
      else {
            doneLabel.hidden = NO;
      }
}
```

CounterViewController.m

```
- (void) createDoneLabel {
    CGRect frame = CGRectMake(20, 220, 150, 30);
    [self setDoneLabel: [[UILabel alloc] initWithFrame:frame]];
    self.doneLabel.textAlignment = NSTextAlignmentCenter;
    self.doneLabel.font = [UIFont systemFontOfSize:40];
    self.doneLabel.text = @"Done";
    [self.view addSubview:self.doneLabel];
}
```

CounterViewController.m

```
- (IBAction)decreasedPressed {
    count--;
    [self updateDisplay];
    if (count < 3) {
        self.doneLabel.hidden = YES;
    }
}
```

self.doneLabel starts as nil

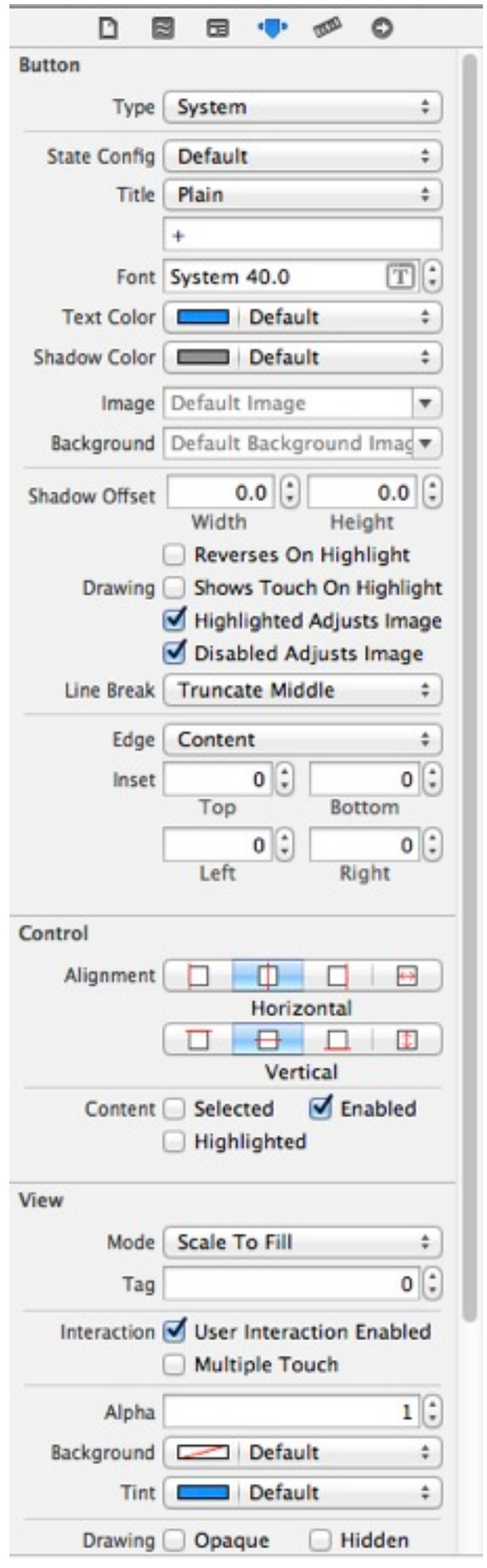
Important Points

Hiding & Showing Widgets

```
doneLabel.hidden = NO;
doneLabel.hidden = YES;
```

Properties of widgets can be set in

- Xcode
- In your code
- You can change location of widget



Properties of Widgets

The documentation

Tasks

Creating Buttons

+ buttonWithType:

Configuring the Button Title

titleLabel *property*
– titleForState:
– setTitle:forState:
– attributedTitleForState:
– setAttributedTitle:forState:
– titleColorForState:
– setTitleColor:forState:
– titleShadowColorForState:
– setTitleShadowColor:forState:
reversesTitleShadowWhenHighlighted *property*

Configuring Button Presentation

adjustsImageWhenHighlighted *property*
adjustsImageWhenDisabled *property*
showsTouchWhenHighlighted *property*
– backgroundImageForState:
– imageForState:
– setBackgroundImage:forState:
– setImage:forState:
tintColor *property*

Configuring Edge Insets

contentEdgeInsets *property*
titleEdgeInsets *property*
imageEdgeInsets *property*

Getting the Current State

buttonType *property*
currentTitle *property*
currentAttributedTitle *property*
currentTitleColor *property*
currentTitleShadowColor *property*
currentImage *property*
currentBackgroundImage *property*

Some Graphics

Structures & Functions

CGPoint
location in space: { x , y }

CGSize
dimensions: { width , height }

CGRect
location and dimension: { origin , size }

CGPointMake (x, y)

CGSizeMake (width, height)

CGRectMake (x, y, width, height)

Swift & Structures

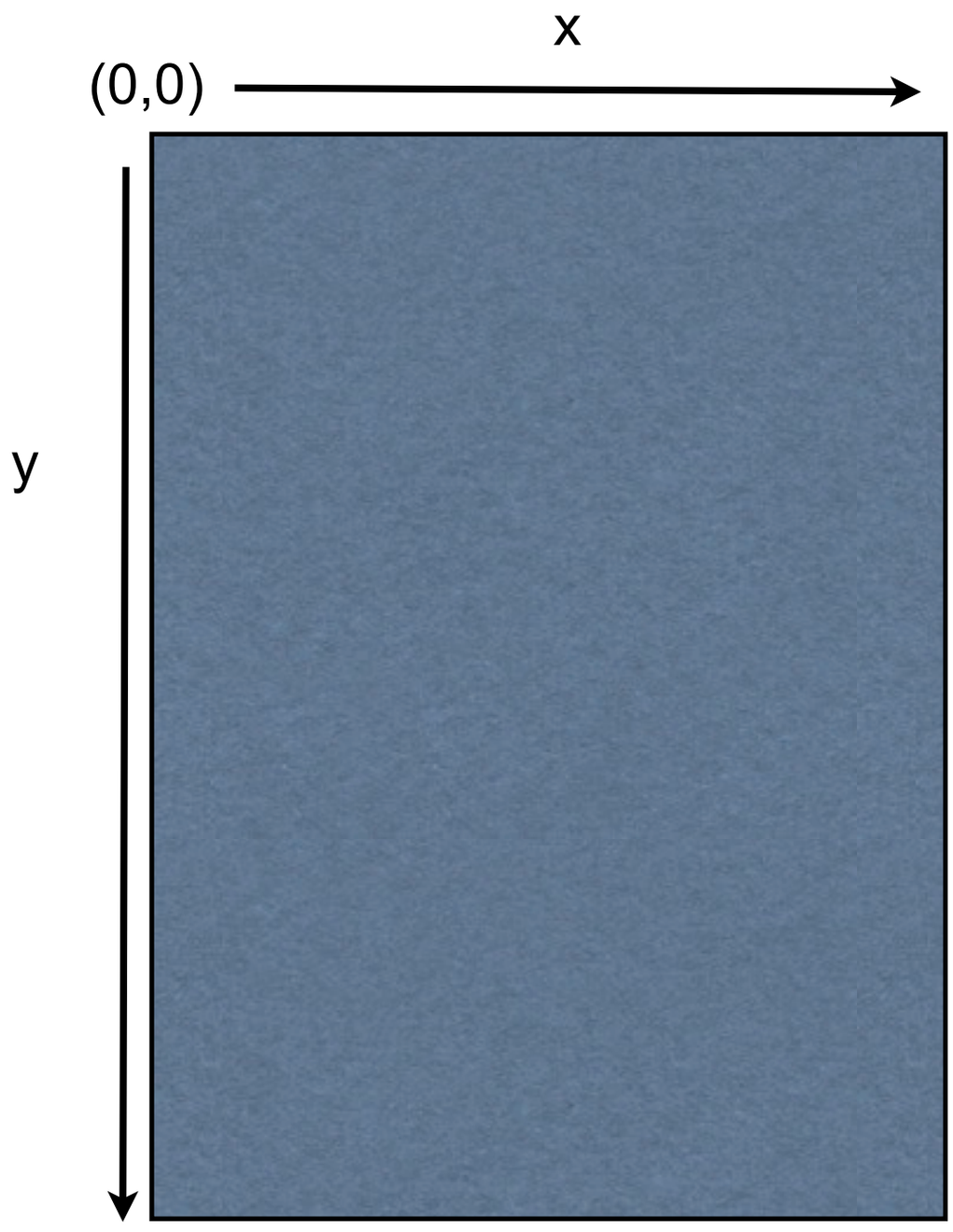
CGPointMake(10, 20)

CGPoint(x:10,y:20)

Both work

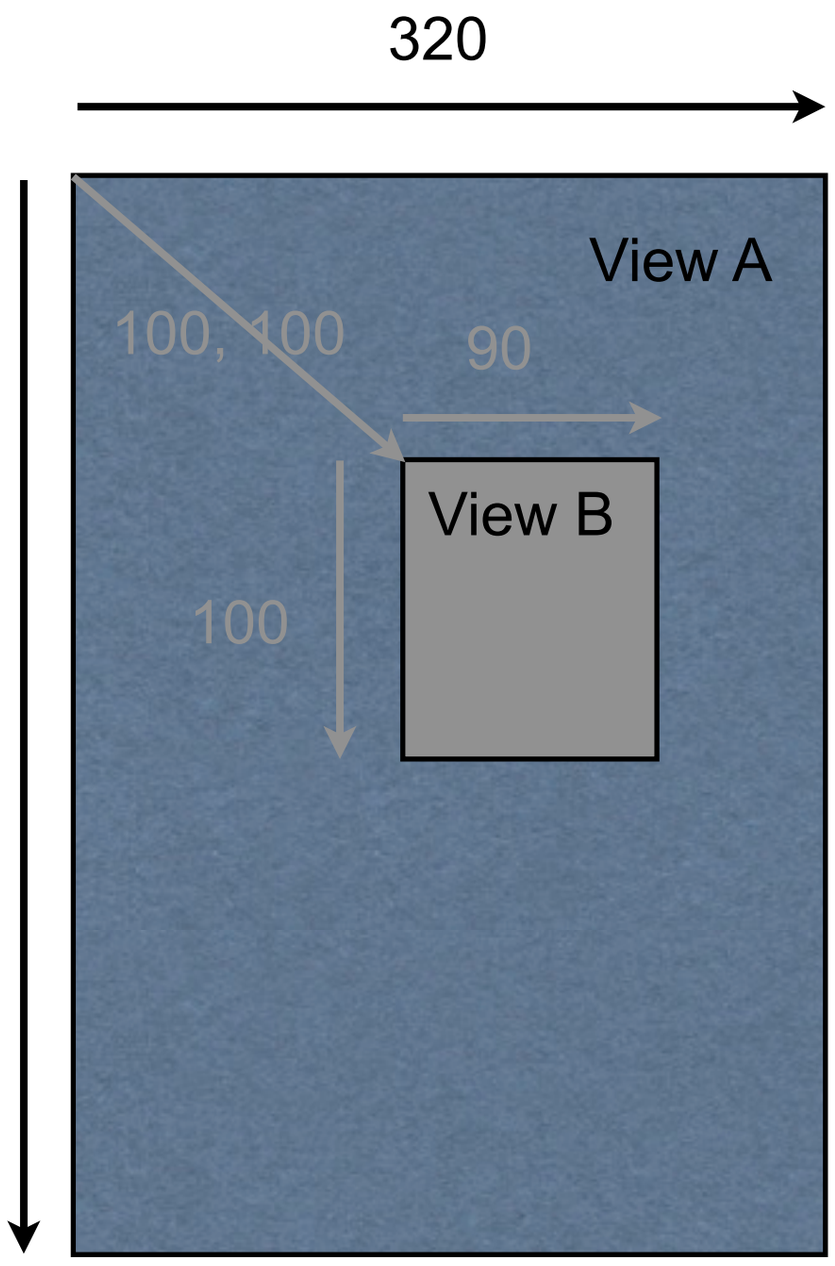
Later more Swift like

Quartz 2D Coordinates



Frame & Bounds

Both give location & size of a View



Frame

In superview coordinates

Computed

Bounds

In local coordinates

View A frame

size: 320 x 480

origin: 0, 0

View A bounds

size: 320 x 480

origin: 0, 0

View B frame

size: 90 x 100

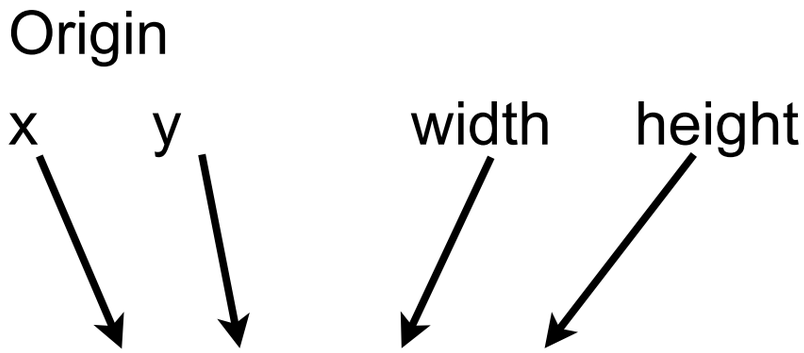
origin: 100, 100

View B bounds

size: 90 x 100

origin: 0, 0

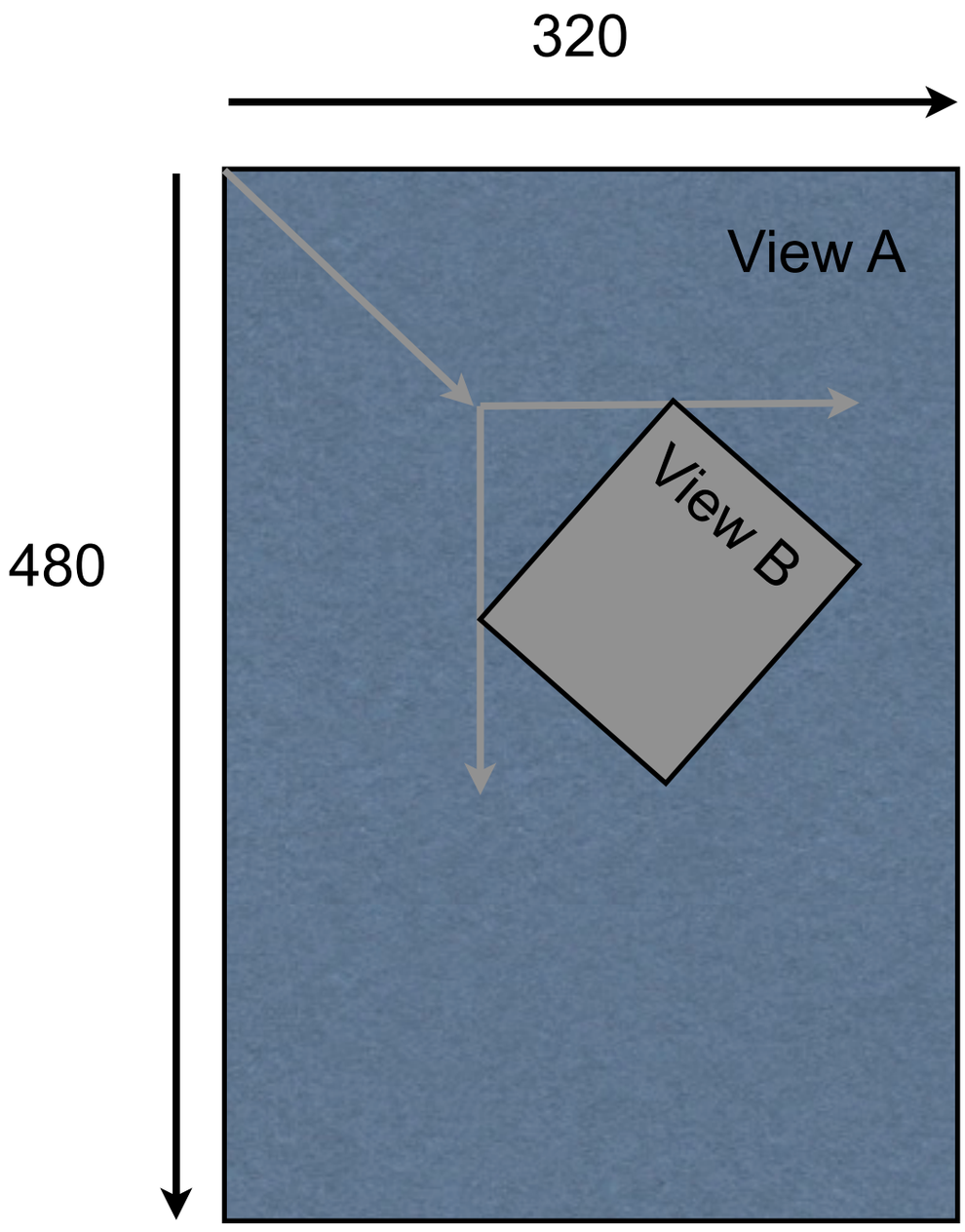
Label in Code




```
- (void) createDoneLabel {
    CGRect frame = CGRectMake(20, 220, 150, 30);
    [self setDoneLabel: [[UILabel alloc] initWithFrame:frame]];
    self.doneLabel.textAlignment = NSTextAlignmentCenter;
    self.doneLabel.font = [UIFont systemFontOfSize:40];
    self.doneLabel.text = @"Done";
    [self.view addSubview:self.doneLabel];
}
```

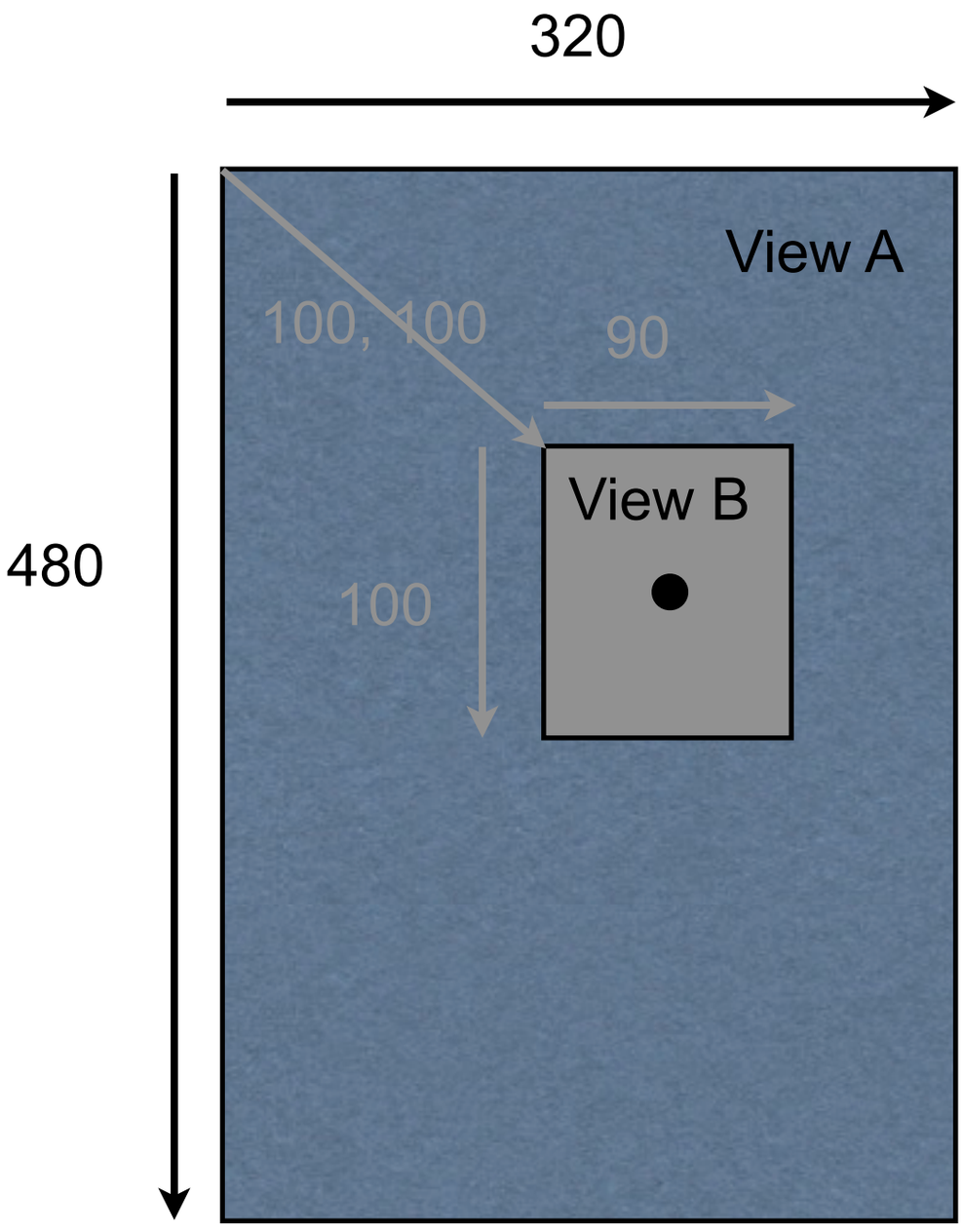
Frame

Smallest rectangle that contains view



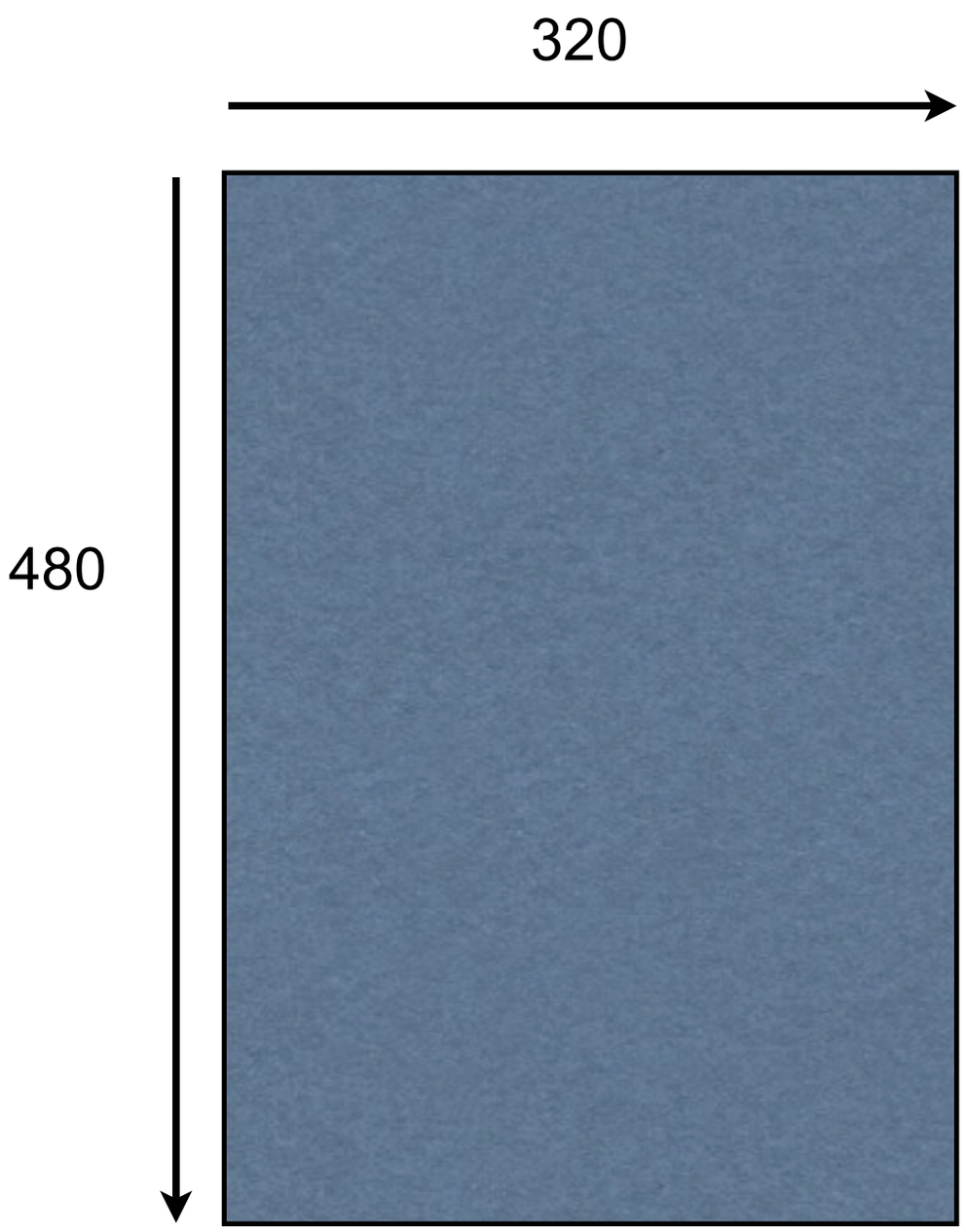
Center

Given in superview's coordinates



View B Center
145, 150

Points Verse Graphics



iPhone/iPod Touch Screens

- 320 points by 480 points
- 320 points by 568 points
  - iPhone 5, 5s, 5c
- 375 points by 667 points
  - iPhone 6
- 414 points by 736
  - iPhone 6 Plus

iPad screen

- iPad 2, Retina Display, iPad Mini
- 768 points by 1024 points

You draw using points

Points are floats

Points are mapped to pixels

Some Widget Properties

Moving Widgets

Assume that widget is a IBOutlet property connected to some UI widget

self.widget.center = CGPointMake(self. widget.center.x + 10, self. widget.center.y);

If the widget has constraints this does not work

Colors

In UIColor class

Some Predefined Colors

+ blackColor
+ darkGrayColor
+ lightGrayColor
+ whiteColor
+ grayColor
+ redColor
+ greenColor

Some System Colors

+ lightTextColor
+ darkTextColor
+ groupTableViewBackgroundColor

Creating colors

+ colorWithHue:saturation:brightness:alpha:
+ colorWithRed:green:blue:alpha:

Parameters 0.0 to 1.0

Changing Background Color

Assume that widget is a IBOutlet property connected to some UI widget

```
UIColor * newColor = [UIColor colorWithRed: 1.0 green:0.5 blue: 0 alpha:1];
[self.widget setBackgroundColor: newColor];
```

Changing Background Color - Swift

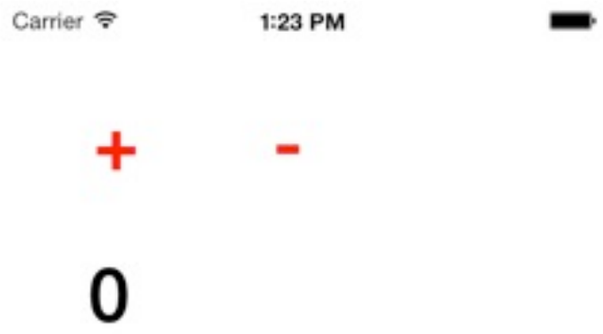
Assume that widget is a IBOutlet property connected to some UI widget

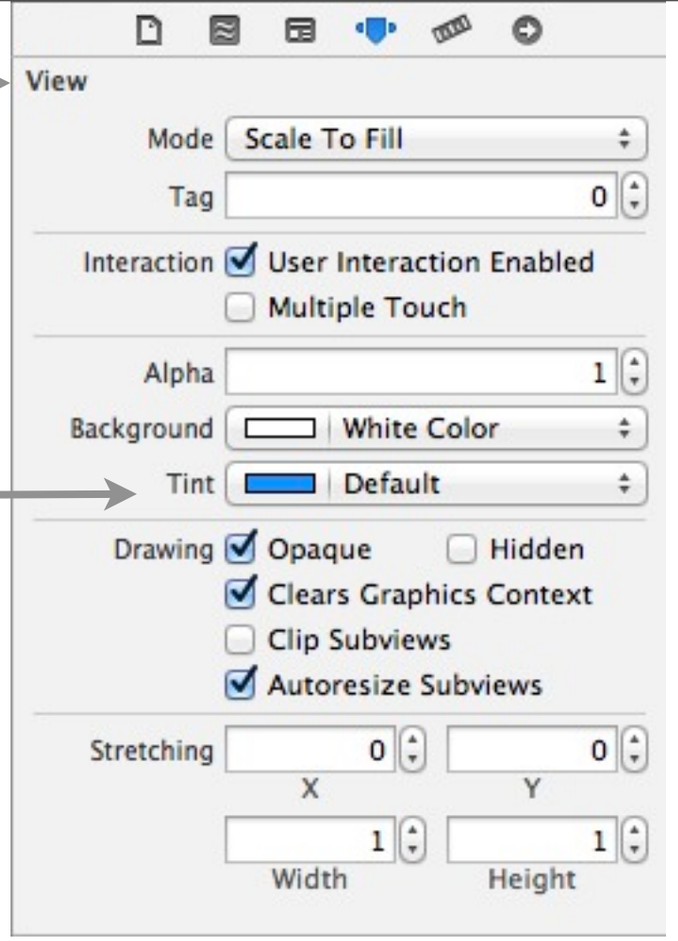
```
let newColor = UIColor(red: 1.0, green: 0.5, blue: 0, alpha: 1)
widget.backgroundColor = newColor
```

iOS 7 - TintColor

Sets the default color for
buttons, some other controls

Set in App delegate or Top level view





Views & TextField

Views, Controls & Controllers

Views

- Subclass of UIView
- Label, Web view, Toolbar, etc

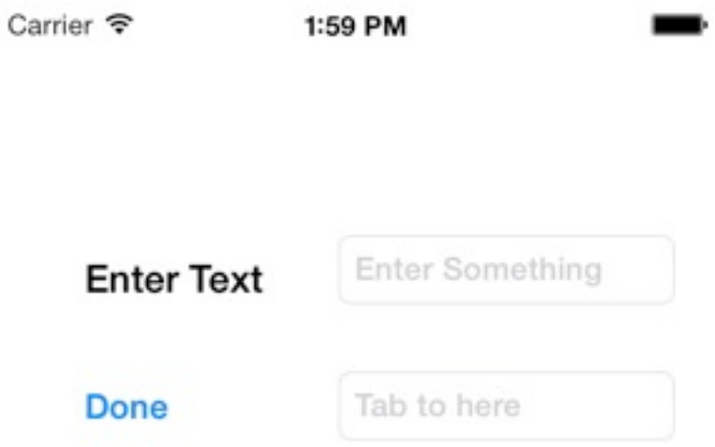
Controls (types of views)

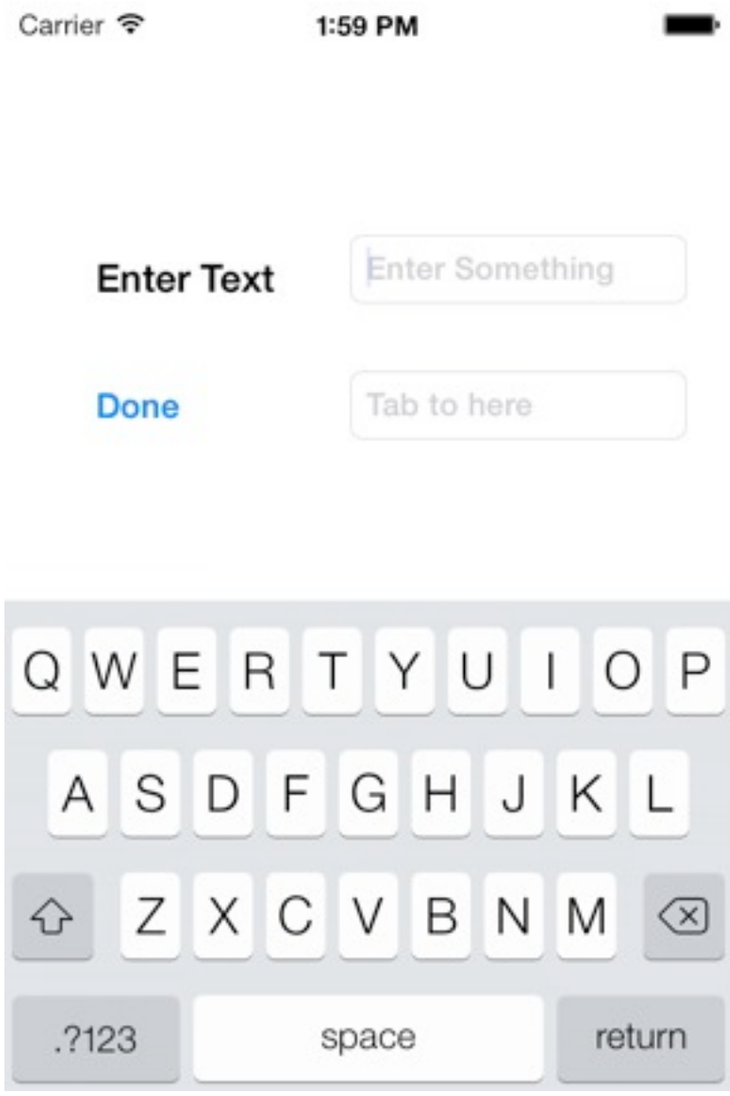
- Subclass of UIControl which is subclass of UIView
- Views that can call methods on controllers
- Button, date picker, slider, text field, switch

Controllers

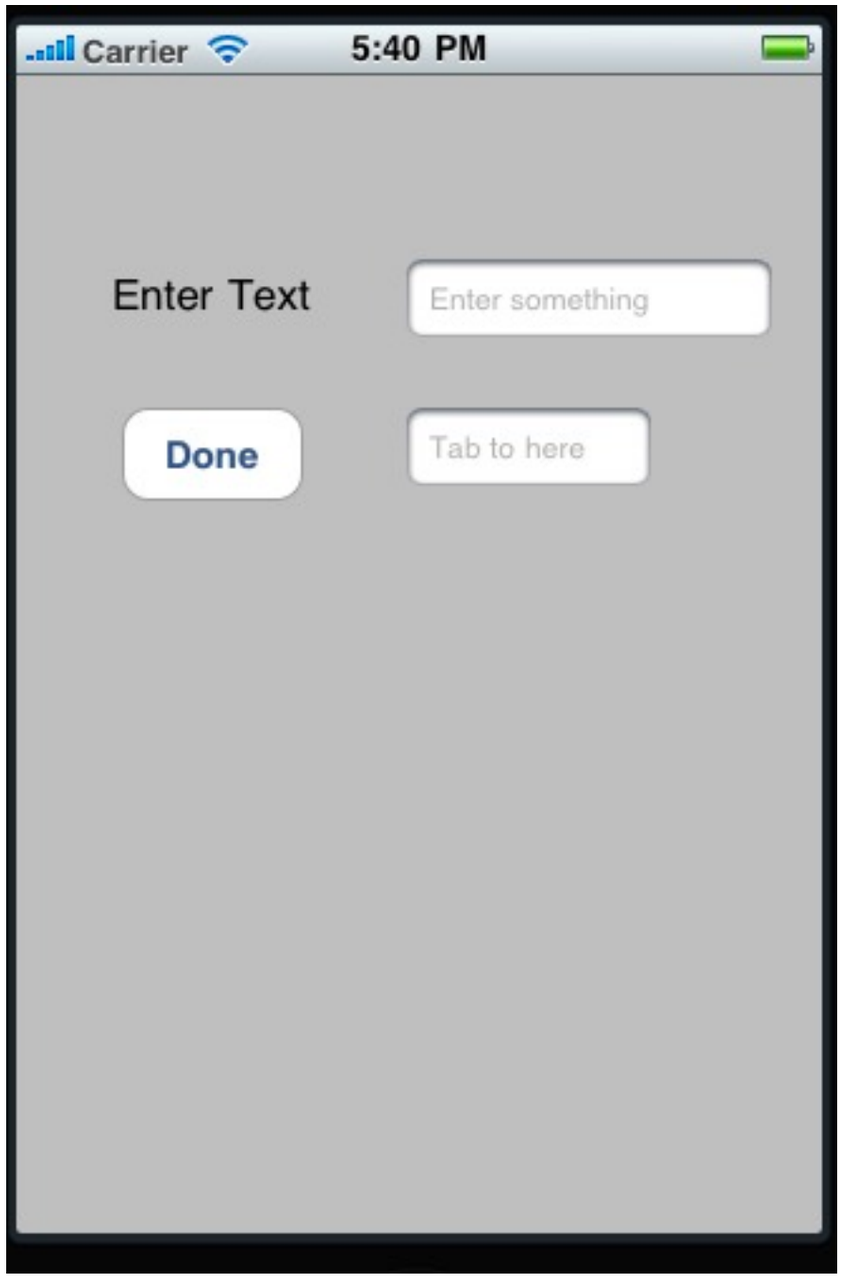
- Subclass of UIViewController

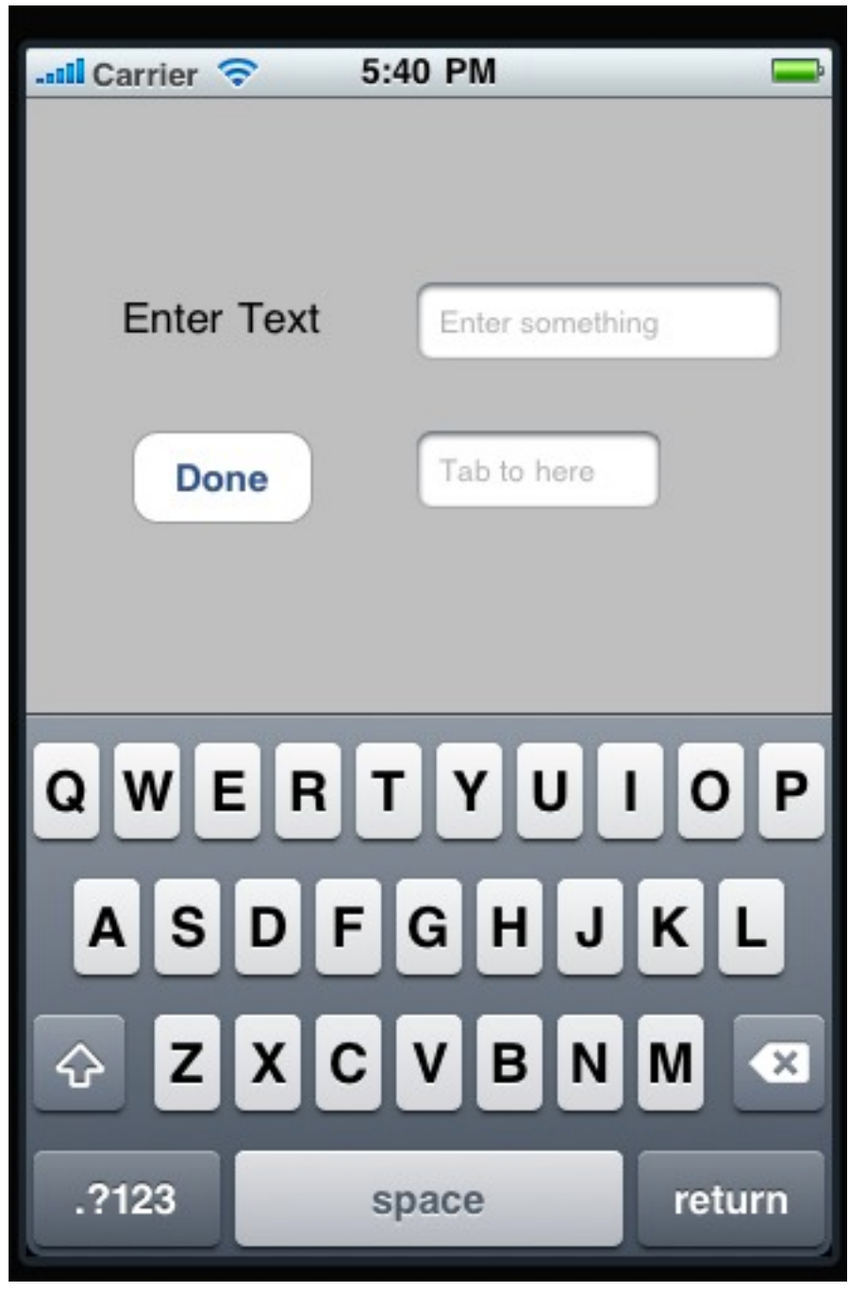
TextField Example - iOS 7 Version



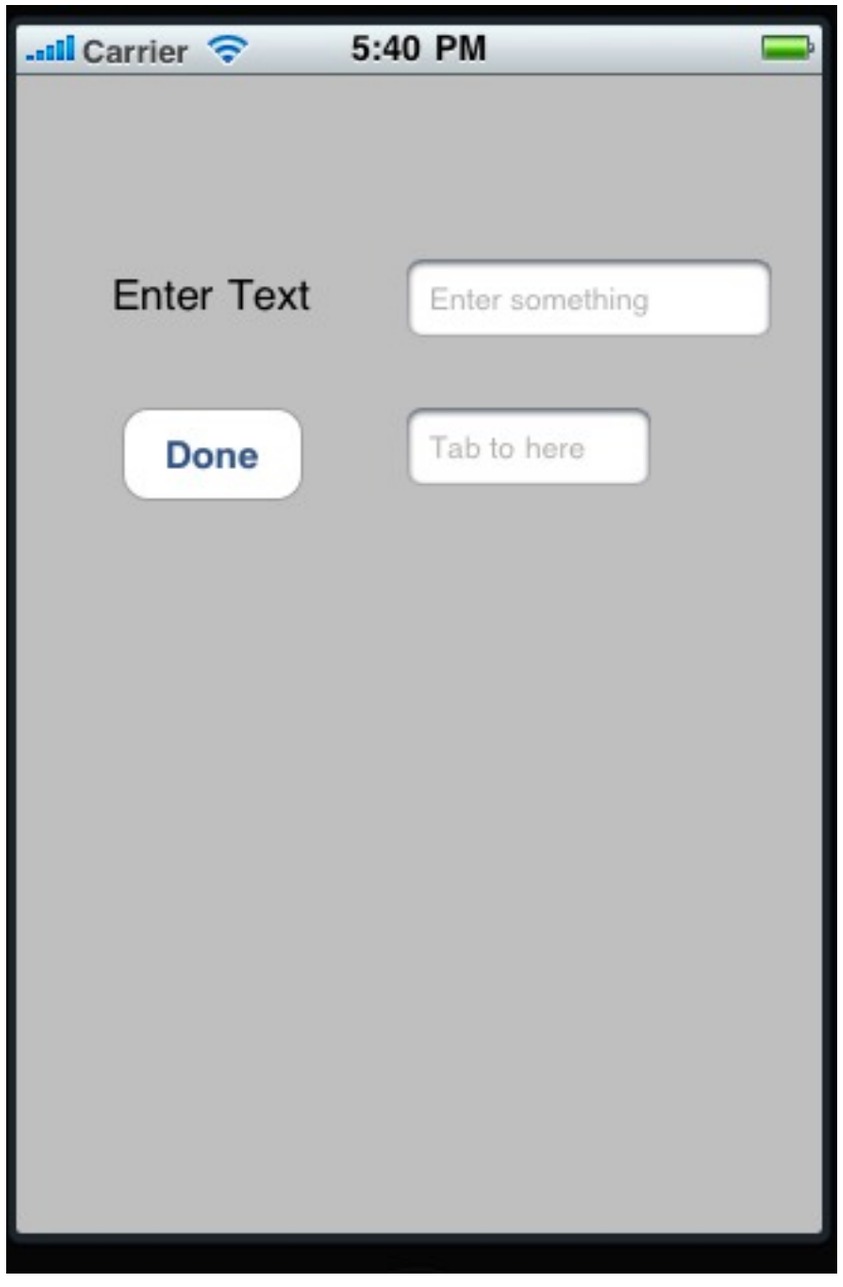


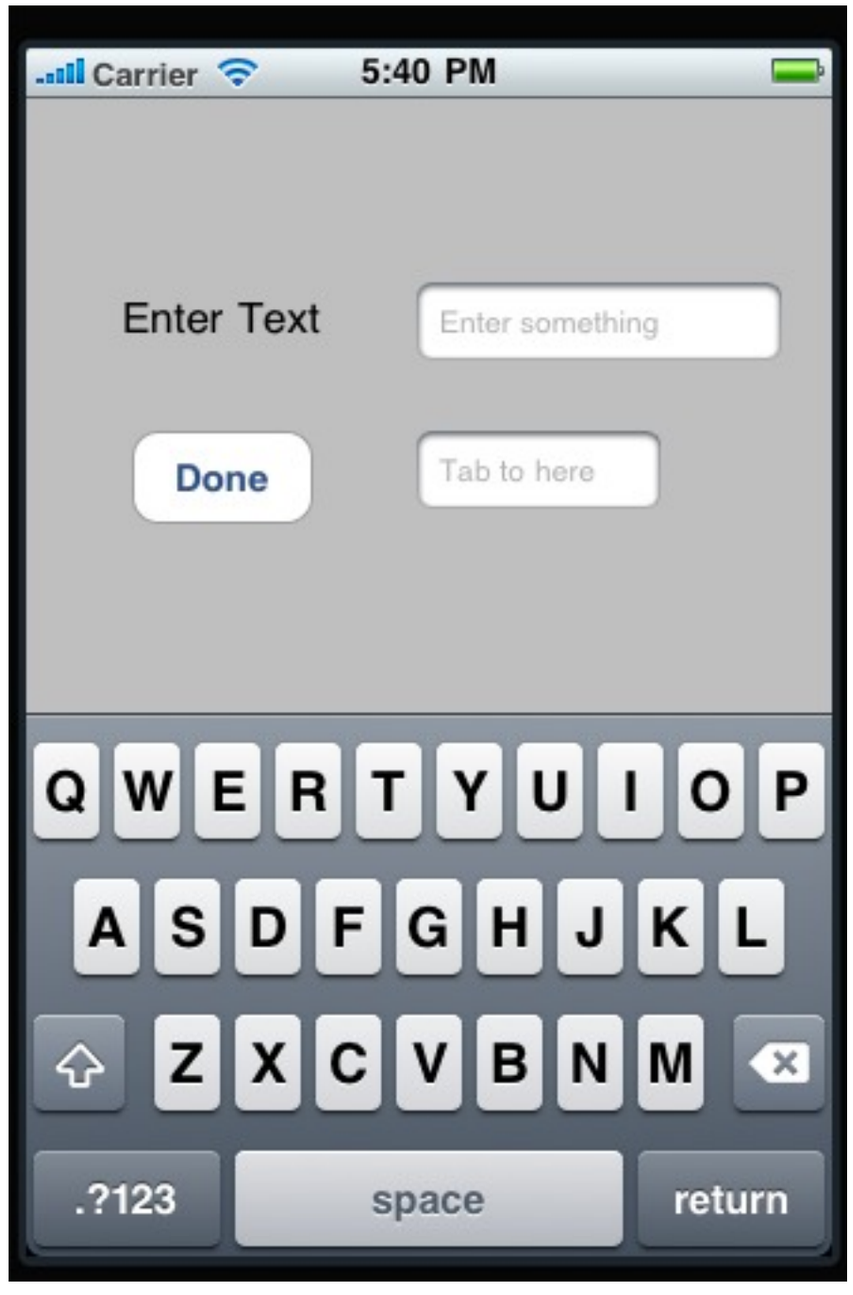
TextField Example - iOS 6 Version





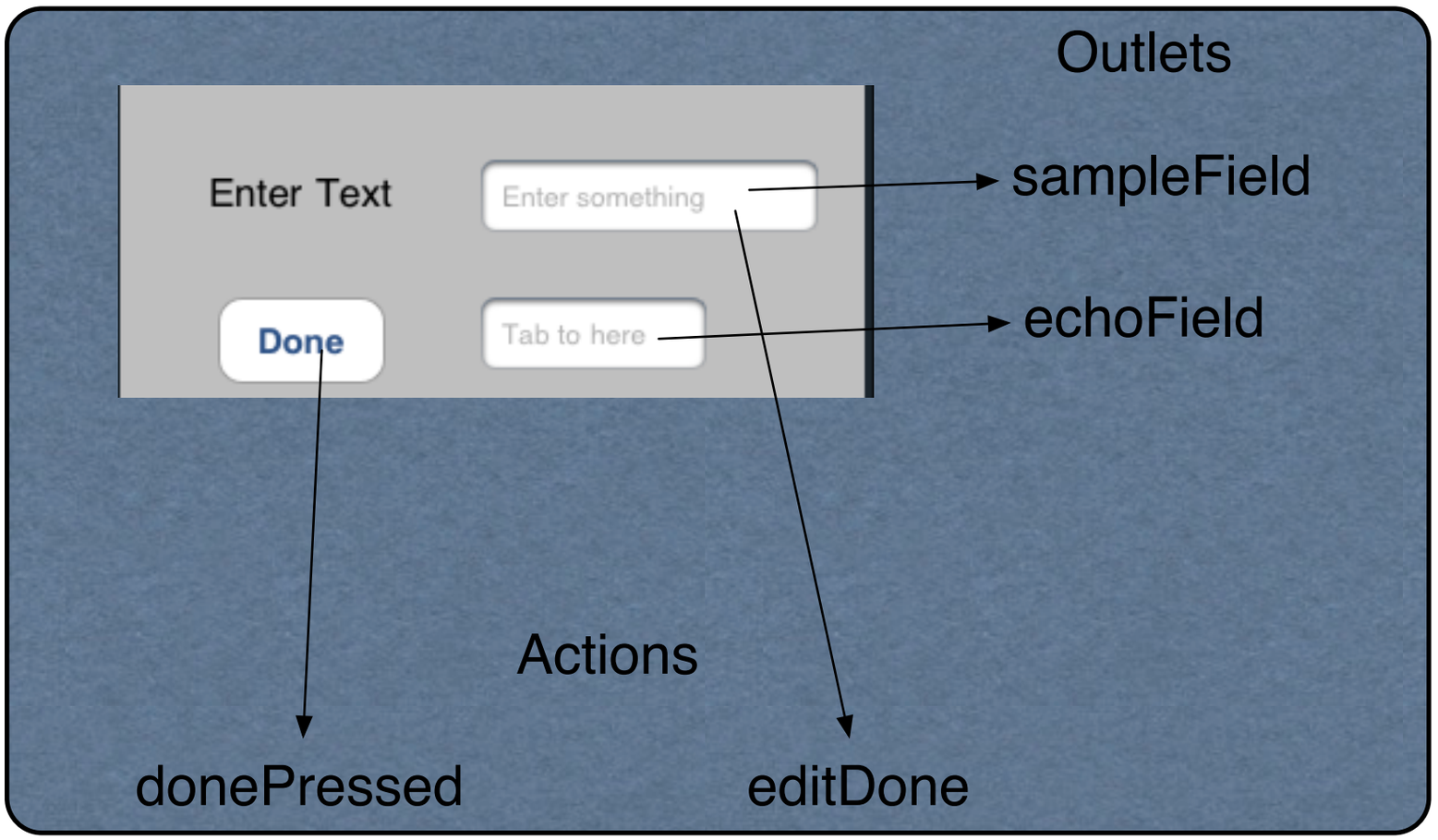
Running iOS 6 Version on iOS6





Program has to be recompiled in Xcode 5 for iOS 7 to get iOS7 look

The connections




```
@interface RWTextFieldExampleController : UIViewController
@property (strong, nonatomic) IBOutlet UITextField *sampleField;
@property (strong, nonatomic) IBOutlet UITextField *echoField;
- (IBAction)donePressed;
- (IBAction)editDidEnd;
@end
```


editDone is triggered by the "editing did end" action on the sampleField

Actions

```
- (IBAction)editDidEnd {
    [self hideKeyboard];
    self.echoField.text = self.sampleField.text;
}

- (IBAction)donePressed {
    [self editDidEnd];
}
```

How to know when user is done editing?

TextField events

Edit Did Begin

When field gets focus

Edit Changed

When any change is made

Characters added/removed, cursor changes position

Edit Did End

When focus leaves field

Tab to another field

Did End On Exit

When user tabs on "return" or "done" key on keyboard

How to know when user is done editing?

Hard to tell by just text field events

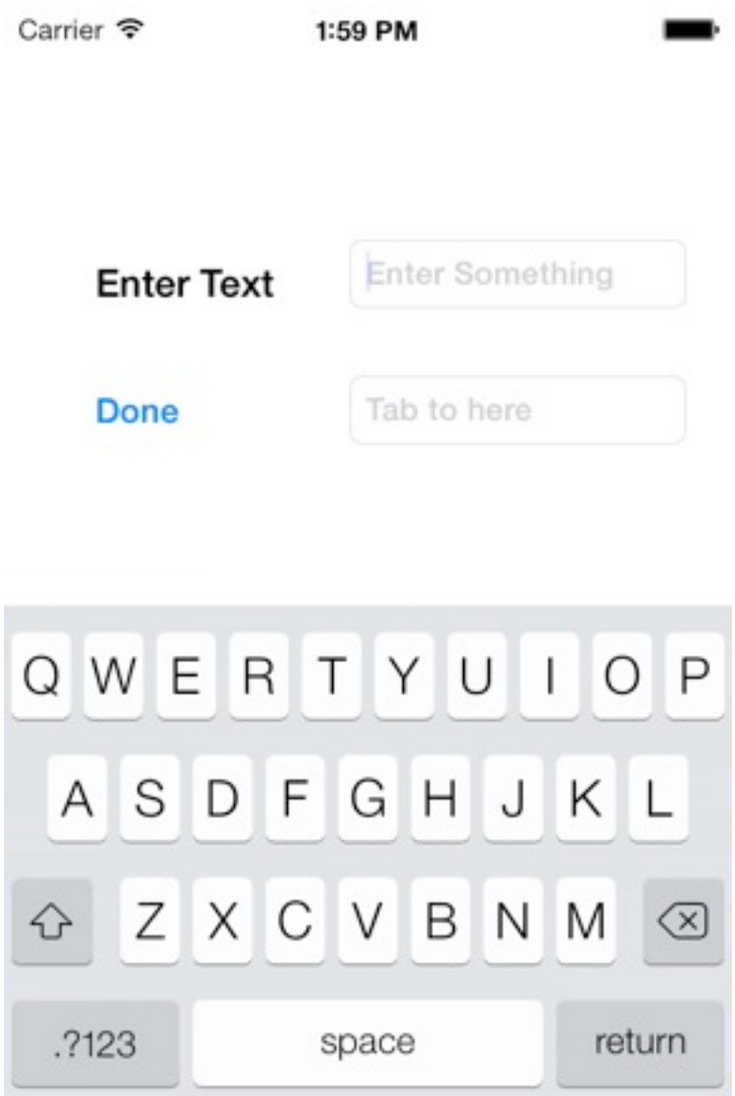
Provide some other way for user to indicate done

Keyboard

When textfield gets focus keyboard appears

Your code has to hide it

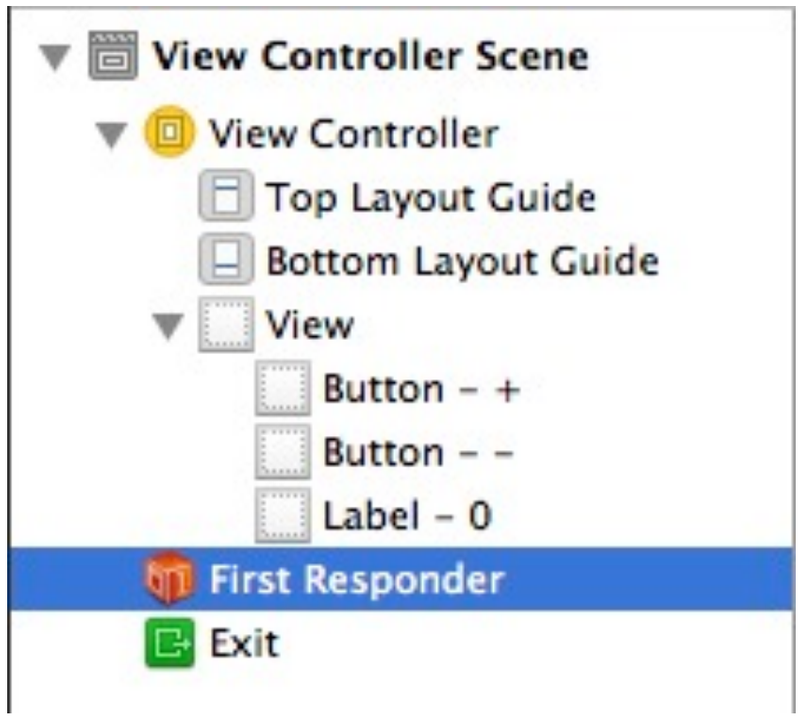
Multiple keyboards available



First Responder

UI element the user in interacting with

Window currently the focus for user events



Hiding the Keyboard

When text field is notified that should stop being first responder

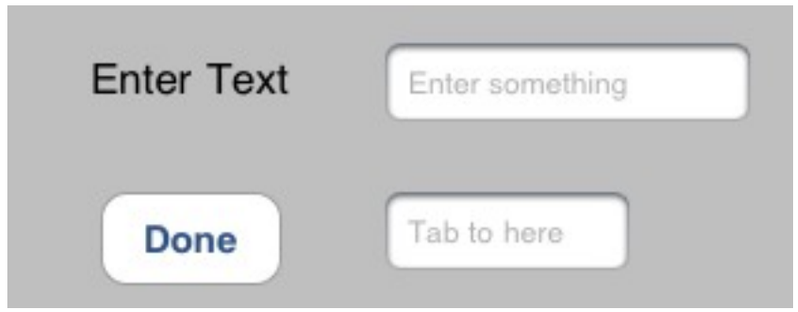
Send resignFirstResponder to active text field

Hiding the Keyboard

```
- (IBAction)editDidEnd {
    [self hideKeyboard];
    self.echoField.text = self.sampleField.text;
}

- (IBAction)donePressed {
    [self editDidEnd];
}

- (void) hideKeyboard {
    [self.sampleField resignFirstResponder];
      [self.echoField resignFirstResponder];
}
```



Here is the standard solution

Do we have to list all text fields?

UIView class has method "firstResponder"

```
- (IBAction) donePressed {
      [self editDone];
      UIView * firstResponder = [[self view] firstResponder];
      if( [firstResponder isKindOfClass:[UITextField class]] )
            [firstResponder resignFirstResponder];
}
```

But ...

firstResponder is not in public API

Apple rejects apps that use non public API methods

Question for thought

How does one find out about methods not in public API?

New UIView Method

```
#import <Foundation/Foundation.h>
@interface UIView (FirstResponder)
- (UIView  *) getFirstResponder;
@end
```

```
@implementation UIView (FirstResponder)
- (UIView *) getFirstResponder
{
    if (self.isFirstResponder) {
        return self;
    }

    for (UIView *subView in self.subviews) {
        UIView *firstResponder = [subView getFirstResponder];
        if (firstResponder != nil) {
            return firstResponder;
        }
    }
    return nil;
}
@end
```


Adapted from http://stackoverflow.com/questions/949806/is-there-a-way-to-detect-what-uiview-is-currently-visible

Legal Solution

```
- (IBAction) donePressed {
        [self editDone];
        UIView * firstResponder = [[self view] getFirstResponder];
        if( [firstResponder isKindOfClass:[UITextField class]] )
                [firstResponder resignFirstResponder];
}
```

```
- (IBAction) editDone {
        echoField.text = sampleField.text;
}
```


But people just list all the text fields

Hiding Keyboard

Common to hide keyboard when user taps on background

First Solution

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
      [self hideKeyboard];
}

- (IBAction)donePressed {
    [self editDidEnd];
}

- (IBAction)editDidEnd {
    [self hideKeyboard];
    self.echoField.text = self.sampleField.text;
}

- (void) hideKeyboard {
    [self.sampleField resignFirstResponder];
      [self.echoField resignFirstResponder];
}
```


add touchesBegan:withEvent: method to your controller .m file

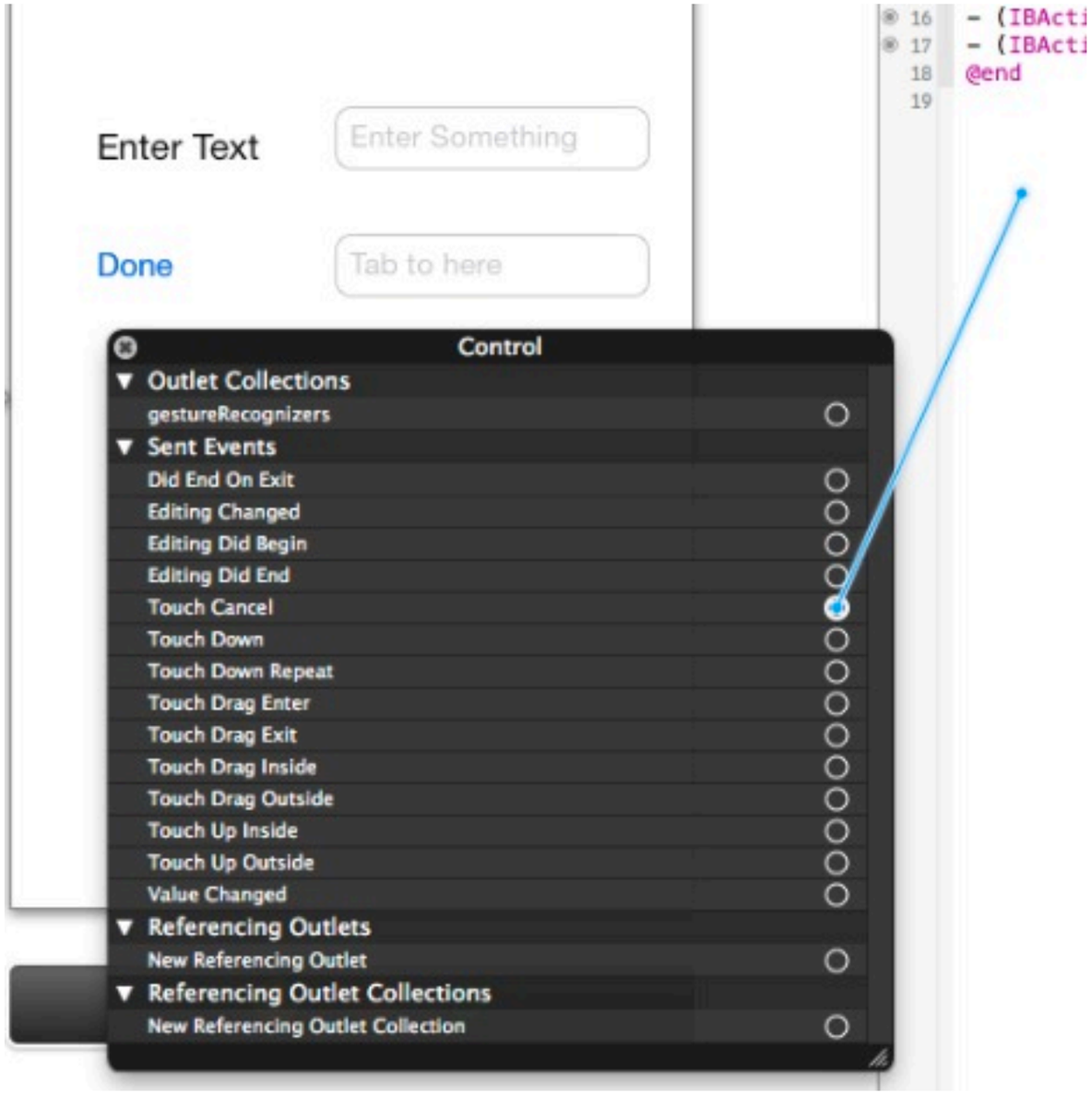
Second Solution

Change the class of your View to UIControl

UIControl has touch actions you can use to trigger keyboard hiding

But lose ability to get some touch events

Tap Background



You have to change the class of the View to be UIControl

Demo

Some Keyboard Types - iOS 6-

UIKeyboardTypeASCIICapable

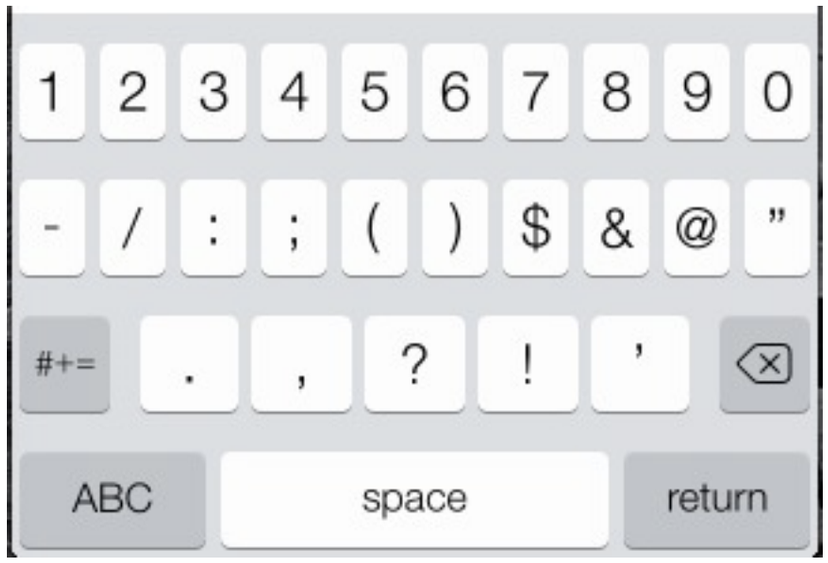
Numbers

URL

Number Pad

Some Keyboard Types - iOS 7

UIKeyboardTypeASCIICapable

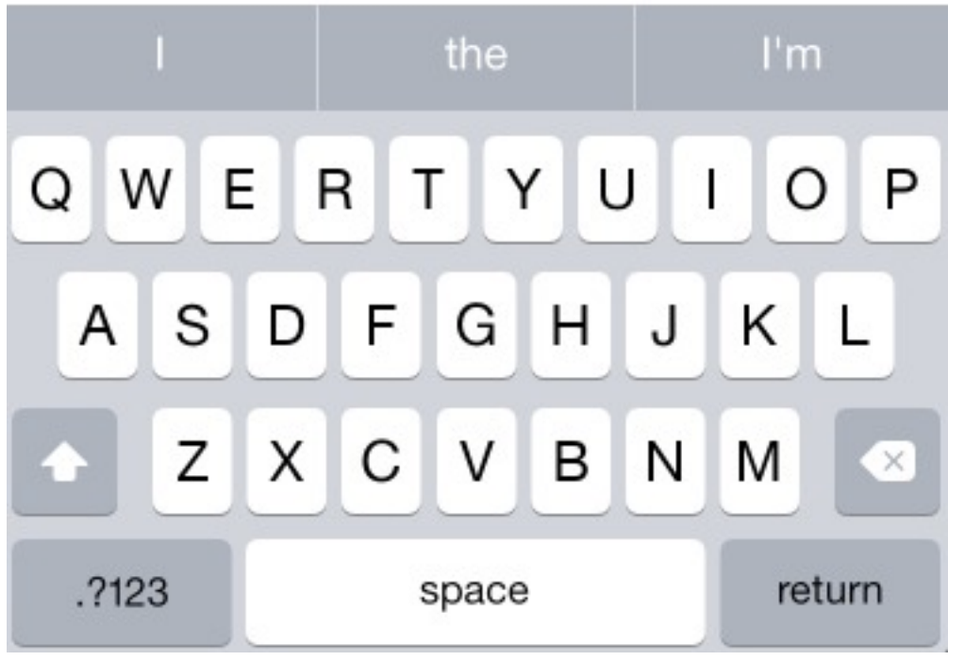


Numbers

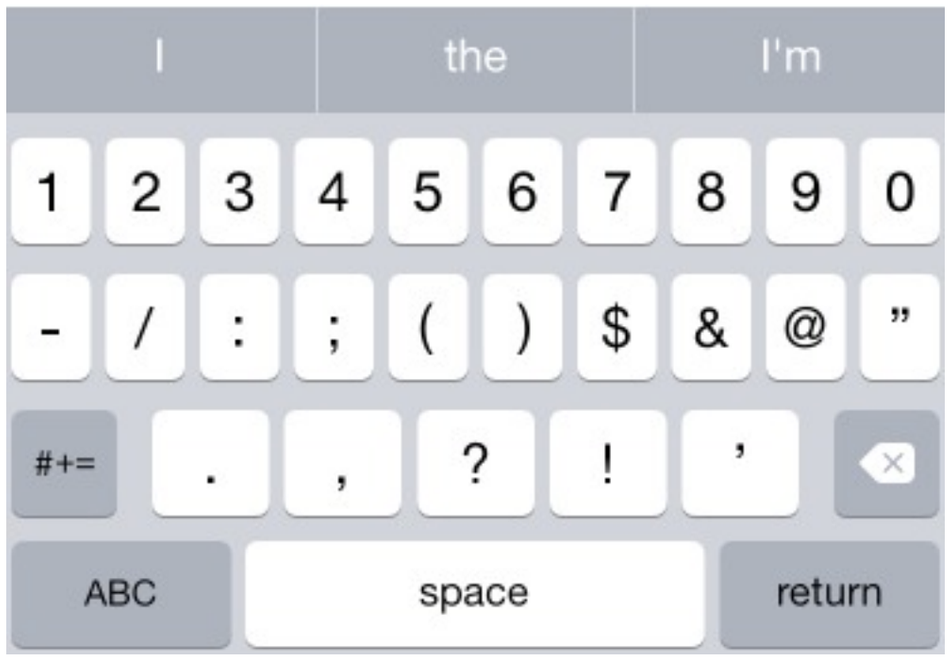
URL

Number Pad

Some Keyboard Types - iOS 8



UIKeyboardTypeASCIICapable

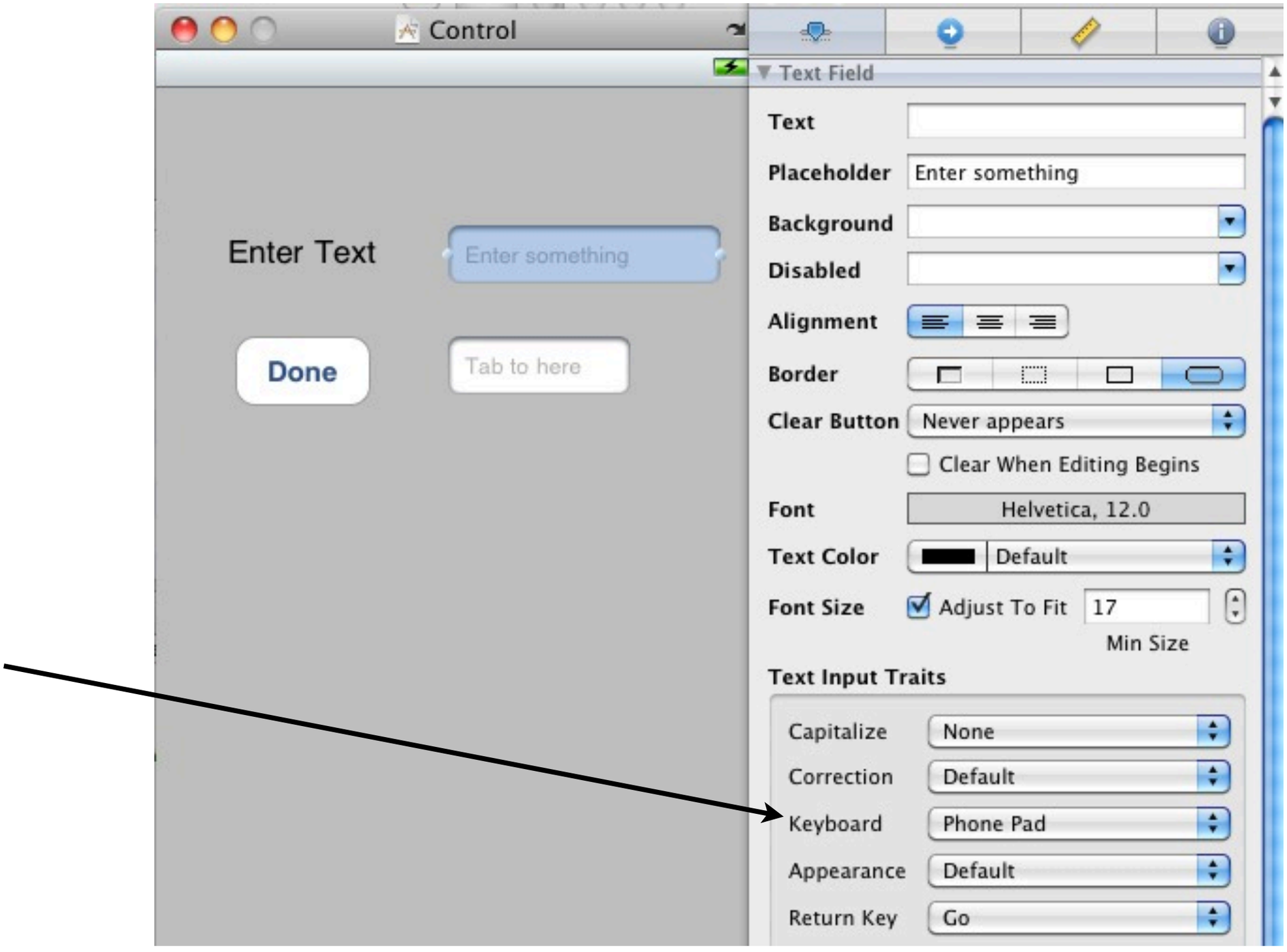


Numbers and Punctuation

URL

Number Pad

Keyboard Type in IB



Keyboard Type in code

```
- (void)viewDidLoad {
    [super viewDidLoad];
       sampleField.keyboardType = UIKeyboardTypeDefault;
       sampleField.returnKeyType = UIReturnKeyGo;
}
```

Keyboard Types

UIKeyboardTypeDefault,
UIKeyboardTypeASCIICapable,
UIKeyboardTypeNumbersAndPunctuation,
UIKeyboardTypeURL,
UIKeyboardTypeNumberPad,
UIKeyboardTypePhonePad,
UIKeyboardTypeNamePhonePad,
UIKeyboardTypeEmailAddress,
UIKeyboardTypeAlphabet = UIKeyboardTypeASCIICapable

Return Key Types

UIReturnKeyDefault,
UIReturnKeyGo,
UIReturnKeyGoogle,
UIReturnKeyJoin,
UIReturnKeyNext,
UIReturnKeyRoute,
UIReturnKeySearch,
UIReturnKeySend,
UIReturnKeyYahoo,
UIReturnKeyDone,
UIReturnKeyEmergencyCall,

Other Keyboard Properties

autocapitalizationType
autocorrectionType
enablesReturnKeyAutomatically
keyboardAppearance
returnKeyType
secureTextEntry

See UITextInputTraits Protocol for more details

Multi-line text field

API does not support multi-line text fields

See:

http://www.hanspinckaers.com/multi-line-uitextview-similar-to-sms

Swift

Complete Example

```
class ViewController: UIViewController {

    @IBOutlet var sampleField: UITextField!
    @IBOutlet var echoField: UITextField!

    override func viewDidLoad() {
        super.viewDidLoad()
        sampleField.keyboardType = UIKeyboardType.ASCIICapable
        sampleField.returnKeyType = UIReturnKeyType.Go;
    }

    @IBAction func donePressed() {
        editDone()
    }
```

```
@IBAction func editDone() {
    echoField.text = sampleField.text
    hideKeyboard()
    NSLog("edit done")
}

override func touchesBegan(touches: NSSet!, withEvent event: UIEvent!) {
    hideKeyboard()
    NSLog("touhes")
    }

    private func hideKeyboard() {
        sampleField.resignFirstResponder()
        echoField.resignFirstResponder()
    }
}
```

Adding getFirstResponder to UIView

UIViewExtension.swift

```
extension UIView {
    func getFirstResponder() -> UIView? {
        if self.isFirstResponder() {
            return self
        }
        for view in subviews {
            if let responder = view.getFirstResponder() {
                return responder
            }
        }
        return nil
    }
}
```

Keyboard Type in code

```
override func viewDidLoad() {
    super.viewDidLoad()
    sampleField.keyboardType = UIKeyboardType.ASCIICapable
    sampleField.returnKeyType = UIReturnKeyType.Go;
}
```

Keyboard Types

UIKeyboardType.Default,
UIKeyboardType.ASCIICapable,
UIKeyboardType.NumbersAndPunctuation,
UIKeyboardType.URL,
UIKeyboardType.NumberPad,
UIKeyboardType.PhonePad,
UIKeyboardType.NamePhonePad,
UIKeyboardType.EmailAddress,
UIKeyboardType.Alphabet = UIKeyboardType.ASCIICapable

Return Key Types

UIReturnKeyType.Default,
UIReturnKeyType.Go,
UIReturnKeyType.Google,
UIReturnKeyType.Join,
UIReturnKeyType.Next,
UIReturnKeyType.Route,
UIReturnKeyType.Search,
UIReturnKeyType.Send,
UIReturnKeyType.Yahoo,
UIReturnKeyType.Done,
UIReturnKeyType.EmergencyCall,