Even More Storyboarding: Actions

LAB 04: Actions

Objective

You now know the basics of **Storyboarding**. However, in order to unleash the full power of iOS you're going to have to write some code.

Writing code… we're not going to open that can of worms yet. First we'll take a look at the bridge between **Storyboards** and coding.

This bridge is comprised of:

- Actions (We'll cover this now.)
- Outlets (We'll cover this in the next lab.)

Actions are like the exploding dynamite on the other end of the big red button. They're the robot arm at the end of the robot arm on a stick toy that you get at Dave & Busters. They're the **Action** that happens in response to the user doing something (like pressing a button).

But, how do they work? Let's find out.

NOTE: I'm not going to show you how to do the things that we covered in previous labs. I'm now assuming you know how to do things like start up Xcode and create a new project.

Create another project for the iPad

This project is going to look a lot like the first one (the one from LAB 1). That's the one where you created all the colored rectangular views.

Now, instead of using plain ol' UIViews, we're going to use UIButtons.

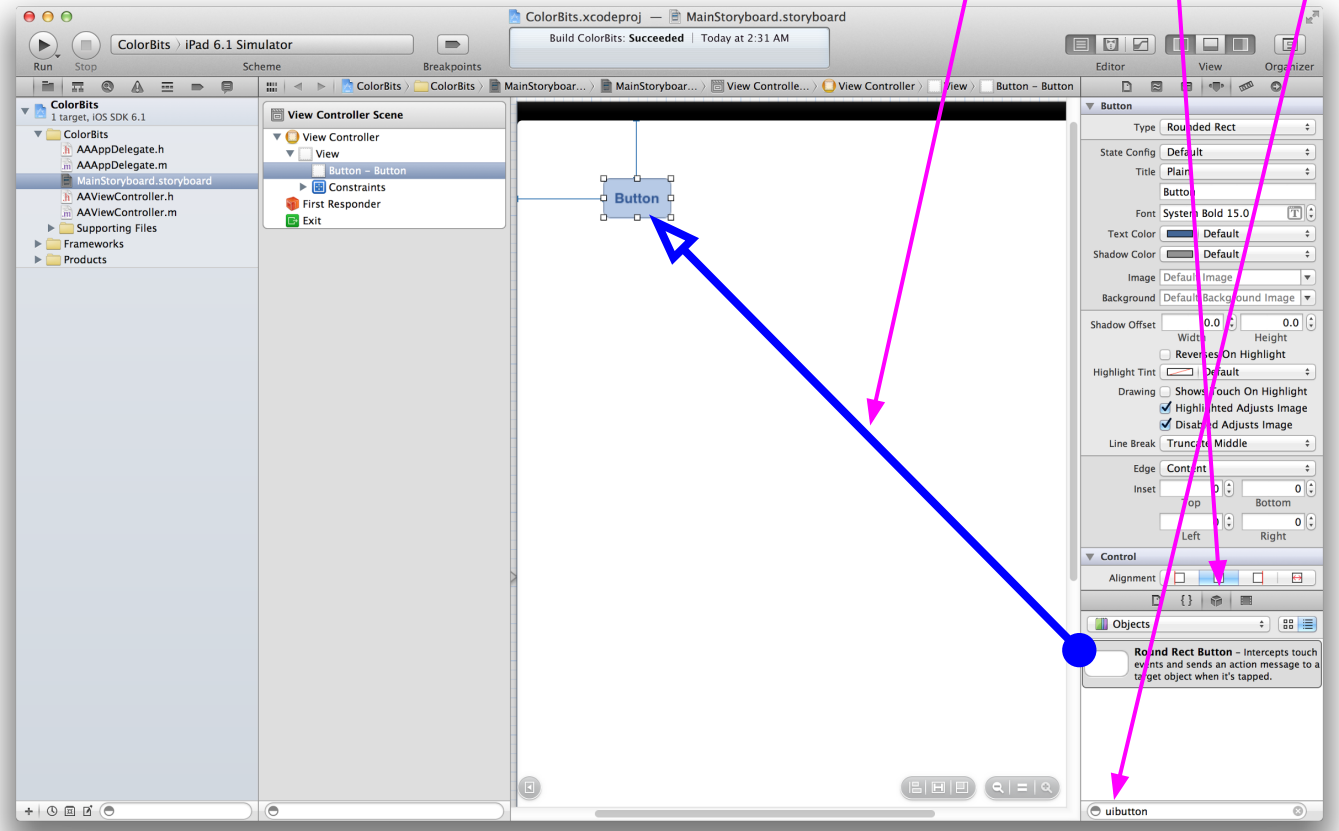
Create another iPad project, using the same settings as before:.

- Single View Application
- Devices: iPad
- Use Storyboards
- Use Automatic Reference Counting

Add a button. Please.

Use the **Object library** to find the **UIButton** object.

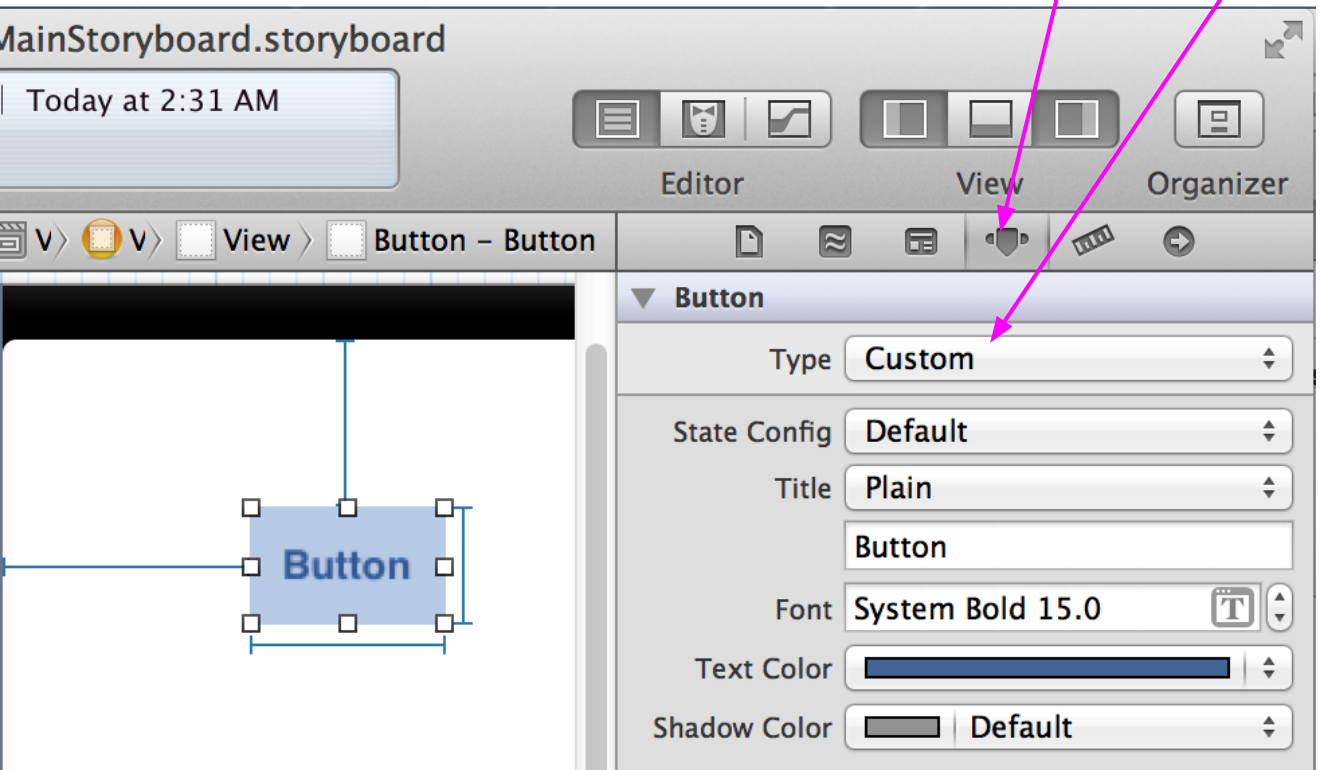
Drag one onto the **Storyboard**.



Customize the button

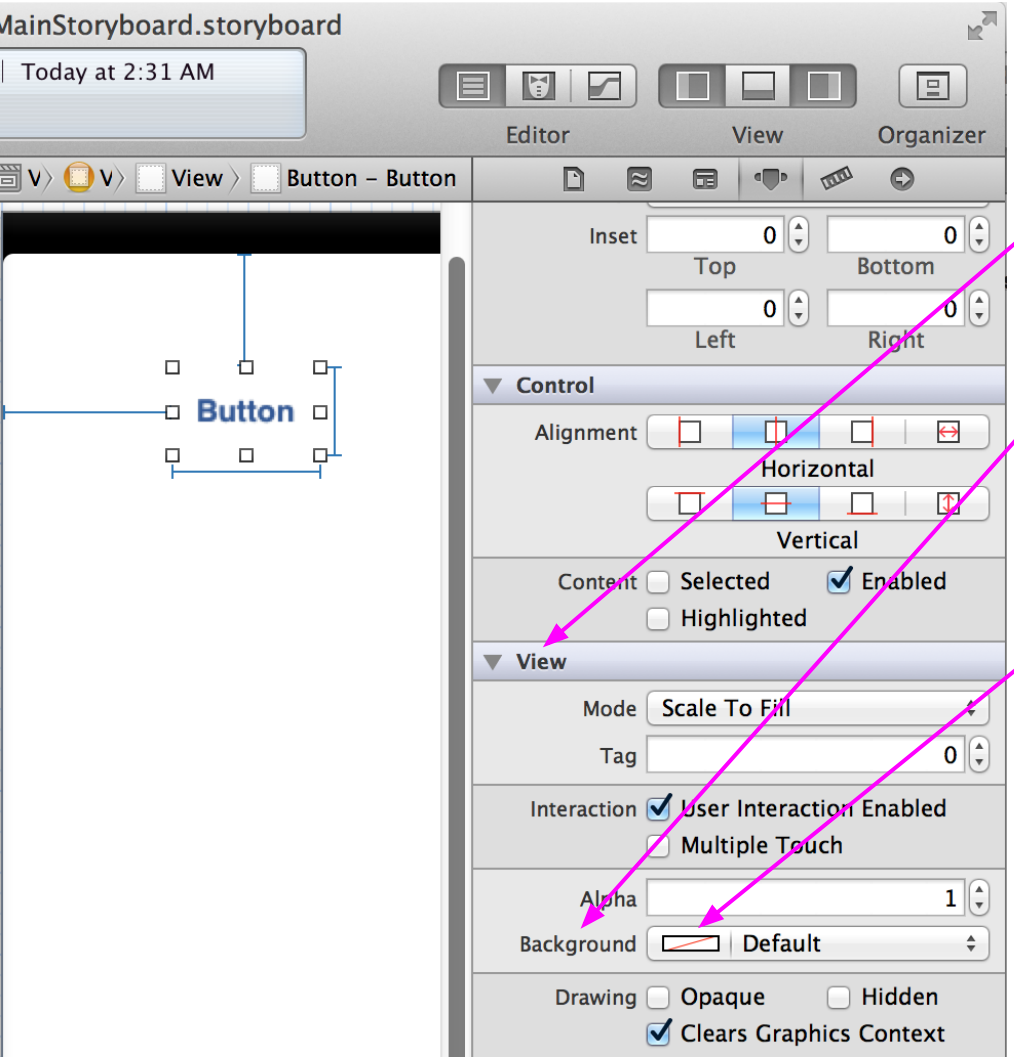
We want to make this button look like one of those colored rectangles we were using in LAB 1.

In the **Attributes inspector**, change the type from **Rounded Rect** to **Custom**.



This allows us to customize the look of the button. In our case, we want to choose a custom background color. We'll do this next.

Button background color



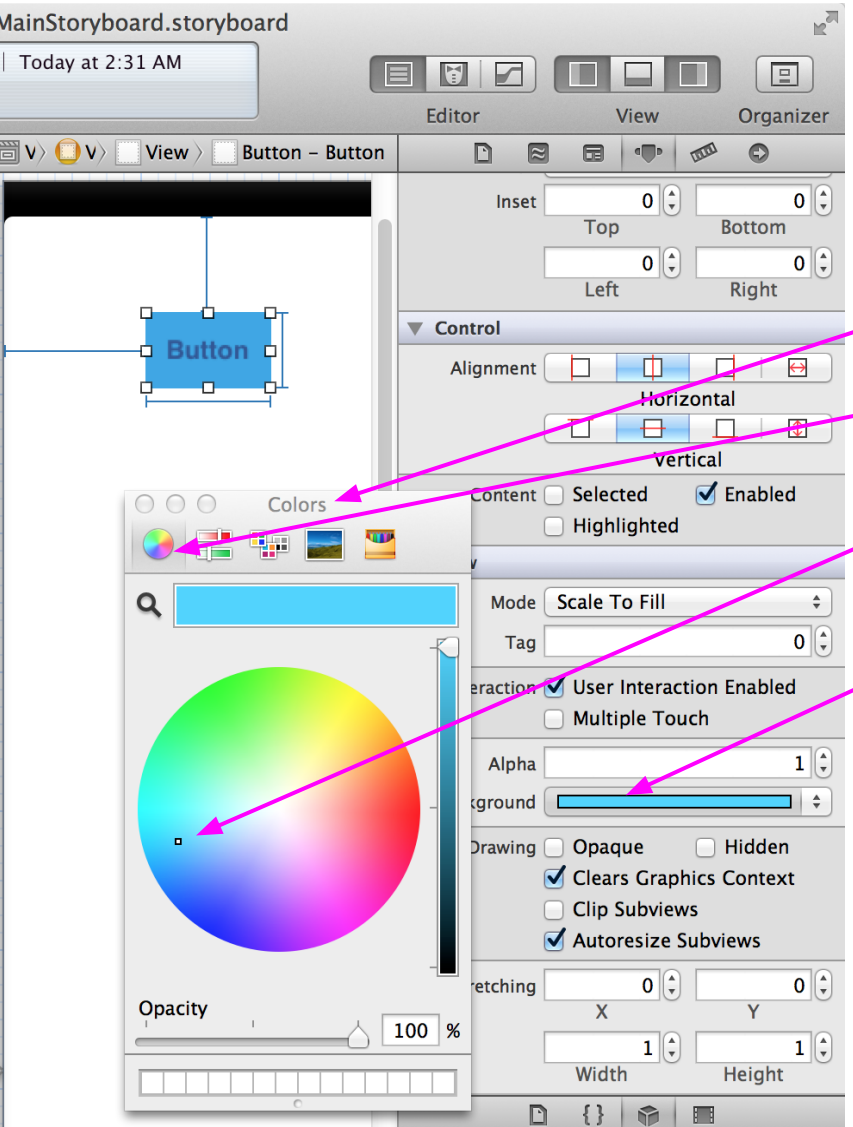
In the **Attributes inspector**, scroll down until you see the **View** section of attributes.

Find the **Background** attribute and click on the colored bar.

The colored bar will be white with a red diagonal line through it.

Click on it...

Select a new background



Once you click on the color bar, the color picker window will appear.

The default color selection method is the color wheel.

From here, you can click on the wheel to select a background color for the button.

The **Background** color attribute is now filled with your selection.

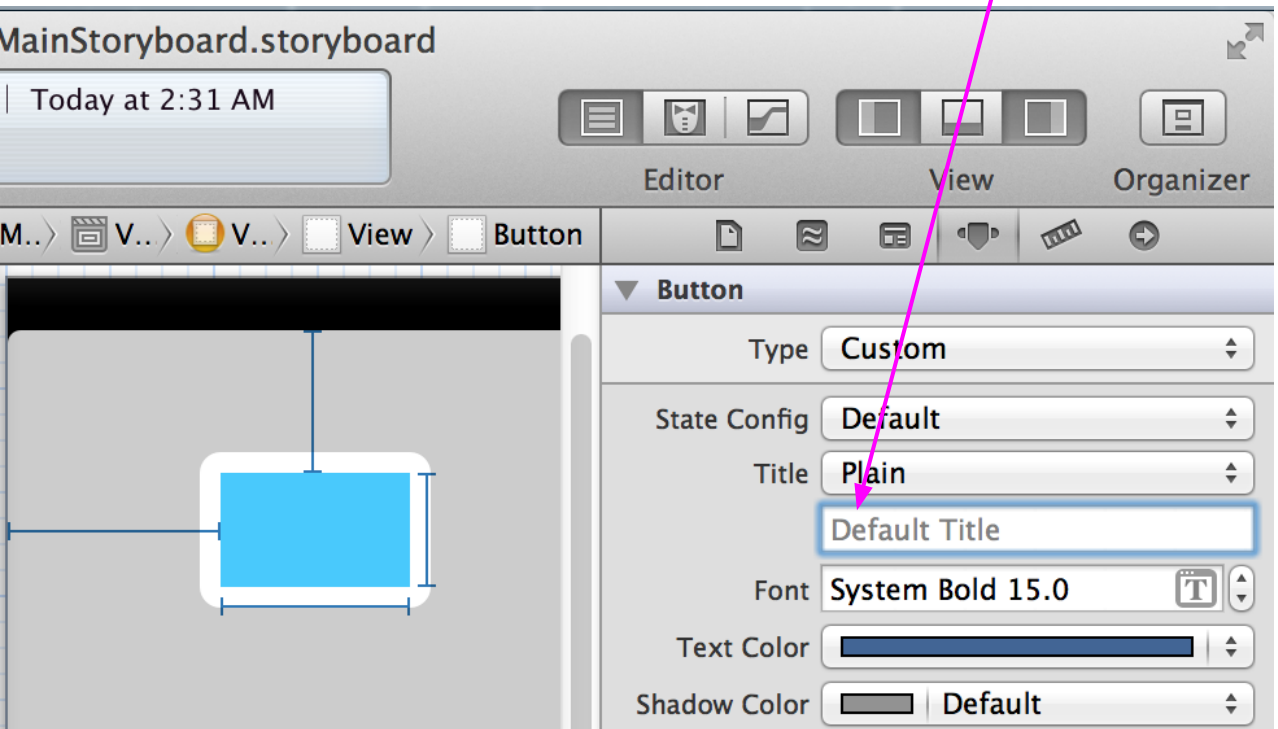
You're done with the color picker window. Close it now.

Yes, it's a button.

We don't need the text on the button that tells you that it's a "button."

Scroll to the top of the **Attributes inspector** again and remove the **Title** text.

You must hit tab to move the selection focus out of that field in order for Xcode to read your changes.



Once you do that, the button will no longer contain the word "button" in it.

If you run your project now, you' ll see that you have a tappable button. It looks and acts like a button. See how it changes tinting when pressed?

But, it doesn't do anything. Yet.

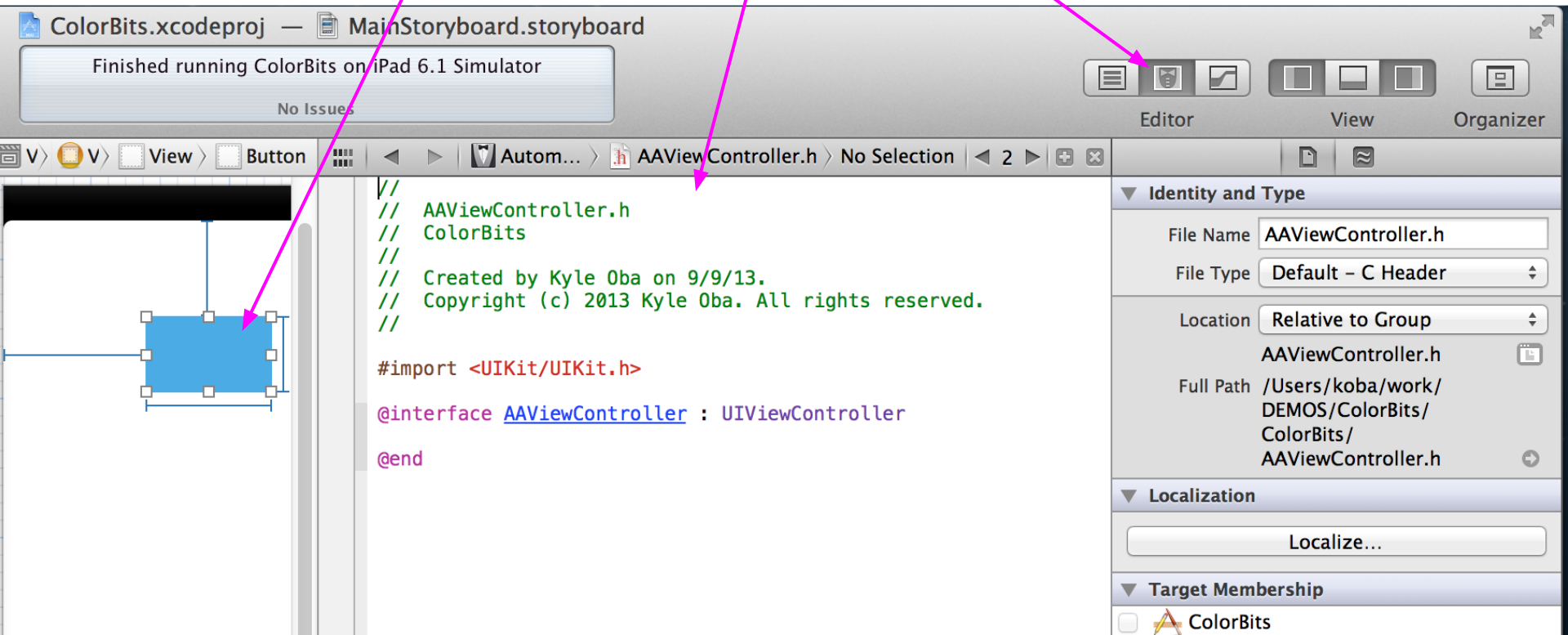
Fire up the Assistant Editor

With your button selected in the **Storyboard**.

Tap to turn on the **Assistant editor** in the top menu.

This displays a new pane in the coding area.

This pane contains code that is related to what you have selected in the **Storyboard**.



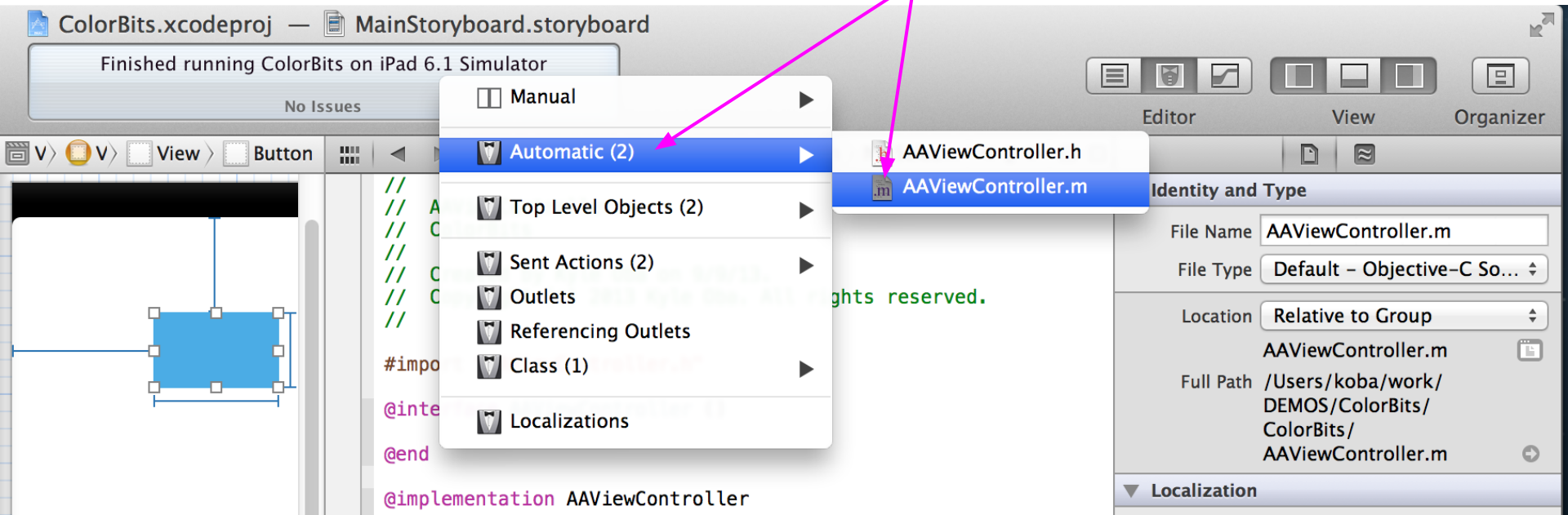
Select the implementation file.

The **Assistant editor** tries to be helpful. It tries to figure out which file you'd like to edit while working with your button.

Unfortunately, it guesses wrong sometimes.

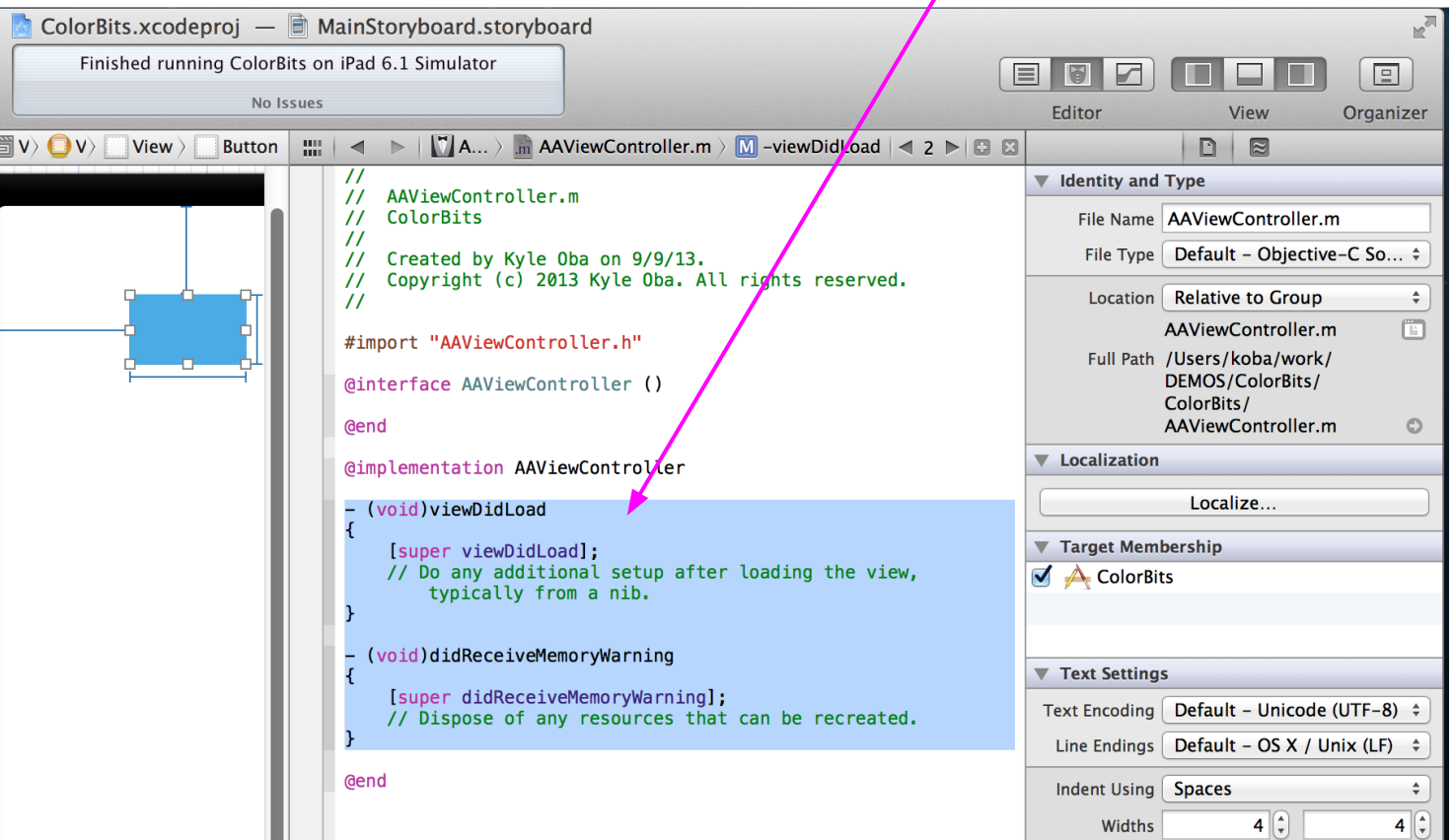
Tell it which file you *really* want to edit by clicking on the **Automatic** menu item.

Then, select your **AAViewController.m** file. Or, whatever you named your main View Controller's implementation (.m) file.



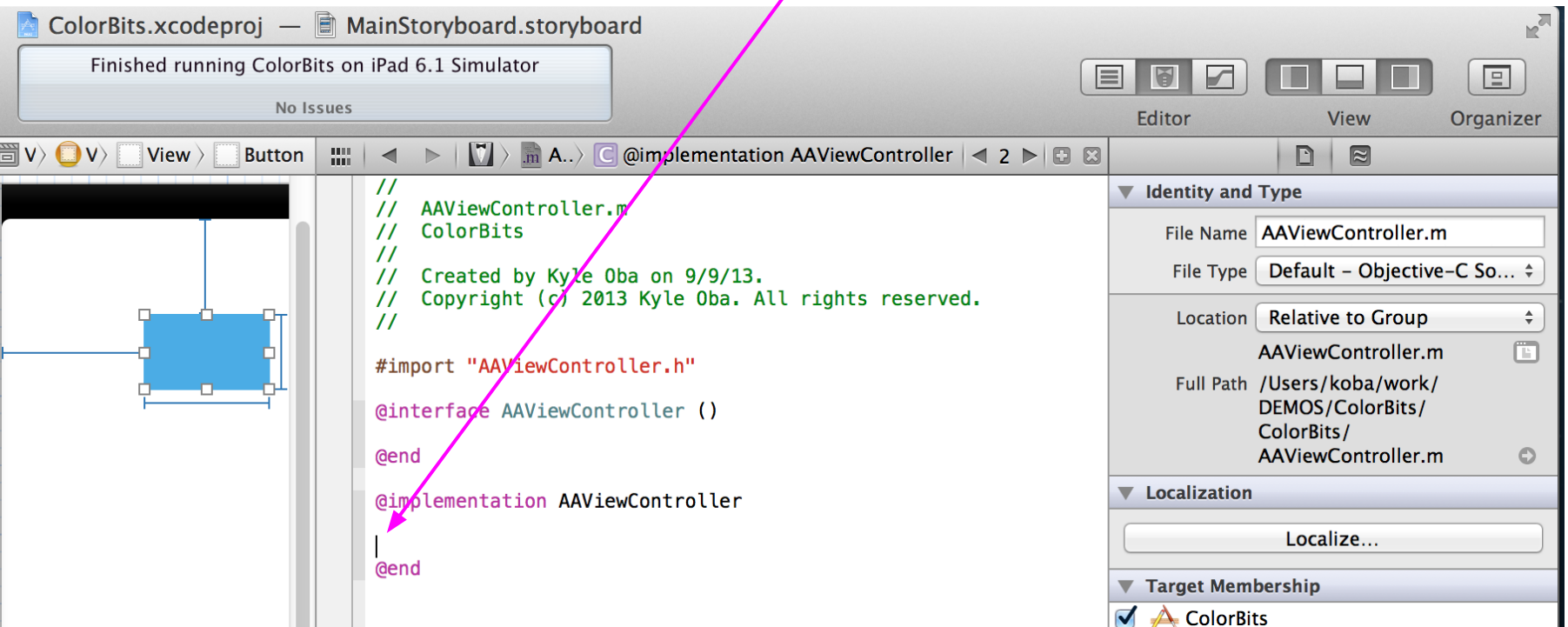
Select the implementation contents

Select all of code between “@implementation AAViewController” and the final “@end” line.



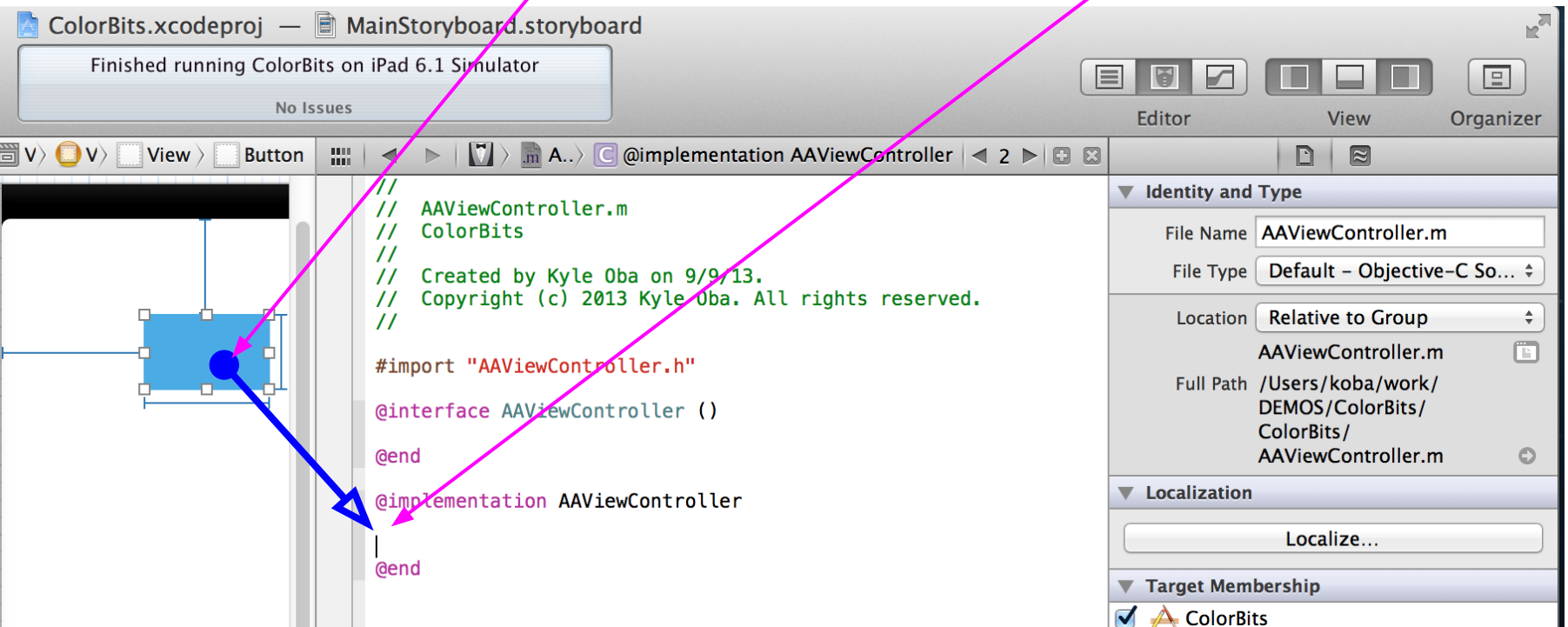
Delete the implementation contents

Delete the selected contents of the implementation section.



Add a button Action

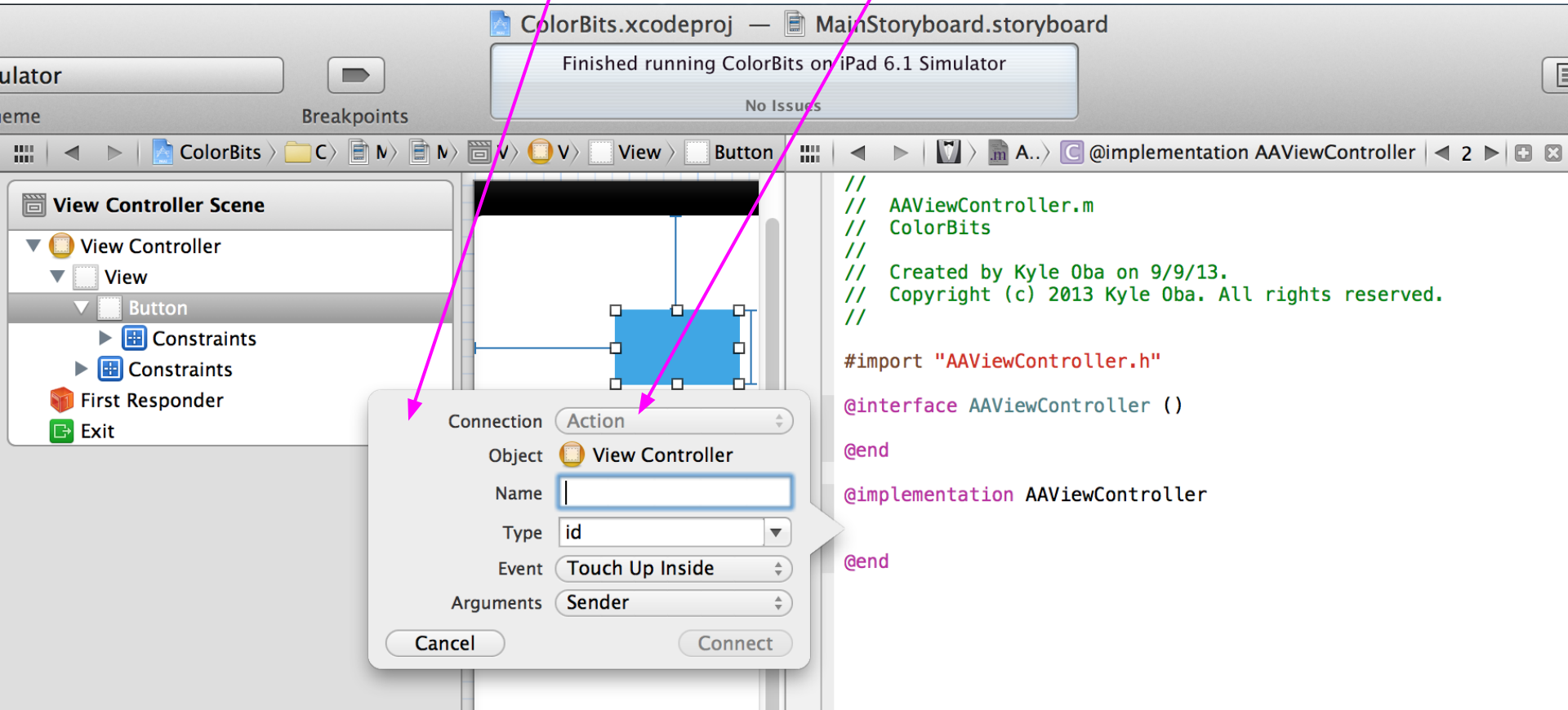
In order to add an **Action** for your button, hold down the **control** key, then click and drag from the button to the implementation file here.



Release (the drag and drop)

When you release, you will see this context menu pop up.

This is where you configure **Outlets** and **Actions**. Since you dropped a connection from a button into an implementation file, Xcode knows that you're trying to create an **Action**.

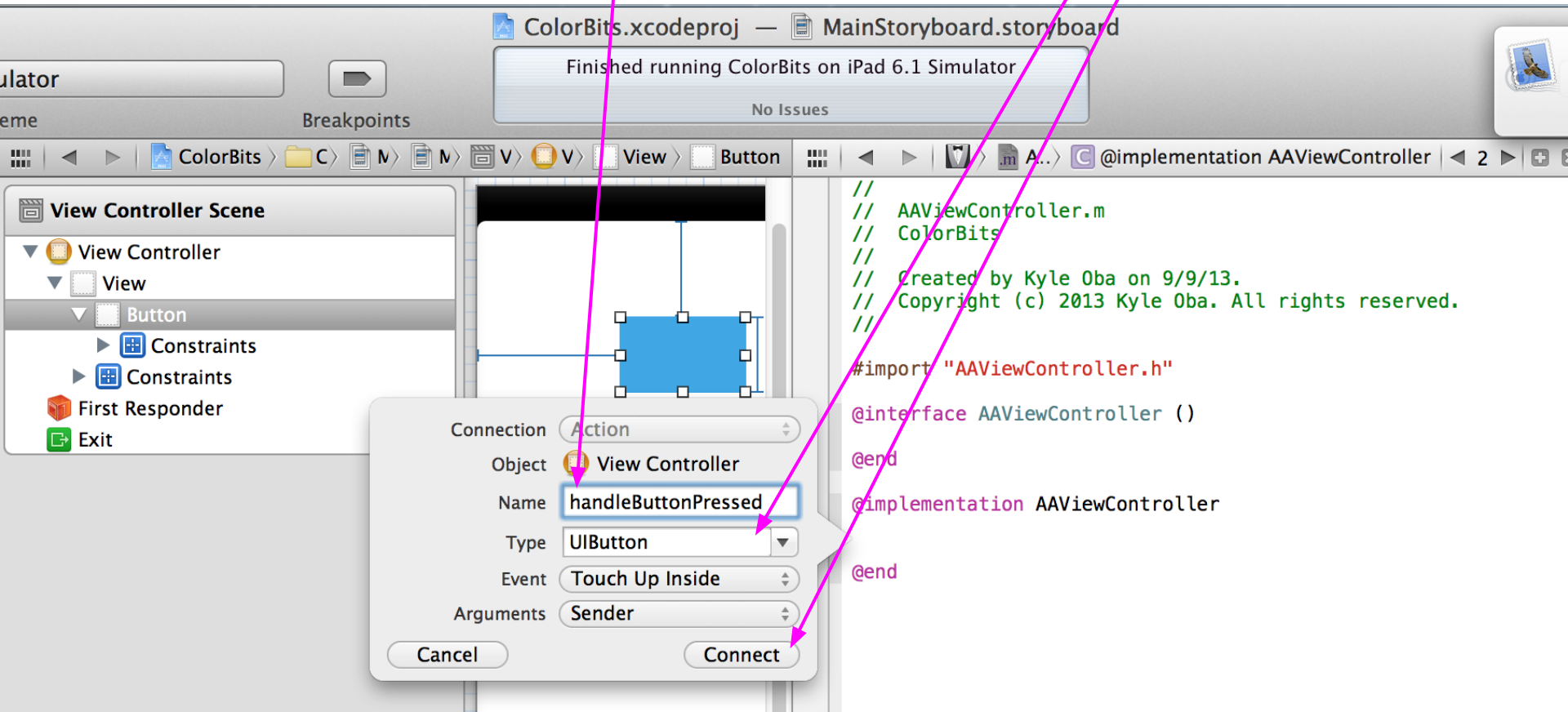


Configure the Action

There are two things you need to do to configure this **Action**.

1) Name it. **2)** Specify the argument type as **UIButton**.

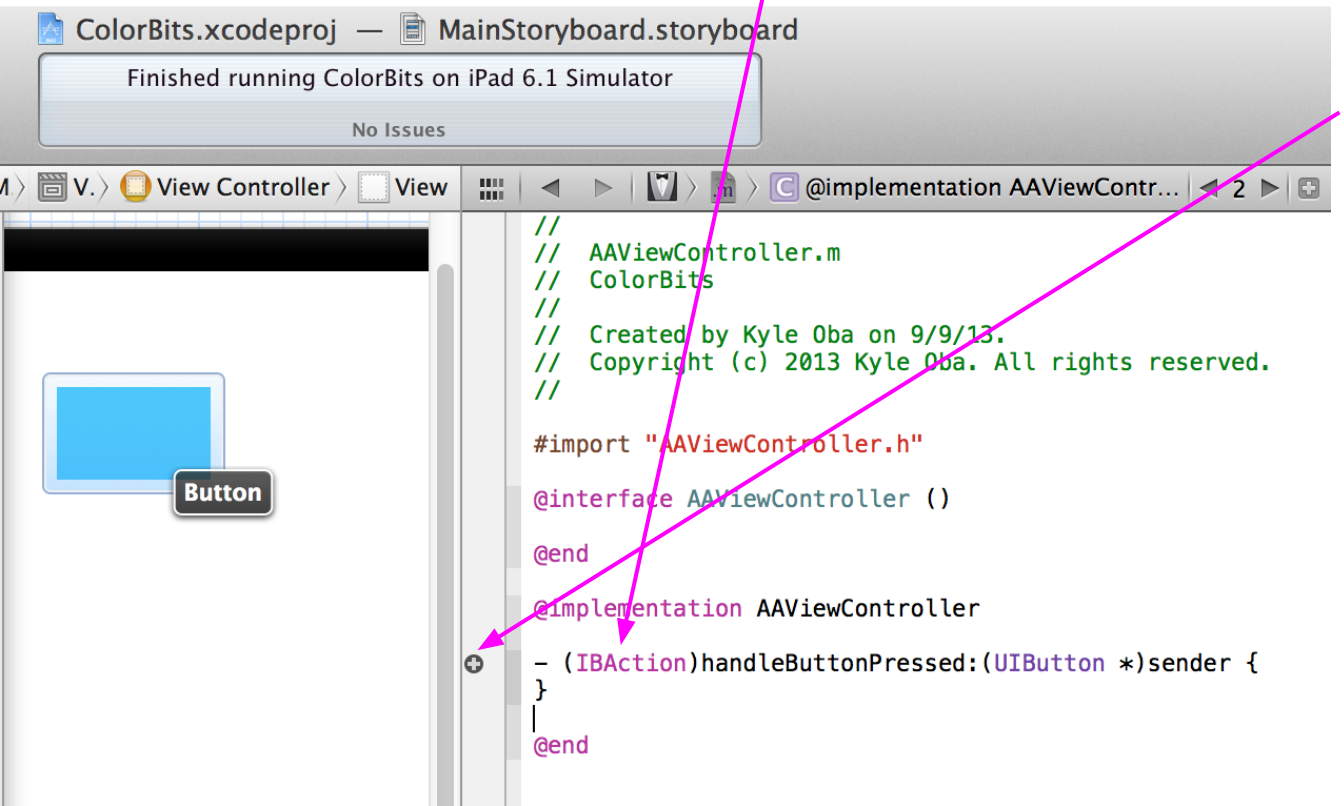
After you've done that, hit the **Connect** button.



Code!

Xcode has just dumped little bit of code onto your implementation file. Don't be alarmed. You'll learn all about how that code works later.

The important thing to understand is that Xcode has created an **Action** method for you.



Hover your mouse over the little connector dot, and you'll see that it has also connected this Action to your button.

More Code!

Today is an (possibly) important day in your professional life as a programmer. It is (possibly) the first day that someone is going to ask you to write some code, but isn't going to offer to pay you for it.

Normally, I'd advise you not to do it. But, since you're learning, please do this.

Copy this code into Xcode. It should go into the **Action** you just created.

```
#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

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

@end
```

Take a minute to look at what this **Action** does.

The sender is the button.

The button has a **backgroundColor** property.

We are setting that backgroundColor to a **UIColor**, called **magentaColor**.

Sanity Check

Don’t worry if you have no idea how any of that code works.

We’ll cover all of that code stuff in class, and later in the course.

The important thing is to copy it into the action and make sure it looks exactly the same.

```
#import "AAViewController.h"

@interface AAViewController ()

@end

@implementation AAViewController

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

@end
```

Run it.

Press the **Run** button on Xcode. You should see your simulator again.
(If you're targeting your iPad, it will run on your iPad.)
Tap on the button. It should turn magenta.