iPhone Application Development Lesson 10

By Dannis Mok

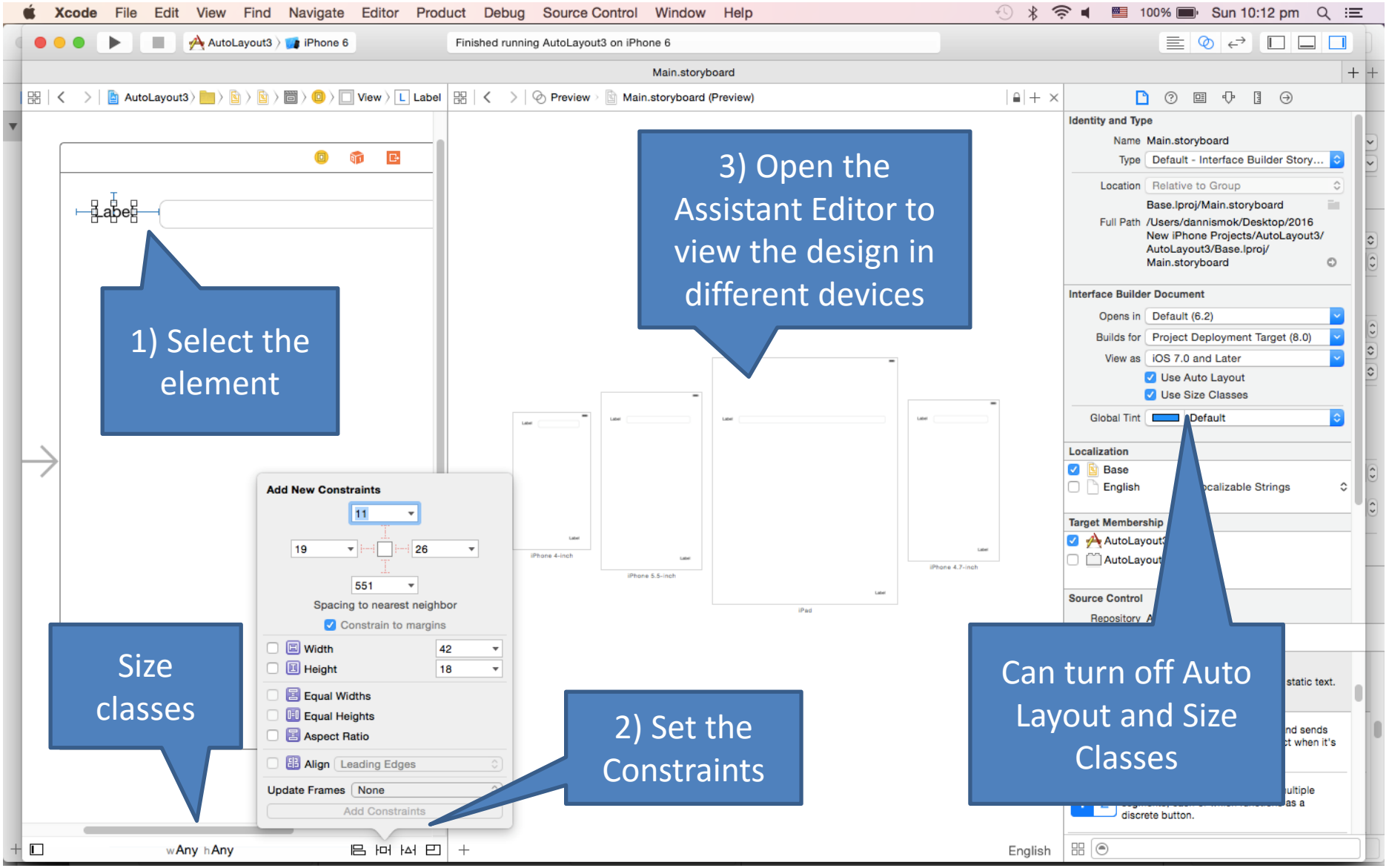
Auto Layout

- Auto Layout helps to add the constraints onto the UI elements and make them to adapt to different devices' sizes and resolutions.

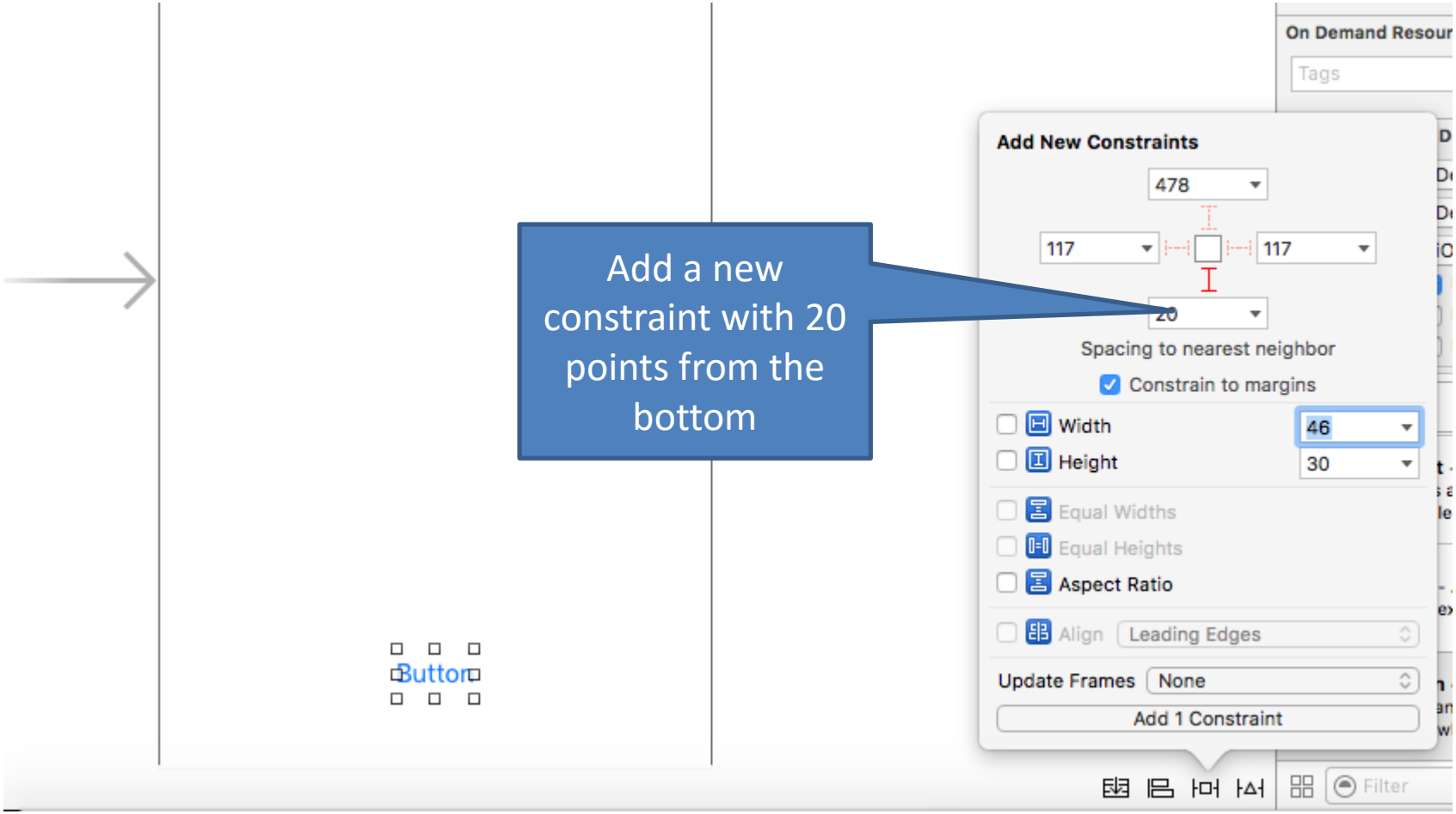
- Size Classes

  iOS defines two size classes: regular and compact. The *regular* size class is associated with expansive space and the *compact* size class is associated with constrained space.

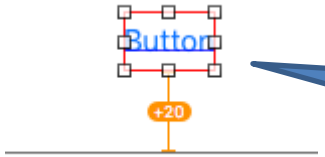
Auto Layout


1) Select the element
3) Open the Assistant Editor to view the design in different devices
Size classes
2) Set the Constraints
Can turn off Auto Layout and Size Classes
Add New Constraints
Spacing to nearest neighbor
Constrain to margins
Width
Height
Equal Widths
Equal Heights
Aspect Ratio
Align
Leading Edges
Update Frames
None
Add Constraints
Identity and Type
Interface Builder Document
Localization
Target Membership
Source Control
Use Auto Layout
Use Size Classes
Global Tint
Default

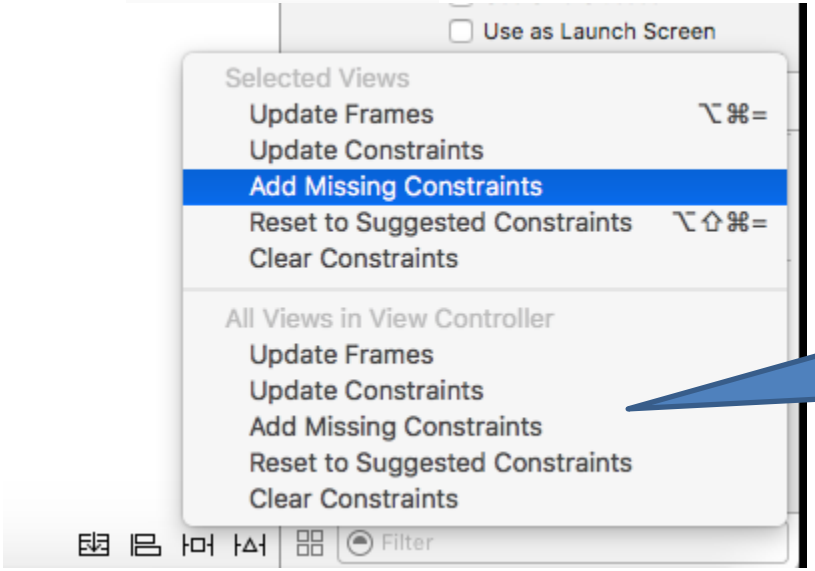
Auto Layout Example



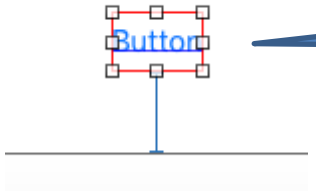
Auto Layout Example



1) Orange line means the view's current position is not matched with the constraint set.

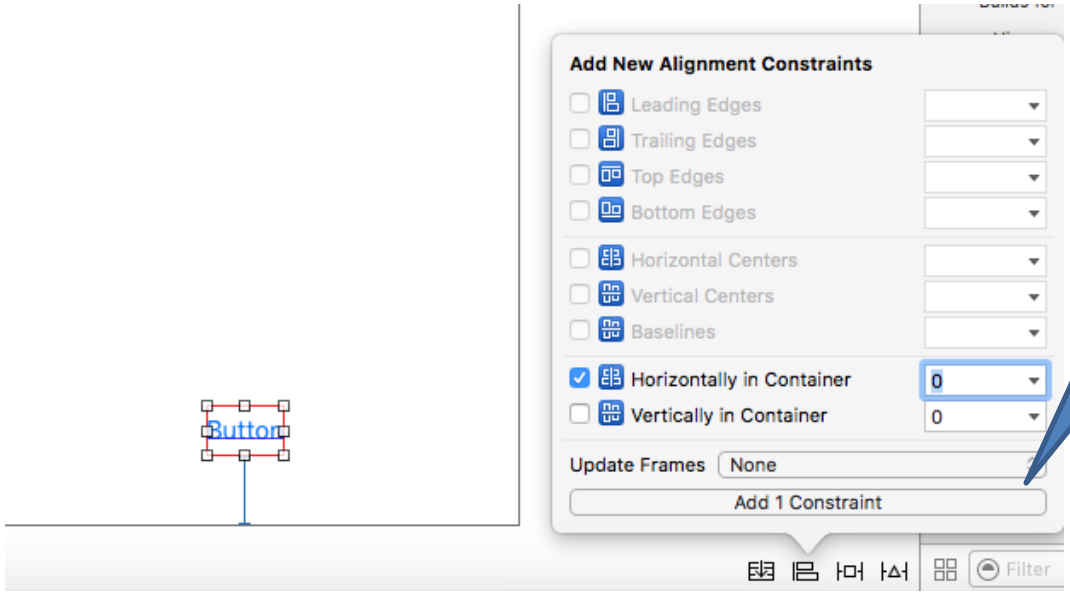


2) Can update frames or update constraints to remove the orange line

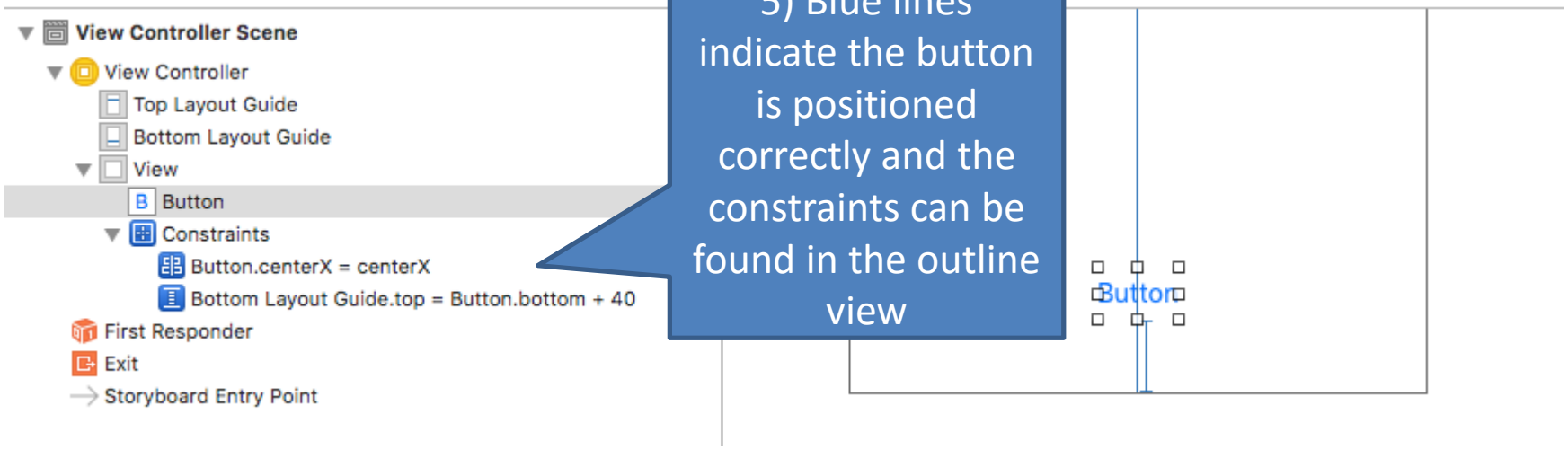


3) Red line means the constraints are not enough for fixing the view

Auto Layout Example

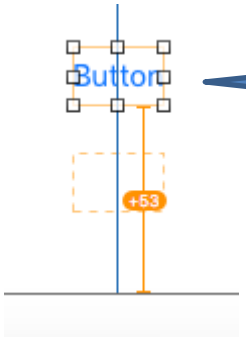


4) Center the button using Alignment constraints by centering it the superview

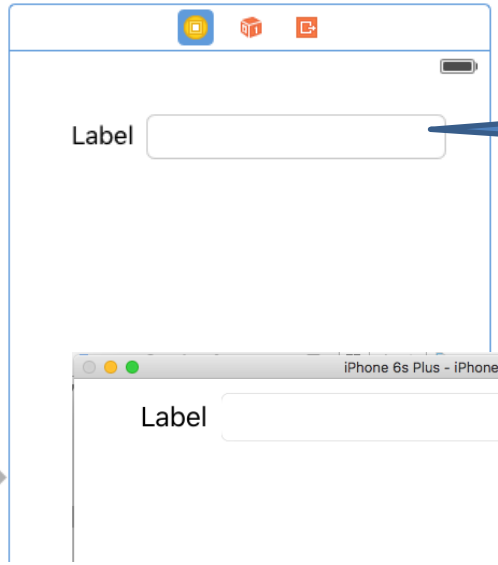


5) Blue lines indicate the button is positioned correctly and the constraints can be found in the outline view

Auto Layout Example



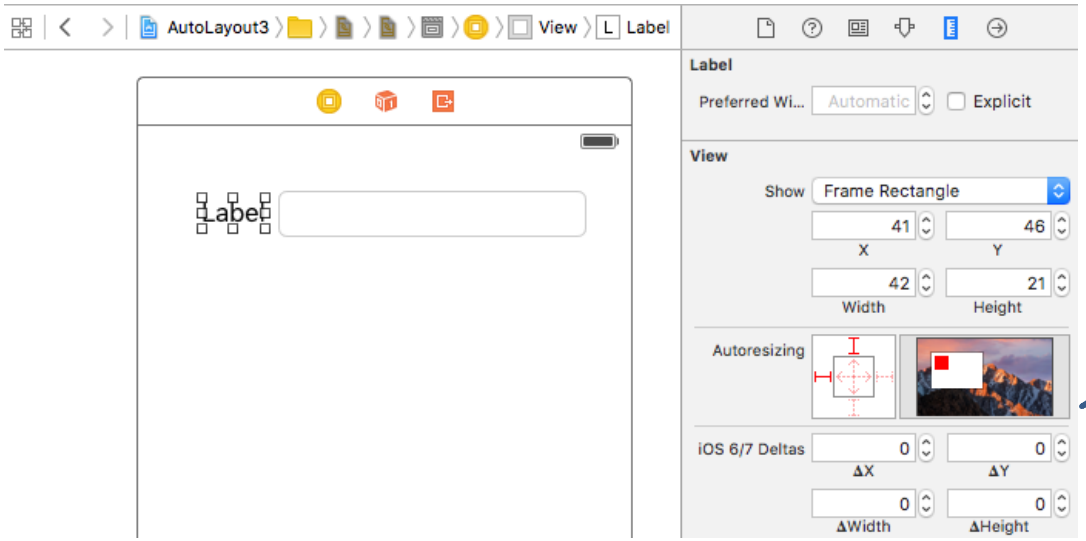
** Dotted orange line is the position indicated by the constraints and the solid orange line is the current position of the view in the Xcode. It can be solved by updating frame or update constraint



The layout is fine when in portrait mode. But the textfield will not occupy the full width if not autolayout or autosizing applied

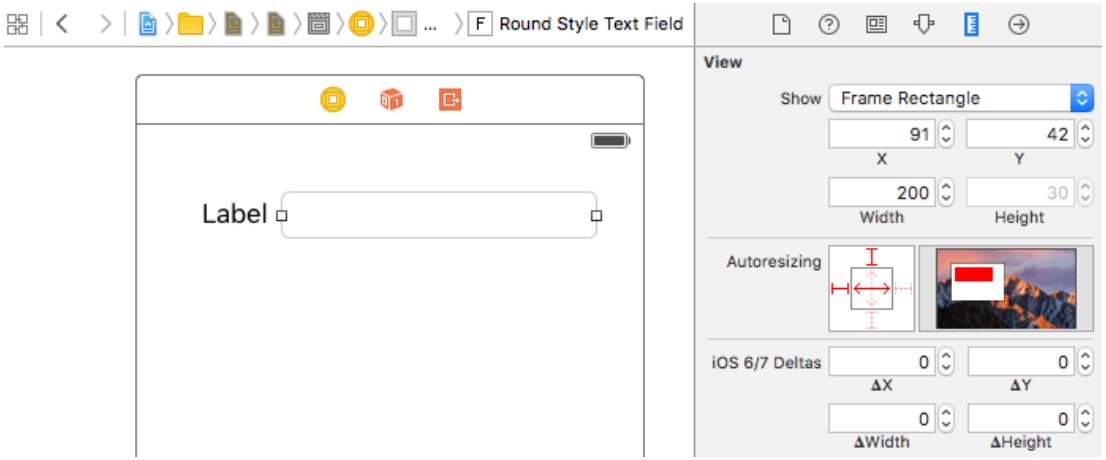
Not occupying the full width

Auto Layout Example



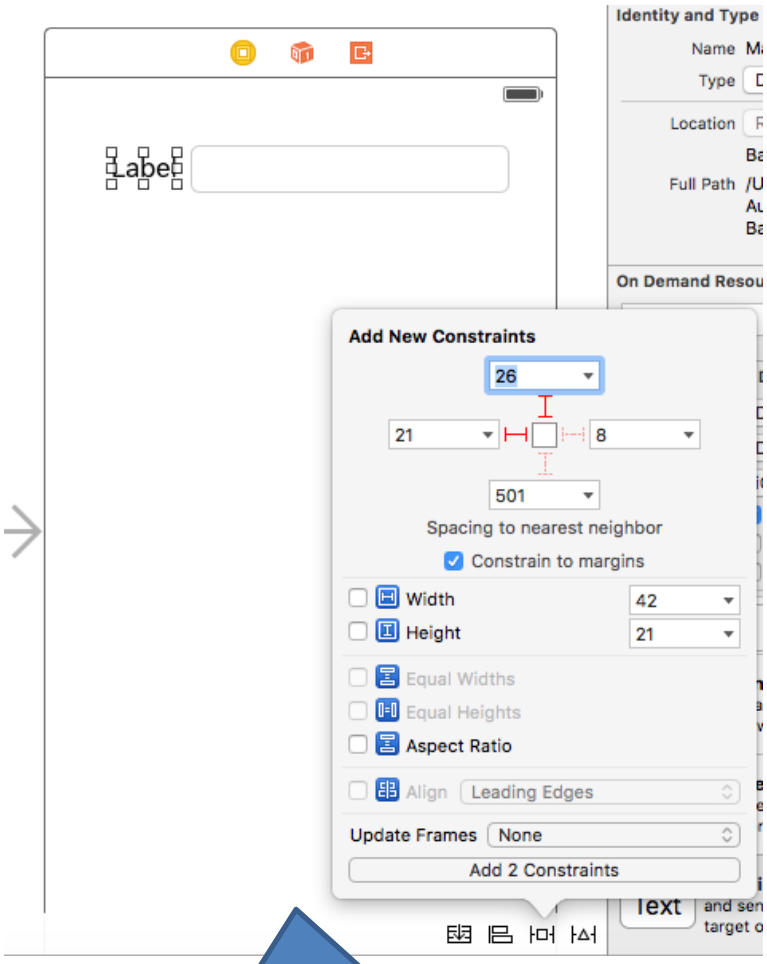
Auto Sizing Solution

Auto Sizing for the Label



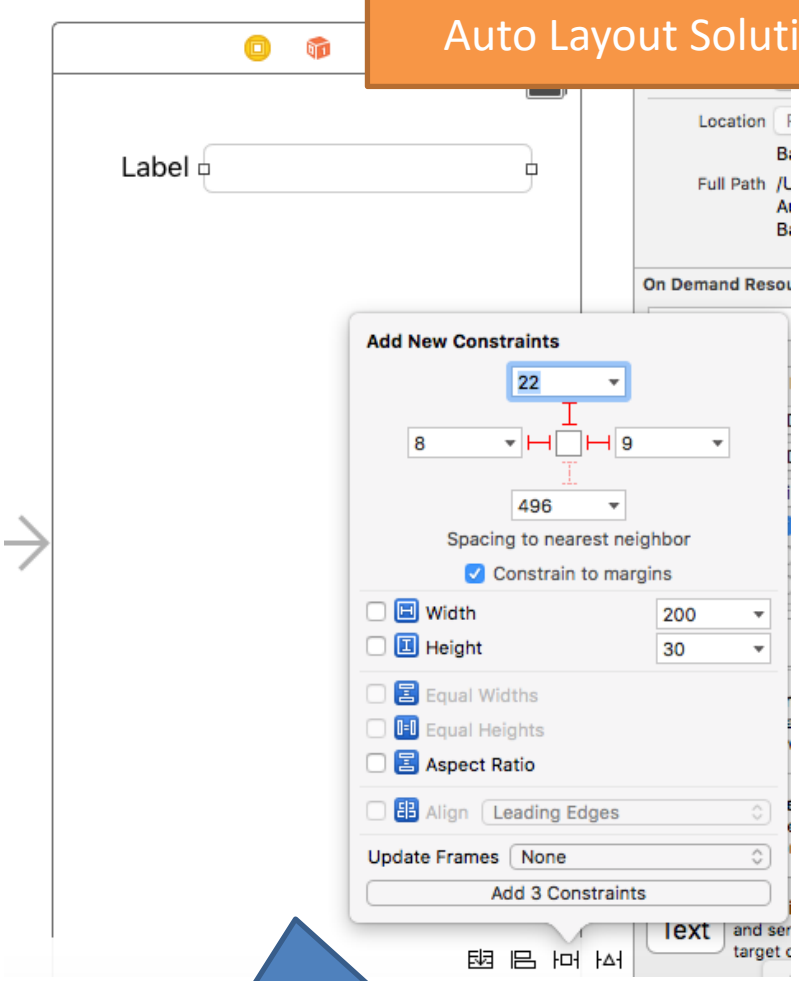
Auto Sizing for the TextField

Auto Layout Example



Add 2 constraints for Label

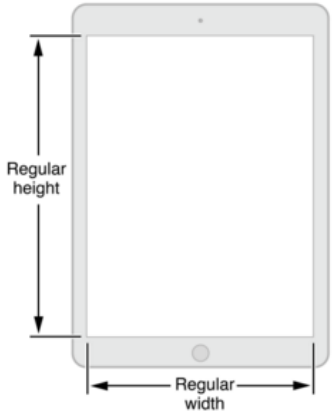
Auto Layout Solution



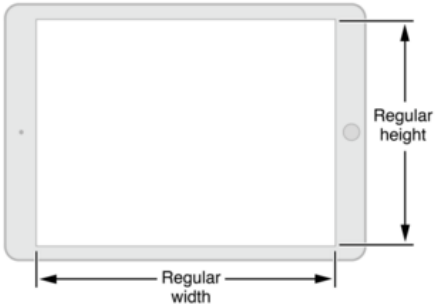
Add 3 constraints for TextField

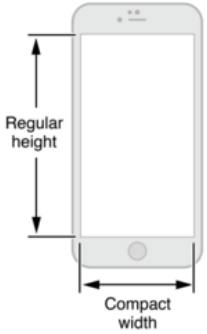
Size Classes

The size classes of iPad in portrait

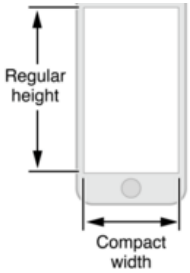


The size classes of iPad in landscape



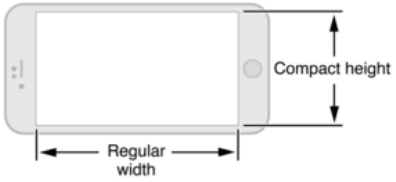


In portrait, iPhone 6 Plus uses the compact horizontal and regular vertical size classes.

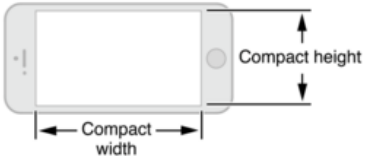


In portrait, iPhone 6, iPhone 5, and iPhone 4s use the compact horizontal and regular vertical size classes.

In landscape, iPhone 6 Plus uses the regular horizontal and compact vertical size classes.



In landscape, these devices use the compact size class in both the horizontal and vertical dimensions.

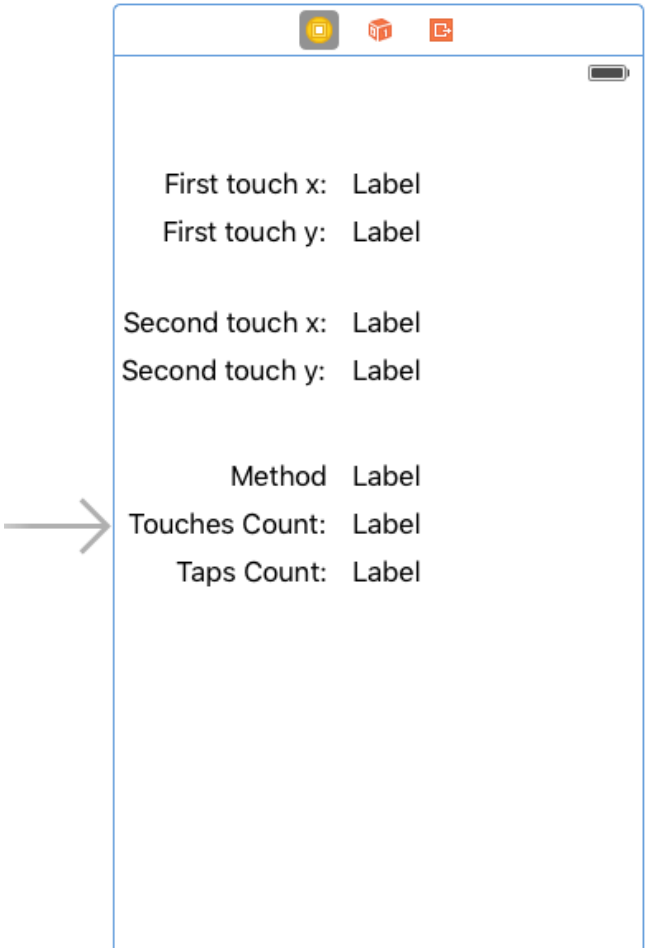


Touch Processing

The system continually sends TouchEvent objects to an application as fingers touch and move across a surface.

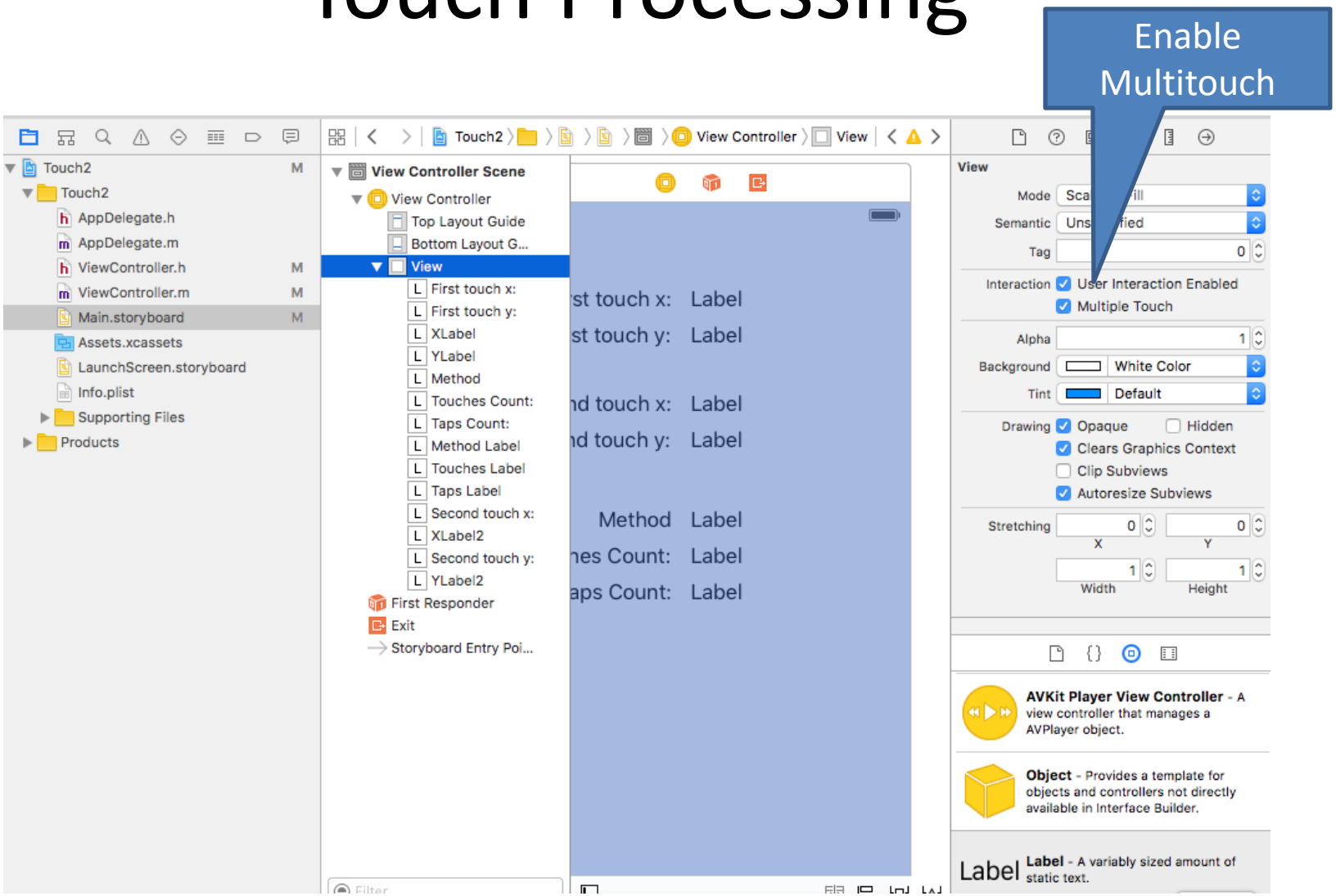
- touchstart
  - Sent when a finger for a given event touches the surface.
- touchmove
  - Sent when a given event moves on the surface.
- touchend
  - Sent when a given event lifts from the surface.
- touchcancel
  - Sent when the system cancels tracking for the touch.

Touch Processing




```
#import <UIKit/UIKit.h>
@interface ViewController : UIViewController

@property (weak, nonatomic) IBOutlet UILabel *xLabel;
@property (weak, nonatomic) IBOutlet UILabel *yLabel;
@property (weak, nonatomic) IBOutlet UILabel *xLabel2;
@property (weak, nonatomic) IBOutlet UILabel *yLabel2;
@property (weak, nonatomic) IBOutlet UILabel *methodLabel;
@property (weak, nonatomic) IBOutlet UILabel *touchesLabel;
@property (weak, nonatomic) IBOutlet UILabel *tapsLabel;
@end
```

Touch Processing



Touch Processing

```
-(void)touchesBegan:(NSSet<UITouch *> *)touches withEvent:(UIEvent *)event {
    NSUInteger touchCount = [touches count];
    NSArray *touchArray = [touches allObjects];
    UITouch *touchObject = [touchArray objectAtIndex:0];
    NSUInteger tapCount = [touchObject tapCount];
    CGPoint point = [touchObject locationInView:self.view];
    CGFloat pointX = point.x;
    CGFloat pointY = point.y;
    _xLabel.text = [NSString stringWithFormat:@"%f",pointX ];
    _yLabel.text = [NSString stringWithFormat:@"%f",pointY ];
    _methodLabel.text = @"Touch Begin";
    _touchesLabel.text = [NSString stringWithFormat:@"Touch Counts: %lu", (unsigned long)touchCount];
    _tapsLabel.text = [NSString stringWithFormat:@"Tap Counts: %lu", (unsigned long)tapCount];
    if(touchCount == 2) {
        UITouch *touchObject = [touchArray objectAtIndex:1];
        CGPoint point = [touchObject locationInView:self.view];
        CGFloat pointX = point.x;
        CGFloat pointY = point.y;
        _xLabel2.text = [NSString stringWithFormat:@"%f",pointX ];
        _yLabel2.text = [NSString stringWithFormat:@"%f",pointY ];
    }
}
```

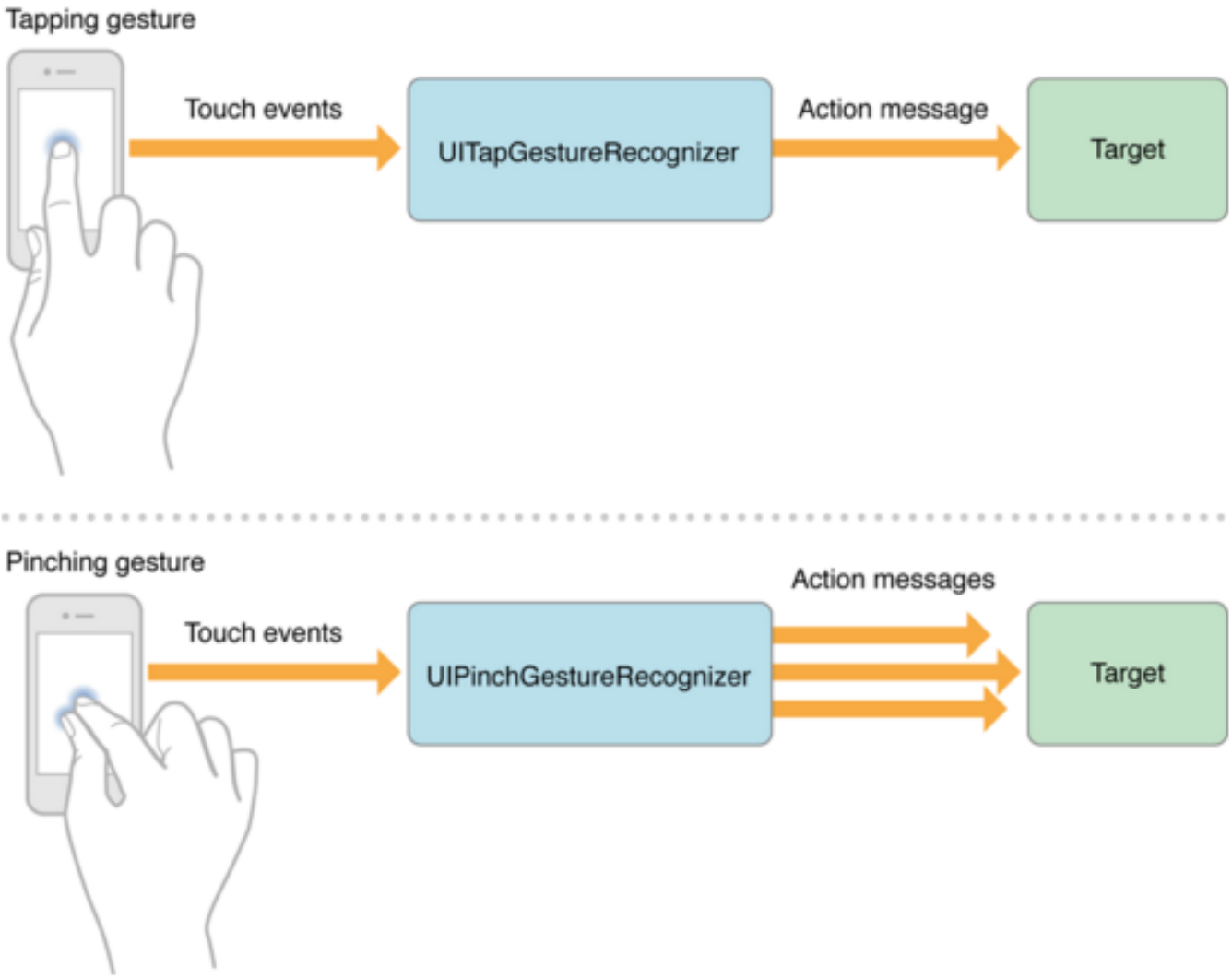
Convert the NSSet to an array and then extract the UITouch object for processing. The UITouch object means the finger touch. If there are 2 fingers touching, there are 2 UITouch object in the set

Gesture Processing

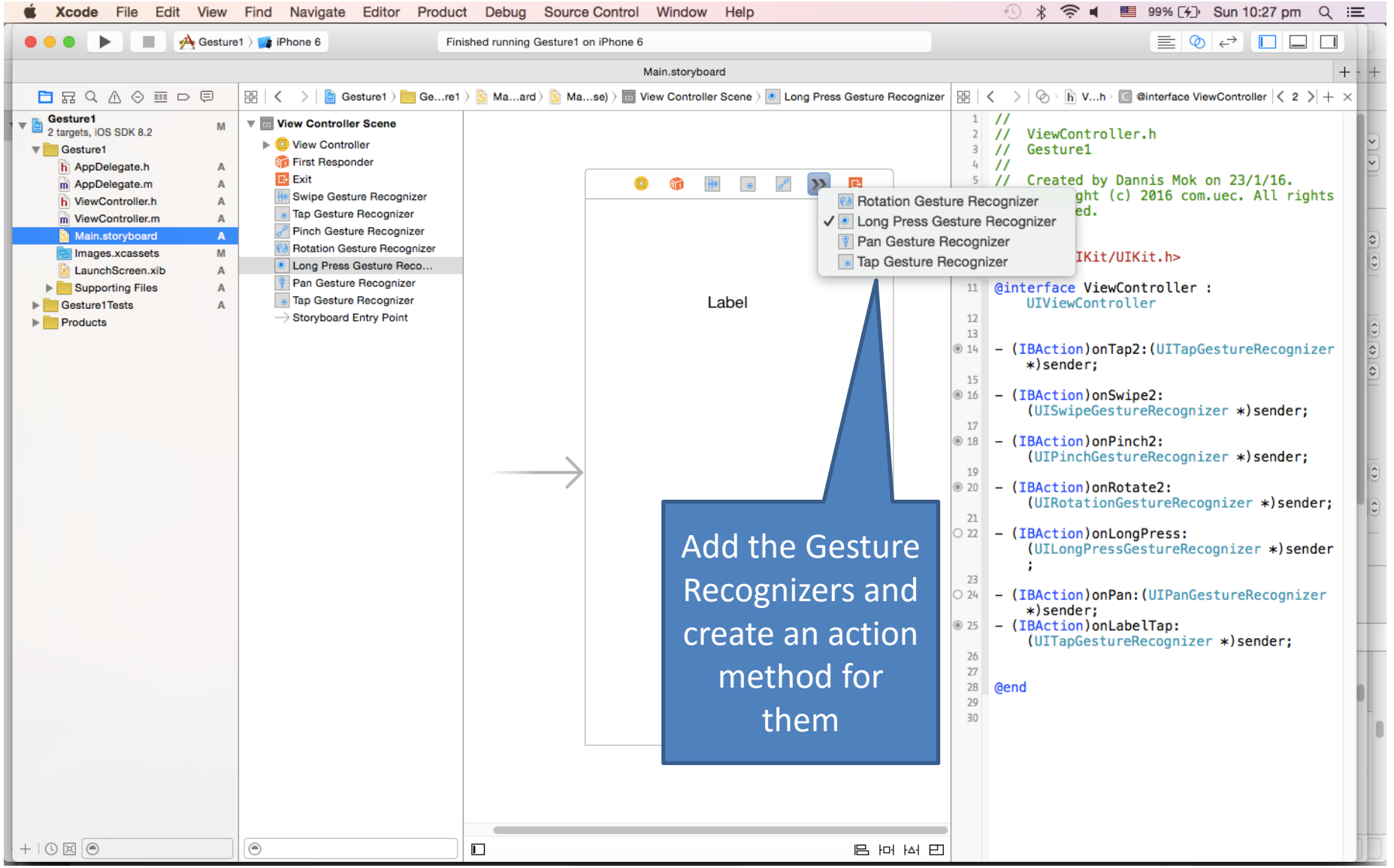
The UIKit framework provides predefined gesture recognizers that detect common gestures

Gesture	UIKit class
Tapping (any number of taps)	`UITapGestureRecognizer`
Pinching in and out (for zooming a view)	`UIPinchGestureRecognizer`
Panning or dragging	`UIPanGestureRecognizer`
Swiping (in any direction)	`UISwipeGestureRecognizer`
Rotating (fingers moving in opposite directions)	`UIRotationGestureRecognizer`
Long press (also known as “touch and hold”)	`UILongPressGestureRecognizer`

Gesture Processing



Gesture Processing

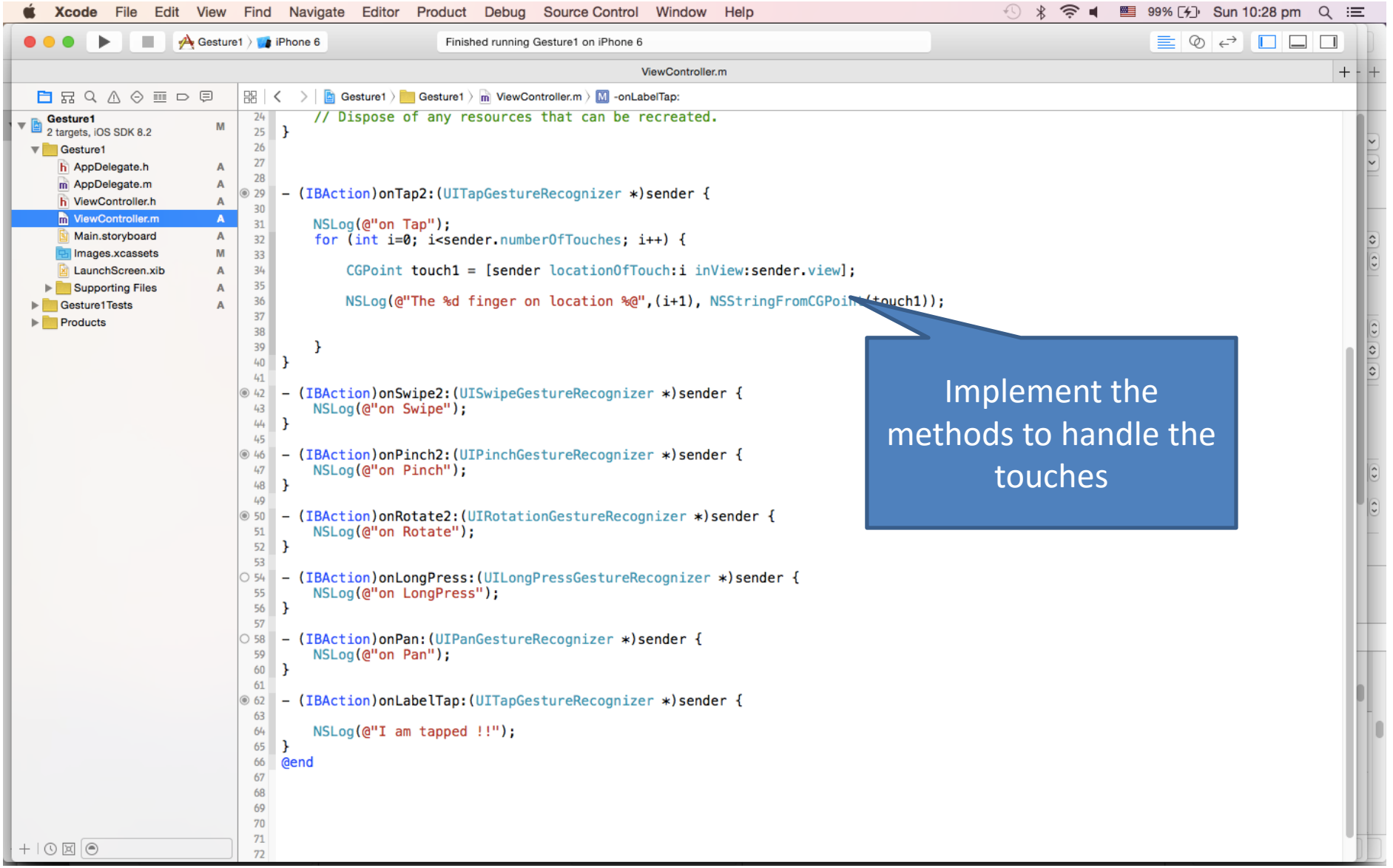


Gesture Processing

```
- (IBAction)onLongPress:(UILongPressGestureRecognizer *)sender;
- (IBAction)onSwipe:(UISwipeGestureRecognizer *)sender;
- (IBAction)onRotation:(UIRotationGestureRecognizer *)sender;
- (IBAction)onPinch:(UIPinchGestureRecognizer *)sender;
- (IBAction)onTap:(UITapGestureRecognizer *)sender;
```

Please remember to change the class type from id to the above class types for various GestureRecognizers when creating the action methods

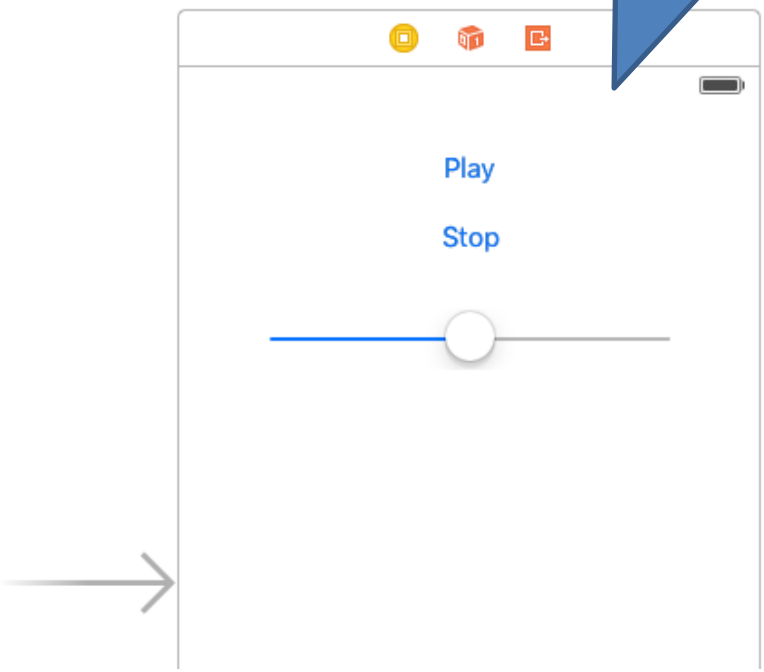
Gesture Processing



Play Audio

Create the necessary outlets and actions

1) Import AVFoundation framework and declare to implement the AVAudioPlayerDelegate protocol




```
#import <UIKit/UIKit.h>
#import <AVFoundation/AVFoundation.h>

@interface AudioViewController : UIViewController
   <AVAudioPlayerDelegate>

@property (strong, nonatomic) AVAudioPlayer *audioPlayer;

@property (weak, nonatomic) IBOutlet UISlider *volumeControl;

- (IBAction)stopAudio:(id)sender;

- (IBAction)playAudio:(id)sender;

- (IBAction)adjustVolume:(id)sender;

@end
```

Play Audio

```
@implementation AudioViewController

- (void)viewDidLoad
{
       [super viewDidLoad];
       NSURL *url = [NSURL fileURLWithPath:[[NSBundle mainBundle]
            pathForResource:@"Moderato"
            ofType:@"mp3"]];

        NSError *error;
        _audioPlayer = [[AVAudioPlayer alloc]
                        initWithContentsOfURL:url
                        error:&error];
        if (error)
        {
                NSLog(@"Error in audioPlayer: %@",
                      [error localizedDescription]);
        } else {
                _audioPlayer.delegate = self;
                [_audioPlayer prepareToPlay];
        }
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

- (IBAction)stopAudio:(id)sender {
    [_audioPlayer stop];
}

- (IBAction)playAudio:(id)sender {
    [_audioPlayer play];
}

- (IBAction)adjustVolume:(id)sender {
    if (_audioPlayer != nil)
    {
        _audioPlayer.volume = _volumeControl.value;
    }
}
```

The mp3 file to be played

Create the AVAudioPlayer object

Prepare the file to play

Use to button to play, stop or adjust volume

Play Audio

```
-(void)audioPlayerDidFinishPlaying:
(AVAudioPlayer *)player successfully:(BOOL)flag
{
}

-(void)audioPlayerDecodeErrorDidOccur:
(AVAudioPlayer *)player error:(NSError *)error
{
}

-(void)audioPlayerBeginInterruption:(AVAudioPlayer *)player
{
}

-(void)audioPlayerEndInterruption:(AVAudioPlayer *)player
{
}

@end
```

Other delegate related methods can be implemented as well.

Photo Taking

Photo Taking

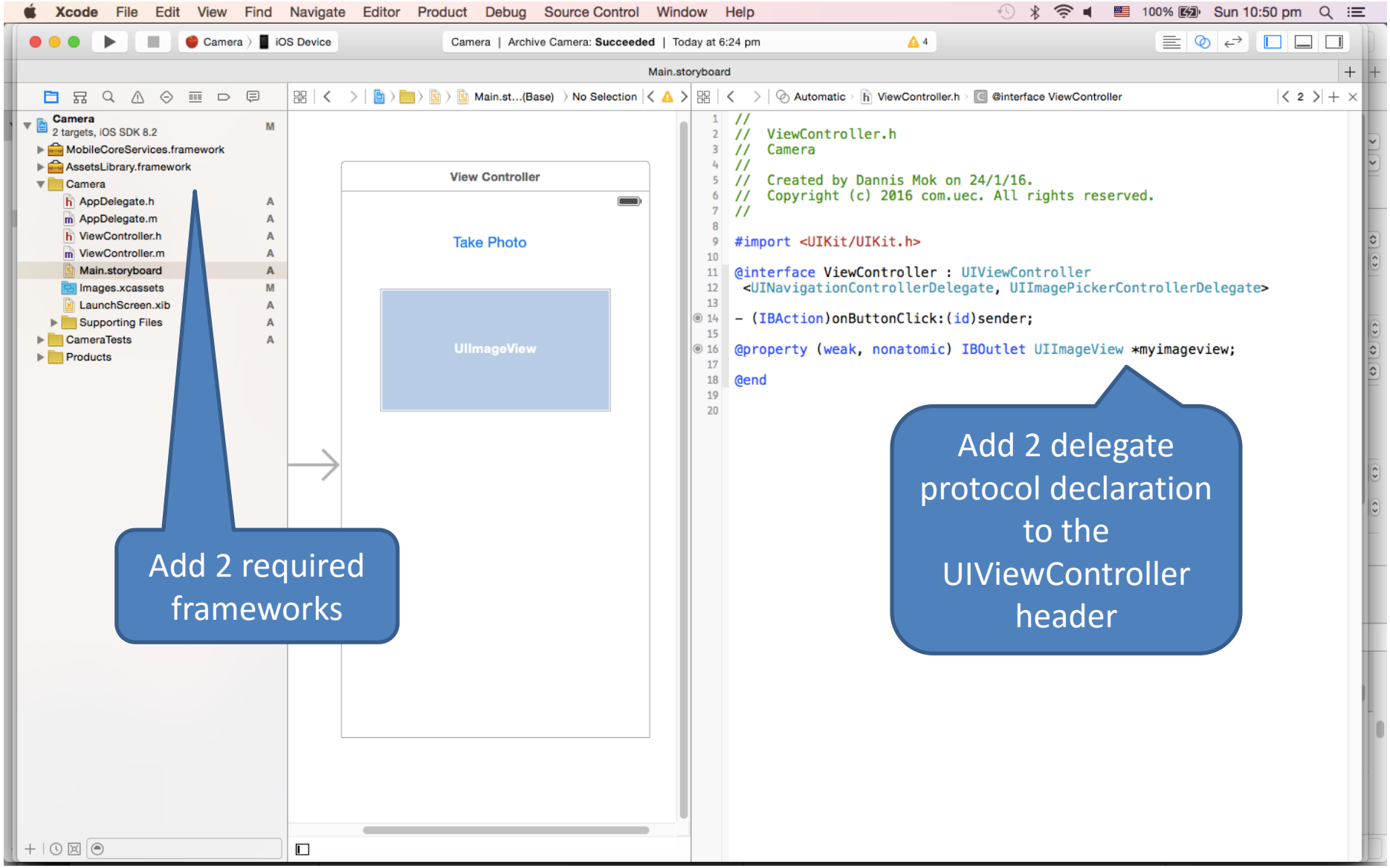
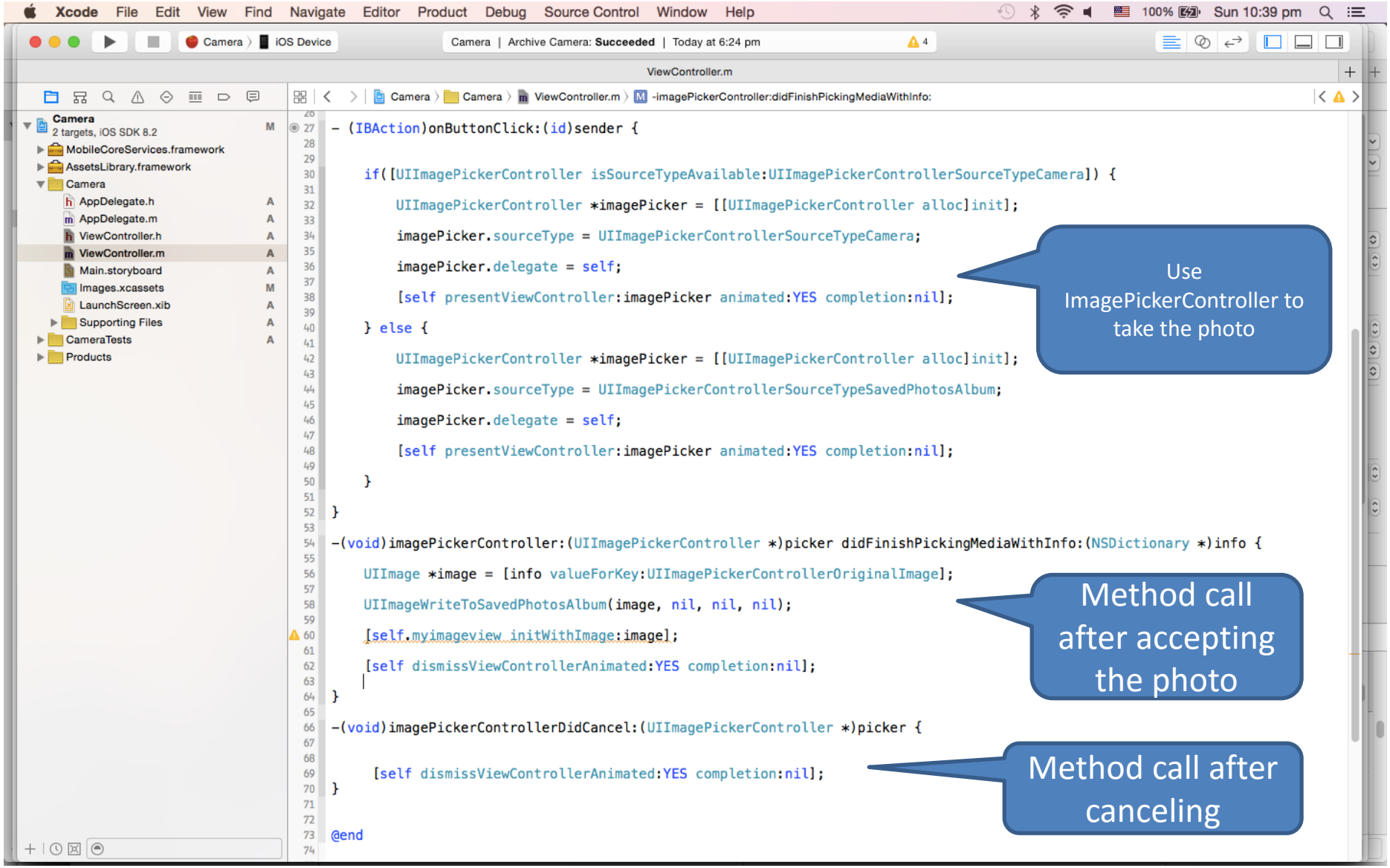
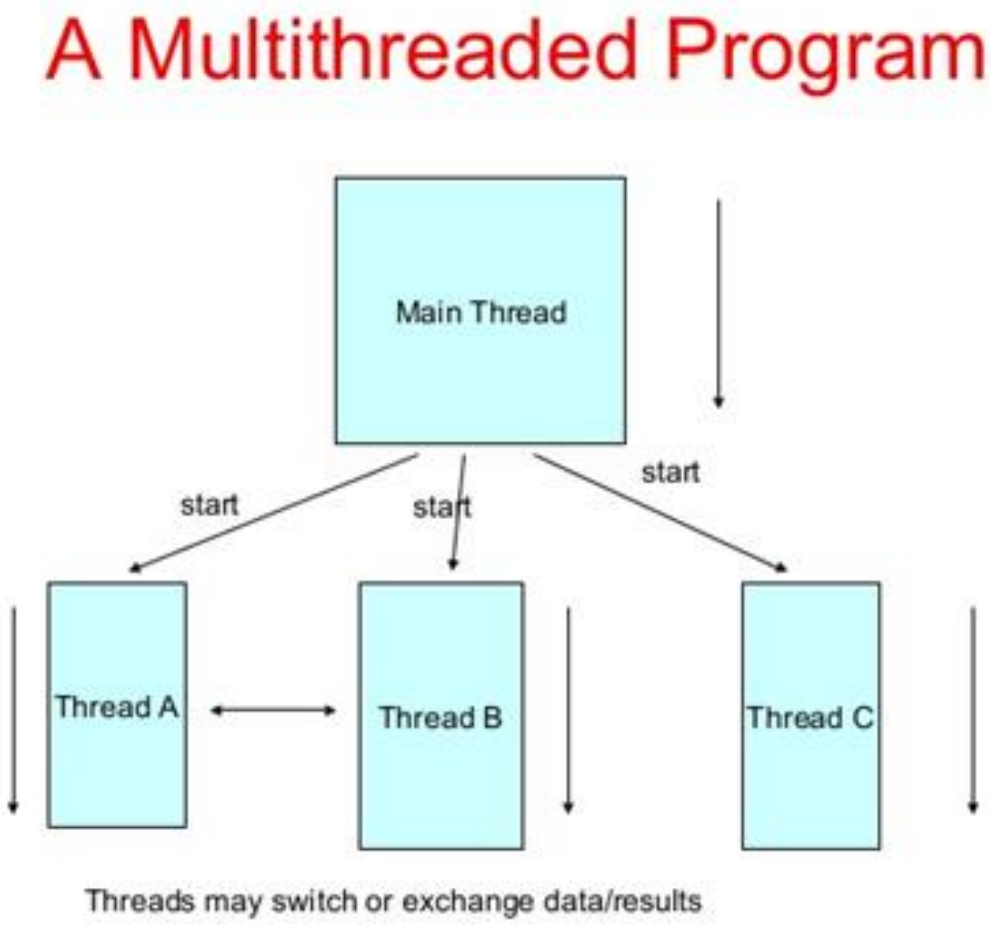


Photo Taking



Multi-Threading



Multi-Threading

```
-(void)run1:(NSTimer *)timer {
    static int n = 0;
    if(n == 10) {
        [timer invalidate];
    }
    NSLog(@"%d",n++);
}
-(void)thread1 {
    for(int i=0;i<5;i++) {
        NSLog(@"%d",i);
    }
}

-(void)thread2 {
    for(int i=100;i<110;i++) {
        NSLog(@"%d",i);
    }
}
```

NSTimer has a built-in thread. It will be scheduled to run this method and display the n value till it reaches 10

This thread method will print 0 to 5 to the log file

This thread method will print 100 to 110 to the log file

Multi-Threading

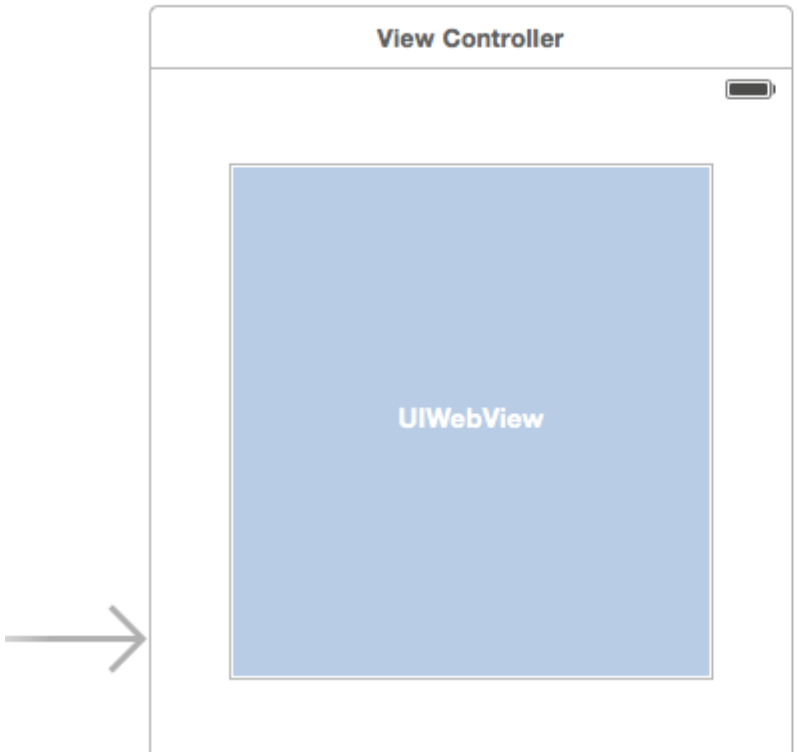
```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    NSLog(@"A");

    [NSThread sleepForTimeInterval:3.0];

    NSLog(@"B");

    [NSTimer scheduledTimerWithTimeInterval:1.0 target:self selector:@selector(run1:) userInfo:
        nil repeats:YES];


    NSThread *t1 = [[NSThread alloc]initWithTarget:self selector:@selector(thread1) object:nil];

    NSThread *t2 = [[NSThread alloc]initWithTarget:self selector:@selector(thread2) object:nil];

    [t1 start];

    [t2 start];

}
```

1) Make the main thread sleep for 3 seconds

2) Use NSTimer to run the method

3) Create 2 threads to run the 2 different methods

Web View




```
1  //
2  //  ViewController.h
3  //  Network1
4  //
5  //  Created by Dannis Mok on 24/1/16.
6  //  Copyright (c) 2016 com.uec. All rights
       reserved.
7  //
8
9  #import <UIKit/UIKit.h>
10
11 @interface ViewController : UIViewController
12
13
14 @property (weak, nonatomic) IBOutlet
       UIWebView *myWebView;
15
16
17
18 @end
19
20
```

Web View

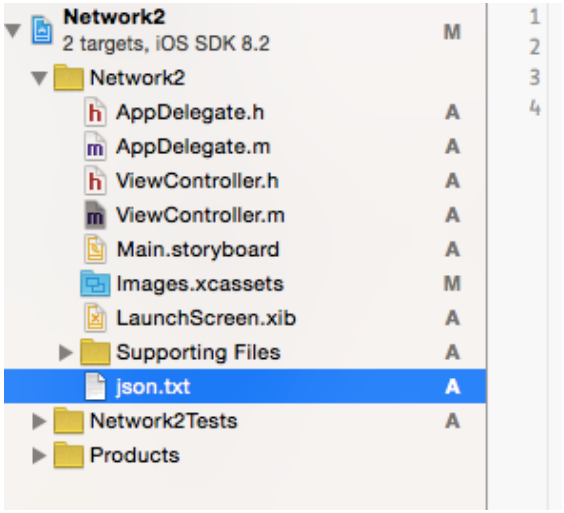
```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Print the HTML code in the WebView

    NSString *html = @"<h1>iPhone Programming</h1><h2>from Apple</h2><img src='apple.jpg' />";

    [myWebView loadHTMLString:html baseURL:nil];

    // Print the file content in the WebView

    NSURL *url = [NSURL URLWithString:@"http://localhost:8888/hello.html"];

    NSURLRequest *request = [NSURLRequest requestWithURL:url];

    [myWebView loadRequest:request];

    // Create an Image from the local file and then display it

    UIImageView *newimage = [UIImageView new];

    [newimage setImage:[UIImage imageNamed:@"apple.jpg"]];

    CGPoint point = CGPointMake(0,0);

    [newimage setCenter:point];

    [self.view addSubview:newimage];

}
```

Use the Web View to display HTML

Use the Web View to show web page content

Create and Image and show it dynamically

JSON & Network




```
1 [{"id":"1","name":"Peter","phone":"99434939","email":"flower1.jpeg"},
2  {"id":"2","name":"Mary","phone":"29349349","email":"flower2.jpeg"},
3  {"id":"3","name":"Sam","phone":"9239432","email":"flower3.jpeg"},
4  {"id":"4","name":"David","phone":"39493943","email":"flower4.jpeg"}]
```

Can create a local JSON format file

JSON & Network

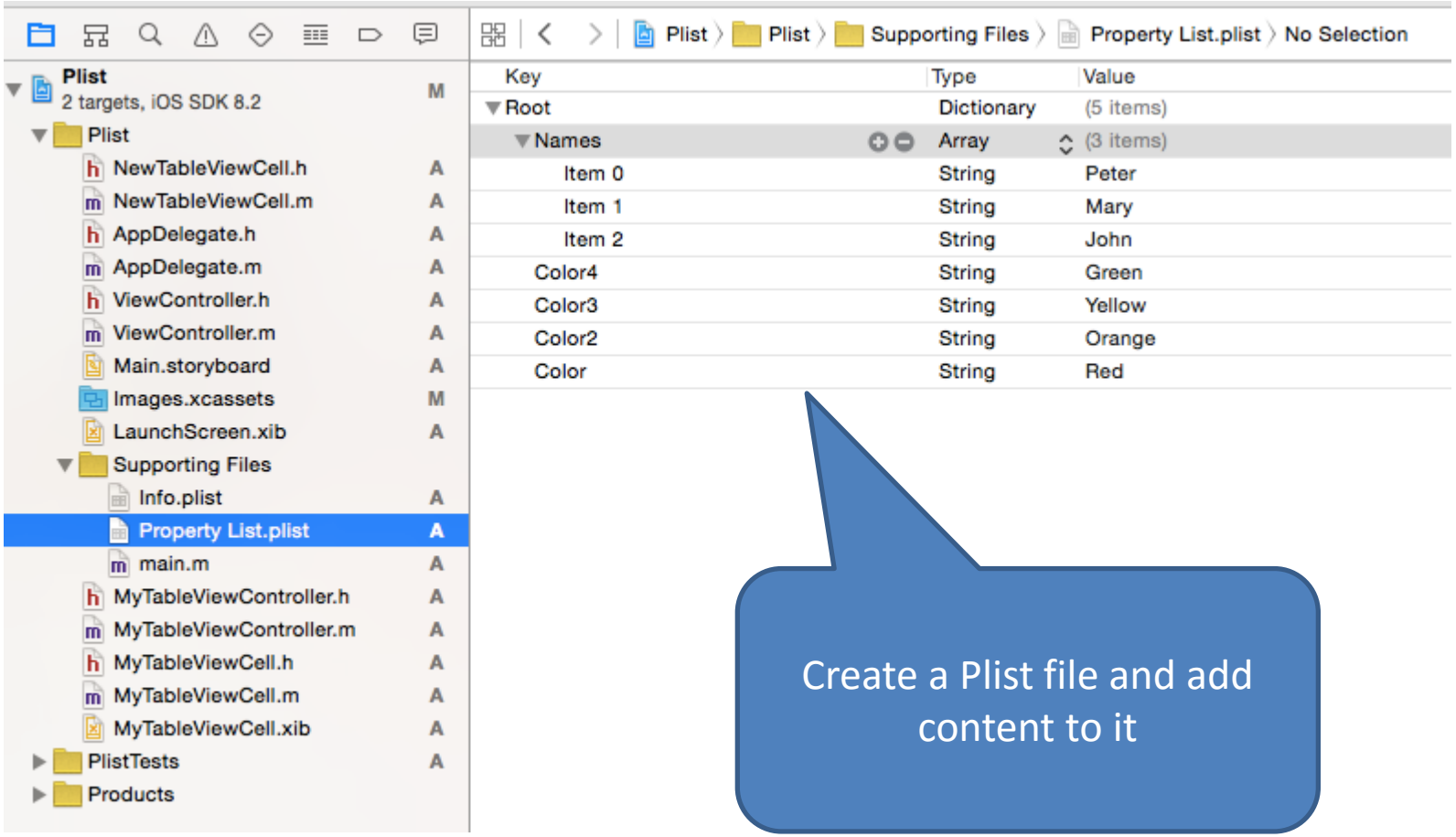
```
- (void)viewDidLoad {
    [super viewDidLoad];

    // Read data from JSON and then read the image from the network

    NSString *path = [[NSBundle mainBundle]pathForResource:@"json" ofType:@"txt"];

    NSData *data = [NSData dataWithContentsOfFile:path];

    NSArray *jsonarray = [NSJSONSerialization JSONObjectWithData:data options:
        NSJSONReadingMutableContainers error:nil];

    for(NSDictionary *p in jsonarray) {

        NSString *sid = [p objectForKey:@"id"];
        NSString *name = [p objectForKey:@"name"];
        NSString *tel = [p objectForKey:@"phone"];
        NSString *photo = [p objectForKey:@"email"];

        NSLog(@"%@ %@ %@ %@",sid,name,tel,photo);

        NSMutableString *path = [NSMutableString stringWithString:@"http://localhost:8888/"];

        path = [[path stringByAppendingString:photo]mutableCopy];

        NSLog(@"%@",path);

        NSURL *url = [NSURL URLWithString:path];

        NSURLRequest *request = [NSURLRequest requestWithURL:url];

        [NSURLConnection sendAsynchronousRequest:request queue:[NSOperationQueue mainQueue]
            completionHandler:^(NSURLResponse *response, NSData *data, NSError *connectionError)
            {
            UIImage *image = [UIImage imageWithData:data];

            NSLog(@"%@, %@",response,  image);
        }];

    }
```

Read the file content into the NSData object

Read the data and the use the image path to download the image

Use asynchronous thread to download the image

Plist file

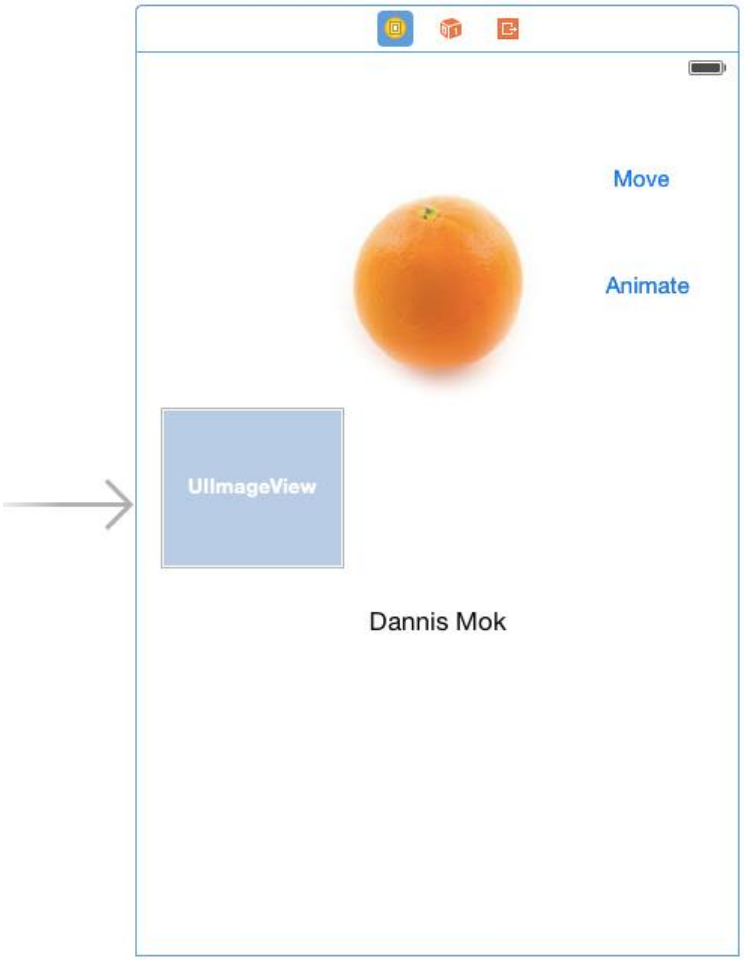

Plist
2 targets, iOS SDK 8.2
Plist
NewTableViewCell.h
NewTableViewCell.m
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Images.xcassets
LaunchScreen.xib
Supporting Files
Info.plist
Property List.plist
main.m
MyTableViewController.h
MyTableViewController.m
MyTableViewCell.h
MyTableViewCell.m
MyTableViewCell.xib
PlistTests
Products
Plist
Plist
Supporting Files
Property List.plist
No Selection
Key
Type
Value
Root
Dictionary
(5 items)
Names
Array
(3 items)
Item 0
String
Peter
Item 1
String
Mary
Item 2
String
John
Color4
String
Green
Color3
String
Yellow
Color2
String
Orange
Color
String
Red
Create a Plist file and add content to it

Plist file

```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    NSString *plistpath = [[NSBundle mainBundle]pathForResource:@"Property List" ofType:@"plist"
        ];

    NSMutableDictionary *plist = [NSMutableDictionary dictionaryWithContentsOfFile:plistpath];

    NSString *color = [plist objectForKey:@"Color"];

    NSLog(@"The color is %@",color);

}
```

How to read the content of the Plist file and then print it out

UIView Properties




```
1  //
2  //  ViewController.h
3  //  UIViewTest
4  //
5  //  Created by Dannis Mok on 20/1/16.
6  //  Copyright (c) 2016 com.uec. All rights reserved.
7  //
8
9  #import <UIKit/UIKit.h>
10
11 @interface ViewController : UIViewController
12
13
14 @property (weak, nonatomic) IBOutlet UIImageView *myOrange;
15
16
17 @property (weak, nonatomic) IBOutlet UILabel *myName;
18
19 - (IBAction)onClick:(id)sender;
20 @property (weak, nonatomic) IBOutlet UIImageView *myOrange2
       ;
21 - (IBAction)onClick2:(id)sender;
22
23 @end
24
25
```

Can use the UIView animation properties

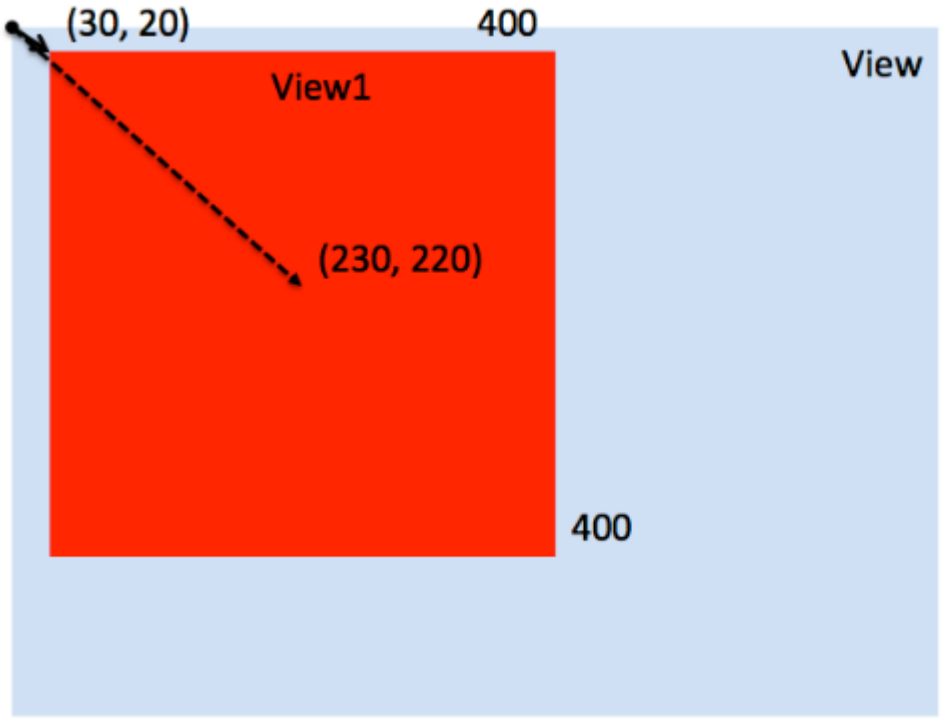
UIView Properties

```
- (IBAction)onClick:(id)sender {

    float x1 = myName.center.x + 10;
    float y1 = myName.center.y - 10;


    CGPoint newPoint = CGPointMake(x1,y1);

    myName.center = newPoint;

    NSLog(@"%f",myName.center.x);
    NSLog(@"%f",myName.center.y);


    [UIView beginAnimations:nil context:nil];
    [UIView setAnimationRepeatCount:2];
    [UIView setAnimationDuration:2];


    myOrange.bounds = CGRectMake(0, 0, 100, 100);

    myOrange.backgroundColor = [UIColor redColor];
    myName.backgroundColor = [UIColor blueColor];
    myName.textColor = [UIColor yellowColor];

    myOrange.alpha = 0.5;
    myOrange.transform = CGAffineTransformMakeRotation(30);
    myName.transform = CGAffineTransformMakeScale(1.5, 1.5);
    myOrange.transform = CGAffineTransformMakeTranslation(20, 0);

    [UIView commitAnimations];



    UIImage *orange = [UIImage imageNamed:@"orange.jpg"];

    UIImageView *newImageView = [[UIImageView alloc]initWithImage:orange];

    newImageView.frame = CGRectMake(0, 0, 100, 100);
    newImageView.center = CGPointMake(50,50);

}
```

Move the UI element by moving its center

Wrap the property changes in the animation block

UIView Properties



View1's Frame
(30, 20, 400, 400)

View1's Bounds
(0,0, 400, 400)

View1's Center
(230, 220)

Frame: A view's frame (CGRect) is the position of its rectangle in the superview's coordinate system. By default it starts at the top left.

Bounds: A view's bounds (CGRect) expresses a view rectangle in its own coordinate system.

Center: A center is a CGPoint expressed in terms of the superview's coordinate system and it determines the position of the exact center point of the view.

Notification

```
- (IBAction)onPinch:(UIPinchGestureRecognizer *)sender {

    if(sender.state == UIGestureRecognizerStateBegan) {


        NSLog(@"Start");

    } else if(sender.state == UIGestureRecognizerStateChanged) {

        NSLog(@"Changed");

    } else if(sender.state == UIGestureRecognizerStateEnded) {

        NSLog(@"Ended");

        UILocalNotification *ln = [UILocalNotification new];

        ln.alertBody = @"Message I send";
        ln.applicationIconBadgeNumber = 2;
        ln.fireDate = [NSDate dateWithTimeIntervalSinceNow:5.0];

        [[UIApplication sharedApplication]scheduleLocalNotification:ln];

    }

}
```

Send out notification

Notification

```
#import "AppDelegate.h"

@interface AppDelegate ()

@end

@implementation AppDelegate


- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
    launchOptions {
    // Override point for customization after application launch.

    if ([UIApplication instancesRespondToSelector:@selector(registerUserNotificationSettings:)])
        {
        [application registerUserNotificationSettings:[UIUserNotificationSettings
            settingsForTypes:UIUserNotificationTypeAlert|UIUserNotificationTypeBadge|
            UIUserNotificationTypeSound categories:nil]];
    }

    return YES;
}
```

Add this to the AppDelegate's applicationDidFinishLaunchingWithOptions to grant the permission to accept notification (IOS 8 later)