Chapter 12
Touch Events and UIResponder

- Touch Events
- Creating the TouchTracker Application
- Drawing with INIDrawView
- Turning Touches to Lines

Touch Events

- As a subclass of UIResponder, a UIView can override four methods to handle the four distinct touch events:

  a finger or fingers touches the screen

  - (void) touchesBegan:(NSSet *) touches withEvent:(UIEvent *) event;

  a finger or fingers moves across the screen (this message is sent repeatedly as a finger moves)

  - (void) touchesMoved:(NSSet *) touches withEvent:(UIEvent *) event;

  a finger or fingers removed from the screen

  - (void) touchesEnded:(NSSet *) touches withEvent:(UIEvent *) event;

  a system event, like an incoming phone call, interrupts a touch before it ends

  - (void) touchesCanceled:(NSSet *) touches withEvent:(UIEvent *) event;

UITouch

- A UITouch object represents the presence or movement of a finger on the screen for a particular event.

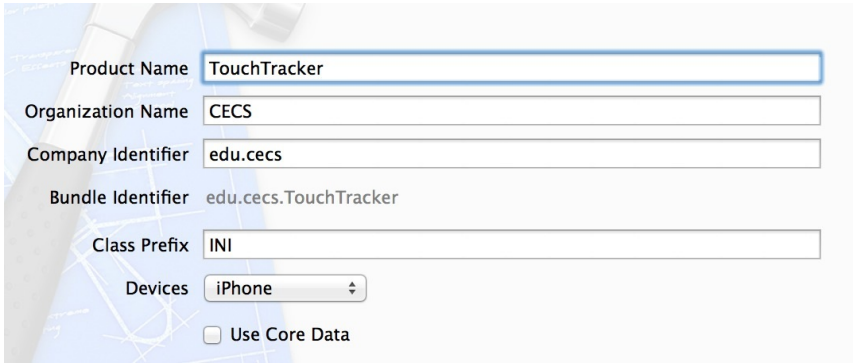
  When a finger touches the screen, an instance of UITouch is created. The UIView that this finger touched is sent the message touchesBegan: withEvent: and the UITouch is in the NSSet of touches.

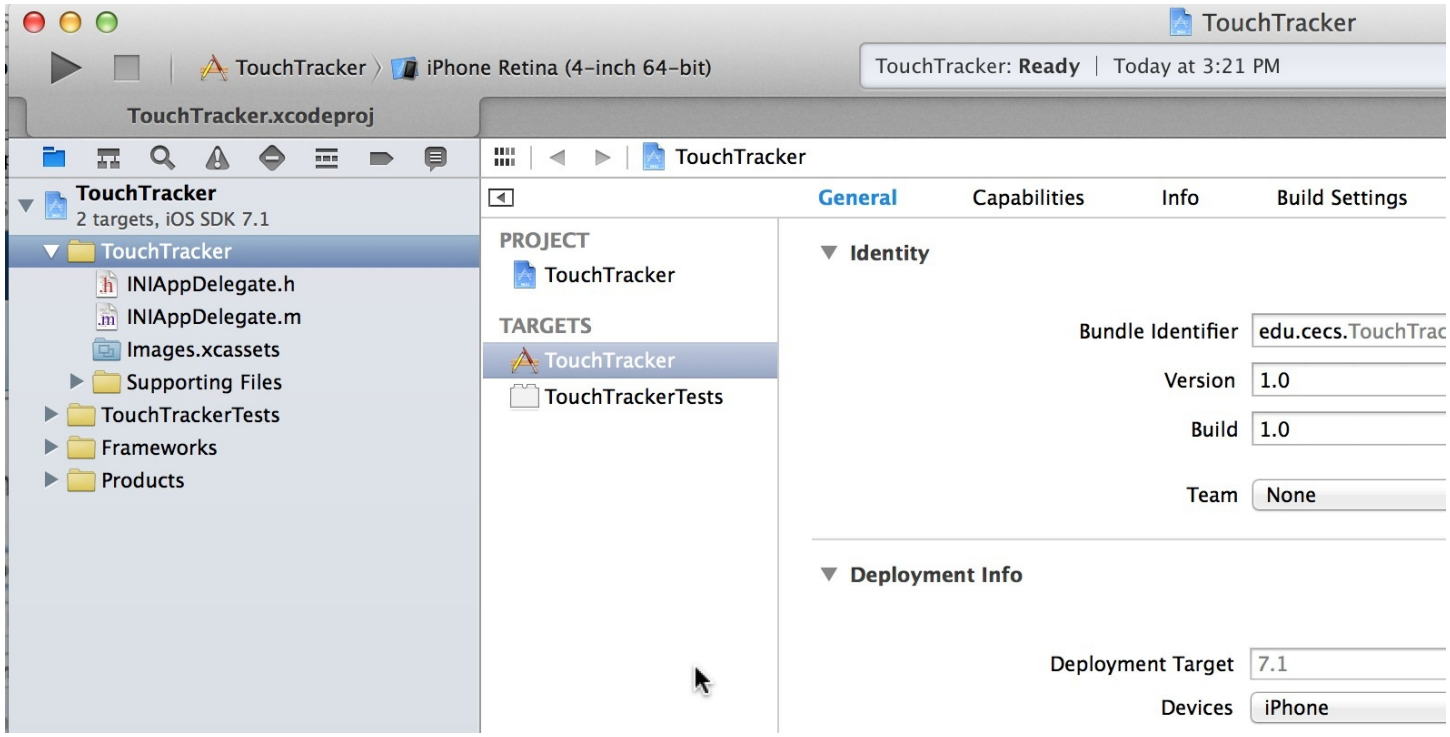
  As that finger moves around the screen, the touch object is updated to contain the current location of the finger on the screen. Then, the same UIView that the touch began on is sent the message touchesMoved: withEvent:. The NSSet that is passed as an argument to this method contains the same UITouch that originally was created when the finger it represents touched the screen.

  When a finger is removed from the screen, the touch object is updated one last time to contain the current location of the finger, and the view that the touch began on is sent the message touchesEnded: withEvent:.

  After that method finishes executing, the UITouch object is destroyed.

Creating TouchTracker





INILine & INIDrawView

- Create an NSObject extension called INILine
  - Add a begin and end properties as nonatomic CGPoints
- Create an NSObject extension called INIDrawView
- Change the super-class of INIDrawView to an UIView

INIDrawViewController

- Create an NSObject extension called INIDrawViewController
- Change the super-class of INIDrawViewController to an UIViewController
- Set INIDrawViewController as the root controller for our app

T... › INIDrawViewController.h › No Selection

```
1  //
2  //  INIDrawViewController.h
3  //  TouchTracker
4  //
5  //  Created by Ibrahim Imam on 6/22/14.
6  //  Copyright (c) 2014 CECS. All rights reserved.
7  //
8
9  #import <Foundation/Foundation.h>
10
11 @interface INIDrawViewController : UIViewController
12
13 @end
14
```

Counterparts › INIDrawViewController.m › -loadView

```
1  //
2  //  INIDrawViewController.m
3  //  TouchTracker
4  //
5  //  Created by Ibrahim Imam on 6/22/14.
6  //  Copyright (c) 2014 CECS. All rights reserved.
7  //
8
9  #import "INIDrawViewController.h"
10 #import "INIDrawView.h"
11
12 @implementation INIDrawViewController
13
14 - (void) loadView
15 {
16     self.view = [[INIDrawView alloc] initWithFrame: CGRectZero];
17 }
```

ick Help

Description: A rectangle constant with location (0,0), and width and height of 0. The zero rectangle is equivalent to CGRectMake(0,0,0,0).

Availability: iOS (2.0 and later)

Declared In: CGGeometry.h

Reference: CGGeometry Reference

Related: CGPointZero, CGSizeZero

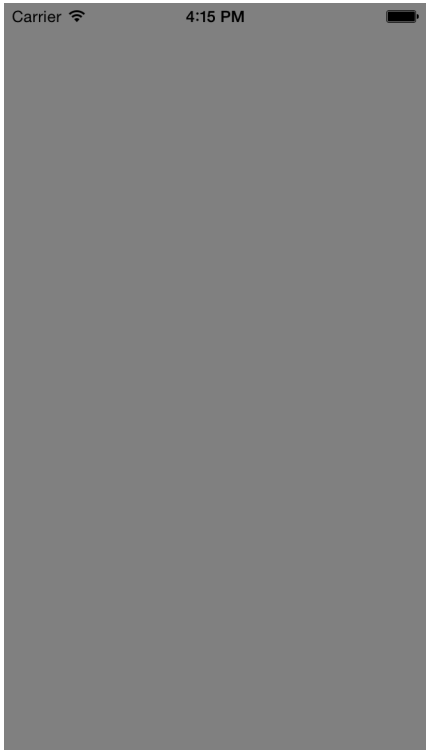
Guides: Quartz 2D Programming Guide

```
9  #import "INIAppDelegate.h"
10 #import "INIDrawViewController.h"
11
12 @implementation INIAppDelegate
13
14 - (BOOL)application:(UIApplication *)application didFinishLaunchingWithO
15 {
16     self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen]
17     // Override point for customization after application launch.
18
19     INIDrawViewController *dvc = [[INIDrawViewController alloc] init];
20     self.window.rootViewController = dvc;
21
22     self.window.backgroundColor = [UIColor whiteColor];
23     [self.window makeKeyAndVisible];
24     return YES;
25 }
```

Modify INIDrawView

- Add a currentLine property of type INILine
- Add finishedLines as a mutable array to hold completed lines
- Override the initializer to draw all finished lines and to turn background gray

```
8
9   #import "INIDrawView.h"
10  #import "INILine.h"
11
12  @interface INIDrawView()
13
14  @property (nonatomic, strong) INILine *currentLine;
15  @property (nonatomic, strong) NSMutableArray *finishedLines;
16
17  @end
18
19  @implementation INIDrawView
20
21  - (instancetype) initWithFrame:(CGRect) r
22  {
23      self = [super initWithFrame: r];
24      if (self) {
25          self.finishedLines = [[NSMutableArray alloc] init];
26          self.backgroundColor = [UIColor grayColor];
27      }
28      return self;
29  }
30
31  @end
```



Add Line And Rectangle Drawing Methods

```
30
31  - (void) strokeLine:(INILine *) line
32  {
33      UIBezierPath *bp = [UIBezierPath bezierPath];
34
35      bp.lineWidth = 10;
36      bp.lineCapStyle = kCGLineCapRound;
37
38      [bp moveToPoint: line.begin];
39      [bp addLineToPoint: line.end];
40      [bp stroke];
41  }
42
43  - (void) drawRect:(CGRect) rect {
44
45      // Draw finished lines in black
46      [[UIColor blackColor] set];
47      for (INILine *line in self.finishedLines) {
48          [self strokeLine: line];
49      }
50
51      if (self.currentLine) {
52          // If there is a line currently being drawn, do it in red
53          [[UIColor redColor] set];
54          [self strokeLine: self.currentLine];
55      }
56  }
57
```

Turning Touches into Lines

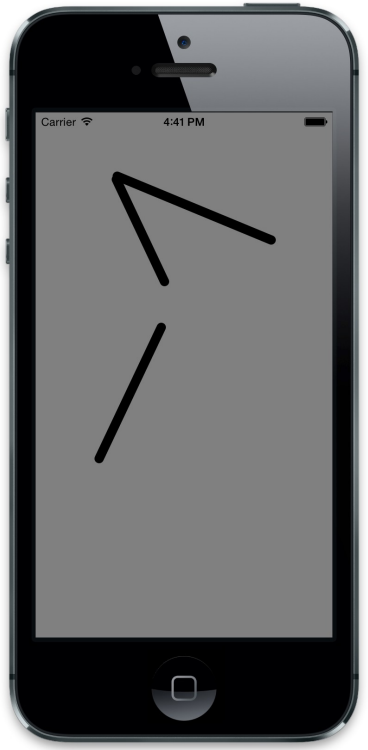
- A line is defined by two points.
- INILine stores these points as the properties begin and end.
- When a touch begins, we will create a line and set both begin and end to the point where the touch began.
- When the touch moves, you will update end.
- When the touch ends, you will have your complete line.

```
}

- (void) touchesBegan:(NSSet *) touches withEvent:(UIEvent *) event
{
    UITouch * t = [touches anyObject];
    // Get location of the touch in view's coordinate system
    CGPoint location = [t locationInView: self];
    self.currentLine = [[INILine alloc] init];
    self.currentLine.begin = location;
    self.currentLine.end = location;
    [self setNeedsDisplay];
}

- (void) touchesMoved:(NSSet *) touches withEvent:(UIEvent *) event
{
    UITouch * t = [touches anyObject];
    CGPoint location = [t locationInView: self];
    self.currentLine.end = location;
    [self setNeedsDisplay];
}

- (void) touchesEnded:(NSSet *) touches withEvent:(UIEvent *) event
{
    [self.finishedLines addObject: self.currentLine];
    self.currentLine = nil;
    [self setNeedsDisplay];
}
```



Handling multiple touches

- Add self.multipleTouchEnabled = YES; to the initializer of INIDrawView
- Now that we turned multi-touch on, there is a possibility that we may have more than one line being drawn at the same time. These may have been:

  Started at the same time (low possibility but could happen). This will generate multiple touches in one call to touchesBegan:withEvent:

  Started at different time instances in this case every touch will generate a call to touchesBegan:withEvent:
- To handle these multiple touches we no longer have one “currentLine” being drawn but several.
- Thus, we need to house all these active lines in some sort of a collection.
- We use NSMutable Dictionary to house lines in progress.

Implementing Multiple Touches

```
13
14  // @property (nonatomic, strong) INILine *currentLine;      //Single line being drawn
15  @property (nonatomic, strong) NSMutableDictionary * linesInProgress;  //Multiple lines being drawn
16  @property (nonatomic, strong) NSMutableArray *finishedLines;
17
```

```
- (void) touchesBegan:(NSSet *) touches withEvent:(UIEvent *) event
{
    // Let's put in a log statement to see the order of events
    NSLog(@"%@", NSStringFromSelector(_cmd));  // Display the name of the current method

//    UITouch * t = [touches anyObject];
//    // Get location of the touch in view's coordinate system
//    CGPoint location = [t locationInView: self];
//    self.currentLine = [[INILine alloc] init];
//    self.currentLine.begin = location;
//    self.currentLine.end = location;

    for (UITouch *t in touches) {
        CGPoint location = [t locationInView: self];
        INILine * line = [[INILine alloc] init];
        line.begin = location;
        line.end = location;
        NSValue *key = [NSValue valueWithNonretainedObject: t];
        self.linesInProgress[key] = line;
    }

    [self setNeedsDisplay];
}
```

```
- (void) touchesMoved:(NSSet *) touches withEvent:(UIEvent *) event
{
    // Let's put in a log statement to see the order of events
    NSLog(@"%@", NSStringFromSelector(_cmd));
    for (UITouch *t in touches) {
        NSValue *key = [NSValue valueWithNonretainedObject: t];
        INILine * line = self.linesInProgress[key];
        line.end = [t locationInView: self];
    }

//    UITouch * t = [touches anyObject];
//    CGPoint location = [t locationInView: self];
//    self.currentLine.end = location;

    [self setNeedsDisplay];
}
```

```
- (void) touchesEnded:(NSSet *) touches withEvent:(UIEvent *) event
{
    // Let's put in a log statement to see the order of events
    NSLog(@"%@", NSStringFromSelector(_cmd));
    for (UITouch *t in touches) {
        NSValue *key = [NSValue valueWithNonretainedObject: t];
        INILine *line = self.linesInProgress[key];
        [self.finishedLines addObject: line];
        [self.linesInProgress removeObjectForKey: key];
    }

//    [self.finishedLines addObject: self.currentLine];
//    self.currentLine = nil;

    [self setNeedsDisplay];
}
```

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

    // Draw finished lines in black
    [[UIColor blackColor] set];
    for (INILine *line in self.finishedLines) {
        [self strokeLine: line];
    }

    [[UIColor redColor] set];
    for (NSValue *key in self.linesInProgress) {
        [self strokeLine: self.linesInProgress[key]];
    }

//    if (self.currentLine) {
//        // If there is a line currently being drawn, do it in red
//        [[UIColor redColor] set];
//        [self strokeLine: self.currentLine];
//    }
}
```

Canceling Touches

- A touch can be cancelled when an application is interrupted by the operating system (for example, a phone call comes in) when a touch is currently on the screen.
- When a touch is cancelled, any state it set up should be reverted. In this case, you should remove any lines in progress.

```
}

- (void) touchesCancelled:(NSSet *) touches withEvent:(UIEvent *) event
{
    // Let's put in a log statement to see the order of events
    NSLog(@"%@", NSStringFromSelector(_cmd));
    for (UITouch *t in touches) {
        NSValue *key = [NSValue valueWithNonretainedObject: t];
        [self.linesInProgress removeObjectForKey: key];
    }
    [self setNeedsDisplay];
}
```

```
2014-06-23 11:13:13.309 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:13.388 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:14.086 TouchTracker[5145:60b] touchesEnded:withEvent:
2014-06-23 11:13:16.606 TouchTracker[5145:60b] touchesBegan:withEvent:
2014-06-23 11:13:16.910 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:16.926 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:16.943 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:16.960 TouchTracker[5145:60b] touchesMoved:withEvent:
2014-06-23 11:13:16.977 TouchTracker[5145:60b] touchesMoved:withEvent:
```