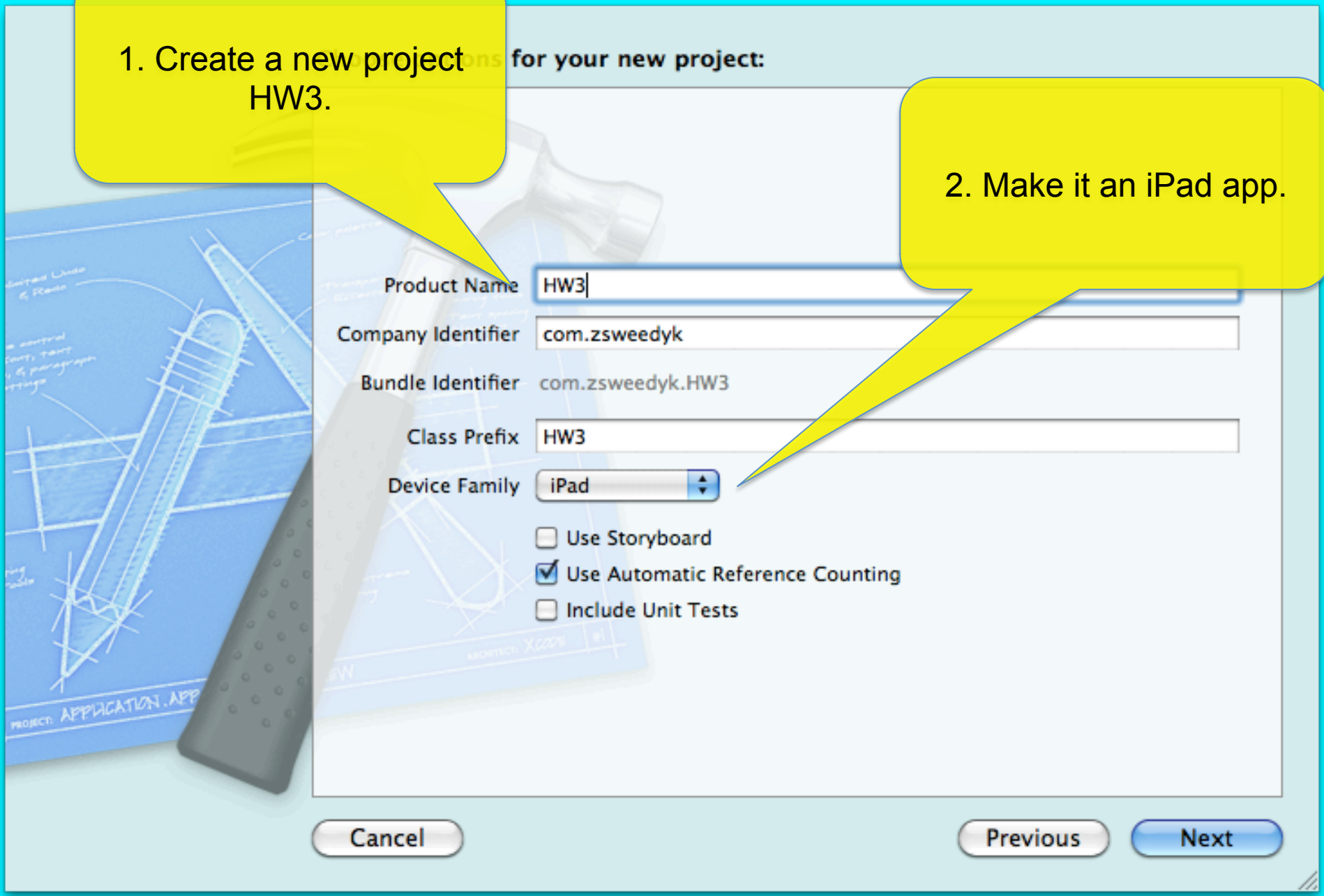
CS121 Tutorial 3

Intro to graphics!

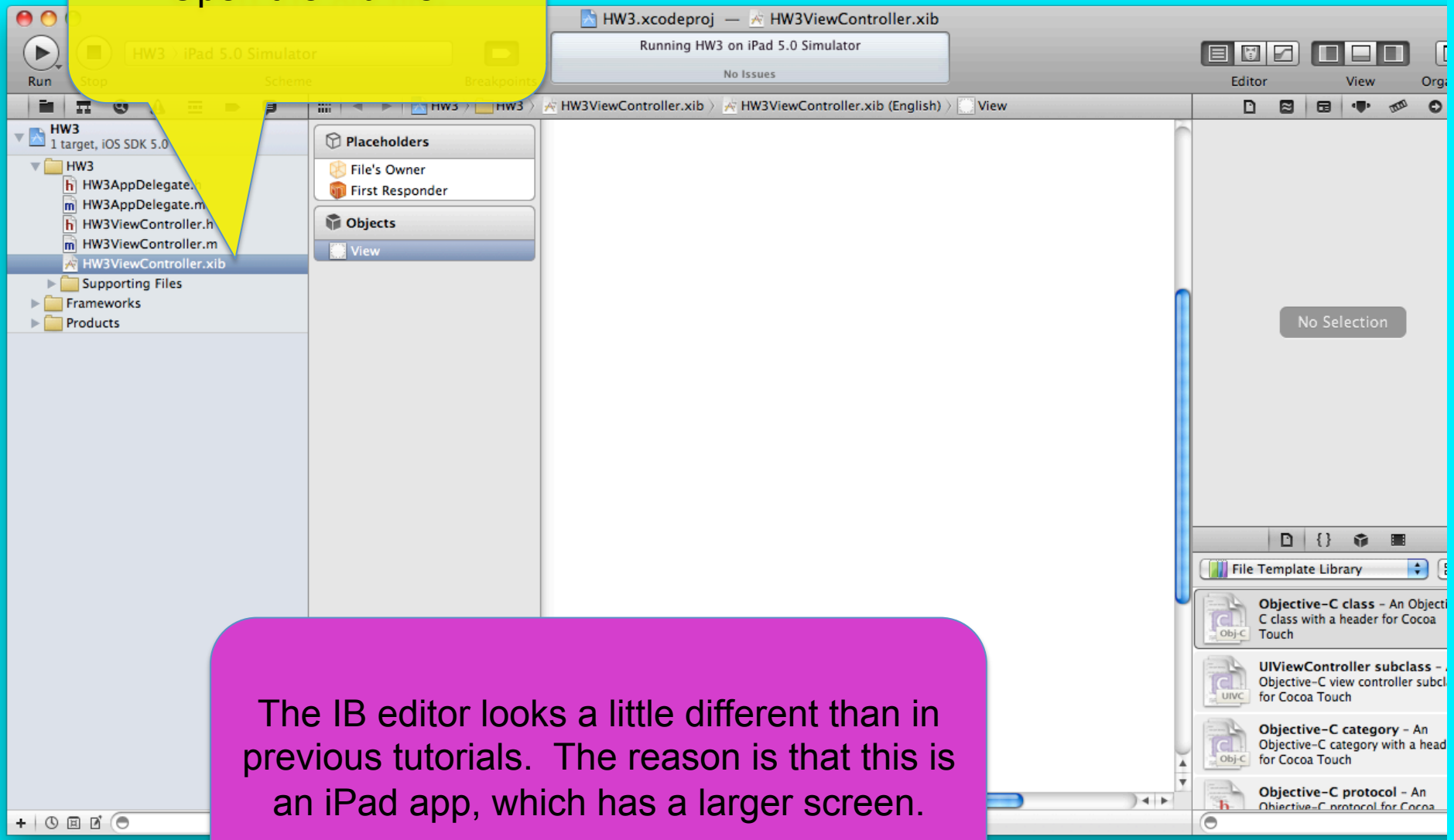
Purple bubbles give you information you'll need to know.

Yellow Bubbles tell you what to do.

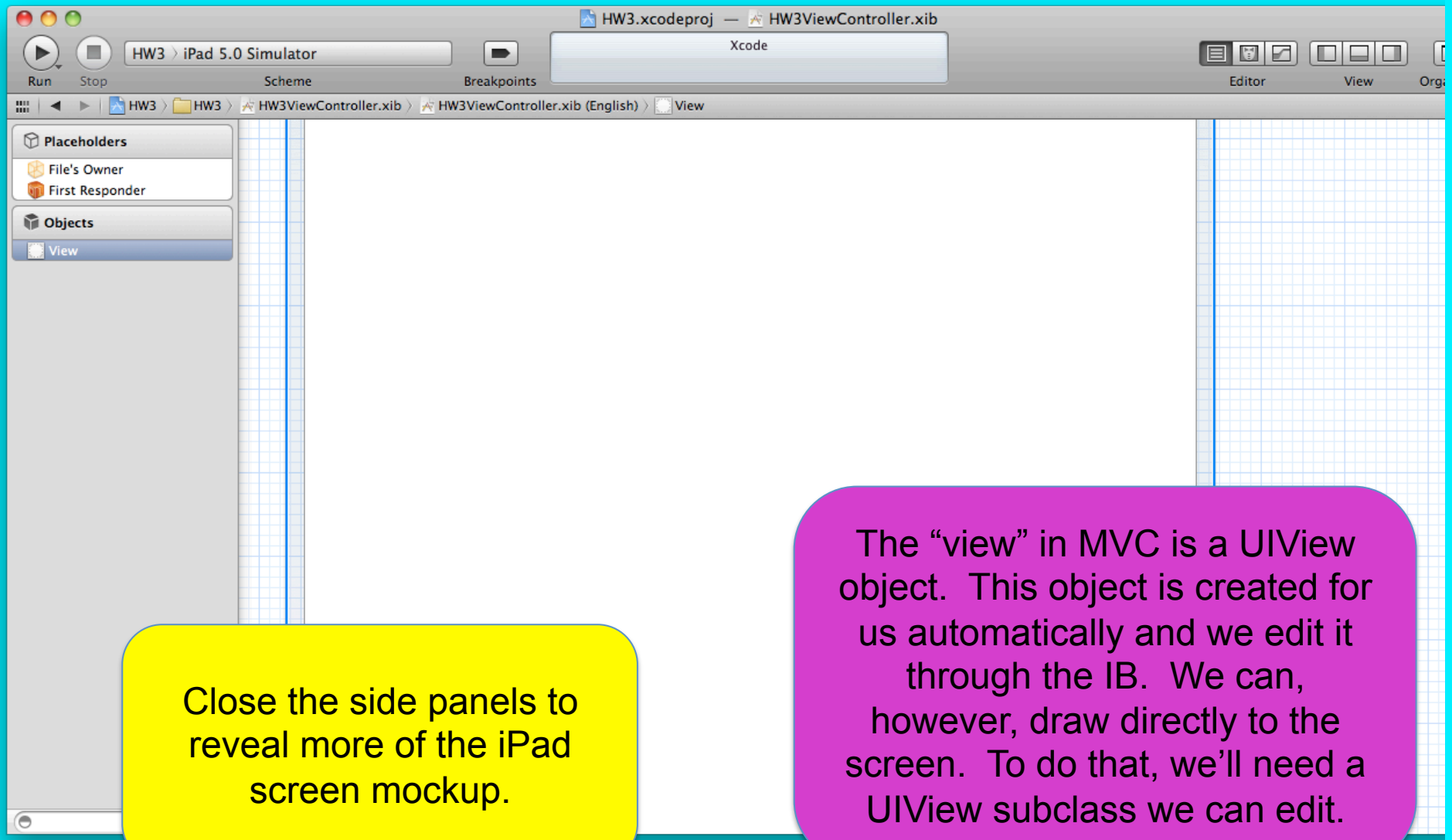
Orange bubbles tell you what you're not expected to understand yet. ☺


1. Create a new project HW3.
for your new project:
2. Make it an iPad app.
Product Name
HW3
Company Identifier
com.zsweedyk
Bundle Identifier
com.zsweedyk.HW3
Class Prefix
HW3
Device Family
iPad
Use Storyboard
Use Automatic Reference Counting
Include Unit Tests
Cancel
Previous
Next

Open the xib file.

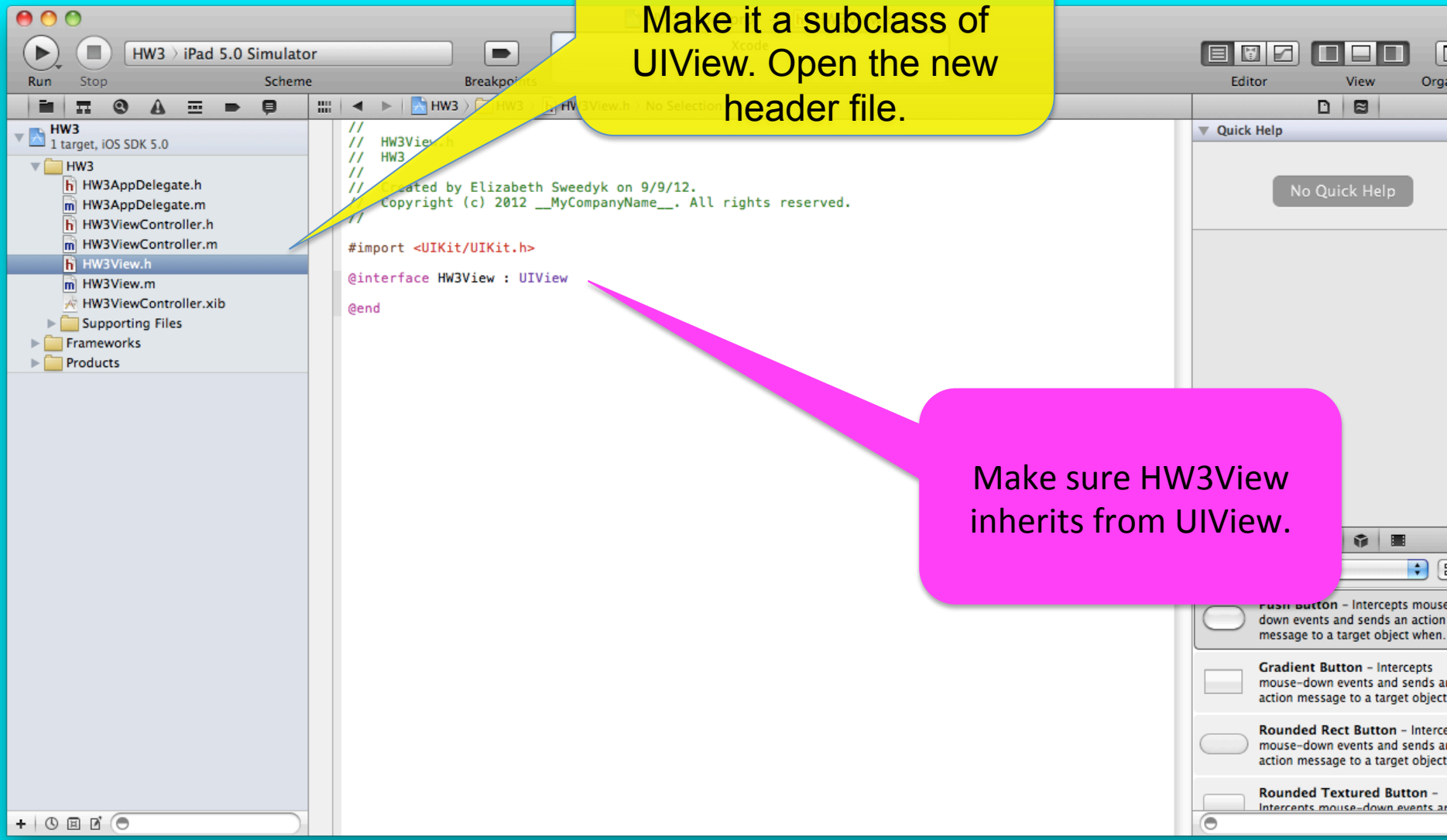


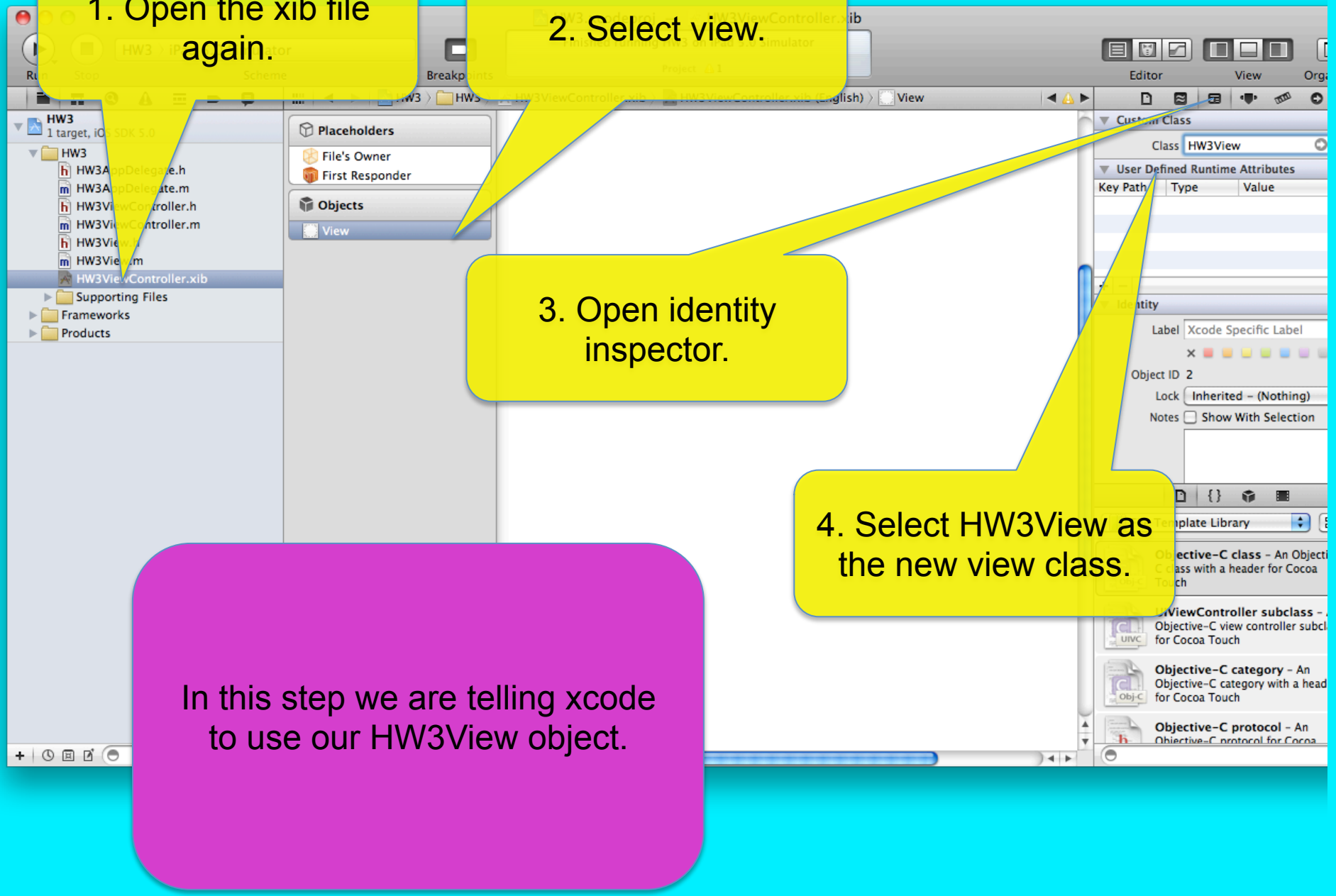
The IB editor looks a little different than in previous tutorials. The reason is that this is an iPad app, which has a larger screen.



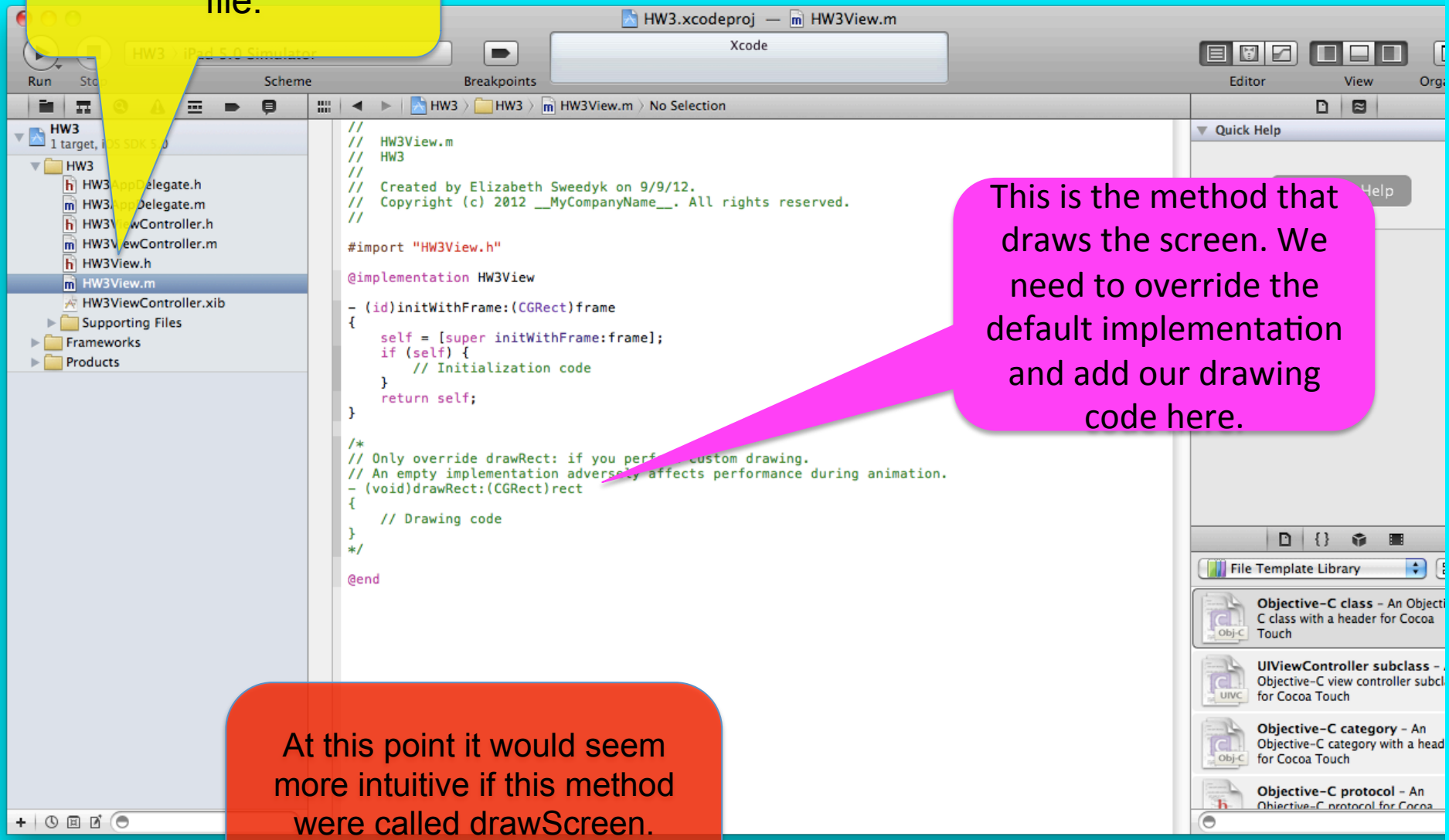
Close the side panels to reveal more of the iPad screen mockup.

The “view” in MVC is a UIView object. This object is created for us automatically and we edit it through the IB. We can, however, draw directly to the screen. To do that, we’ll need a UIView subclass we can edit.


Create a new Objective-c class called HW3View. Make it a subclass of UIView. Open the new header file.
Make sure HW3View inherits from UIView.
HW3 › iPad 5.0 Simulator
Run
Stop
Scheme
Breakpoints
Editor
View
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
HW3ViewController.xib
Supporting Files
Frameworks
Products
//
// HW3View.h
// HW3
//
// Created by Elizabeth Sweedyk on 9/9/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import <UIKit/UIKit.h>
@interface HW3View : UIView
@end
Quick Help
No Quick Help
Push Button – Intercepts mouse down events and sends an action message to a target object when.
Gradient Button – Intercepts mouse-down events and sends a action message to a target object
Rounded Rect Button – Interce mouse-down events and sends a action message to a target object
Rounded Textured Button –

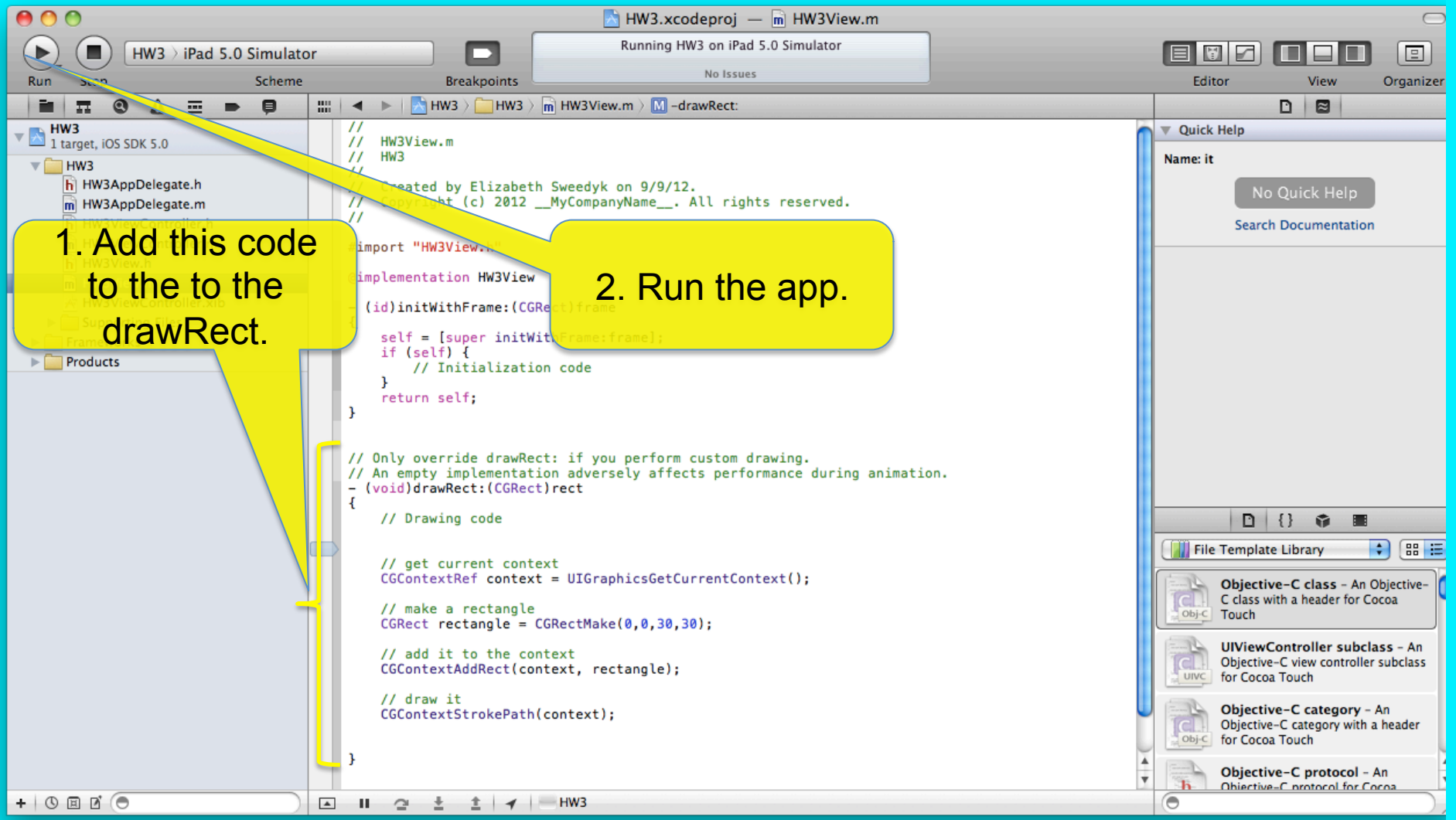

1. Open the xib file again.
2. Select view.
3. Open identity inspector.
4. Select HW3View as the new view class.
In this step we are telling xcode to use our HW3View object.
HW3
HW3ViewController.xib
Supporting Files
Frameworks
Products
Placeholders
File's Owner
First Responder
Objects
View
Class HW3View
User Defined Runtime Attributes
Key Path
Type
Value
Label Xcode Specific Label
Object ID 2
Lock Inherited - (Nothing)
Notes Show With Selection
Objective-C class
Objective-C category
Objective-C protocol

Open the HW3View.m file.

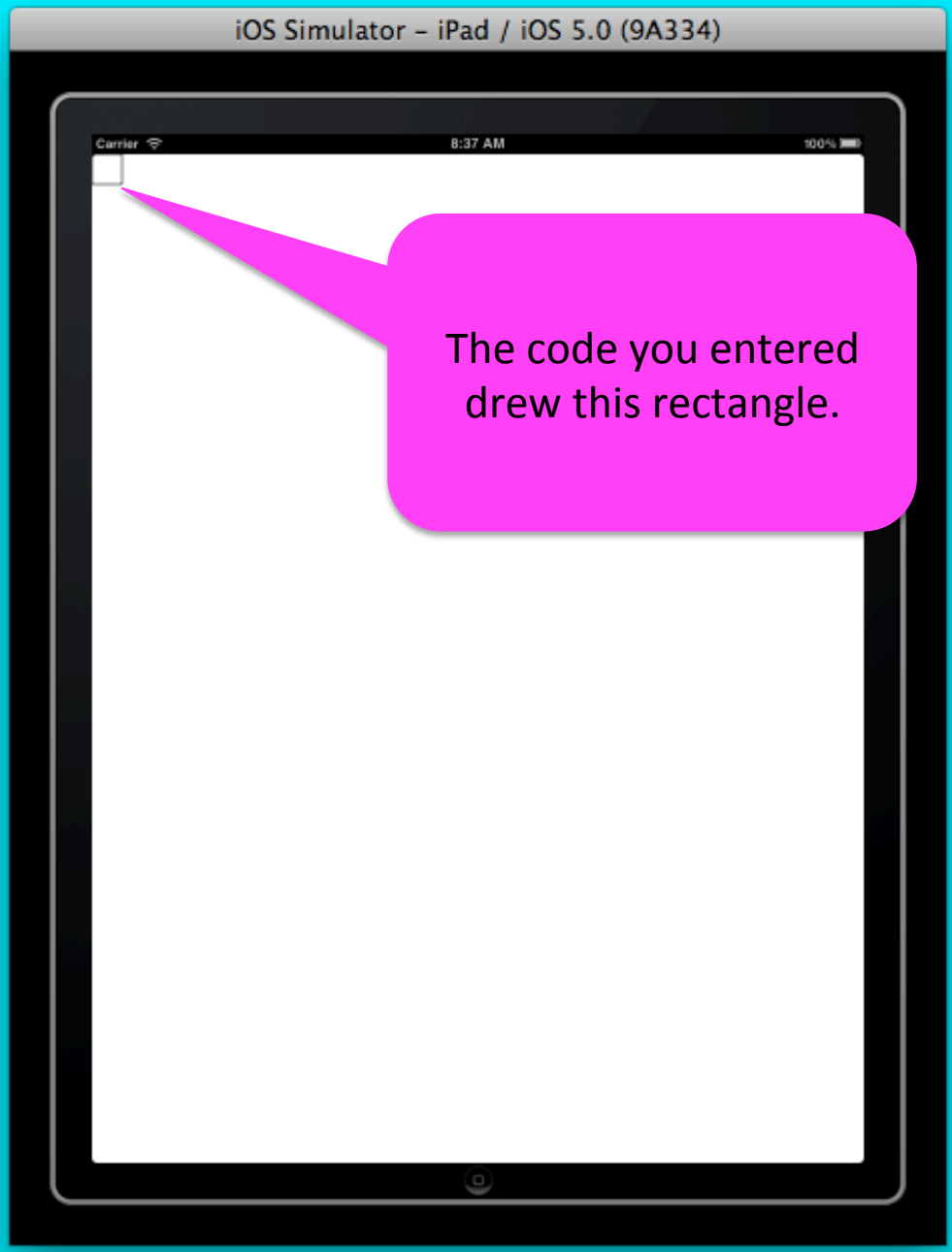


This is the method that draws the screen. We need to override the default implementation and add our drawing code here.

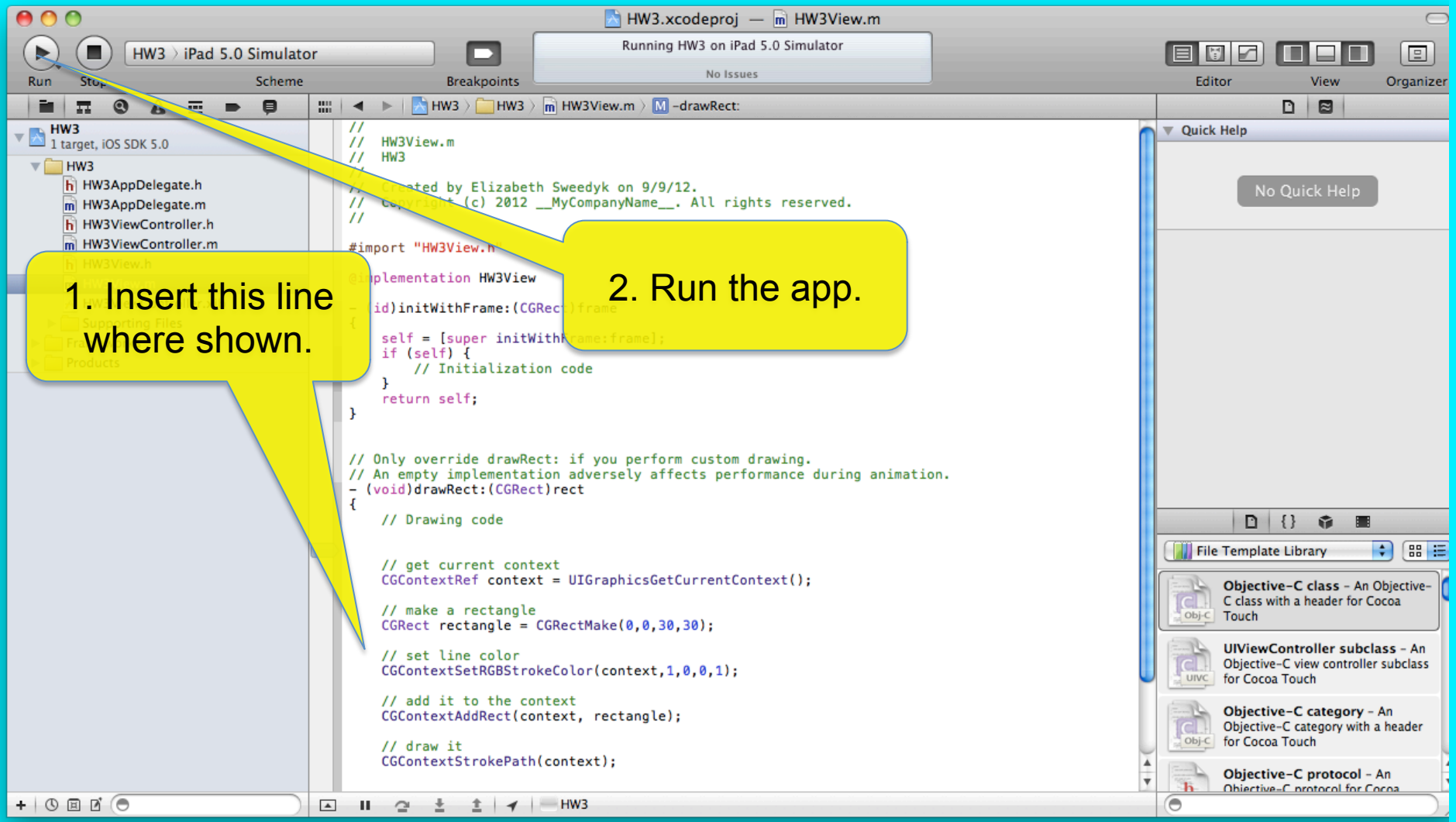
At this point it would seem more intuitive if this method were called drawScreen. Later on we'll see why drawRect makes sense.


HW3.xcodeproj — HW3View.m
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Run
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
Products
HW3 › HW3 › HW3View.m › -drawRect:
1. Add this code to the to the drawRect.
2. Run the app.
// HW3View.m
// HW3
// Created by Elizabeth Sweedyk on 9/9/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
implementation HW3View
(id)initWithFrame:(CGRect)frame
self = [super initWithFrame:frame];
if (self) {
// Initialization code
}
return self;
}
// Only override drawRect: if you perform custom drawing.
// An empty implementation adversely affects performance during animation.
- (void)drawRect:(CGRect)rect
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextStrokePath(context);
}
Quick Help
Name: it
No Quick Help
Search Documentation
File Template Library
Objective-C class – An Objective-C class with a header for Cocoa Touch
UIViewController subclass – An Objective-C view controller subclass for Cocoa Touch
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An
HW3

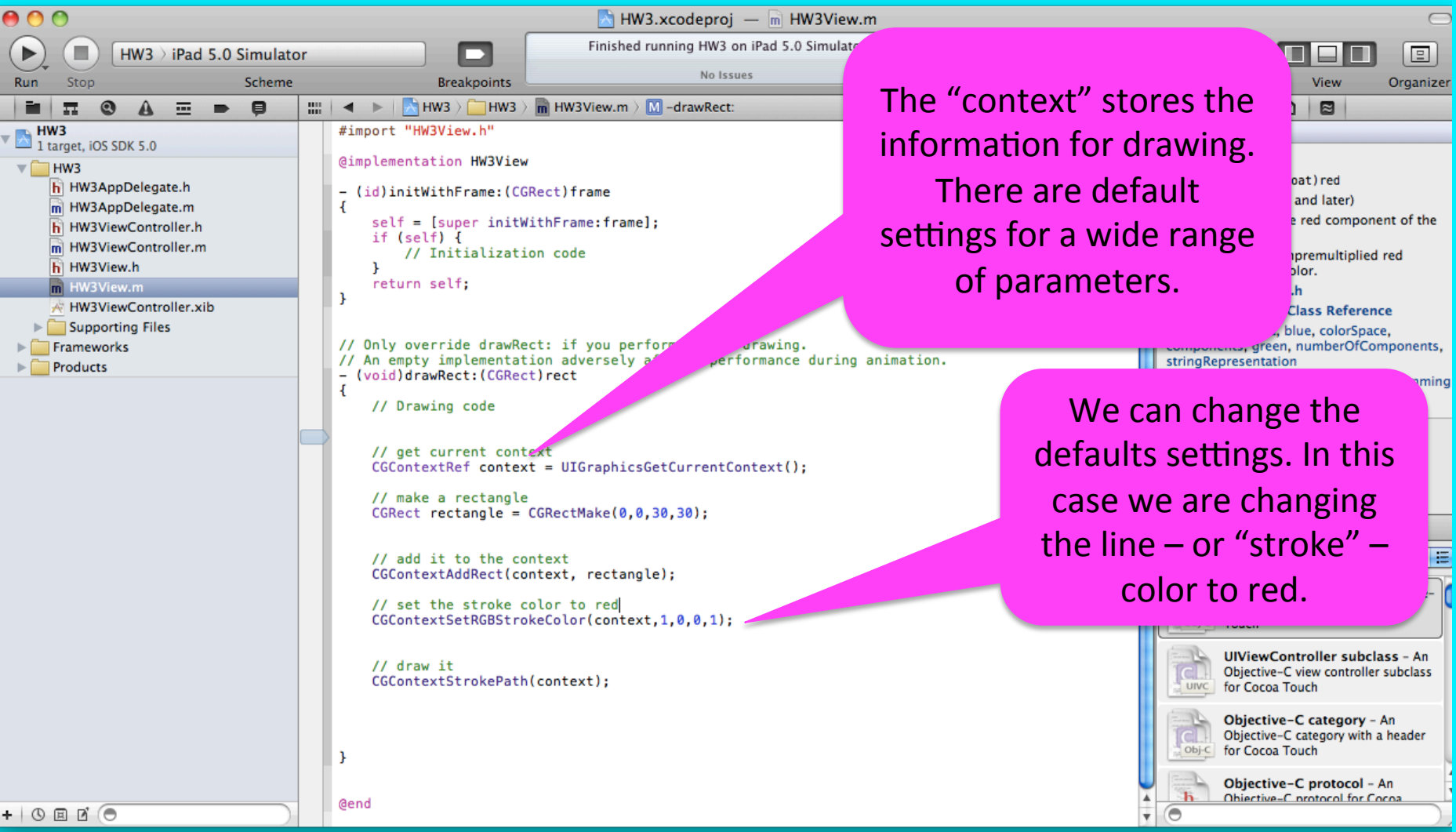
The iPad simulator should appear on your desktop.



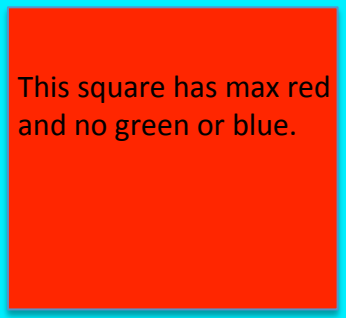
The code you entered drew this rectangle.


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3 › HW3 › HW3View.m › -drawRect:
//
// HW3View.m
// HW3
// Created by Elizabeth Sweedyk on 9/9/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import "HW3View.h"
@implementation HW3View
- (id)initWithFrame:(CGRect)frame
{
self = [super initWithFrame:frame];
if (self) {
// Initialization code
}
return self;
}
// Only override drawRect: if you perform custom drawing.
// An empty implementation adversely affects performance during animation.
- (void)drawRect:(CGRect)rect
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// set line color
CGContextSetRGBStrokeColor(context,1,0,0,1);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextStrokePath(context);
Quick Help
No Quick Help
File Template Library
Objective-C class – An Objective-C class with a header for Cocoa Touch
UIViewController subclass – An Objective-C view controller subclass for Cocoa Touch
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An
1. Insert this line where shown.
2. Run the app.


iOS Simulator – iPad / iOS 5.0 (9A334)
Carrier
8:42 AM
100%
Now the rectangle is red!


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Finished running HW3 on iPad 5.0 Simulator
No Issues
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
HW3ViewController.xib
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
#import "HW3View.h"
@implementation HW3View
- (id)initWithFrame:(CGRect)frame
{
self = [super initWithFrame:frame];
if (self) {
// Initialization code
}
return self;
}
// Only override drawRect: if you perfor drawing.
// An empty implementation adversely a performance during animation.
- (void)drawRect:(CGRect)rect
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// add it to the context
CGContextAddRect(context, rectangle);
// set the stroke color to red
CGContextSetRGBStrokeColor(context,1,0,0,1);
// draw it
CGContextStrokePath(context);
}
@end
The “context” stores the information for drawing. There are default settings for a wide range of parameters.
We can change the defaults settings. In this case we are changing the line – or “stroke” – color to red.
UIViewController subclass – An Objective-C view controller subclass for Cocoa Touch
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An

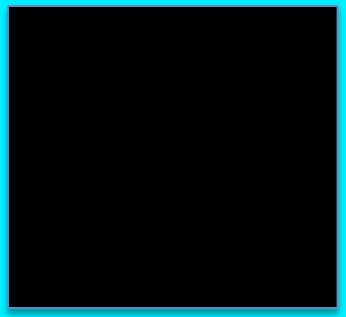
iOS uses an RGB color model. A color is specified in terms of its intensity in three distinct channels red, green, blue. Full intensity is 1 and completely off is 0.



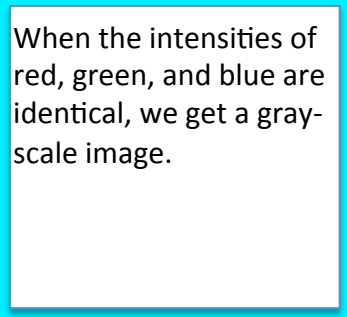
(1,0,0)

(0,1,0)

(0,0,1)

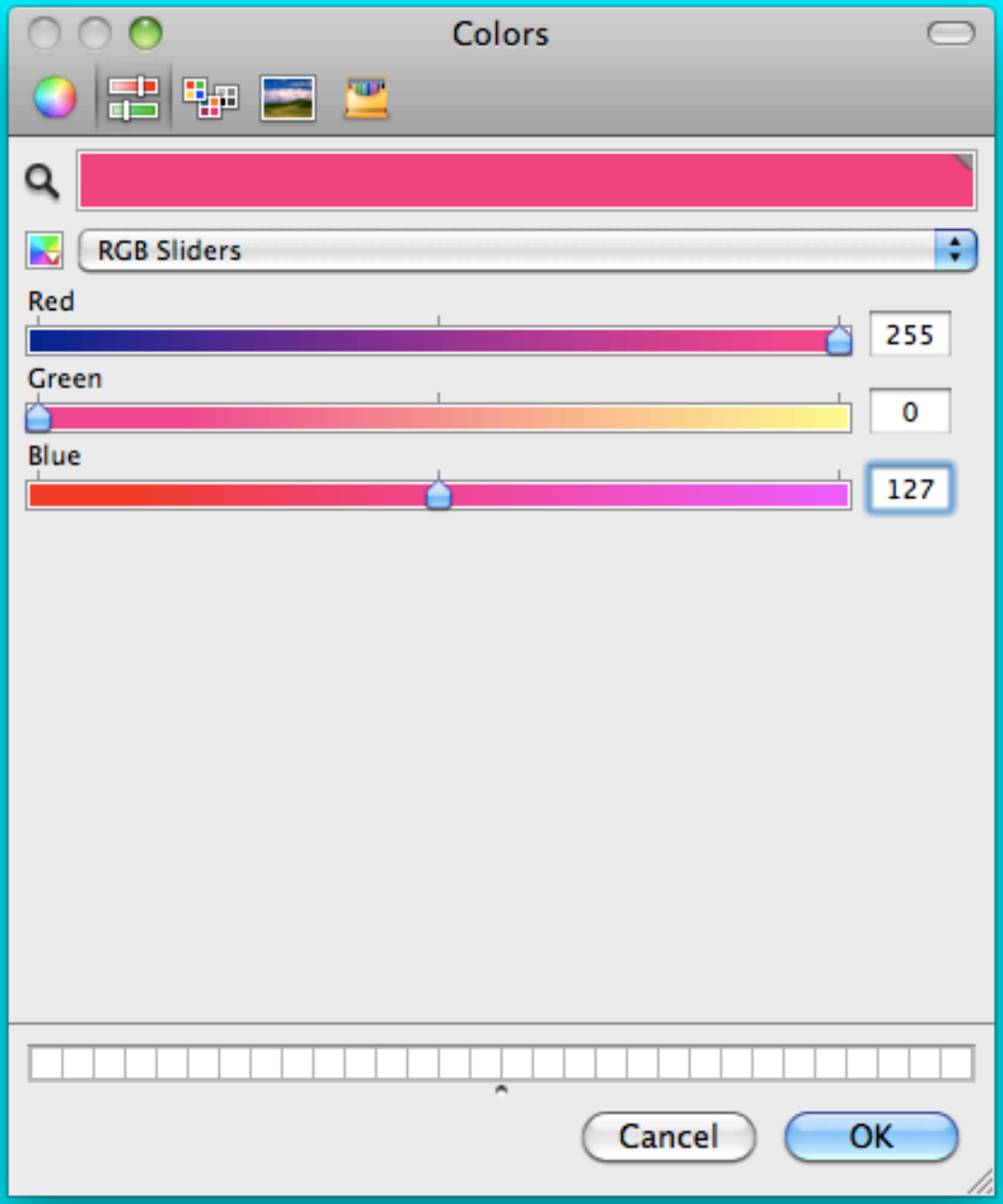


(0,0,0)



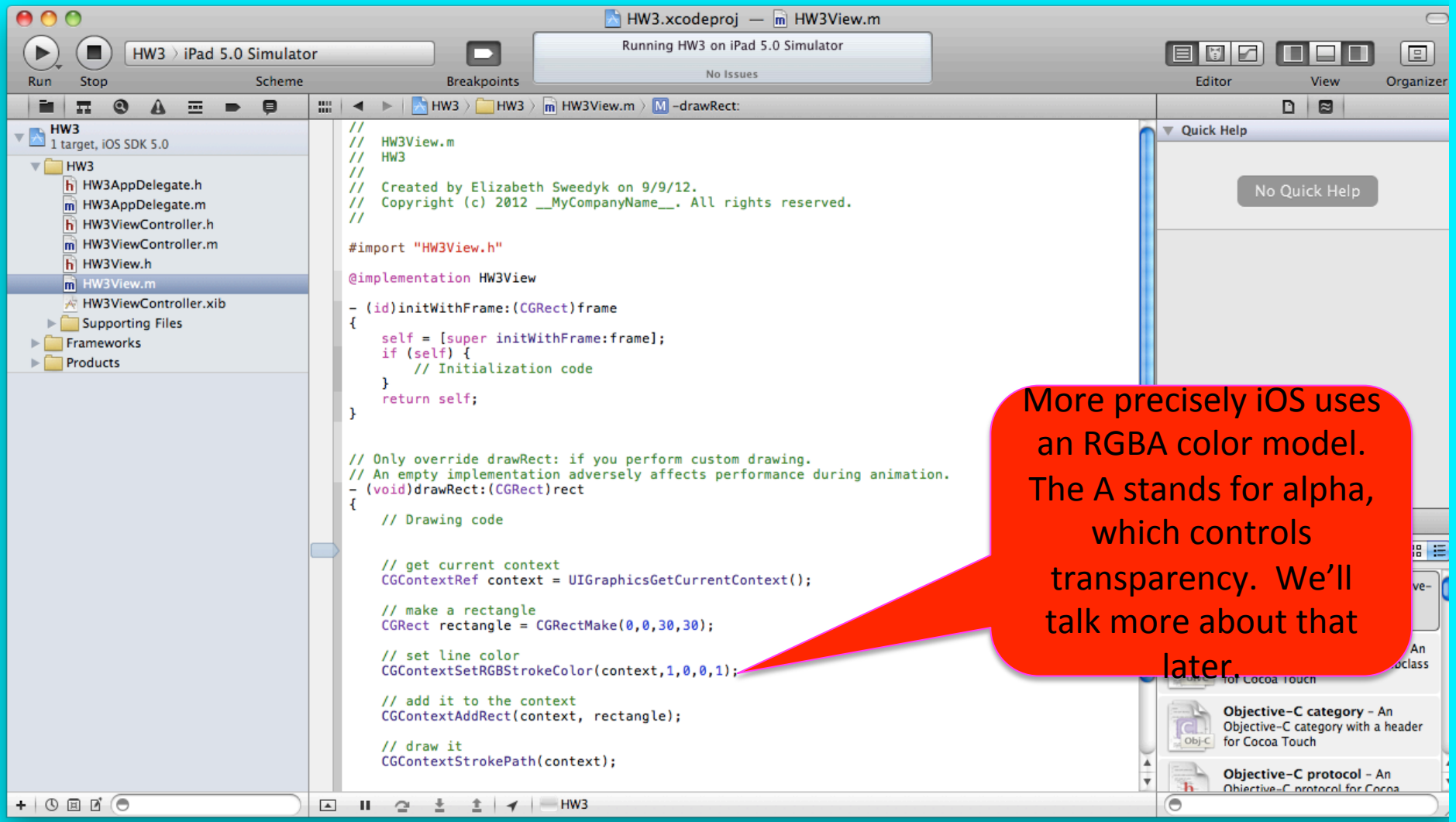
(1,1,1)

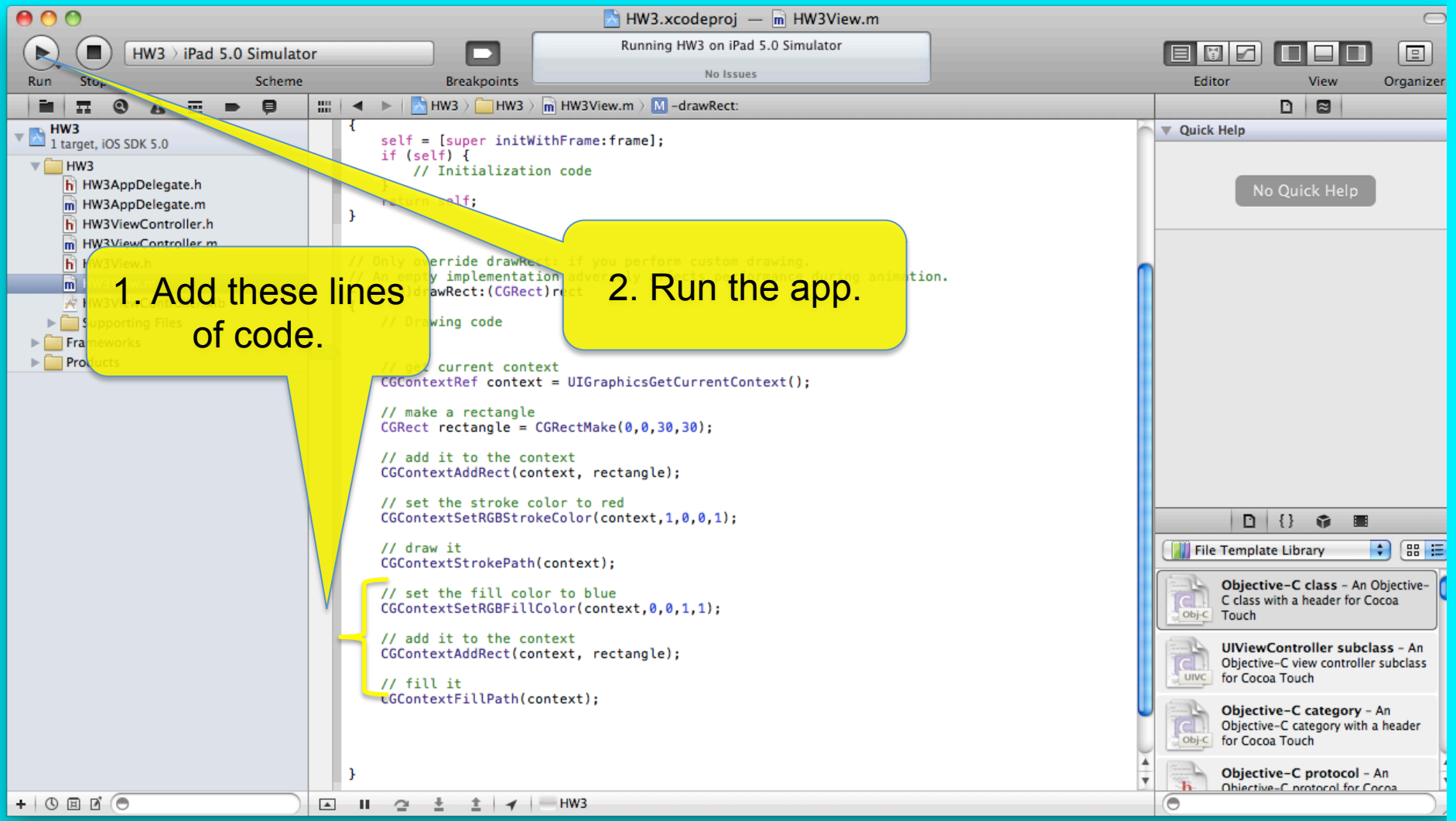
(1,0,.5)

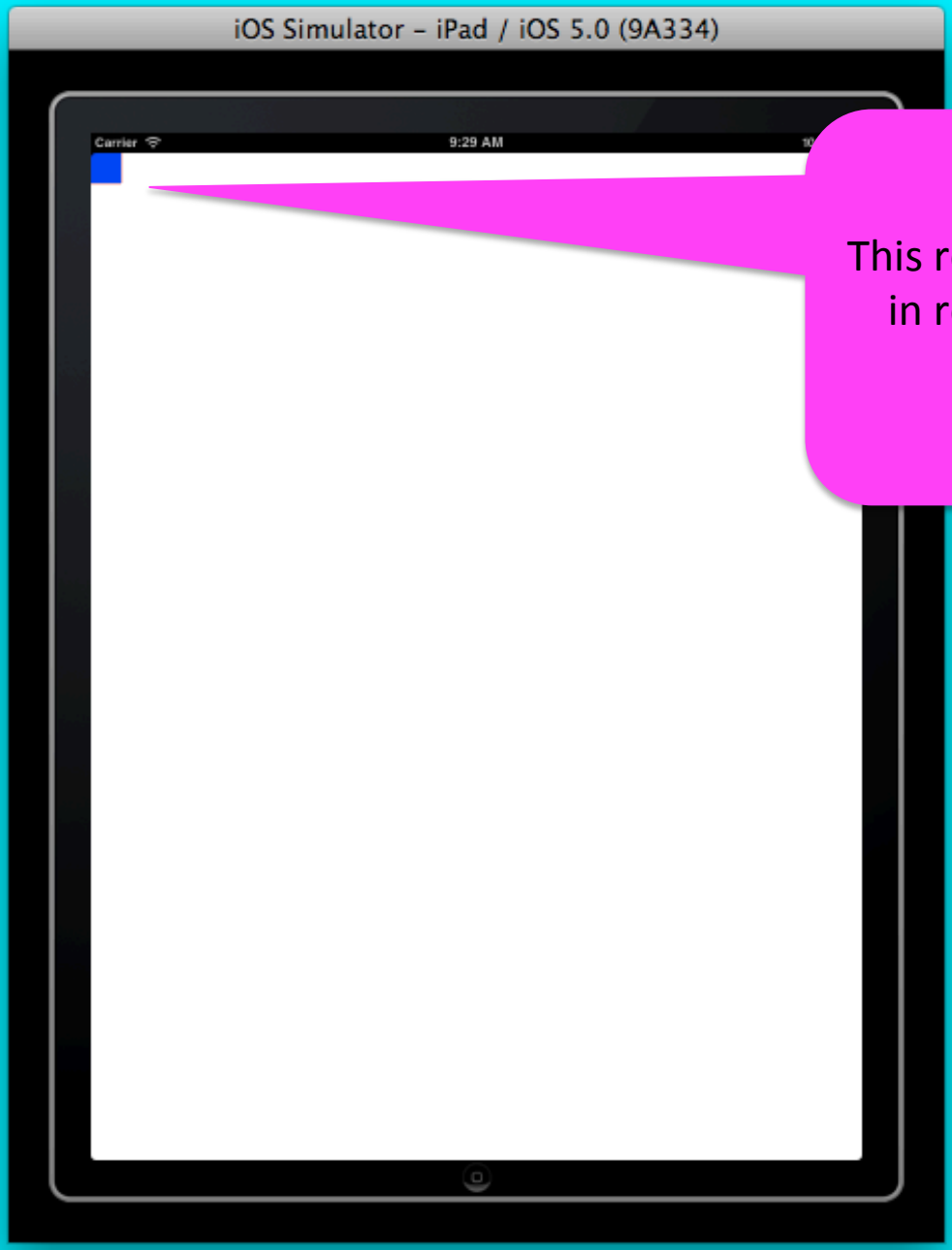


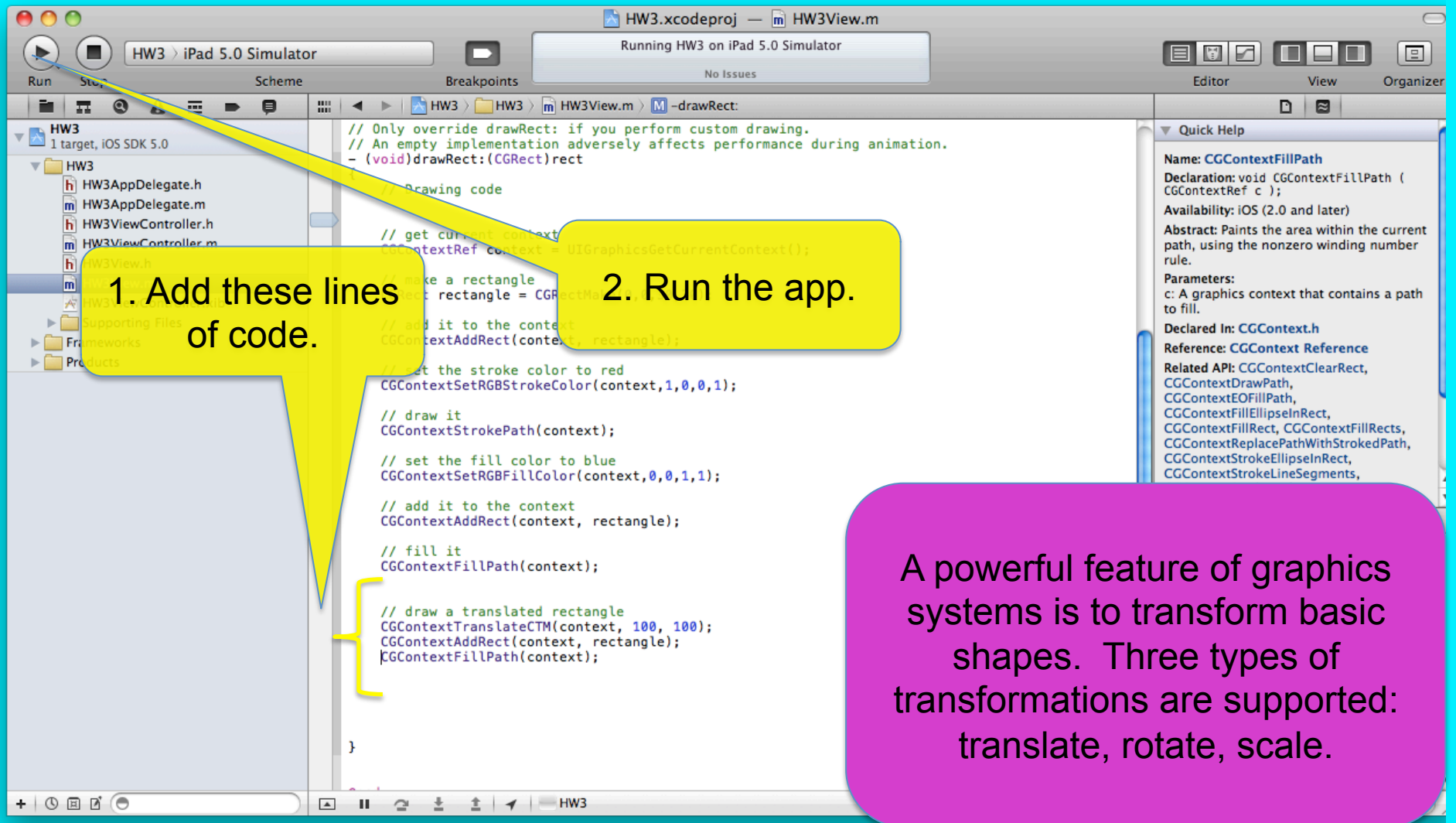
Another RGB convention is to specify color intensity as a number in [0,255]. This is a screen shot from powerpoint, which allows users to create colors using sliders.

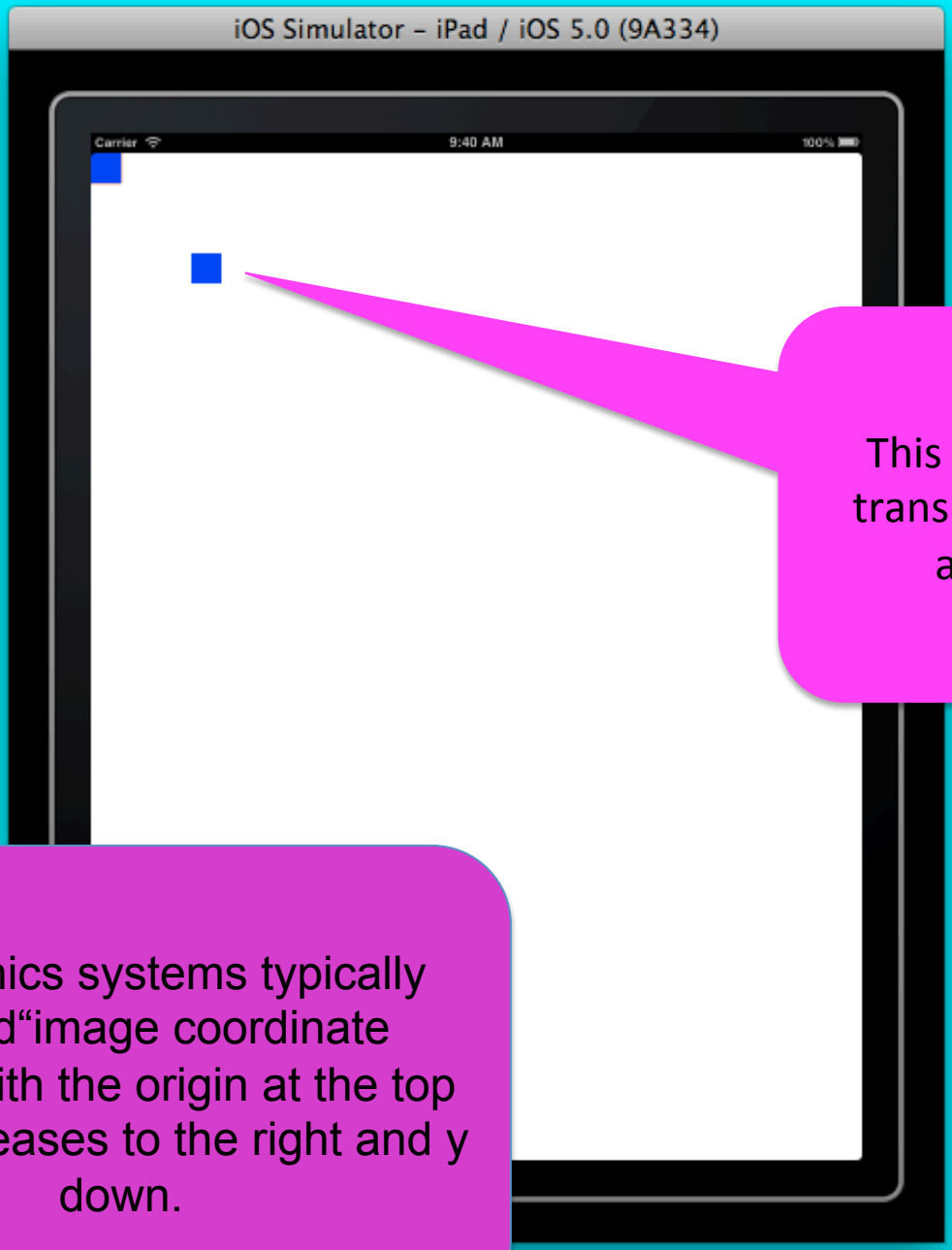
(You can judge for yourself the effectiveness of this UI, particularly the colors on the sliders.)


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
HW3ViewController.xib
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
//
// HW3View.m
// HW3
//
// Created by Elizabeth Sweedyk on 9/9/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import "HW3View.h"
@implementation HW3View
- (id)initWithFrame:(CGRect)frame
{
self = [super initWithFrame:frame];
if (self) {
// Initialization code
}
return self;
}
// Only override drawRect: if you perform custom drawing.
// An empty implementation adversely affects performance during animation.
- (void)drawRect:(CGRect)rect
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// set line color
CGContextSetRGBStrokeColor(context,1,0,0,1);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextStrokePath(context);
Quick Help
No Quick Help
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An
HW3
More precisely iOS uses an RGBA color model. The A stands for alpha, which controls transparency. We'll talk more about that later.


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
{
self = [super initWithFrame:frame];
if (self) {
// Initialization code
}
return self;
}
// Only override drawRect: if you perform custom drawing.
// An empty implementation adversely affects performance during animation.
- (void)drawRect:(CGRect)rect
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// add it to the context
CGContextAddRect(context, rectangle);
// set the stroke color to red
CGContextSetRGBStrokeColor(context,1,0,0,1);
// draw it
CGContextStrokePath(context);
// set the fill color to blue
CGContextSetRGBFillColor(context,0,0,1,1);
// add it to the context
CGContextAddRect(context, rectangle);
// fill it
CGContextFillPath(context);
}
HW3
Quick Help
No Quick Help
File Template Library
Objective-C class – An Objective-C class with a header for Cocoa Touch
UIViewController subclass – An Objective-C view controller subclass for Cocoa Touch
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An
1. Add these lines of code.
2. Run the app.

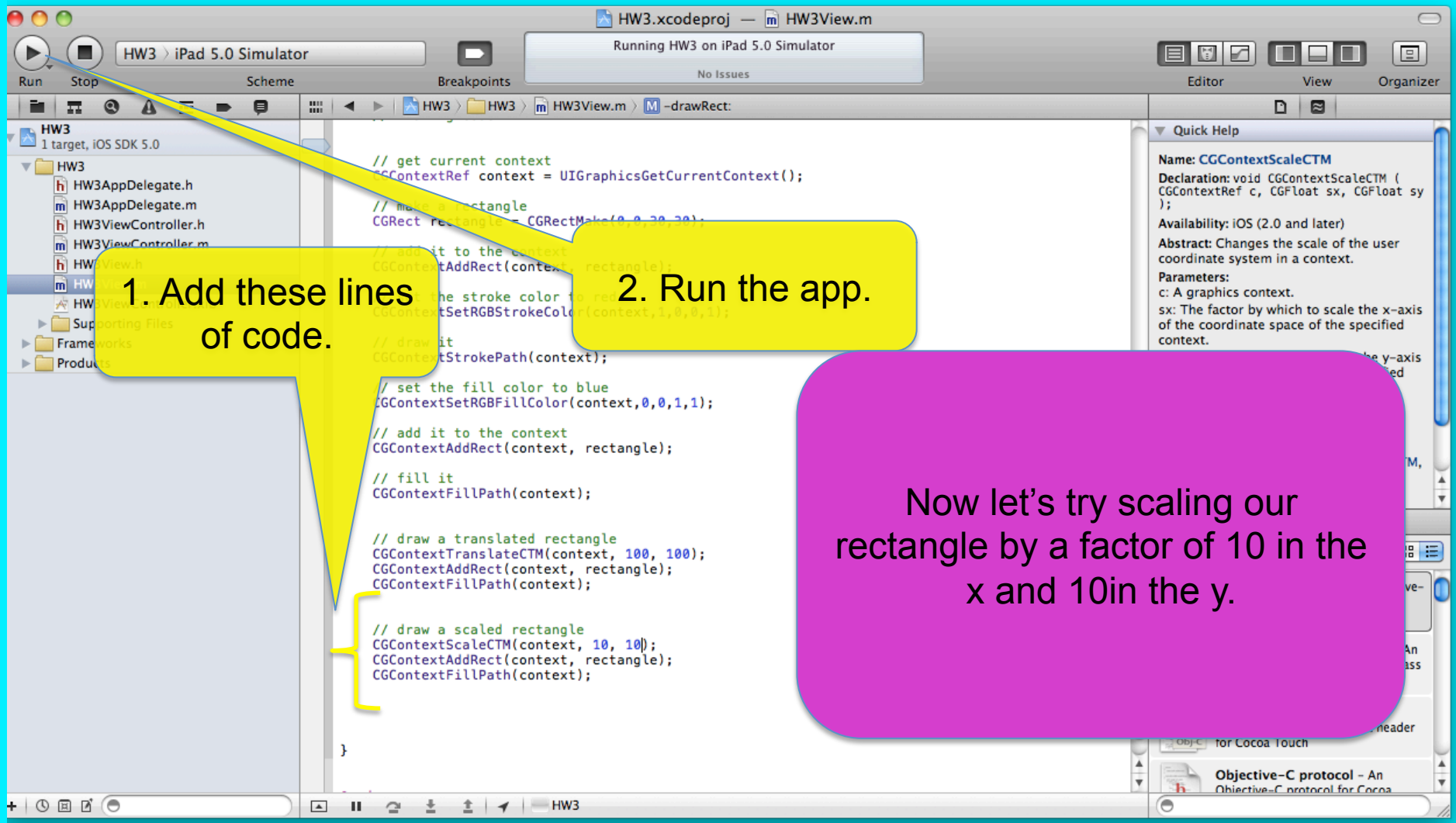

iOS Simulator – iPad / iOS 5.0 (9A334)
Carrier
9:29 AM
This rectangle is outlined in red and filled with blue.

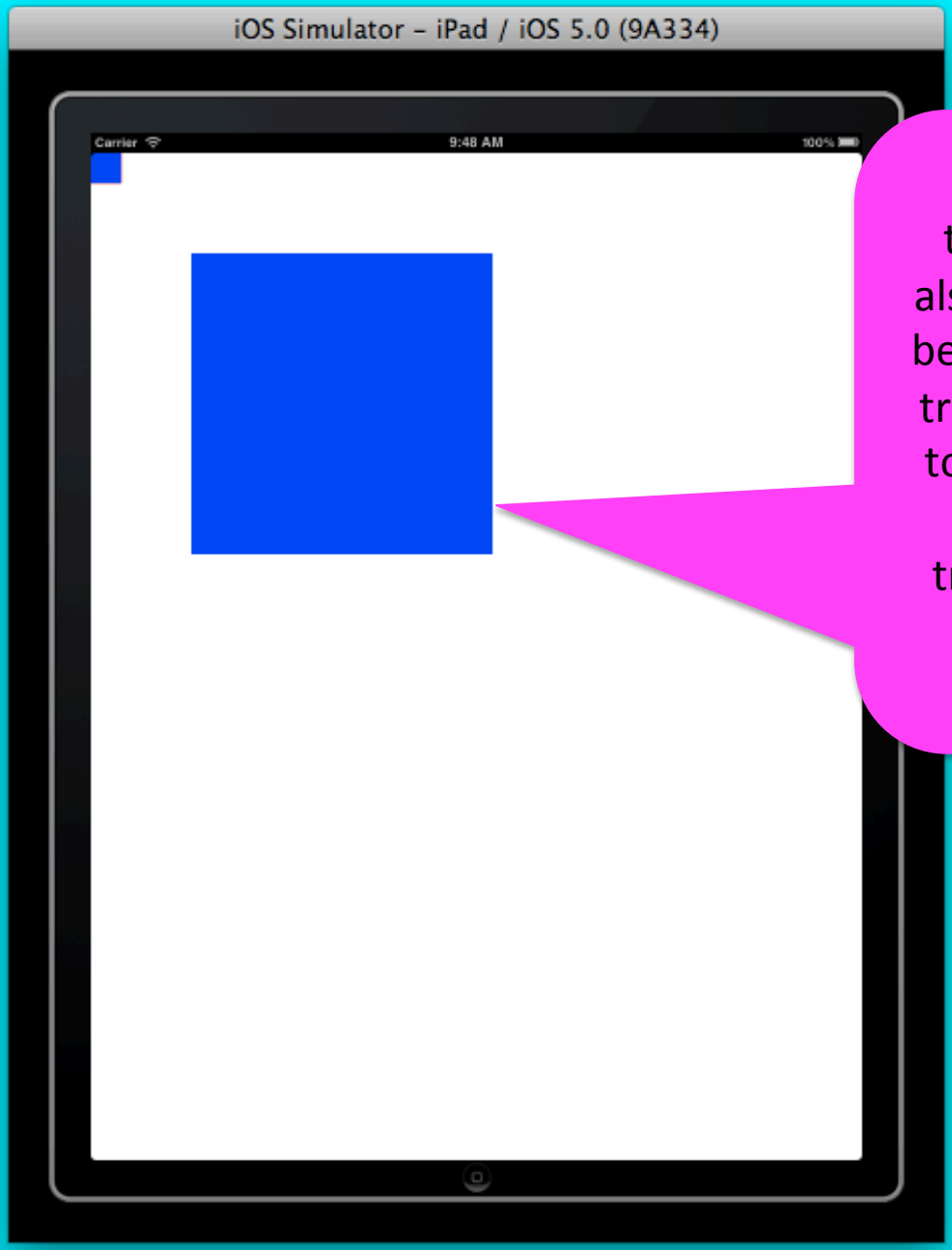

HW3.xcodeproj — HW3View.m
Run
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
// Only override drawRect: if you perform custom drawing.
// An empty implementation adversely affects performance during animation.
- (void)drawRect:(CGRect)rect
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// make a rectangle
CGRect rectangle = CGRectMake(60,170,200,80);
// add it to the context
CGContextAddRect(context, rectangle);
// set the stroke color to red
CGContextSetRGBStrokeColor(context,1,0,0,1);
// draw it
CGContextStrokePath(context);
// set the fill color to blue
CGContextSetRGBFillColor(context,0,0,1,1);
// add it to the context
CGContextAddRect(context, rectangle);
// fill it
CGContextFillPath(context);
// draw a translated rectangle
CGContextTranslateCTM(context, 100, 100);
CGContextAddRect(context, rectangle);
CGContextFillPath(context);
}
Quick Help
Name: CGContextFillPath
Declaration: void CGContextFillPath (CGContextRef c);
Availability: iOS (2.0 and later)
Abstract: Paints the area within the current path, using the nonzero winding number rule.
Parameters:
c: A graphics context that contains a path to fill.
Declared In: CGContext.h
Reference: CGContext Reference
Related API: CGContextClearRect, CGContextDrawPath, CGContextEOFillPath, CGContextFillEllipseInRect, CGContextFillRect, CGContextFillRects, CGContextReplacePathWithStrokedPath, CGContextStrokeEllipseInRect, CGContextStrokeLineSegments,
HW3
1. Add these lines of code.
2. Run the app.
A powerful feature of graphics systems is to transform basic shapes. Three types of transformations are supported: translate, rotate, scale.



This is our rectangle translated by 100 in x and 100 in y.

2D graphics systems typically use and“image coordinate system” with the origin at the top left. x increases to the right and y down.


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Editor
View
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3 › HW3 › HW3View.m › -drawRect:
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextStrokePath(context);
// set the fill color to blue
CGContextSetRGBFillColor(context,0,0,1,1);
// add it to the context
CGContextAddRect(context, rectangle);
// fill it
CGContextFillPath(context);
// draw a translated rectangle
CGContextTranslateCTM(context, 100, 100);
CGContextAddRect(context, rectangle);
CGContextFillPath(context);
// draw a scaled rectangle
CGContextScaleCTM(context, 10, 10);
CGContextAddRect(context, rectangle);
CGContextFillPath(context);
}
Quick Help
Name: CGContextScaleCTM
Declaration: void CGContextScaleCTM (CGContextRef c, CGFloat sx, CGFloat sy);
Availability: iOS (2.0 and later)
Abstract: Changes the scale of the user coordinate system in a context.
Parameters:
c: A graphics context.
sx: The factor by which to scale the x-axis of the coordinate space of the specified context.
Objective-C protocol - An
1. Add these lines of code.
2. Run the app.
Now let's try scaling our rectangle by a factor of 10 in the x and 10in the y.



This is our scaled triangle. Notice it is also translated. That is because both scale and translate were applied to the third rectangle. (Note the second triangle is behind the third one, so we can’tsee it.

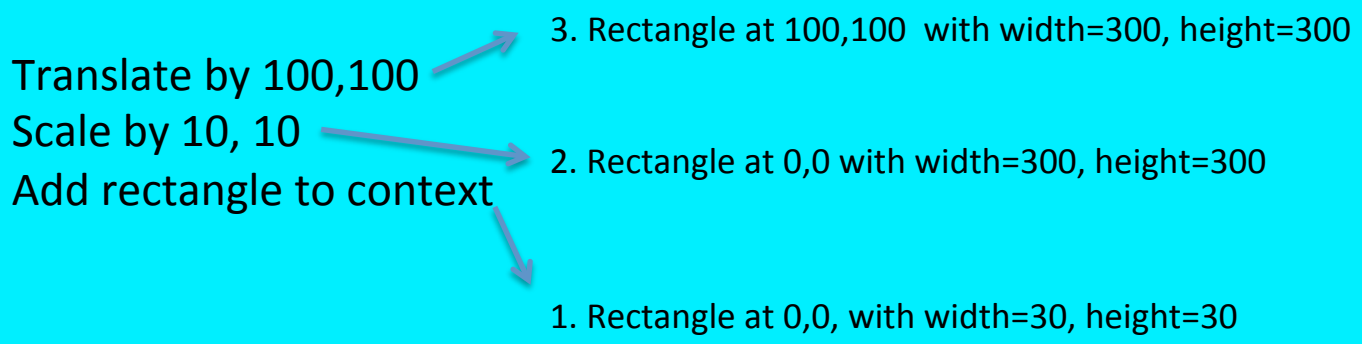
Current transformation matrix (CTM)

The CTM remembers the transformations that have been defined when we add our rectangle to the context:

Translate by 100,100
Scale by 10, 10
Add rectangle to context

This seems unintuitive at first. But it will make sense later. (Hopefully after the next tutorial.)

When we draw a rectangle, these transforms are applied in *reverse* order!



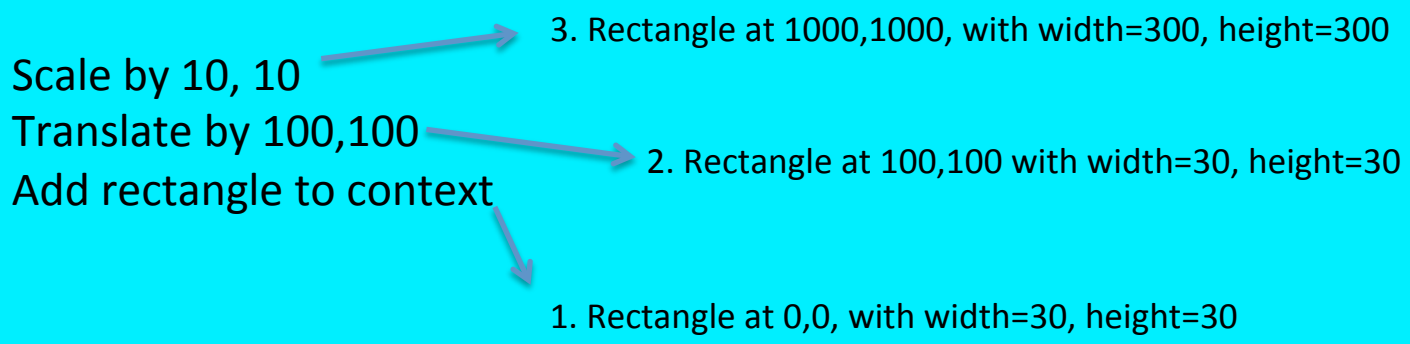
Current transformation matrix (CTM)

Suppose we reversed the transforms:

Scale by 10, 10
Translate by 100,100

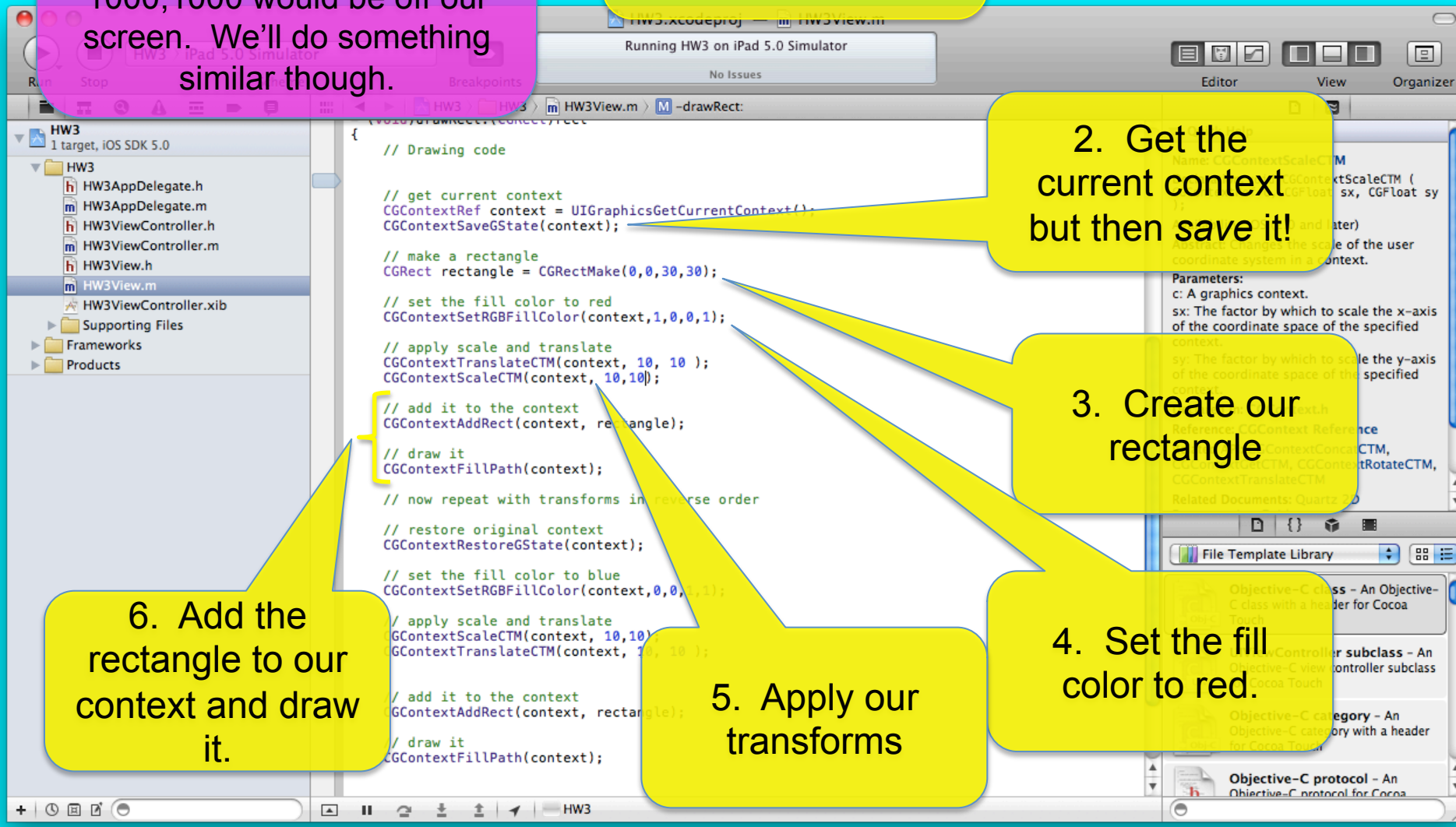
Add rectangle to context

When we draw a rectangle, these transforms are applied in *reverse* order!



We can't do the experiment described on the last slide because a rectangle at 1000,1000 would be off our screen. We'll do something similar though.

1. Let start over. Delete the code we've written so far.



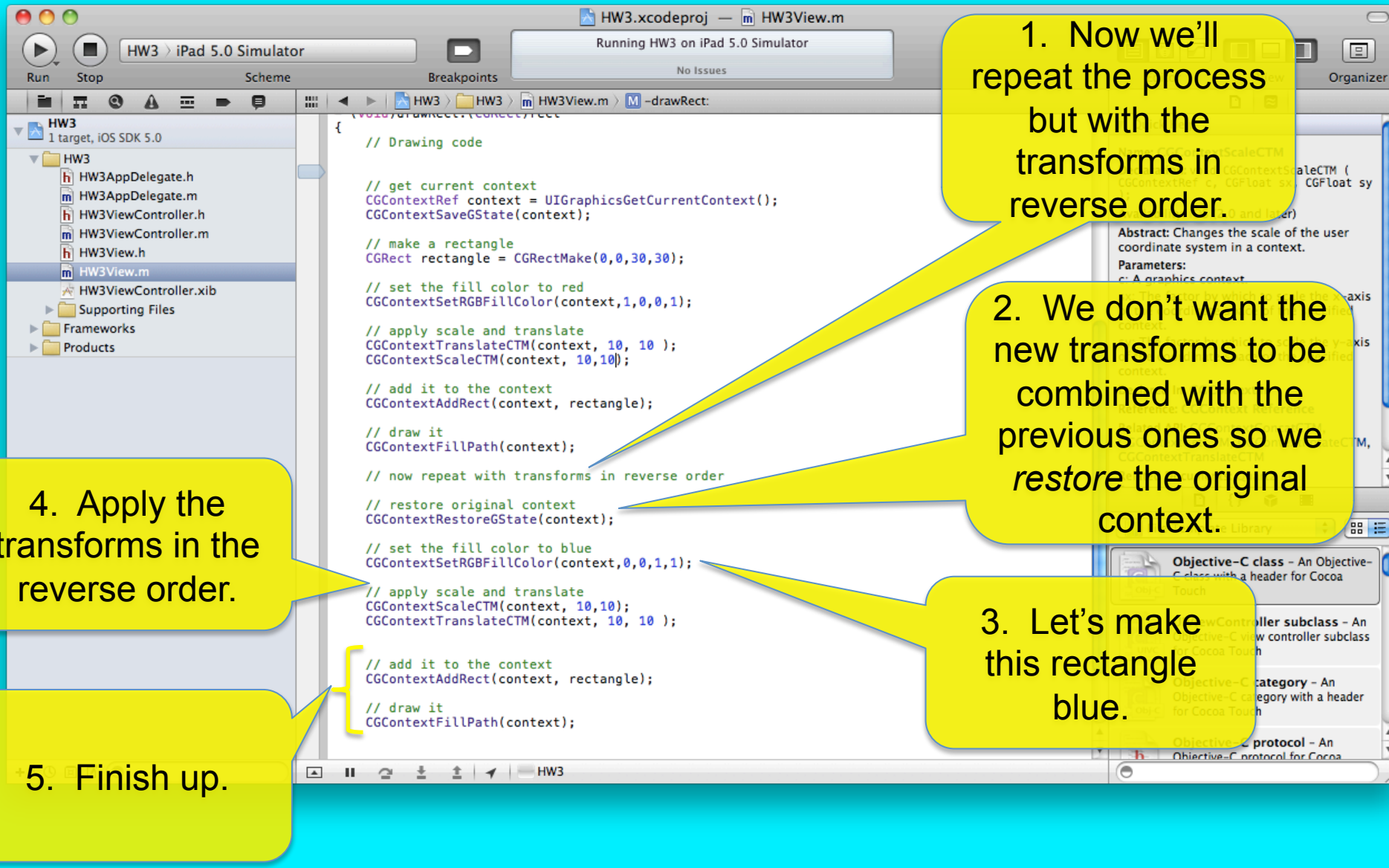
2. Get the current context but then *save* it!

3. Create our rectangle

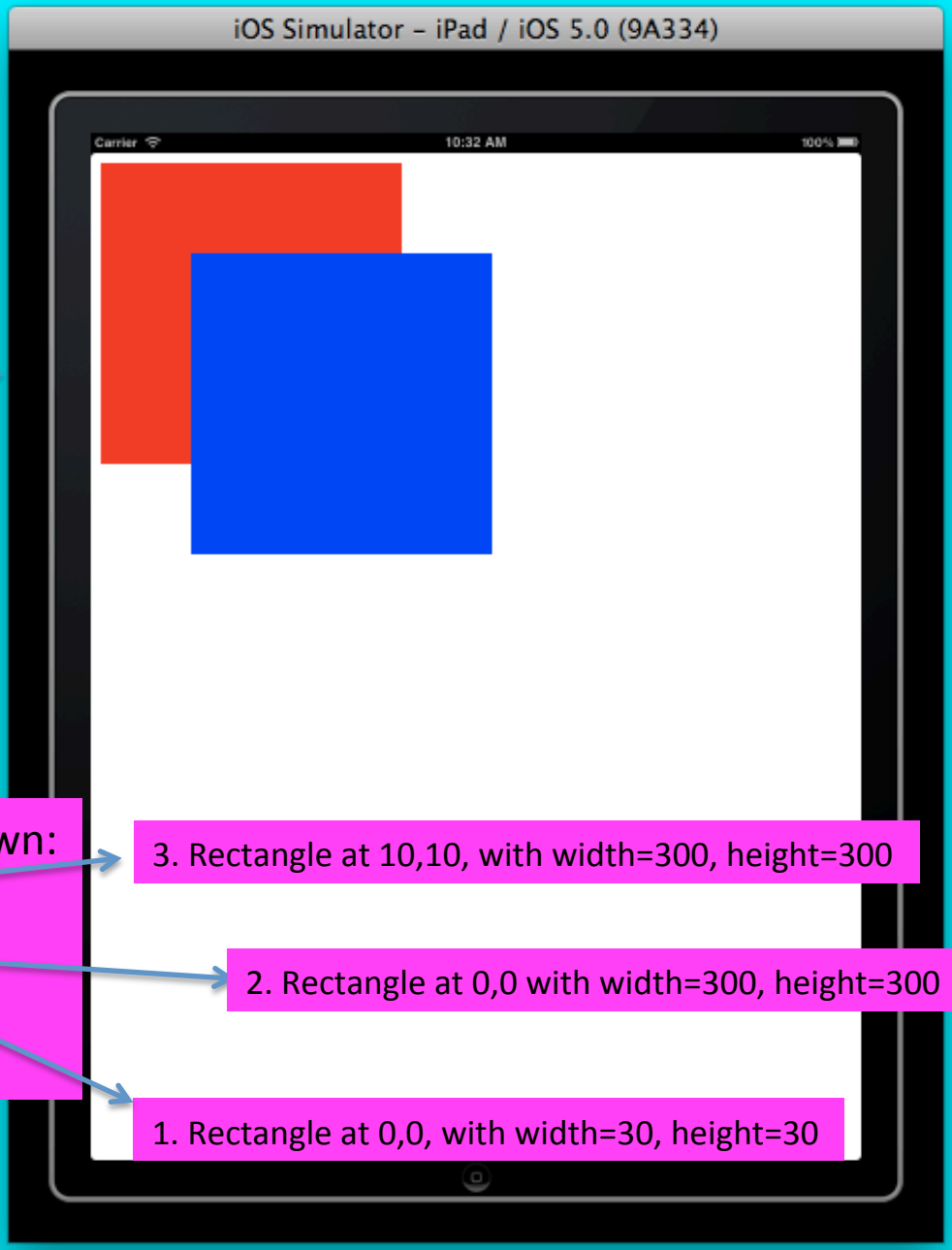
4. Set the fill color to red.

5. Apply our transforms

6. Add the rectangle to our context and draw it.


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
HW3ViewController.xib
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
CGContextSaveGState(context);
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// set the fill color to red
CGContextSetRGBFillColor(context,1,0,0,1);
// apply scale and translate
CGContextTranslateCTM(context, 10, 10);
CGContextScaleCTM(context, 10,10);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextFillPath(context);
// now repeat with transforms in reverse order
// restore original context
CGContextRestoreGState(context);
// set the fill color to blue
CGContextSetRGBFillColor(context,0,0,1,1);
// apply scale and translate
CGContextScaleCTM(context, 10,10);
CGContextTranslateCTM(context, 10, 10);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextFillPath(context);
HW3
Abstract: Changes the scale of the user coordinate system in a context.
Parameters:
Objective-C class – An Objective-C class with a header for Cocoa Touch
Objective-C protocol – An Objective-C protocol for Cocoa
1. Now we'll repeat the process but with the transforms in reverse order.
2. We don't want the new transforms to be combined with the previous ones so we restore the original context.
3. Let's make this rectangle blue.
4. Apply the transforms in the reverse order.
5. Finish up.

Run your app and this is the result.

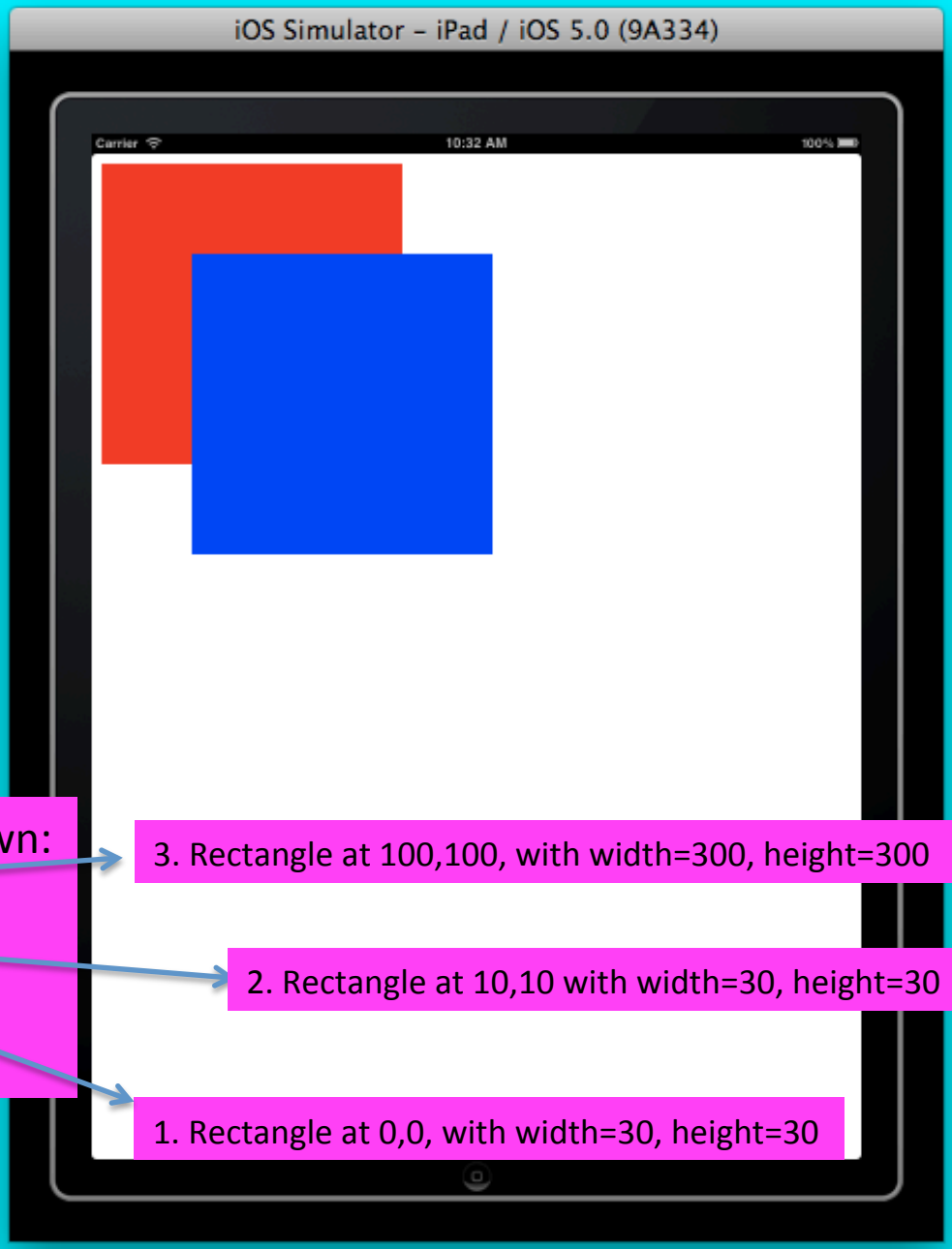


First the red triangle is drawn:
Translate by 10, 10
Scale by 10,10
Add rectangle to context

3. Rectangle at 10,10, with width=300, height=300

2. Rectangle at 0,0 with width=300, height=300

1. Rectangle at 0,0, with width=30, height=30

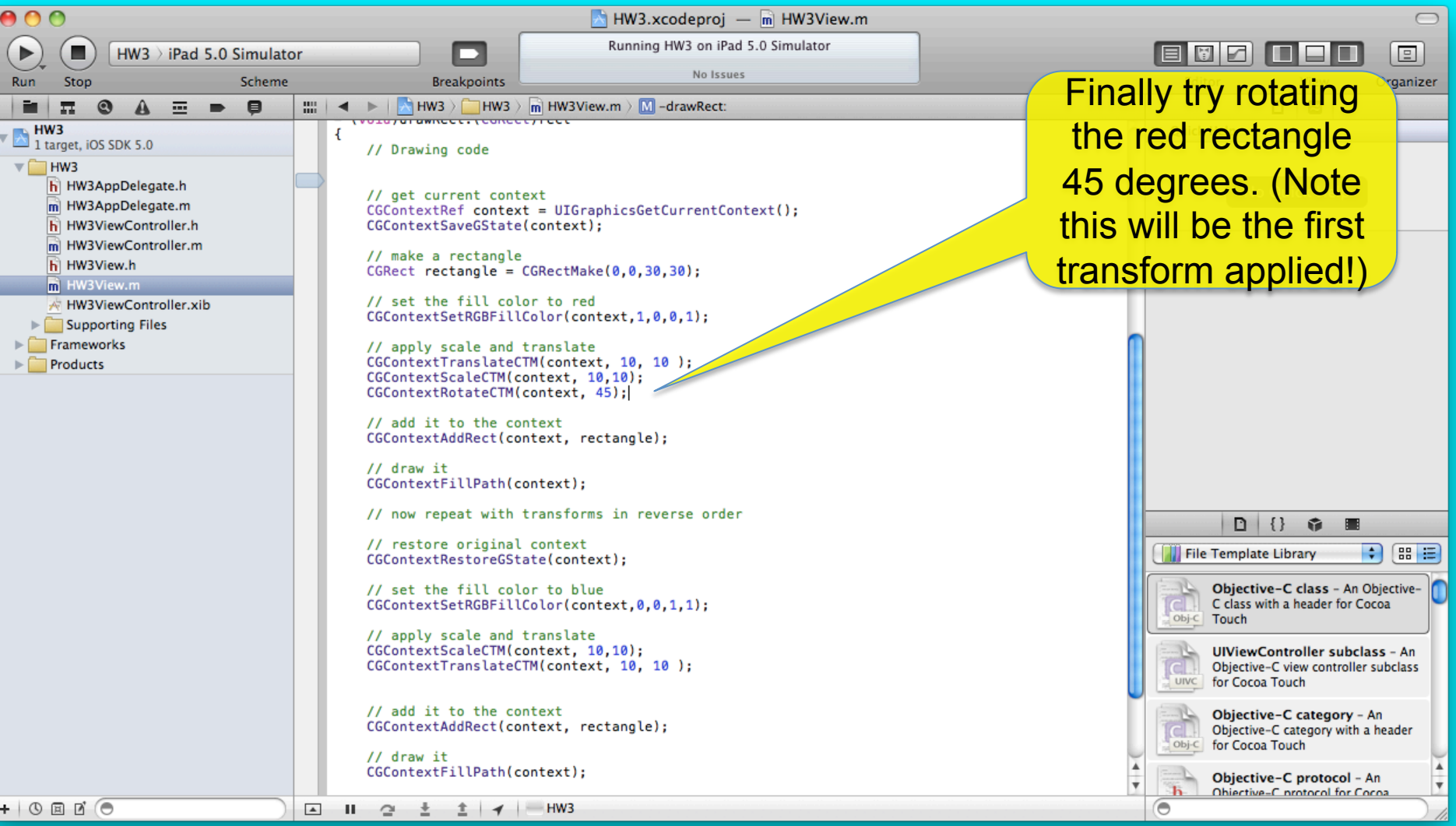


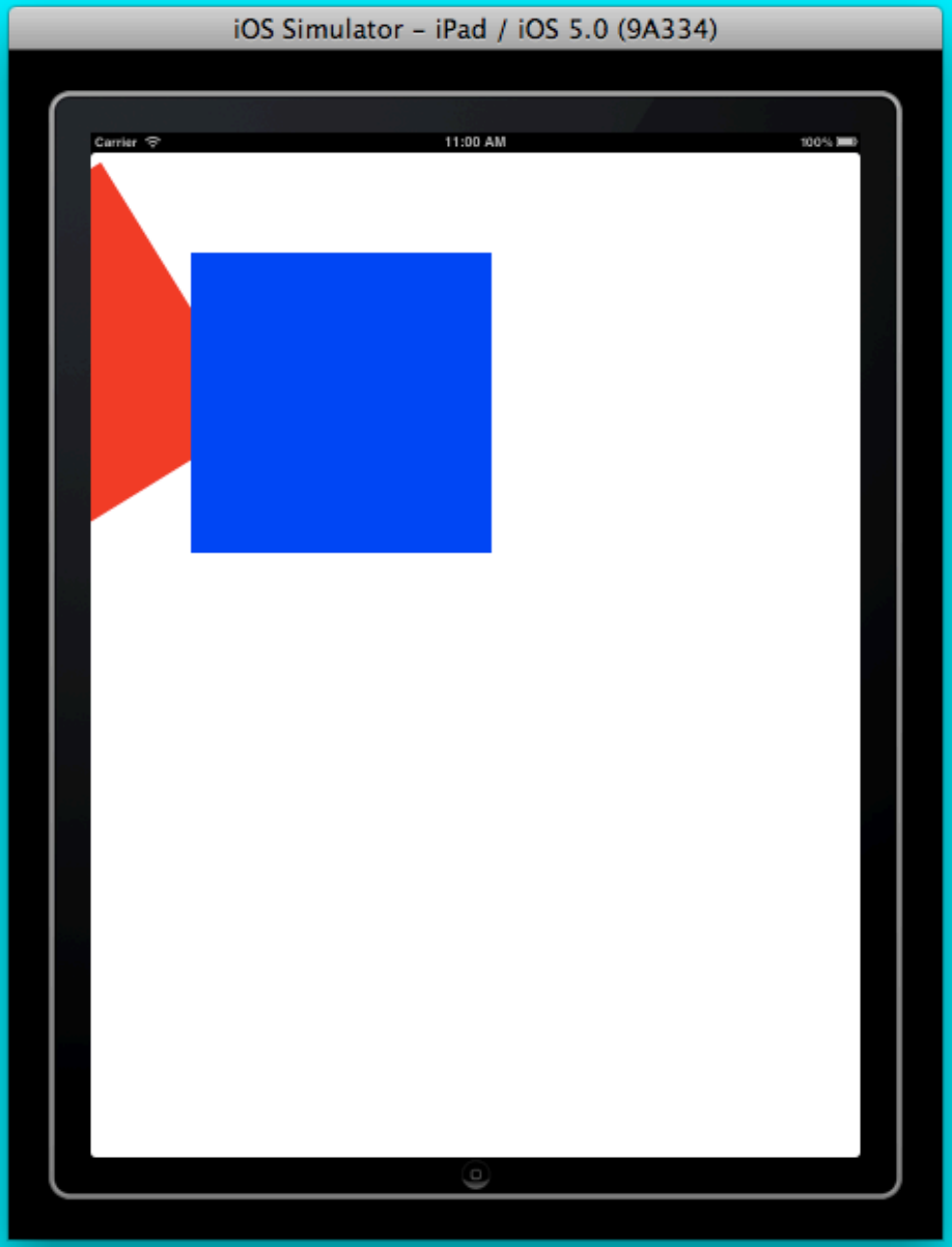
Then the blue triangle is drawn:
Scale by 10, 10
Translate by 10,10
Add rectangle to context

3. Rectangle at 100,100, with width=300, height=300

2. Rectangle at 10,10 with width=30, height=30

1. Rectangle at 0,0, with width=30, height=30


HW3.xcodeproj — HW3View.m
Run
Stop
HW3 › iPad 5.0 Simulator
Scheme
Breakpoints
Running HW3 on iPad 5.0 Simulator
No Issues
Organizer
HW3
1 target, iOS SDK 5.0
HW3
HW3AppDelegate.h
HW3AppDelegate.m
HW3ViewController.h
HW3ViewController.m
HW3View.h
HW3View.m
HW3ViewController.xib
Supporting Files
Frameworks
Products
HW3 › HW3 › HW3View.m › -drawRect:
{
// Drawing code
// get current context
CGContextRef context = UIGraphicsGetCurrentContext();
CGContextSaveGState(context);
// make a rectangle
CGRect rectangle = CGRectMake(0,0,30,30);
// set the fill color to red
CGContextSetRGBFillColor(context,1,0,0,1);
// apply scale and translate
CGContextTranslateCTM(context, 10, 10);
CGContextScaleCTM(context, 10,10);
CGContextRotateCTM(context, 45);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextFillPath(context);
// now repeat with transforms in reverse order
// restore original context
CGContextRestoreGState(context);
// set the fill color to blue
CGContextSetRGBFillColor(context,0,0,1,1);
// apply scale and translate
CGContextScaleCTM(context, 10,10);
CGContextTranslateCTM(context, 10, 10);
// add it to the context
CGContextAddRect(context, rectangle);
// draw it
CGContextFillPath(context);
Finally try rotating the red rectangle 45 degrees. (Note this will be the first transform applied!)
File Template Library
Objective-C class – An Objective-C class with a header for Cocoa Touch
UIViewController subclass – An Objective-C view controller subclass for Cocoa Touch
Objective-C category – An Objective-C category with a header for Cocoa Touch
Objective-C protocol – An
HW3

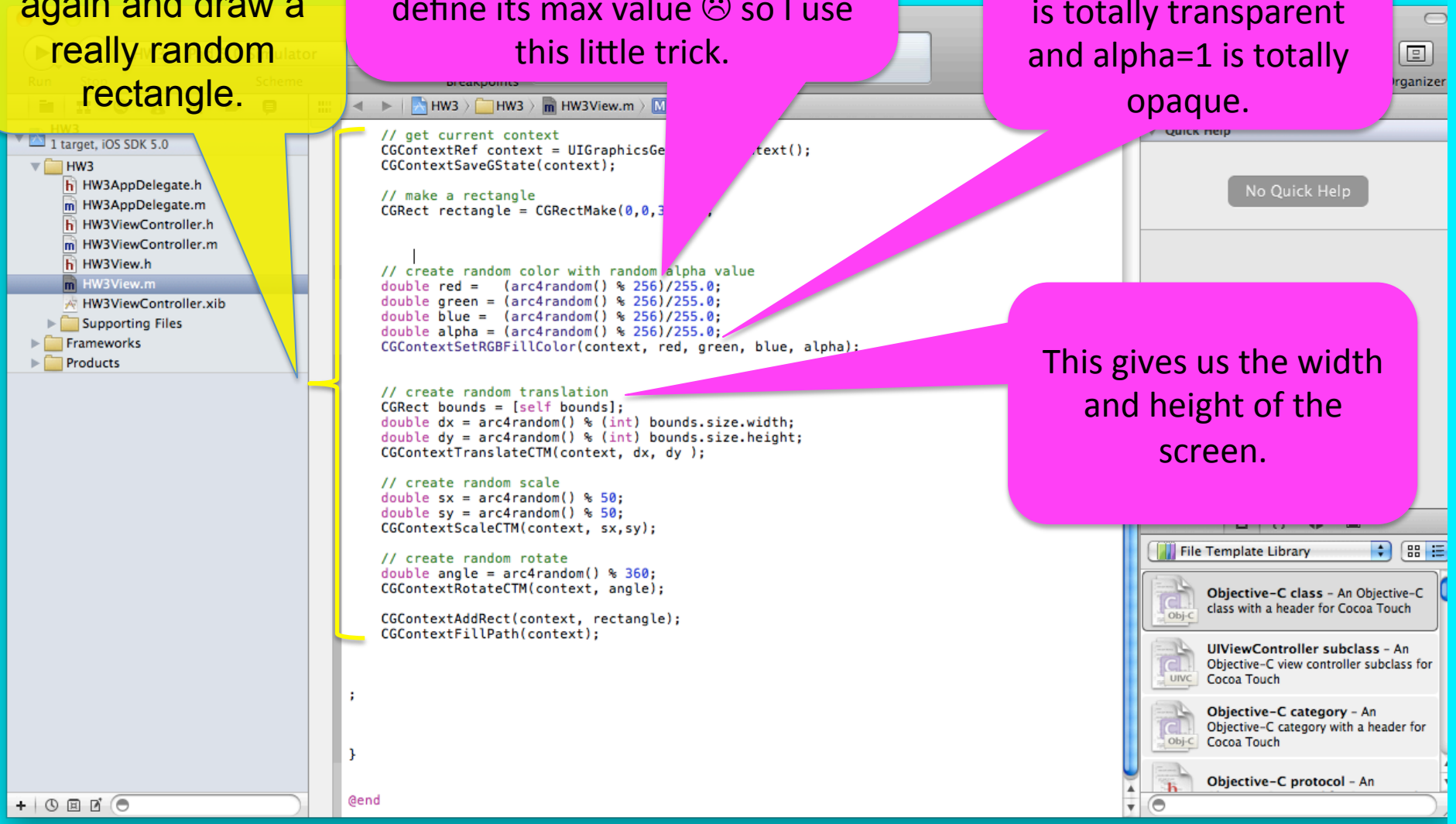

iOS Simulator – iPad / iOS 5.0 (9A334)
Carrier
11:00 AM
100%

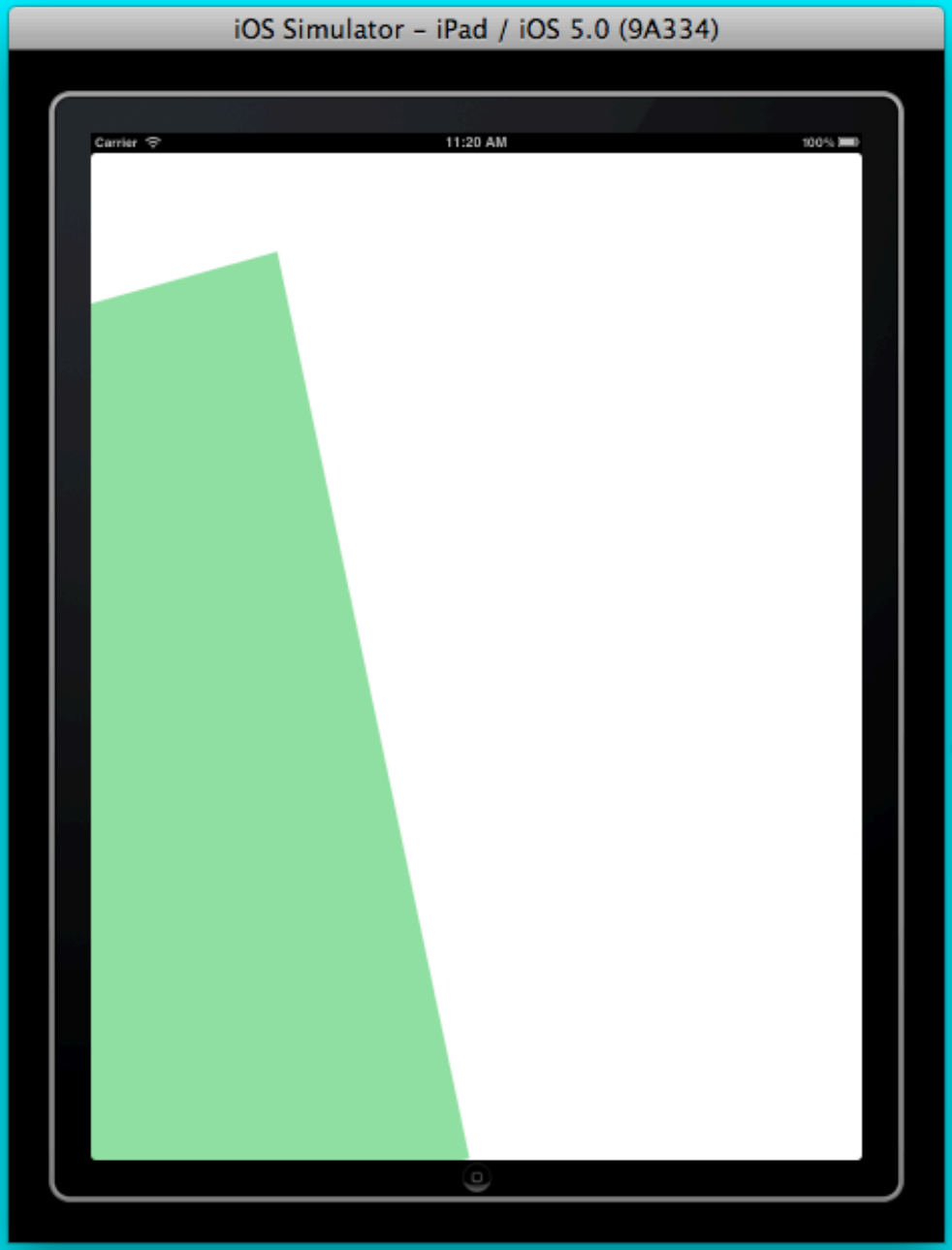
Let's start over again and draw a really random rectangle.

I want to take a random number divided by its max value to get a random number in [0,1]. Arc4random does not seem to define its max value ☹ so I use this little trick.

We are using random transparency. Alpha=0 is totally transparent and alpha=1 is totally opaque.

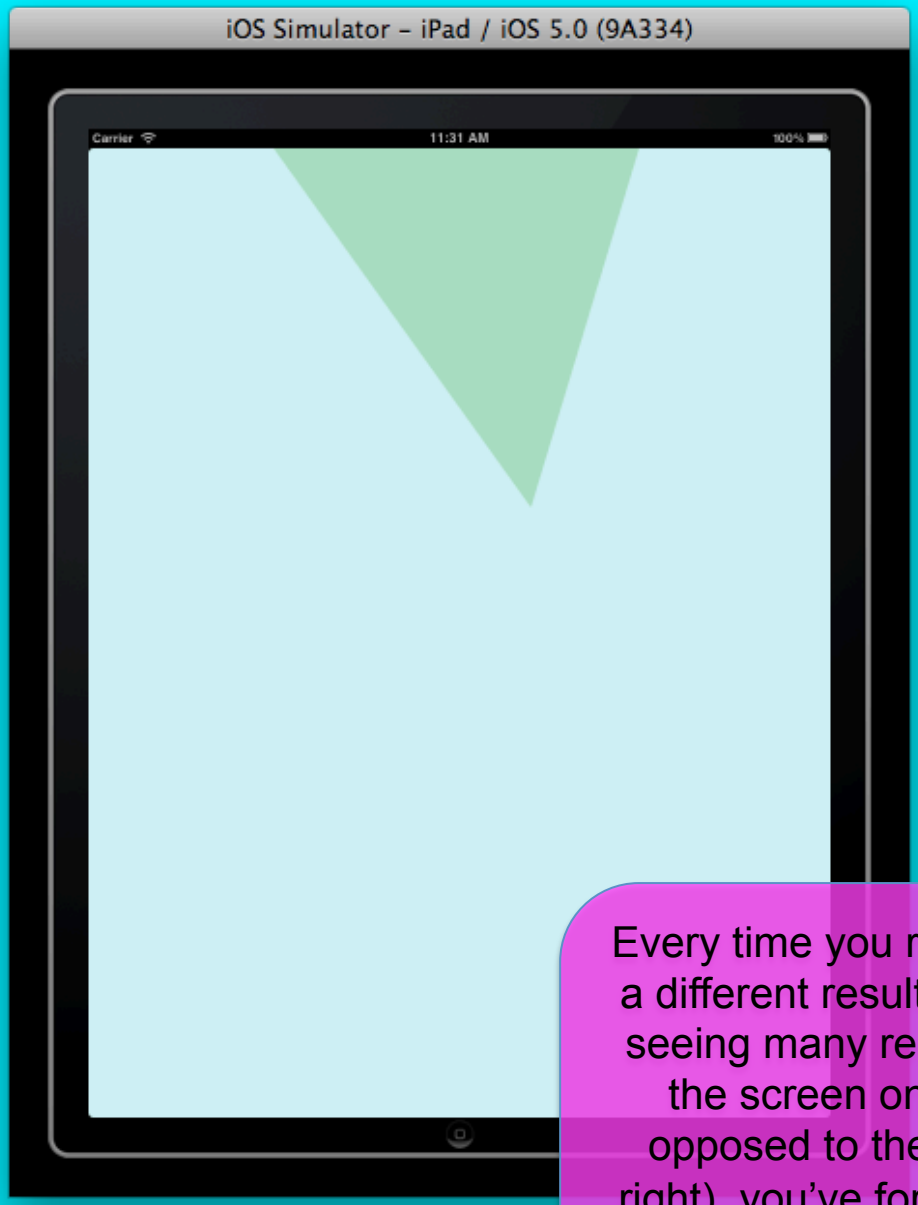
This gives us the width and height of the screen.





This is what I got. (Yawn.)
Try generating 20 random rectangles and see if it gets a little more interesting!

Go to next page for important hint.



Every time you run it you'll get a different result. If you aren't seeing many rectangles (like the screen on the left as opposed to the one on the right), you've forgotten one of the key lesson about transforms! Figure it out.