Yes, More Storyboarding: Outlets

LAB 05: Outlets

Objective

Yes, we’re going to do a little bit more foundation-building ground work with **Storyboards**.

To recap, we’re rounding out our look **Storyboards** by looking at the bridge between **Storyboards** and coding.

This bridge is comprised of:

- Actions (This was our last lab.)
- Outlets (This is what we’re looking at now.)

What’s an outlet? Outlets connect things (like buttons) on your Storyboard, to your code. These outlet connections allow you to see and change your Storyboard from code.

Let’s get started...

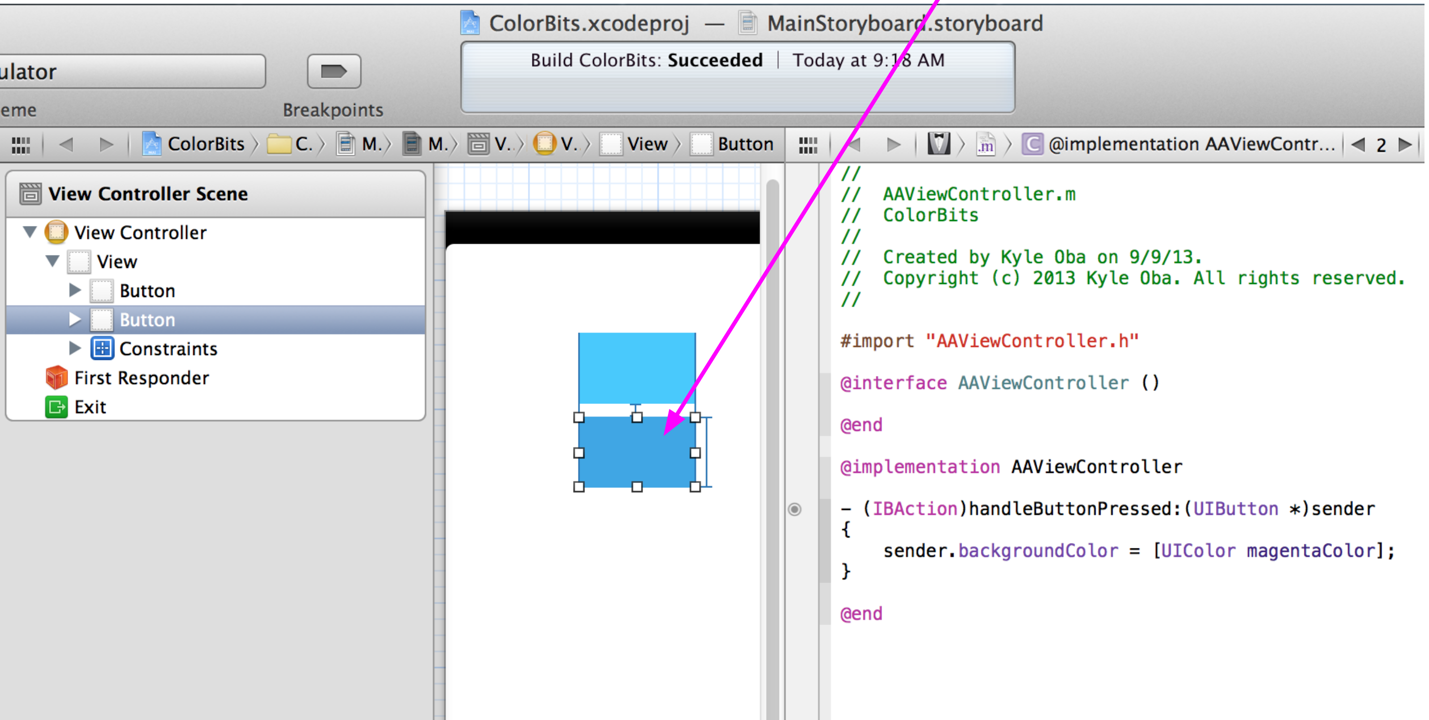
Reuse your project from LAB 4.

Seriously, use the same project. This lab builds upon what you built in LAB 4.

Add more buttons.

Did you know you can create a copy of an existing object on the **Storyboard**? You can.

Hold down the **option** key, then drag and drop from the existing button. When you release the mouse, it should clone a new button onto your **Storyboard**.



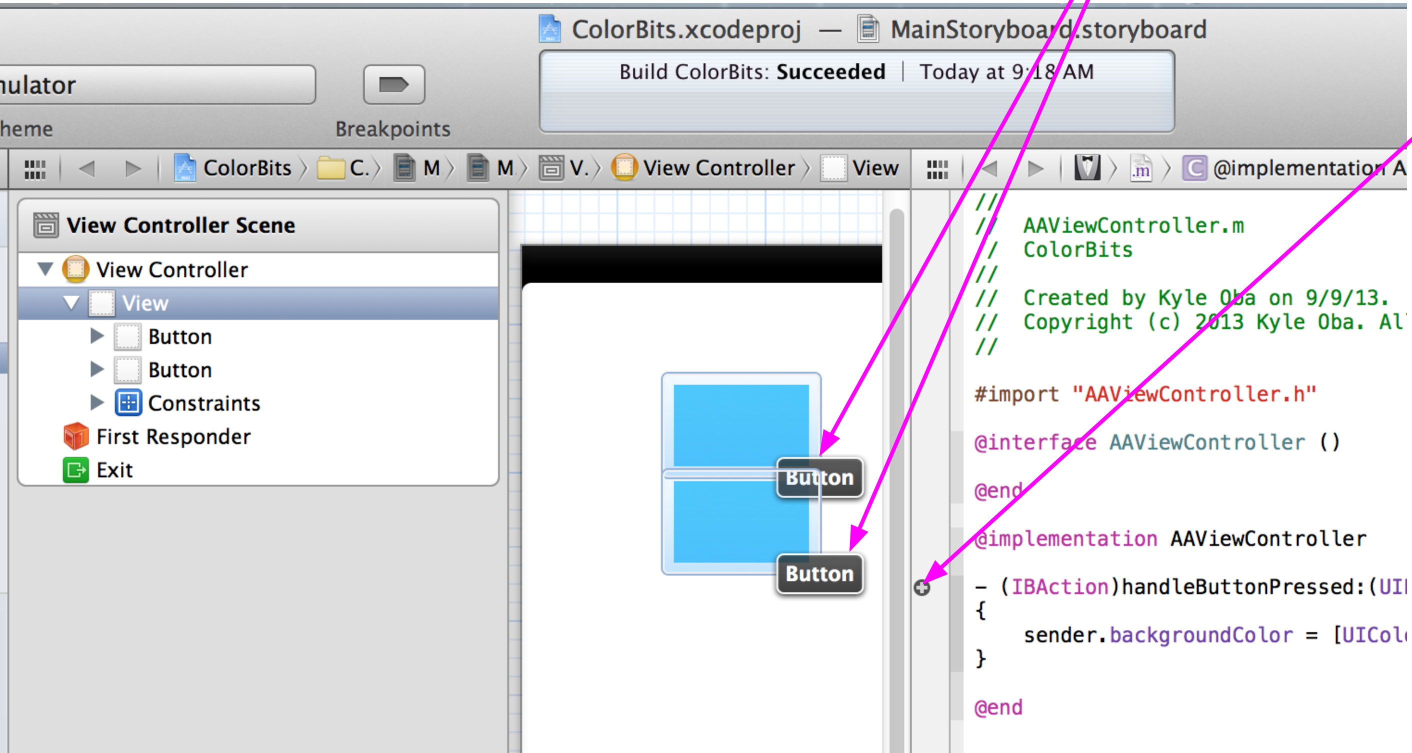
```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

This new button, is a clone.

This new button is a clone of the first button.

If you hover your mouse over the Action's little connector nub, you'll see that both buttons are connected to it.

See, both buttons light up when you hover.


ColorBits.xcodeproj — MainStoryboard.storyboard
Build ColorBits: Succeeded | Today at 9:18 AM
ulator
heme
Breakpoints
ColorBits › C. › M › M › V. › View Controller › View
@implementation A
View Controller Scene
View Controller
View
Button
Button
Constraints
First Responder
Exit
Button
Button

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. Al
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UI
{
    sender.backgroundColor = [UICol
}

@end
```

What does it mean? Well for one, it was a lot faster to create that second button.

You didn't have to customize it yourself. Xcode just creates an identically configured button.

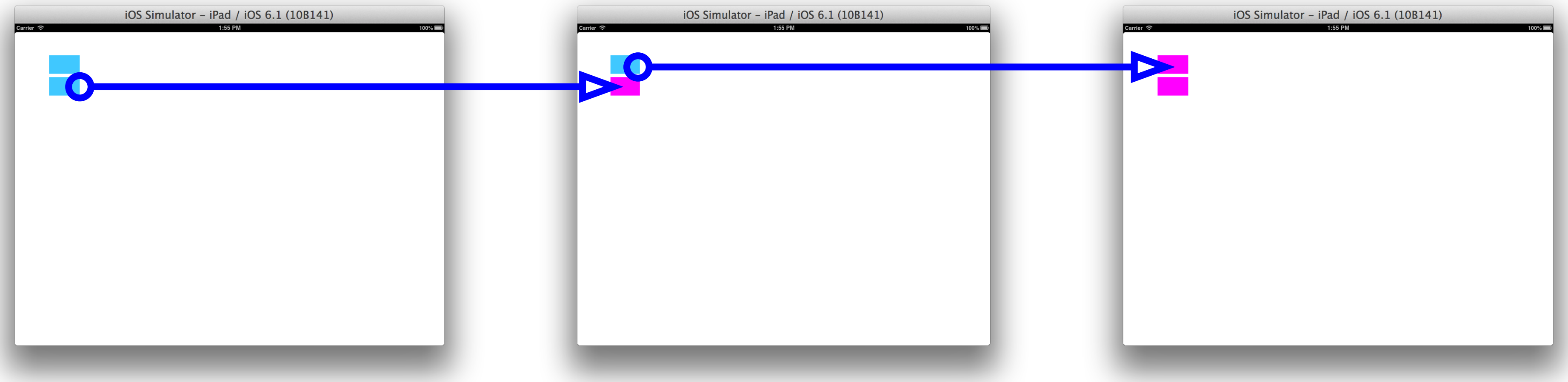
Second, you didn't have to attach the **Action** to the new button. It's already attached.

Run it.

Go ahead and run this app now.

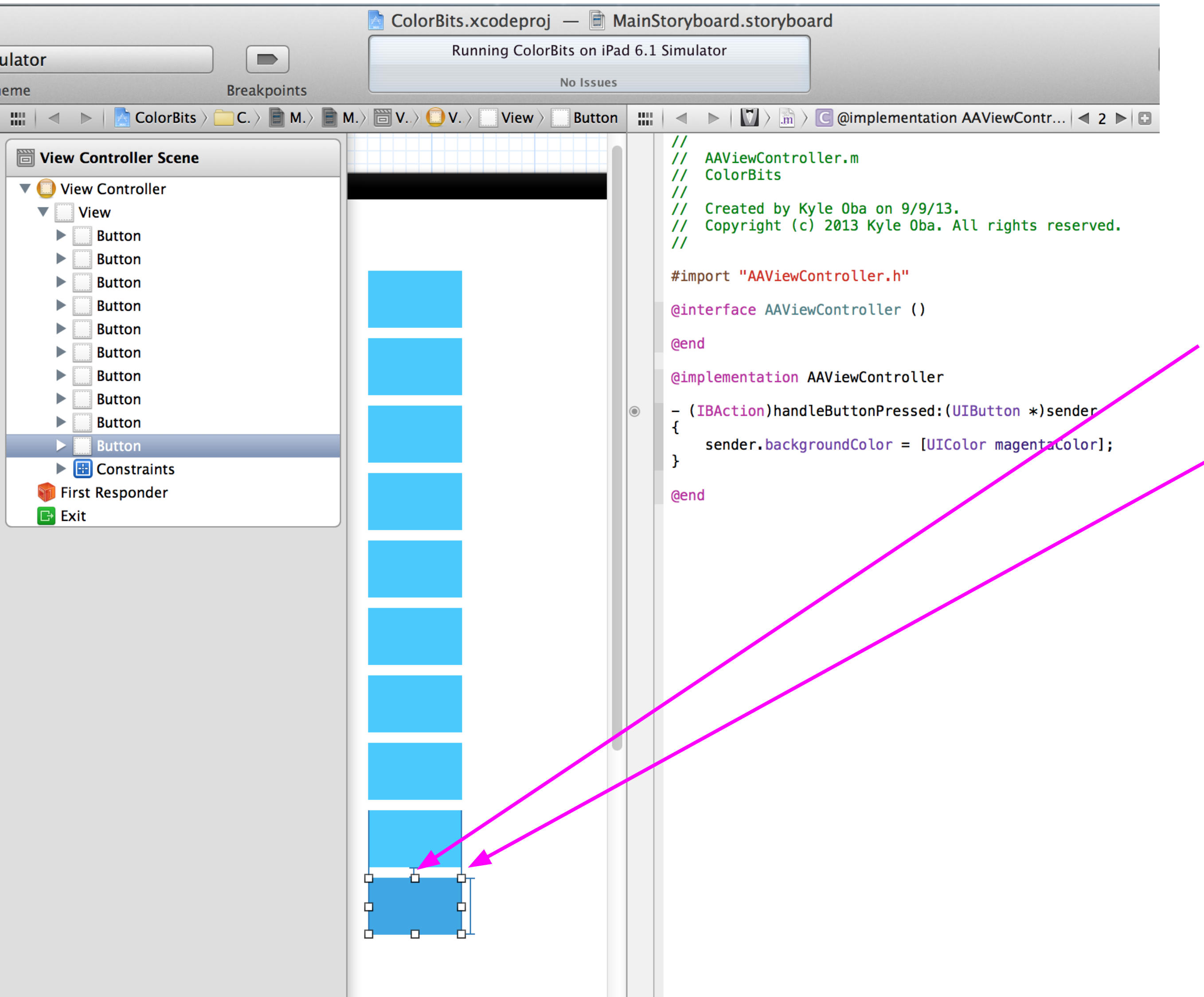
You'll see that you now have two somewhat identical buttons. If you tap on the new one. It turns magenta. If you tap on the original one, it does the same.

Good start.



iOS Simulator – iPad / iOS 6.1 (10B141)
Carrier
1:55 PM
100%
iOS Simulator – iPad / iOS 6.1 (10B141)
Carrier
1:55 PM
100%
iOS Simulator – iPad / iOS 6.1 (10B141)
Carrier
1:55 PM
100%

Add a few more buttons.


ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
Breakpoints
ColorBits › C. › M. › M. › V. › V. › View › Button
@implementation AAViewContr...
View Controller Scene
View Controller
View
Button
Button
Button
Button
Button
Button
Button
Button
Button
Button
Constraints
First Responder
Exit

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Let's create some more buttons.

Use the same technique of holding down the **option** key, and dragging an existing button to create 8 more.

Create a column of 10 buttons like this.

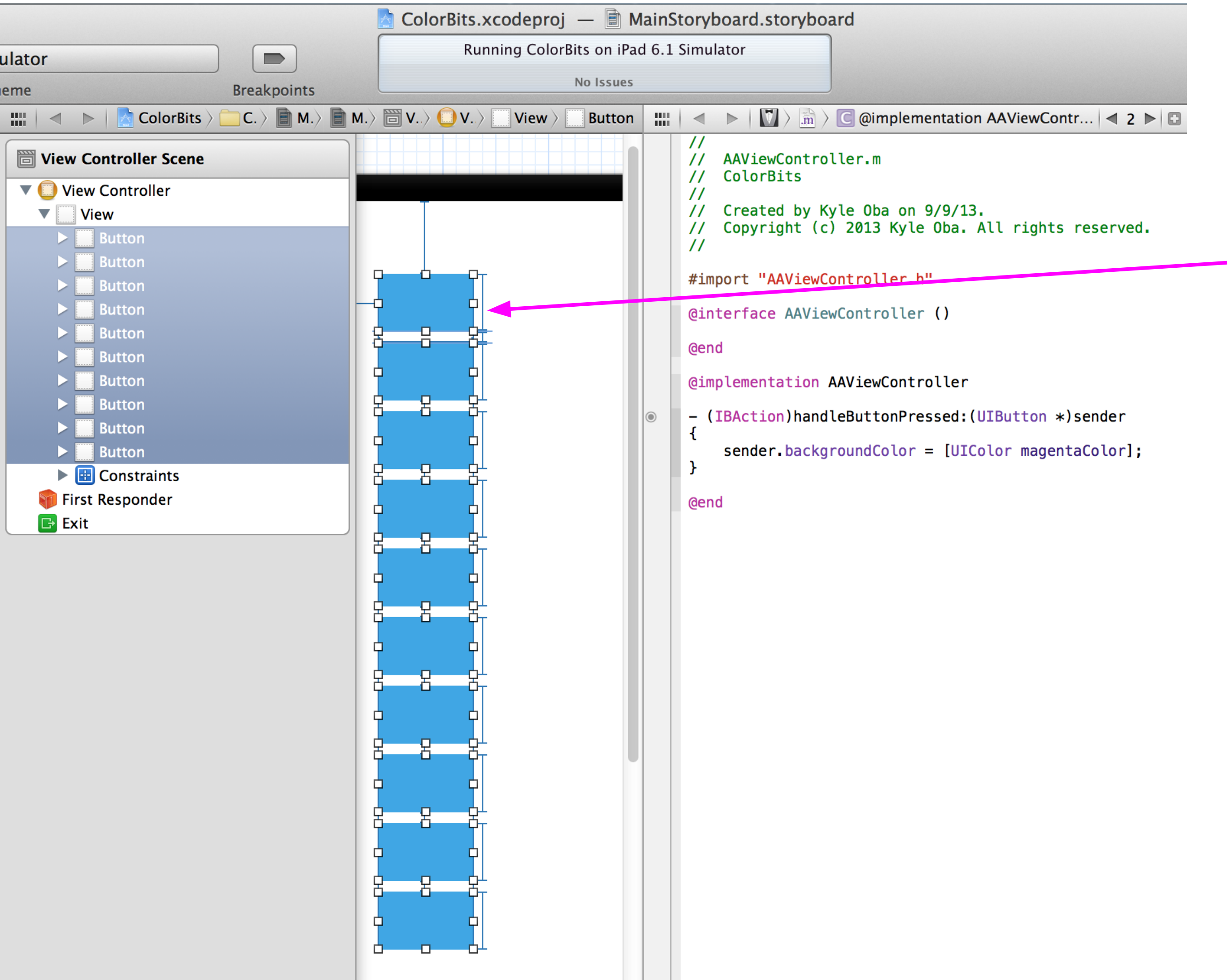
Notice how you can Xcode provides guides so you can have even spacing between you buttons.

It also provides guides which align the left and right sides of your buttons.

These are there so to help you position things in a somewhat sane way. You'll notice that they're somewhat "magnetic." Meaning as you move a button around, it *snaps* into position around these guides.

If the guides annoy you for some reason, you can turn them on and off.

Select all the buttons.


ulator
eme
Breakpoints
ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
ColorBits › C. › M. › M. › V. › V. › View › Button
@implementation AAViewContr… 2

View Controller Scene
- View Controller
  - View
    - Button
    - Button
    - Button
    - Button
    - Button
    - Button
    - Button
    - Button
    - Button
    - Button
    - Constraints
- First Responder
- Exit

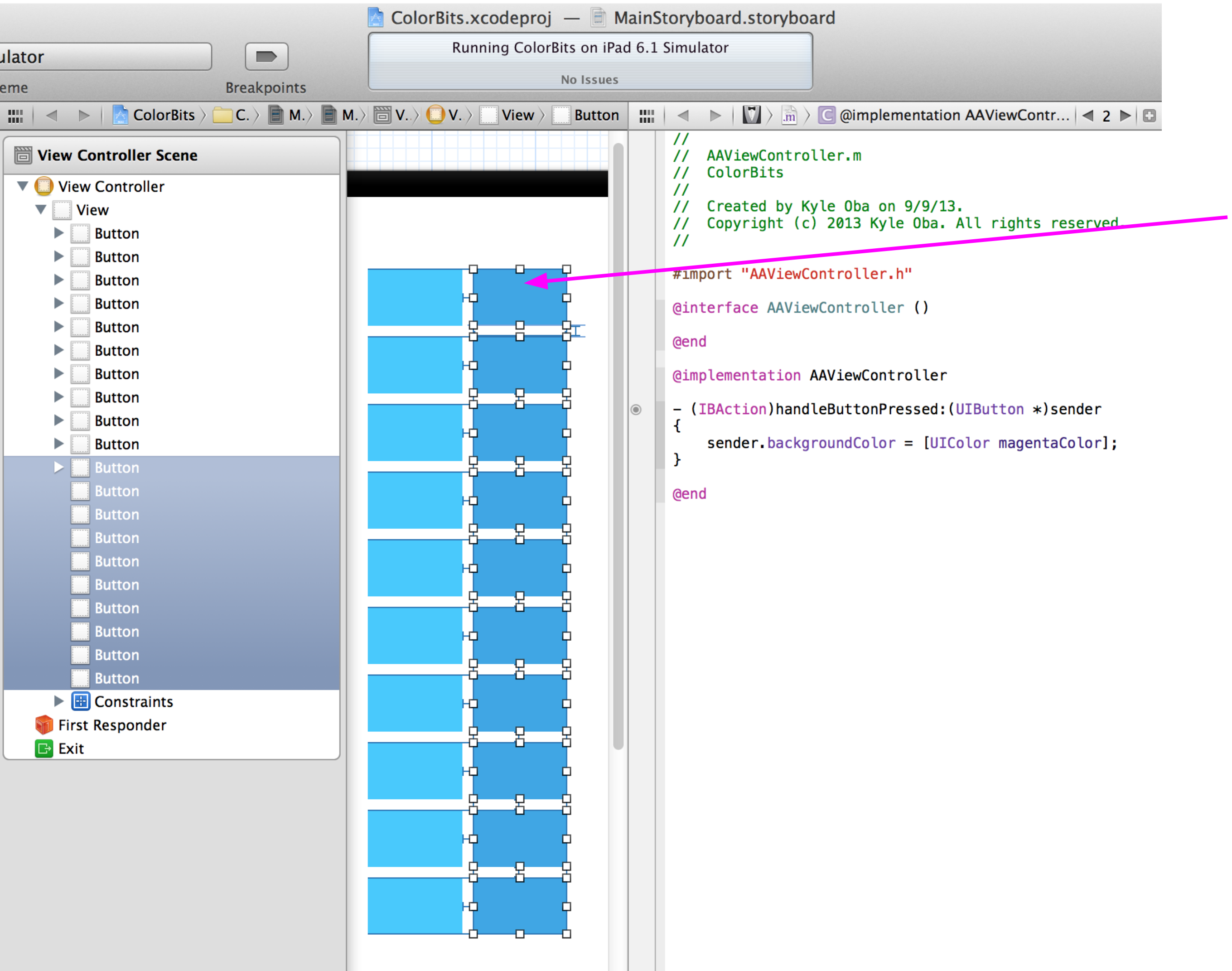
```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

You already know that you can select something on your Storyboard by clicking on it.

To select multiple things, hold down the **shift** key and click on all the buttons.

Each one should have the *selected* look as you see here.

Clone them all


ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
Breakpoints
ColorBits › C. › M. › M. › V. › V. › View › Button
@implementation AAViewController
View Controller Scene
View Controller
View
Button
Constraints
First Responder
Exit

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Now we're going to clone that entire row of buttons.

Holding down the **option** key, click and drag from any of the selected buttons over to the right.

When you release the mouse, you'll plop a whole new column of buttons down. Each is a clone of the original.

Run the app again. You should be able to tap each button and change its color.

Objective: Updated

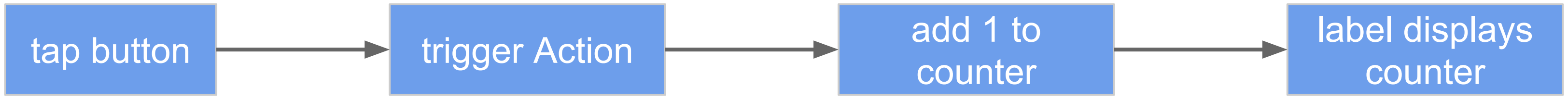
So, with buttons, Actions and Outlets, we should be able to record and display a counter every time someone taps a button.

Let's just say we're building a game.

We'll add 1 point to the score whenever a user taps on a button (turning it magenta).

This means we have to handle the button tap Action, record the score, then display the score.

The process looks something like this…



Lots of games have a score that's displayed at the top. Let's start by adding this label to the top of our app.

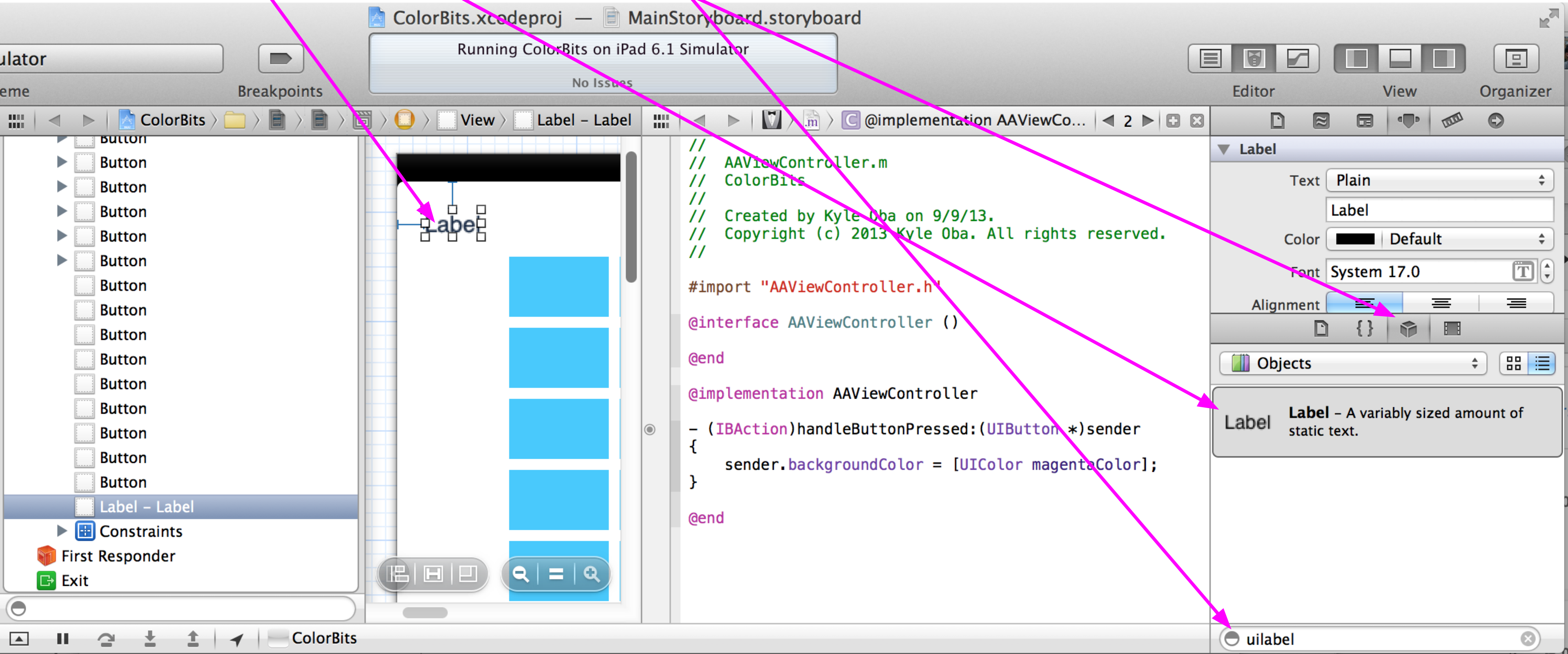
Add a label

You know the drill by now. But, let's run through the steps to add a new label.

With the **Object library** selected, search for the "uilabel" object.

Drag the **Label** object into your **Storyboard**.

Drop in here.



```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Label

Text: Plain

Label

Color: Default

Font: System 17.0

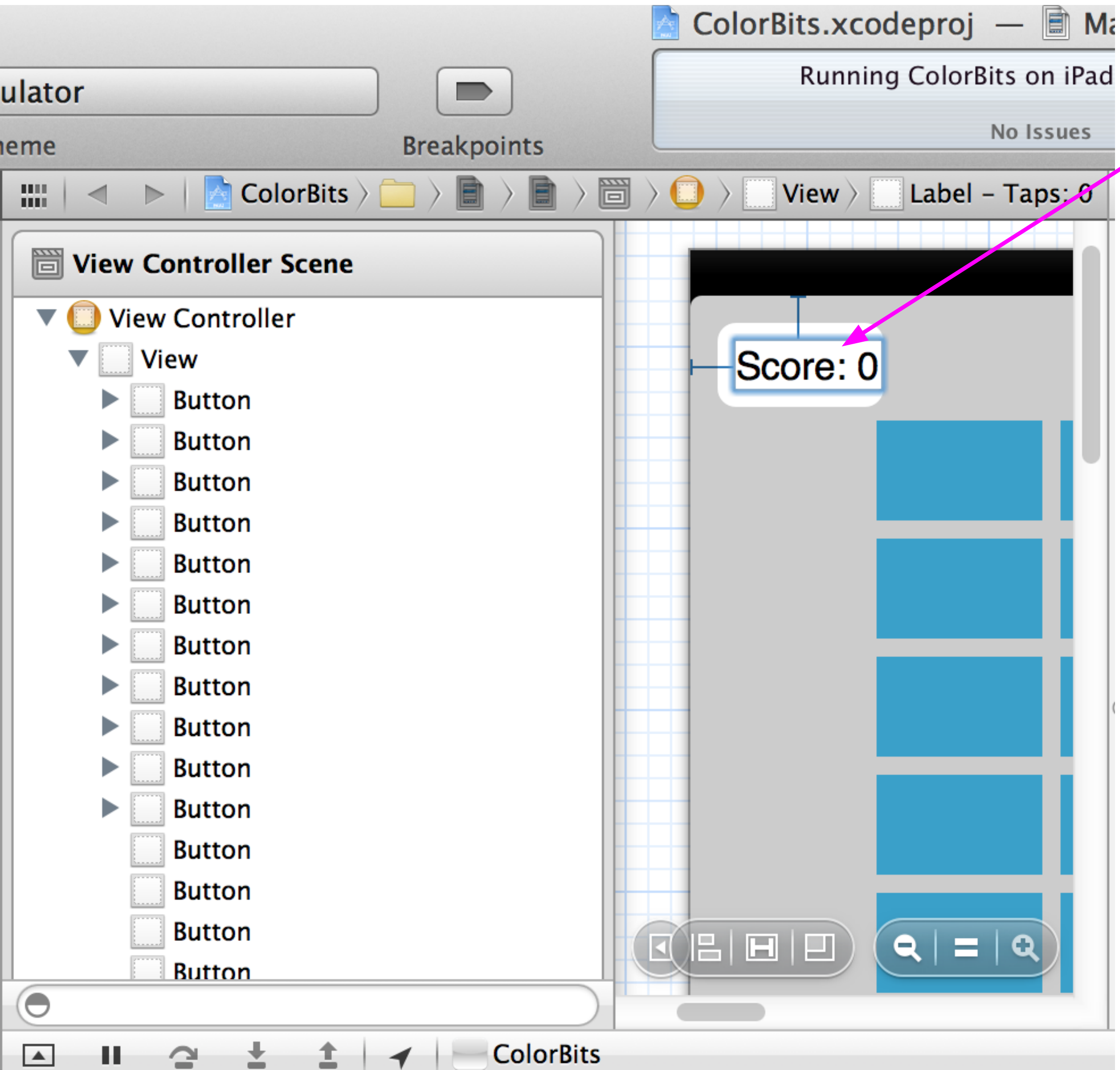
Alignment

Objects

Label – A variably sized amount of static text.

uilabel

Set up the label.



We want our label to say something more interesting than “Label.”

Double click the label. This will allow you to change the text in the label.

Make it say:

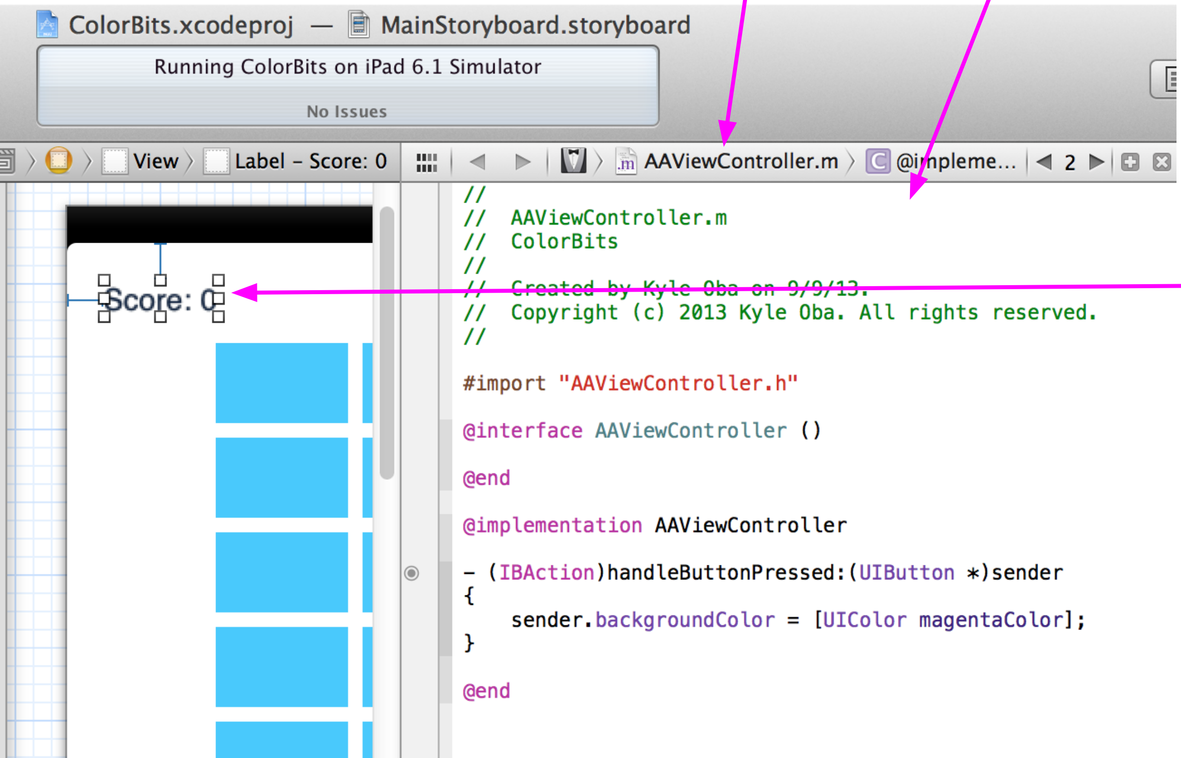
Score: 0

Click outside the label to save your changes to the label text.

About that Outlet.

We're going to connect the label to the implementation file using an **Outlet**.

Verify that the **AAViewController.m** file is visible in the **Assistant editor**.


ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
View
Label – Score: 0
AAViewController.m
@impleme...
2
Score: 0

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Arrange the **Storyboard** (on the left-hand side) so that you can see your label.

Create the Outlet.

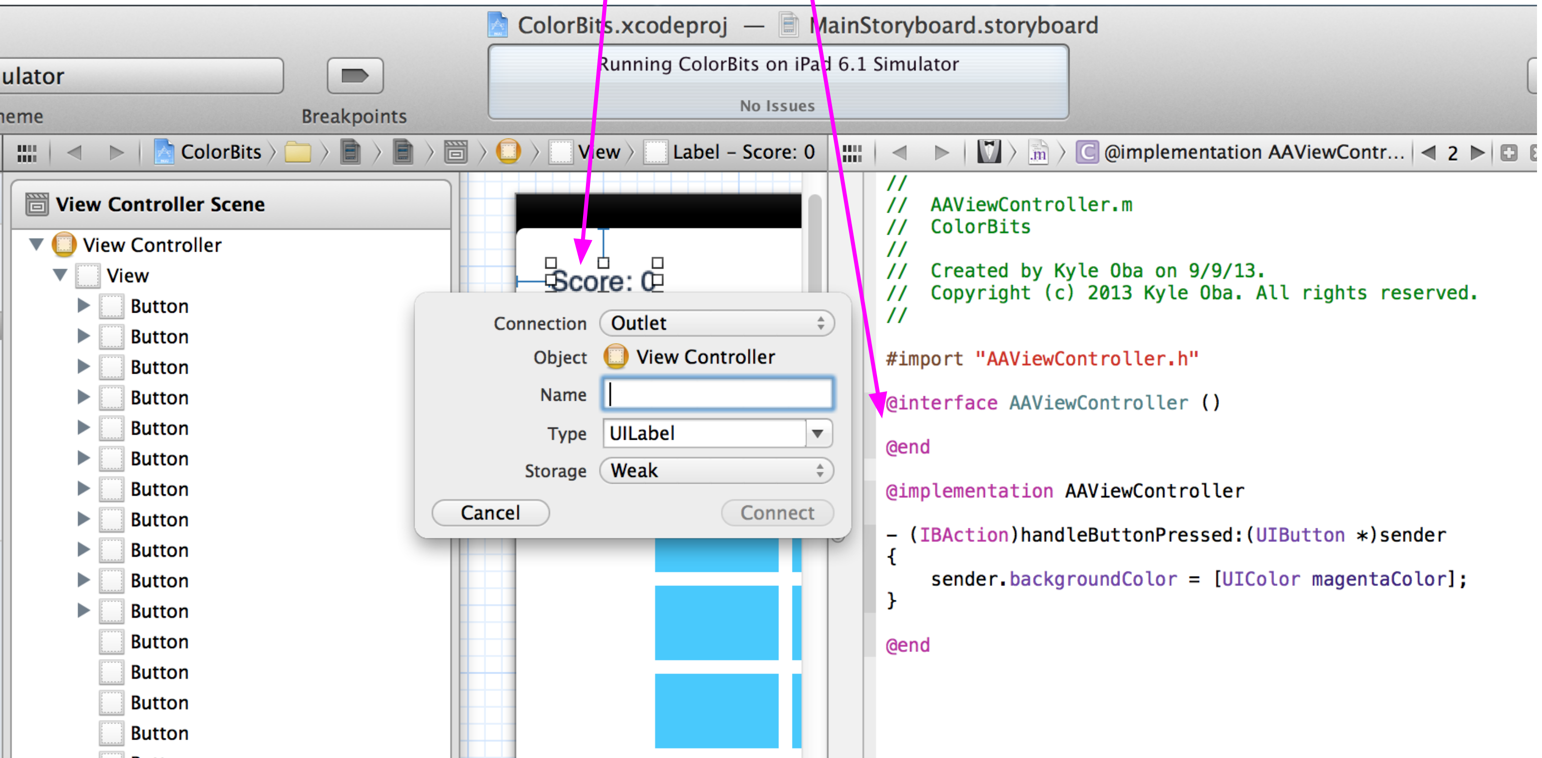
Creating an **Outlet** is very similar to how you created your **Action**.

To do this, select the label. Then, while holding down the **control** key,

click and drag into the area between the “@interface AAViewController ()” line and “@end” line.

Xcode will pop up this context menu, which allows you to configure your **Outlet**.

We’ll do that on the next slide.



Connection: Outlet
Object: View Controller
Name:
Type: UILabel
Storage: Weak

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Configure the Outlet.

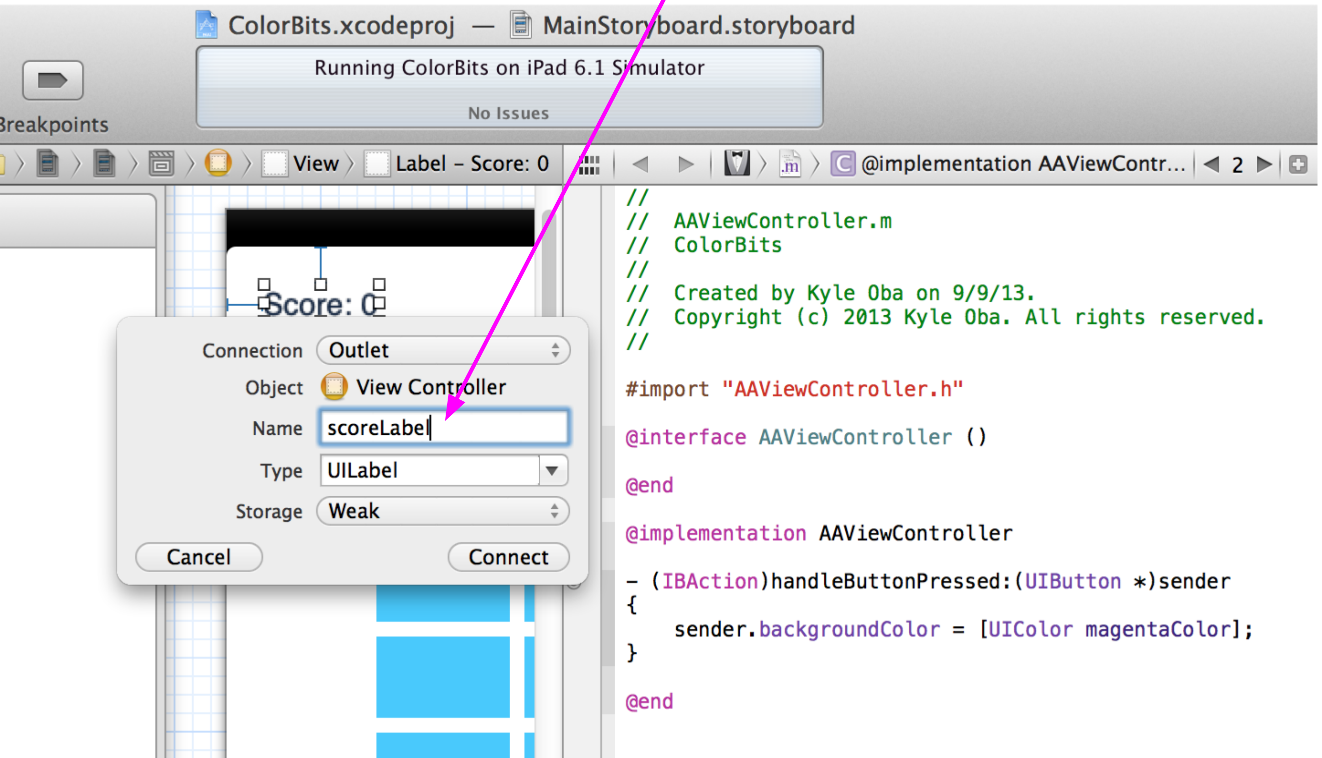
The only thing you have to add is the **Name** field. Good names from labels end in “Label.”

I used the name:

scoreLabel

I suggest you do the same.

Press the **Connect** button.



ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
Breakpoints
View
Label – Score: 0
@implementation AAViewContr...
2
Score: 0
Cancel
Connect

Connection: Outlet
Object: View Controller
Name: scoreLabel
Type: UILabel
Storage: Weak

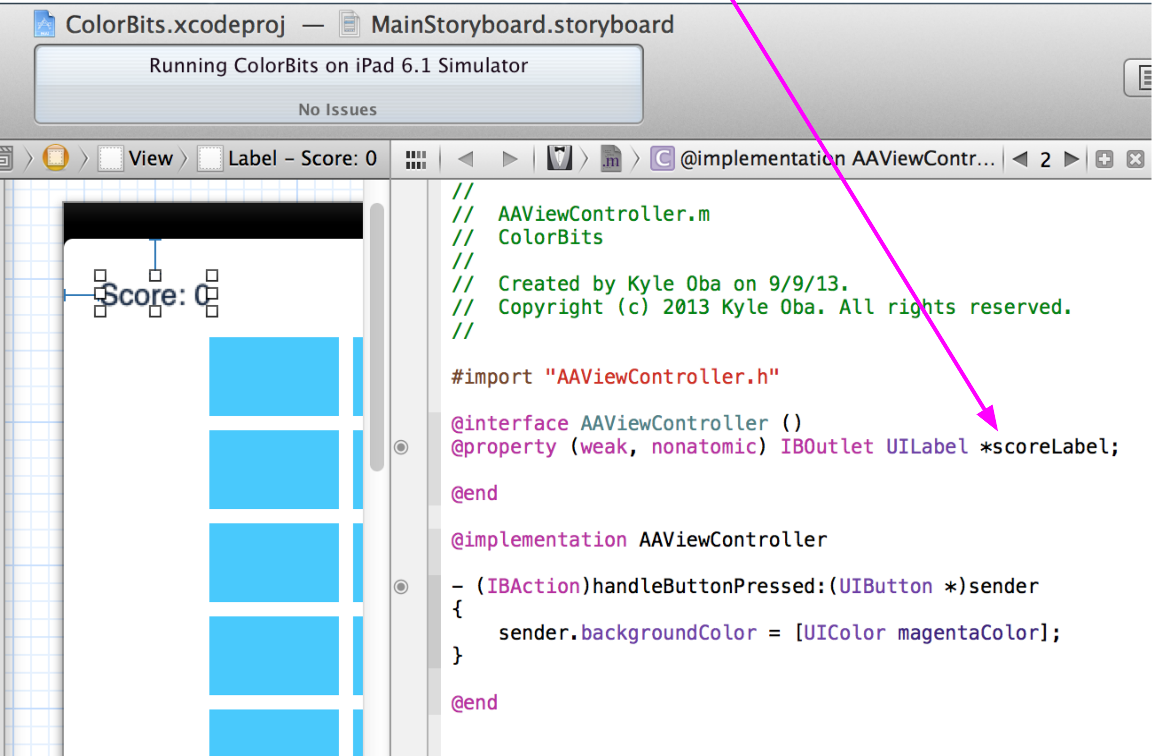
```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Thank you for the property.

Xcode created a **property** for you.

See how it has the same name as what you specified in the configuration window?

This is what we'll use in the code to refer to the label in the **Storyboard**.


ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
View
Label – Score: 0
@implementation AAViewContr…
2
Score: 0

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()
@property (weak, nonatomic) IBOutlet UILabel *scoreLabel;

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

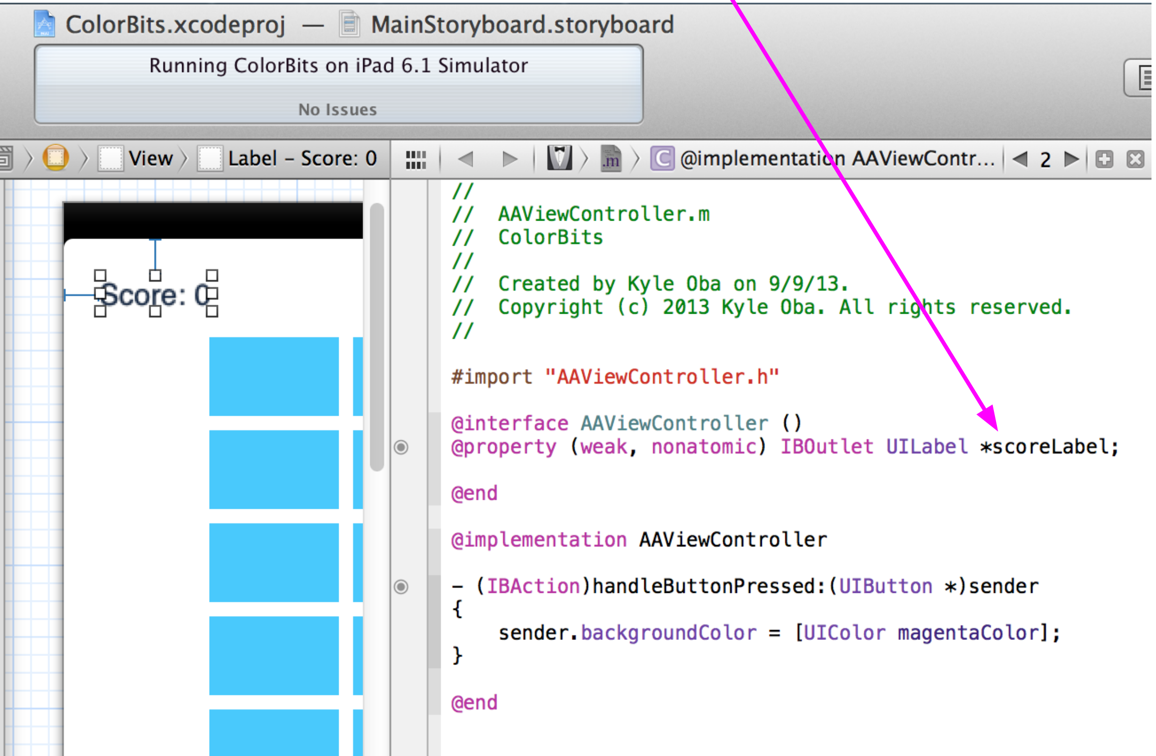
What are we missing?

Xcode created a **property** for you.

See how it has the same name as what you specified in the configuration window?

This is what we'll use in the code to refer to the label in the **Storyboard**. This property stores a connection to your label.

But, what's missing? Well, we need *another* property to store the score. The score is a number, we'll add a property which can store a number next.


ColorBits.xcodeproj — MainStoryboard.storyboard
Running ColorBits on iPad 6.1 Simulator
No Issues
View
Label – Score: 0
@implementation AAViewContr...
2
Score: 0

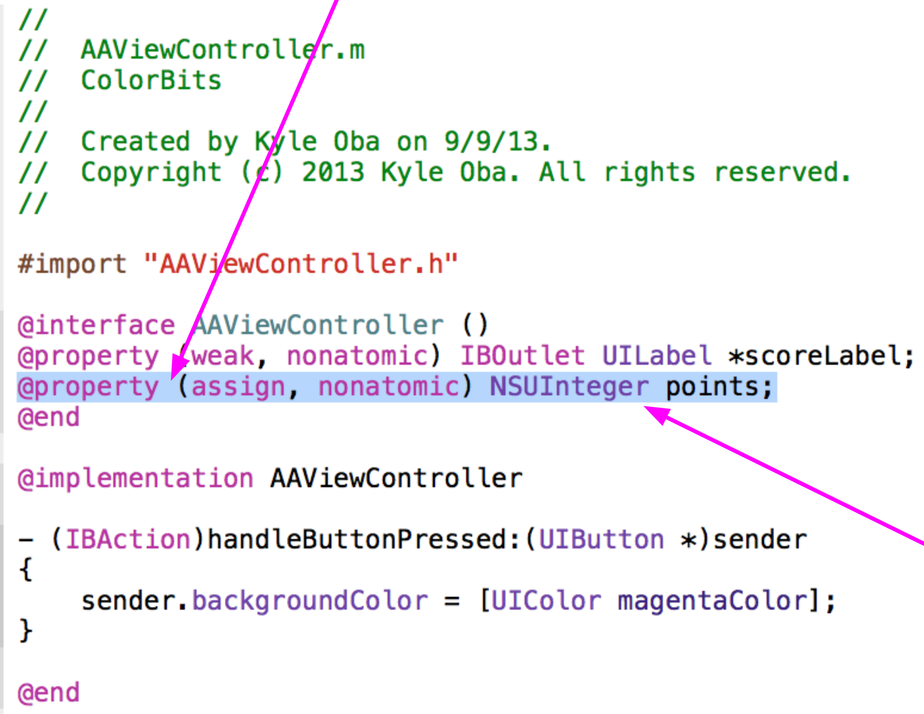
```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()
@property (weak, nonatomic) IBOutlet UILabel *scoreLabel;

@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

Add a new property for scores.

Okay. You need to add a property to store the points that the user gets each time they change a button from blue to magenta.

Think of this property as the *memory* of your app. This is how your app *remembers* how many points the user has scored.

Add this line of code to declare a new property.



```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()
@property (weak, nonatomic) IBOutlet UILabel *scoreLabel;
@property (assign, nonatomic) NSUInteger points;
@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

There's a lot going on here. Don't worry if you don't understand every bit of it. We'll get more into the code-y stuff next week.

Make sure you understand that this is where we store the number of points the user has earned.

For now, I just want to point out a few more interesting things.

The type of thing we're storing is declared here as NSUInteger. This is a technical way of saying that our **points** property contains integer values. Furthermore, they are unsigned (that means they are always 0 or positive).

WAT?

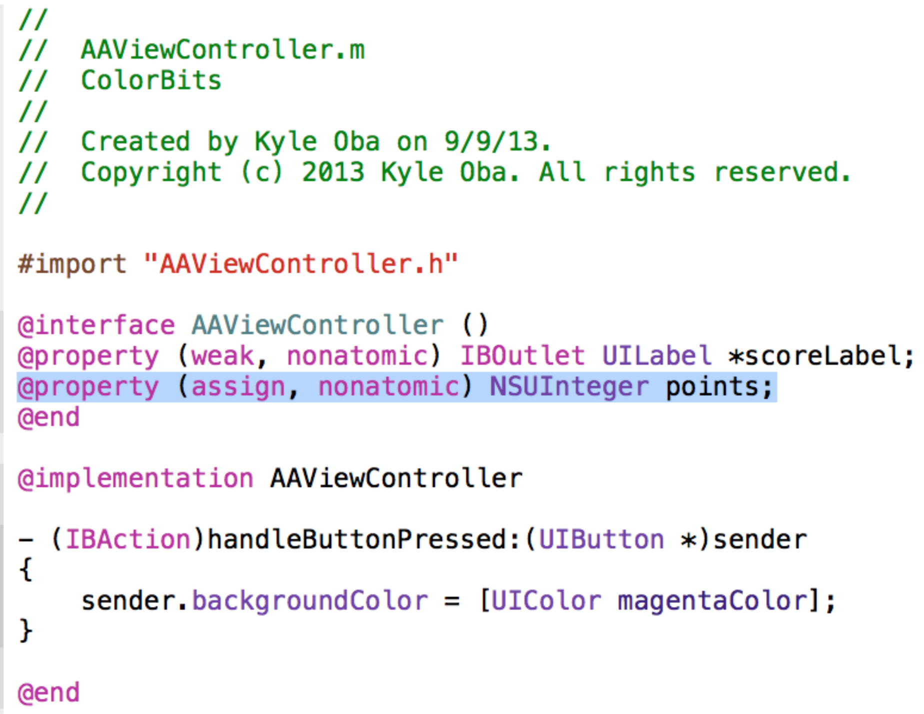
Okay. That last part bears a little explaining.

NSUInteger, what is that? Taking it backwards:

Integer == whole numbers

U == Unsigned (no negatives allowed… that would be a minus *sign)*

NS == a naming convention used to specify where this type of thing comes from.



```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()
@property (weak, nonatomic) IBOutlet UILabel *scoreLabel;
@property (assign, nonatomic) NSUInteger points;
@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];
}

@end
```

NS is the initials of the NextStep technology.

Steve Jobs purchased NextStep when he created his Next computer company (after getting kicked out of Apple).

Much of the iOS framework is built on NextStep technology.

So, you'll see a lot of "NS" stuff laying about. This won't be the last time you'll see those initials.

Please ignore the **assign**, **weak**, and **nonatomic** declarations for now. If I tried to explain them now, my head might explode. It's really just not important today. Maybe next week.

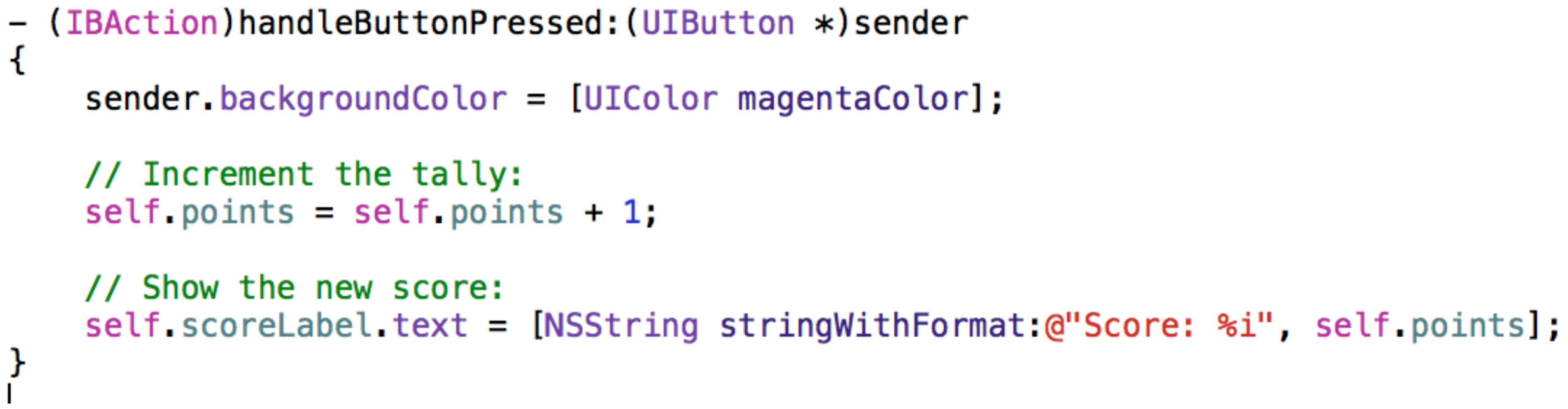
Let's cross that bridge when we get to it.

Type all this in.

This is called “cargo culting.” I want you all to type this code in. But, don’t worry about where it came from.

It’s not cheating, it’s programming.

This is your new Action method. Add the new part there (the last four lines).



```
- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];

    // Increment the tally:
    self.points = self.points + 1;

    // Show the new score:
    self.scoreLabel.text = [NSString stringWithFormat:@"Score: %i", self.points];
}
```

Increment the points tally

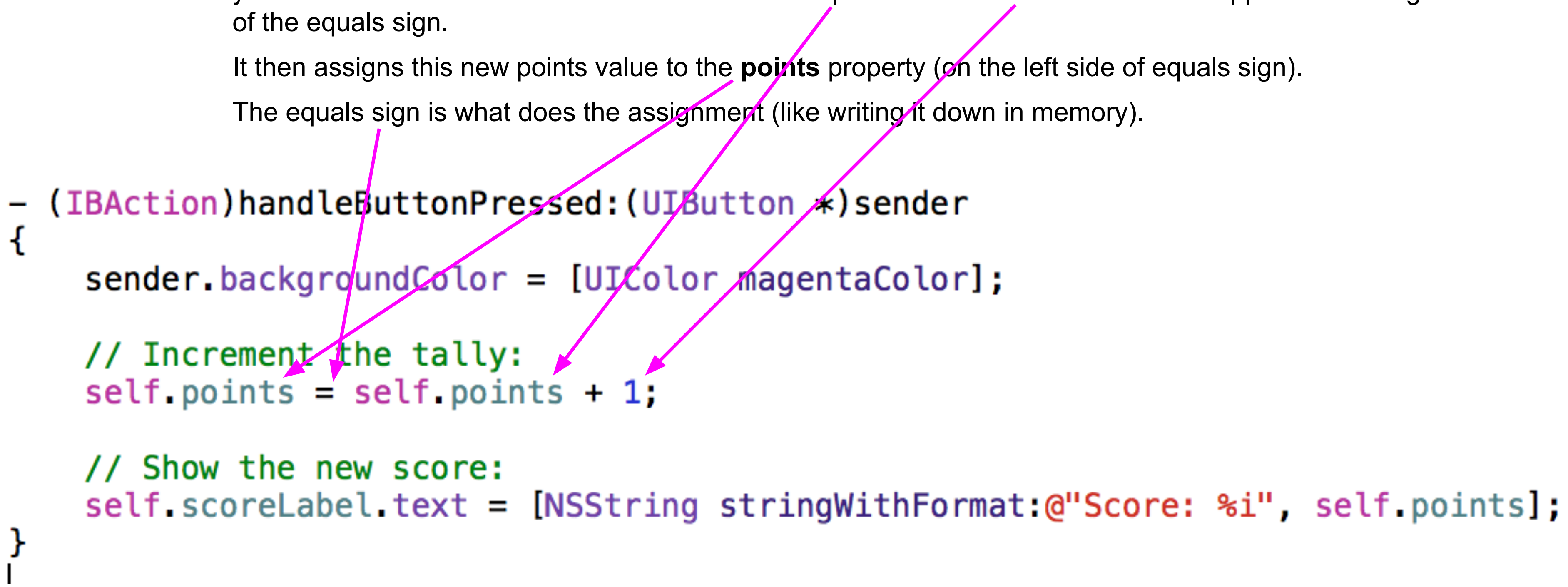
Let’s tackle that first new line first.

The one that adds 1 to the **points** property.

Do a lot of squinting, and the requisite amount of suspension of disbelief, and you should be able to convince yourself that this code takes the current number of points and adds 1. That’s what happens on the right side of the equals sign.

It then assigns this new points value to the **points** property (on the left side of equals sign).

The equals sign is what does the assignment (like writing it down in memory).



```
- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];

    // Increment the tally:
    self.points = self.points + 1;

    // Show the new score:
    self.scoreLabel.text = [NSString stringWithFormat:@"Score: %i", self.points];
}
```

Display the new score.

Now let's look at that last line.

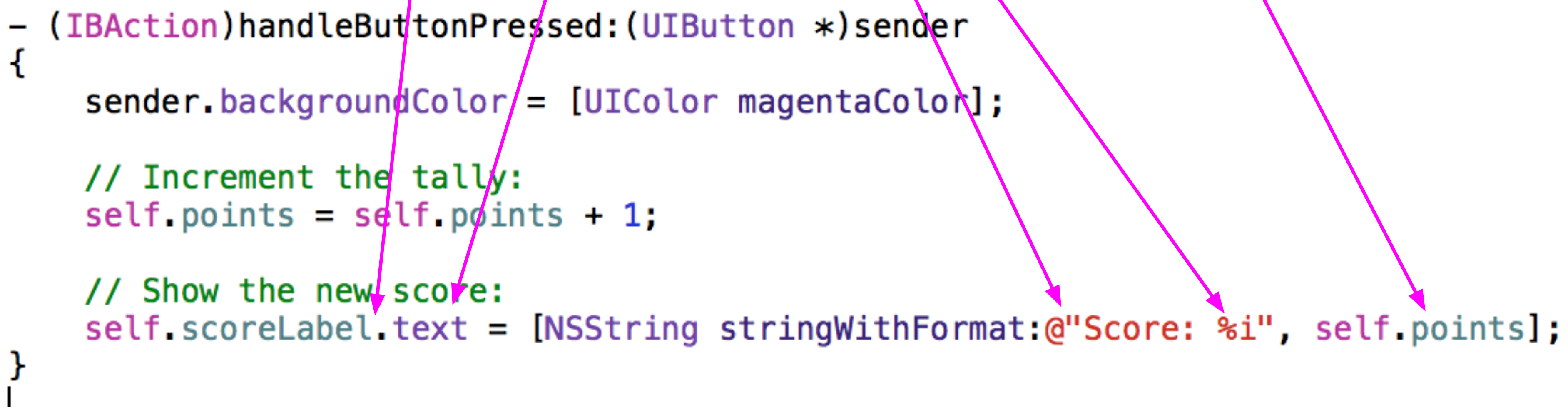
This is the one that updates the label with your new score.

This is the text that we display. It is a formatted string. Again, don't worry if this is crazy talk to you.

There is an integer substitution going on here, where **%i** is replaced with the **points** property's value.

It's kind of like a madlib, or a template of some kind.

The **scoreLabel** property's text property is then updated with this string.



```
- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];

    // Increment the tally:
    self.points = self.points + 1;

    // Show the new score:
    self.scoreLabel.text = [NSString stringWithFormat:@"Score: %i", self.points];
}
```

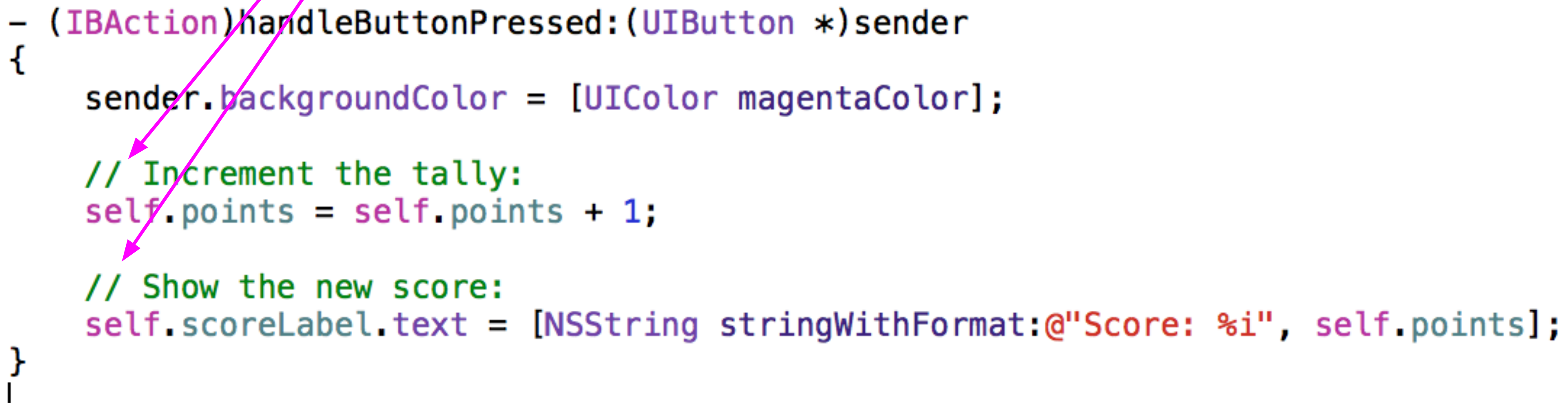
Wait, what were those // green lines?

Those lines starting in // are called “comments.”

The application just ignores those lines. They do not affect your running application at all.

They’re just notes that programmers leave for other programmers. They’re there to help you out, so that you can understand what the code is doing.

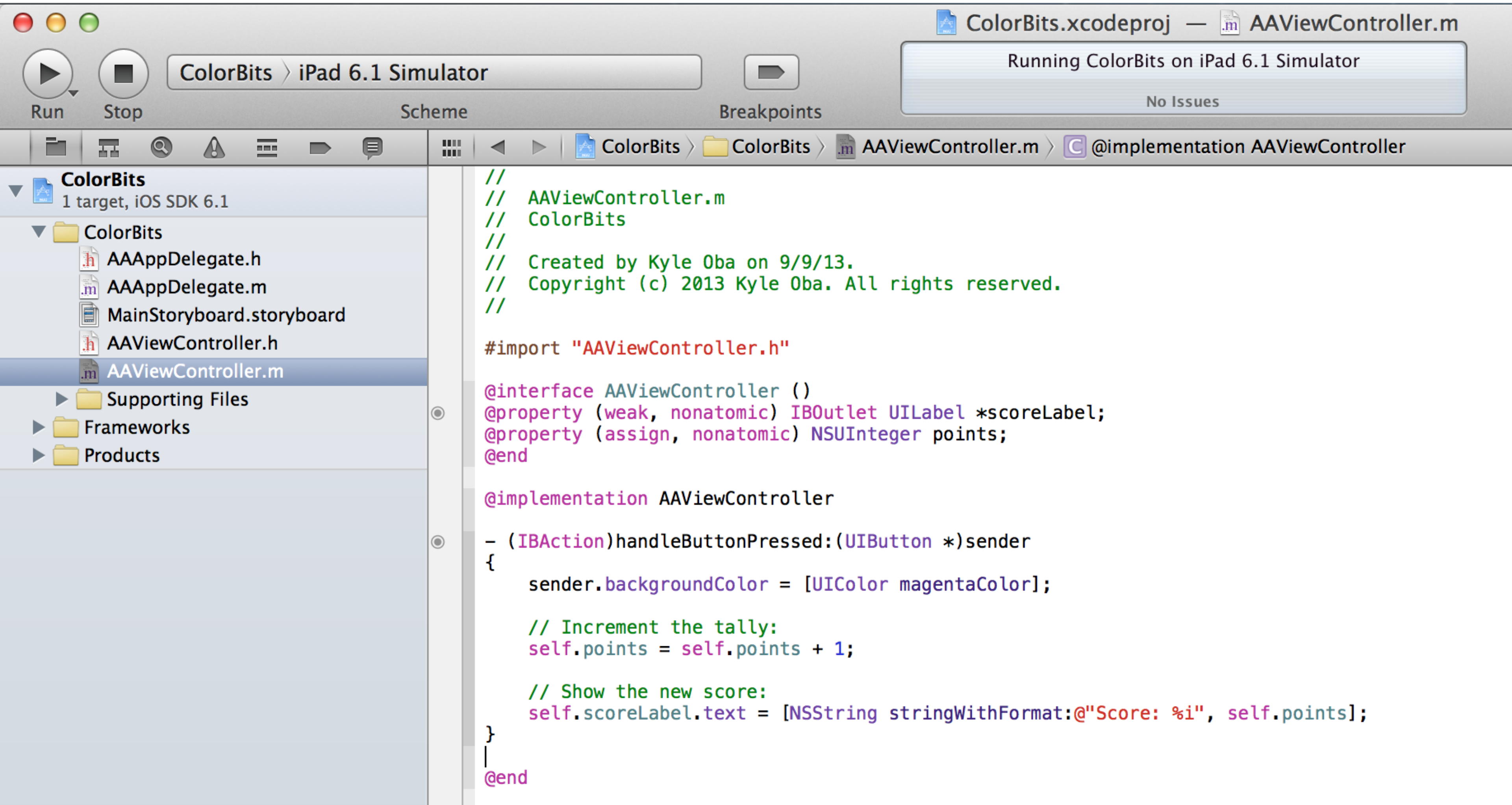
It’s kind of like time travel. You get to travel into the future and tell your self just what you were thinking when you wrote that code.



```
- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];

    // Increment the tally:
    self.points = self.points + 1;

    // Show the new score:
    self.scoreLabel.text = [NSString stringWithFormat:@"Score: %i", self.points];
}
```

It should look like this.

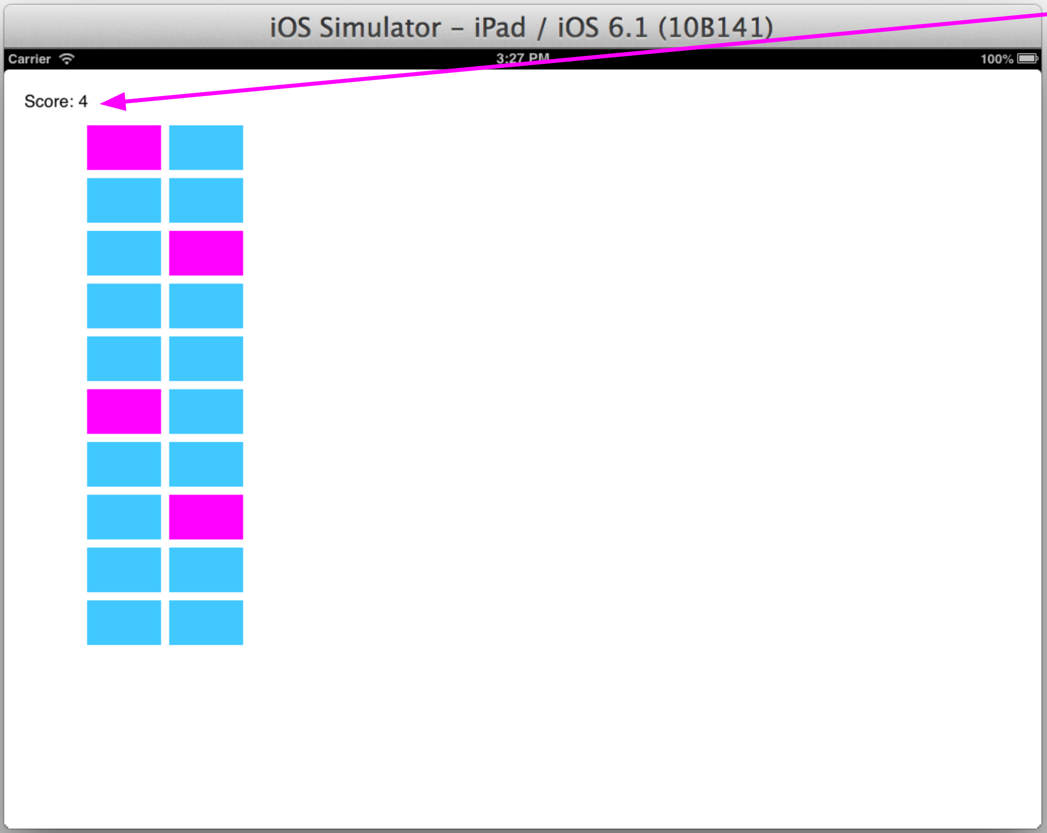
Make sense? You can run it now.



ColorBits.xcodeproj — AAViewController.m
Run
Stop
ColorBits › iPad 6.1 Simulator
Scheme
Breakpoints
Running ColorBits on iPad 6.1 Simulator
No Issues
ColorBits › ColorBits › AAViewController.m › @implementation AAViewController
ColorBits
1 target, iOS SDK 6.1
ColorBits
AAAppDelegate.h
AAAppDelegate.m
MainStoryboard.storyboard
AAViewController.h
AAViewController.m
Supporting Files
Frameworks
Products

```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All rights reserved.
//

#import "AAViewController.h"

@interface AAViewController ()
@property (weak, nonatomic) IBOutlet UILabel *scoreLabel;
@property (assign, nonatomic) NSUInteger points;
@end

@implementation AAViewController

- (IBAction)handleButtonPressed:(UIButton *)sender
{
    sender.backgroundColor = [UIColor magentaColor];

    // Increment the tally:
    self.points = self.points + 1;

    // Show the new score:
    self.scoreLabel.text = [NSString stringWithFormat:@"Score: %i", self.points];
}

@end
```

Yay! Scores!



The scores label should now be changing every time you tap on a button.

Recap

You’ve now seen all the basic parts of a Storyboard:

- View Controllers
- Controls - Like these buttons.
- Segues
- Actions
- Outlets

With all this under your belt, you should we can dive into all that code-y stuff.

You have no doubt noticed that I’ve been doing a lot arm waving whenever code topics come up.

We’ll I’m going to stop doing the arm waving thing and dive into some much better explanations.

We’ll start that next.

Change things

Now that you’ve got this little demo working. Here are a few things you can do to test your understanding of how all this works.

- Change the buttons’ starting colors, so they are not all initially the same color.
- Add more buttons (clone them), so that you fill up the entire screen.

Change things : Crazy

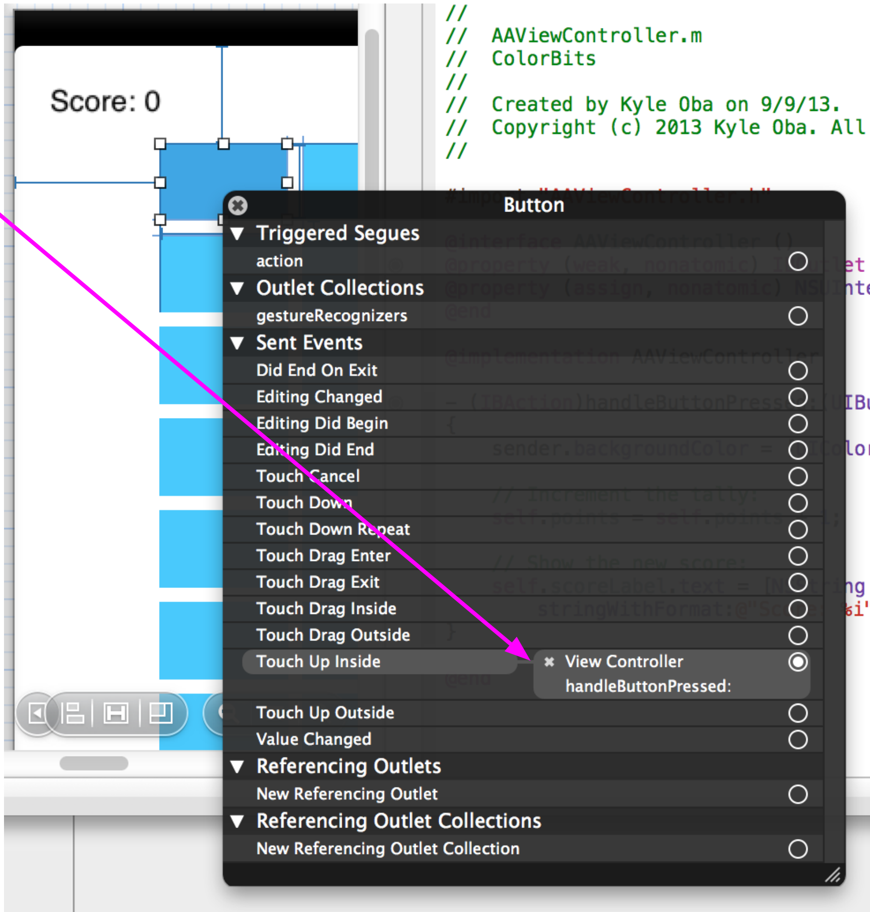
If you’re ready to get crazy:

- Remove the Action from *some* of the buttons. You can do this by two-finger tapping a button and hitting the close “x” next to the current connection.

- Repeat what you did in the lab to create a new Action for this button. Don’t reuse the old Action, create a totally new one, with a different name.

- In the new Action, change the background color to something other than magenta.

- Now you have a button that changes the color to something else. Perhaps that button deserves more points per tap?



```
//
//  AAViewController.m
//  ColorBits
//
//  Created by Kyle Oba on 9/9/13.
//  Copyright (c) 2013 Kyle Oba. All
//
```