Stanford CS193p

Developing Applications for iOS
Fall 2013-14

Today

- Introduction to Objective-C (con't)
  Continue showing Card Game Model with `Deck`, `PlayingCard`, `PlayingCardDeck`
- Xcode 5 Demonstration
  Start building the simple Card Game

Objective-C

Card.h

```
#import <Foundation/Foundation.h>



@interface Card : NSObject

@property (strong, nonatomic) NSString *contents;

@property (nonatomic, getter=isChosen) BOOL chosen;
@property (nonatomic, getter=isMatched) BOOL matched;

- (int)match:(NSArray *)otherCards;









@end
```

Card.m

```
#import "Card.h"

@interface Card()

@end

@implementation Card




- (int)match:(NSArray *)otherCards
{
    int score = 0;

    for (Card *card in otherCards) {
        if ([card.contents isEqualToString:self.contents]) {
            score = 1;
        }
    }

    return score;
}



@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>

@interface Deck : NSObject




@end
```

Let's look at another class.
This one represents a deck of cards.

Deck.m

```
#import "Deck.h"

@interface Deck()

@end

@implementation Deck

@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;

- (Card *)drawRandomCard;

@end
```

Note that this method has 2 arguments (and returns nothing). It's called "addCard:atTop:".

And this one takes no arguments and returns a Card (i.e. a pointer to an instance of a Card in the heap).

Deck.m

```
#import "Deck.h"

@interface Deck()

@end

@implementation Deck

@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;


- (Card *)drawRandomCard;

@end
```

We must #import the header file for any class we use in this file (e.g. Card).

Deck.m

```
#import "Deck.h"

@interface Deck()

@end

@implementation Deck









@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Arguments to methods (like the `atTop:` argument) are never "optional."

However, if we want an `addCard:` method without `atTop:`, we can define it separately.

Deck.m

```
#import "Deck.h"

@end

@implementation Deck



- (void)addCard:(Card *)card atTop:(BOOL)atTop
{



}



- (Card *)drawRandomCard { }

@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Arguments to methods (like the atTop: argument) are never "optional."

However, if we want an addCard: method without atTop:, we can define it separately.

And then simply implement it in terms of the the other method.

Deck.m

```
#import "Deck.h"
@end
@implementation Deck



- (void)addCard:(Card *)card atTop:(BOOL)atTop
{




}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

A deck of cards obviously needs some storage to keep the cards in. We need an `@property` for that. But we don't want it to be public (since it's part of our private, internal implementation).

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (void)addCard:(Card *)card atTop:(BOOL)atTop
{

}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

So we put the `@property` declaration we need here in our `@implementation`.

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (void)addCard:(Card *)card atTop:(BOOL)atTop
{
    if (atTop) {
        [self.cards insertObject:card atIndex:0];
    } else {
        [self.cards addObject:card];
    }
}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

Now that we have a property to store our cards in, let's take a look at a sample implementation of the addCard:atTop: method.

self.cards is an NSMutableArray ...

... and these are NSMutableArray methods. (insertObject:atIndex: and addObject:).

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

But there's a problem here.
When does the object pointed to by the pointer returned by `self.cards` ever get created?

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (void)addCard:(Card *)card atTop:(BOOL)atTop
{
    if (atTop) {
        [self.cards insertObject:card atIndex:0];
    } else {
        [self.cards addObject:card];
    }
}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

Declaring a `@property` makes space in the instance for the pointer itself, but not does not allocate space in the heap for the object the pointer points to.

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

The place to put this needed heap allocation is in the getter for the cards @property.

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    return _cards;
}


- (void)addCard:(Card *)card atTop:(BOOL)atTop
{
    if (atTop) {
        [self.cards insertObject:card atIndex:0];
    } else {
        [self.cards addObject:card];
    }
}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

The place to put this needed heap allocation is in the getter for the cards @property.

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    if (!_cards) _cards = [[NSMutableArray alloc] init];
    return _cards;
}

- (void)addCard:(Card *)card atTop:(BOOL)atTop
{
    if (atTop) {
        [self.cards insertObject:card atIndex:0];
    } else {
        [self.cards addObject:card];
    }
}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

All properties start out with a value of 0 (called nil for pointers to objects).
So all we need to do is allocate and initialize the object if the pointer to it is nil.
This is called "lazy instantiation".
Now you can start to see the usefulness of a @property.

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    if (!_cards) _cards = [[NSMutableArray alloc] init];
    return _cards;
}

- (void)addCard:(Card *)card atTop:(BOOL)atTop
{
    if (atTop) {
        [self.cards insertObject:card atIndex:0];
    } else {
        [self.cards addObject:card];
    }
}

- (void)addCard:(Card *)card
{
    [self addCard:card atTop:NO];
}

- (Card *)drawRandomCard { }

@end
```

We'll talk about allocating and initializing objects more later, but here's a simple way to do it.

Objective-C

Deck.h

```objc
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Deck.m

```objc
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    if (!_cards) _cards = [[NSMutableArray alloc] init];
    return _cards;
}

- (void)addCard:(Card *)card atTop:(BOOL)atTop { ... }
- (void)addCard:(Card *)card { ... }

- (Card *)drawRandomCard
{
    Card *randomCard = nil;

    return randomCard;
}

@end
```

drawRandomCard simply grabs a card from a random spot in our `self.cards` array.

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    if (!_cards) _cards = [[NSMutableArray alloc] init];
    return _cards;
}

- (void)addCard:(Card *)card atTop:(BOOL)atTop { ... }
- (void)addCard:(Card *)card { ... }

- (Card *)drawRandomCard
{
    Card *randomCard = nil;

    unsigned index = arc4random() % [self.cards count];
    randomCard = self.cards[index];
    [self.cards removeObjectAtIndex:index];

    return randomCard;
}

@end
```

arc4random() returns a random integer.

This is the C modulo operator.

These square brackets actually are the equivalent of sending the message objectAtIndexedSubscript: to the array.

Objective-C

Deck.h

```
#import <Foundation/Foundation.h>
#import "Card.h"

@interface Deck : NSObject

- (void)addCard:(Card *)card atTop:(BOOL)atTop;
- (void)addCard:(Card *)card;

- (Card *)drawRandomCard;

@end
```

Deck.m

```
#import "Deck.h"

@interface Deck()
@property (strong, nonatomic) NSMutableArray *cards; // of Card
@end

@implementation Deck

- (NSMutableArray *)cards
{
    if (!_cards) _cards = [[NSMutableArray alloc] init];
    return _cards;
}

- (void)addCard:(Card *)card atTop:(BOOL)atTop { ... }
- (void)addCard:(Card *)card { ... }

- (Card *)drawRandomCard
{
    Card *randomCard = nil;

    if ([self.cards count]) {
        unsigned index = arc4random() % [self.cards count];
        randomCard = self.cards[index];
        [self.cards removeObjectAtIndex:index];
    }

    return randomCard;
}

@end
```

Calling `objectAtIndexedSubscript:` with an argument of zero on an empty array will **crash** (array index out of bounds)!

So let's protect against that case.

Objective-C

PlayingCard.h

PlayingCard.m

Let's see what it's like to make a subclass of one of our own classes.
In this example, a subclass of `Card` specific to a playing card (e.g. A♠).

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard


@end
```

Of course we must #import our superclass.

And #import our own header file in our implementation file.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard



@end
```

A PlayingCard has some properties that a vanilla Card doesn't have.
Namely, the PlayingCard's suit and rank.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

@end
```

We'll represent the suit as an `NSString` that simply contains a single character corresponding to the suit (i.e. one of these characters: ♠♣♥♦).
If this property is `nil`, it'll mean "suit not set".

We'll represent the rank as an integer from 0 (rank not set) to 13 (a King).

`NSUInteger` is a typedef for an unsigned integer.

We could just use the C type `unsigned int` here.
It's mostly a style choice.
Many people like to use `NSUInteger` and `NSInteger` in public API and `unsigned int` and `int` inside implementation.
But be careful, `int` is 32 bits, `NSInteger` might be 64 bits.
If you have an `NSInteger` that is really big (i.e. > 32 bits worth) it could get truncated if you assign it to an `int`.
Probably safer to use one or the other everywhere.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{

    return [NSString stringWithFormat:@"%d%@", self.rank, self.suit];
}



@end
```

Users of our PlayingCard class might well simply access suit and rank properties directly. But we can also support our superclass's contents property by overriding the getter to return a suitable (no pun intended) NSString.

Even though we are overriding the implementation of the contents method, we are not re-declaring the contents property in our header file. We'll just inherit that declaration from our superclass.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{

    return [NSString stringWithFormat:@"%d%@", self.rank, self.suit];
}


@end
```

The method `stringWithFormat:` is an `NSString` method that's sort of like using the C function `printf` to create the string.

Note we are creating an `NSString` here in a different way than `alloc/init`. We'll see more about "class methods" like `stringWithFormat:` a little later.

Users of our `PlayingCard` class might well simply access `suit` and `rank` properties directly. But we can also support our superclass's `contents` property by overriding the getter to return a suitable (no pun intended) `NSString`.

Even though we are overriding the implementation of the `contents` method, we are not re-declaring the `contents` property in our header file. We'll just inherit that declaration from our superclass.

- Java: string concatenation, e.g.: "Hello" + var + "world"

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{

    return [NSString stringWithFormat:@"%d%@", self.rank, self.suit];
}



@end
```

Calling the getters of our two properties (rank and suit) on ourself.

But this is a pretty bad representation of the card (e.g., it would say 11♣ instead of J♣ and 1♥ instead of A♥).

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}
```

We'll create an `NSArray` of `NSString`s, each of which corresponds to a given rank.
Again, `0` will be "rank not set" (so we'll use ?).
11, 12 and 13 will be `J Q K` and 1 will be `A`.

Then we'll create our "J♠" string by appending (with the `stringByAppendingString:` method) the suit onto the end of the string we get by looking in the array.

```
@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}



@end
```

Notice the @[] notation to create an array.

Here's the array-accessing [] notation again (like we used with `self.cards[index]` earlier).

Also note the @"" notation to create a (constant) `NSString`.

All of these notations are converted into normal message-sends by the compiler.
For example, `@[ ... ]` is `[[NSArray alloc] initWithObjects:...]`.
`rankStrings[self.rank]` is `[rankStrings objectAtIndexedSubscript:self.rank]`.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}
```

This is nice because a "not yet set" rank shows up as ?.

But what about a "not yet set" suit?
Let's override the getter for suit to make a suit of nil return ?.

Yet another nice use for properties versus direct instance variables.

```
- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}
```

Let's take this a little further and override the setter for `suit` to have it check to be sure no one tries to set a `suit` to something invalid.

```
- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}









- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Notice that we can embed the array creation as the target of this message send. We're simply sending `containsObject:` to the array created by the `@[ ]`.

`containsObject:` is an `NSArray` method.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;


@end
```

But there's a problem here now.
A compiler warning will be generated if we do this.
Why?
Because if you implement BOTH the setter and the getter for a property, then you have to create the instance variable for the property yourself.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}









- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;


@end
```

But there's a problem here now.
A compiler warning will be generated if we do this.
Why?
Because if you implement BOTH the setter and the getter for a property, then you have to create the instance variable for the property yourself.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
```

Luckily, the compiler can help with this using the `@synthesize` directive.

If you implement only the setter OR the getter (or neither), the compiler adds this `@synthesize` for you.

```
- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
```

Name of the property we're creating an instance variable for.

Name of the instance variable to associate with the property.

We almost always pick an instance variable name that is underbar followed by the name of the property.

```
- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
```

You should only ever access the instance variable directly ...

```
- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

... in the property's setter ...

... in its getter ...

... or in an initializer (more on this later).

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

All of the methods we've seen so far are "instance methods".
They are methods sent to instances of a class.
But it is also possible to create methods that are sent to the class itself.
Usually these are either creation methods (like `alloc` or `stringWithFormat:`) or utility methods.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter




- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;


@end
```

Class methods start with +
Instance methods start with –

Here's an example of a class utility method which returns an NSArray of the NSStrings which are valid suits (e.g. ♠, ♣, ♥, and ♦).

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter

+ (NSArray *)validSuits
{
    return
}

- (void)setSuit:(NSString *)suit
{
    if ([@[@"♥",@"♦",@"♠",@"♣"] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Since a class method is not sent to an instance, we cannot reference our properties in here (since properties represent per-instance storage).

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

Here's an example of a class utility method which returns an NSArray of the NSStrings which are valid suits (e.g. ♠, ♣, ♥, and ♦).

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter

+ (NSArray *)validSuits
{
    return @[@"♥",@"♦",@"♠",@"♣"];
}

- (void)setSuit:(NSString *)suit
{
    if ([                          containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

We actually already have the array of valid suits, so let's just move that up into our new class method.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter

+ (NSArray *)validSuits
{
    return @[@"♥",@"♦",@"♠",@"♣"];
}

- (void)setSuit:(NSString *)suit
{
    if ([[PlayingCard validSuits] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Now let's invoke our new class method here.

See how the name of the class appears in the place you'd normally see a pointer to an instance of an object?

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;



@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter

+ (NSArray *)validSuits
{
    return @[@"♥",@"♦",@"♠",@"♣"];
}

- (void)setSuit:(NSString *)suit
{
    if ([[PlayingCard validSuits] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Now let's invoke our new class method here.

See how the name of the class appears in the place you'd normally see a pointer to an instance of an object?

It'd probably be instructive to go back and look at the invocation of the NSString class method stringWithFormat: a few slides ago.

Also, make sure you understand that stringByAppendingString: above is not a class method, it is an instance method.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;


@end
```

The validSuits class method might be useful to users of our PlayingCard class, so let's make it public.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter

+ (NSArray *)validSuits
{
    return @[@"♥",@"♦",@"♠",@"♣"];
}

- (void)setSuit:(NSString *)suit
{
    if ([[PlayingCard validSuits] containsObject:suit]) {
        _suit = suit;
    }
}

- (NSString *)suit
{
    return _suit ? _suit : @"?";
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;


@end
```

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }




@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;


@end
```

Let's move our other array (the strings of the ranks) into a class method too.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings =
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }

+ (NSArray *)rankStrings
{
    return @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
}



@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;


@end
```

We'll leave this one private because the public API for the rank is purely numeric.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = [PlayingCard rankStrings];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }

+ (NSArray *)rankStrings
{
    return @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
}

@end
```

And now let's call that class method.

Note that we are not required to declare this earlier in the file than we use it.

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;
+ (NSUInteger)maxRank;

@end
```

But here's another class method that might be good to make public.

So we'll add it to the header file.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = [PlayingCard rankStrings];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }

+ (NSArray *)rankStrings
{
    return @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
}

+ (NSUInteger)maxRank { return [[self rankStrings] count]-1; }

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;
+ (NSUInteger)maxRank;

@end
```

And, finally, let's use maxRank inside the setter for the rank @property to make sure the rank is never set to an improper value.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = [PlayingCard rankStrings];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }

+ (NSArray *)rankStrings
{
    return @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
}

+ (NSUInteger)maxRank { return [[self rankStrings] count]-1; }

- (void)setRank:(NSUInteger)rank
{
    if (rank <= [PlayingCard maxRank]) {
        _rank = rank;
    }
}

@end
```

Objective-C

PlayingCard.h

```
#import "Card.h"

@interface PlayingCard : Card

@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;

+ (NSArray *)validSuits;
+ (NSUInteger)maxRank;

@end
```

That's it for our PlayingCard. It's a good example of array notation, @synthesize, class methods, and using getters and setters for validation.

PlayingCard.m

```
#import "PlayingCard.h"

@implementation PlayingCard

- (NSString *)contents
{
    NSArray *rankStrings = [PlayingCard rankStrings];
    return [rankStrings[self.rank] stringByAppendingString:self.suit];
}

@synthesize suit = _suit; // because we provide setter AND getter
+ (NSArray *)validSuits { ... }
- (void)setSuit:(NSString *)suit { ... }
- (NSString *)suit { ... }

+ (NSArray *)rankStrings
{
    return @[@"?",@"A",@"2",@"3",...,@"10",@"J",@"Q",@"K"];
}

+ (NSUInteger)maxRank { return [[self rankStrings] count]-1; }

- (void)setRank:(NSUInteger)rank
{
    if (rank <= [PlayingCard maxRank]) {
        _rank = rank;
    }
}

@end
```

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

Let's look at one last class. This one is a subclass of Deck and represents a full 52-card deck of PlayingCards.

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"

@implementation PlayingCardDeck









@end
```

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

It appears to have no public API, but it is going to override a method that `Deck` inherits from `NSObject` called `init`.

`init` will contain everything necessary to initialize a `PlayingCardDeck`.

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"

@implementation PlayingCardDeck









@end
```

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{

}

@end
```

Initialization in Objective-C happens immediately after allocation.
We always nest a call to `init` around a call to `alloc`.
e.g. `Deck *myDeck = [[PlayingCardDeck alloc] init]`
or `NSMutableArray *cards = [[NSMutableArray alloc] init]`

Classes can have more complicated initializers than just plain "`init`"
(e.g. `initWithCapacity:` or some such).
We'll talk more about that next week as well.

Only call an `init` method immediately after calling `alloc` to make space in the heap for that new object.
And never call `alloc` without immediately calling some `init` method on the newly allocated object.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{




}

@end
```

Notice this weird "return type" of `instancetype`.
It basically tells the compiler that this method returns an object which will be the same type as the object that this message was sent to.
We will pretty much only use it for `init` methods.
Don't worry about it too much for now.
But always use this return type for your `init` methods.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {

    }

    return self;
}

@end
```

This sequence of code might also seem weird.
Especially an assignment to `self`!
This is the ONLY time you would ever assign something to `self`.
The idea here is to return `nil` if you cannot initialize this object.
But we have to check to see if our `super`class can `init`ialize itself.
The assignment to `self` is a bit of protection against our trying to continue to initialize ourselves if our `super`class couldn't initialize.
Just always do this and don't worry about it too much.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {









    }

    return self;
}

@end
```

Sending a message to `super` is how we send a message to ourselves, but use our superclass's implementation instead of our own.
Standard object-oriented stuff.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

The implementation of `init` is quite simple. We'll just iterate through all the `suits` and then through all the `ranks` in that `suit` ...

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {
        for (NSString *suit in [PlayingCard validSuits]) {
            for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {




            }
        }
    }

    return self;
}

@end
```

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"


@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {
        for (NSString *suit in [PlayingCard validSuits]) {
            for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {
                PlayingCard *card = [[PlayingCard alloc] init];
                card.rank = rank;
                card.suit = suit;

            }
        }
    }

    return self;
}

@end
```

Then we will allocate and initialize a PlayingCard and then set its suit and rank.

We never implemented an init method in PlayingCard, so it just inherits the one from NSObject. Even so, we must always call an init method after alloc.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"
#import "PlayingCard.h"

@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {
        for (NSString *suit in [PlayingCard validSuits]) {
            for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {
                PlayingCard *card = [[PlayingCard alloc] init];
                card.rank = rank;
                card.suit = suit;

            }
        }
    }

    return self;
}

@end
```

We will need to #import PlayingCard's header file since we are referencing it now in our implementation.

Then we will allocate and initialize a PlayingCard and then set its suit and rank.

We never implemented an init method in PlayingCard, so it just inherits the one from NSObject. Even so, we must always call an init method after alloc.

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"
#import "PlayingCard.h"

@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {
        for (NSString *suit in [PlayingCard validSuits]) {
            for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {
                PlayingCard *card = [[PlayingCard alloc] init];
                card.rank = rank;
                card.suit = suit;
                [self addCard:card];
            }
        }
    }

    return self;
}

@end
```

Finally we just add each PlayingCard we create to ourself (we are a Deck, remember).

Objective-C

PlayingCardDeck.h

```
#import "Deck.h"

@interface PlayingCardDeck : Deck

@end
```

PlayingCardDeck.m

```
#import "PlayingCardDeck.h"
#import "PlayingCard.h"

@implementation PlayingCardDeck

- (instancetype)init
{
    self = [super init];

    if (self) {
        for (NSString *suit in [PlayingCard validSuits]) {
            for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {
                PlayingCard *card = [[PlayingCard alloc] init];
                card.rank = rank;
                card.suit = suit;
                [self addCard:card];
            }
        }
    }

    return self;
}

@end
```

And that's it!
We inherit everything else we need to be a Deck of cards (like the ability to drawRandomCard) from our superclass.

Demo

- Let's start building a Card Game out of these classes
  Today we'll just have a single card that we can flip over to reveal the Ace of clubs.

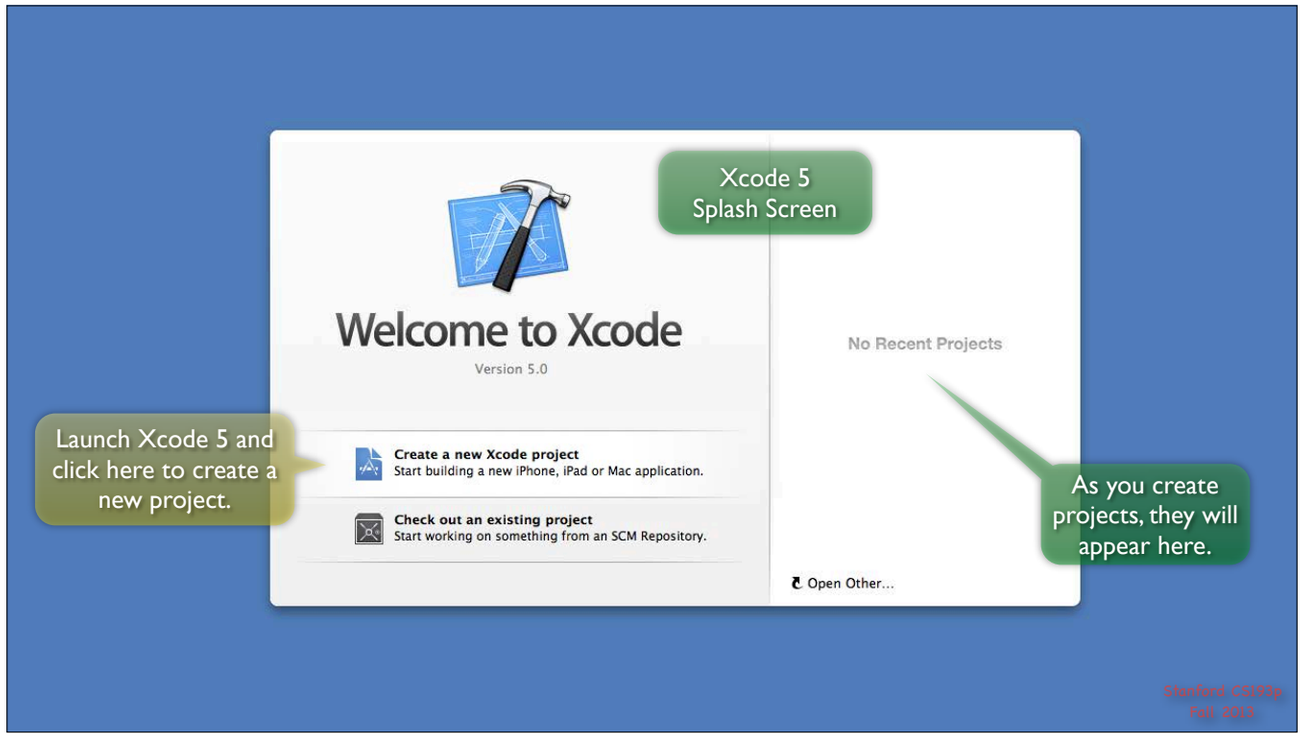
The following slides are a walkthrough of the demonstration done in class.
You will need this walkthrough to do your first homework assignment.

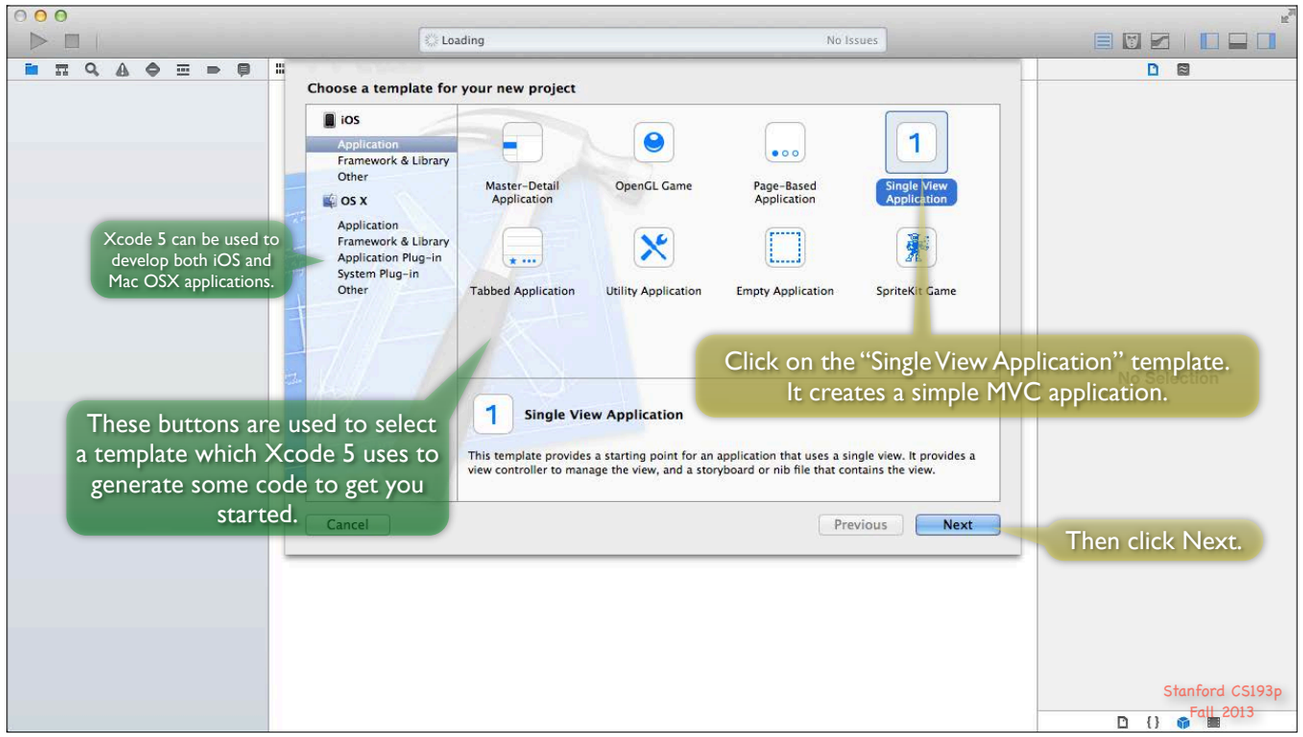
Yellow Bubbles
mean "do something."

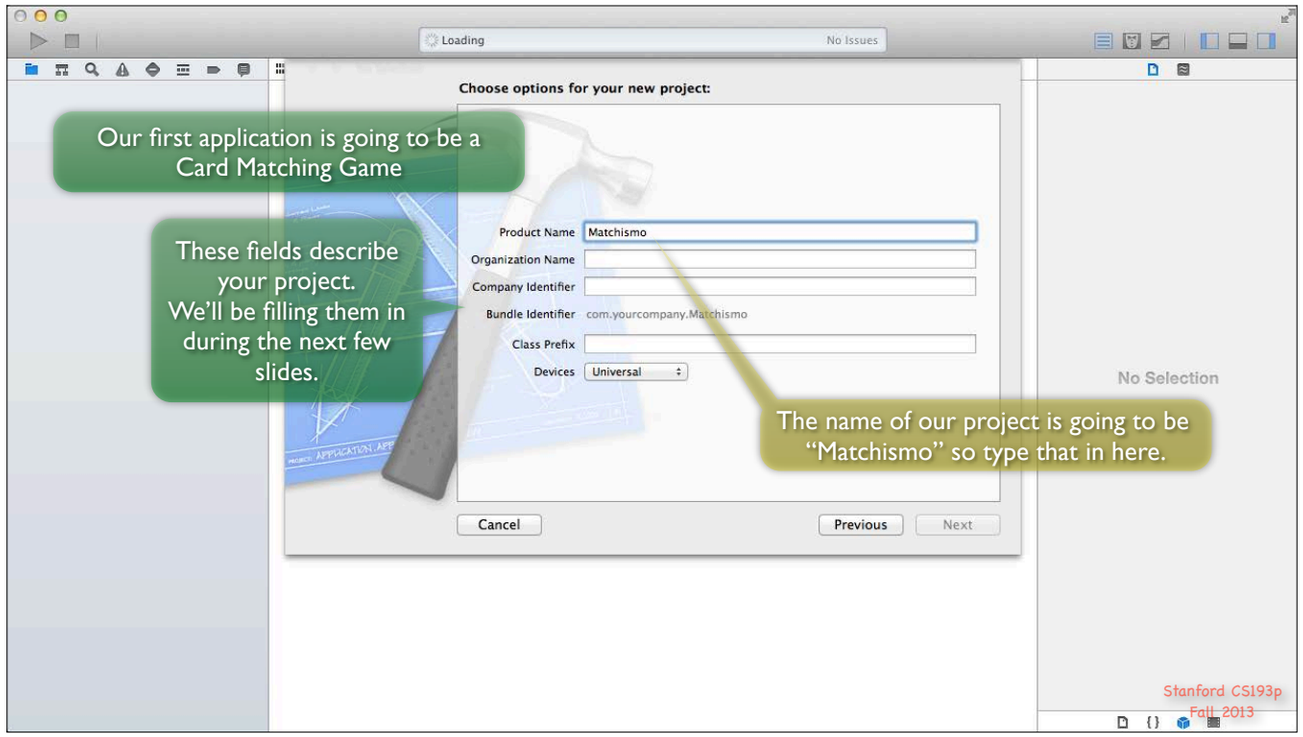
Green Bubbles
are just for
"information."

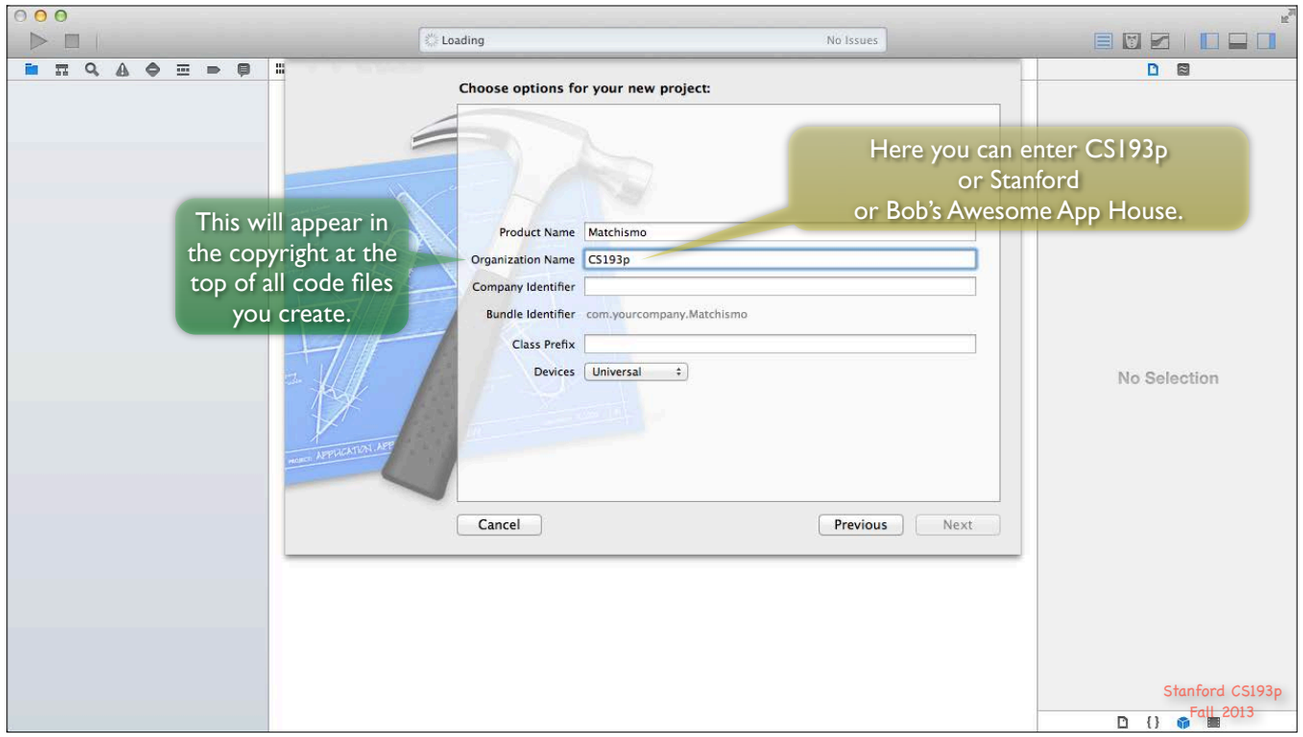
Red Bubbles
mean "important!"

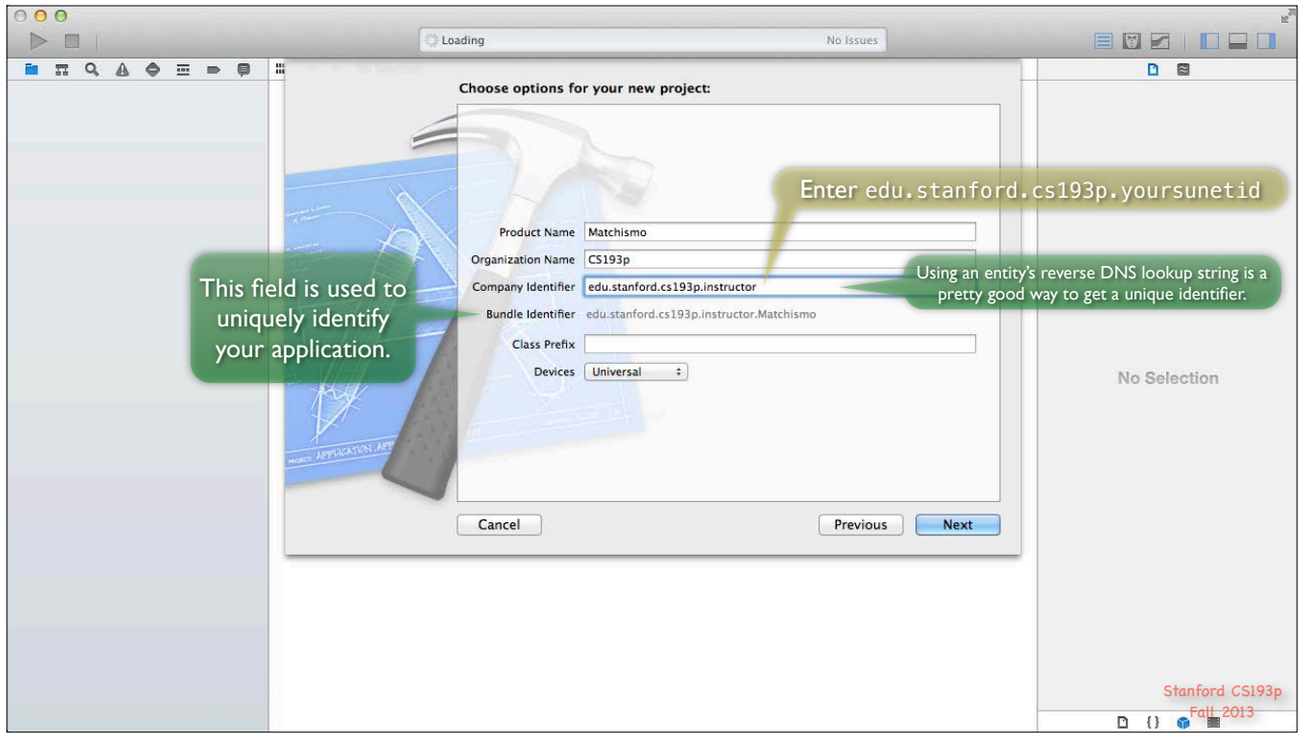
Green Bubbles
with small text is for
"minor notes."

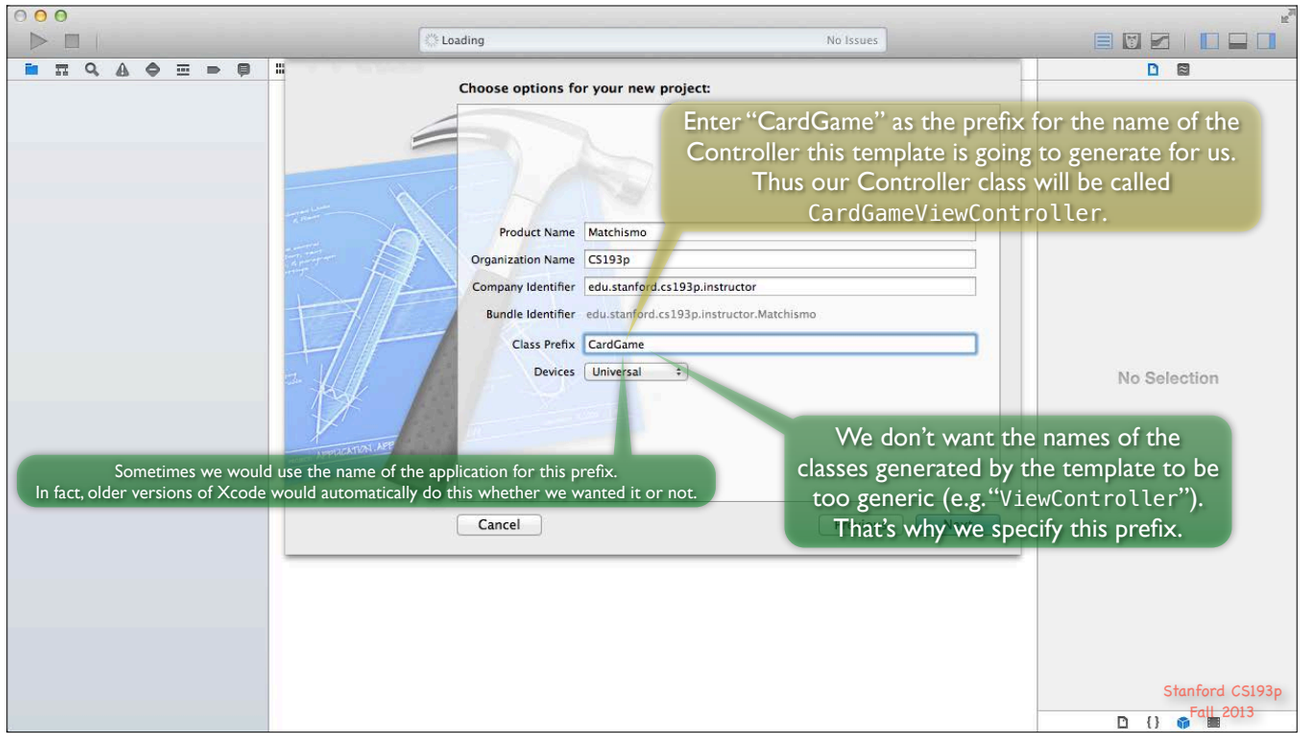

Xcode 5
Splash Screen
Welcome to Xcode
Version 5.0
Launch Xcode 5 and click here to create a new project.
Create a new Xcode project
Start building a new iPhone, iPad or Mac application.
Check out an existing project
Start working on something from an SCM Repository.
No Recent Projects
As you create projects, they will appear here.
Open Other…
Stanford CS193p
Fall 2013

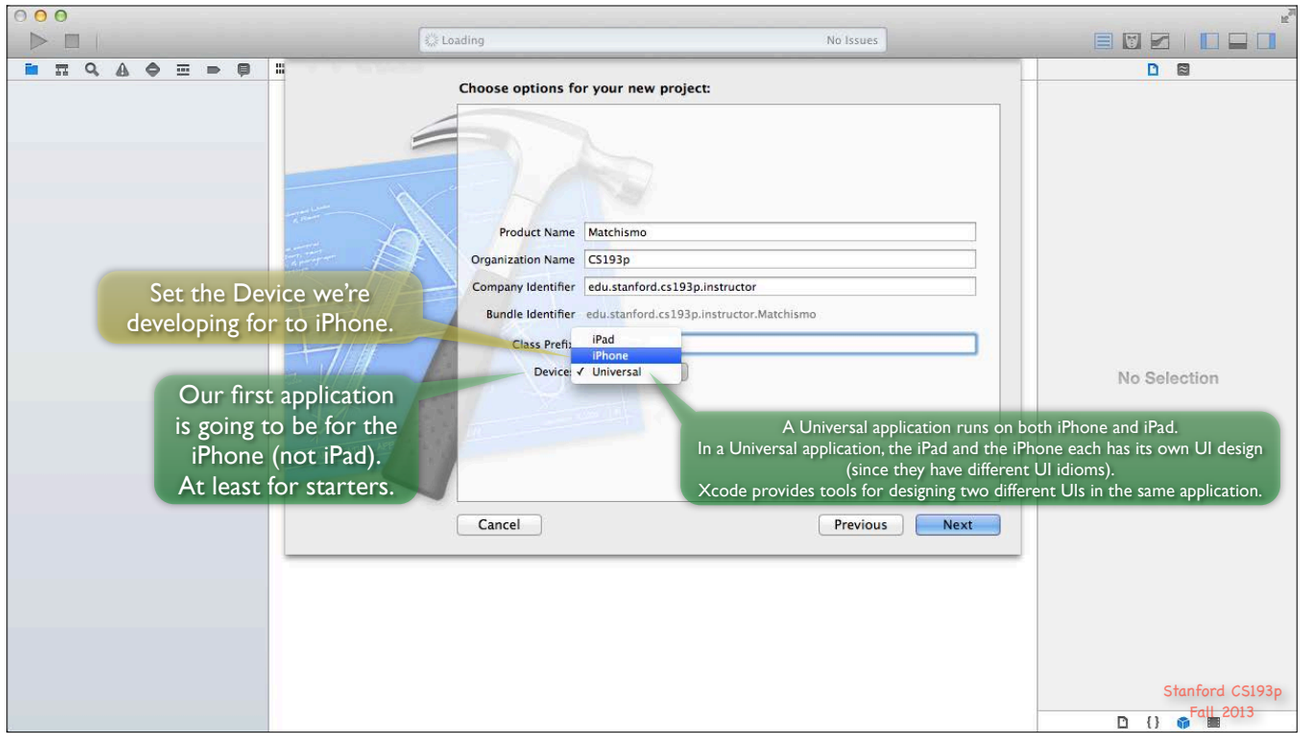

Loading
No Issues
Choose a template for your new project
iOS
Application
Framework & Library
Other
OS X
Application
Framework & Library
Application Plug-in
System Plug-in
Other
Master-Detail Application
OpenGL Game
Page-Based Application
Single View Application
Tabbed Application
Utility Application
Empty Application
SpriteKit Game
Single View Application
This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a storyboard or nib file that contains the view.
Cancel
Previous
Next
Xcode 5 can be used to develop both iOS and Mac OSX applications.
These buttons are used to select a template which Xcode 5 uses to generate some code to get you started.
Click on the "Single View Application" template. It creates a simple MVC application.
Then click Next.
Stanford CS193p
Fall 2013

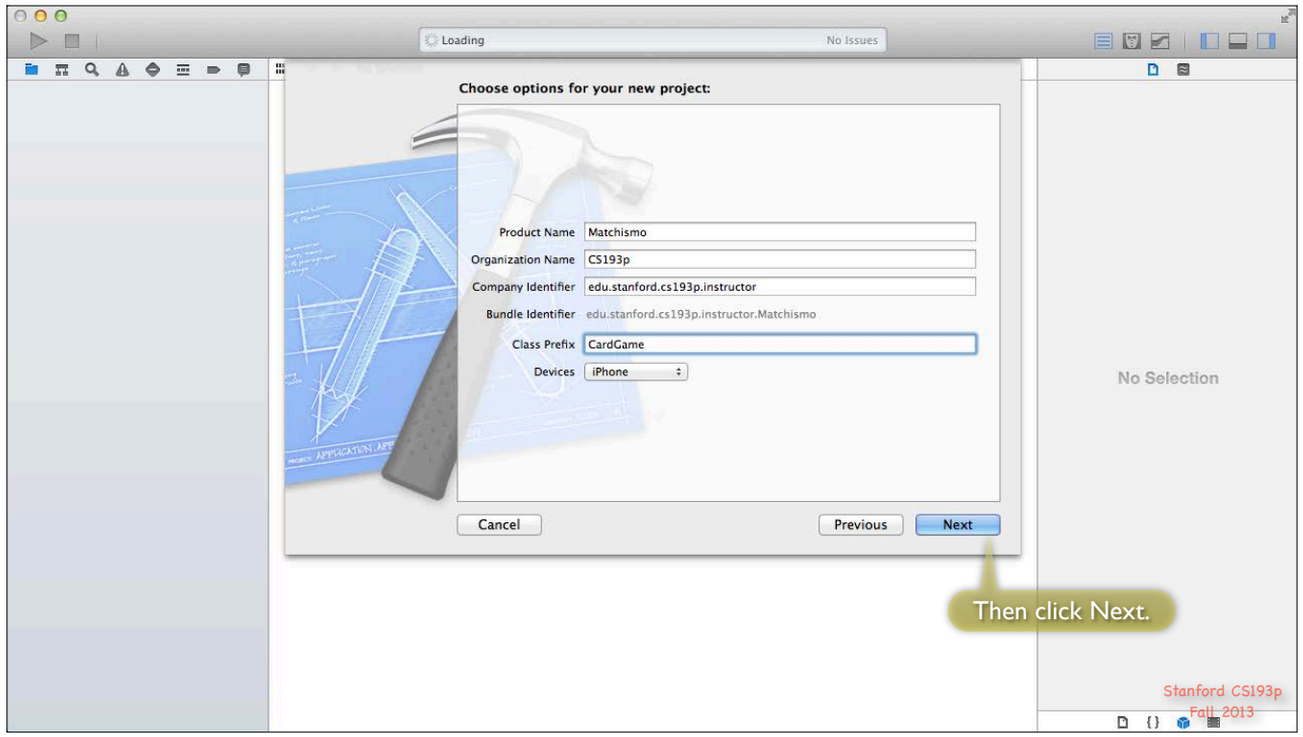

Loading
No Issues
Choose options for your new project:
Our first application is going to be a Card Matching Game
These fields describe your project. We'll be filling them in during the next few slides.
Product Name
Matchismo
Organization Name
Company Identifier
Bundle Identifier
com.yourcompany.Matchismo
Class Prefix
Devices
Universal
The name of our project is going to be "Matchismo" so type that in here.
Cancel
Previous
Next
No Selection
Stanford CS193p
Fall 2013

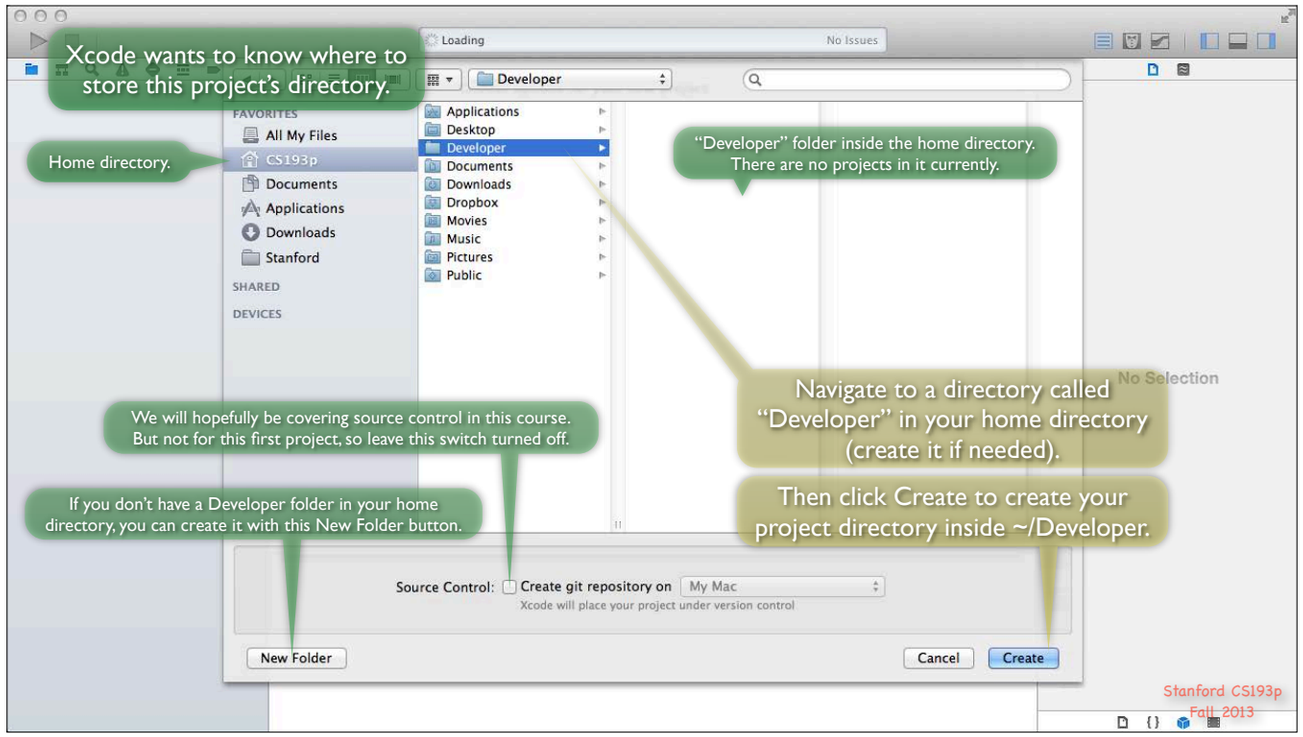

Loading
No Issues
Choose options for your new project:
This will appear in the copyright at the top of all code files you create.
Here you can enter CS193p or Stanford or Bob's Awesome App House.
Product Name
Matchismo
Organization Name
CS193p
Company Identifier
Bundle Identifier
com.yourcompany.Matchismo
Class Prefix
Devices
Universal
Cancel
Previous
Next
No Selection
Stanford CS193p
Fall 2013

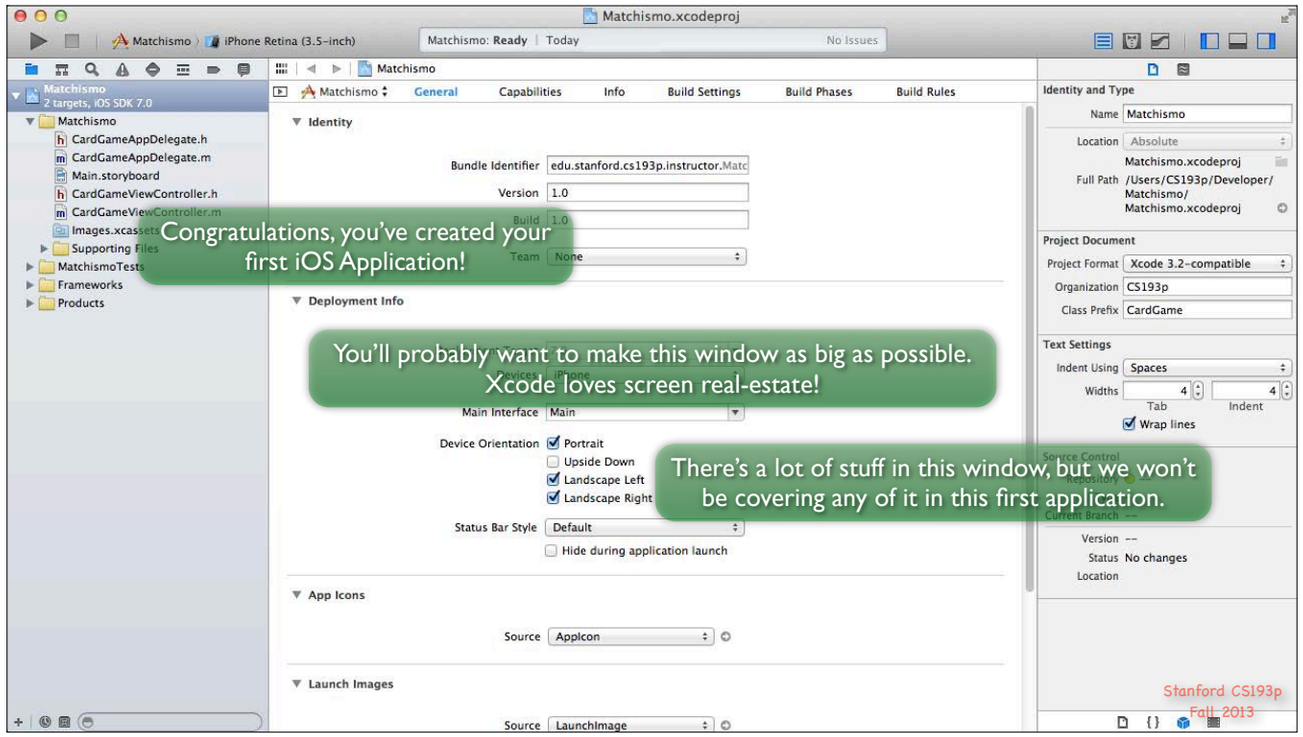

Loading
No Issues
Choose options for your new project:
Product Name
Matchismo
Organization Name
CS193p
Company Identifier
edu.stanford.cs193p.instructor
Bundle Identifier
edu.stanford.cs193p.instructor.Matchismo
Class Prefix
Devices
Universal
Cancel
Previous
Next
Enter edu.stanford.cs193p.yoursunetid
This field is used to uniquely identify your application.
Using an entity's reverse DNS lookup string is a pretty good way to get a unique identifier.
No Selection
Stanford CS193p
Fall 2013

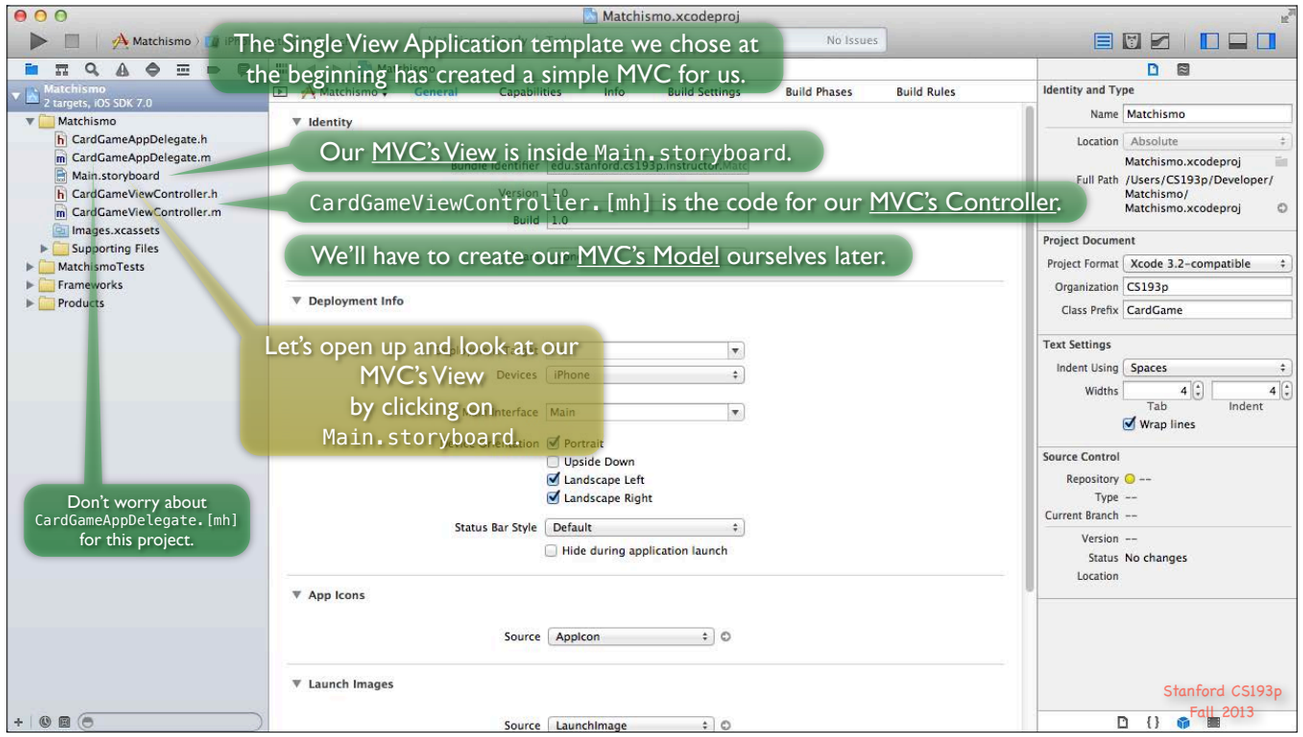

Loading
No Issues
Choose options for your new project:
Product Name
Matchismo
Organization Name
CS193p
Company Identifier
edu.stanford.cs193p.instructor
Bundle Identifier
edu.stanford.cs193p.instructor.Matchismo
Class Prefix
CardGame
Devices
Universal
Cancel
No Selection
Enter "CardGame" as the prefix for the name of the Controller this template is going to generate for us. Thus our Controller class will be called CardGameViewController.
Sometimes we would use the name of the application for this prefix.
In fact, older versions of Xcode would automatically do this whether we wanted it or not.
We don't want the names of the classes generated by the template to be too generic (e.g. "ViewController"). That's why we specify this prefix.
Stanford CS193p
Fall 2013

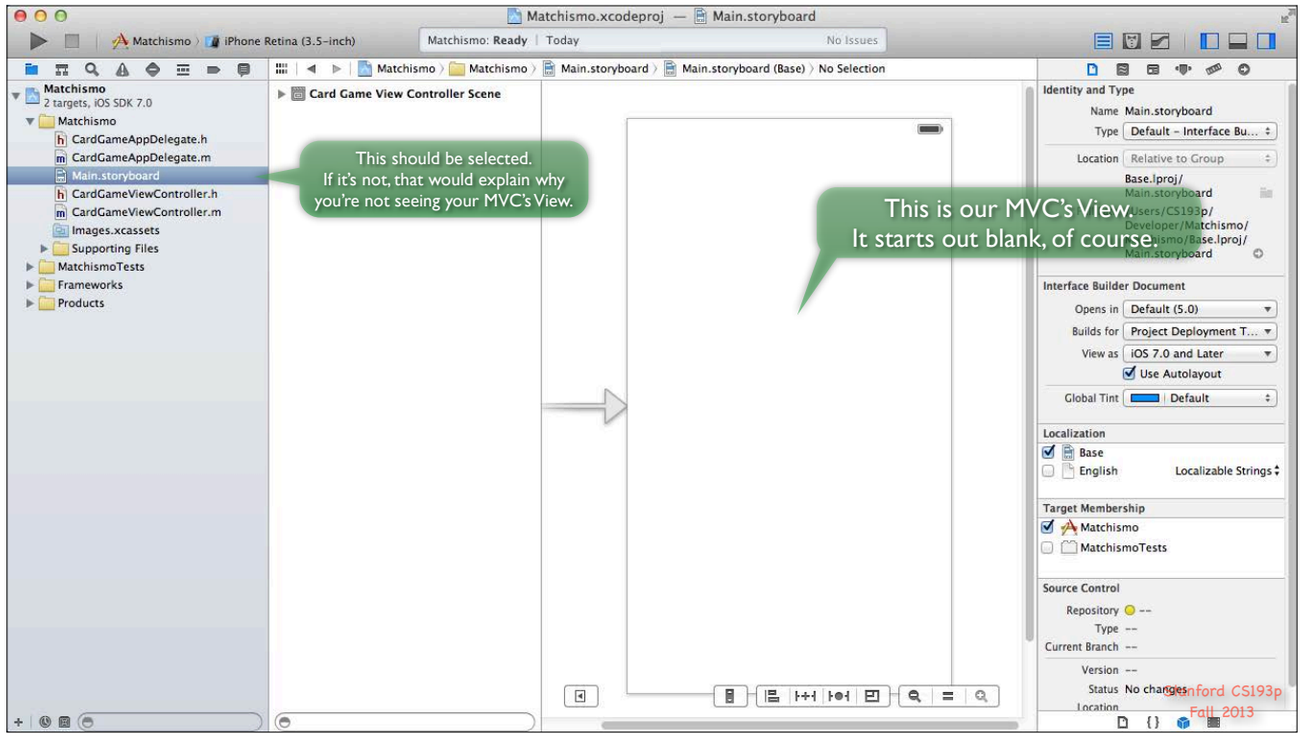

Loading
No Issues
Choose options for your new project:
Product Name
Matchismo
Organization Name
CS193p
Company Identifier
edu.stanford.cs193p.instructor
Bundle Identifier
edu.stanford.cs193p.instructor.Matchismo
iPad
iPhone
Universal
Cancel
Previous
Next
No Selection
Set the Device we're developing for to iPhone.
Our first application is going to be for the iPhone (not iPad). At least for starters.
A Universal application runs on both iPhone and iPad.
In a Universal application, the iPad and the iPhone each has its own UI design (since they have different UI idioms).
Xcode provides tools for designing two different UIs in the same application.
Stanford CS193p
Fall 2013

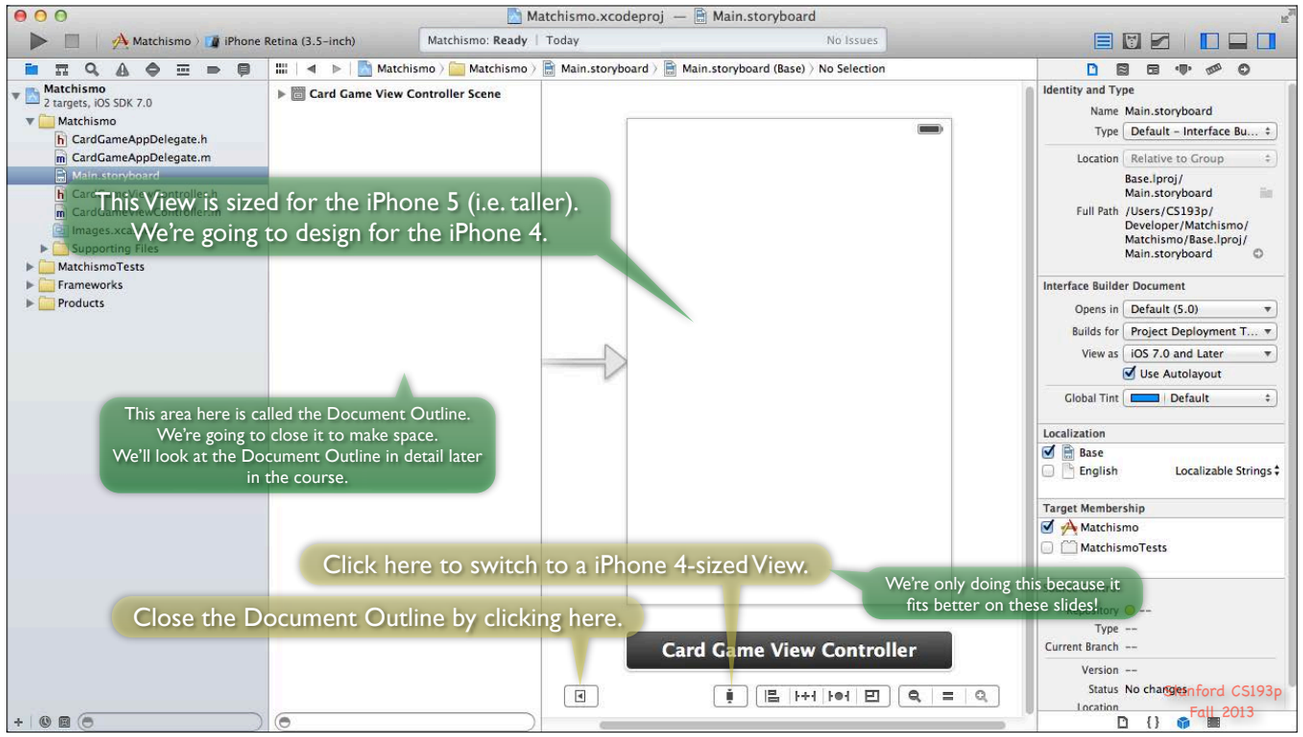

Loading
No Issues
Choose options for your new project:
Product Name
Matchismo
Organization Name
CS193p
Company Identifier
edu.stanford.cs193p.instructor
Bundle Identifier
edu.stanford.cs193p.instructor.Matchismo
Class Prefix
CardGame
Devices
iPhone
Cancel
Previous
Next
Then click Next.
No Selection
Stanford CS193p
Fall 2013

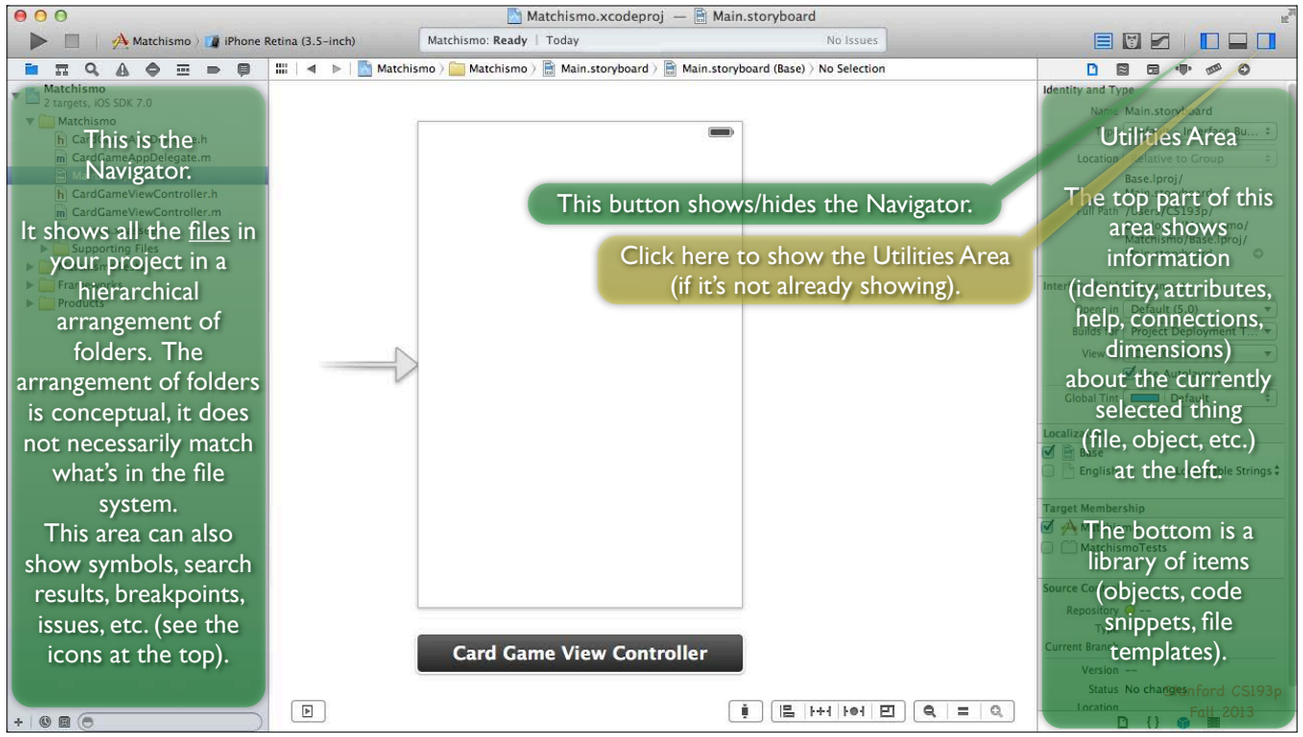

Xcode wants to know where to store this project's directory.
Loading
No Issues
Developer
FAVORITES
All My Files
CS193p
Documents
Applications
Downloads
Stanford
SHARED
DEVICES
Applications
Desktop
Developer
Documents
Downloads
Dropbox
Movies
Music
Pictures
Public
Home directory.
"Developer" folder inside the home directory. There are no projects in it currently.
No Selection
Navigate to a directory called "Developer" in your home directory (create it if needed).
We will hopefully be covering source control in this course. But not for this first project, so leave this switch turned off.
If you don't have a Developer folder in your home directory, you can create it with this New Folder button.
Then click Create to create your project directory inside ~/Developer.
Source Control:
Create git repository on
My Mac
Xcode will place your project under version control
New Folder
Cancel
Create
Stanford CS193p
Fall 2013

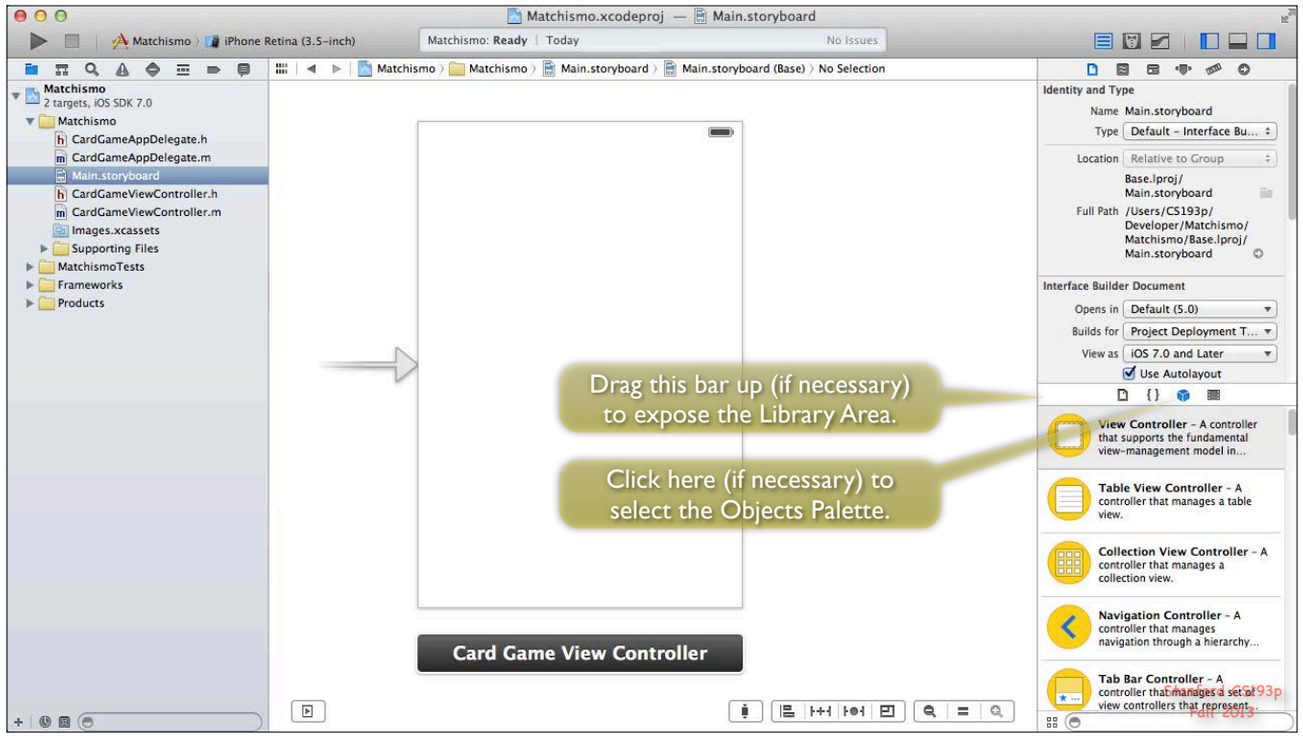

Matchismo.xcodeproj
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
General
Capabilities
Info
Build Settings
Build Phases
Build Rules
Identity
Bundle Identifier
edu.stanford.cs193p.instructor.Matc
Version
1.0
Build
1.0
Team
None
Deployment Info
Main Interface
Main
Device Orientation
Portrait
Upside Down
Landscape Left
Landscape Right
Status Bar Style
Default
Hide during application launch
App Icons
Source
AppIcon
Launch Images
Source
LaunchImage
Congratulations, you've created your first iOS Application!
You'll probably want to make this window as big as possible. Xcode loves screen real-estate!
There's a lot of stuff in this window, but we won't be covering any of it in this first application.
Identity and Type
Name
Matchismo
Location
Absolute
Matchismo.xcodeproj
Full Path
/Users/CS193p/Developer/Matchismo/Matchismo.xcodeproj
Project Document
Project Format
Xcode 3.2-compatible
Organization
CS193p
Class Prefix
CardGame
Text Settings
Indent Using
Spaces
Widths
4
4
Tab
Indent
Wrap lines
Source Control
Current Branch
--
Version
--
Status
No changes
Location
Stanford CS193p
Fall 2013

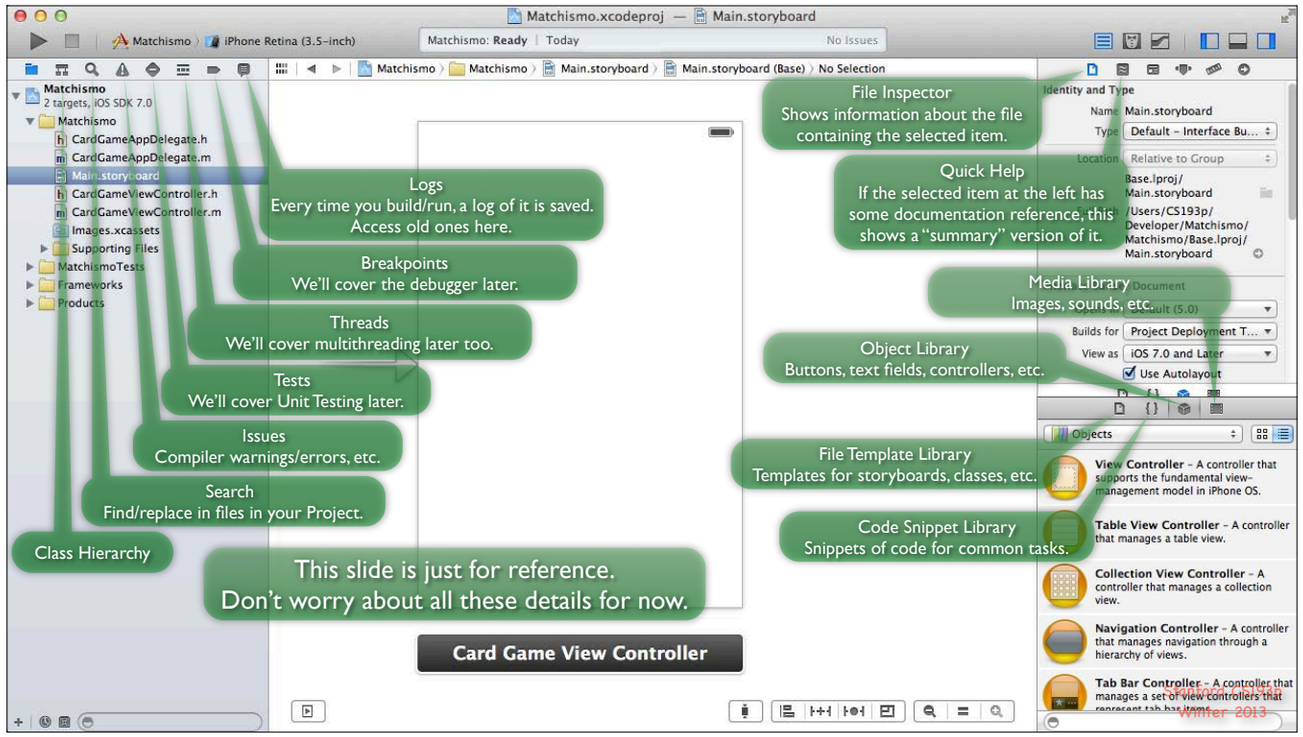

The Single View Application template we chose at the beginning has created a simple MVC for us.
Our MVC's View is inside Main.storyboard.
CardGameViewController.[mh] is the code for our MVC's Controller.
We'll have to create our MVC's Model ourselves later.
Let's open up and look at our MVC's View by clicking on Main.storyboard.
Don't worry about CardGameAppDelegate.[mh] for this project.
Matchismo.xcodeproj
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Identity
Deployment Info
App Icons
Launch Images
Identity and Type
Project Document
Text Settings
Source Control
Stanford CS193p
Fall 2013

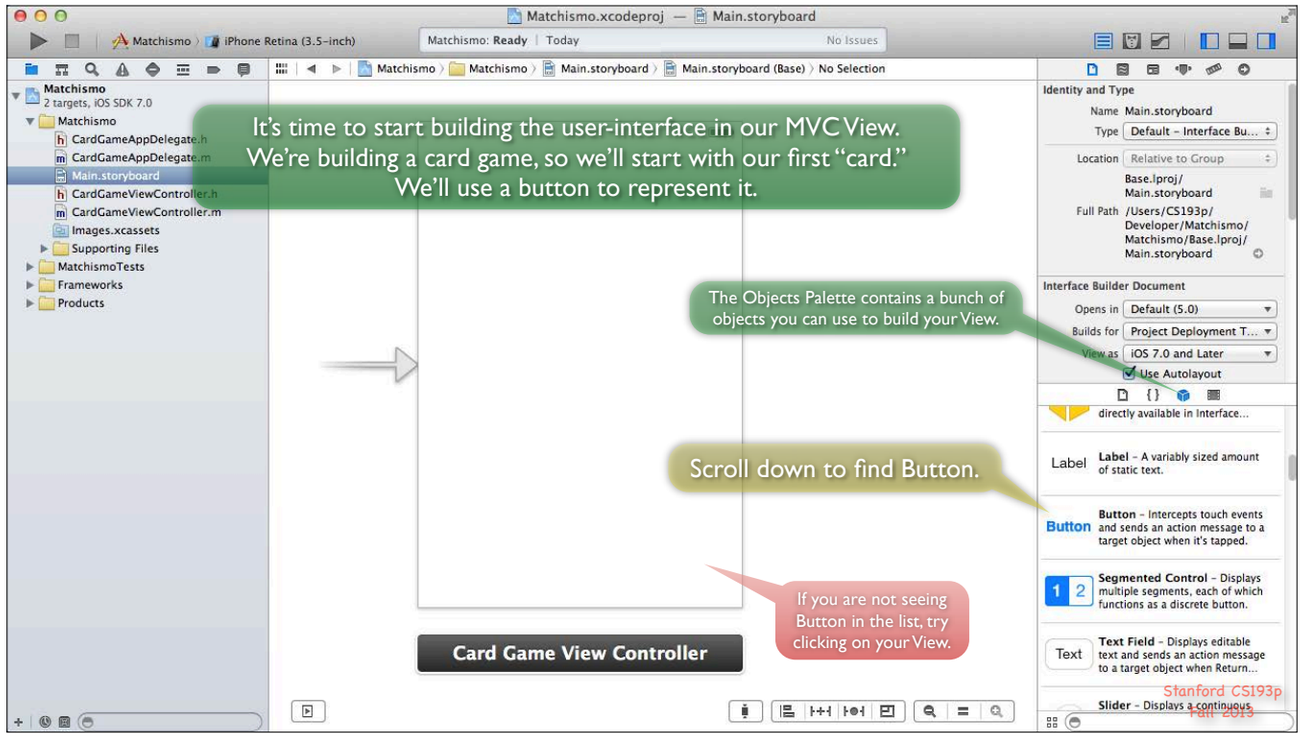

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo 2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller Scene
This should be selected.
If it's not, that would explain why you're not seeing your MVC's View.
This is our MVC's View.
It starts out blank, of course.
Identity and Type
Name Main.storyboard
Type Default - Interface Bu...
Location Relative to Group
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Global Tint Default
Localization
Base
English
Localizable Strings
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Stanford CS193p
Fall 2013

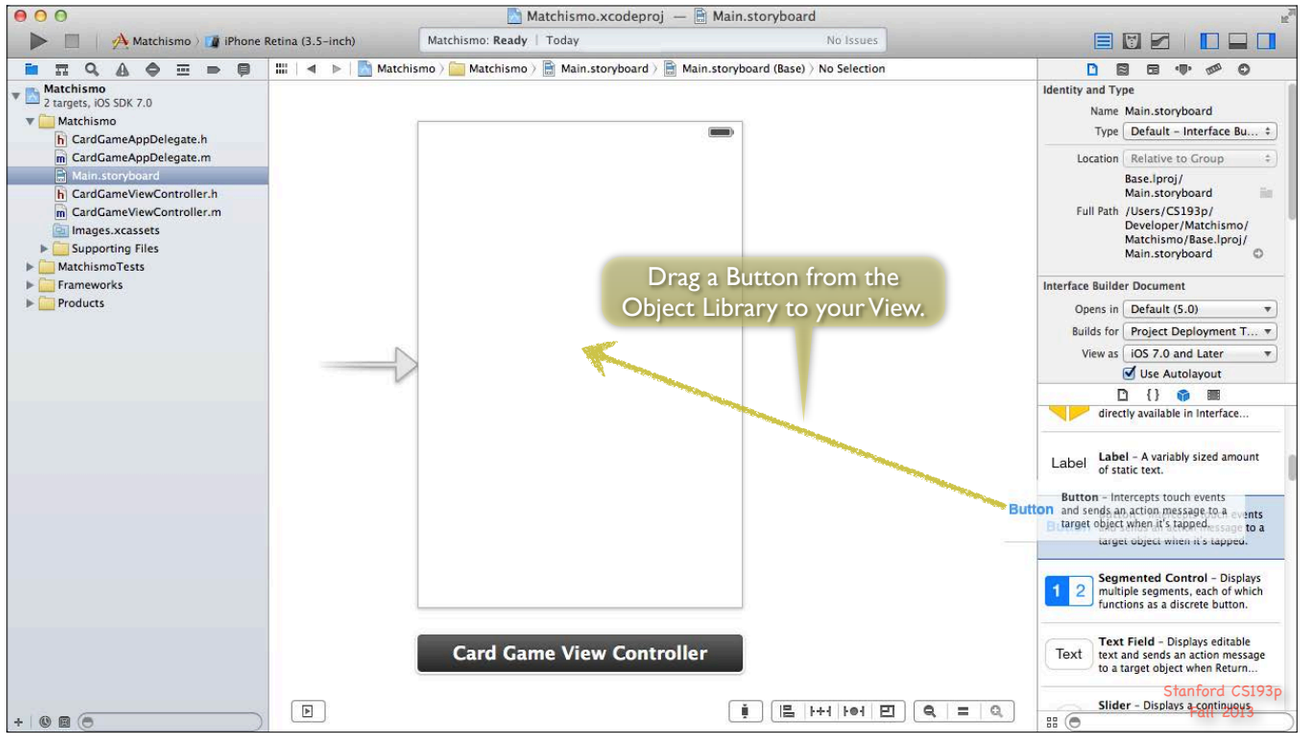

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller Scene
Identity and Type
Name Main.storyboard
Type Default - Interface Bu...
Location Relative to Group
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Base.lproj/Main.storyboard
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Global Tint Default
Localization
Base
English
Localizable Strings
Target Membership
Matchismo
MatchismoTests
Type --
Current Branch --
Version --
Status No changes
This View is sized for the iPhone 5 (i.e. taller).
We're going to design for the iPhone 4.
This area here is called the Document Outline.
We're going to close it to make space.
We'll look at the Document Outline in detail later in the course.
Click here to switch to a iPhone 4-sized View.
We're only doing this because it fits better on these slides!
Close the Document Outline by clicking here.
Card Game View Controller
Stanford CS193p
Fall 2013

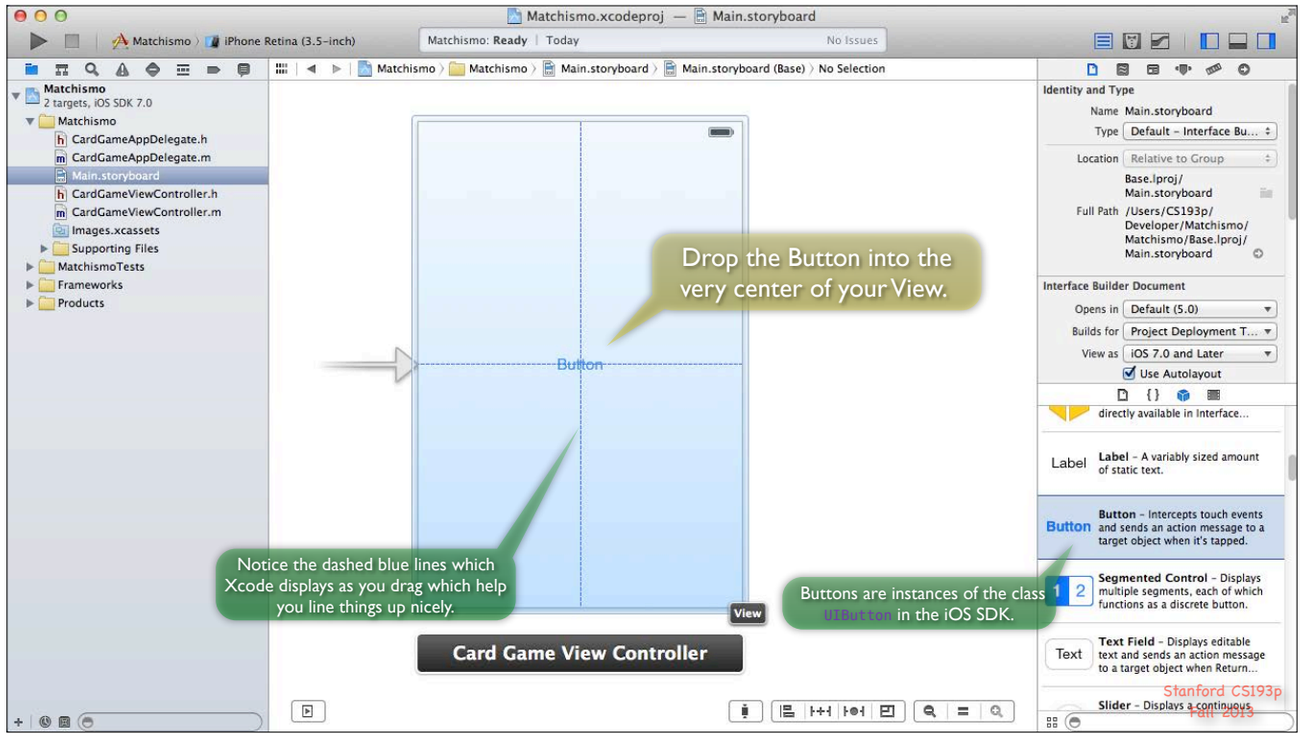

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
This is the Navigator.
It shows all the files in your project in a hierarchical arrangement of folders. The arrangement of folders is conceptual, it does not necessarily match what's in the file system.
This area can also show symbols, search results, breakpoints, issues, etc. (see the icons at the top).
This button shows/hides the Navigator.
Click here to show the Utilities Area (if it's not already showing).
Utilities Area
The top part of this area shows information (identity, attributes, help, connections, dimensions) about the currently selected thing (file, object, etc.) at the left.
The bottom is a library of items (objects, code snippets, file templates).
Card Game View Controller
Stanford CS193p
Fall 2013

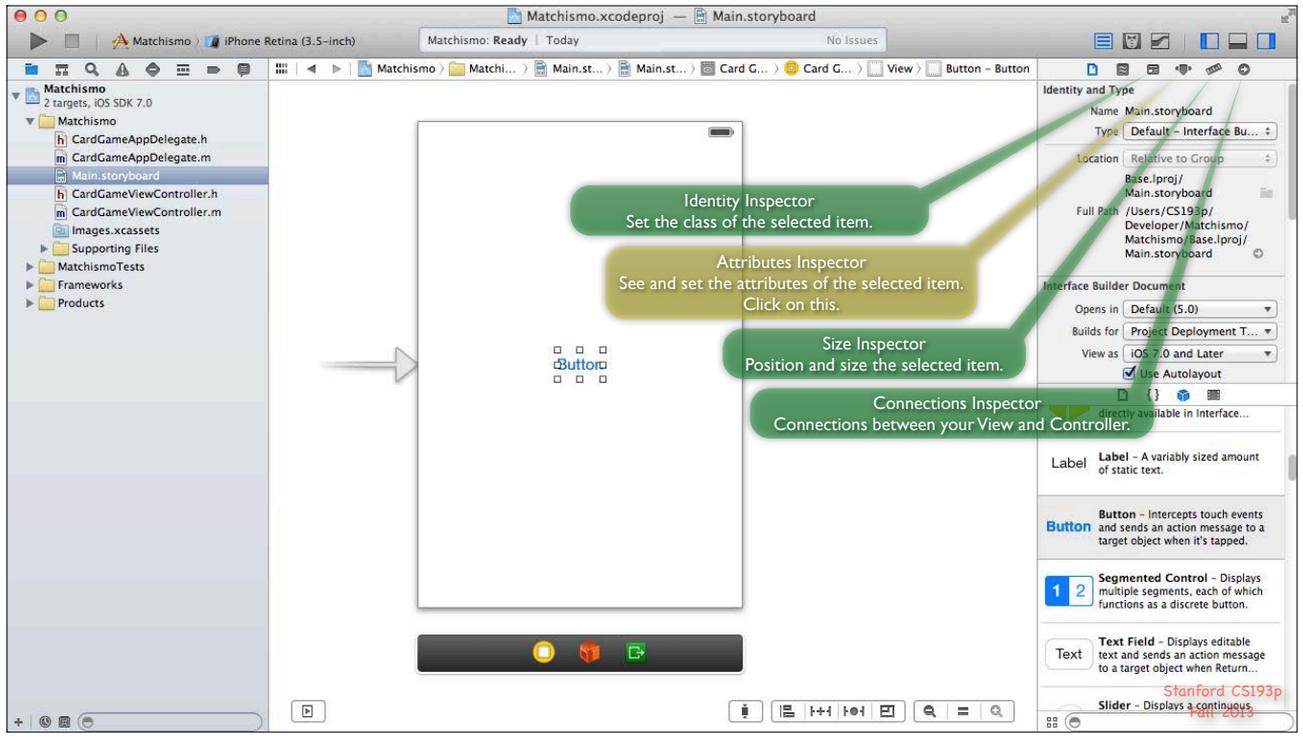

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Main.storyboard (Base)
No Selection
Identity and Type
Name Main.storyboard
Type Default - Interface Bu...
Location Relative to Group
Base.lproj/
Main.storyboard
Full Path /Users/CS193p/
Developer/Matchismo/
Matchismo/Base.lproj/
Main.storyboard
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Drag this bar up (if necessary) to expose the Library Area.
Click here (if necessary) to select the Objects Palette.
Card Game View Controller
View Controller - A controller that supports the fundamental view-management model in...
Table View Controller - A controller that manages a table view.
Collection View Controller - A controller that manages a collection view.
Navigation Controller - A controller that manages navigation through a hierarchy...
Tab Bar Controller - A controller that manages a set of view controllers that represent...

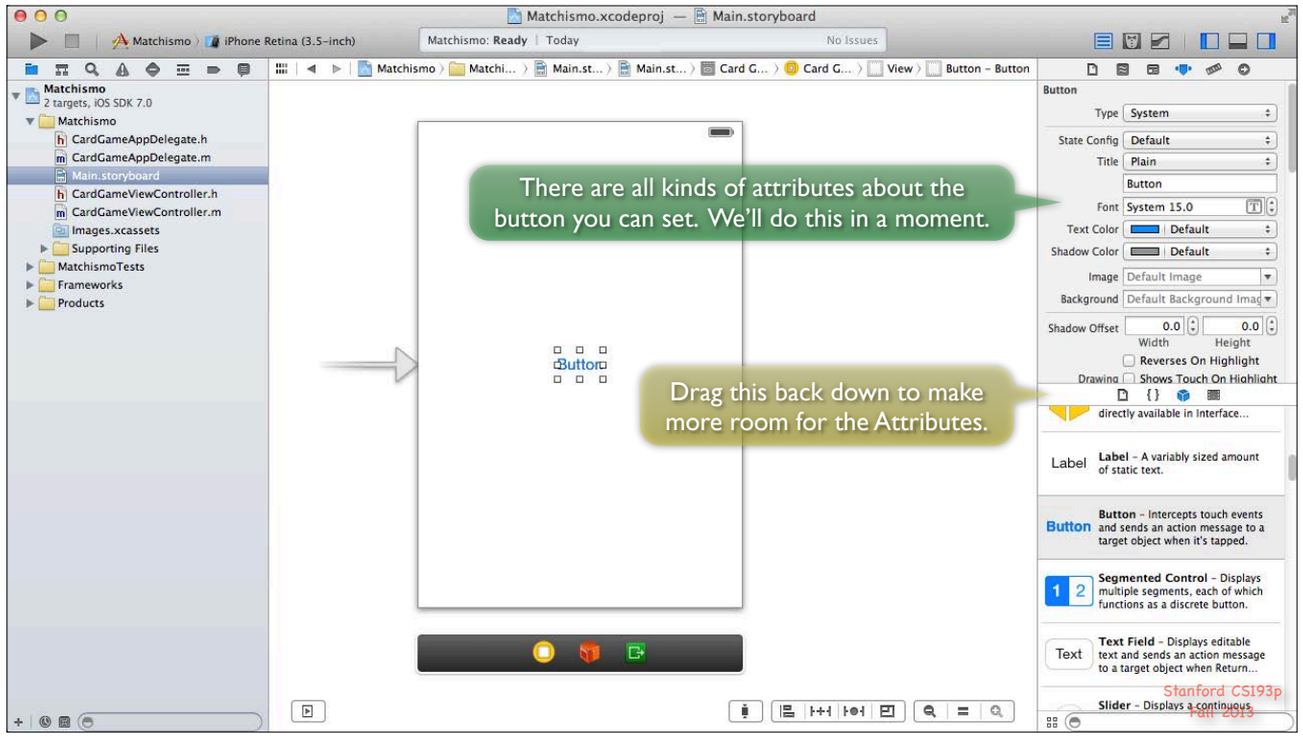

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Logs
Every time you build/run, a log of it is saved.
Access old ones here.
Breakpoints
We'll cover the debugger later.
Threads
We'll cover multithreading later too.
Tests
We'll cover Unit Testing later.
Issues
Compiler warnings/errors, etc.
Search
Find/replace in files in your Project.
Class Hierarchy
This slide is just for reference.
Don't worry about all these details for now.
Card Game View Controller
File Inspector
Shows information about the file containing the selected item.
Quick Help
If the selected item at the left has some documentation reference, this shows a "summary" version of it.
Media Library
Images, sounds, etc.
Object Library
Buttons, text fields, controllers, etc.
File Template Library
Templates for storyboards, classes, etc.
Code Snippet Library
Snippets of code for common tasks.
Identity and Type
Name Main.storyboard
Type Default – Interface Bu...
Location Relative to Group
Base.lproj/
Main.storyboard
/Users/CS193p/
Developer/Matchismo/
Matchismo/Base.lproj/
Main.storyboard
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Objects
View Controller – A controller that supports the fundamental view-management model in iPhone OS.
Table View Controller – A controller that manages a table view.
Collection View Controller – A controller that manages a collection view.
Navigation Controller – A controller that manages navigation through a hierarchy of views.
Tab Bar Controller – A controller that manages a set of view controllers that represent tab bar items.

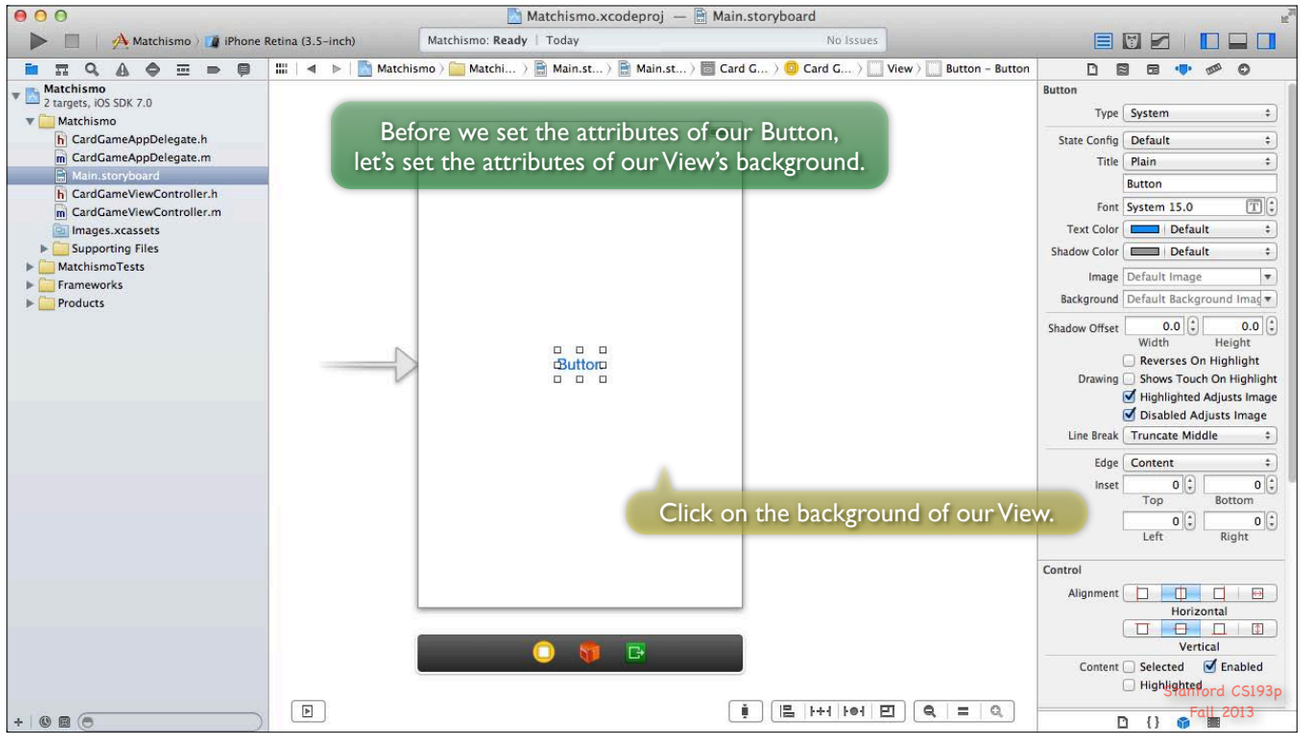

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo 〉 Matchismo 〉 Main.storyboard 〉 Main.storyboard (Base) 〉 No Selection
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
It's time to start building the user-interface in our MVC View.
We're building a card game, so we'll start with our first "card."
We'll use a button to represent it.
The Objects Palette contains a bunch of objects you can use to build your View.
Scroll down to find Button.
If you are not seeing Button in the list, try clicking on your View.
Card Game View Controller
Identity and Type
Name Main.storyboard
Type Default – Interface Bu...
Location Relative to Group
Base.lproj/
Main.storyboard
Full Path /Users/CS193p/
Developer/Matchismo/
Matchismo/Base.lproj/
Main.storyboard
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
directly available in Interface...
Label Label – A variably sized amount of static text.
Button Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

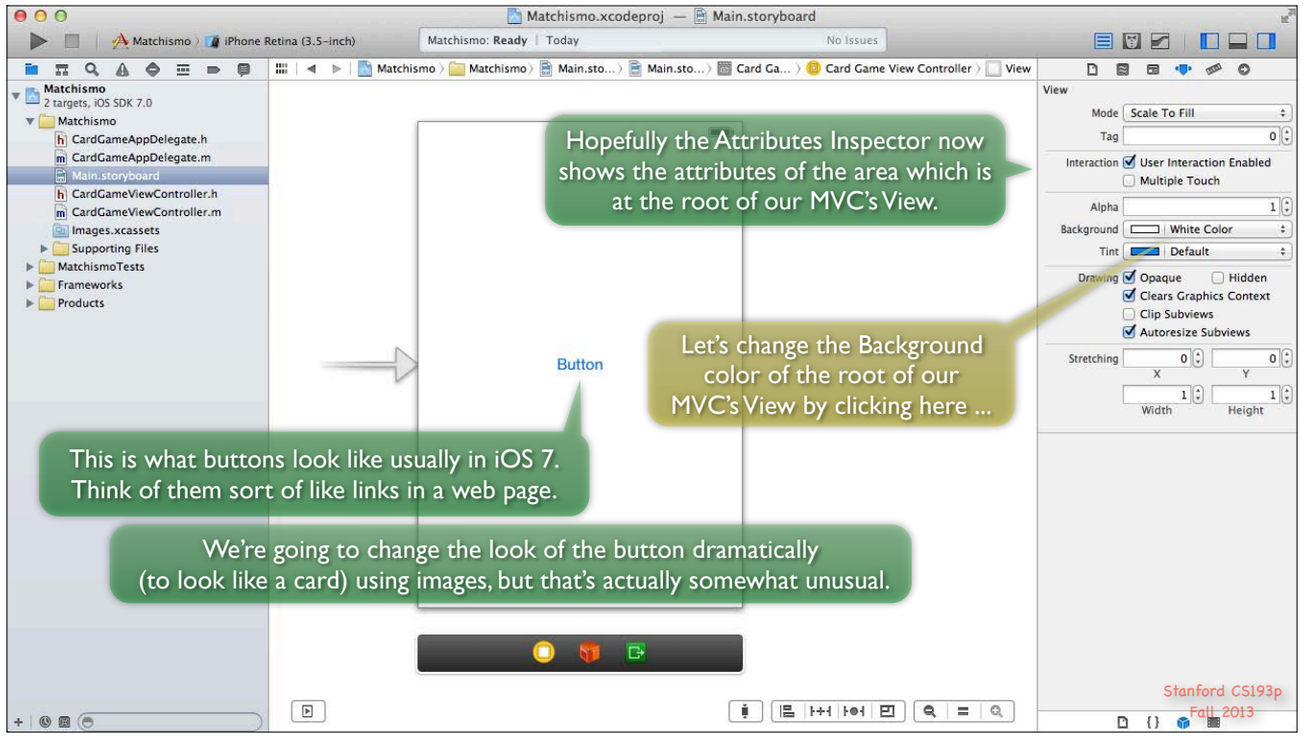

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Main.storyboard (Base)
No Selection
Drag a Button from the Object Library to your View.
Card Game View Controller
Identity and Type
Name Main.storyboard
Type Default – Interface Bu...
Location Relative to Group
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Base.lproj/Main.storyboard
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Label
Label – A variably sized amount of static text.
Button
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text
Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

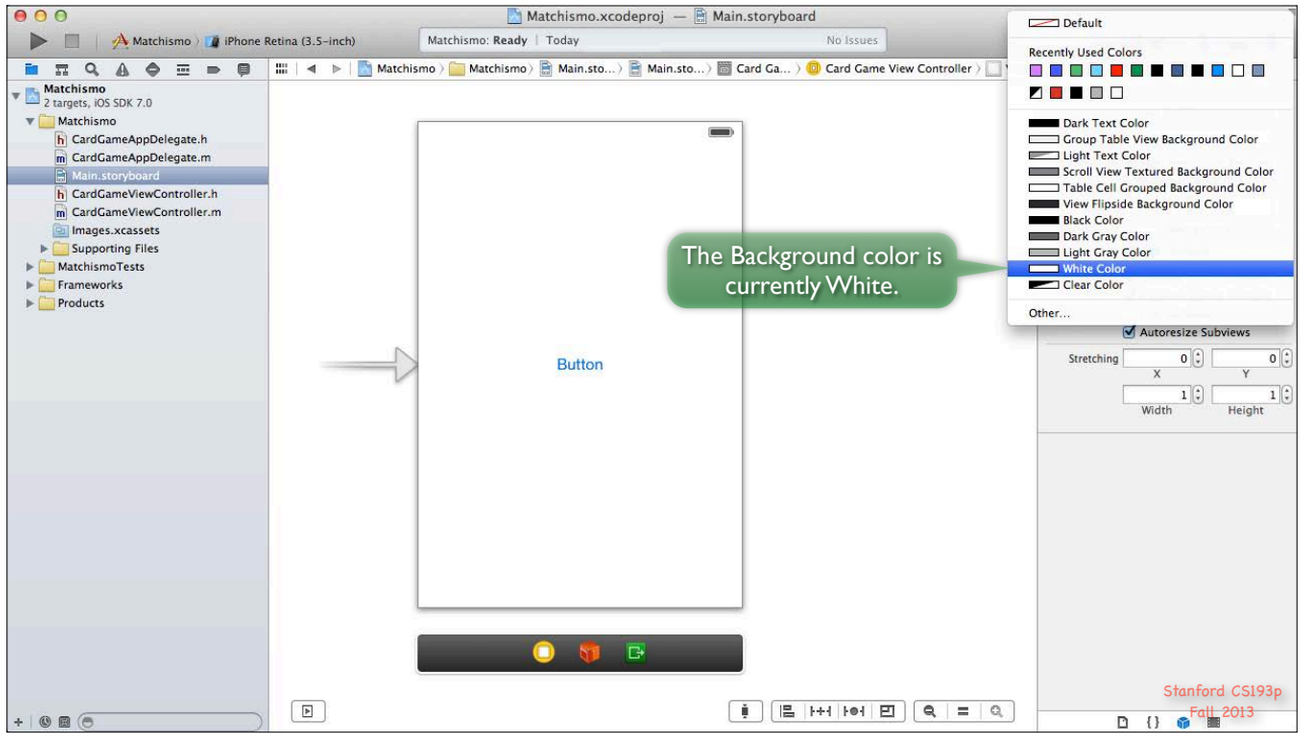

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Identity and Type
Name Main.storyboard
Type Default – Interface Bu...
Location Relative to Group
Base.lproj/
Main.storyboard
Full Path /Users/CS193p/
Developer/Matchismo/
Matchismo/Base.lproj/
Main.storyboard
Interface Builder Document
Opens in Default (5.0)
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Drop the Button into the very center of your View.
Button
Notice the dashed blue lines which Xcode displays as you drag which help you line things up nicely.
Buttons are instances of the class UIButton in the iOS SDK.
View
Card Game View Controller
Label – A variably sized amount of static text.
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

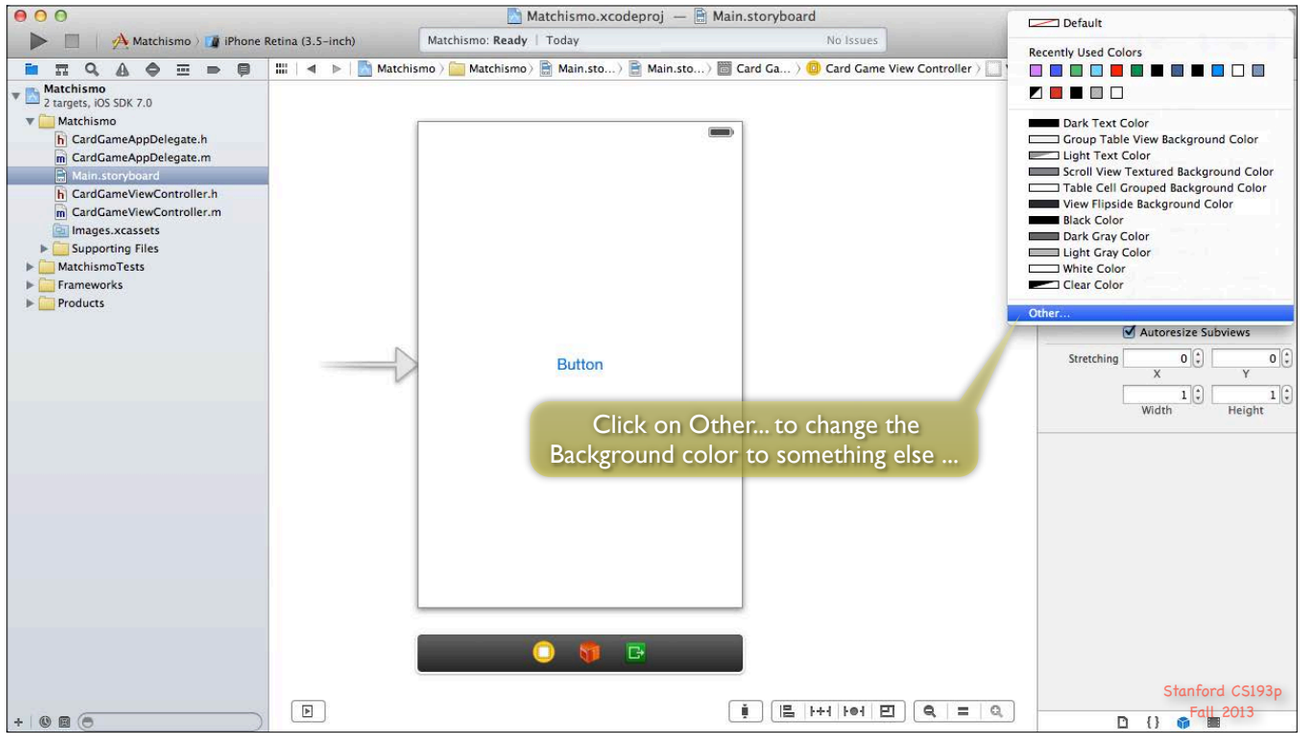

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Identity Inspector
Set the class of the selected item.
Attributes Inspector
See and set the attributes of the selected item.
Click on this.
Size Inspector
Position and size the selected item.
Connections Inspector
Connections between your View and Controller.
Button
Stanford CS193p

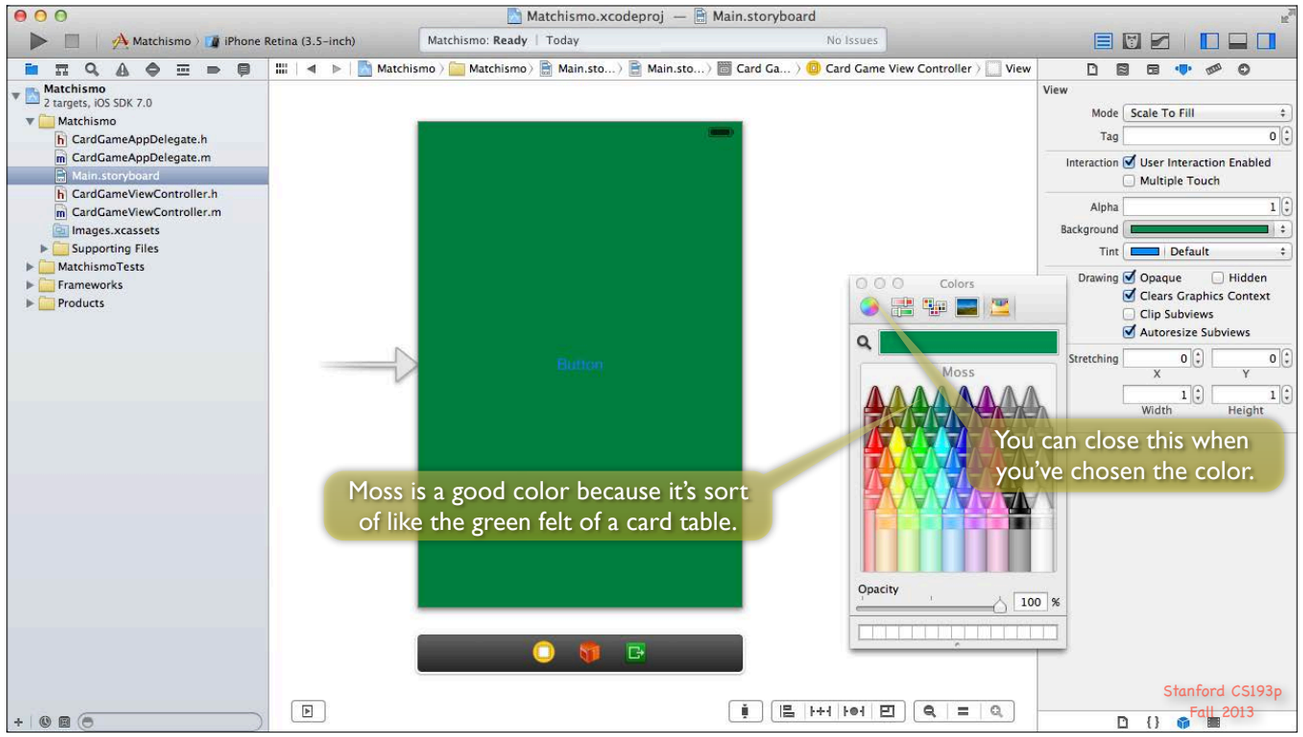

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button – Button
There are all kinds of attributes about the button you can set. We'll do this in a moment.
Drag this back down to make more room for the Attributes.
Button
Type System
State Config Default
Title Plain
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background Default Background Imag
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Shows Touch On Highlight
directly available in Interface...
Label – A variably sized amount of static text.
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

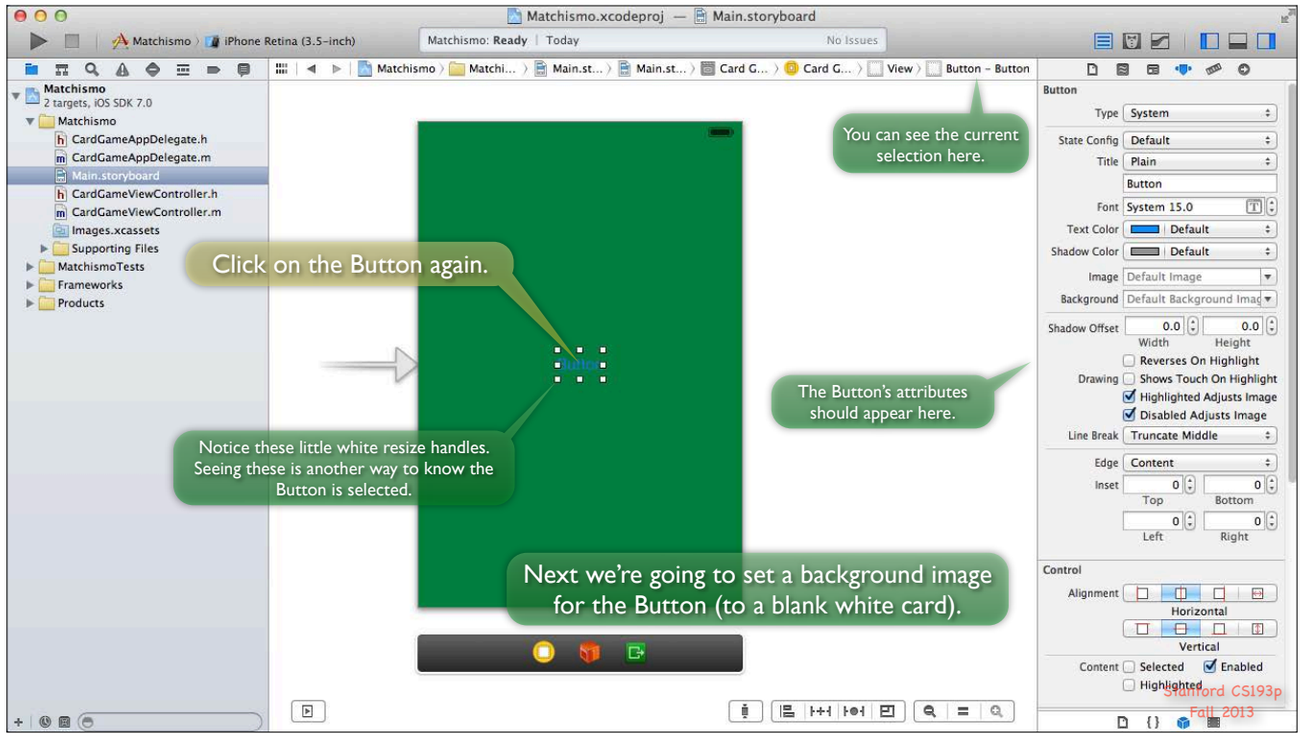

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Before we set the attributes of our Button, let's set the attributes of our View's background.
Button
Click on the background of our View.
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background Default Background Imag
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

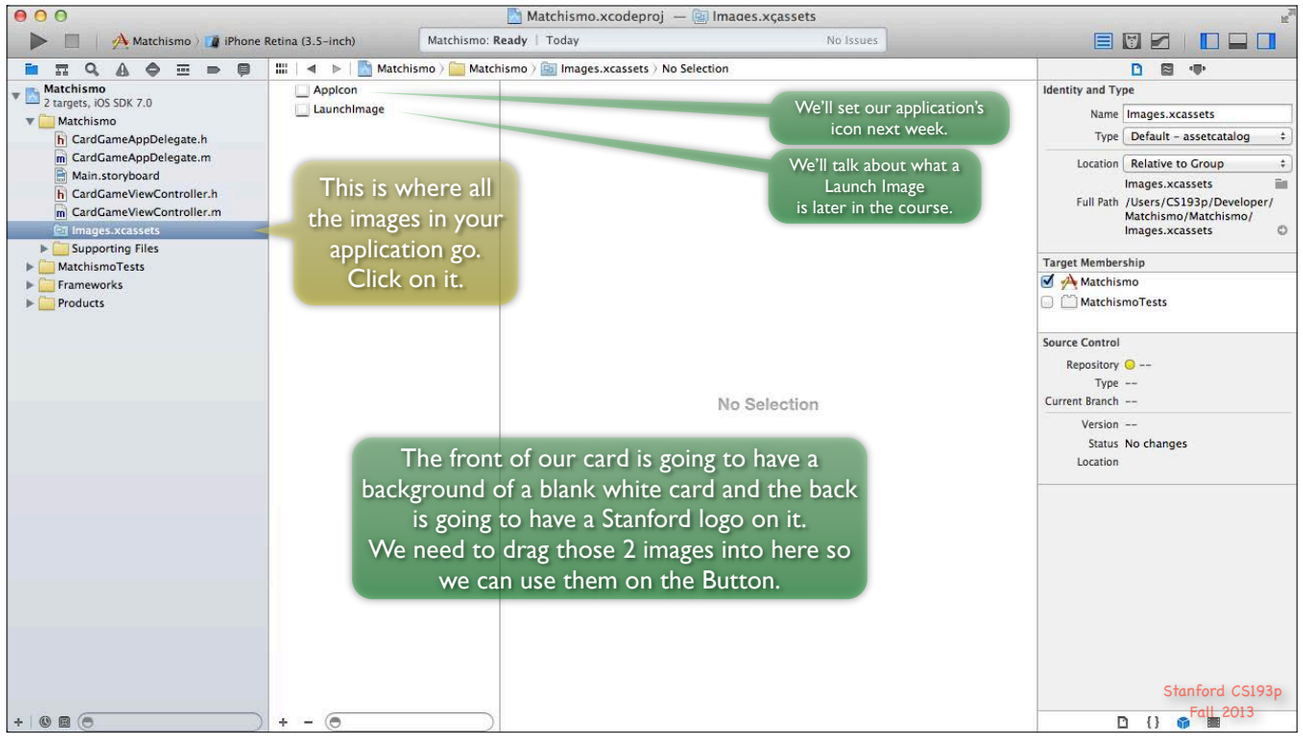

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller
View
Hopefully the Attributes Inspector now shows the attributes of the area which is at the root of our MVC's View.
Button
Let's change the Background color of the root of our MVC's View by clicking here ...
This is what buttons look like usually in iOS 7. Think of them sort of like links in a web page.
We're going to change the look of the button dramatically (to look like a card) using images, but that's actually somewhat unusual.
Mode
Scale To Fill
Tag
Interaction
User Interaction Enabled
Multiple Touch
Alpha
Background
White Color
Tint
Default
Drawing
Opaque
Hidden
Clears Graphics Context
Clip Subviews
Autoresize Subviews
Stretching
X
Y
Width
Height
Stanford CS193p
Fall 2013

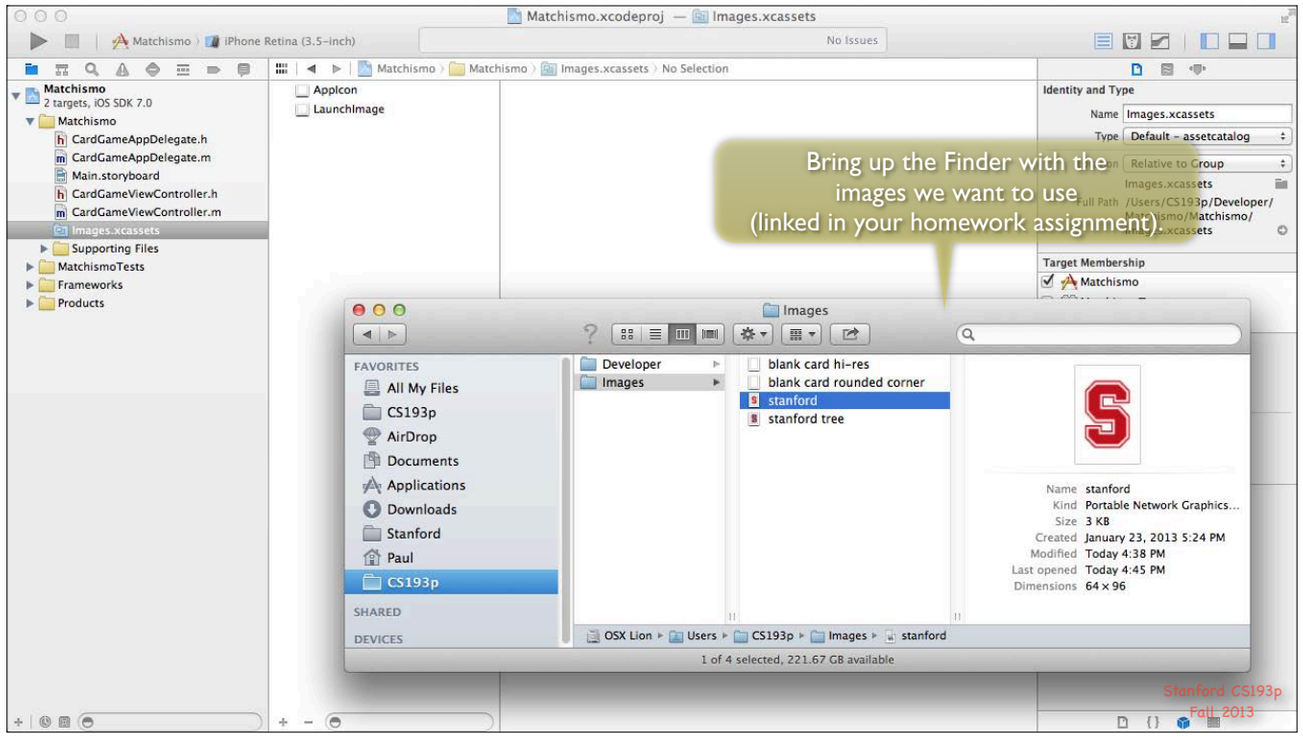

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller
Button
The Background color is currently White.
Default
Recently Used Colors
Dark Text Color
Group Table View Background Color
Light Text Color
Scroll View Textured Background Color
Table Cell Grouped Background Color
View Flipside Background Color
Black Color
Dark Gray Color
Light Gray Color
White Color
Clear Color
Other…
Autoresize Subviews
Stretching
X
Y
Width
Height
Stanford CS193p
Fall 2013

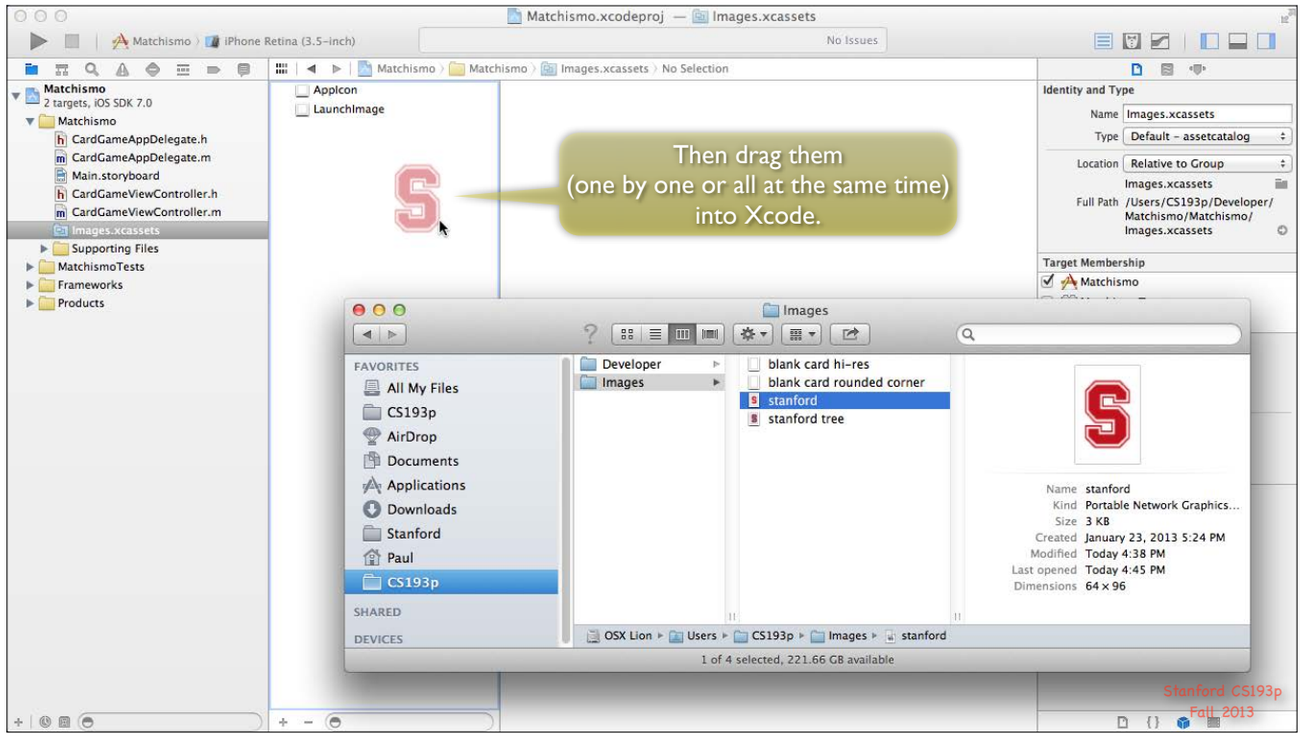

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller
Button
Default
Recently Used Colors
Dark Text Color
Group Table View Background Color
Light Text Color
Scroll View Textured Background Color
Table Cell Grouped Background Color
View Flipside Background Color
Black Color
Dark Gray Color
Light Gray Color
White Color
Clear Color
Other...
Autoresize Subviews
Stretching
X
Y
Width
Height
Click on Other... to change the Background color to something else ...
Stanford CS193p
Fall 2013

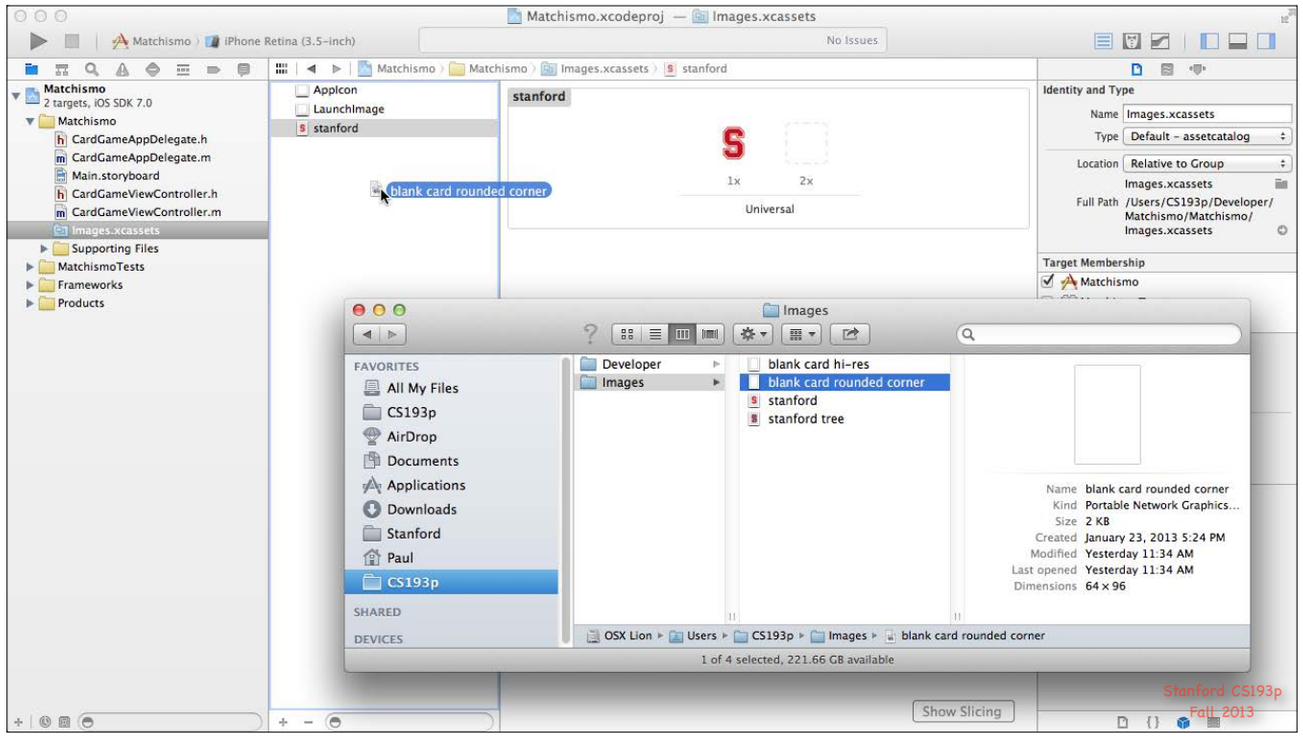

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button
Colors
Moss
Opacity
100 %
Moss is a good color because it's sort of like the green felt of a card table.
You can close this when you've chosen the color.
View
Mode Scale To Fill
Tag
Interaction User Interaction Enabled
Multiple Touch
Alpha
Background
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Autoresize Subviews
Stretching
Width
Height
Stanford CS193p
Fall 2013

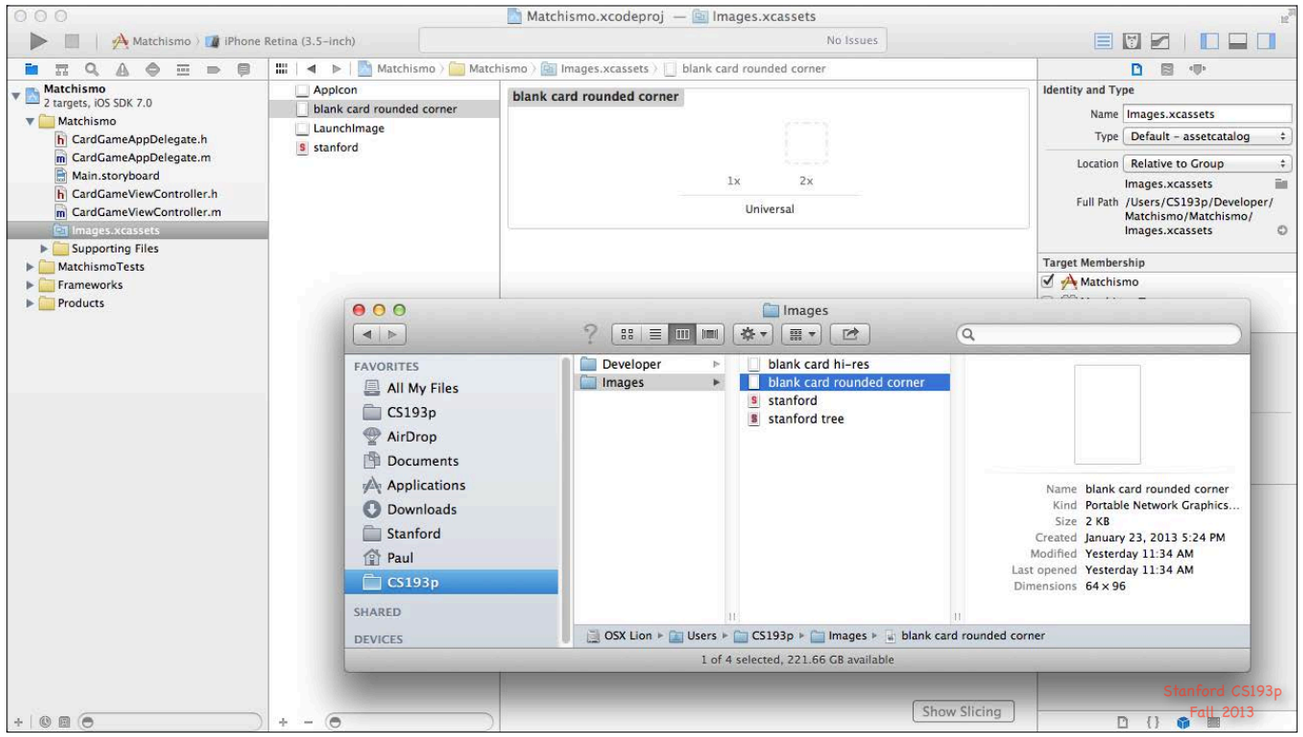

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
You can see the current selection here.
Click on the Button again.
The Button's attributes should appear here.
Notice these little white resize handles. Seeing these is another way to know the Button is selected.
Next we're going to set a background image for the Button (to a blank white card).
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background Default Background Imag
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

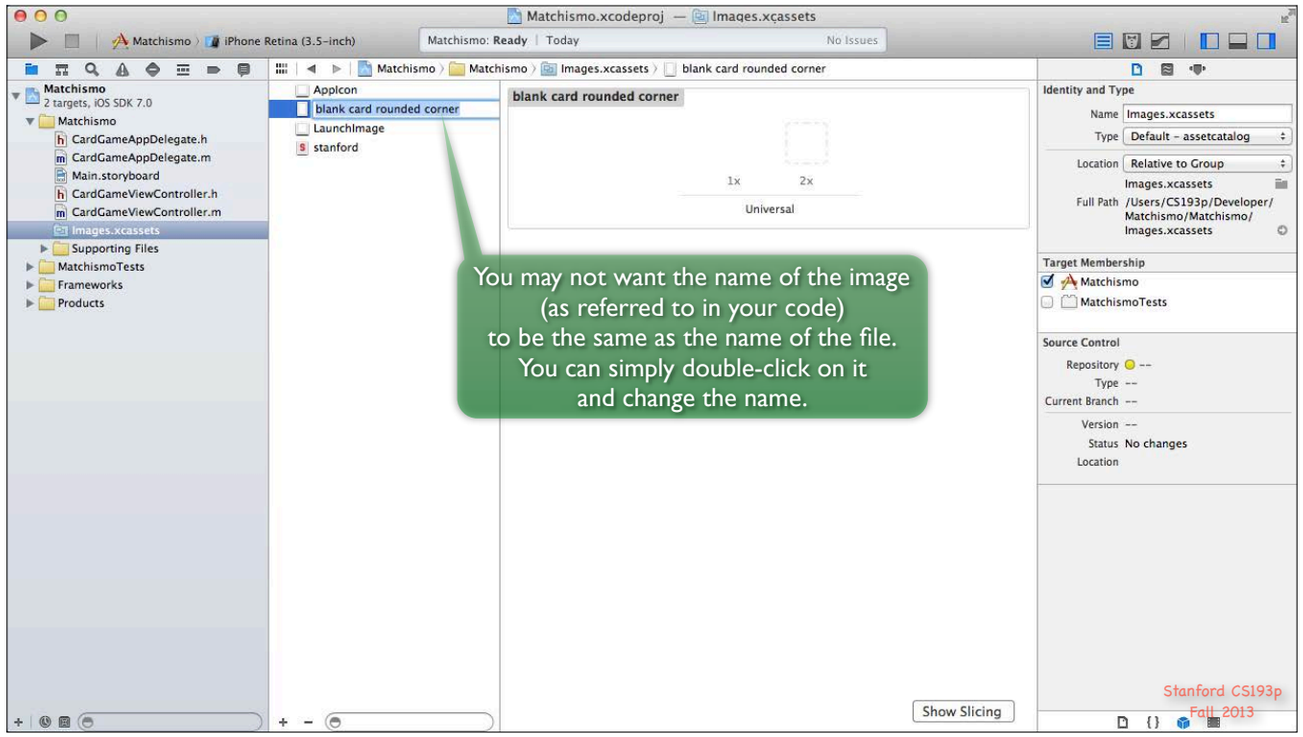

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › No Selection
AppIcon
LaunchImage
We'll set our application's icon next week.
We'll talk about what a Launch Image is later in the course.
This is where all the images in your application go. Click on it.
No Selection
The front of our card is going to have a background of a blank white card and the back is going to have a Stanford logo on it.
We need to drag those 2 images into here so we can use them on the Button.
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013

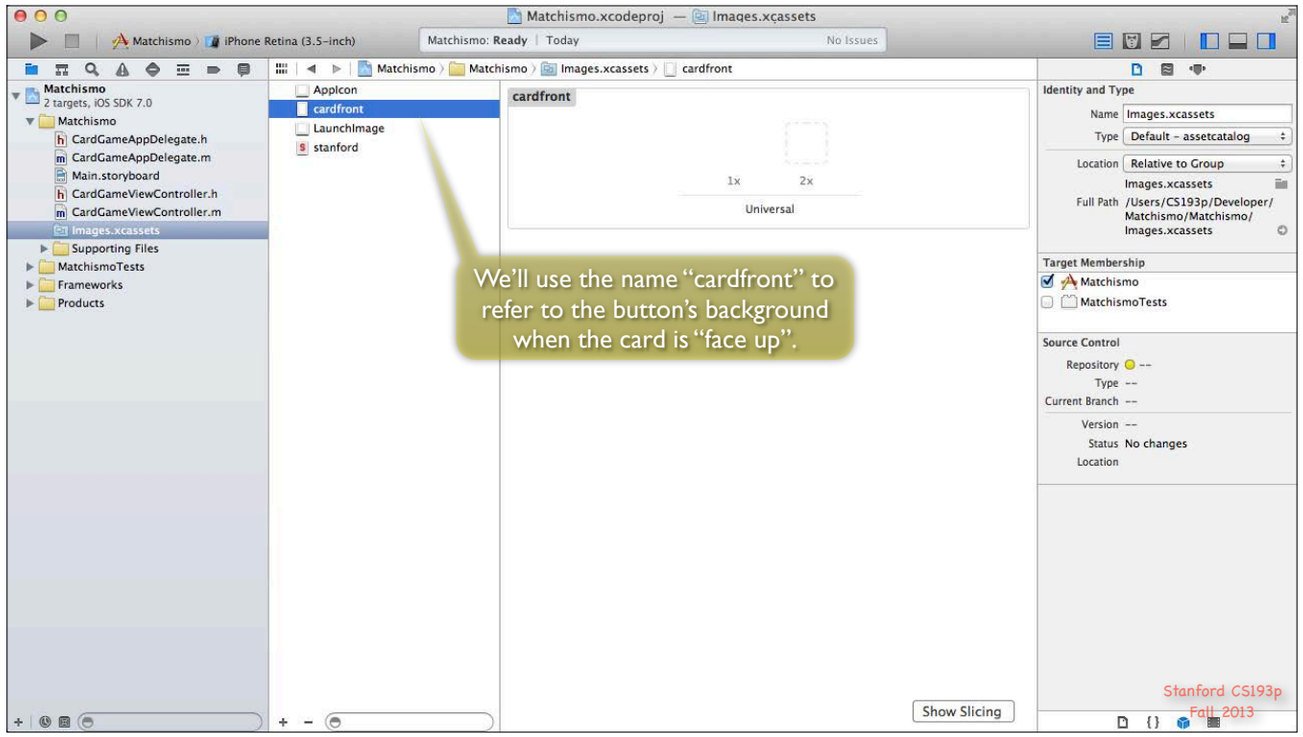

Matchismo.xcodeproj — Images.xcassets
Matchismo
iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › No Selection
AppIcon
LaunchImage
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
Bring up the Finder with the images we want to use (linked in your homework assignment).
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name stanford
Kind Portable Network Graphics...
Size 3 KB
Created January 23, 2013 5:24 PM
Modified Today 4:38 PM
Last opened Today 4:45 PM
Dimensions 64 × 96
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ stanford
1 of 4 selected, 221.67 GB available
Stanford CS193p
Fall 2013

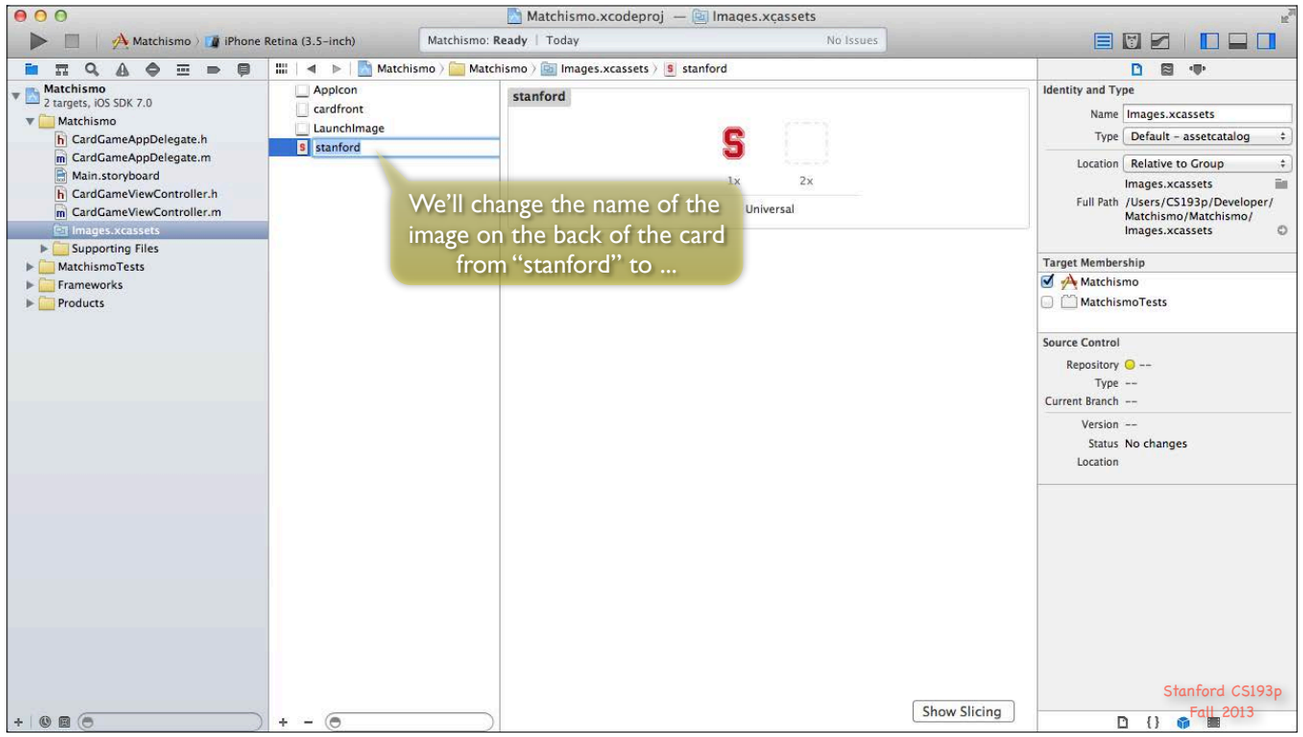

Matchismo.xcodeproj — Images.xcassets
Matchismo
iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › No Selection
AppIcon
LaunchImage
Then drag them
(one by one or all at the same time)
into Xcode.
Identity and Type
Name Images.xcassets
Type Default – assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name stanford
Kind Portable Network Graphics...
Size 3 KB
Created January 23, 2013 5:24 PM
Modified Today 4:38 PM
Last opened Today 4:45 PM
Dimensions 64 × 96
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ stanford
1 of 4 selected, 221.66 GB available
Stanford CS193p
Fall 2013

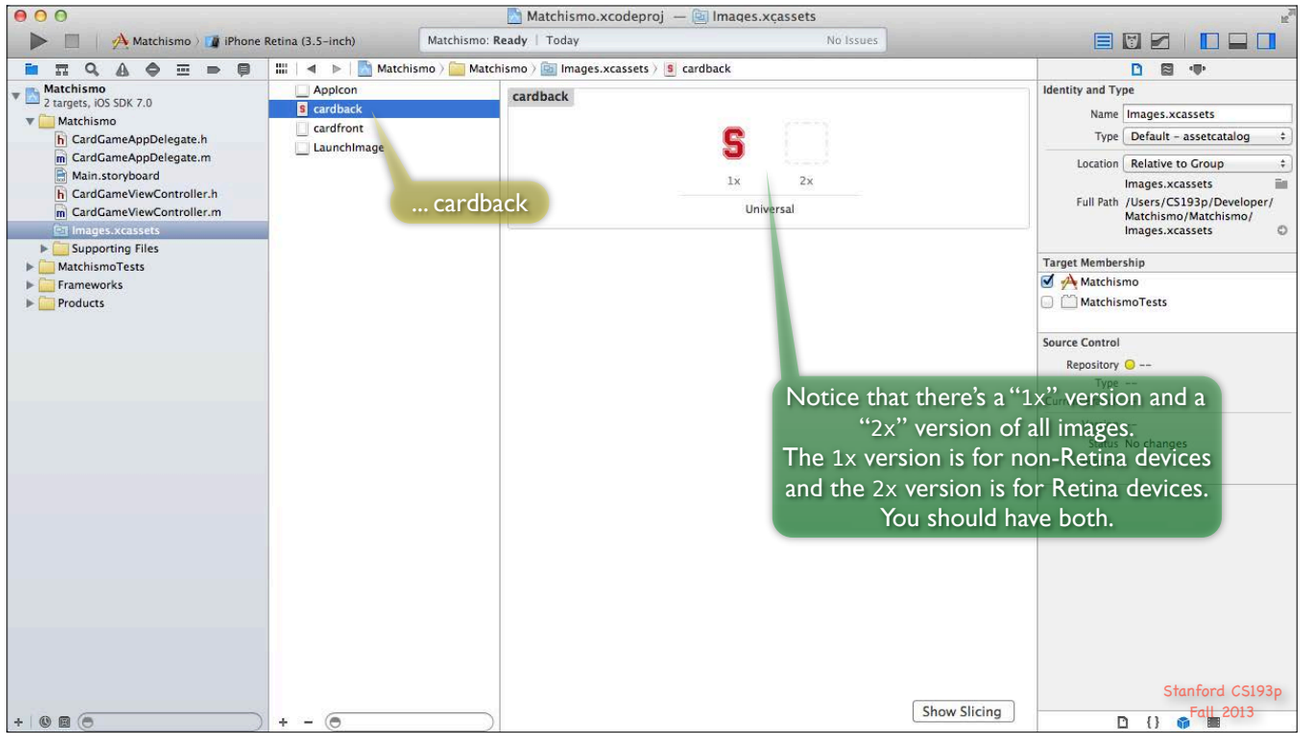

Matchismo.xcodeproj — Images.xcassets
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
AppIcon
LaunchImage
stanford
blank card rounded corner
1x
2x
Universal
Identity and Type
Name Images.xcassets
Type Default – assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name blank card rounded corner
Kind Portable Network Graphics…
Size 2 KB
Created January 23, 2013 5:24 PM
Modified Yesterday 11:34 AM
Last opened Yesterday 11:34 AM
Dimensions 64 × 96
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ blank card rounded corner
1 of 4 selected, 221.66 GB available
Show Slicing
Stanford CS193p
Fall 2013

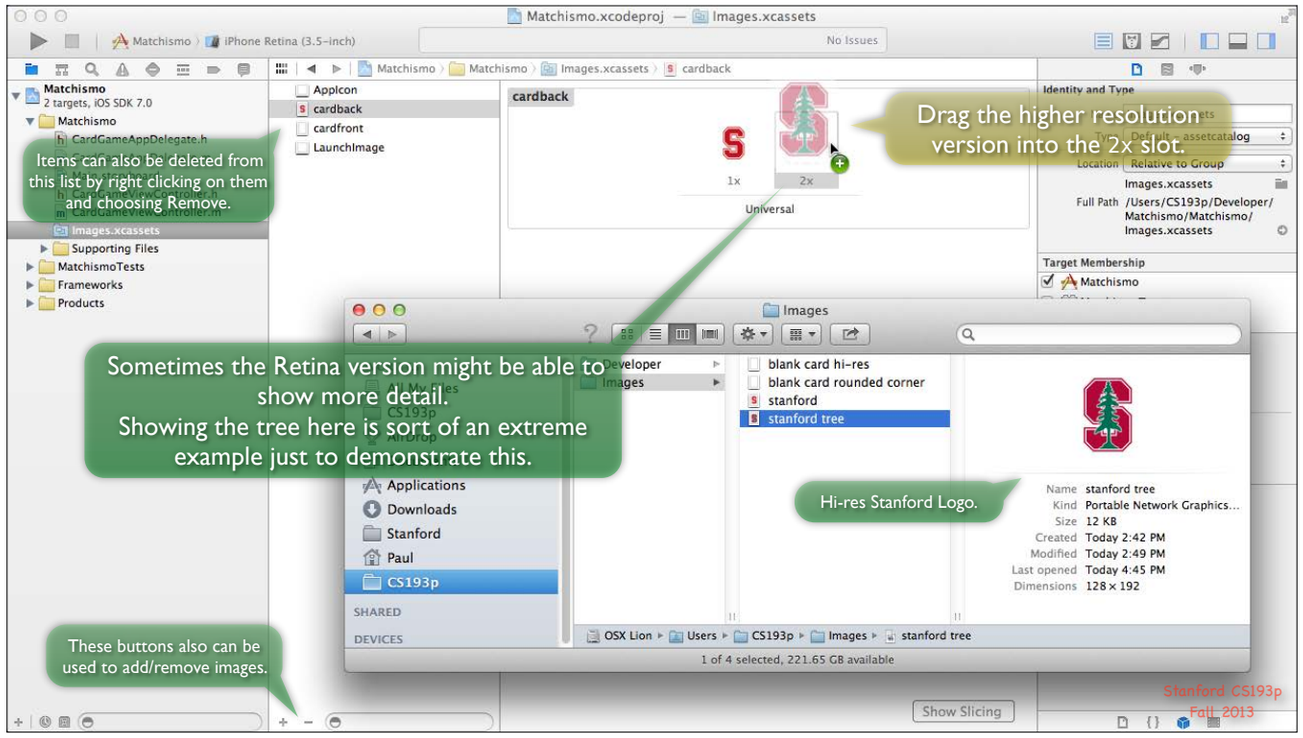

Matchismo.xcodeproj — Images.xcassets
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
AppIcon
blank card rounded corner
LaunchImage
stanford
blank card rounded corner
1x
2x
Universal
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name blank card rounded corner
Kind Portable Network Graphics...
Size 2 KB
Created January 23, 2013 5:24 PM
Modified Yesterday 11:34 AM
Last opened Yesterday 11:34 AM
Dimensions 64 × 96
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ blank card rounded corner
1 of 4 selected, 221.66 GB available
Show Slicing
Stanford CS193p
Fall 2013

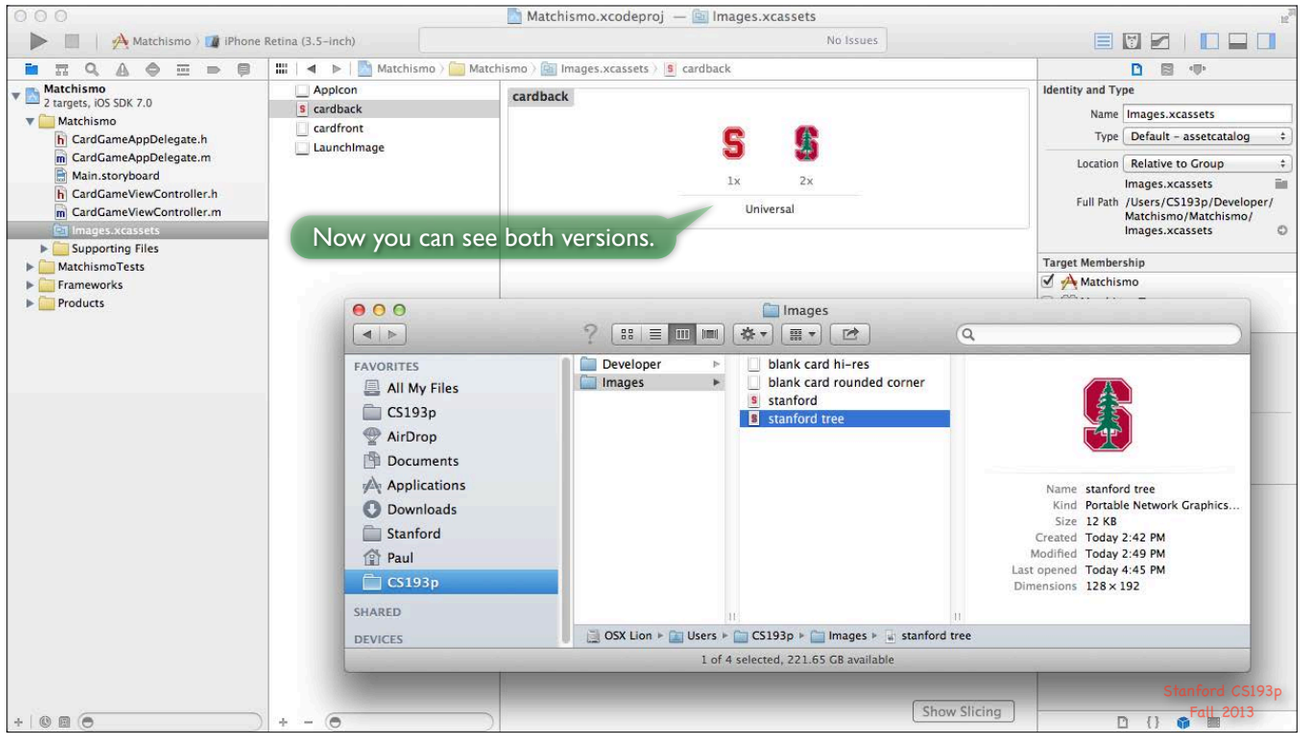

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo › Matchismo › Images.xcassets › blank card rounded corner
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
AppIcon
blank card rounded corner
LaunchImage
stanford
blank card rounded corner
1x
2x
Universal
You may not want the name of the image
(as referred to in your code)
to be the same as the name of the file.
You can simply double-click on it
and change the name.
Show Slicing
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013

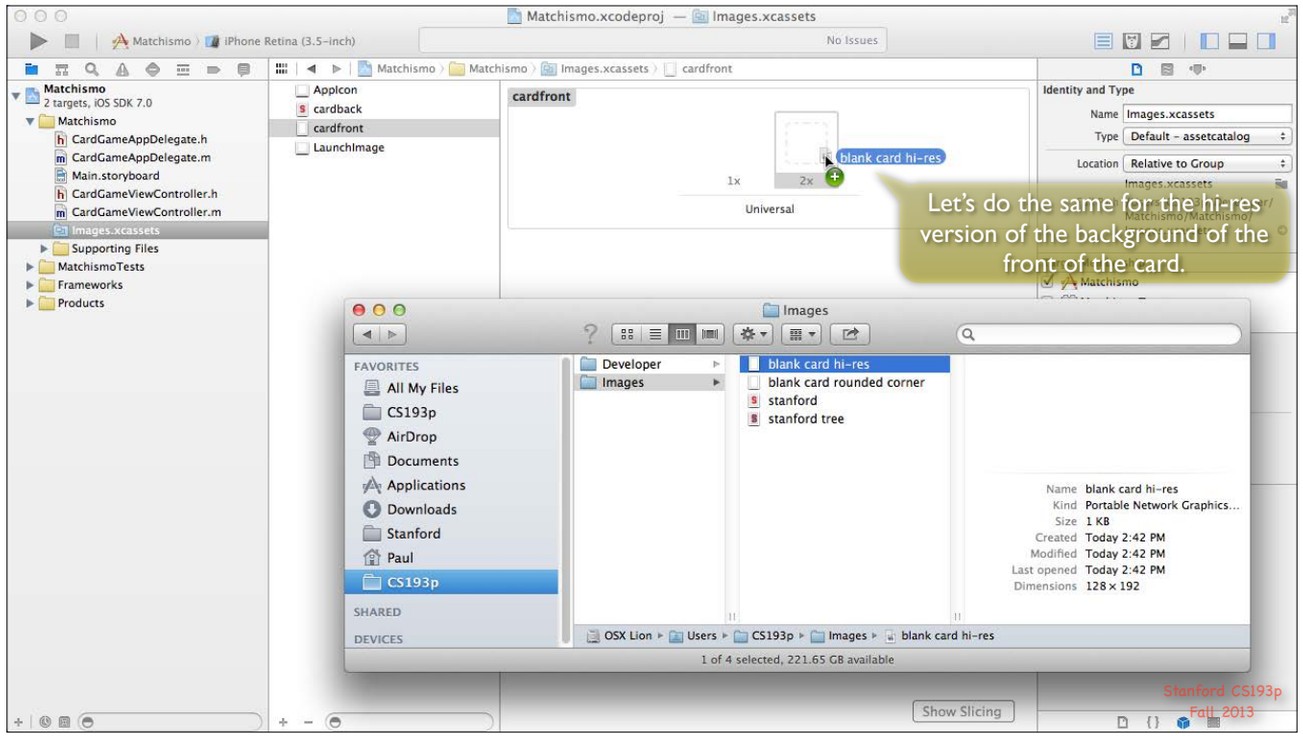

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › cardfront
AppIcon
cardfront
LaunchImage
stanford
cardfront
1x
2x
Universal
We'll use the name "cardfront" to refer to the button's background when the card is "face up".
Show Slicing
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013

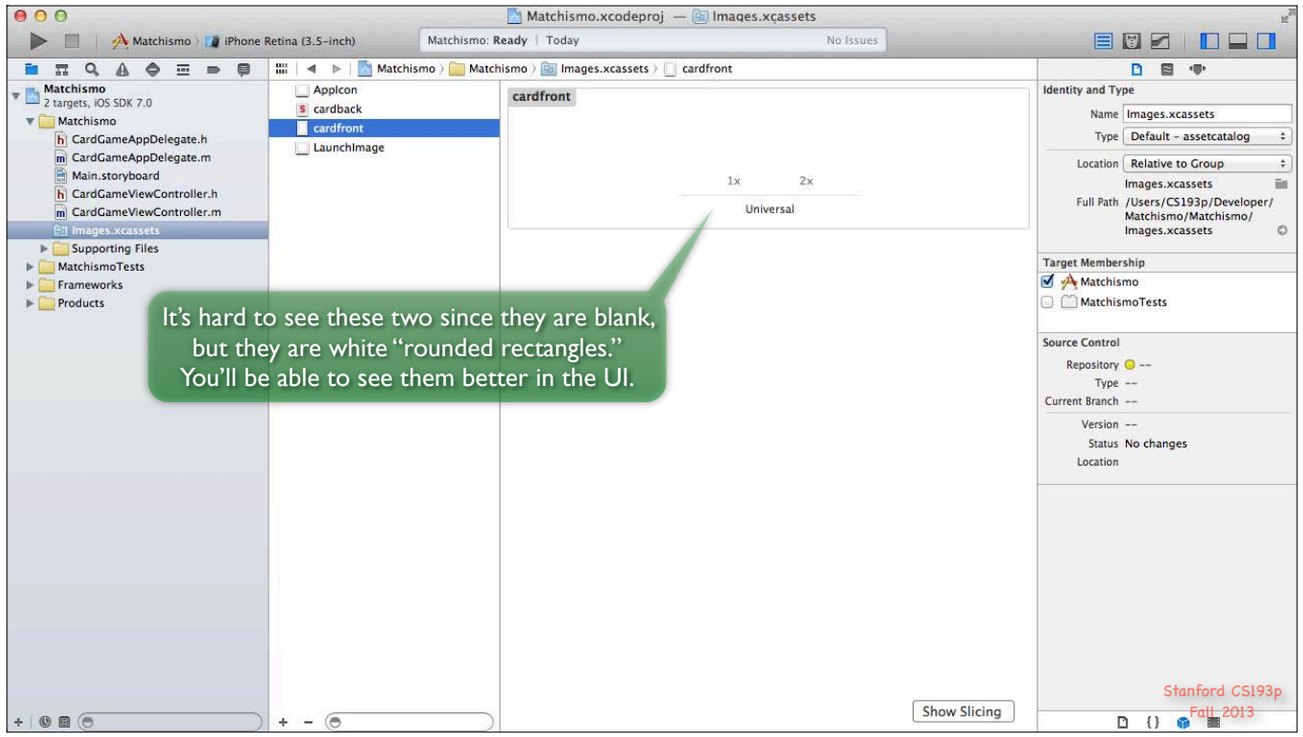

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › stanford
AppIcon
cardfront
LaunchImage
stanford
stanford
1x
2x
Universal
We'll change the name of the image on the back of the card from "stanford" to ...
Show Slicing
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013

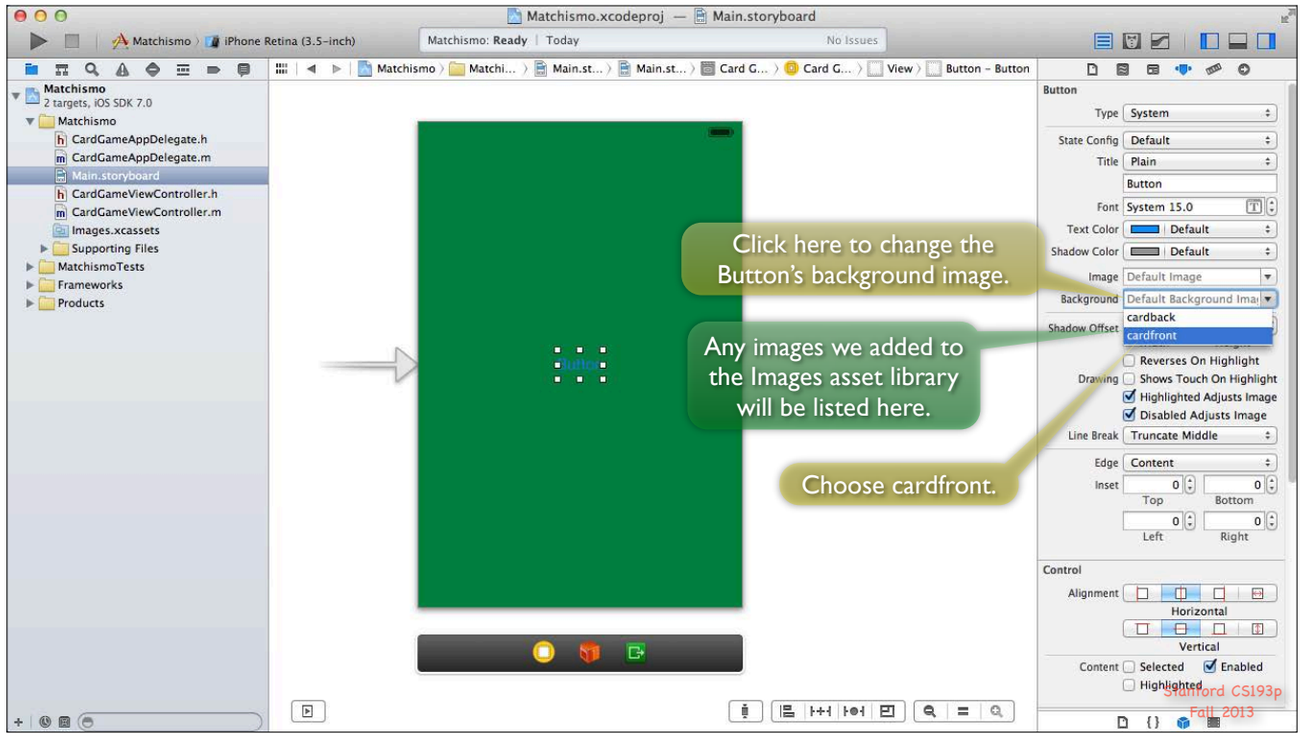

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › cardback
AppIcon
cardback
cardfront
LaunchImage
... cardback
cardback
1x
2x
Universal
Show Slicing
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Notice that there's a "1x" version and a "2x" version of all images.
The 1x version is for non-Retina devices and the 2x version is for Retina devices.
You should have both.
Stanford CS193p
Fall 2013

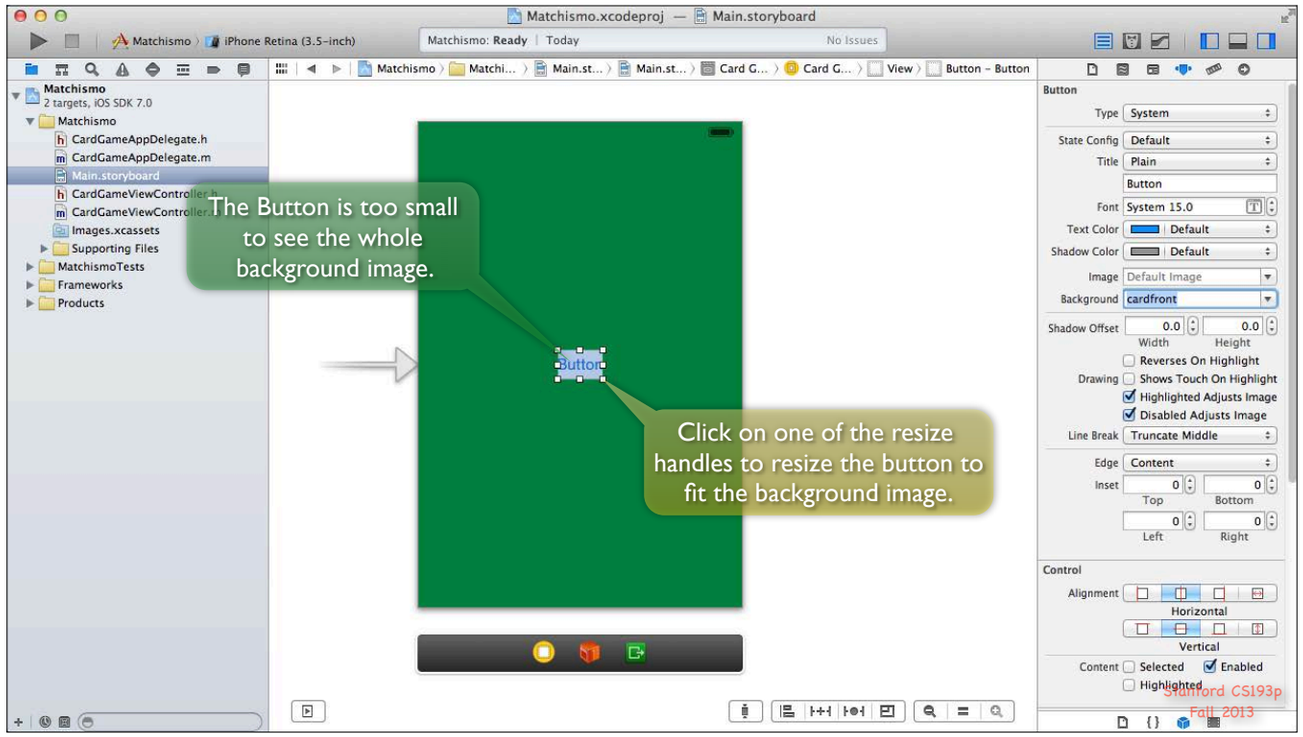

Matchismo.xcodeproj — Images.xcassets
Items can also be deleted from this list by right clicking on them and choosing Remove.
Drag the higher resolution version into the 2x slot.
Sometimes the Retina version might be able to show more detail.
Showing the tree here is sort of an extreme example just to demonstrate this.
Hi-res Stanford Logo.
These buttons also can be used to add/remove images.
Stanford CS193p
Fall 2013

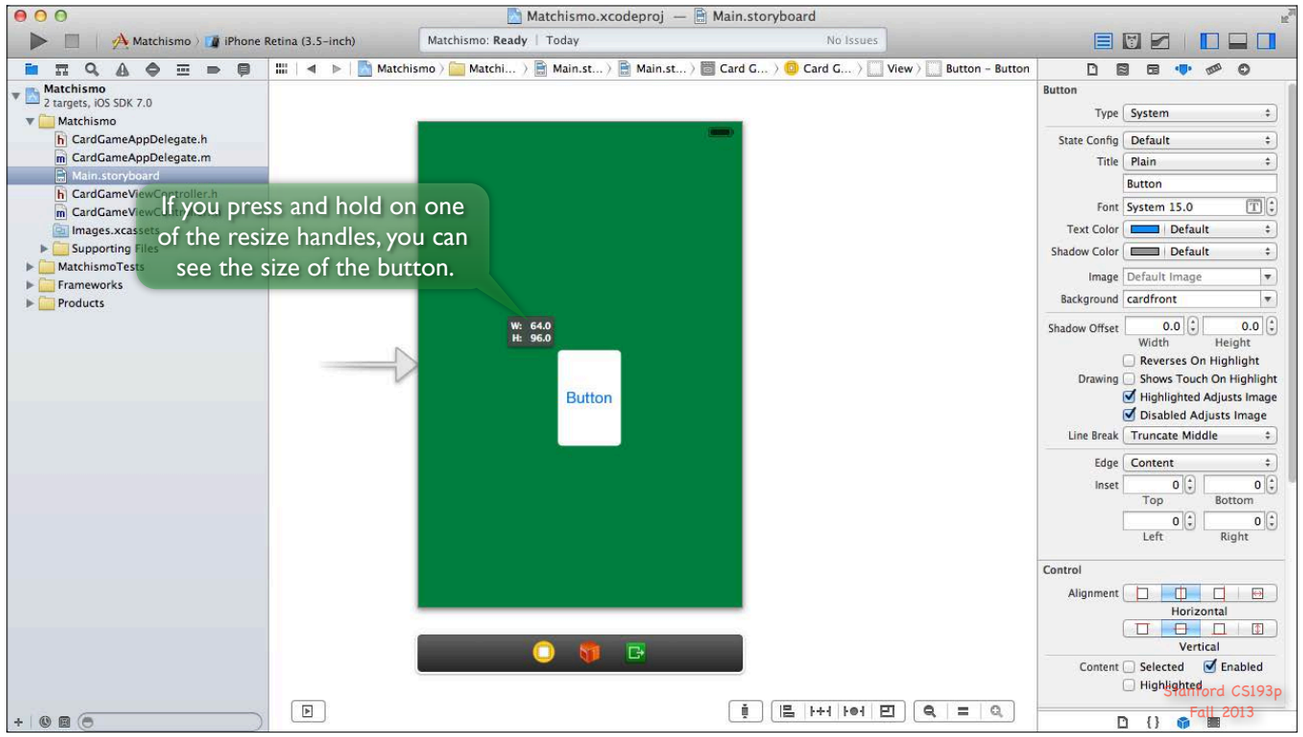

Matchismo.xcodeproj — Images.xcassets
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
AppIcon
cardback
cardfront
LaunchImage
cardback
1x
2x
Universal
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
Now you can see both versions.
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name stanford tree
Kind Portable Network Graphics...
Size 12 KB
Created Today 2:42 PM
Modified Today 2:49 PM
Last opened Today 4:45 PM
Dimensions 128 x 192
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ stanford tree
1 of 4 selected, 221.65 GB available
Show Slicing
Stanford CS193p
Fall 2013

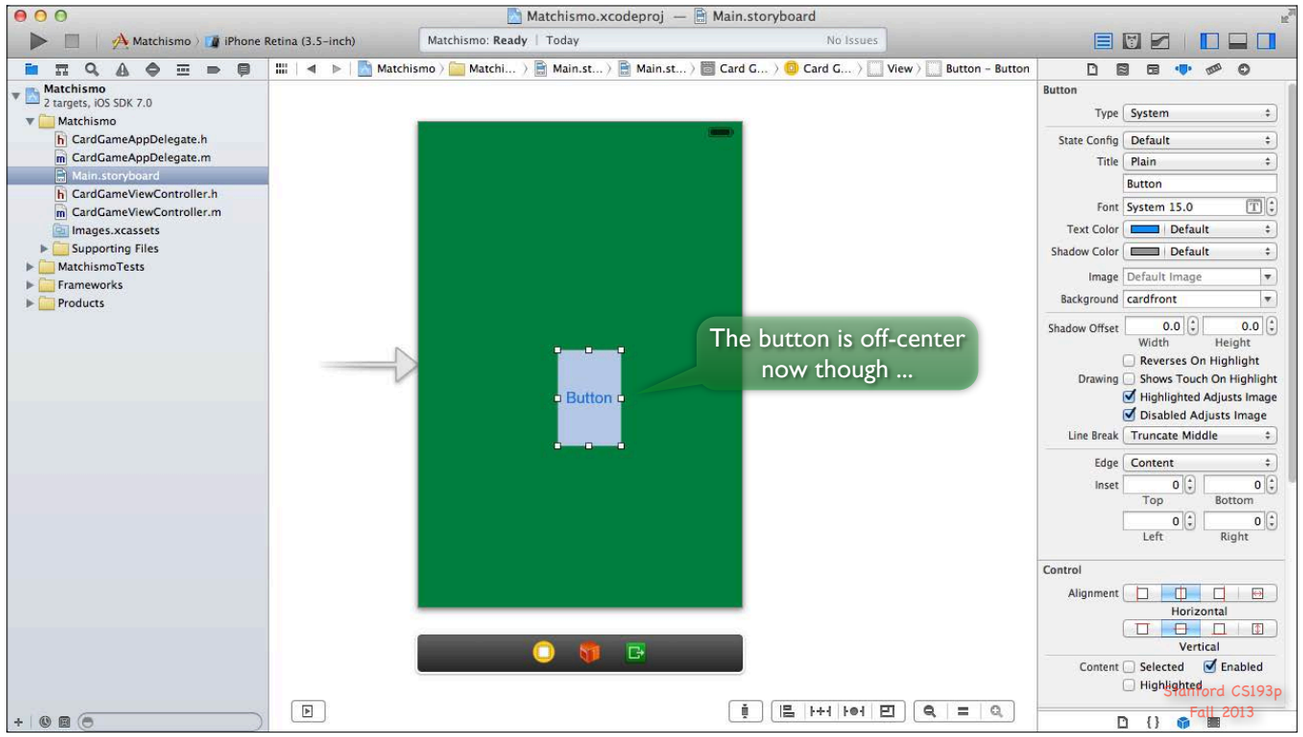

Matchismo.xcodeproj — Images.xcassets
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
AppIcon
cardback
cardfront
LaunchImage
cardfront
blank card hi-res
1x
2x
Universal
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Let's do the same for the hi-res version of the background of the front of the card.
Images
FAVORITES
All My Files
CS193p
AirDrop
Documents
Applications
Downloads
Stanford
Paul
CS193p
SHARED
DEVICES
Developer
Images
blank card hi-res
blank card rounded corner
stanford
stanford tree
Name blank card hi-res
Kind Portable Network Graphics...
Size 1 KB
Created Today 2:42 PM
Modified Today 2:42 PM
Last opened Today 2:42 PM
Dimensions 128 x 192
OSX Lion ▸ Users ▸ CS193p ▸ Images ▸ blank card hi-res
1 of 4 selected, 221.65 GB available
Show Slicing
Stanford CS193p
Fall 2013

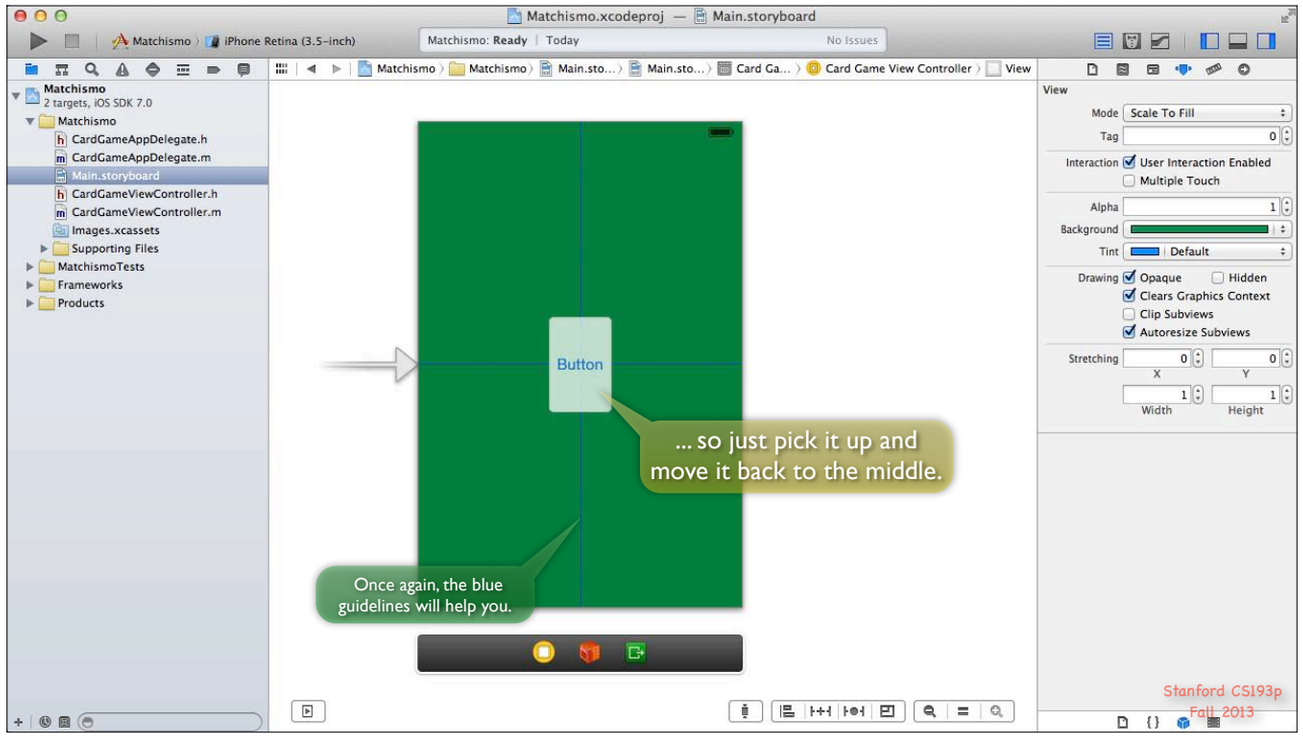

Matchismo.xcodeproj — Images.xcassets
Matchismo › iPhone Retina (3.5-inch)
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Images.xcassets › cardfront
AppIcon
cardback
cardfront
LaunchImage
cardfront
1x
2x
Universal
It's hard to see these two since they are blank, but they are white "rounded rectangles." You'll be able to see them better in the UI.
Show Slicing
Identity and Type
Name Images.xcassets
Type Default - assetcatalog
Location Relative to Group
Images.xcassets
Full Path /Users/CS193p/Developer/Matchismo/Matchismo/Images.xcassets
Target Membership
Matchismo
MatchismoTests
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button – Button
Click here to change the Button's background image.
Any images we added to the Images asset library will be listed here.
Choose cardfront.
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background Default Background Ima
cardback
cardfront
Shadow Offset
Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset
Top
Bottom
Left
Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

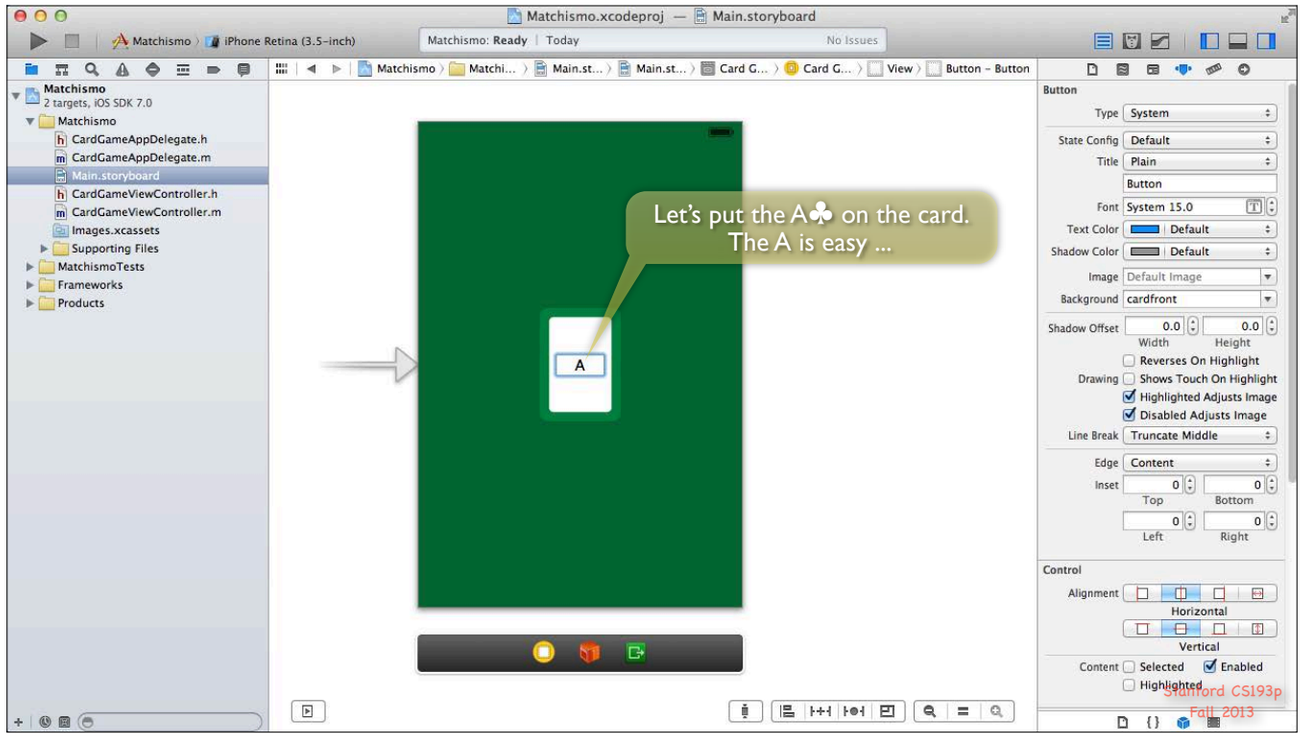

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
The Button is too small to see the whole background image.
Button
Click on one of the resize handles to resize the button to fit the background image.
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Drawing Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

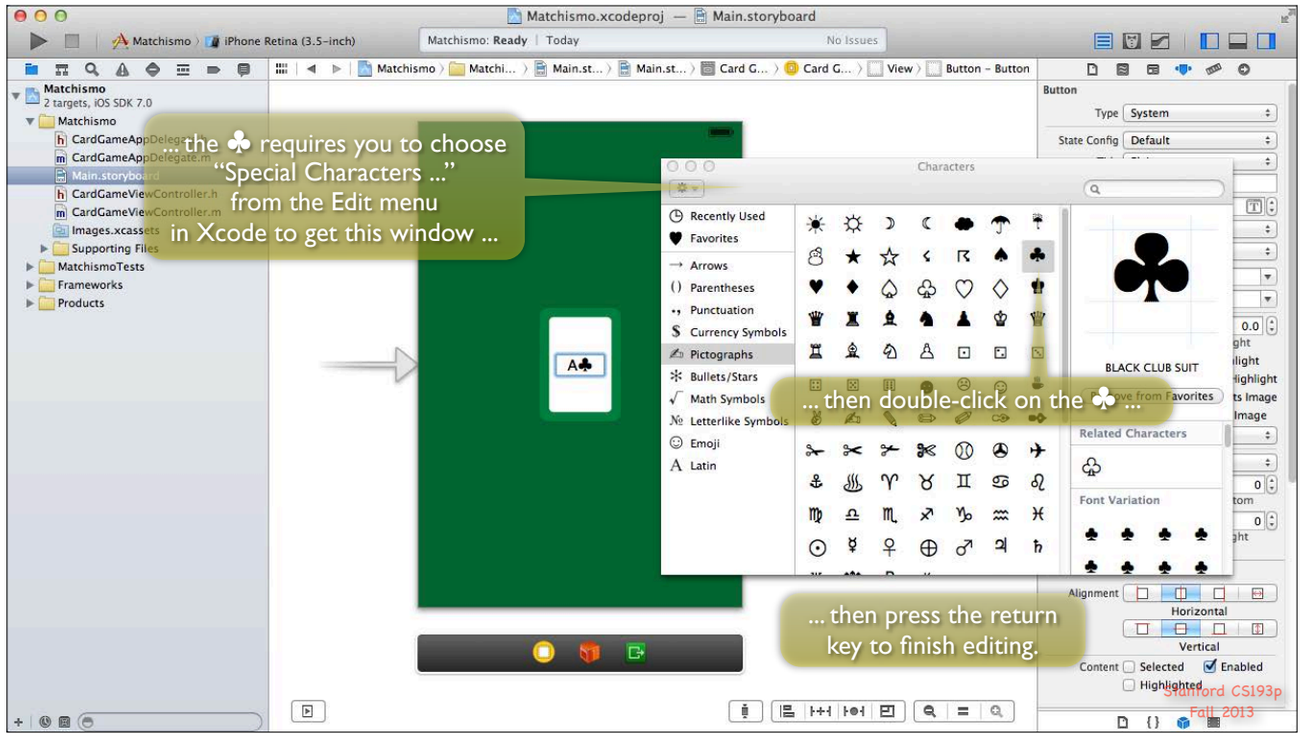

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
If you press and hold on one of the resize handles, you can see the size of the button.
W: 64.0
H: 96.0
Button
Type System
State Config Default
Title Plain
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset
Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

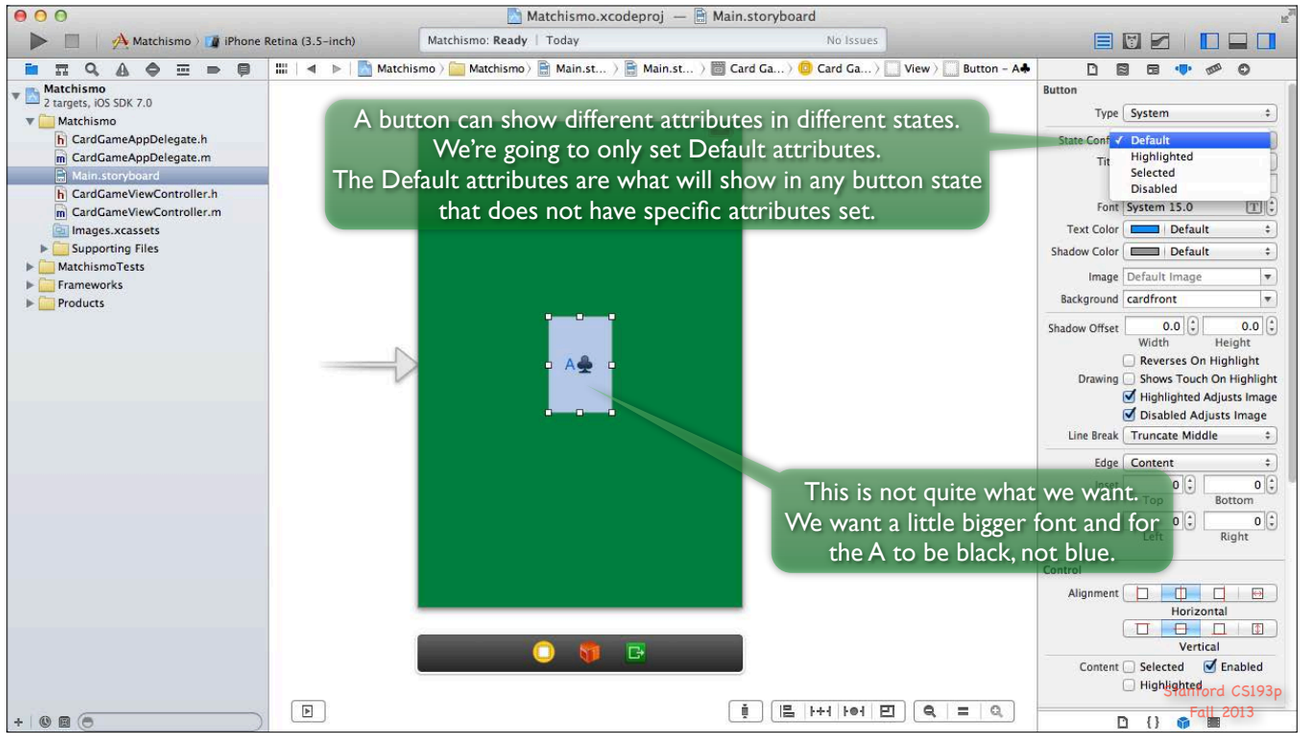

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button
The button is off-center now though ...
Button
Type System
State Config Default
Title Plain
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset
Reverses On Highlight
Drawing
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

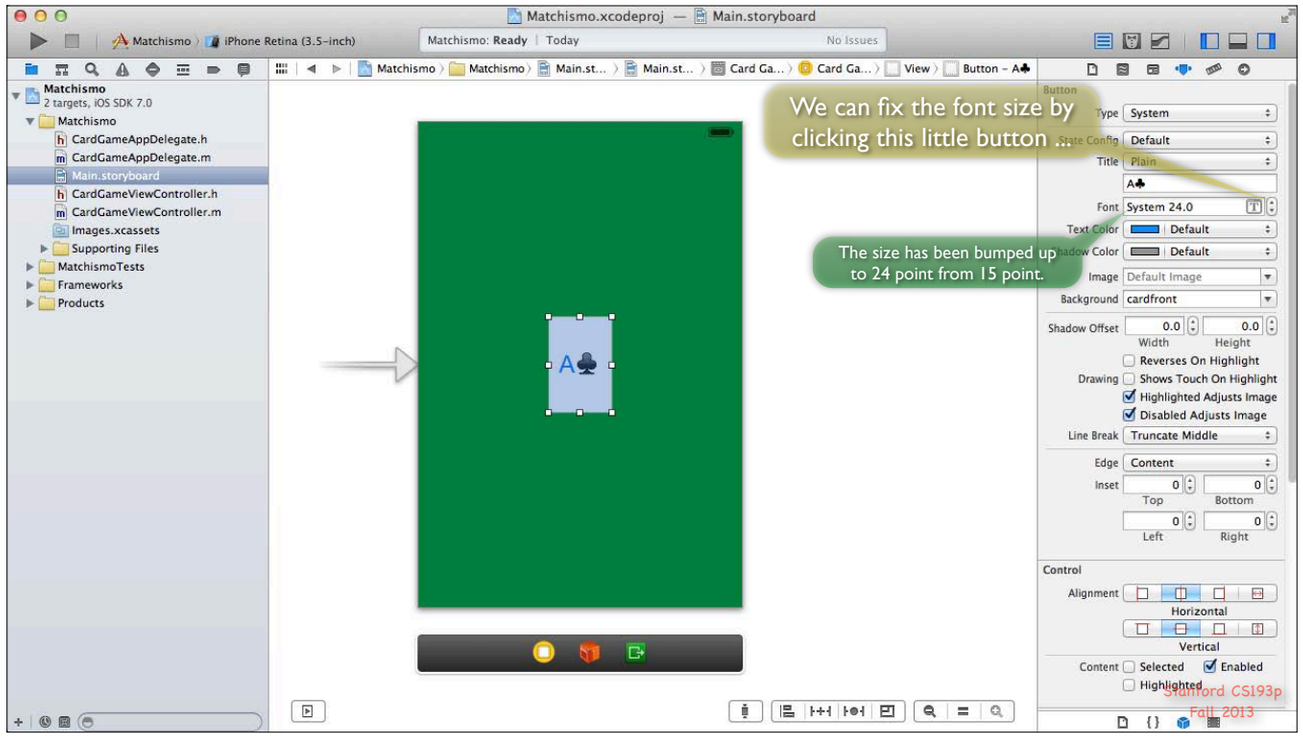

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card Game View Controller
View
Button
... so just pick it up and move it back to the middle.
Once again, the blue guidelines will help you.
Mode
Scale To Fill
Tag
Interaction
User Interaction Enabled
Multiple Touch
Alpha
Background
Tint
Default
Drawing
Opaque
Hidden
Clears Graphics Context
Clip Subviews
Autoresize Subviews
Stretching
X
Y
Width
Height
Stanford CS193p
Fall 2013

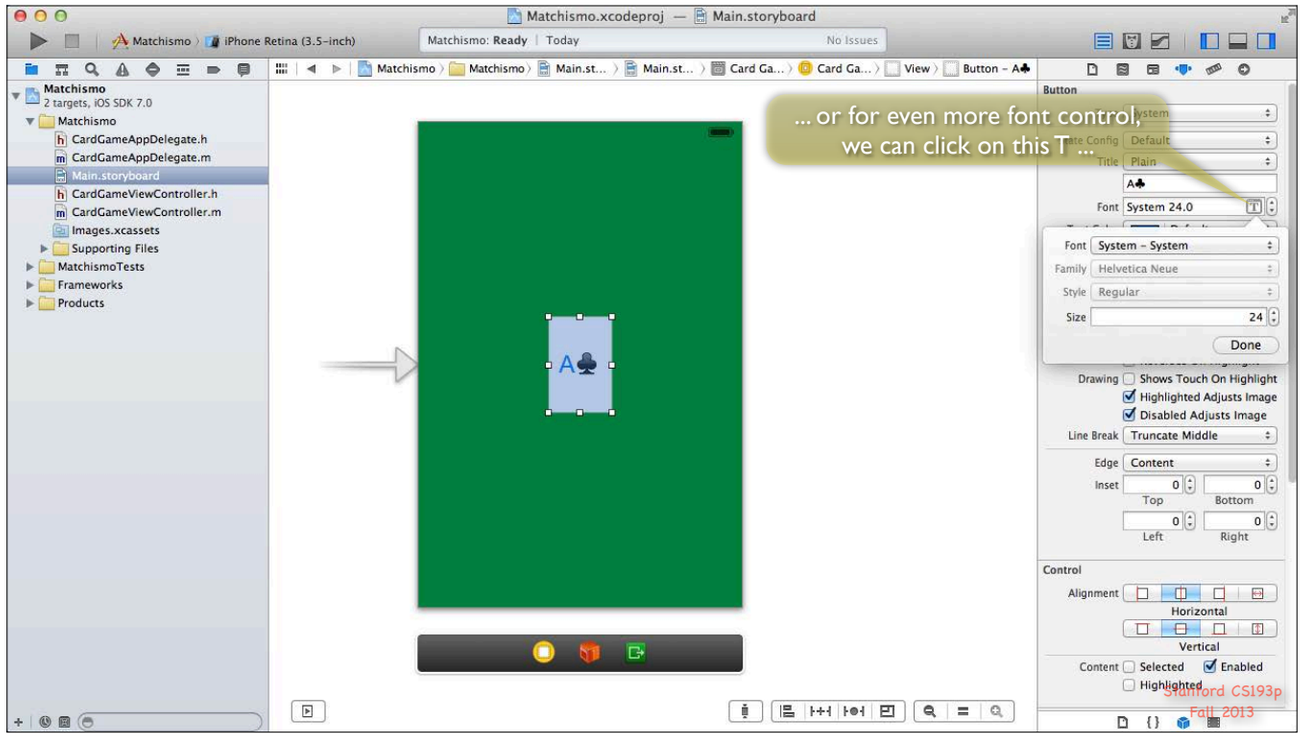

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button – Button
Double-click on the button to change the text on the card.
You can also edit the button's text in the Attributes Inspector here.
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

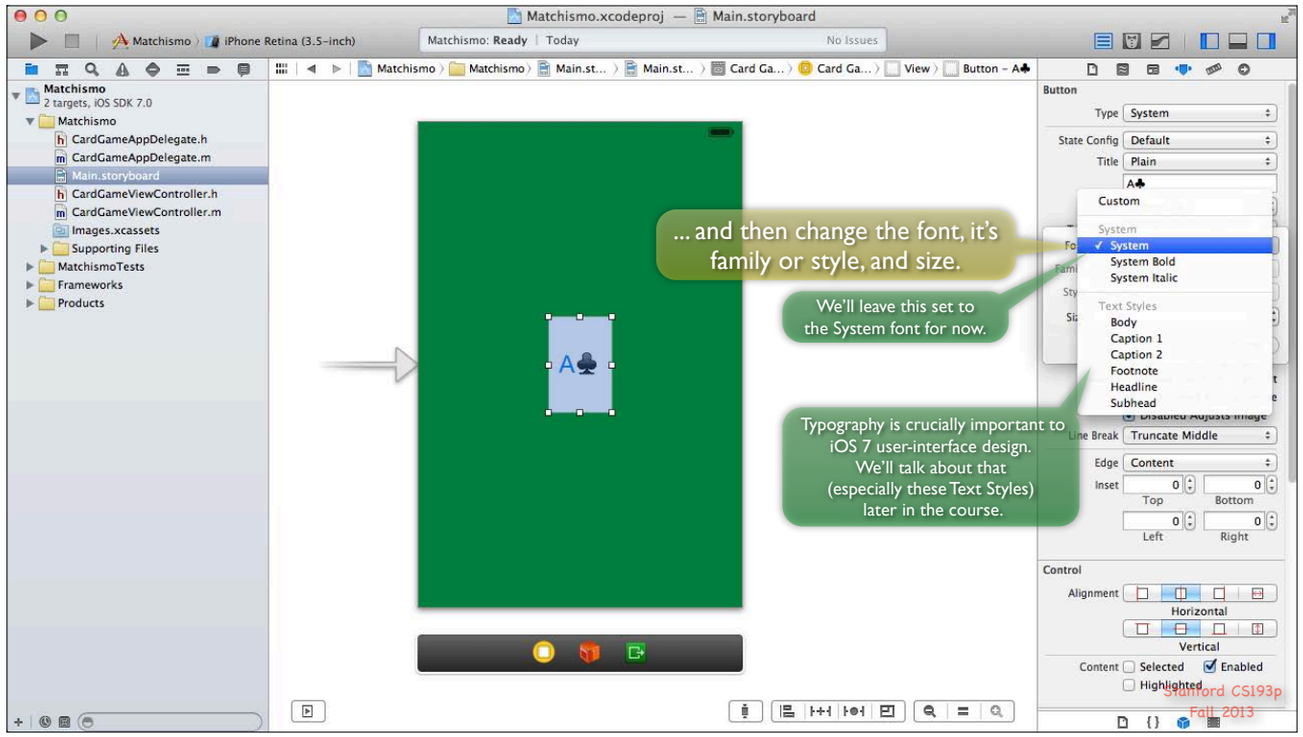

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Let's put the A♣ on the card.
The A is easy ...
A
Button
Type System
State Config Default
Title Plain
Button
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

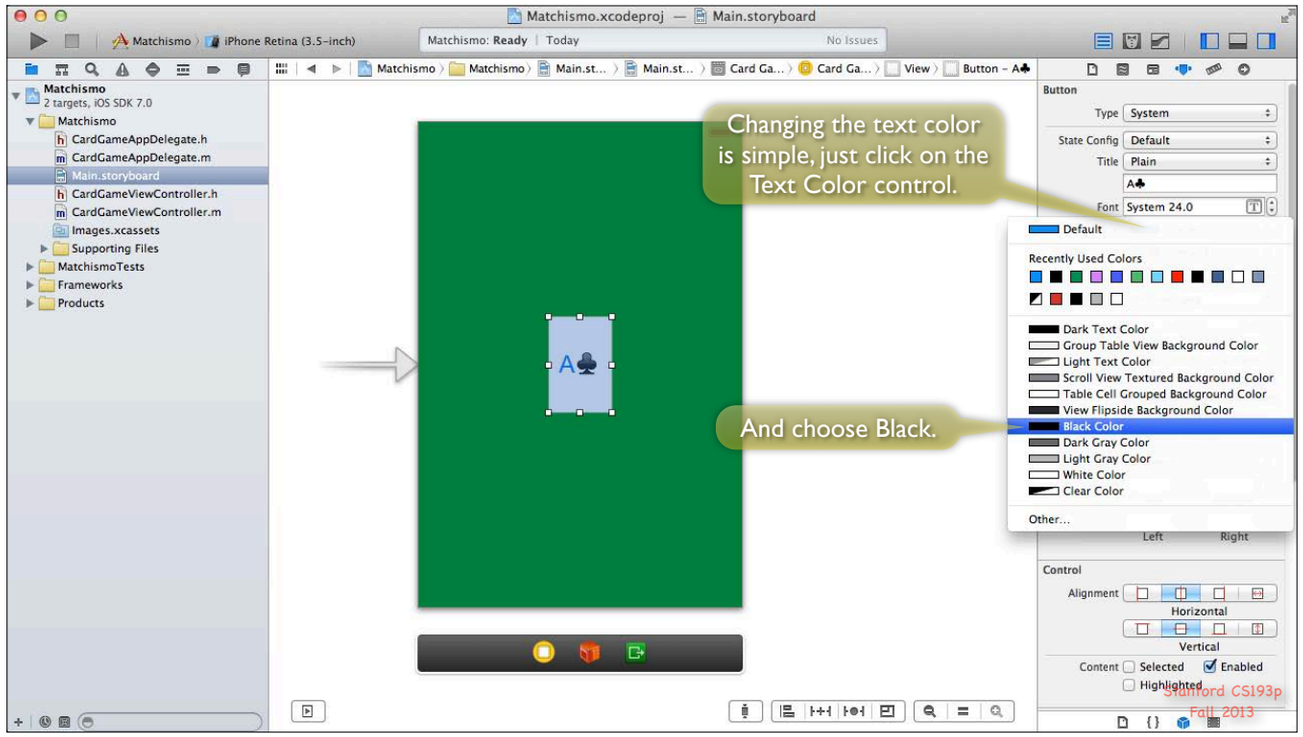

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
... the ♣ requires you to choose "Special Characters ..." from the Edit menu in Xcode to get this window ...
Characters
Recently Used
Favorites
Arrows
Parentheses
Punctuation
Currency Symbols
Pictographs
Bullets/Stars
Math Symbols
Letterlike Symbols
Emoji
Latin
BLACK CLUB SUIT
Remove from Favorites
Related Characters
Font Variation
... then double-click on the ♣ ...
... then press the return key to finish editing.
Button
Type System
State Config Default
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

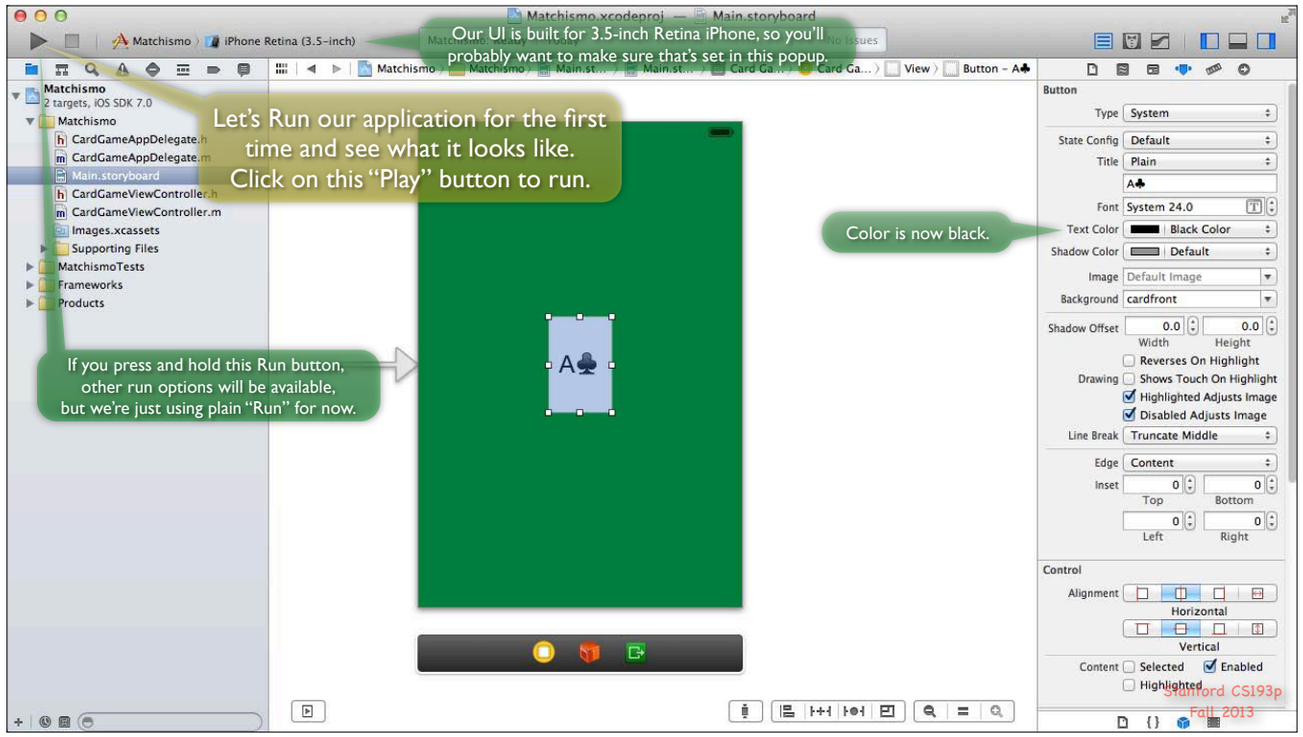

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
A button can show different attributes in different states.
We're going to only set Default attributes.
The Default attributes are what will show in any button state that does not have specific attributes set.
This is not quite what we want.
We want a little bigger font and for the A to be black, not blue.
Button
Type System
Default
Highlighted
Selected
Disabled
Font System 15.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset
Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

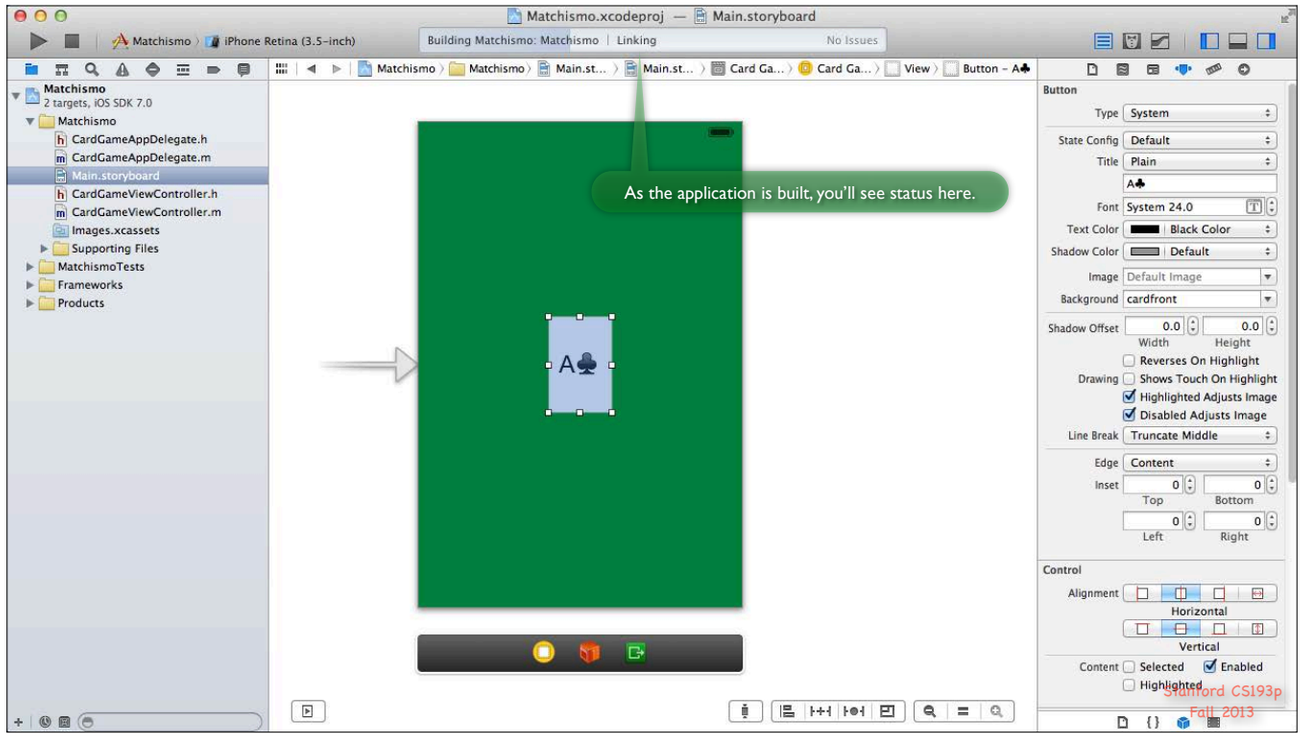

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
We can fix the font size by clicking this little button ...
The size has been bumped up to 24 point from 15 point.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Default
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

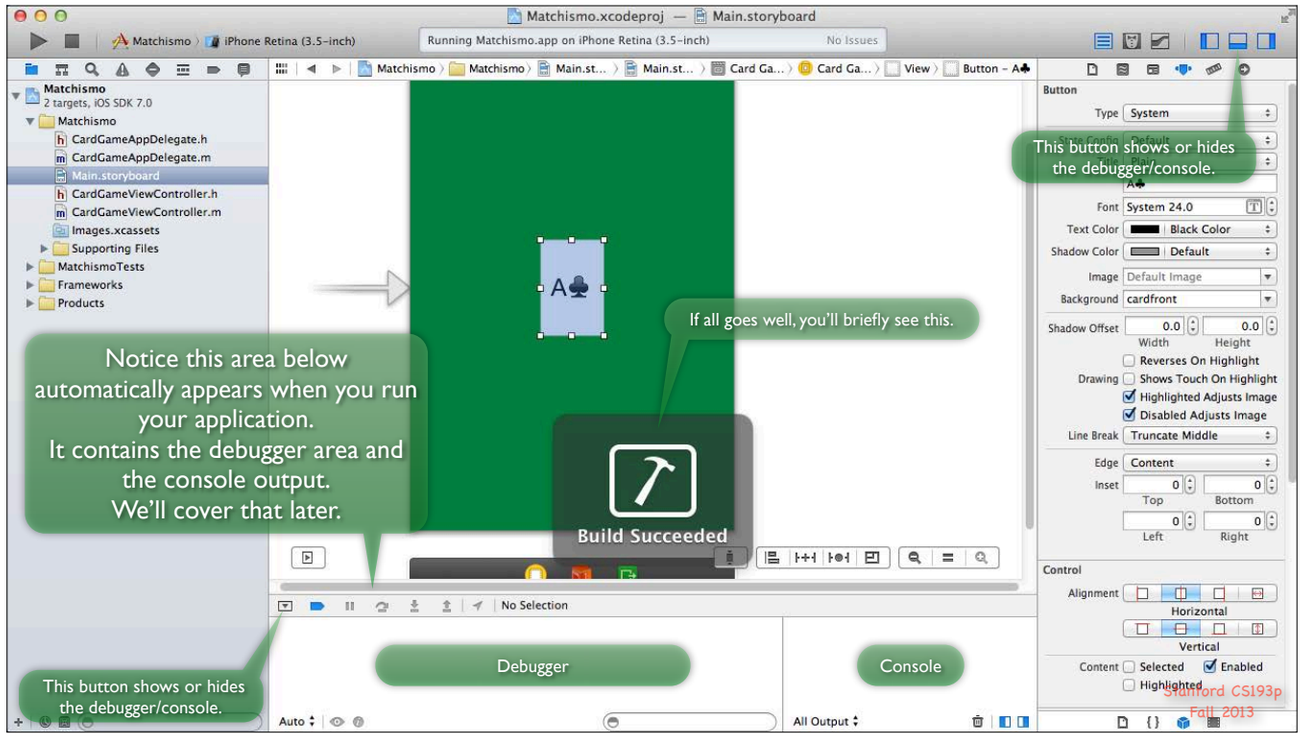

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
... or for even more font control, we can click on this T ...
Button
Font
System 24.0
Font
System – System
Family
Helvetica Neue
Style
Regular
Size
24
Done
Drawing
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break
Truncate Middle
Edge
Content
Inset
Top
Bottom
Left
Right
Control
Alignment
Horizontal
Vertical
Content
Selected
Enabled
Highlighted
Stanford CS193p
Fall 2013

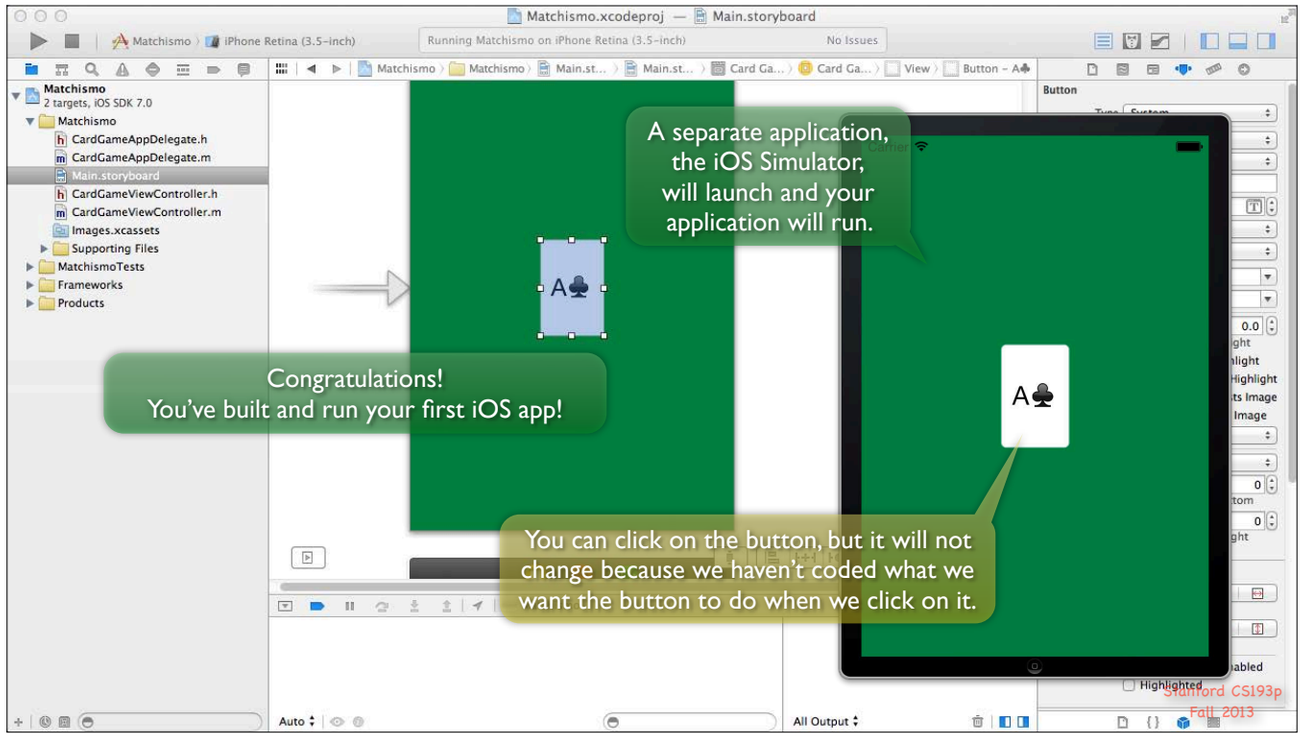

Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
... and then change the font, it's family or style, and size.
We'll leave this set to the System font for now.
Typography is crucially important to iOS 7 user-interface design. We'll talk about that (especially these Text Styles) later in the course.
Button
Type System
State Config Default
Title Plain
Custom
System
System
System Bold
System Italic
Text Styles
Body
Caption 1
Caption 2
Footnote
Headline
Subhead
Line Break Truncate Middle
Edge Content
Inset
Top
Bottom
Left
Right
Control
Alignment
Horizontal
Vertical
Content
Selected
Enabled
Highlighted
Stanford CS193p
Fall 2013

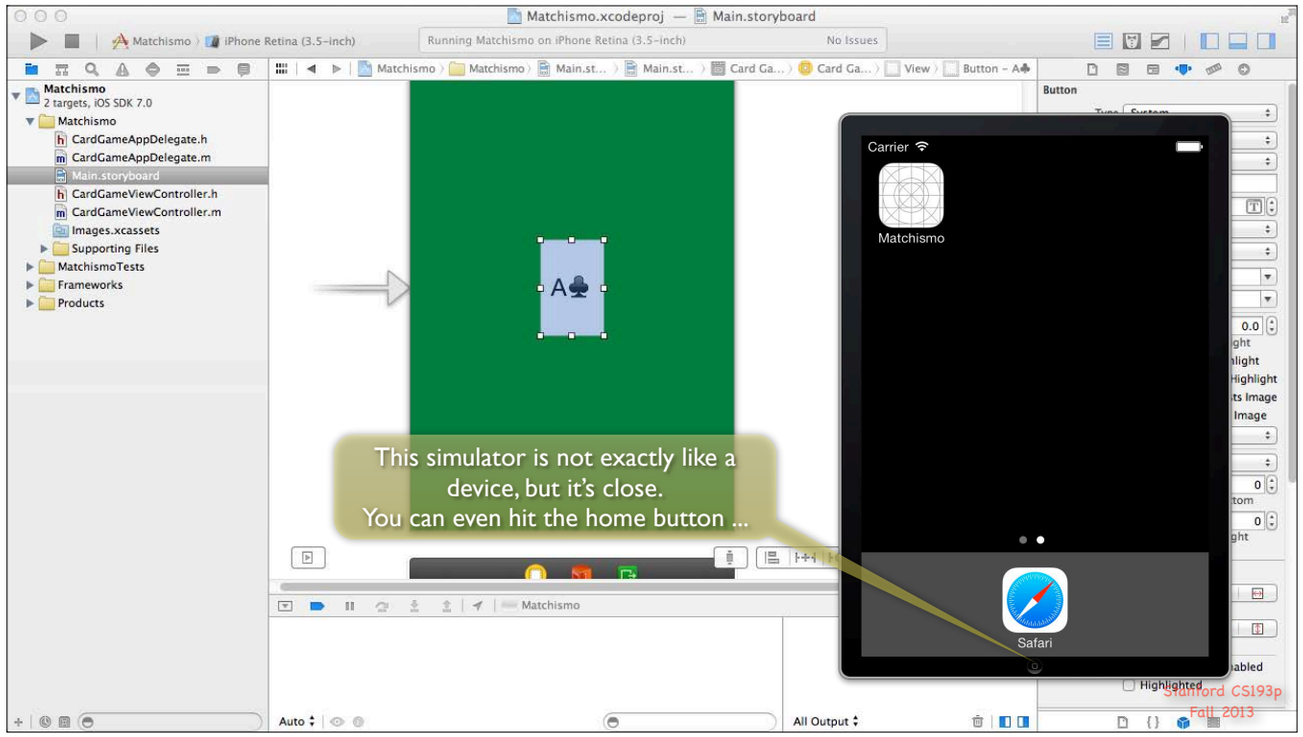

Matchismo.xcodeproj — Main.storyboard
Changing the text color is simple, just click on the Text Color control.
And choose Black.
Default
Recently Used Colors
Dark Text Color
Group Table View Background Color
Light Text Color
Scroll View Textured Background Color
Table Cell Grouped Background Color
View Flipside Background Color
Black Color
Dark Gray Color
Light Gray Color
White Color
Clear Color
Other...
Stanford CS193p
Fall 2013

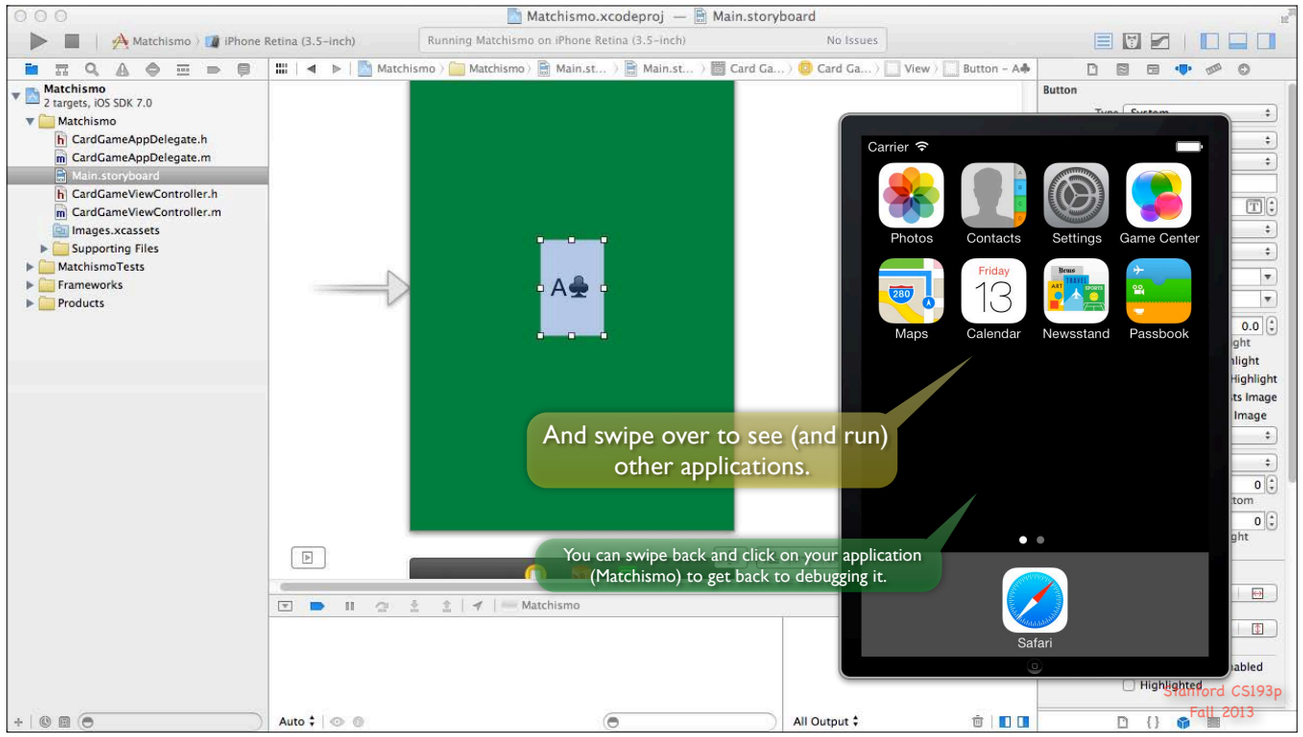

Our UI is built for 3.5-inch Retina iPhone, so you'll probably want to make sure that's set in this popup.
Let's Run our application for the first time and see what it looks like. Click on this "Play" button to run.
If you press and hold this Run button, other run options will be available, but we're just using plain "Run" for now.
Color is now black.
Stanford CS193p
Fall 2013

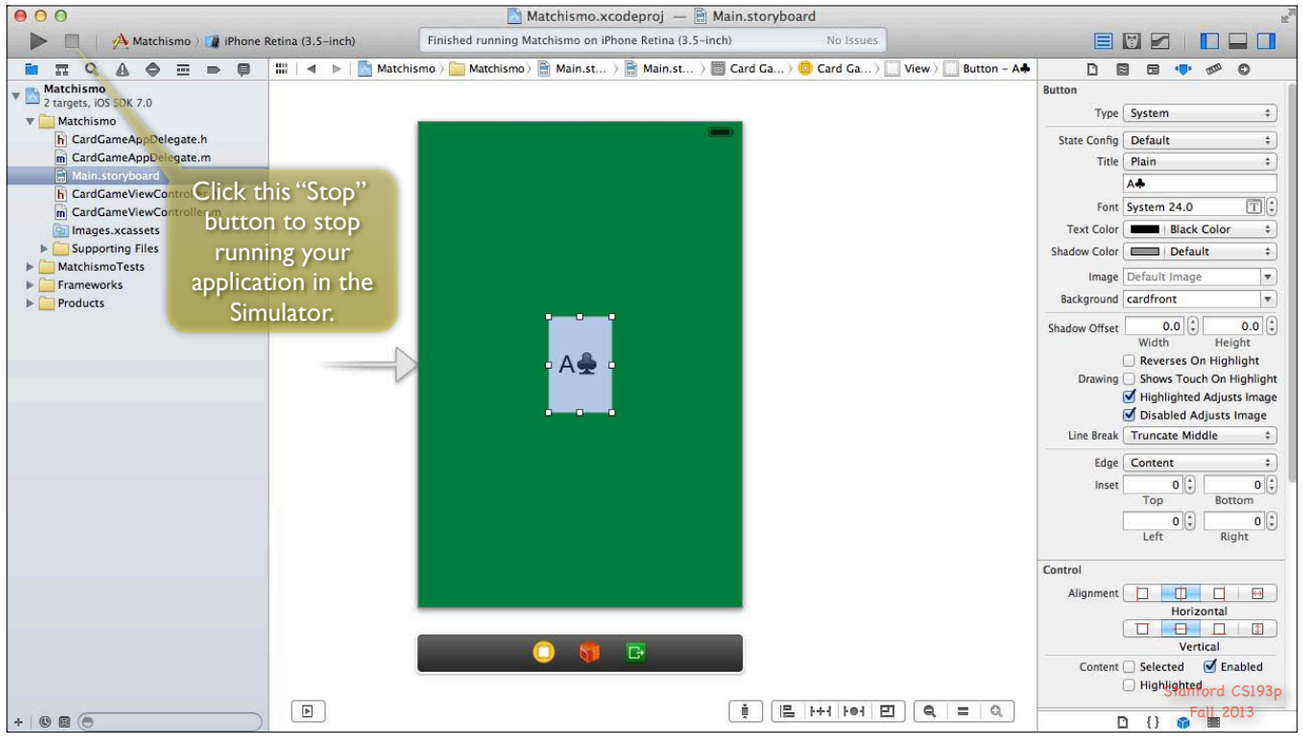

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Building Matchismo: Matchismo | Linking
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
As the application is built, you'll see status here.
A♣
Button
Type System
State Config Default
Title Plain
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

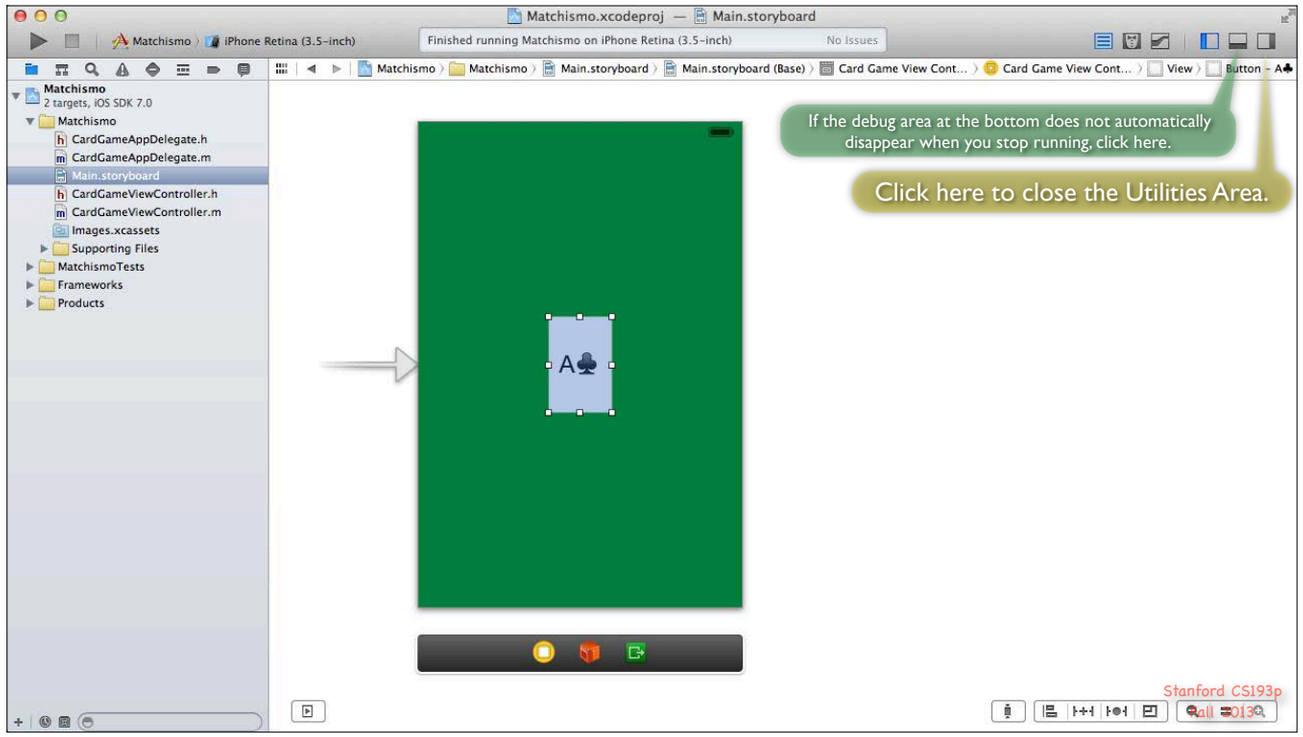

Matchismo.xcodeproj — Main.storyboard
Running Matchismo.app on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button
Type System
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset
Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
This button shows or hides the debugger/console.
If all goes well, you'll briefly see this.
Notice this area below automatically appears when you run your application.
It contains the debugger area and the console output.
We'll cover that later.
Build Succeeded
No Selection
Debugger
Console
This button shows or hides the debugger/console.
Auto
All Output
Stanford CS193p
Fall 2013

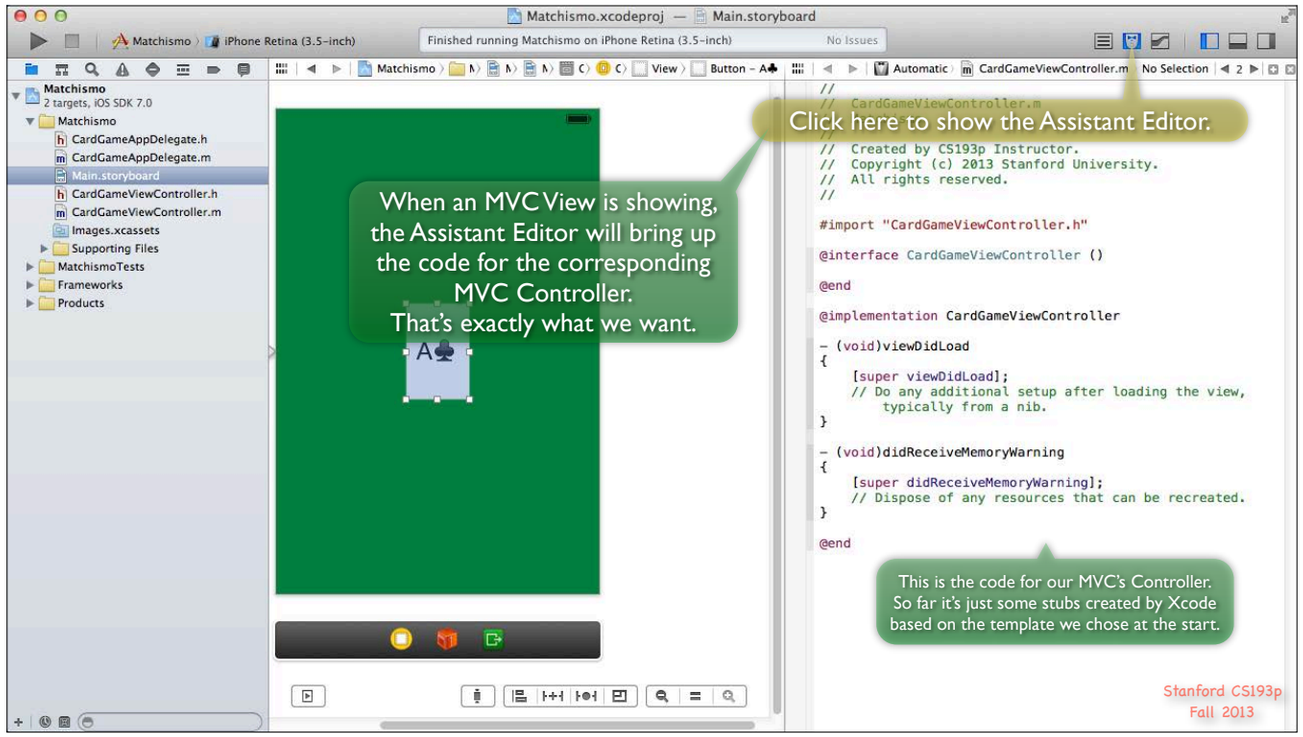

Matchismo.xcodeproj — Main.storyboard
Matchismo
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
A♣
A separate application, the iOS Simulator, will launch and your application will run.
Congratulations! You've built and run your first iOS app!
You can click on the button, but it will not change because we haven't coded what we want the button to do when we click on it.
Highlighted
Stanford CS193p Fall 2013

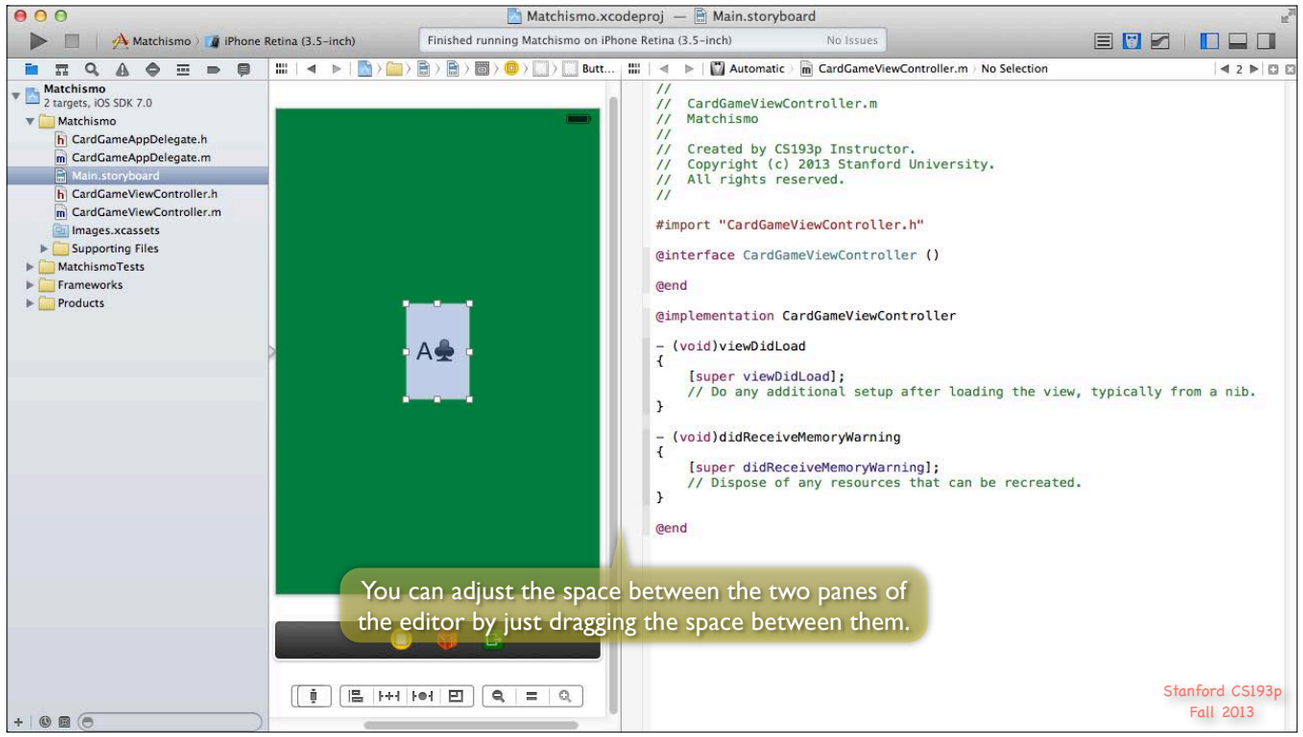

Matchismo.xcodeproj — Main.storyboard
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Carrier
Matchismo
Safari
This simulator is not exactly like a device, but it's close.
You can even hit the home button ...
Highlighted
Stanford CS193p
Fall 2013

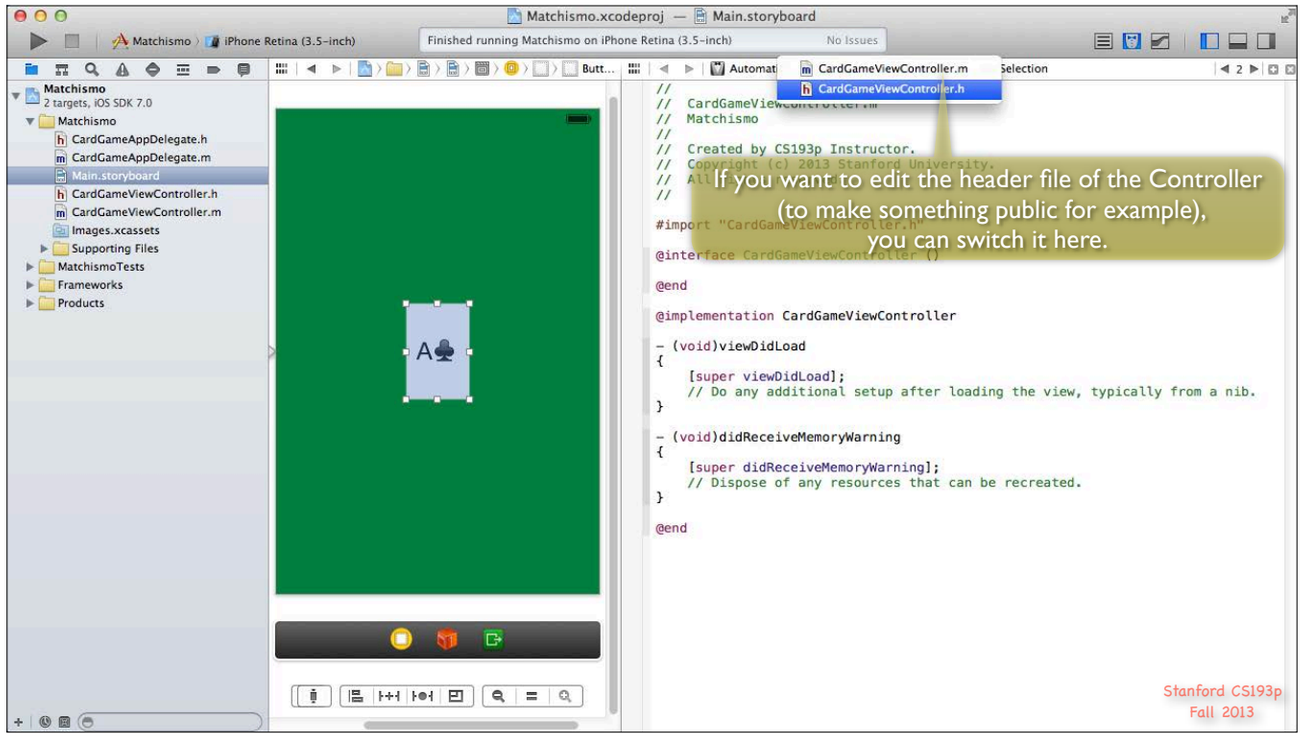

Matchismo.xcodeproj — Main.storyboard
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Carrier
Photos
Contacts
Settings
Game Center
Maps
Calendar
Newsstand
Passbook
Safari
And swipe over to see (and run) other applications.
You can swipe back and click on your application (Matchismo) to get back to debugging it.
Highlighted
Stanford CS193p
Fall 2013

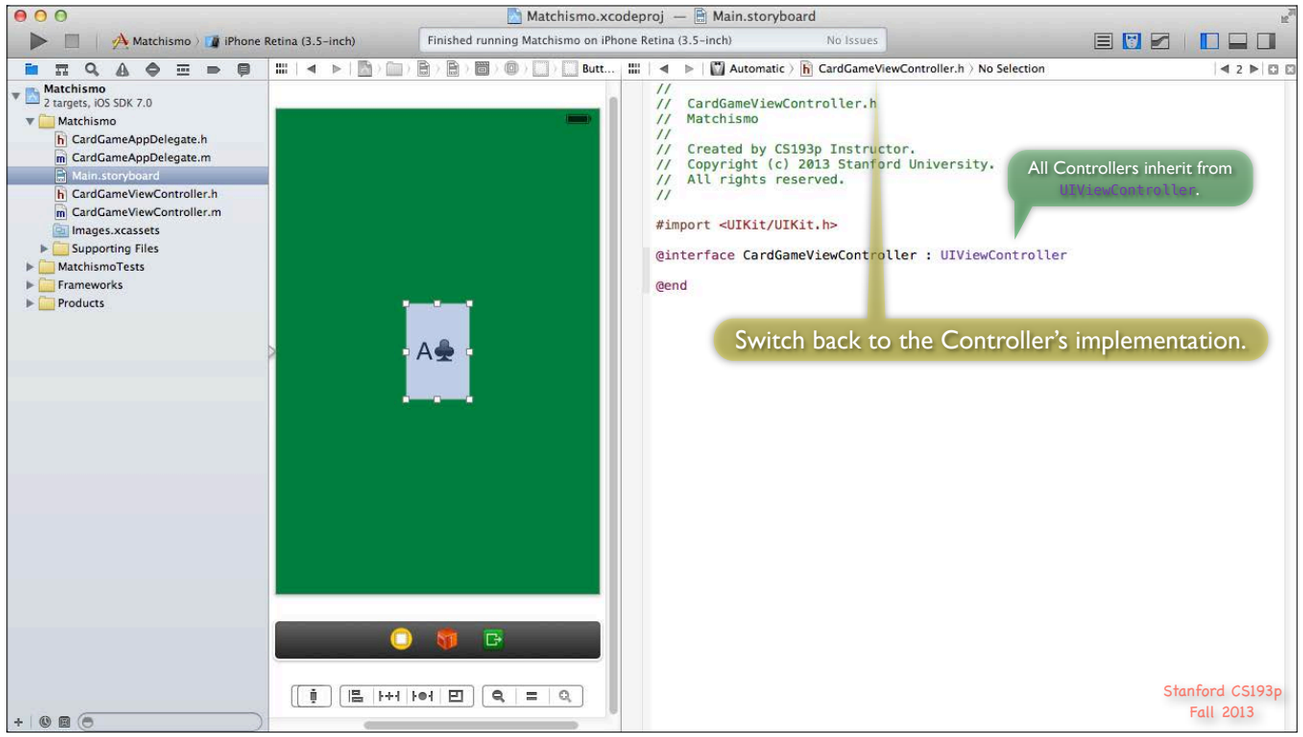

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewControl
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Click this "Stop" button to stop running your application in the Simulator.
A♣
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

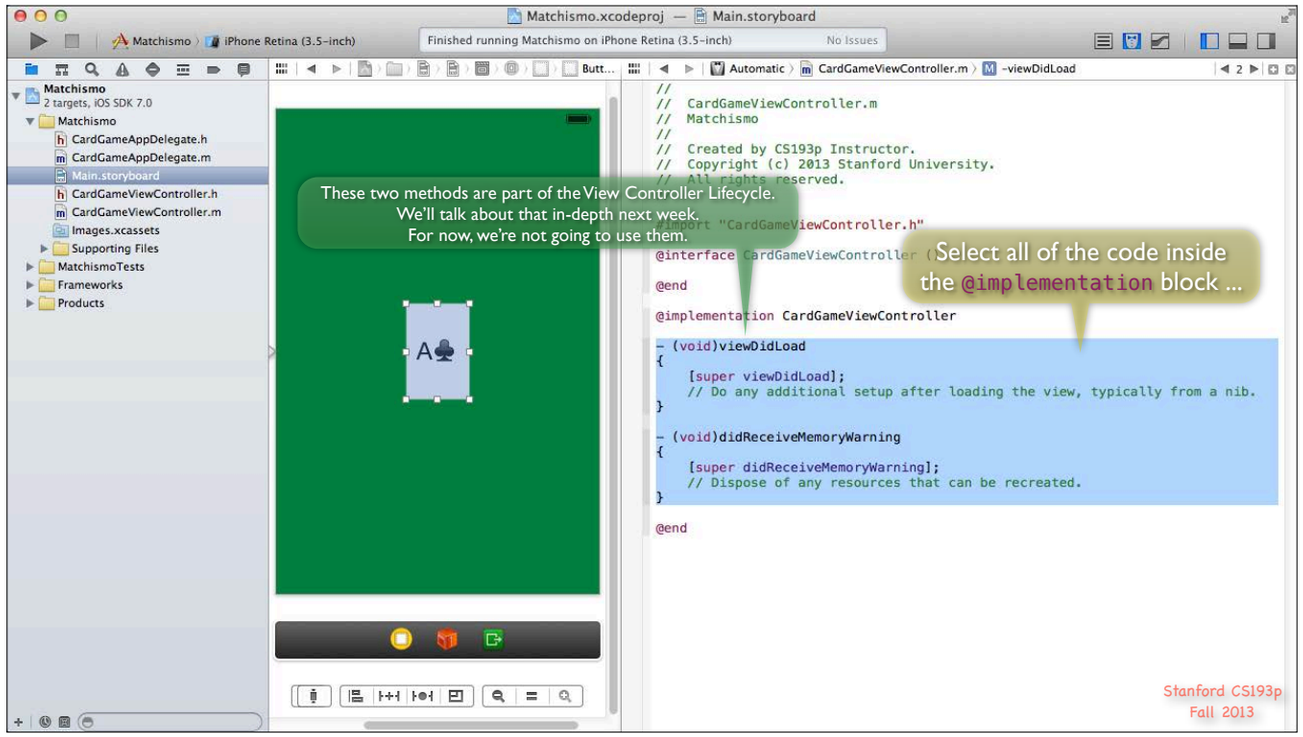

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Main.storyboard › Main.storyboard (Base) › Card Game View Cont… › Card Game View Cont… › View › Button – A♣
A♣
If the debug area at the bottom does not automatically disappear when you stop running, click here.
Click here to close the Utilities Area.
Stanford CS193p
Fall 2013

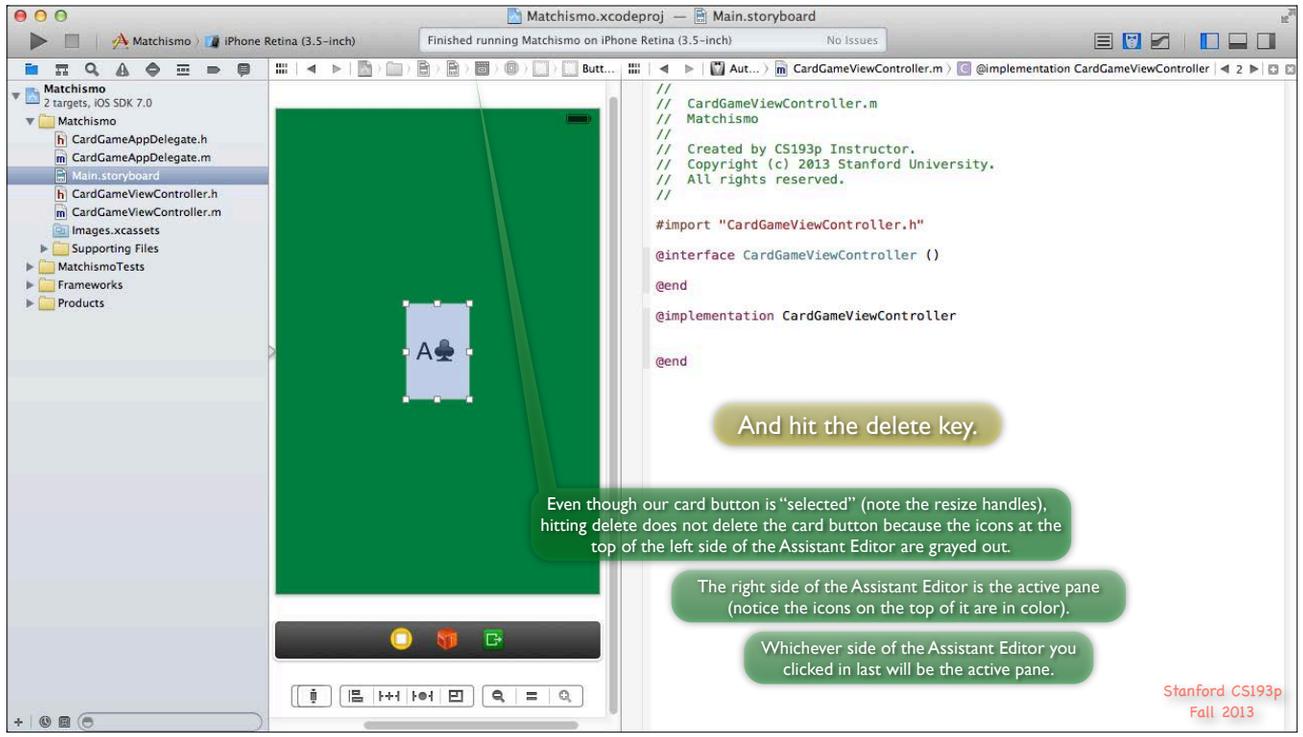

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › View › Button – A♣
Automatic › CardGameViewController.m › No Selection
Click here to show the Assistant Editor.
When an MVC View is showing, the Assistant Editor will bring up the code for the corresponding MVC Controller.
That's exactly what we want.
A♠
//
// CardGameViewController.m
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view,
        typically from a nib.
}
- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}
@end
This is the code for our MVC's Controller.
So far it's just some stubs created by Xcode based on the template we chose at the start.
Stanford CS193p
Fall 2013

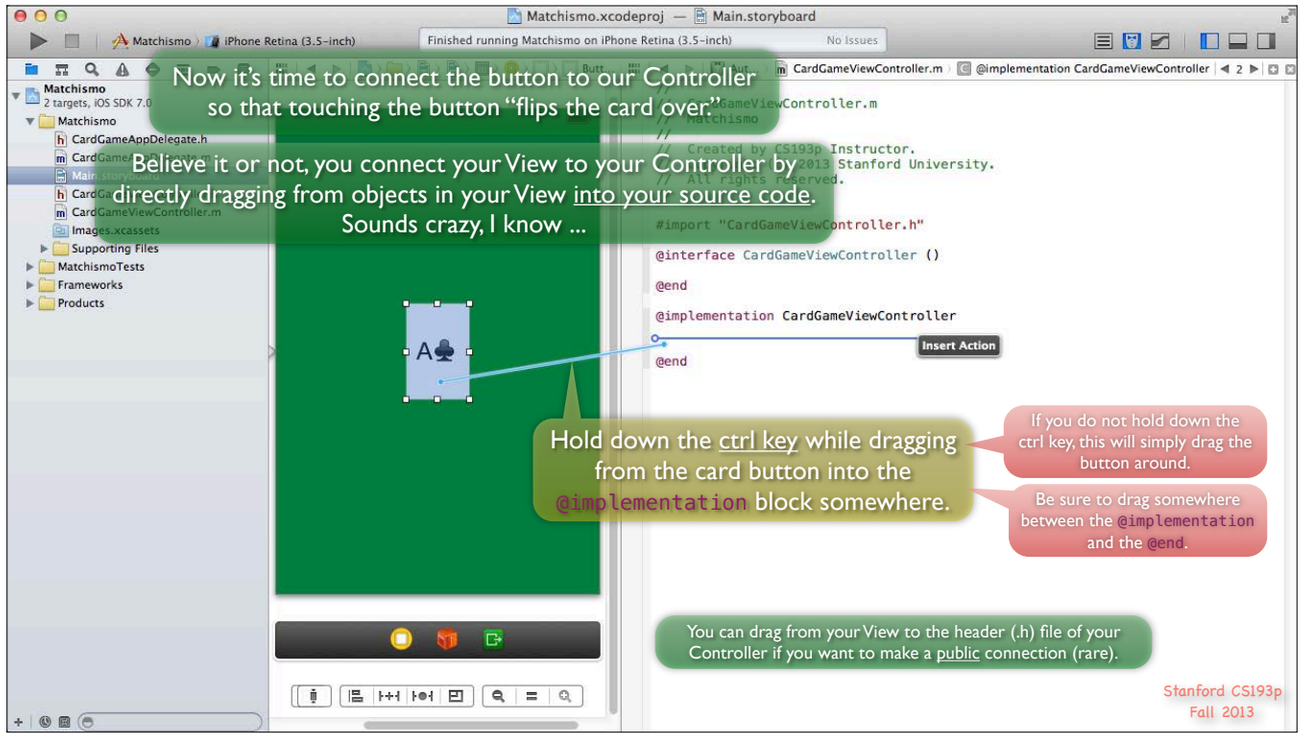

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic
CardGameViewController.m
No Selection
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (void)viewDidLoad
{
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
}
- (void)didReceiveMemoryWarning
{
[super didReceiveMemoryWarning];
// Dispose of any resources that can be recreated.
}
@end
You can adjust the space between the two panes of the editor by just dragging the space between them.
Stanford CS193p
Fall 2013

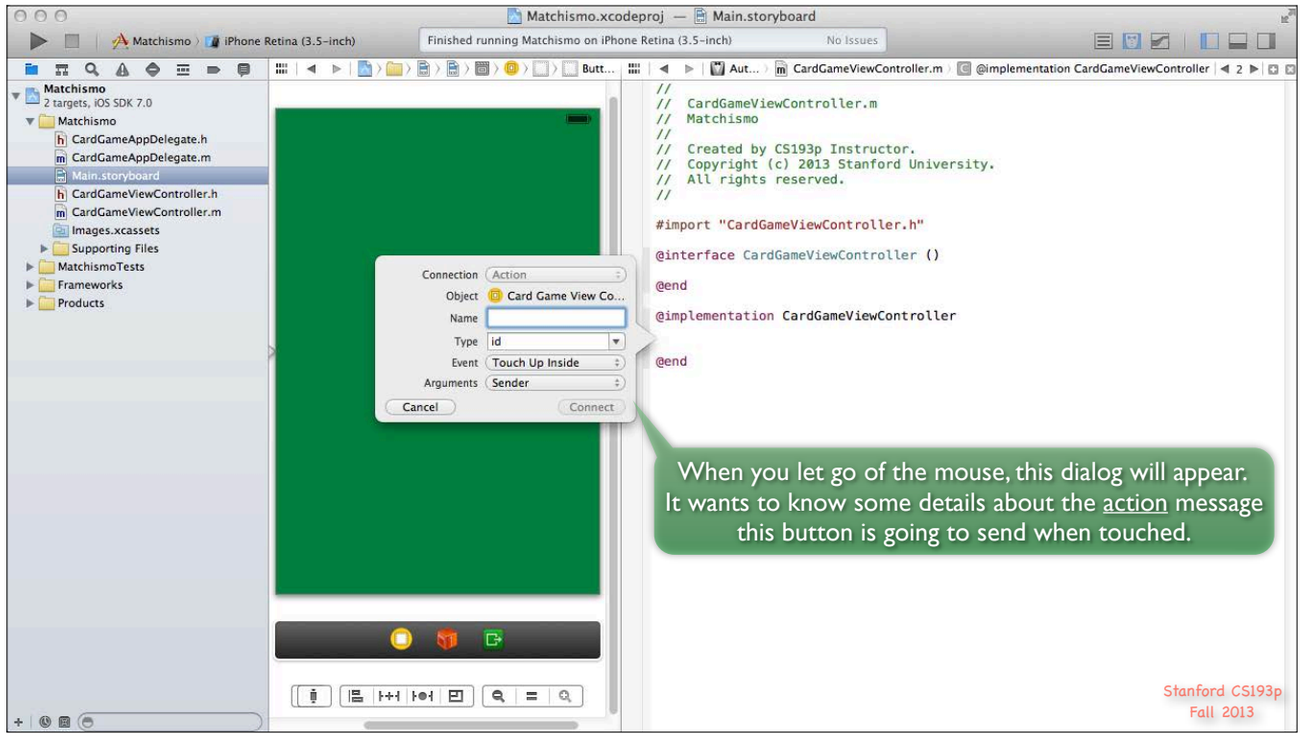

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
CardGameViewController.h
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
A♣
//
// CardGameViewController.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
}
- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}
@end
If you want to edit the header file of the Controller
(to make something public for example),
you can switch it here.
Stanford CS193p
Fall 2013

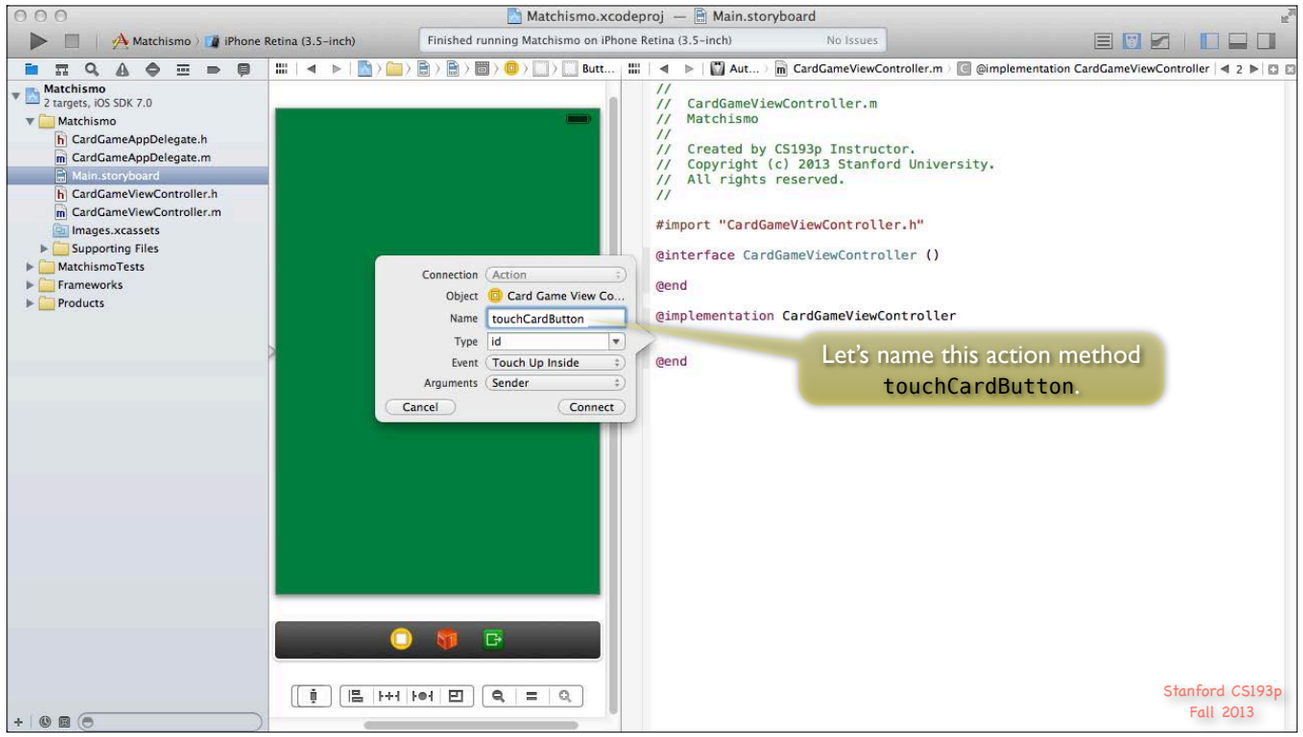

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.h › No Selection
//
// CardGameViewController.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <UIKit/UIKit.h>
@interface CardGameViewController : UIViewController
@end
All Controllers inherit from UIViewController.
Switch back to the Controller's implementation.
Stanford CS193p
Fall 2013

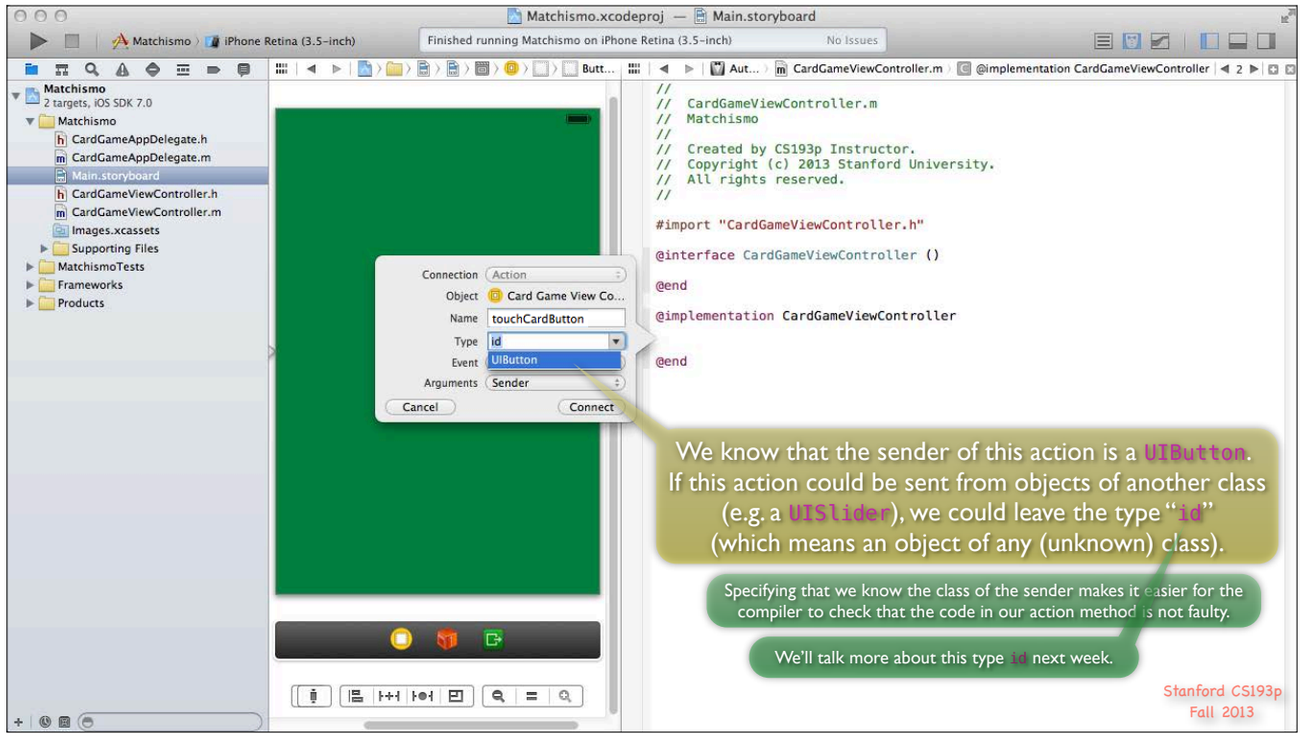

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -viewDidLoad
These two methods are part of the View Controller Lifecycle.
We'll talk about that in-depth next week.
For now, we're not going to use them.
Select all of the code inside the @implementation block ...
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (void)viewDidLoad
{
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
}
- (void)didReceiveMemoryWarning
{
[super didReceiveMemoryWarning];
// Dispose of any resources that can be recreated.
}
@end
Stanford CS193p
Fall 2013

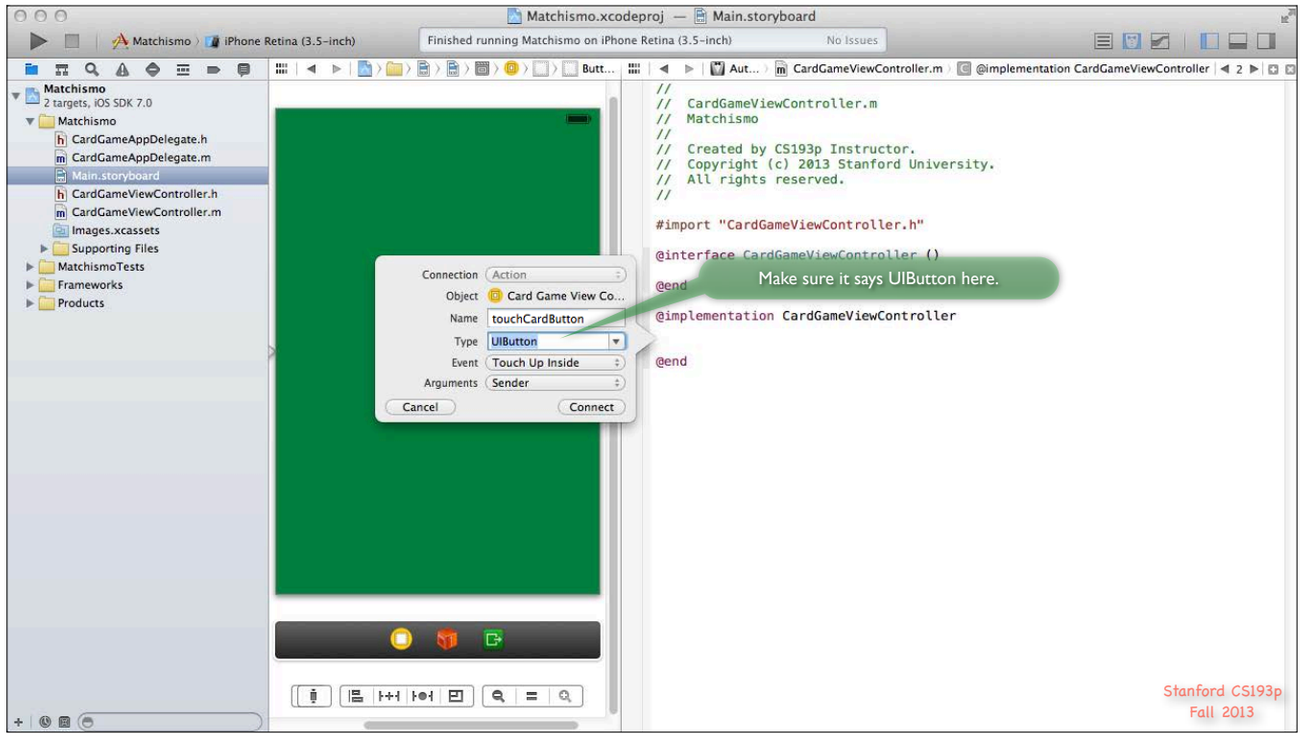

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
And hit the delete key.
Even though our card button is "selected" (note the resize handles), hitting delete does not delete the card button because the icons at the top of the left side of the Assistant Editor are grayed out.
The right side of the Assistant Editor is the active pane (notice the icons on the top of it are in color).
Whichever side of the Assistant Editor you clicked in last will be the active pane.
Stanford CS193p
Fall 2013

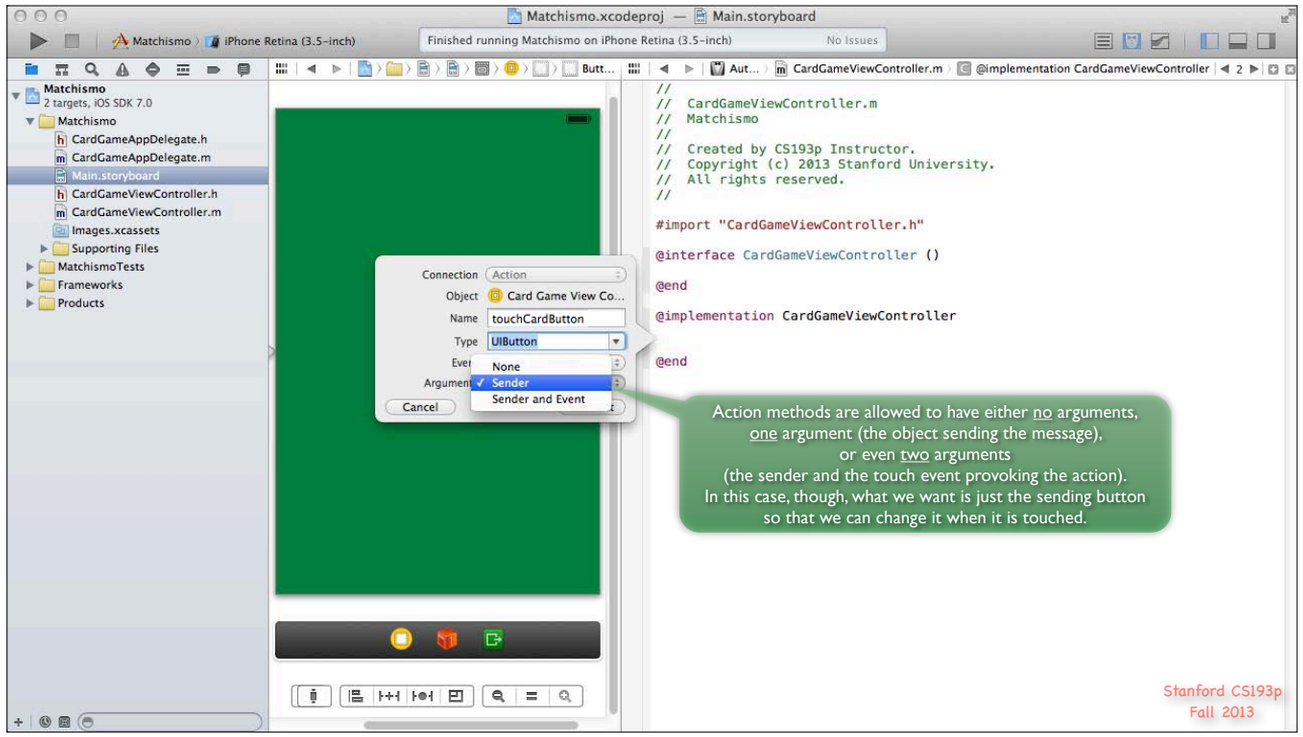

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
CardGameViewController.m › @implementation CardGameViewController
// CardGameViewController.m
// Matchismo
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
Insert Action
@end
Now it's time to connect the button to our Controller so that touching the button "flips the card over."
Believe it or not, you connect your View to your Controller by directly dragging from objects in your View into your source code. Sounds crazy, I know ...
Hold down the ctrl key while dragging from the card button into the @implementation block somewhere.
If you do not hold down the ctrl key, this will simply drag the button around.
Be sure to drag somewhere between the @implementation and the @end.
You can drag from your View to the header (.h) file of your Controller if you want to make a public connection (rare).
Stanford CS193p
Fall 2013

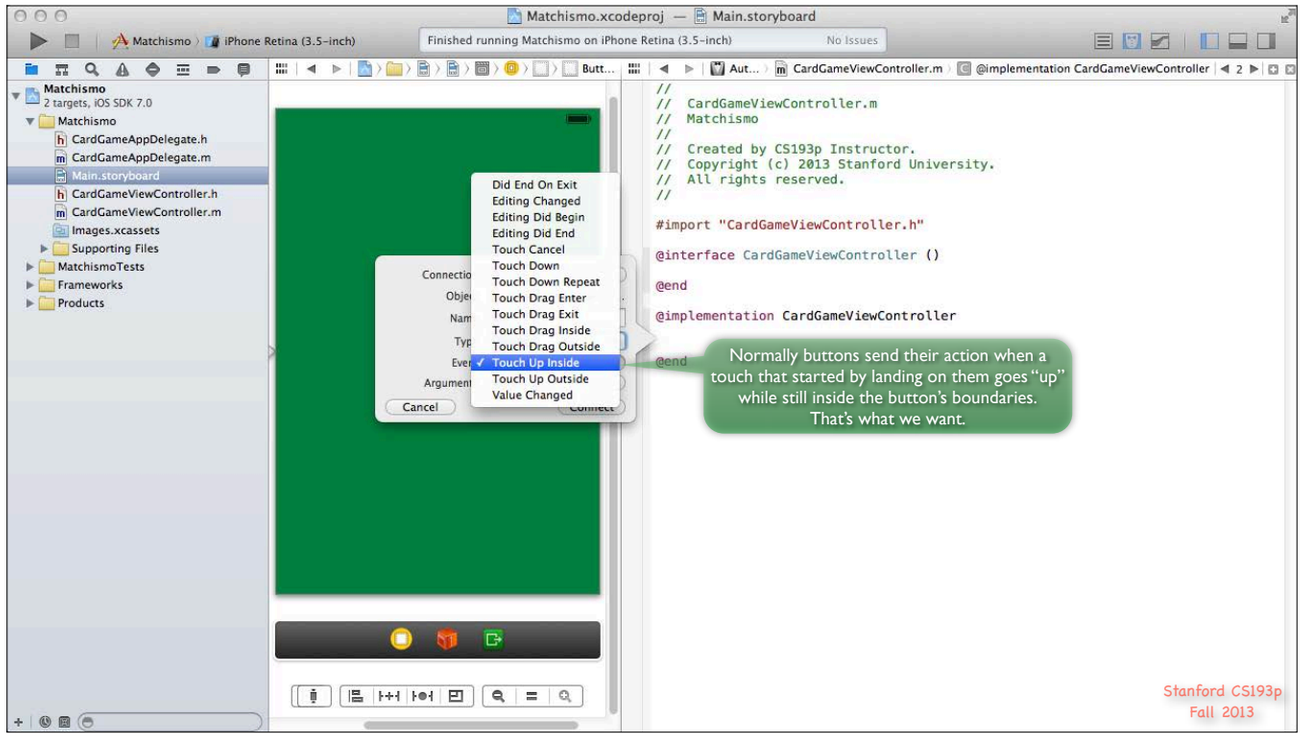

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name
Type id
Event Touch Up Inside
Arguments Sender
Cancel
Connect
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
When you let go of the mouse, this dialog will appear.
It wants to know some details about the action message
this button is going to send when touched.
Stanford CS193p
Fall 2013

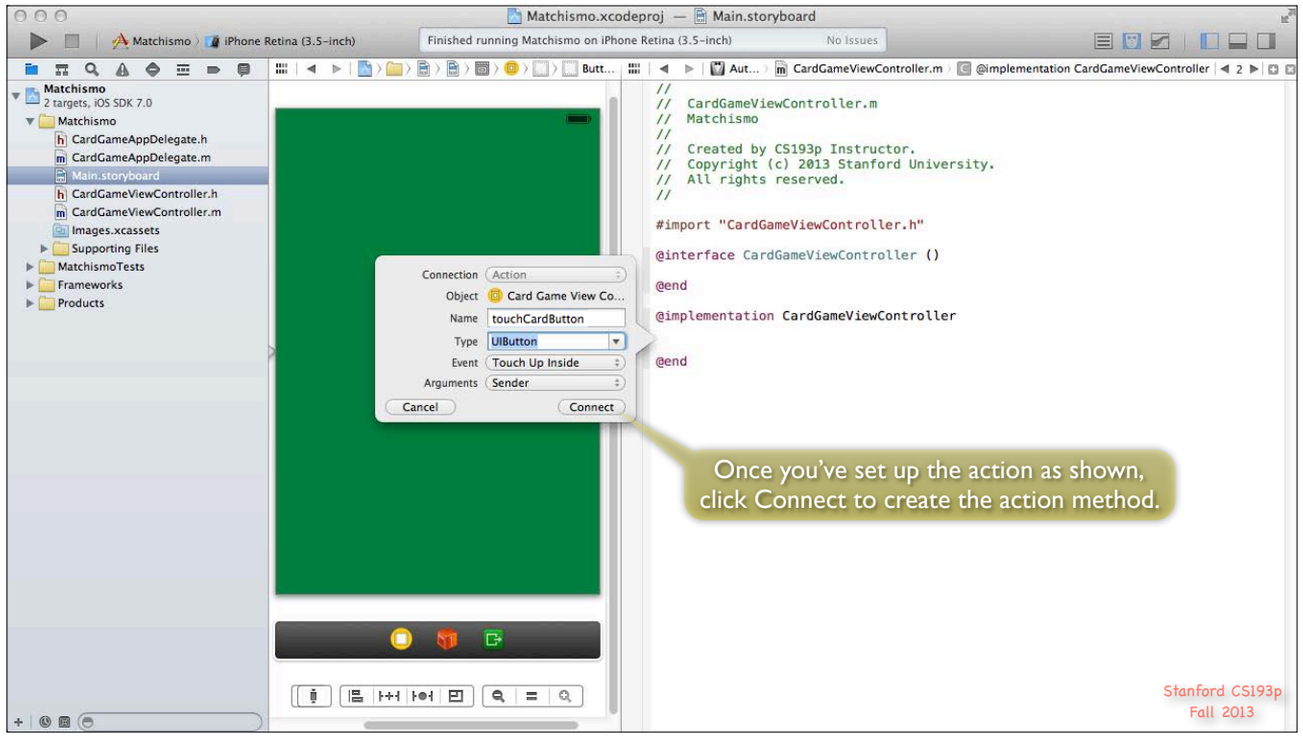

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name touchCardButton
Type id
Event Touch Up Inside
Arguments Sender
Cancel
Connect
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
Let's name this action method touchCardButton.
Stanford CS193p
Fall 2013

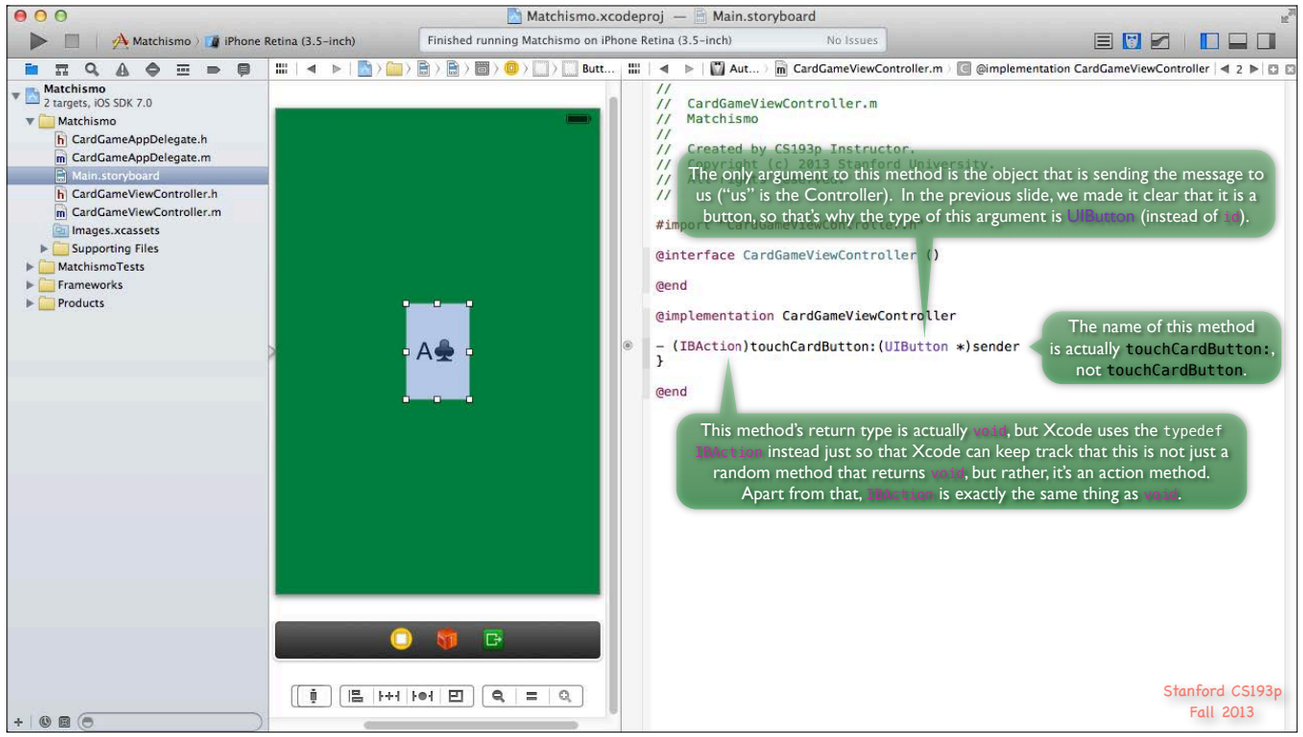

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name touchCardButton
Type id
Event UIButton
Arguments Sender
Cancel
Connect
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
We know that the sender of this action is a UIButton.
If this action could be sent from objects of another class
(e.g. a UISlider), we could leave the type "id"
(which means an object of any (unknown) class).
Specifying that we know the class of the sender makes it easier for the compiler to check that the code in our action method is not faulty.
We'll talk more about this type id next week.
Stanford CS193p
Fall 2013

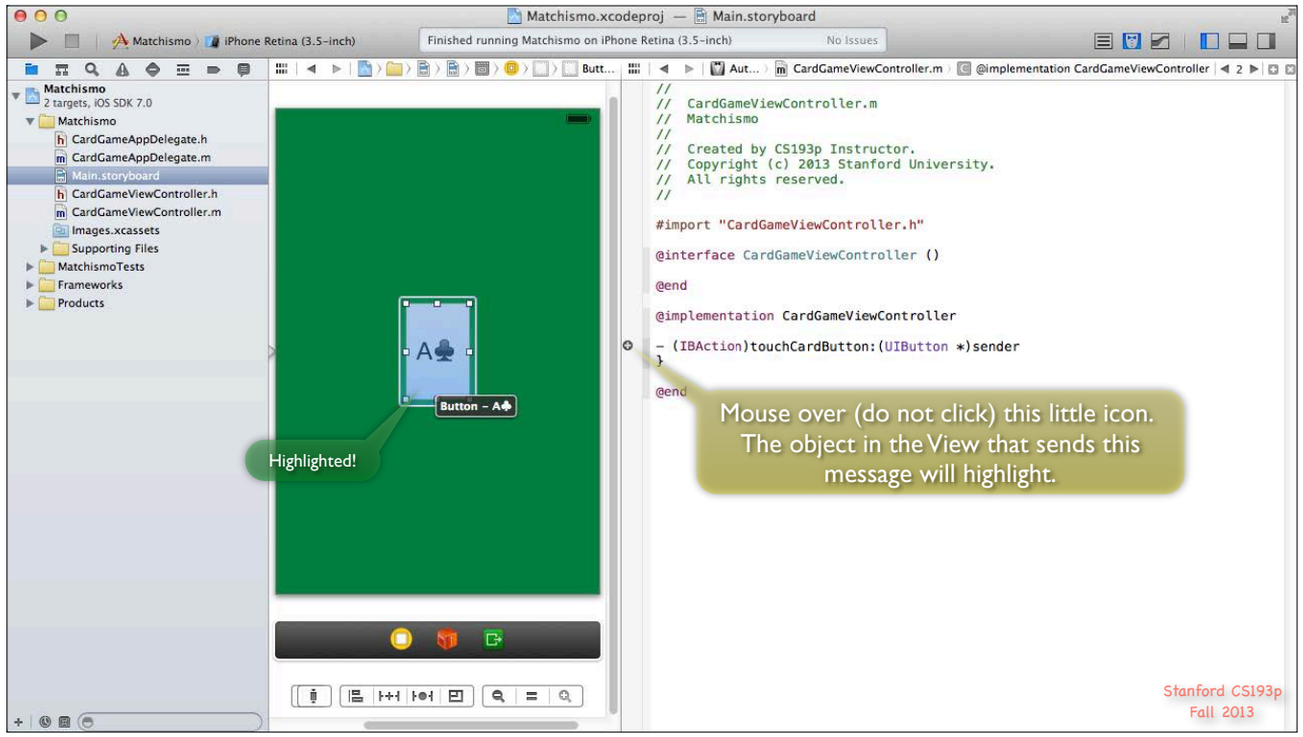

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name touchCardButton
Type UIButton
Event Touch Up Inside
Arguments Sender
Cancel
Connect
Make sure it says UIButton here.
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
Stanford CS193p
Fall 2013

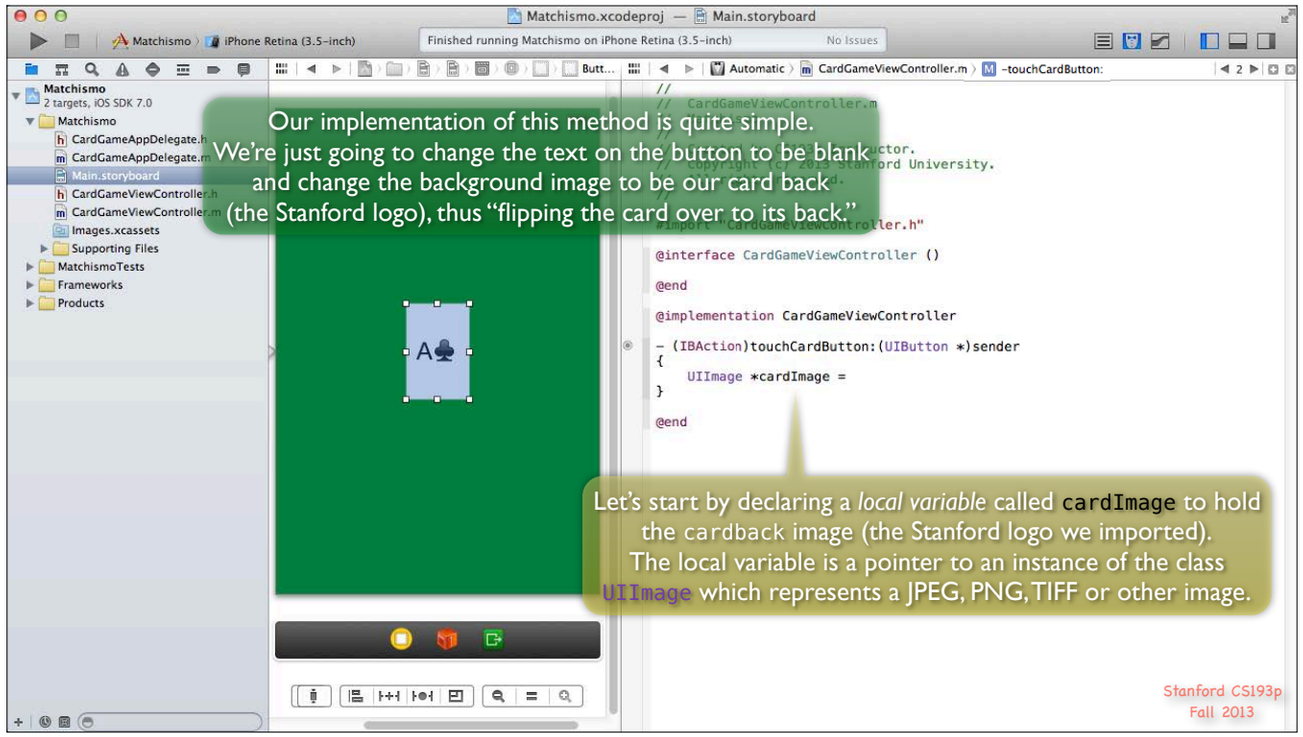

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name touchCardButton
Type UIButton
None
Sender
Sender and Event
Cancel
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
Action methods are allowed to have either no arguments,
one argument (the object sending the message),
or even two arguments
(the sender and the touch event provoking the action).
In this case, though, what we want is just the sending button
so that we can change it when it is touched.
Stanford CS193p
Fall 2013

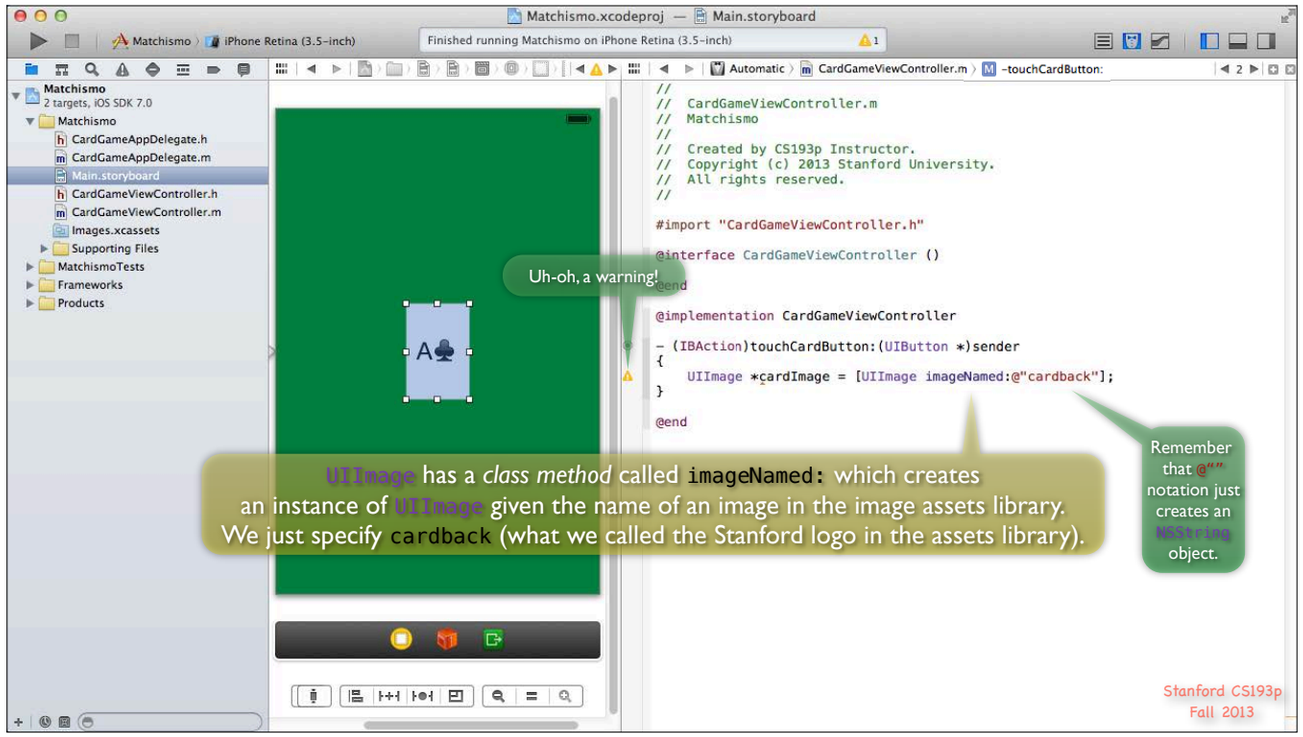

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Did End On Exit
Editing Changed
Editing Did Begin
Editing Did End
Touch Cancel
Touch Down
Touch Down Repeat
Touch Drag Enter
Touch Drag Exit
Touch Drag Inside
Touch Drag Outside
Touch Up Inside
Touch Up Outside
Value Changed
Cancel
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
Normally buttons send their action when a touch that started by landing on them goes "up" while still inside the button's boundaries. That's what we want.
Stanford CS193p
Fall 2013

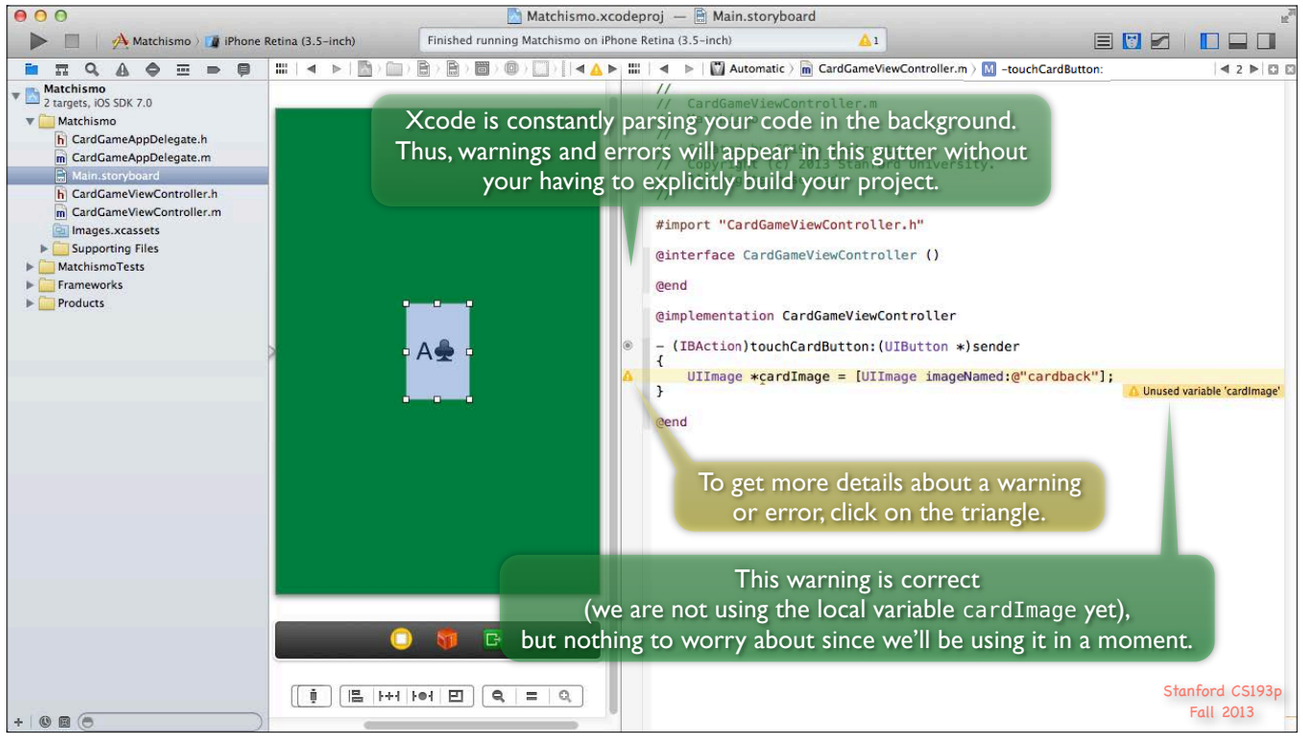

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Connection Action
Object Card Game View Co...
Name touchCardButton
Type UIButton
Event Touch Up Inside
Arguments Sender
Cancel
Connect
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
@end
Once you've set up the action as shown, click Connect to create the action method.
Stanford CS193p
Fall 2013

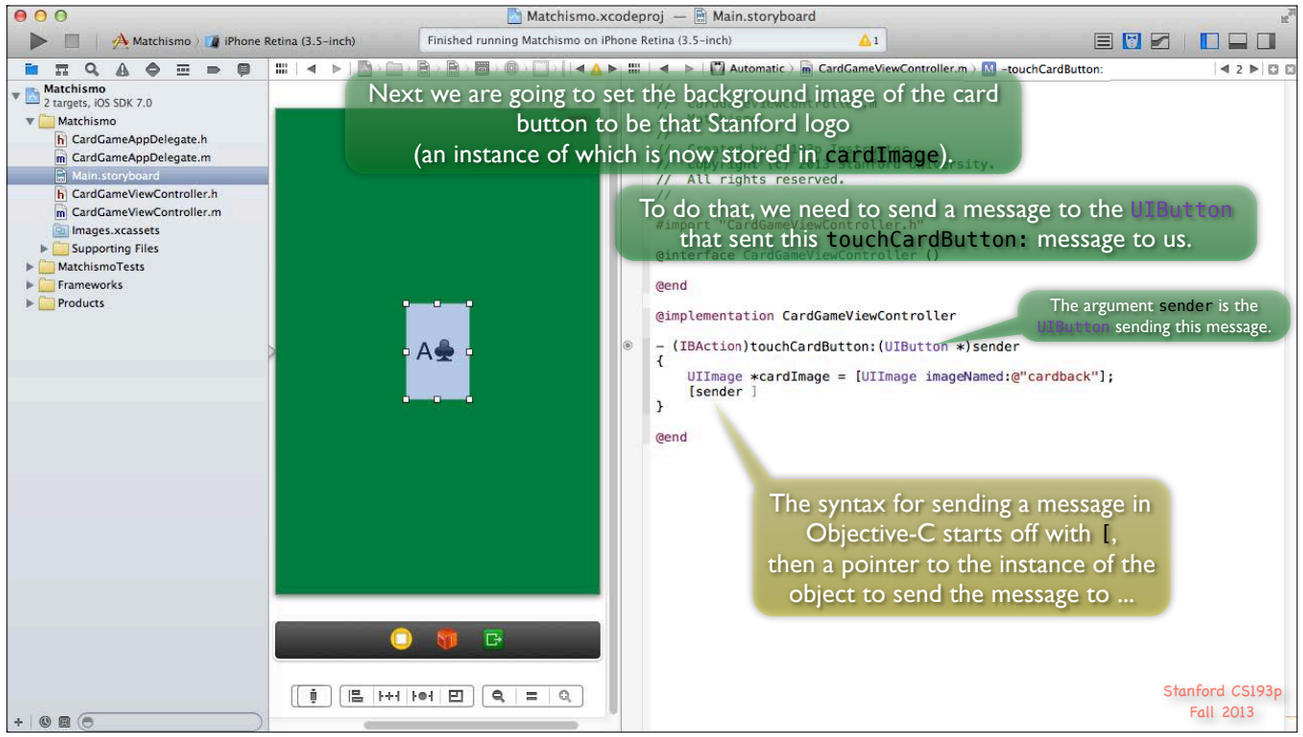

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
//
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
}
@end
The only argument to this method is the object that is sending the message to us ("us" is the Controller). In the previous slide, we made it clear that it is a button, so that's why the type of this argument is UIButton (instead of id).
The name of this method is actually touchCardButton:, not touchCardButton.
This method's return type is actually void, but Xcode uses the typedef IBAction instead just so that Xcode can keep track that this is not just a random method that returns void, but rather, it's an action method. Apart from that, IBAction is exactly the same thing as void.
Stanford CS193p
Fall 2013

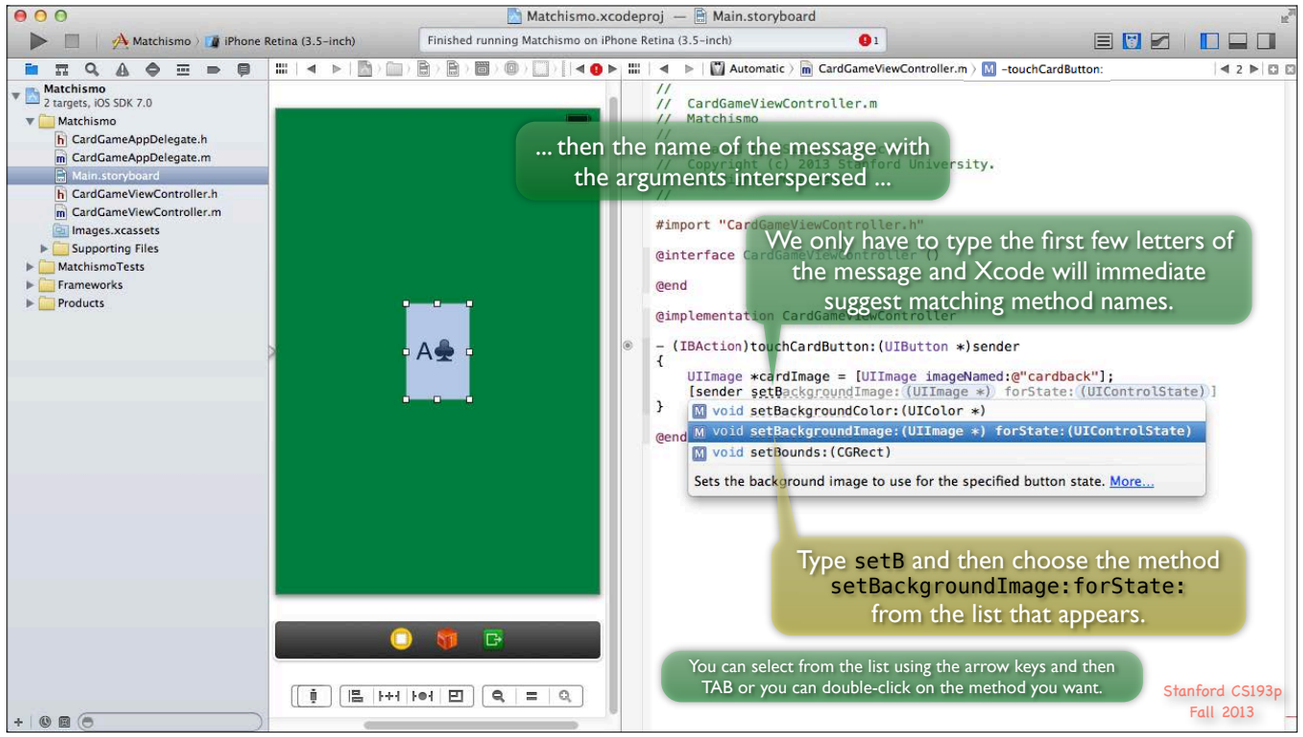

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Button – A♣
Highlighted!
CardGameViewController.m
@implementation CardGameViewController
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
}
@end
Mouse over (do not click) this little icon.
The object in the View that sends this
message will highlight.
Stanford CS193p
Fall 2013

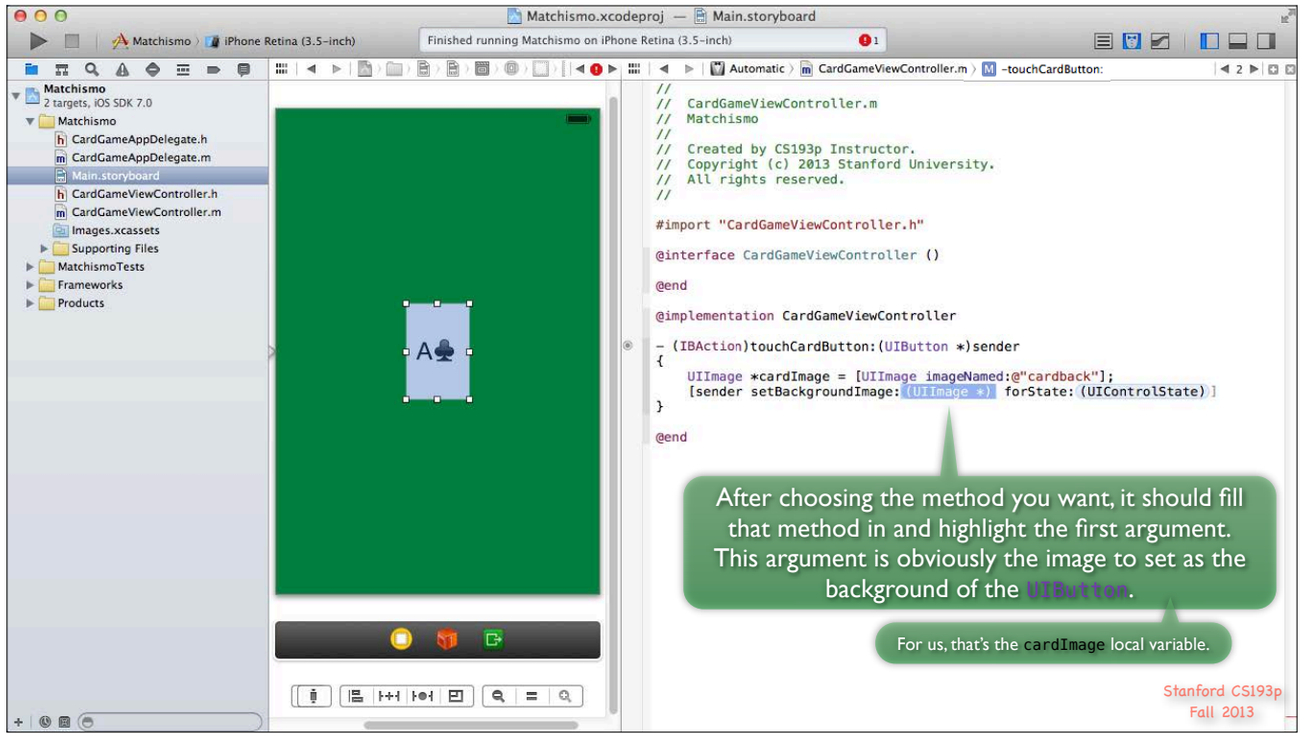

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Copyright (c) 2013 Stanford University.
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage =
}
@end
Our implementation of this method is quite simple.
We're just going to change the text on the button to be blank
and change the background image to be our card back
(the Stanford logo), thus "flipping the card over to its back."
Let's start by declaring a local variable called cardImage to hold
the cardback image (the Stanford logo we imported).
The local variable is a pointer to an instance of the class
UIImage which represents a JPEG, PNG, TIFF or other image.
Stanford CS193p
Fall 2013

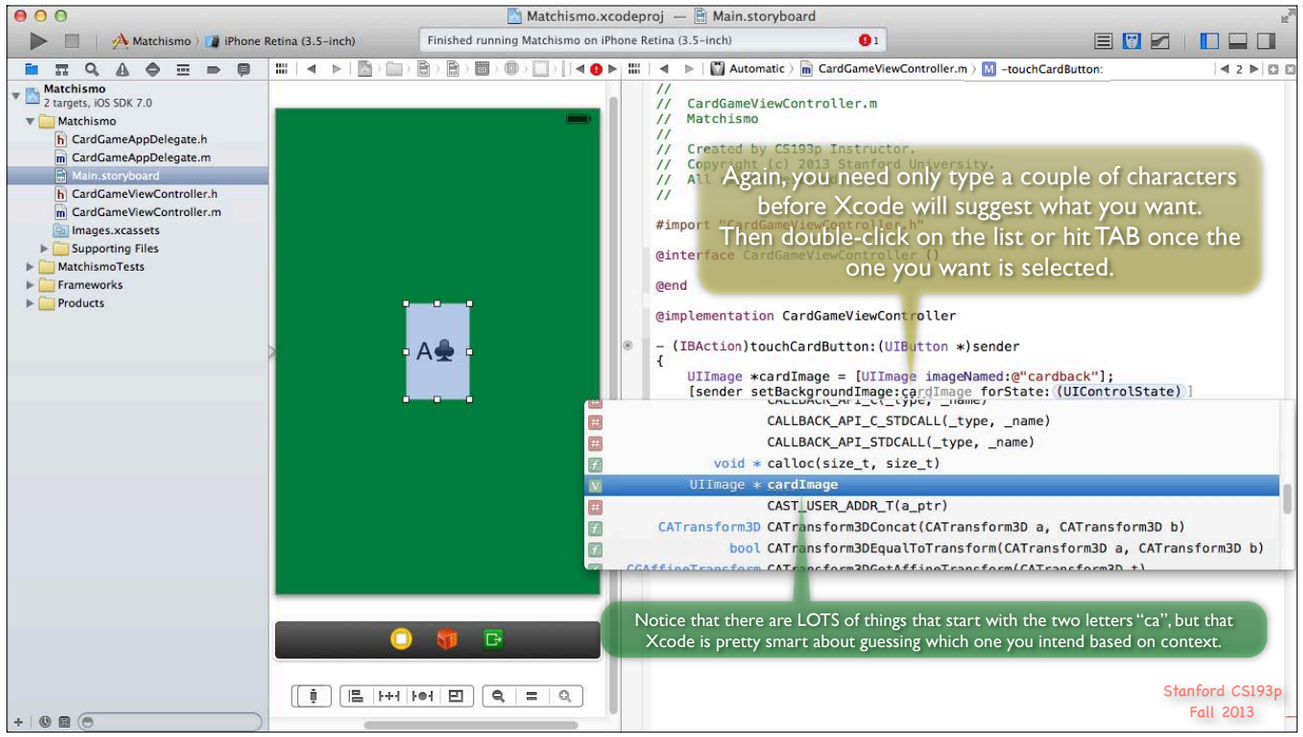

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
}
@end
Uh-oh, a warning!
UIImage has a class method called imageNamed: which creates an instance of UIImage given the name of an image in the image assets library. We just specify cardback (what we called the Stanford logo in the assets library).
Remember that @"" notation just creates an NSString object.
Stanford CS193p
Fall 2013

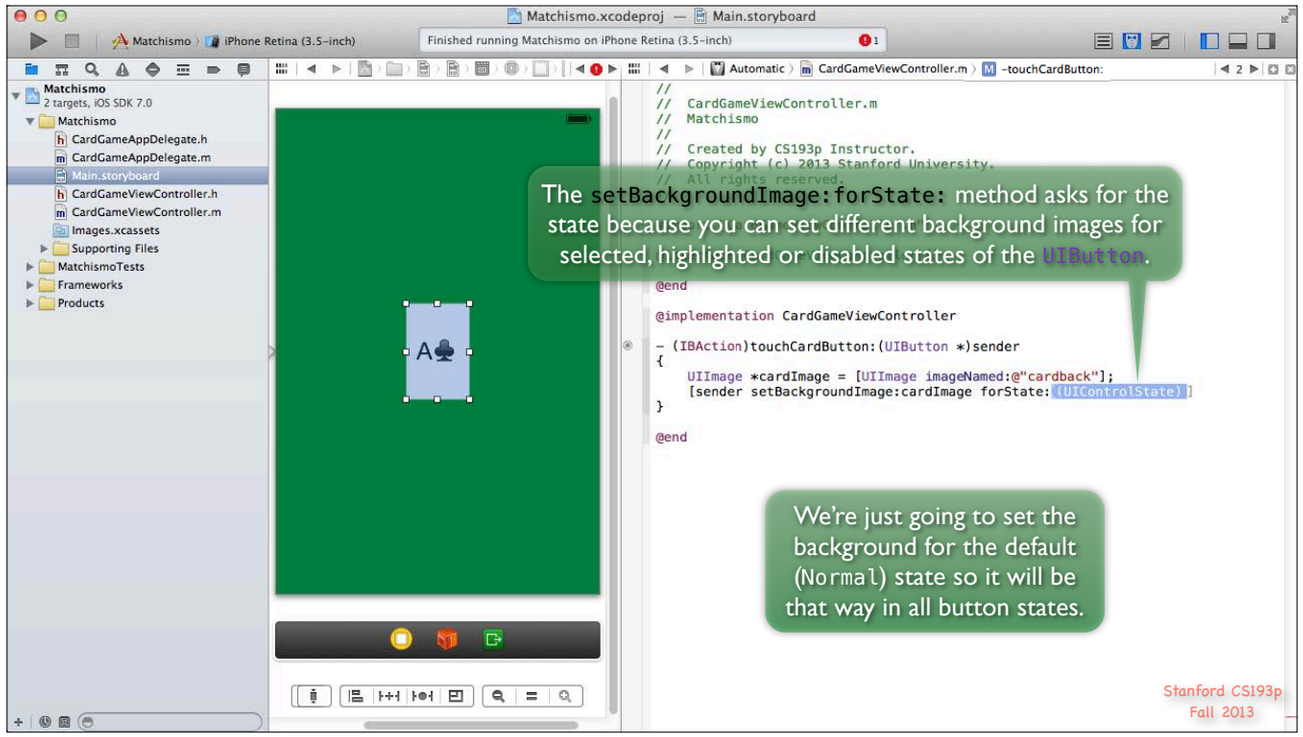

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
Xcode is constantly parsing your code in the background. Thus, warnings and errors will appear in this gutter without your having to explicitly build your project.
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
}
@end
Unused variable 'cardImage'
To get more details about a warning or error, click on the triangle.
This warning is correct (we are not using the local variable cardImage yet), but nothing to worry about since we'll be using it in a moment.
Stanford CS193p
Fall 2013

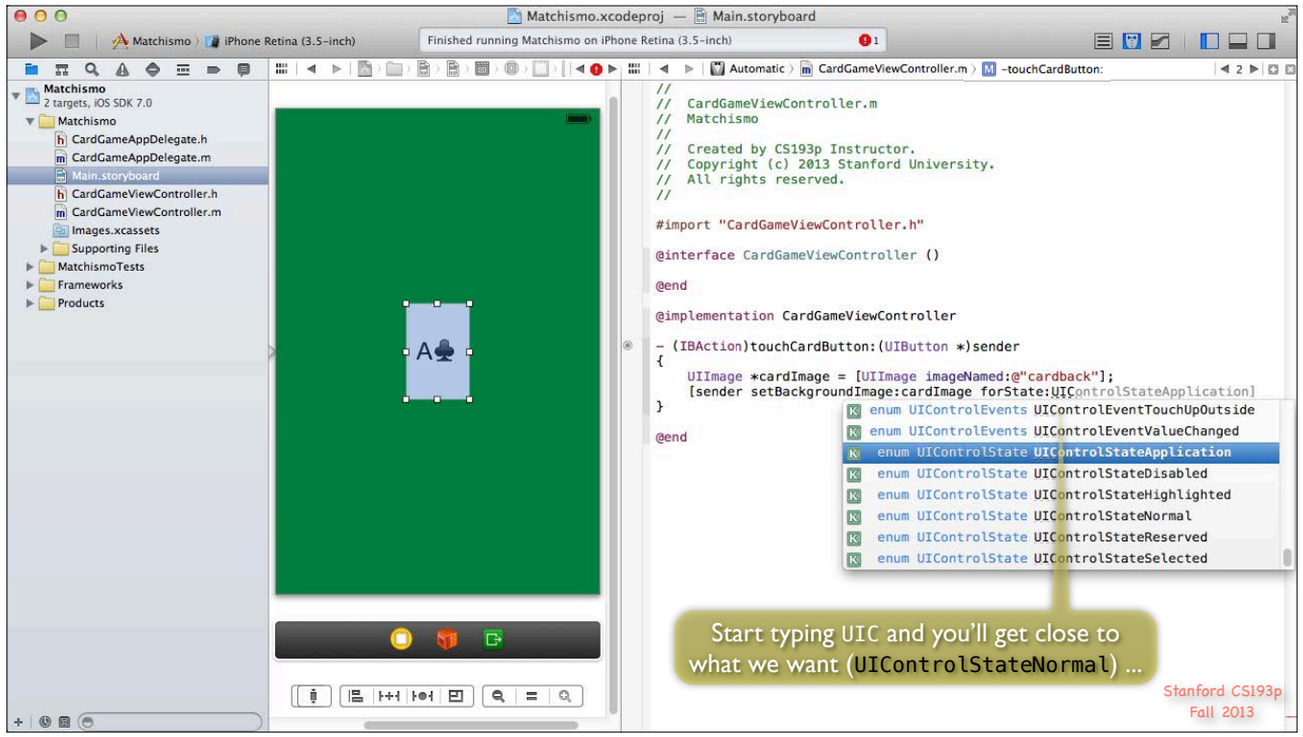

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
Next we are going to set the background image of the card button to be that Stanford logo (an instance of which is now stored in cardImage).
To do that, we need to send a message to the UIButton that sent this touchCardButton: message to us.
The argument sender is the UIButton sending this message.
// All rights reserved.
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender]
}
@end
The syntax for sending a message in Objective-C starts off with [, then a pointer to the instance of the object to send the message to ...
Stanford CS193p
Fall 2013

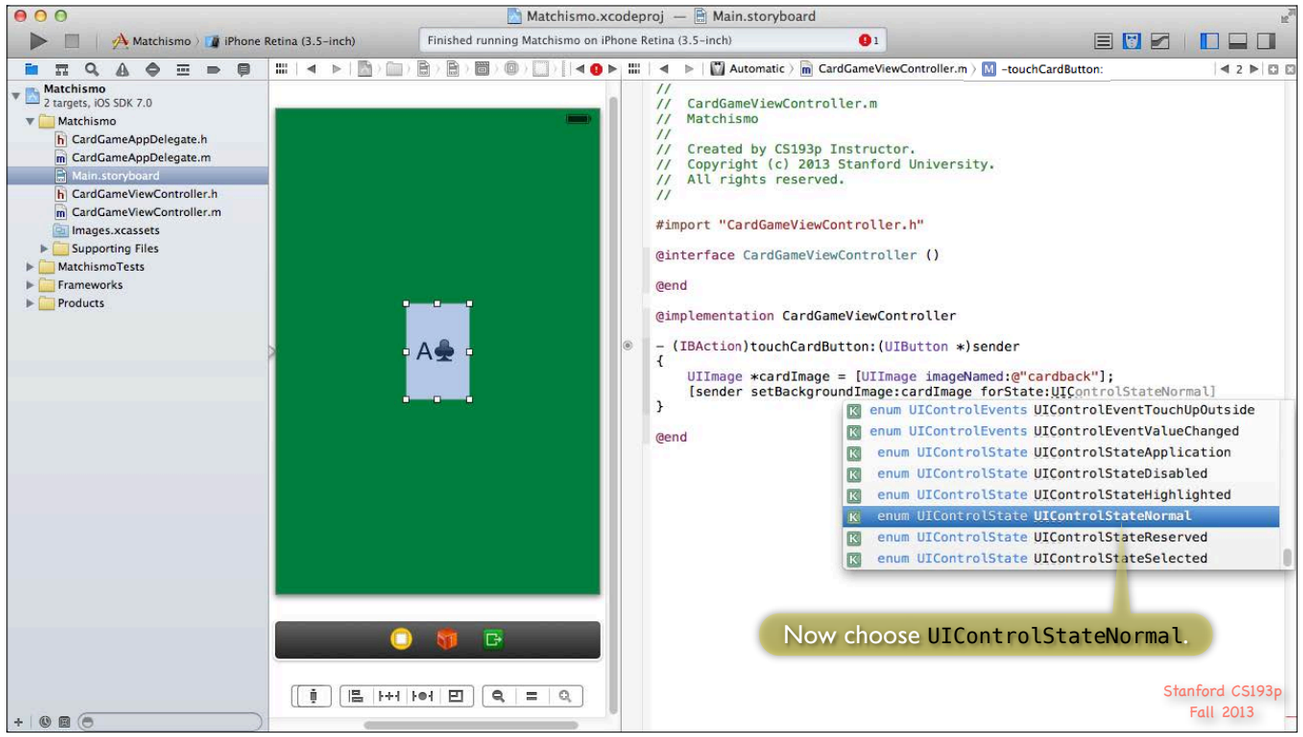

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
... then the name of the message with the arguments interspersed ...
We only have to type the first few letters of the message and Xcode will immediate suggest matching method names.
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setB
}
void setBackgroundColor:(UIColor *)
void setBackgroundImage:(UIImage *) forState:(UIControlState)
void setBounds:(CGRect)
Sets the background image to use for the specified button state. More...
Type setB and then choose the method setBackgroundImage:forState: from the list that appears.
You can select from the list using the arrow keys and then TAB or you can double-click on the method you want.
Stanford CS193p
Fall 2013

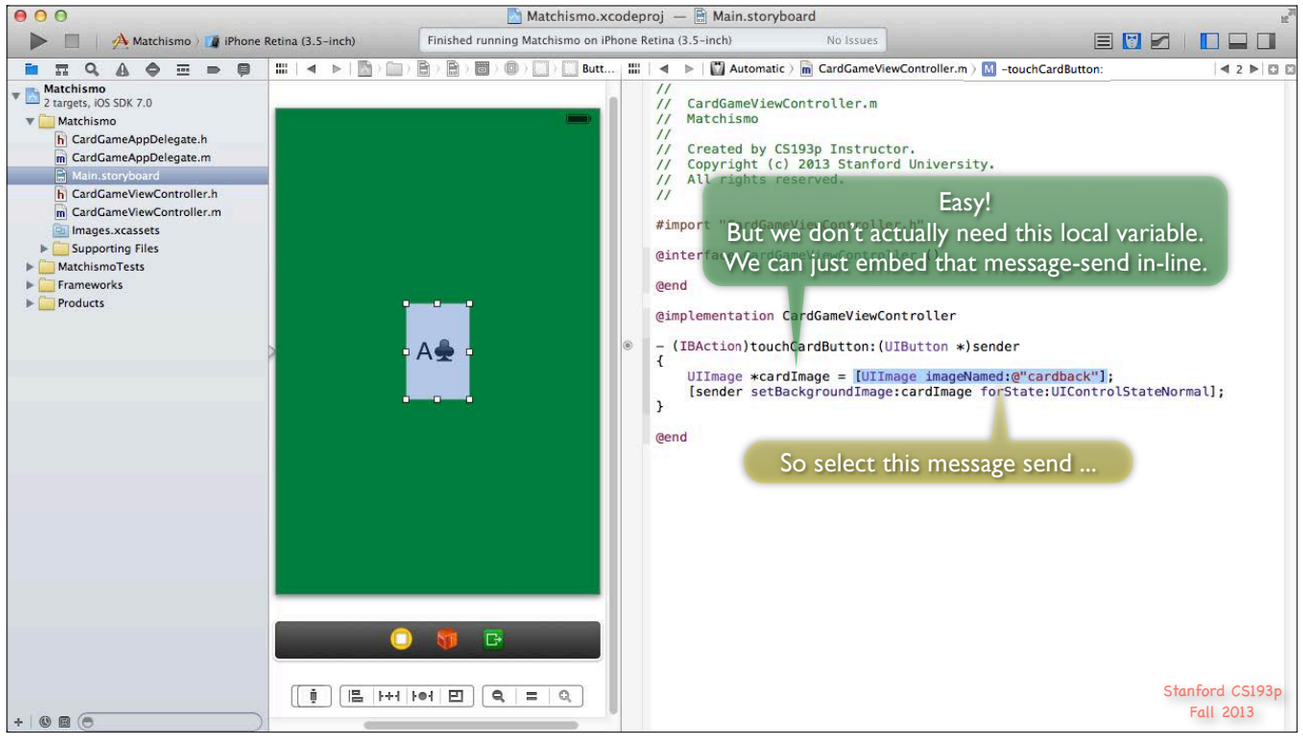

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:(UIImage *) forState:(UIControlState)]
}
@end
After choosing the method you want, it should fill that method in and highlight the first argument. This argument is obviously the image to set as the background of the UIButton.
For us, that's the cardImage local variable.
Stanford CS193p
Fall 2013

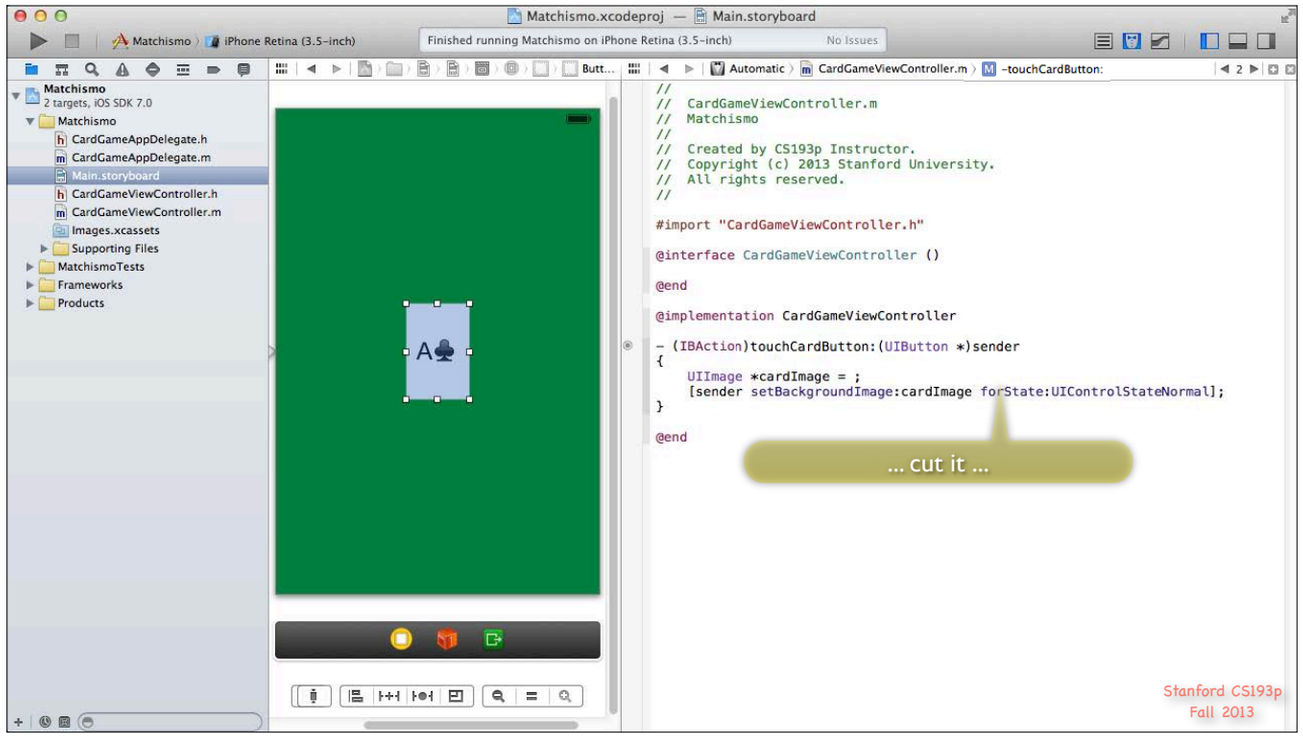

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:cardImage forState:(UIControlState)]
CALLBACK_API_C_STDCALL(_type, _name)
CALLBACK_API_STDCALL(_type, _name)
void * calloc(size_t, size_t)
UIImage * cardImage
CAST_USER_ADDR_T(a_ptr)
CATransform3D CATransform3DConcat(CATransform3D a, CATransform3D b)
bool CATransform3DEqualToTransform(CATransform3D a, CATransform3D b)
Again, you need only type a couple of characters before Xcode will suggest what you want. Then double-click on the list or hit TAB once the one you want is selected.
Notice that there are LOTS of things that start with the two letters "ca", but that Xcode is pretty smart about guessing which one you intend based on context.
Stanford CS193p
Fall 2013

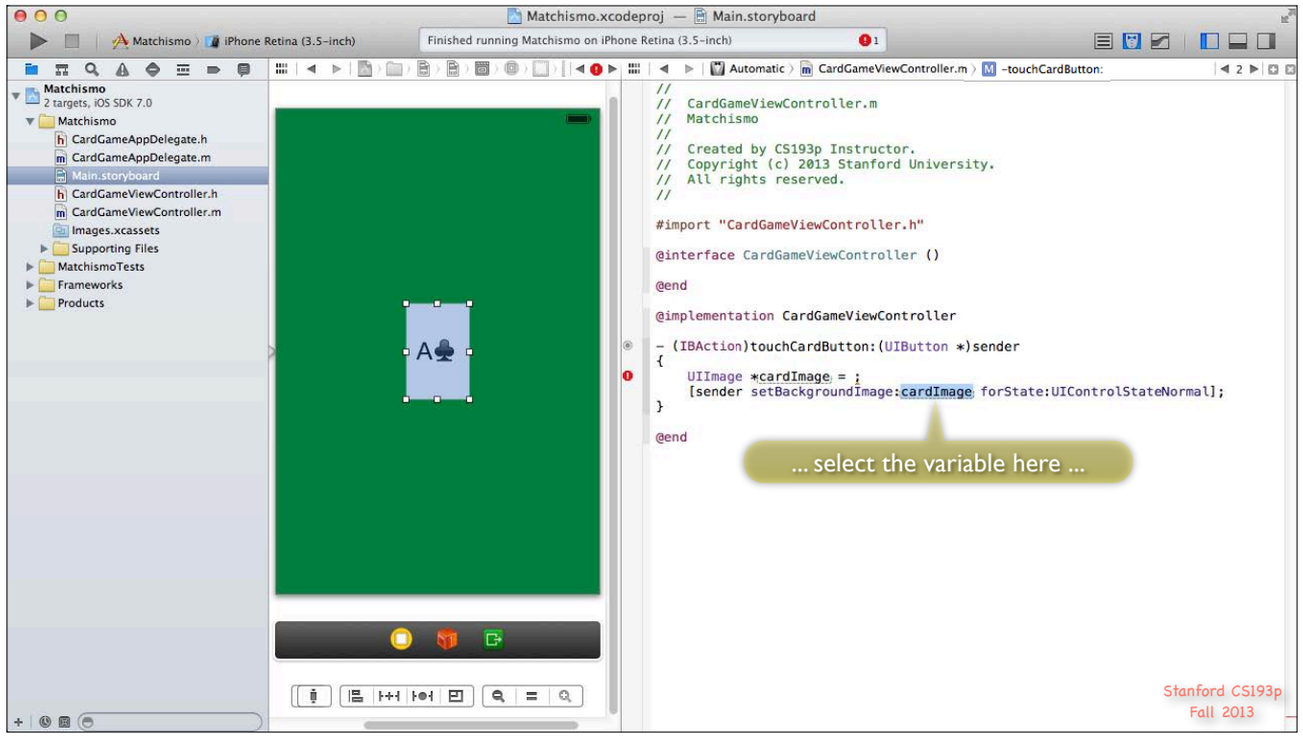

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:cardImage forState:(UIControlState)]
}
@end
The setBackgroundImage:forState: method asks for the state because you can set different background images for selected, highlighted or disabled states of the UIButton.
We're just going to set the background for the default (Normal) state so it will be that way in all button states.
Stanford CS193p
Fall 2013

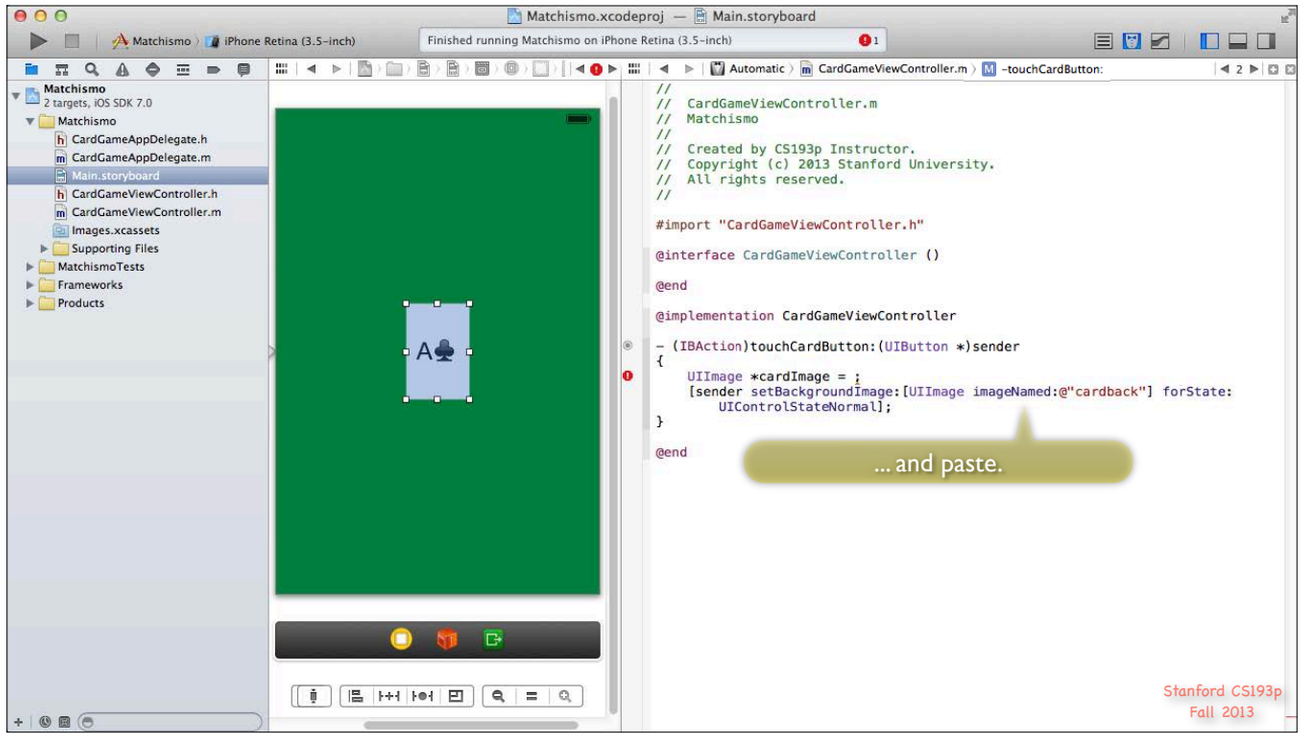

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:cardImage forState:UIControlStateApplication]
}
@end
enum UIControlEvents UIControlEventTouchUpOutside
enum UIControlEvents UIControlEventValueChanged
enum UIControlState UIControlStateApplication
enum UIControlState UIControlStateDisabled
enum UIControlState UIControlStateHighlighted
enum UIControlState UIControlStateNormal
enum UIControlState UIControlStateReserved
enum UIControlState UIControlStateSelected
Start typing UIC and you'll get close to what we want (UIControlStateNormal) ...
Stanford CS193p
Fall 2013

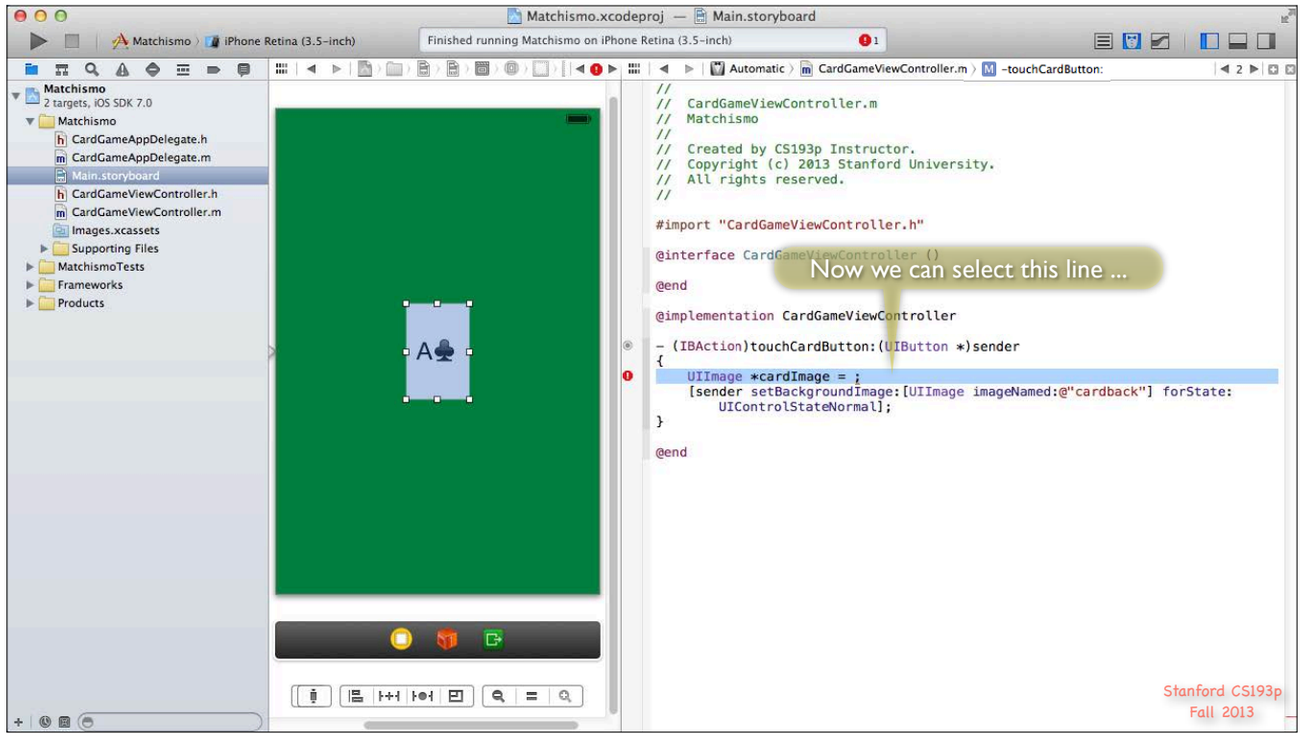

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:cardImage forState:UIControlStateNormal]
}
@end
enum UIControlEvents UIControlEventTouchUpOutside
enum UIControlEvents UIControlEventValueChanged
enum UIControlState UIControlStateApplication
enum UIControlState UIControlStateDisabled
enum UIControlState UIControlStateHighlighted
enum UIControlState UIControlStateNormal
enum UIControlState UIControlStateReserved
enum UIControlState UIControlStateSelected
Now choose UIControlStateNormal.
Stanford CS193p
Fall 2013

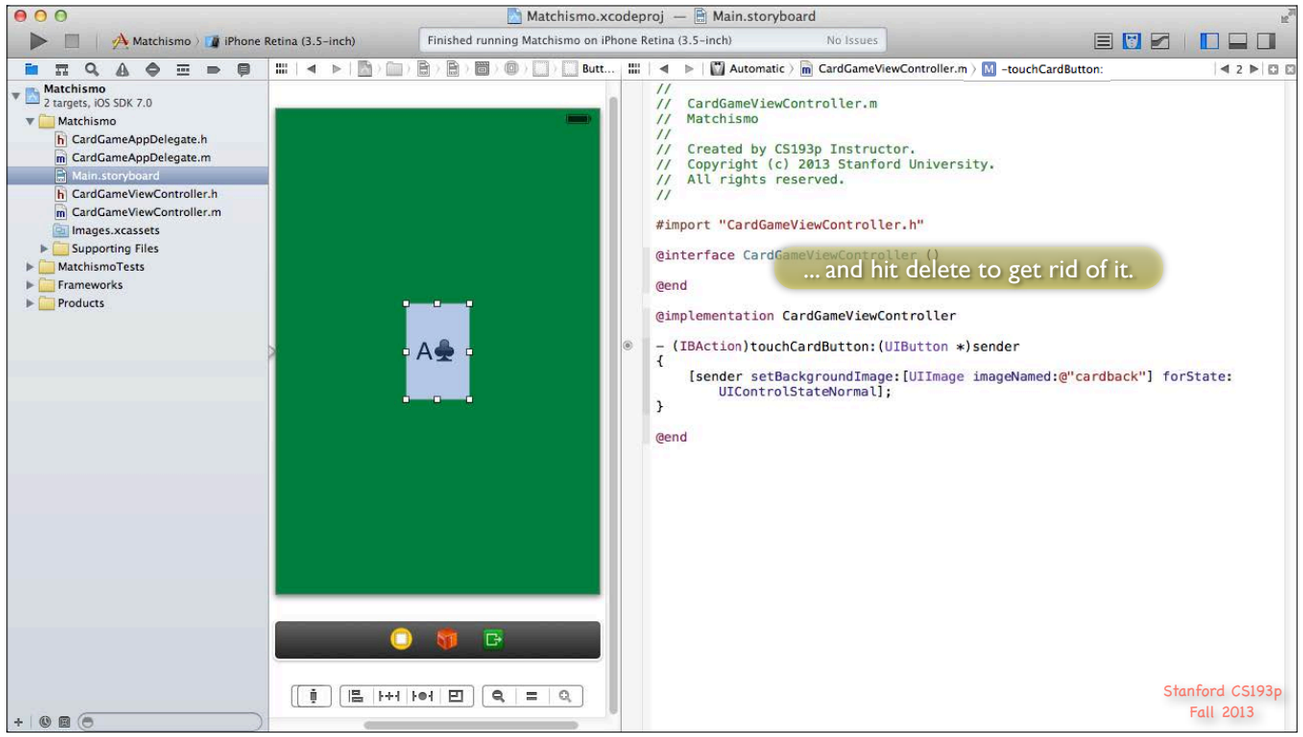

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = [UIImage imageNamed:@"cardback"];
[sender setBackgroundImage:cardImage forState:UIControlStateNormal];
}
@end
Easy!
But we don't actually need this local variable.
We can just embed that message-send in-line.
So select this message send ...
Stanford CS193p
Fall 2013

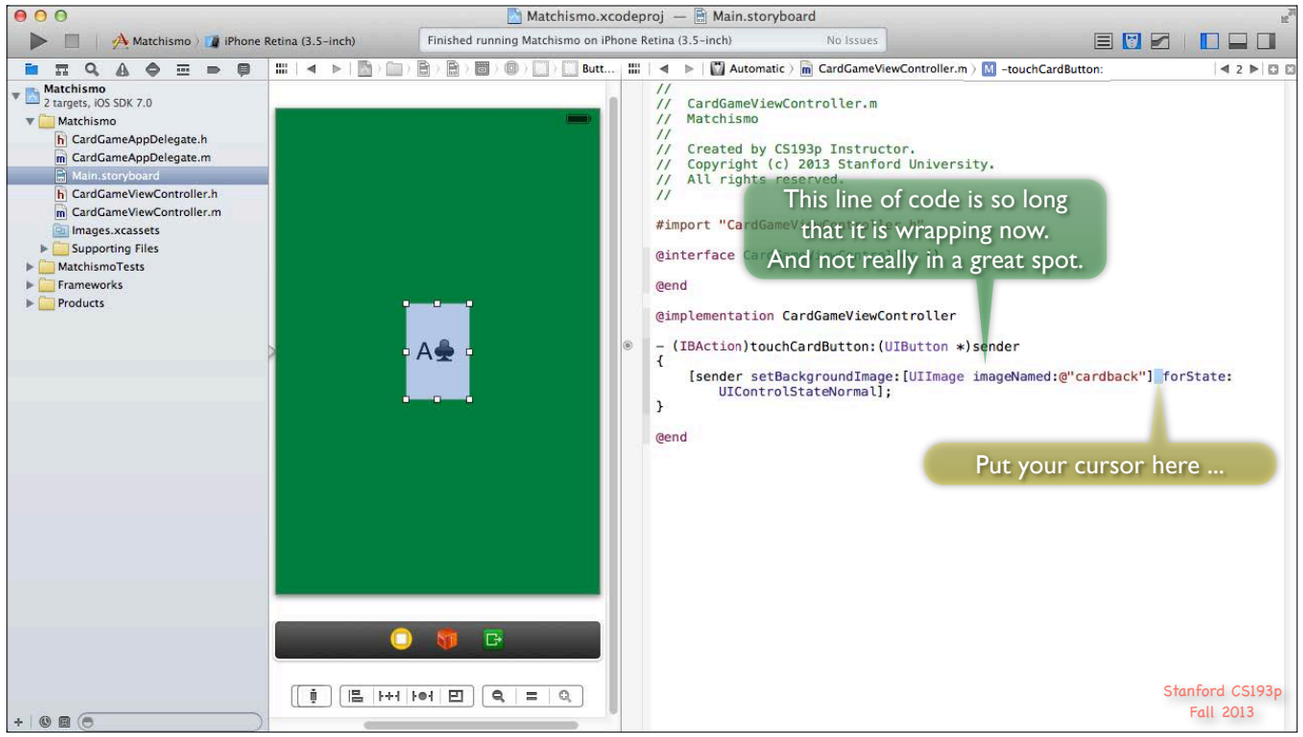

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = ;
[sender setBackgroundImage:cardImage forState:UIControlStateNormal];
}
@end
... cut it ...
Stanford CS193p
Fall 2013

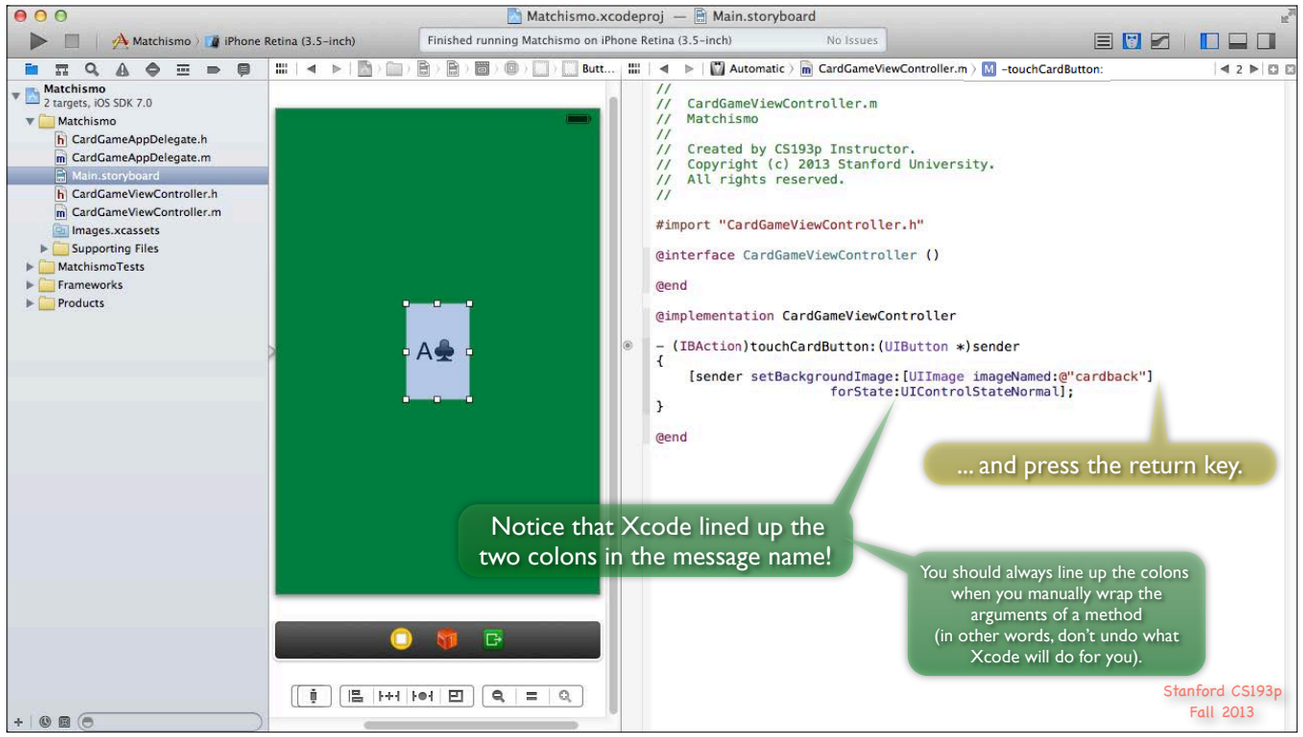

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = ;
[sender setBackgroundImage:cardImage forState:UIControlStateNormal];
}
@end
... select the variable here ...
Stanford CS193p
Fall 2013

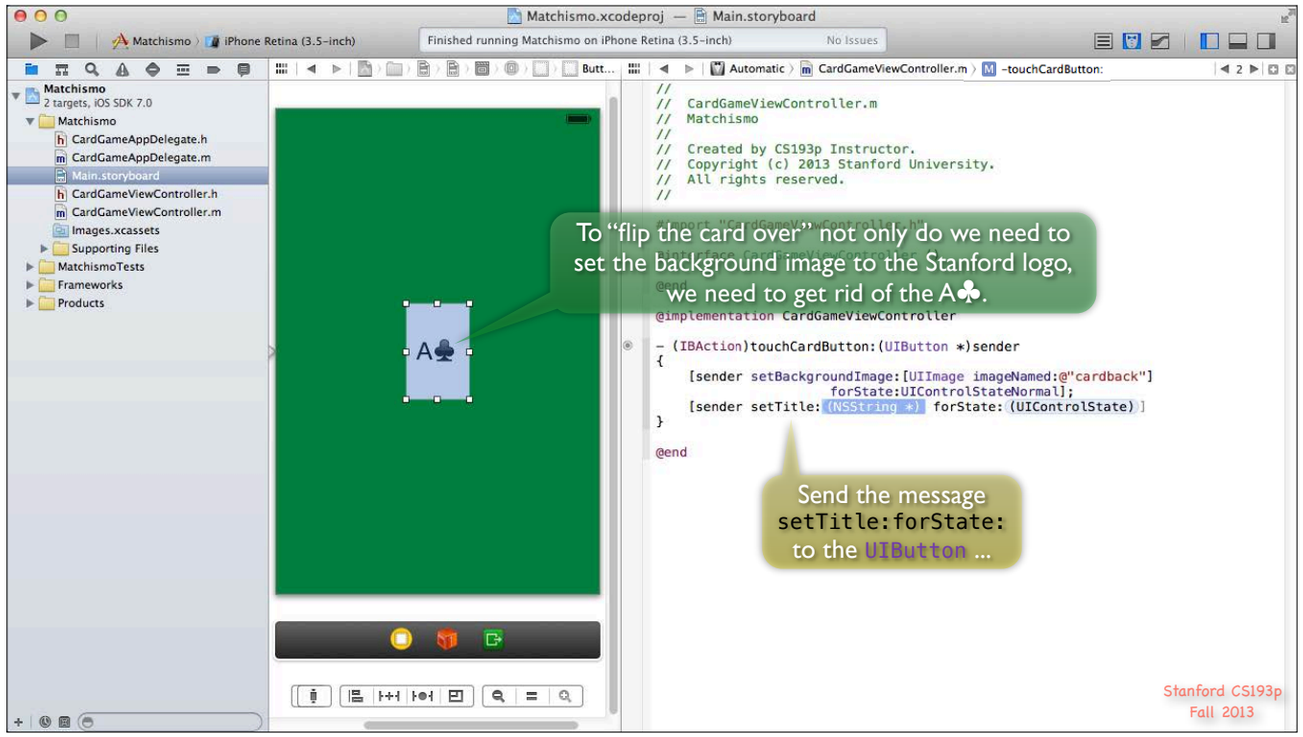

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = ;
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:
UIControlStateNormal];
}
@end
... and paste.
Stanford CS193p
Fall 2013

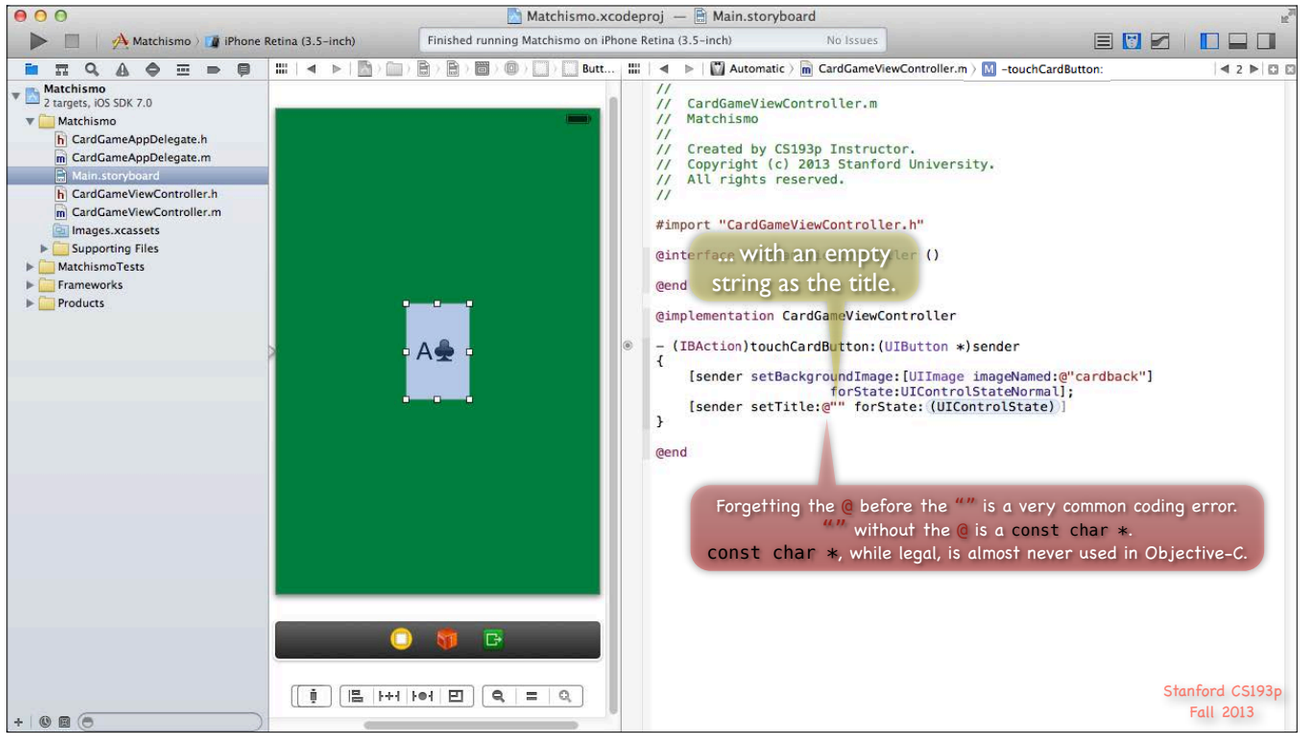

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
UIImage *cardImage = ;
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:
UIControlStateNormal];
}
@end
Now we can select this line ...
Stanford CS193p
Fall 2013

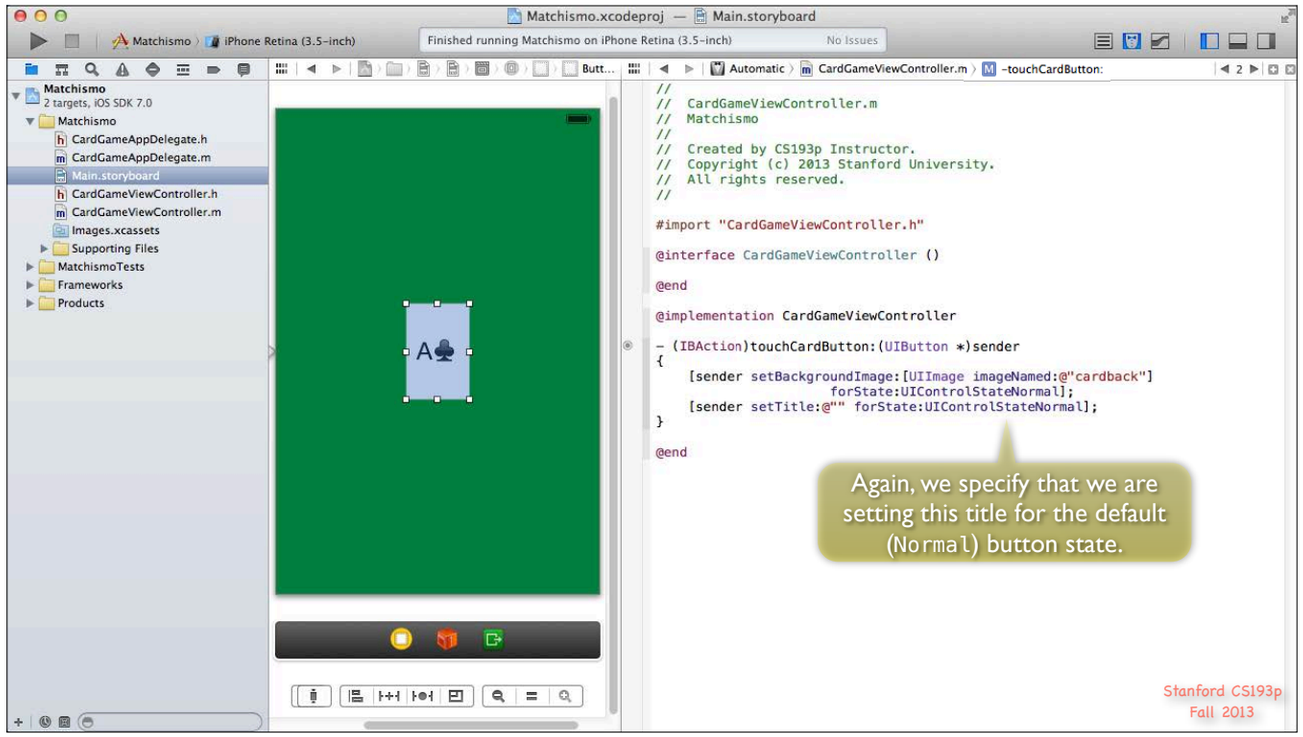

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:
UIControlStateNormal];
}
@end
... and hit delete to get rid of it.
Stanford CS193p
Fall 2013

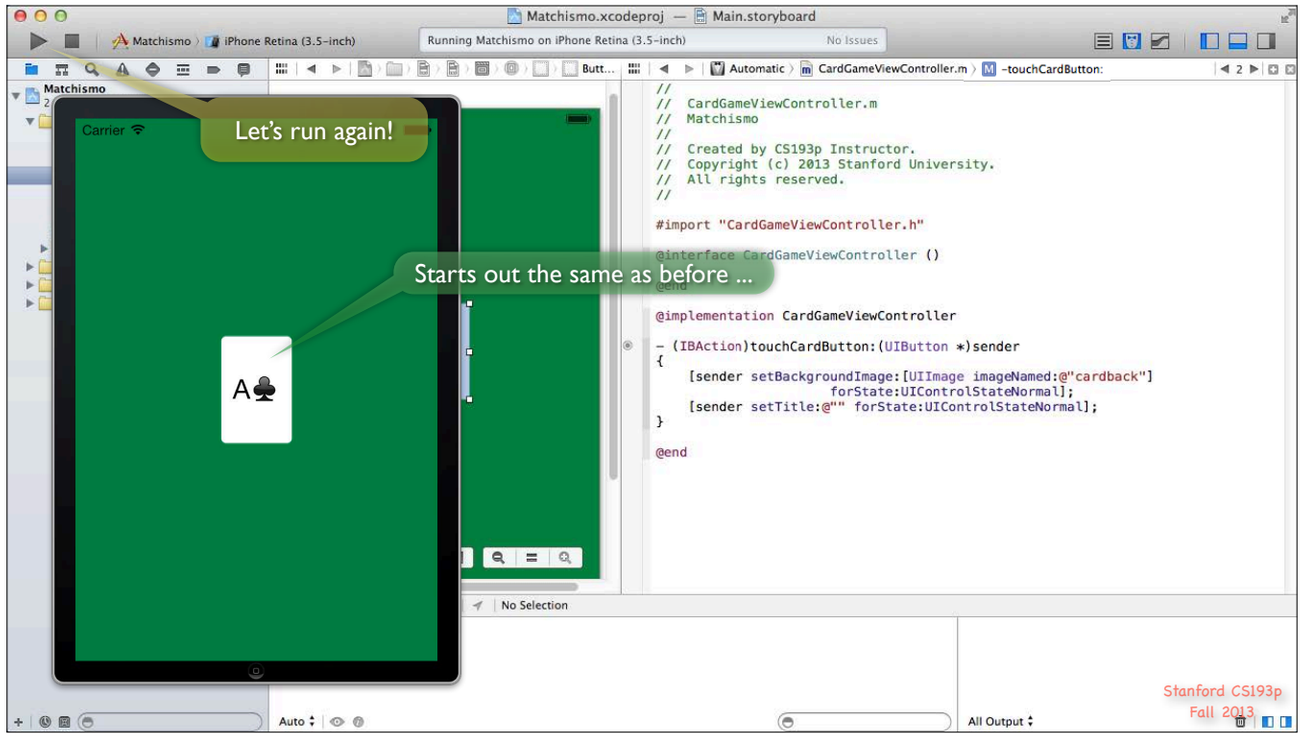

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
@interface
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:
UIControlStateNormal];
}
@end
This line of code is so long that it is wrapping now. And not really in a great spot.
Put your cursor here ...
Stanford CS193p
Fall 2013

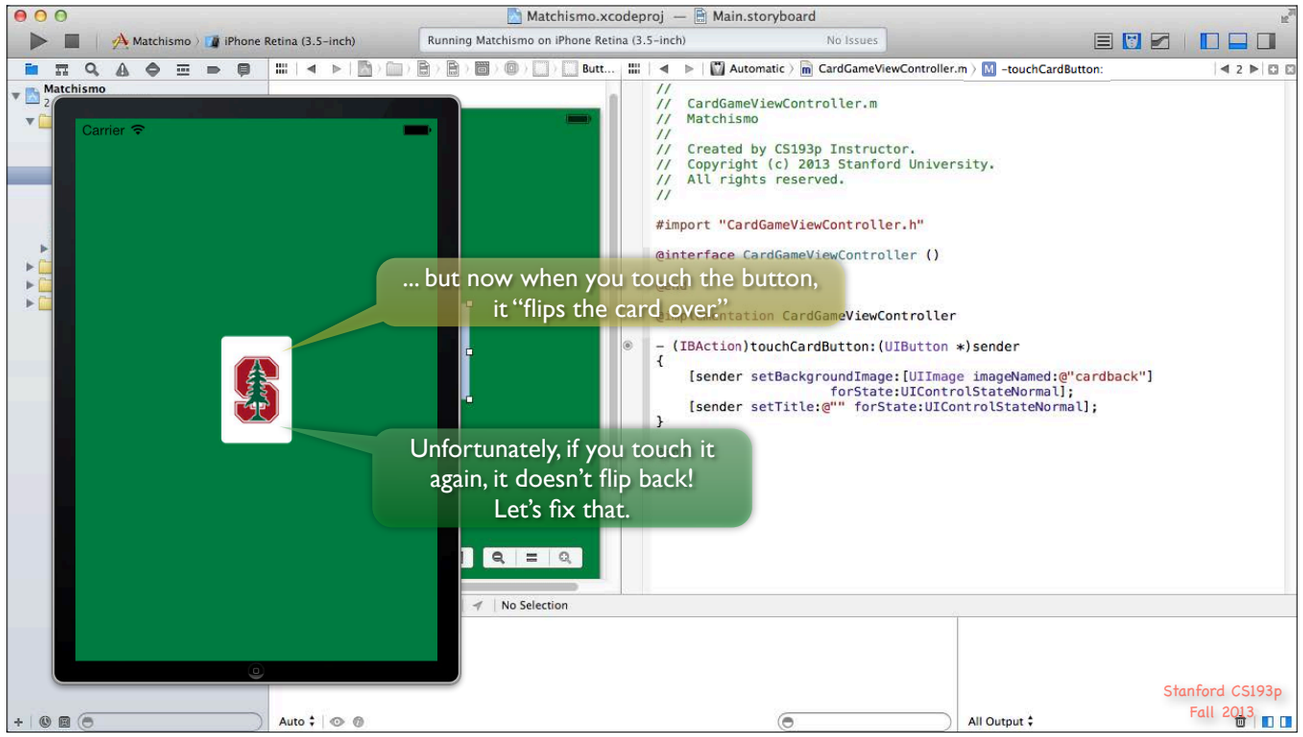

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
}
@end
... and press the return key.
Notice that Xcode lined up the two colons in the message name!
You should always line up the colons when you manually wrap the arguments of a method (in other words, don't undo what Xcode will do for you).
Stanford CS193p
Fall 2013

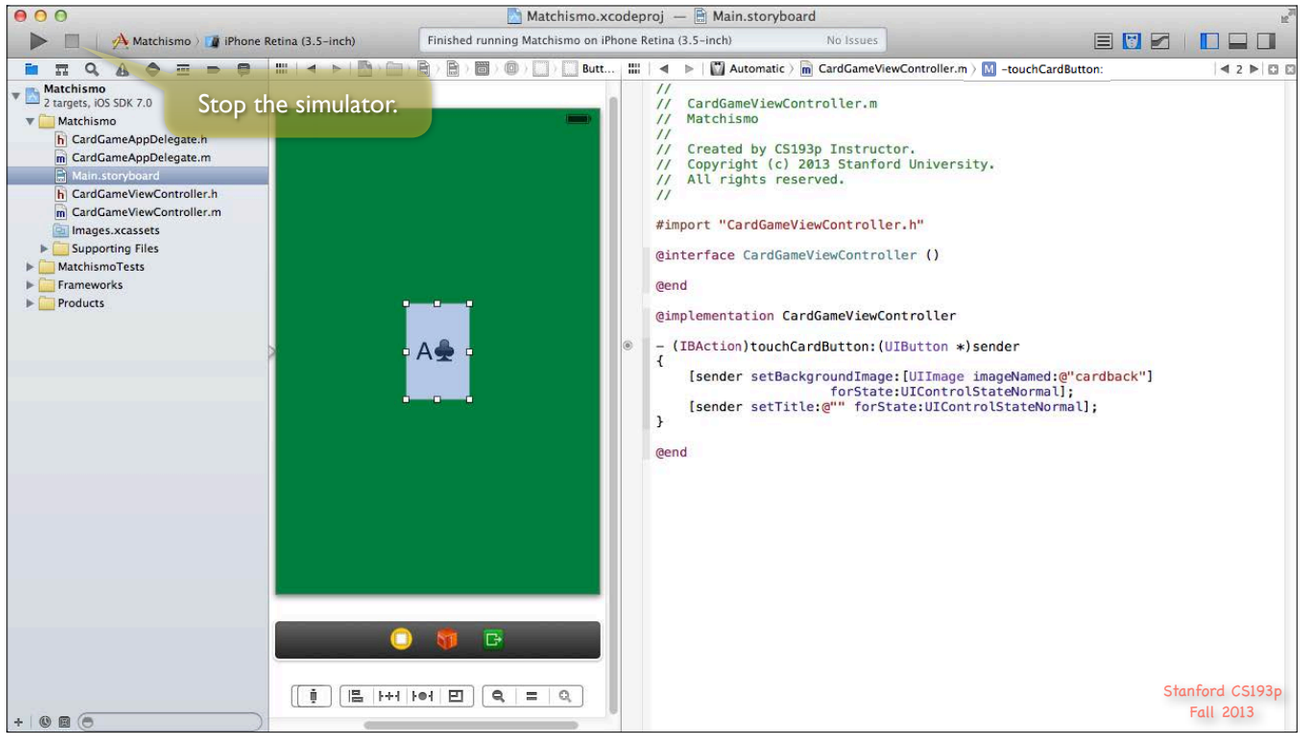

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:(NSString *) forState:(UIControlState)]
}
@end
To “flip the card over” not only do we need to set the background image to the Stanford logo, we need to get rid of the A♣.
Send the message setTitle:forState: to the UIButton …
Stanford CS193p
Fall 2013

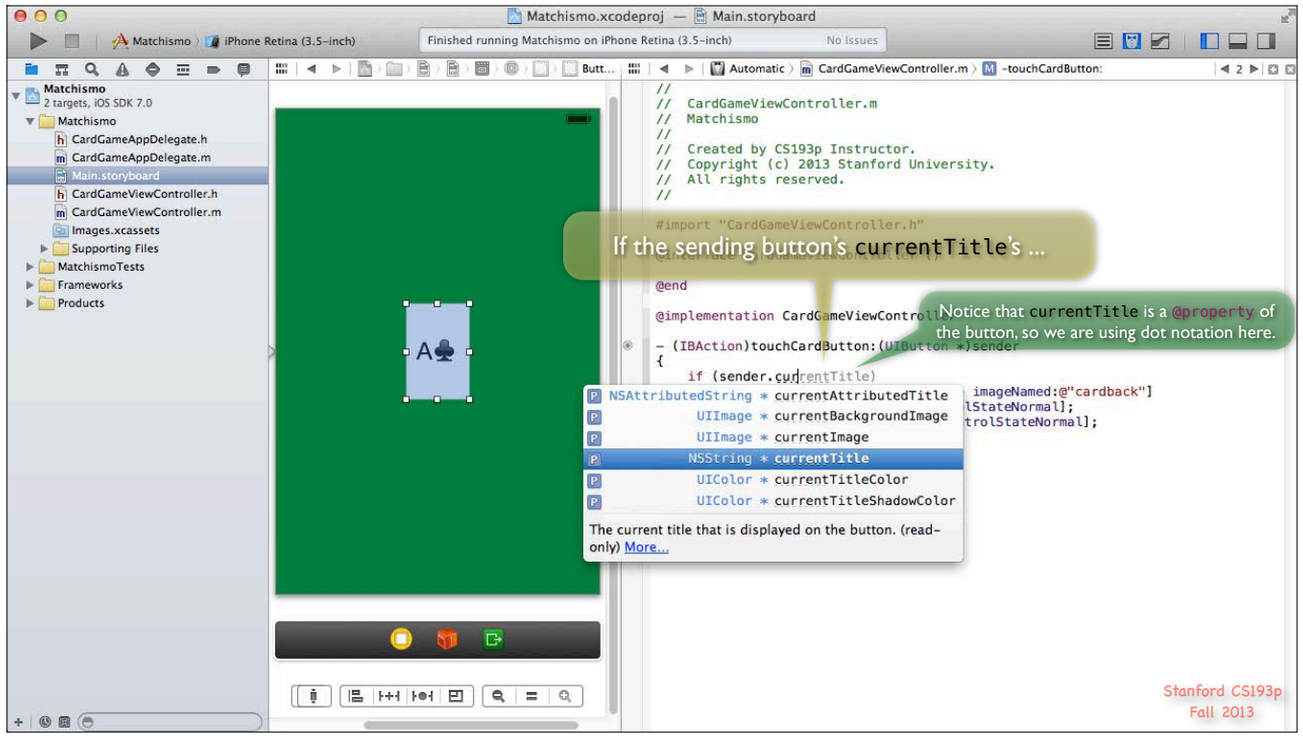

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:(UIControlState)]
}
@end
... with an empty string as the title.
Forgetting the @ before the "" is a very common coding error.
"" without the @ is a const char *.
const char *, while legal, is almost never used in Objective-C.
Stanford CS193p
Fall 2013

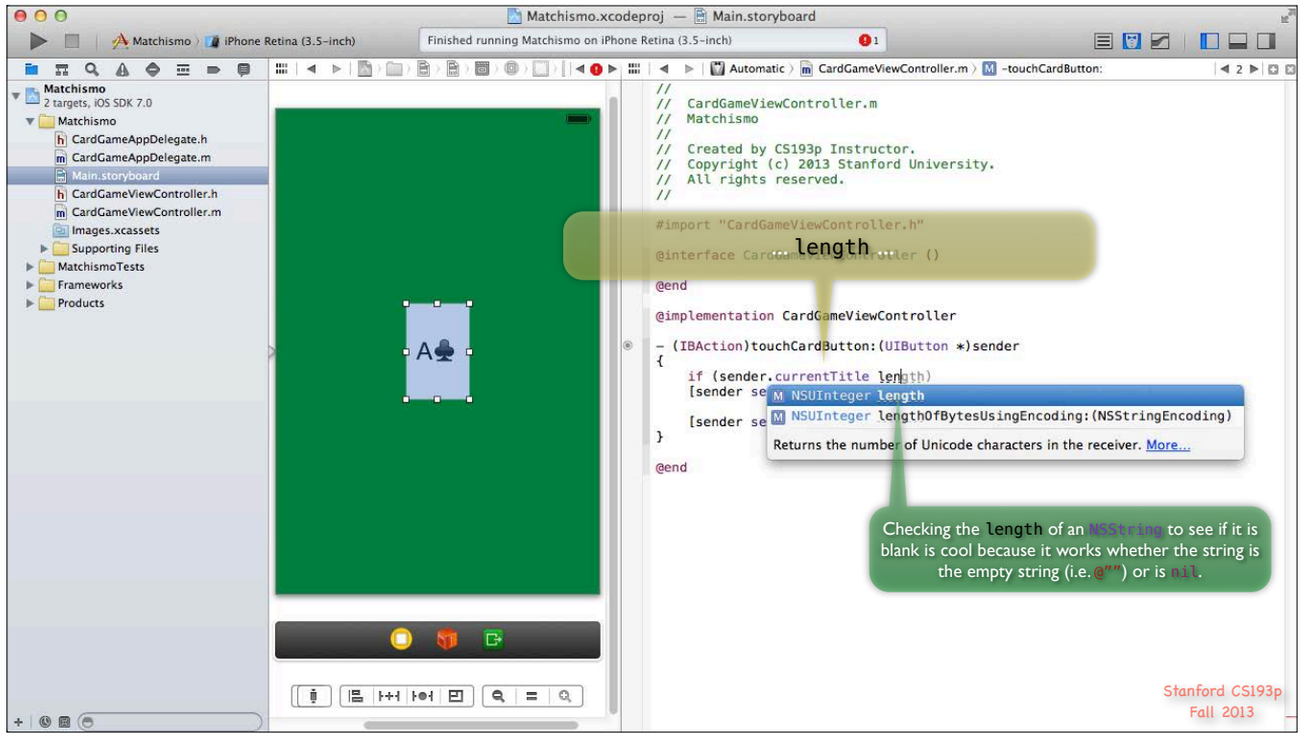

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
Again, we specify that we are setting this title for the default (Normal) button state.
Stanford CS193p
Fall 2013

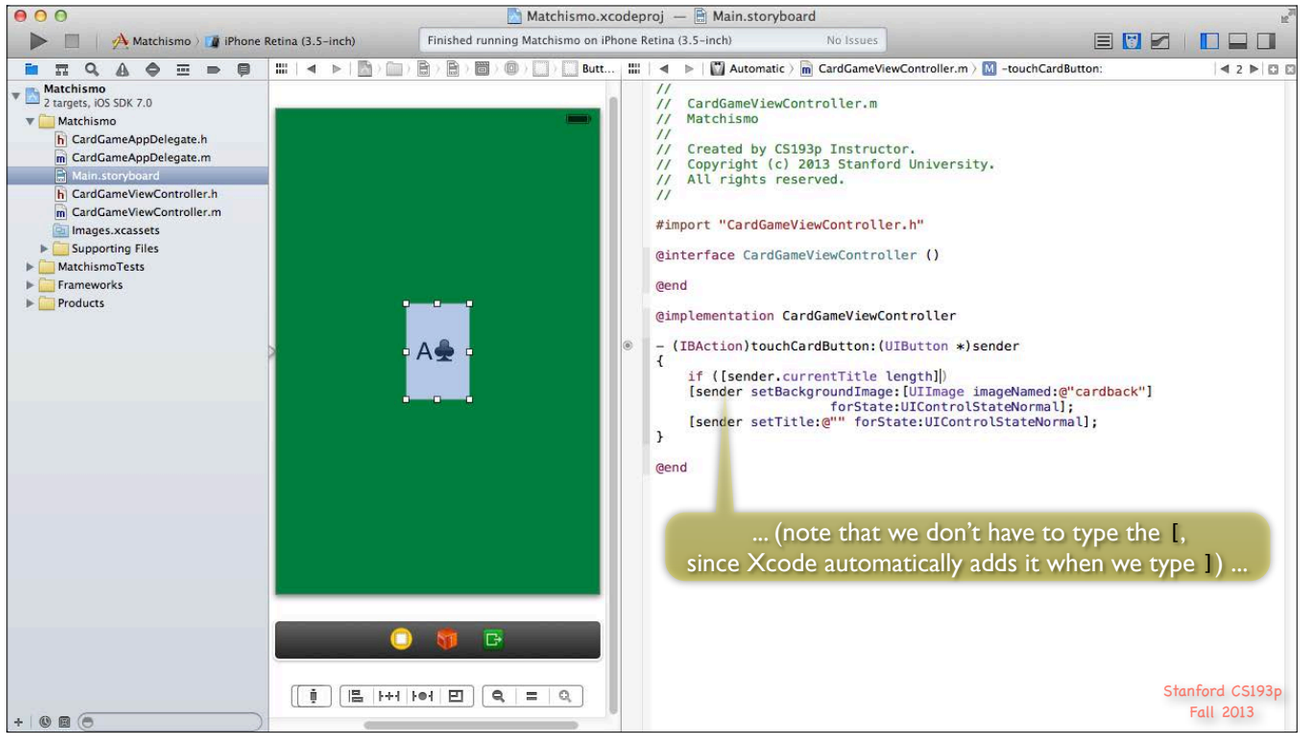

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
Carrier
Let's run again!
Starts out the same as before ...
A♣
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
No Selection
Auto
All Output
Stanford CS193p
Fall 2013

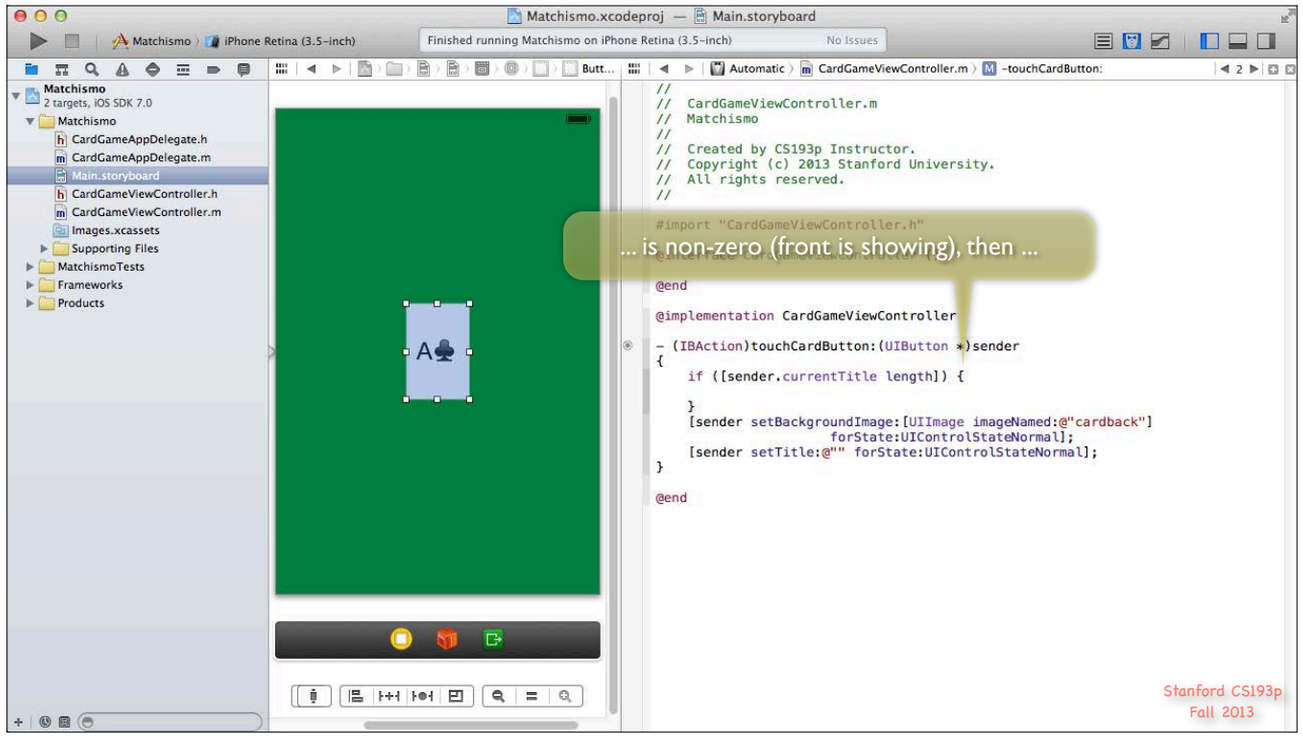

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
Carrier
... but now when you touch the button, it "flips the card over."
Unfortunately, if you touch it again, it doesn't flip back! Let's fix that.
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
No Selection
Auto
All Output
Stanford CS193p
Fall 2013

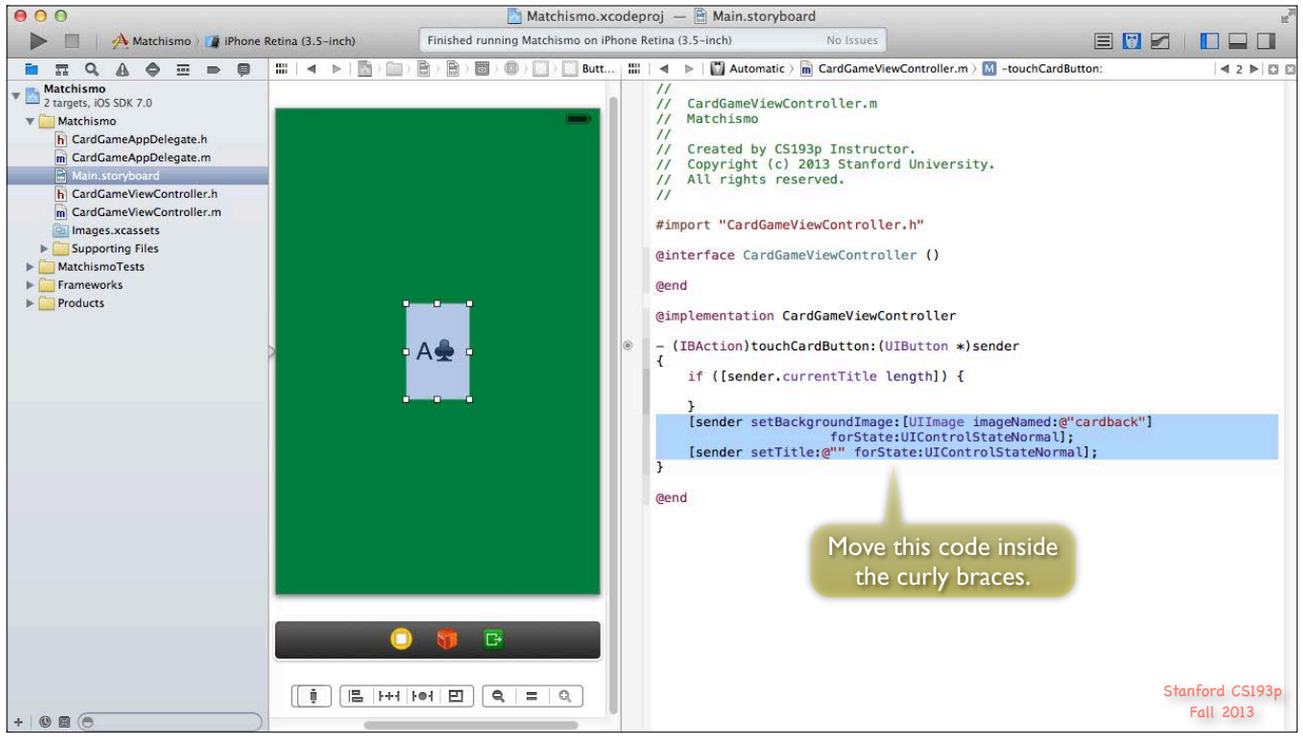

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Stop the simulator.
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
Stanford CS193p
Fall 2013

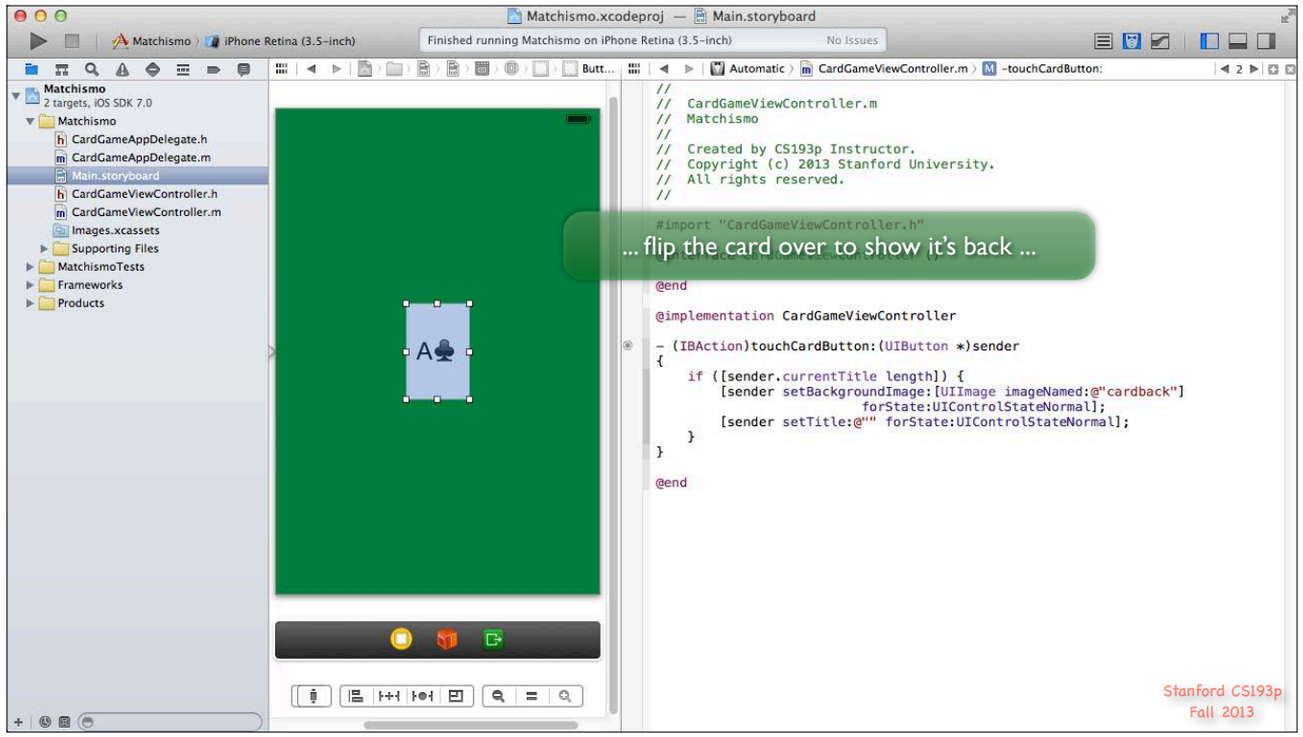

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@end
@implementation CardGameViewControll
- (IBAction)touchCardButton:(UIButton *)sender
{
if (sender.currentTitle)
If the sending button's currentTitle's ...
Notice that currentTitle is a @property of the button, so we are using dot notation here.
NSAttributedString * currentAttributedTitle
UIImage * currentBackgroundImage
UIImage * currentImage
NSString * currentTitle
UIColor * currentTitleColor
UIColor * currentTitleShadowColor
The current title that is displayed on the button. (read-only) More...
imageNamed:@"cardback"]
lStateNormal];
trolStateNormal];
Stanford CS193p
Fall 2013

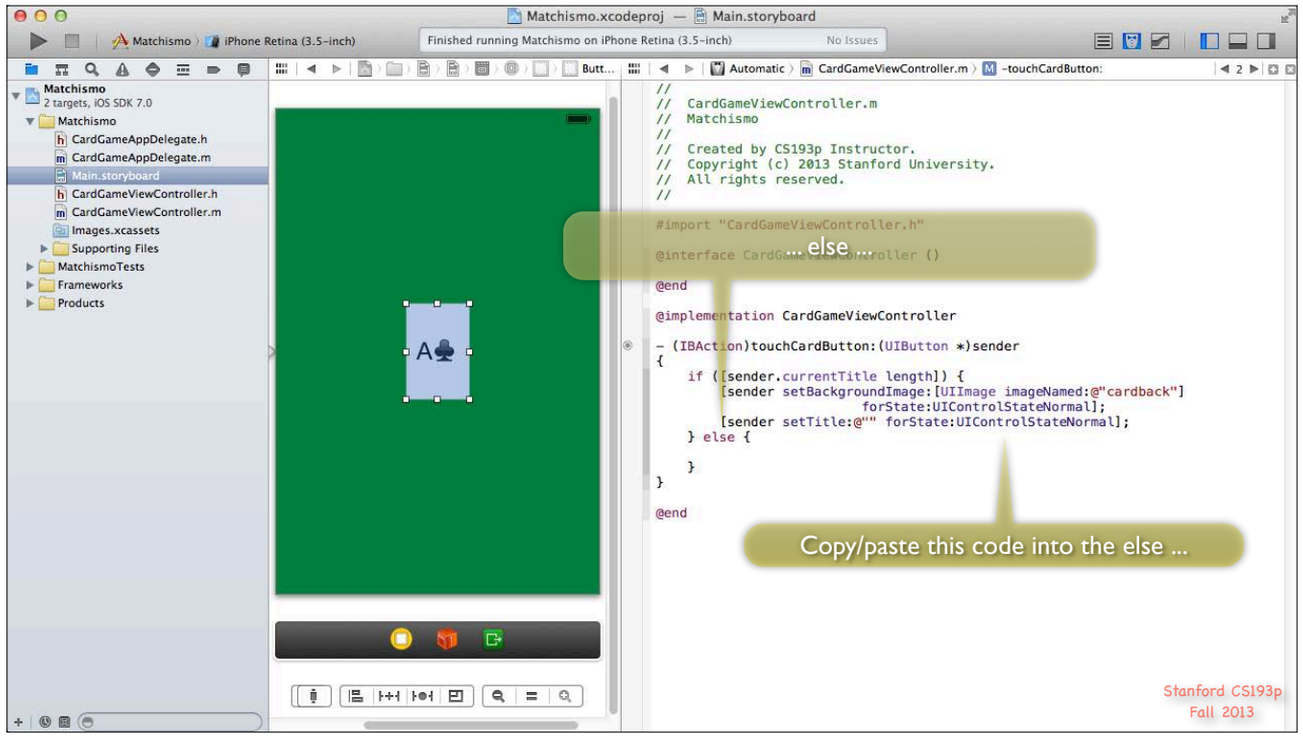

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -touchCardButton:
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
A♣
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if (sender.currentTitle length)
[sender se
[sender se
}
@end
... length ...
NSUInteger length
NSUInteger lengthOfBytesUsingEncoding:(NSStringEncoding)
Returns the number of Unicode characters in the receiver. More...
Checking the length of an NSString to see if it is blank is cool because it works whether the string is the empty string (i.e. @"") or is nil.
Stanford CS193p
Fall 2013

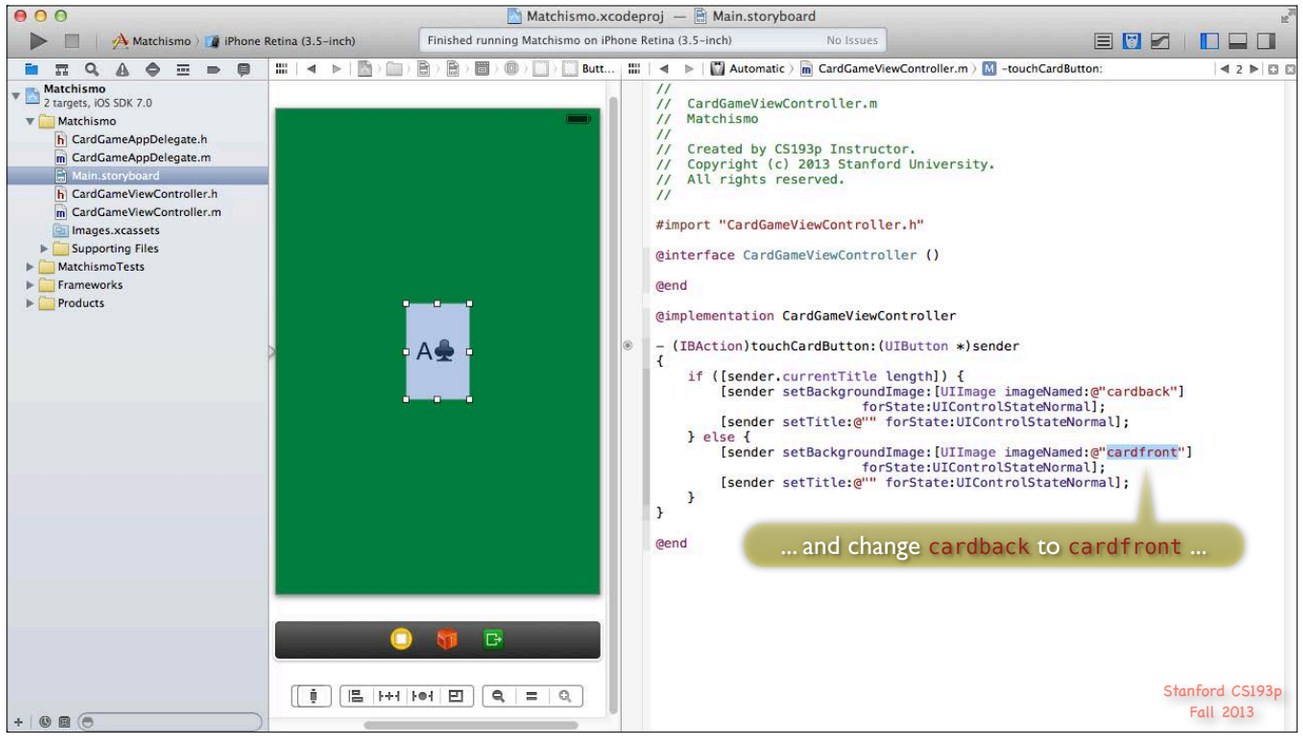

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length])
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
... (note that we don't have to type the [,
since Xcode automatically adds it when we type]) ...
Stanford CS193p
Fall 2013

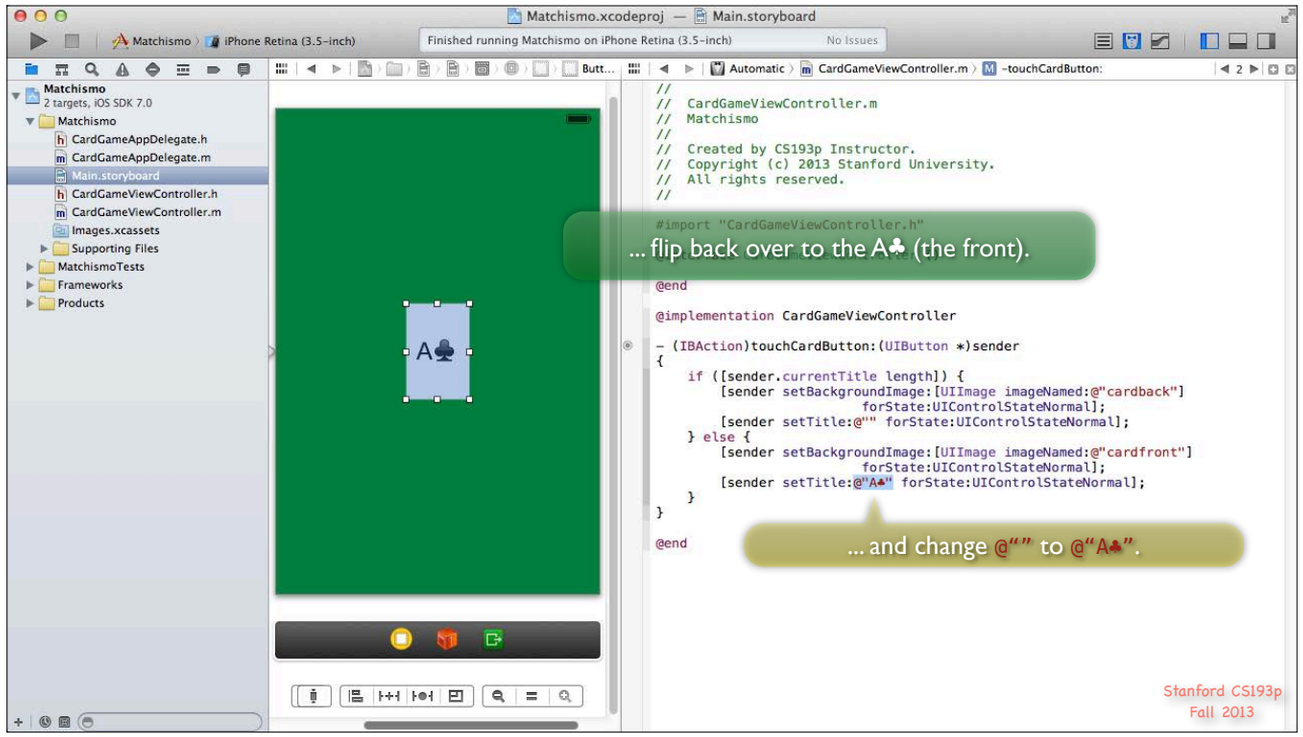

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
}
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
... is non-zero (front is showing), then ...
Stanford CS193p
Fall 2013

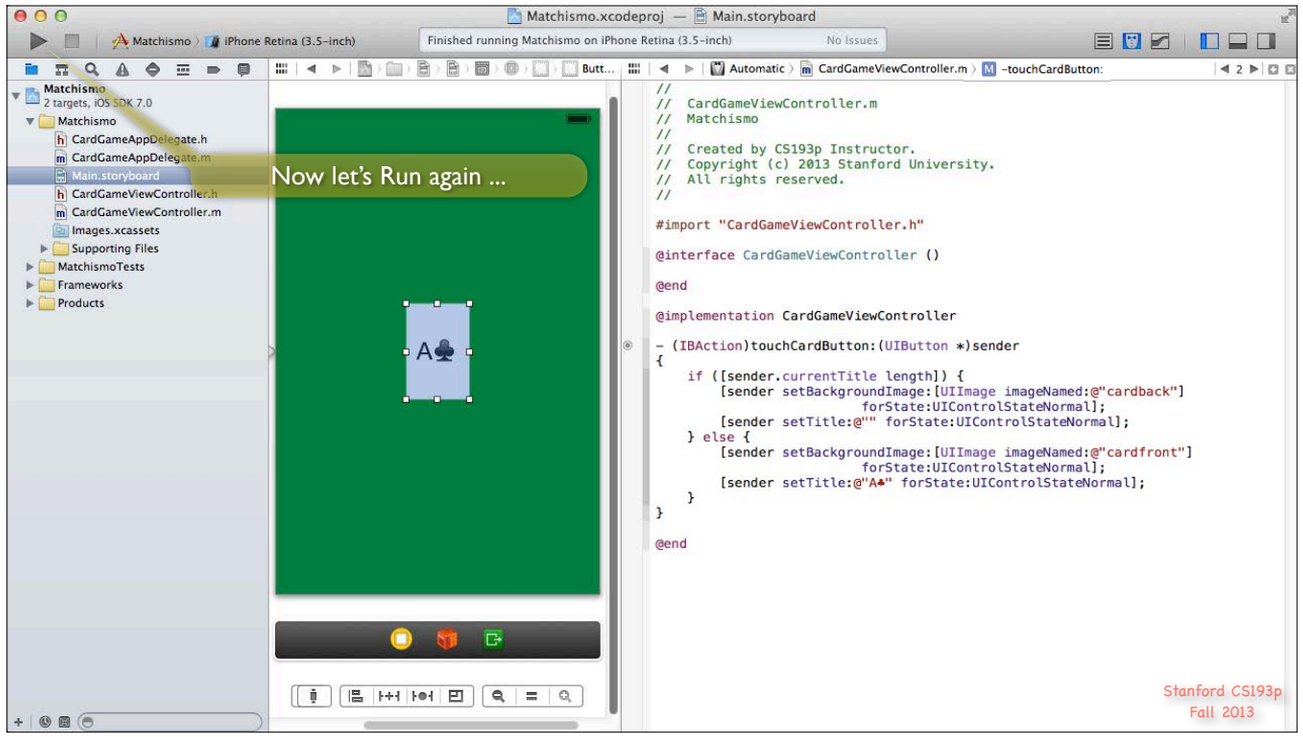

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
}
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
@end
Move this code inside the curly braces.
Stanford CS193p
Fall 2013

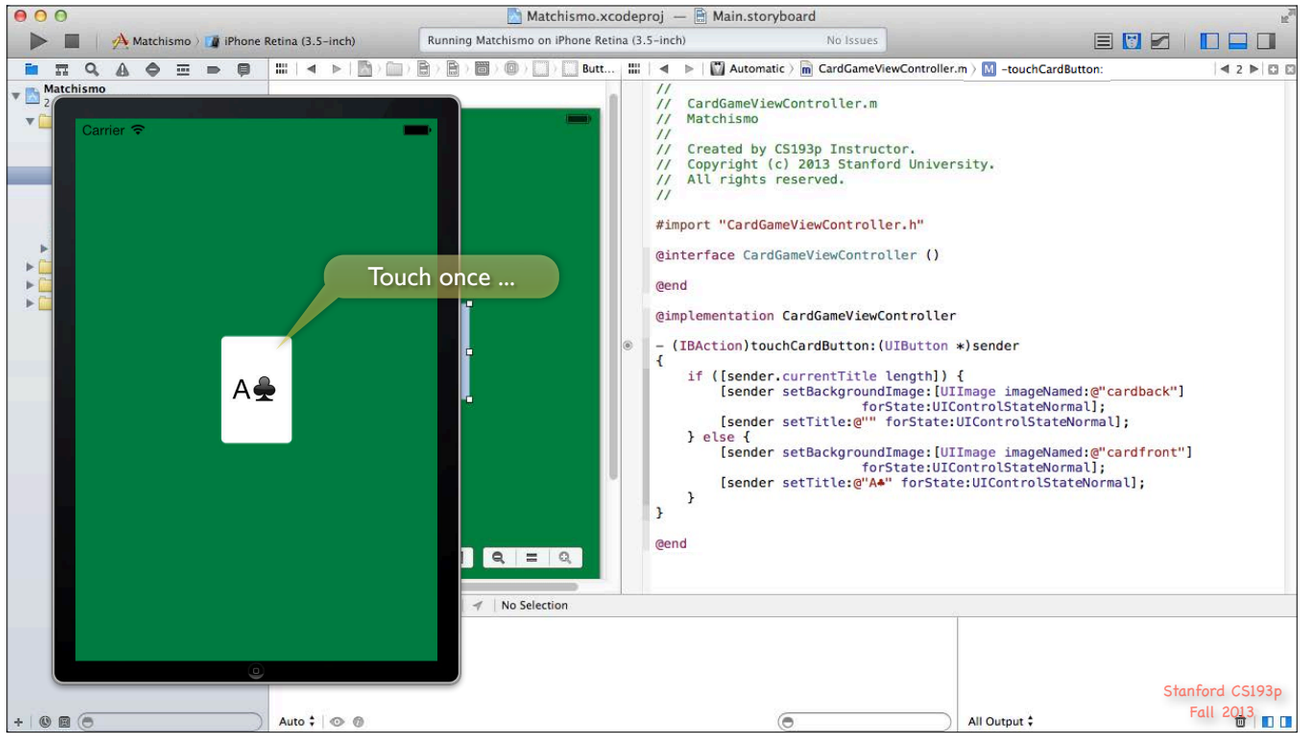

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
// CardGameViewController.m Matchismo // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import "CardGameViewController.h"
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
}
@end
... flip the card over to show it's back ...
Stanford CS193p
Fall 2013

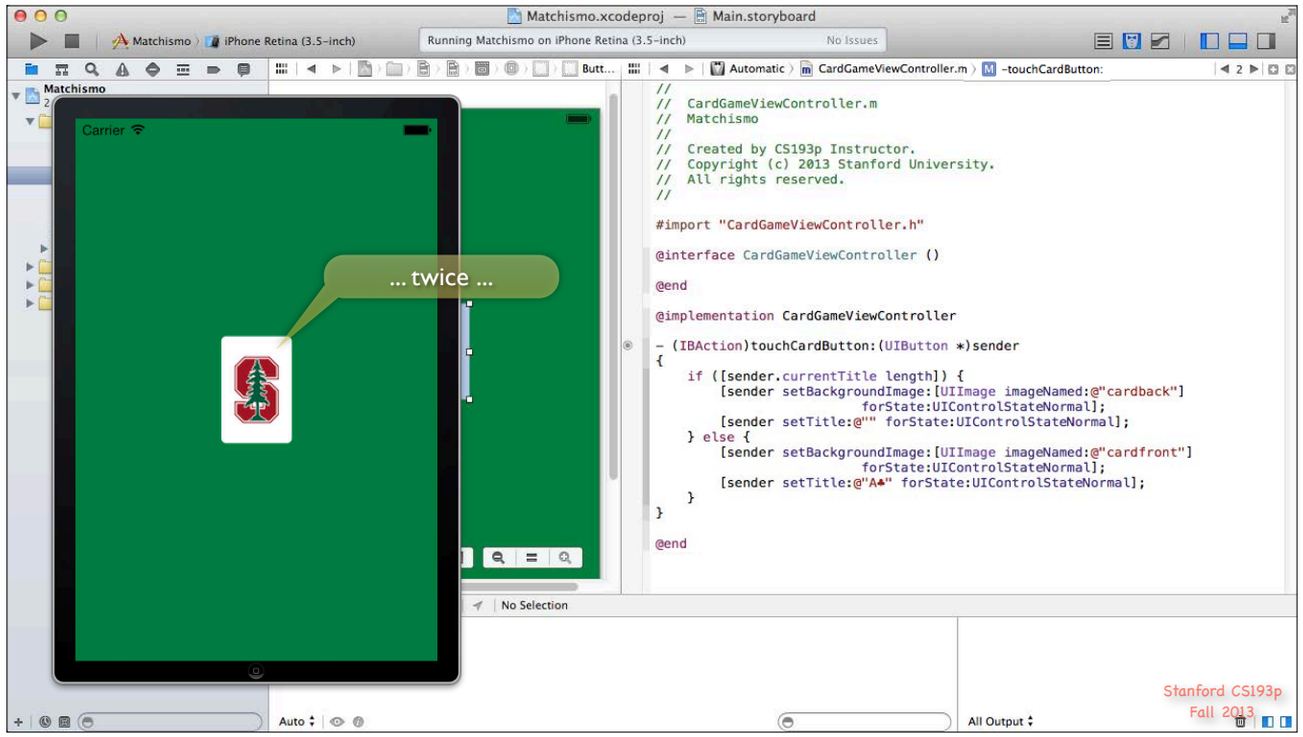

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
}
}
@end
... else ...
Copy/paste this code into the else ...
Stanford CS193p
Fall 2013

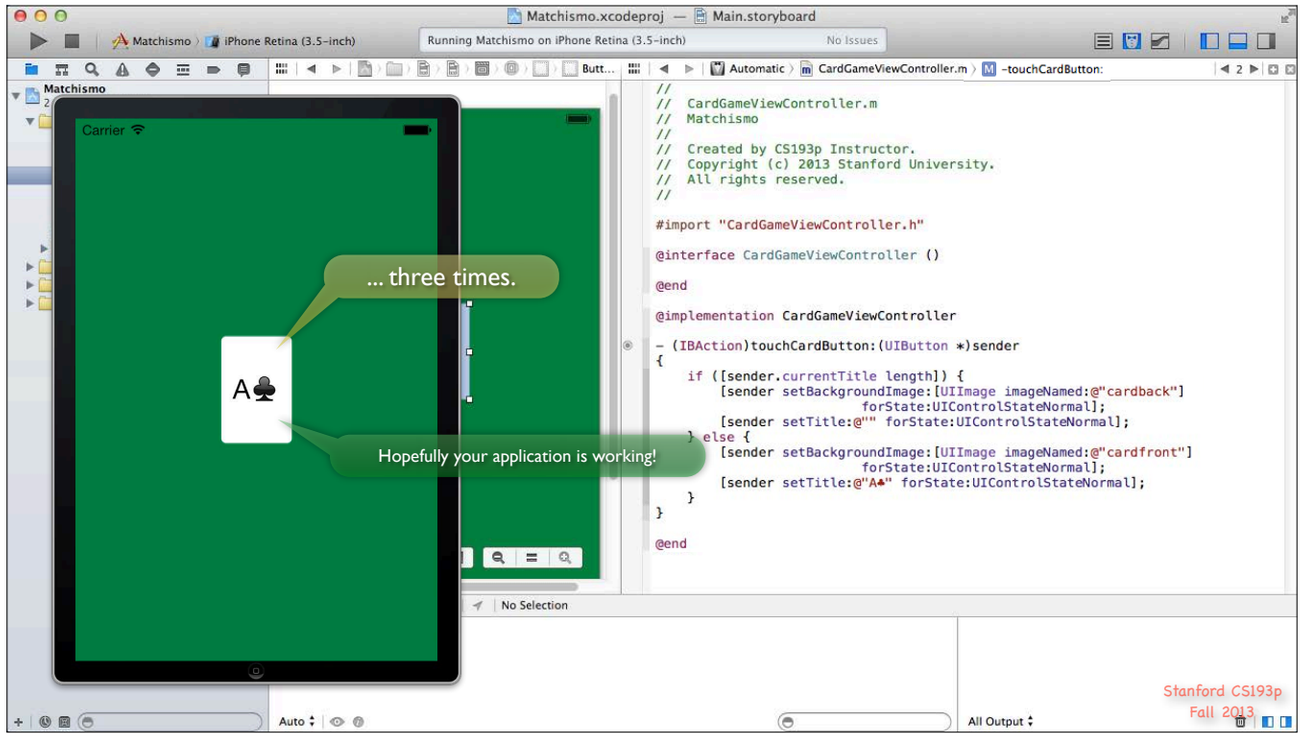

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
}
}
@end
... and change cardback to cardfront ...
Stanford CS193p
Fall 2013

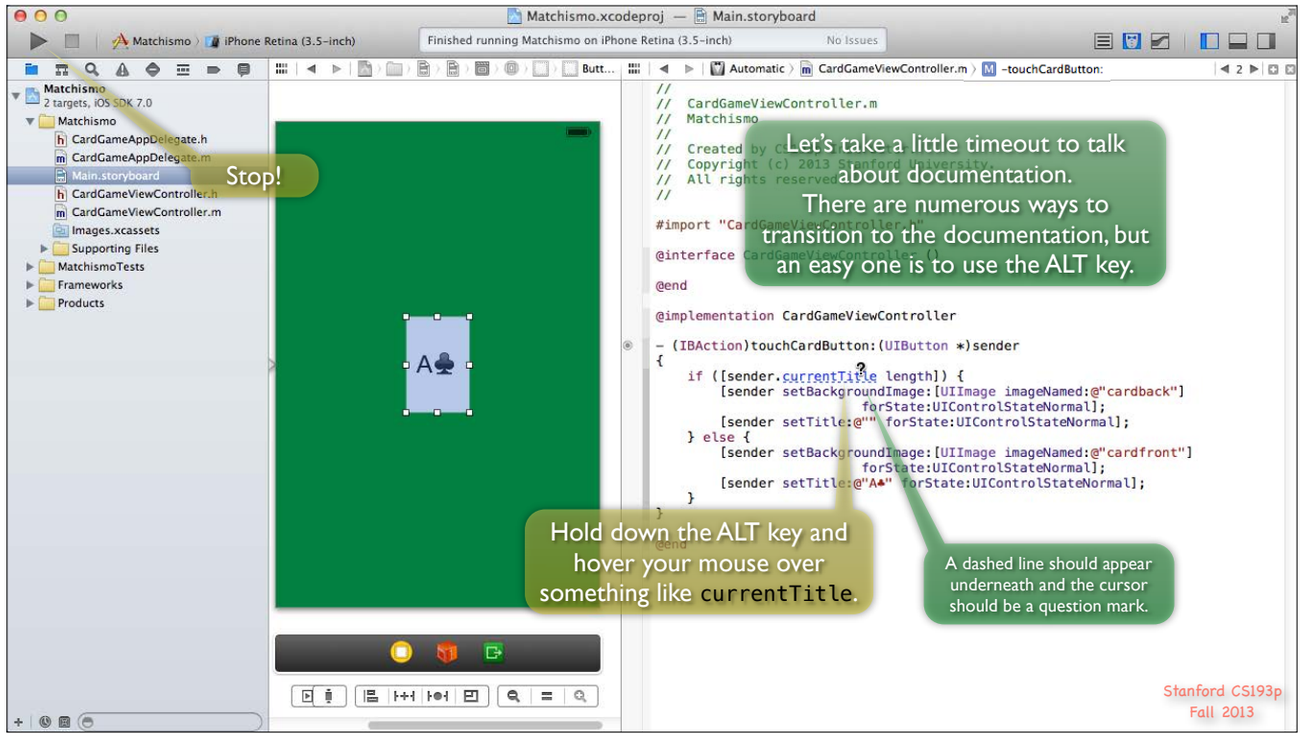

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
A♣
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
... flip back over to the A♣ (the front).
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
... and change @"" to @"A♣".
Stanford CS193p
Fall 2013

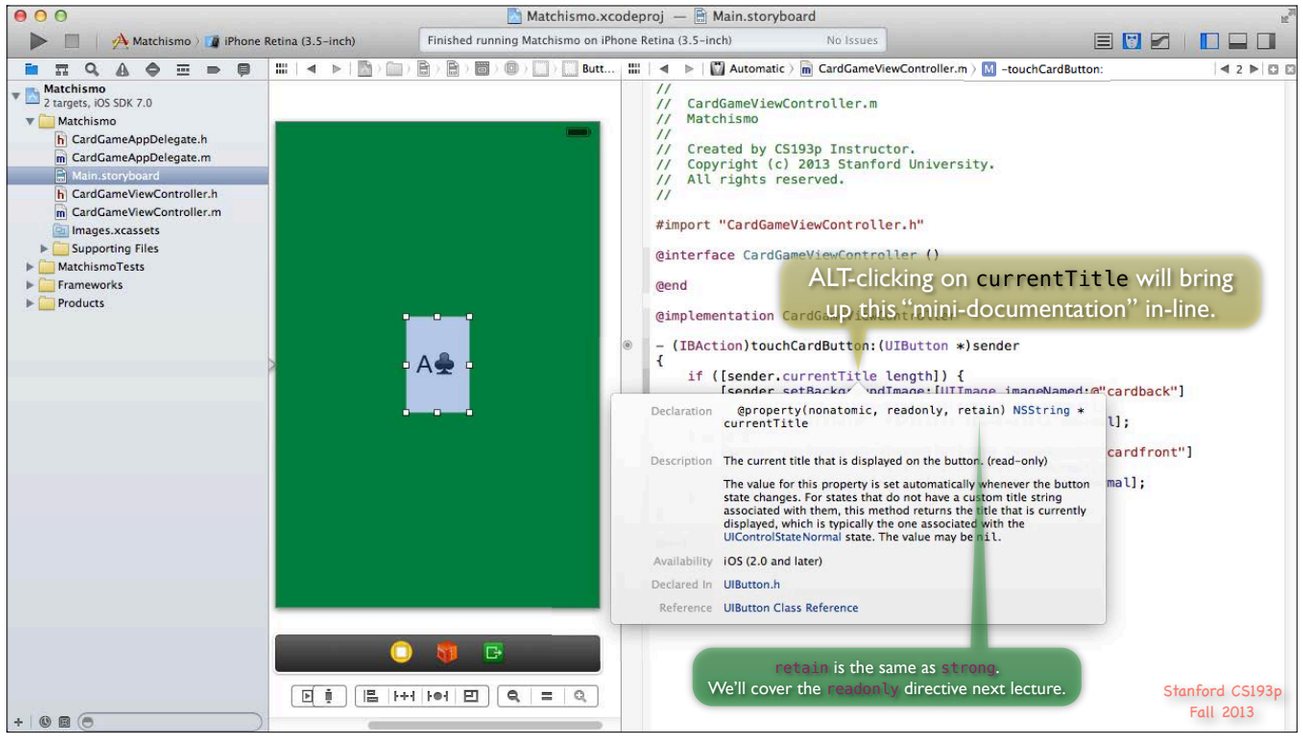

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Now let's Run again ...
A♣
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Stanford CS193p
Fall 2013

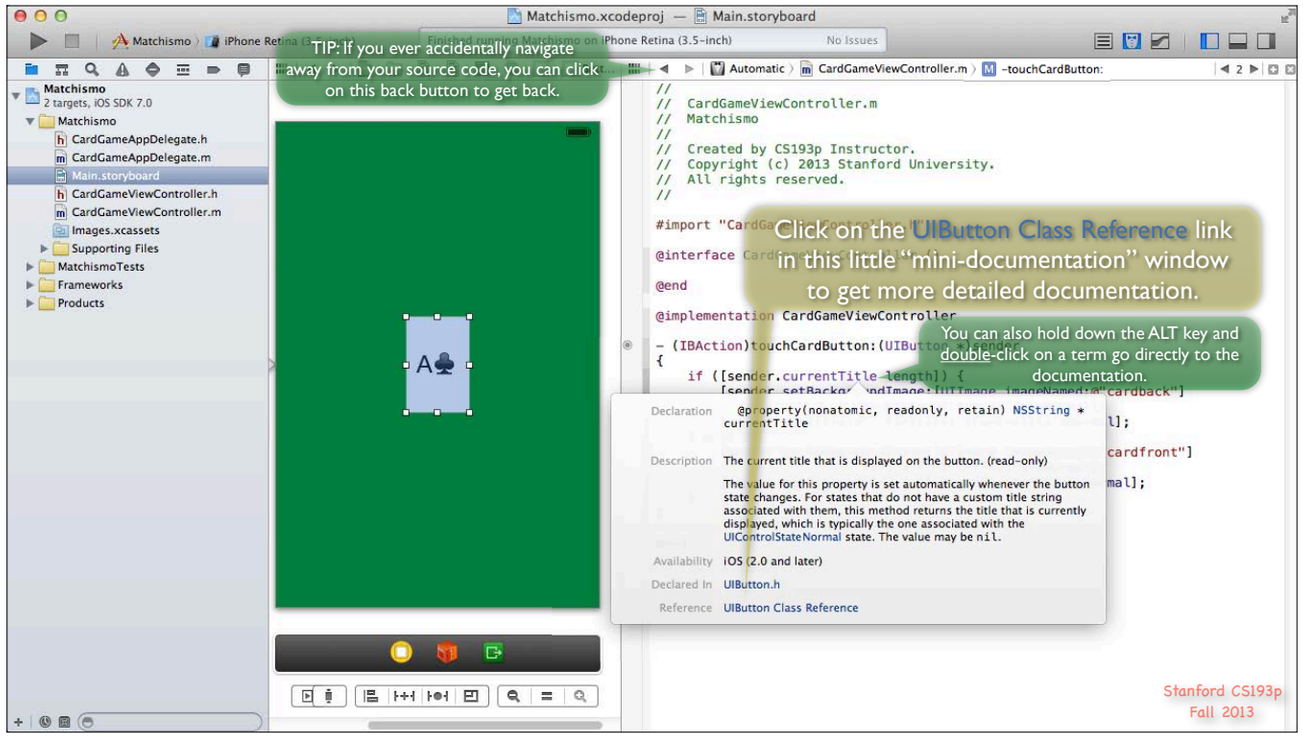

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
Carrier
Touch once ...
A♣
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
No Selection
Auto
All Output
Stanford CS193p
Fall 2013

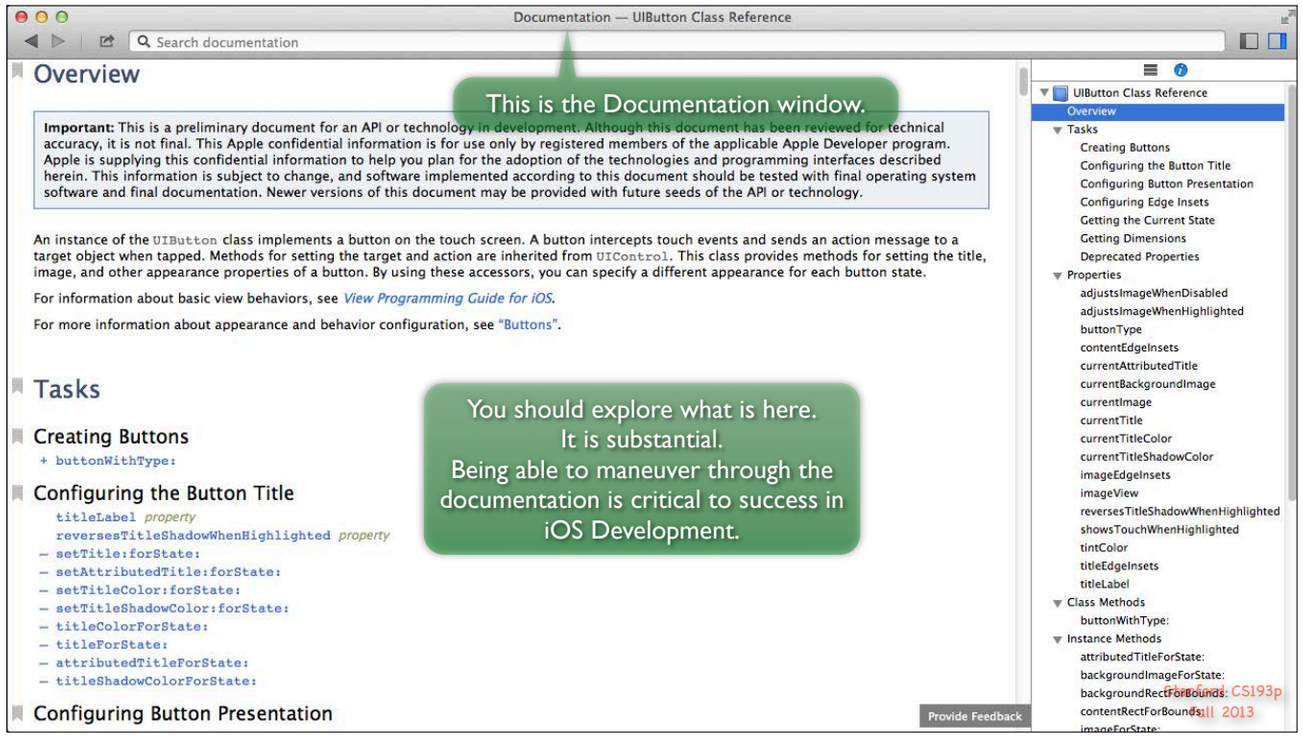

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
Carrier
... twice ...
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
No Selection
Auto
All Output
Stanford CS193p
Fall 2013

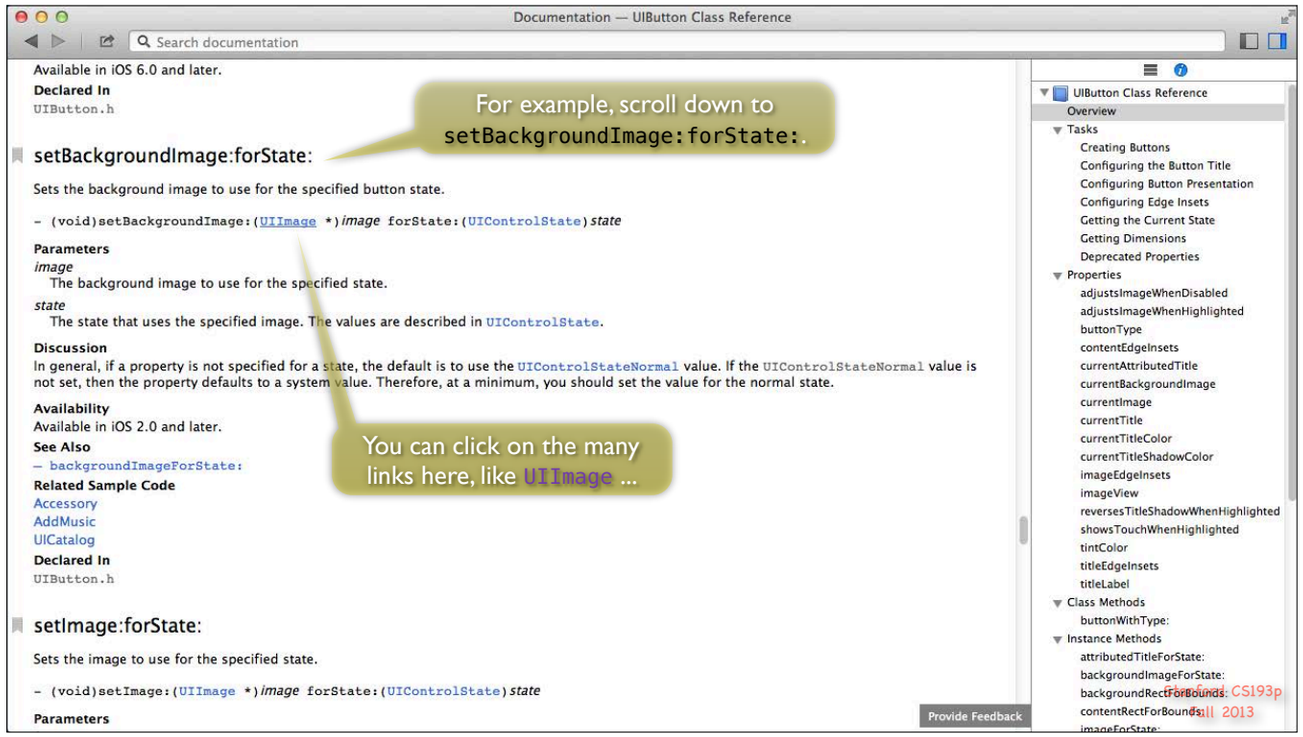

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
Carrier
... three times.
A♣
Hopefully your application is working!
No Selection
Auto
All Output
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Stanford CS193p
Fall 2013

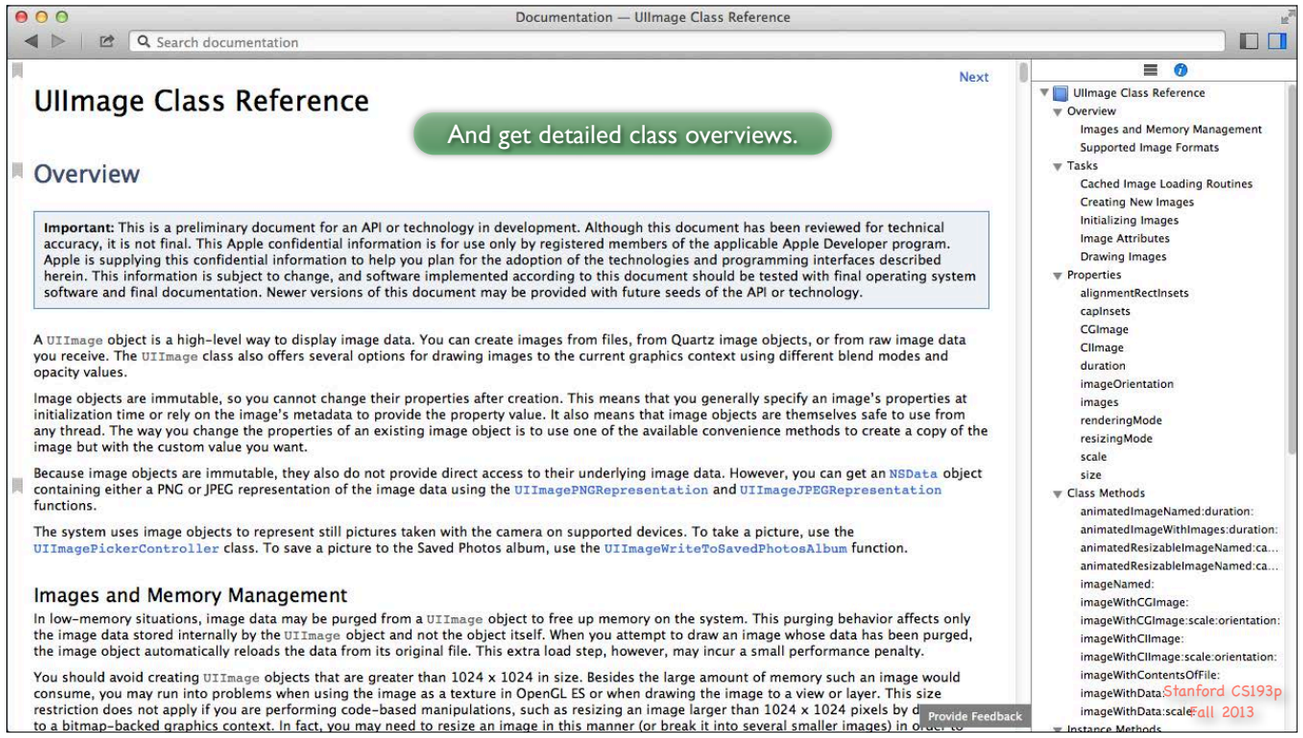

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Stop!
Automatic › CardGameViewController.m › -touchCardButton:
Let's take a little timeout to talk about documentation.
There are numerous ways to transition to the documentation, but an easy one is to use the ALT key.
Hold down the ALT key and hover your mouse over something like currentTitle.
A dashed line should appear underneath and the cursor should be a question mark.
Stanford CS193p
Fall 2013

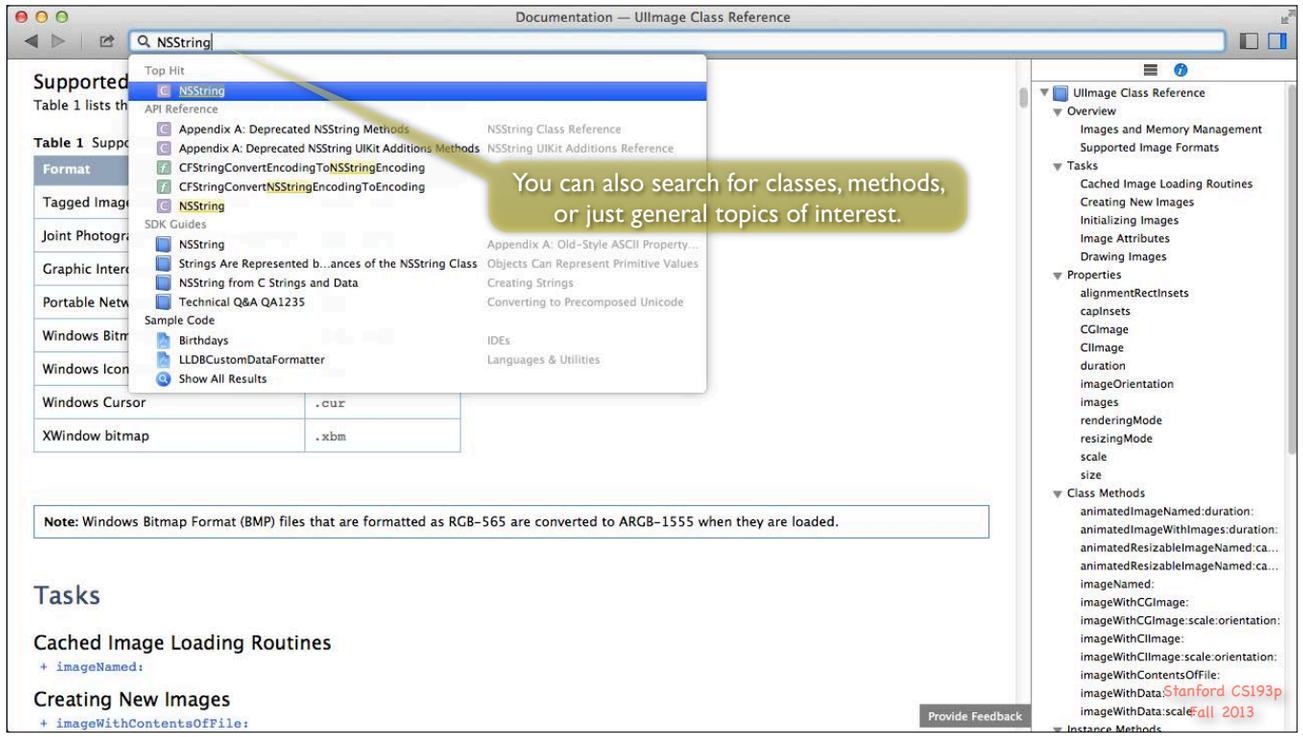

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
];
cardfront"]
mal];
ALT-clicking on currentTitle will bring up this "mini-documentation" in-line.
Declaration @property(nonatomic, readonly, retain) NSString * currentTitle
Description The current title that is displayed on the button. (read-only)
The value for this property is set automatically whenever the button state changes. For states that do not have a custom title string associated with them, this method returns the title that is currently displayed, which is typically the one associated with the UIControlStateNormal state. The value may be nil.
Availability iOS (2.0 and later)
Declared In UIButton.h
Reference UIButton Class Reference
retain is the same as strong.
We'll cover the readonly directive next lecture.
Stanford CS193p
Fall 2013

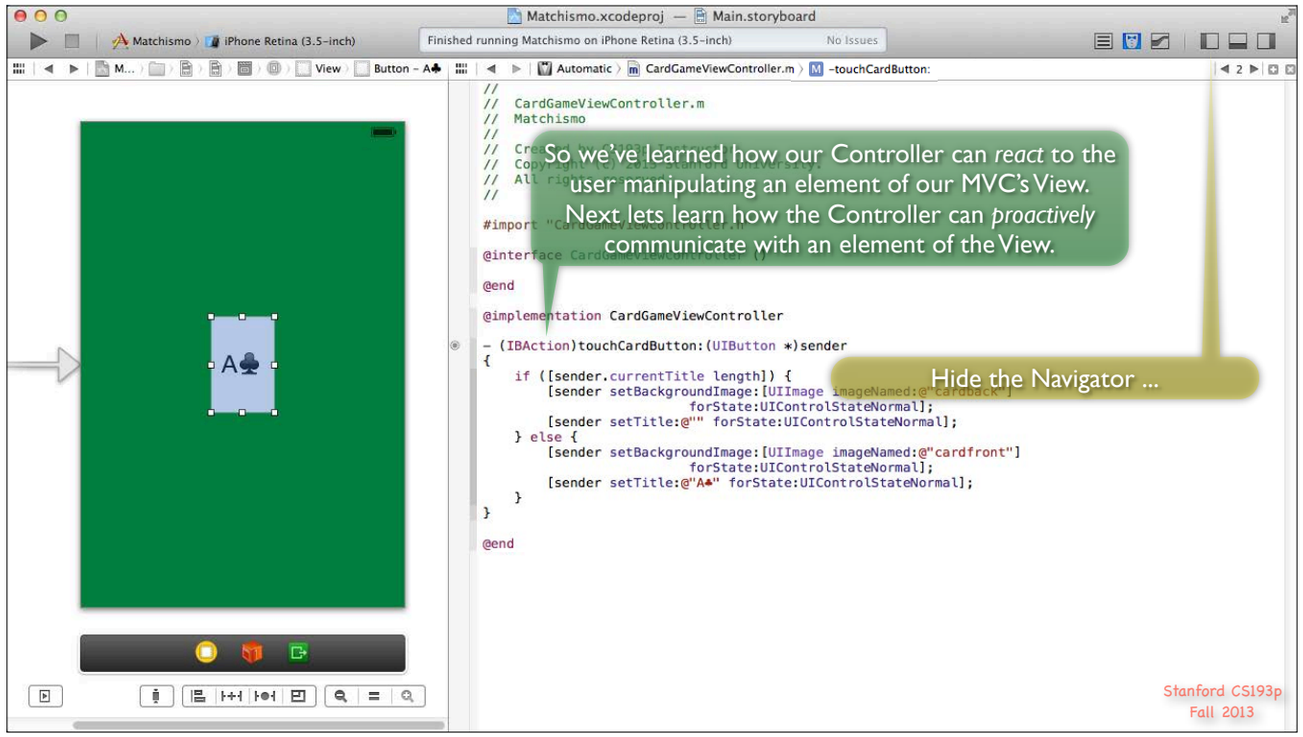

Matchismo.xcodeproj — Main.storyboard
TIP: If you ever accidentally navigate away from your source code, you can click on this back button to get back.
Click on the UIButton Class Reference link in this little "mini-documentation" window to get more detailed documentation.
You can also hold down the ALT key and double-click on a term go directly to the documentation.
Declaration
Description
Availability
Declared In
Reference
UIButton Class Reference
Stanford CS193p
Fall 2013

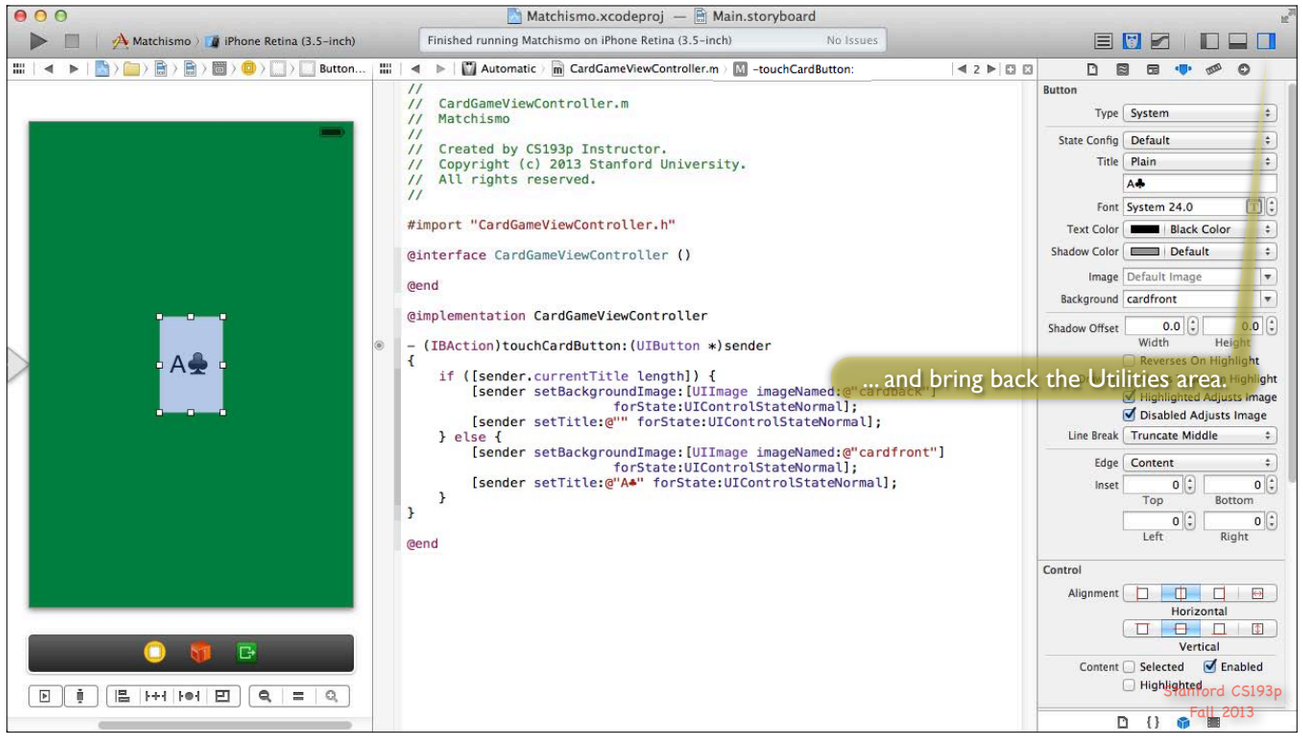

Documentation — UIButton Class Reference
Search documentation
This is the Documentation window.
Overview
Important: This is a preliminary document for an API or technology in development. Although this document has been reviewed for technical accuracy, it is not final. This Apple confidential information is for use only by registered members of the applicable Apple Developer program. Apple is supplying this confidential information to help you plan for the adoption of the technologies and programming interfaces described herein. This information is subject to change, and software implemented according to this document should be tested with final operating system software and final documentation. Newer versions of this document may be provided with future seeds of the API or technology.
An instance of the UIButton class implements a button on the touch screen. A button intercepts touch events and sends an action message to a target object when tapped. Methods for setting the target and action are inherited from UIControl. This class provides methods for setting the title, image, and other appearance properties of a button. By using these accessors, you can specify a different appearance for each button state.
For information about basic view behaviors, see View Programming Guide for iOS.
For more information about appearance and behavior configuration, see "Buttons".
Tasks
Creating Buttons
+ buttonWithType:
Configuring the Button Title
titleLabel property
reversesTitleShadowWhenHighlighted property
– setTitle:forState:
– setAttributedTitle:forState:
– setTitleColor:forState:
– setTitleShadowColor:forState:
– titleColorForState:
– titleForState:
– attributedTitleForState:
– titleShadowColorForState:
Configuring Button Presentation
You should explore what is here.
It is substantial.
Being able to maneuver through the documentation is critical to success in iOS Development.
Provide Feedback
UIButton Class Reference
Overview
Tasks
Creating Buttons
Configuring the Button Title
Configuring Button Presentation
Configuring Edge Insets
Getting the Current State
Getting Dimensions
Deprecated Properties
Properties
adjustsImageWhenDisabled
adjustsImageWhenHighlighted
buttonType
contentEdgeInsets
currentAttributedTitle
currentBackgroundImage
currentImage
currentTitle
currentTitleColor
currentTitleShadowColor
imageEdgeInsets
imageView
reversesTitleShadowWhenHighlighted
showsTouchWhenHighlighted
tintColor
titleEdgeInsets
titleLabel
Class Methods
buttonWithType:
Instance Methods
attributedTitleForState:
backgroundImageForState:
backgroundRectForBounds:
contentRectForBounds:
imageForState:
Stanford CS193p
Fall 2013

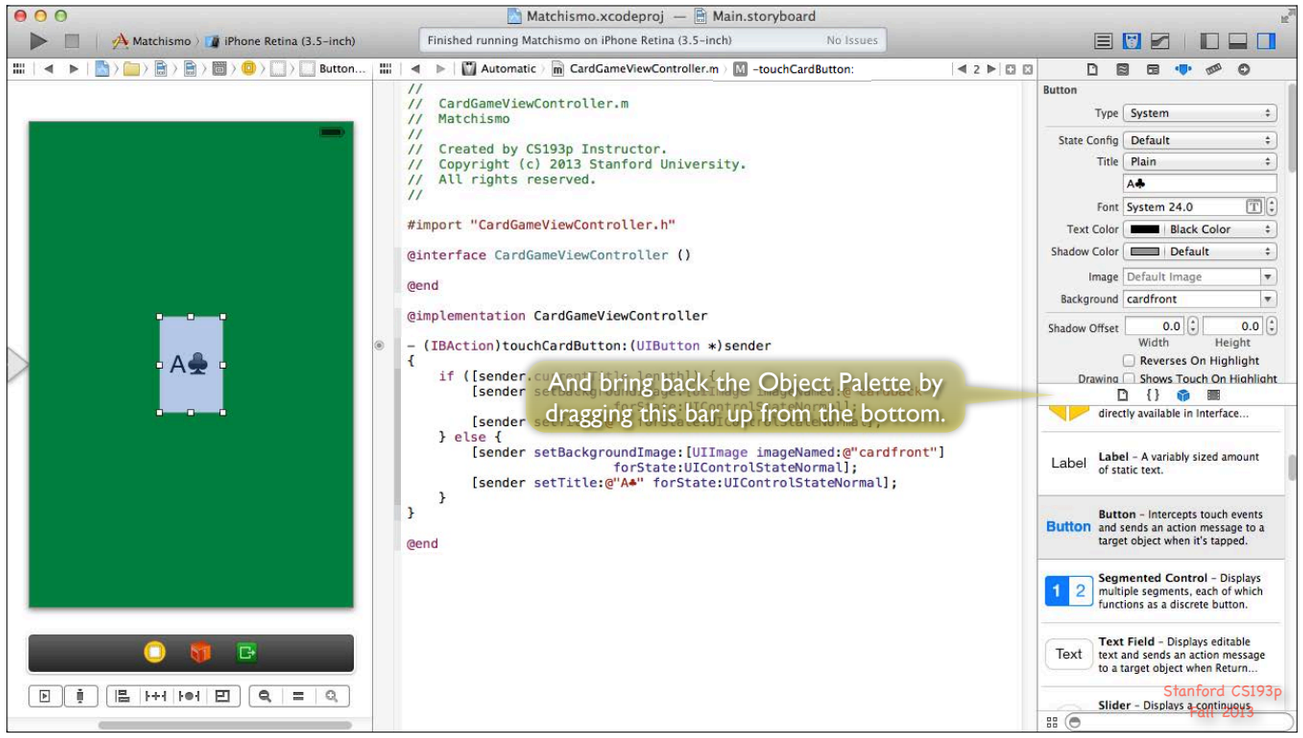

Documentation — UIButton Class Reference
Search documentation
Available in iOS 6.0 and later.
Declared In
UIButton.h
setBackgroundImage:forState:
Sets the background image to use for the specified button state.
- (void)setBackgroundImage:(UIImage *)image forState:(UIControlState)state
Parameters
image
The background image to use for the specified state.
state
The state that uses the specified image. The values are described in UIControlState.
Discussion
In general, if a property is not specified for a state, the default is to use the UIControlStateNormal value. If the UIControlStateNormal value is not set, then the property defaults to a system value. Therefore, at a minimum, you should set the value for the normal state.
Availability
Available in iOS 2.0 and later.
See Also
– backgroundImageForState:
Related Sample Code
Accessory
AddMusic
UICatalog
Declared In
UIButton.h
setImage:forState:
Sets the image to use for the specified state.
- (void)setImage:(UIImage *)image forState:(UIControlState)state
Parameters
Provide Feedback
For example, scroll down to setBackgroundImage:forState:.
You can click on the many links here, like UIImage ...
UIButton Class Reference
Overview
Tasks
Creating Buttons
Configuring the Button Title
Configuring Button Presentation
Configuring Edge Insets
Getting the Current State
Getting Dimensions
Deprecated Properties
Properties
adjustsImageWhenDisabled
adjustsImageWhenHighlighted
buttonType
contentEdgeInsets
currentAttributedTitle
currentBackgroundImage
currentImage
currentTitle
currentTitleColor
currentTitleShadowColor
imageEdgeInsets
imageView
reversesTitleShadowWhenHighlighted
showsTouchWhenHighlighted
tintColor
titleEdgeInsets
titleLabel
Class Methods
buttonWithType:
Instance Methods
attributedTitleForState:
backgroundImageForState:
backgroundRectForBounds:
contentRectForBounds:
imageForState:

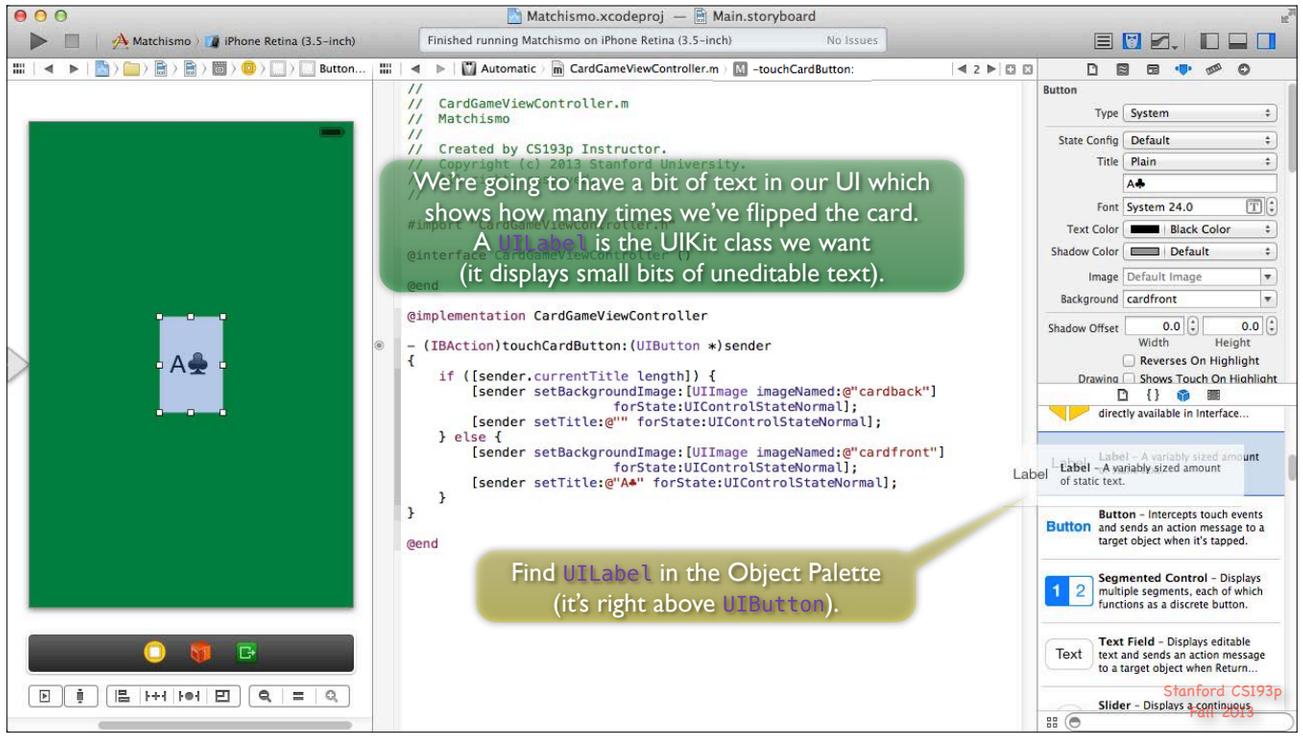

Documentation — UIImage Class Reference
Search documentation
Next
UIImage Class Reference
And get detailed class overviews.
Overview
Important: This is a preliminary document for an API or technology in development. Although this document has been reviewed for technical accuracy, it is not final. This Apple confidential information is for use only by registered members of the applicable Apple Developer program. Apple is supplying this confidential information to help you plan for the adoption of the technologies and programming interfaces described herein. This information is subject to change, and software implemented according to this document should be tested with final operating system software and final documentation. Newer versions of this document may be provided with future seeds of the API or technology.
A UIImage object is a high-level way to display image data. You can create images from files, from Quartz image objects, or from raw image data you receive. The UIImage class also offers several options for drawing images to the current graphics context using different blend modes and opacity values.
Image objects are immutable, so you cannot change their properties after creation. This means that you generally specify an image's properties at initialization time or rely on the image's metadata to provide the property value. It also means that image objects are themselves safe to use from any thread. The way you change the properties of an existing image object is to use one of the available convenience methods to create a copy of the image but with the custom value you want.
Because image objects are immutable, they also do not provide direct access to their underlying image data. However, you can get an NSData object containing either a PNG or JPEG representation of the image data using the UIImagePNGRepresentation and UIImageJPEGRepresentation functions.
The system uses image objects to represent still pictures taken with the camera on supported devices. To take a picture, use the UIImagePickerController class. To save a picture to the Saved Photos album, use the UIImageWriteToSavedPhotosAlbum function.
Images and Memory Management
In low-memory situations, image data may be purged from a UIImage object to free up memory on the system. This purging behavior affects only the image data stored internally by the UIImage object and not the object itself. When you attempt to draw an image whose data has been purged, the image object automatically reloads the data from its original file. This extra load step, however, may incur a small performance penalty.
You should avoid creating UIImage objects that are greater than 1024 x 1024 in size. Besides the large amount of memory such an image would consume, you may run into problems when using the image as a texture in OpenGL ES or when drawing the image to a view or layer. This size restriction does not apply if you are performing code-based manipulations, such as resizing an image larger than 1024 x 1024 pixels by d
to a bitmap-backed graphics context. In fact, you may need to resize an image in this manner (or break it into several smaller images) in o
Provide Feedback
UIImage Class Reference
Overview
Images and Memory Management
Supported Image Formats
Tasks
Cached Image Loading Routines
Creating New Images
Initializing Images
Image Attributes
Drawing Images
Properties
alignmentRectInsets
capInsets
CGImage
CIImage
duration
imageOrientation
images
renderingMode
resizingMode
scale
size
Class Methods
animatedImageNamed:duration:
animatedImageWithImages:duration:
animatedResizableImageNamed:ca...
animatedResizableImageNamed:ca...
imageNamed:
imageWithCGImage:
imageWithCGImage:scale:orientation:
imageWithCIImage:
imageWithCIImage:scale:orientation:
imageWithContentsOfFile:
imageWithData:
imageWithData:scale:
Stanford CS193p
Fall 2013

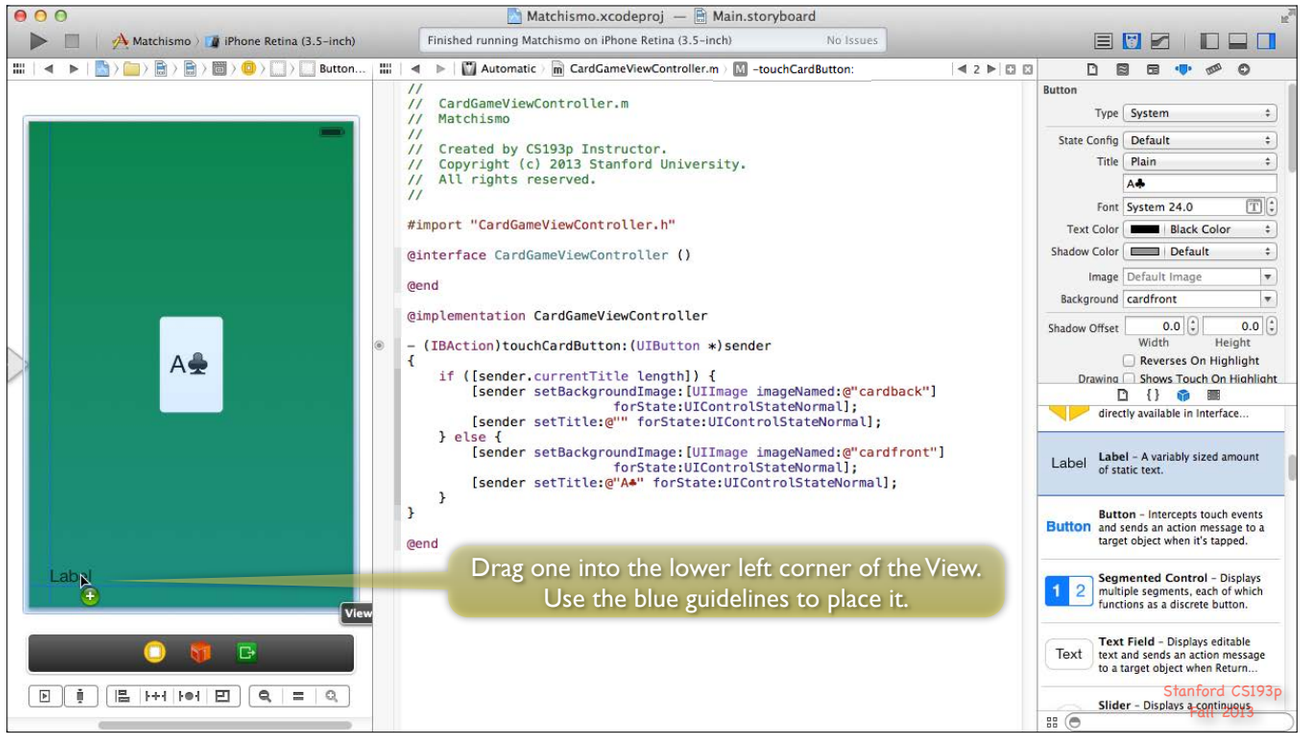

Documentation — UIImage Class Reference
NSString
Top Hit
NSString
API Reference
Appendix A: Deprecated NSString Methods
NSString Class Reference
Appendix A: Deprecated NSString UIKit Additions Methods
NSString UIKit Additions Reference
CFStringConvertEncodingToNSStringEncoding
CFStringConvertNSStringEncodingToEncoding
NSString
SDK Guides
NSString
Appendix A: Old-Style ASCII Property...
Strings Are Represented b...ances of the NSString Class
Objects Can Represent Primitive Values
NSString from C Strings and Data
Creating Strings
Technical Q&A QA1235
Converting to Precomposed Unicode
Sample Code
Birthdays
IDEs
LLDBCustomDataFormatter
Languages & Utilities
Show All Results
You can also search for classes, methods, or just general topics of interest.
Supported
Table 1 lists th
Table 1 Suppo
Format
Tagged Image
Joint Photogra
Graphic Interc
Portable Netw
Windows Bitm
Windows Icon
Windows Cursor
.cur
XWindow bitmap
.xbm
Note: Windows Bitmap Format (BMP) files that are formatted as RGB-565 are converted to ARGB-1555 when they are loaded.
Tasks
Cached Image Loading Routines
+ imageNamed:
Creating New Images
+ imageWithContentsOfFile:
Provide Feedback
UIImage Class Reference
Overview
Images and Memory Management
Supported Image Formats
Tasks
Cached Image Loading Routines
Creating New Images
Initializing Images
Image Attributes
Drawing Images
Properties
alignmentRectInsets
capInsets
CGImage
CIImage
duration
imageOrientation
images
renderingMode
resizingMode
scale
size
Class Methods
animatedImageNamed:duration:
animatedImageWithImages:duration:
animatedResizableImageNamed:ca...
animatedResizableImageNamed:ca...
imageNamed:
imageWithCGImage:
imageWithCGImage:scale:orientation:
imageWithCIImage:
imageWithCIImage:scale:orientation:
imageWithContentsOfFile:
imageWithData:
imageWithData:scale:
Instance Methods
Stanford CS193p
Fall 2013

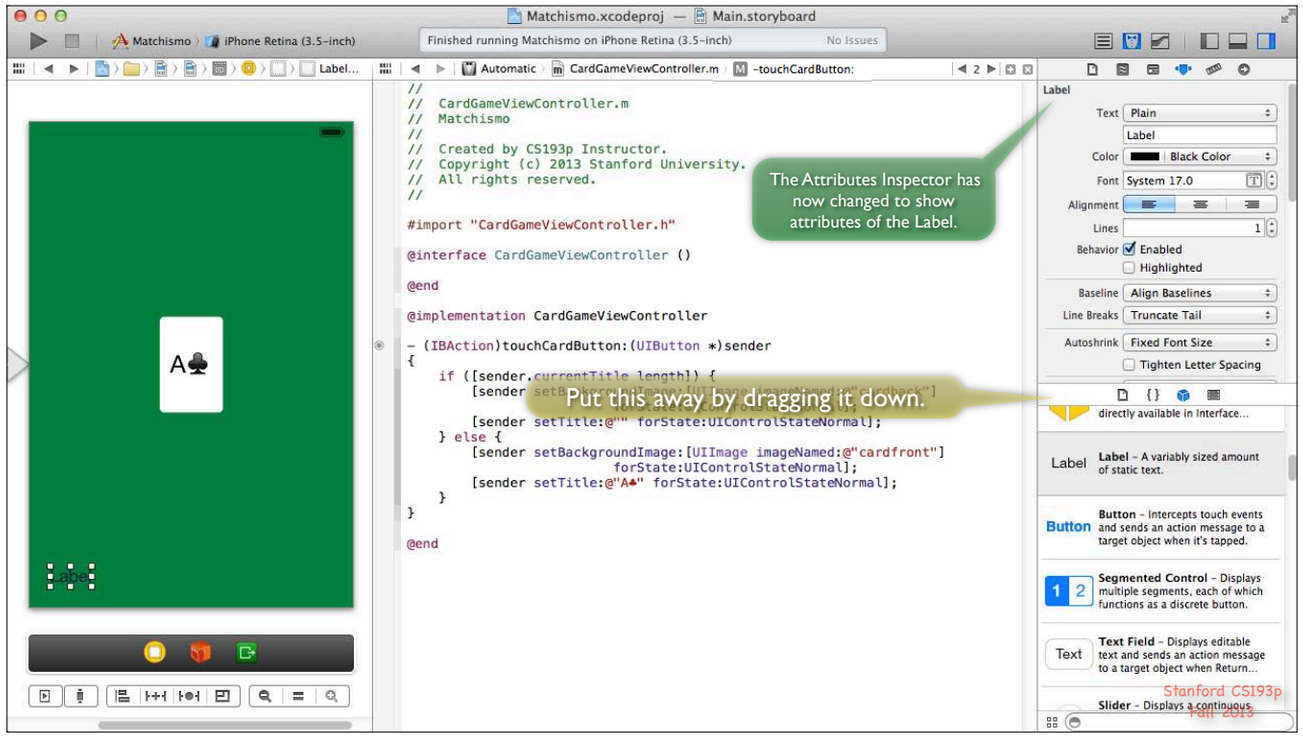

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
View › Button – A♣
Automatic › CardGameViewController.m › -touchCardButton:
A♣
//
// CardGameViewController.m
// Matchismo
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
So we've learned how our Controller can react to the user manipulating an element of our MVC's View. Next lets learn how the Controller can proactively communicate with an element of the View.
Hide the Navigator ...
Stanford CS193p
Fall 2013

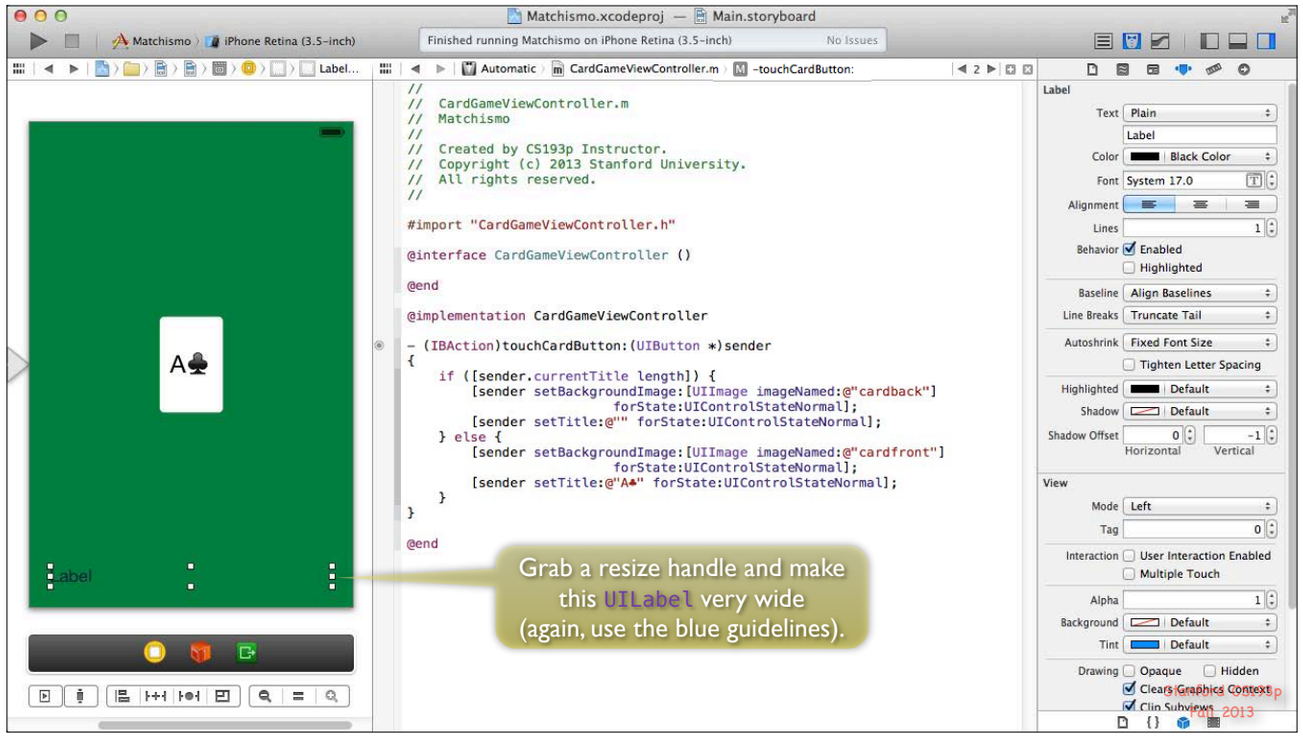

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
...and bring back the Utilities area.
Stanford CS193p
Fall 2013

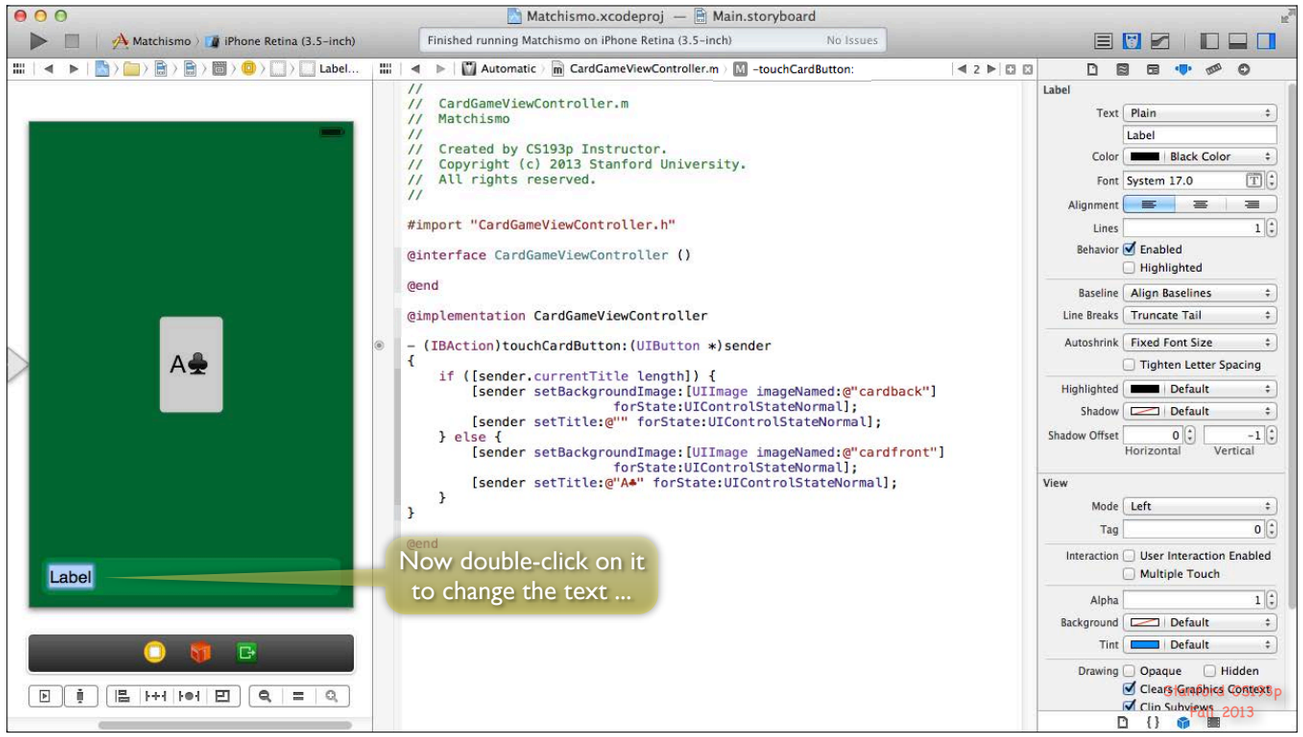

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.
[sender
[sender
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
And bring back the Object Palette by dragging this bar up from the bottom.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
directly available in Interface...
Label Label – A variably sized amount of static text.
Button Button – Intercepts touch events and sends an action message to a target object when it's tapped.
1 2 Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

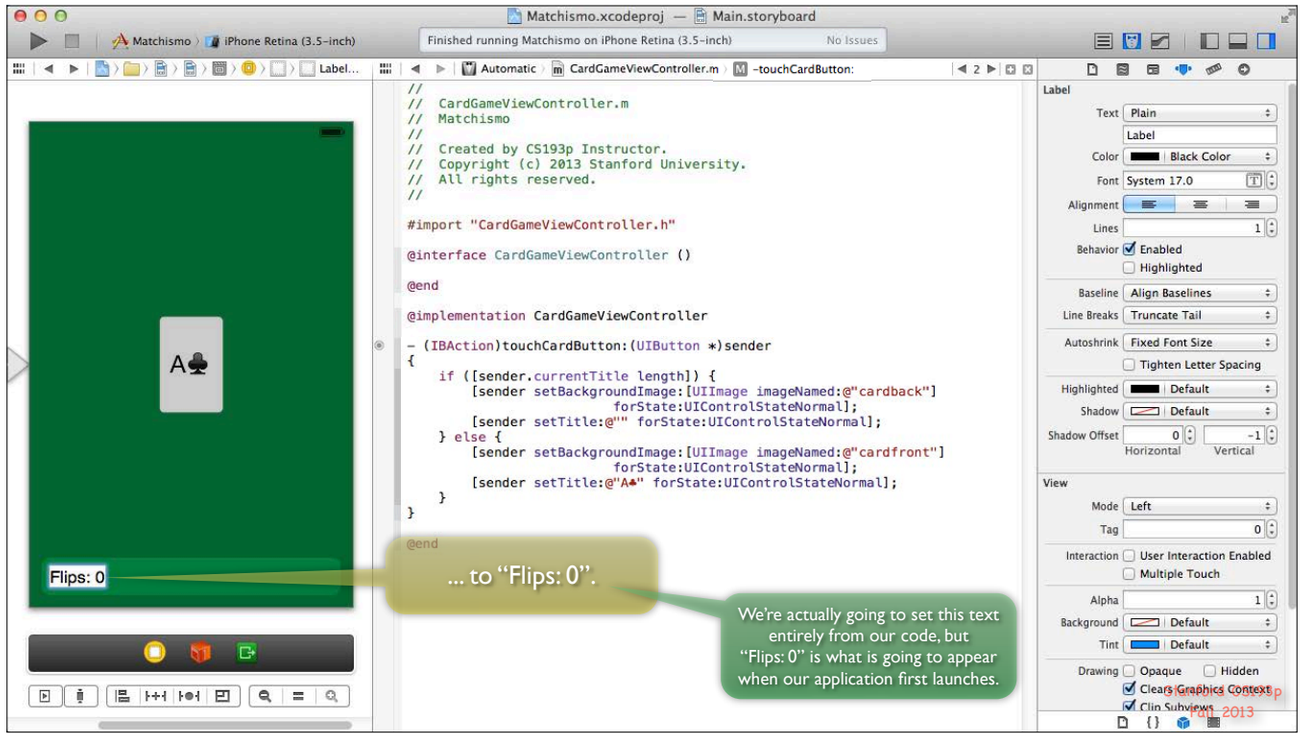

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
//
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
We're going to have a bit of text in our UI which shows how many times we've flipped the card.
A UILabel is the UIKit class we want (it displays small bits of uneditable text).
Find UILabel in the Object Palette (it's right above UIButton).
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Shows Touch On Highlight
Label
Label – A variably sized amount of static text.
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

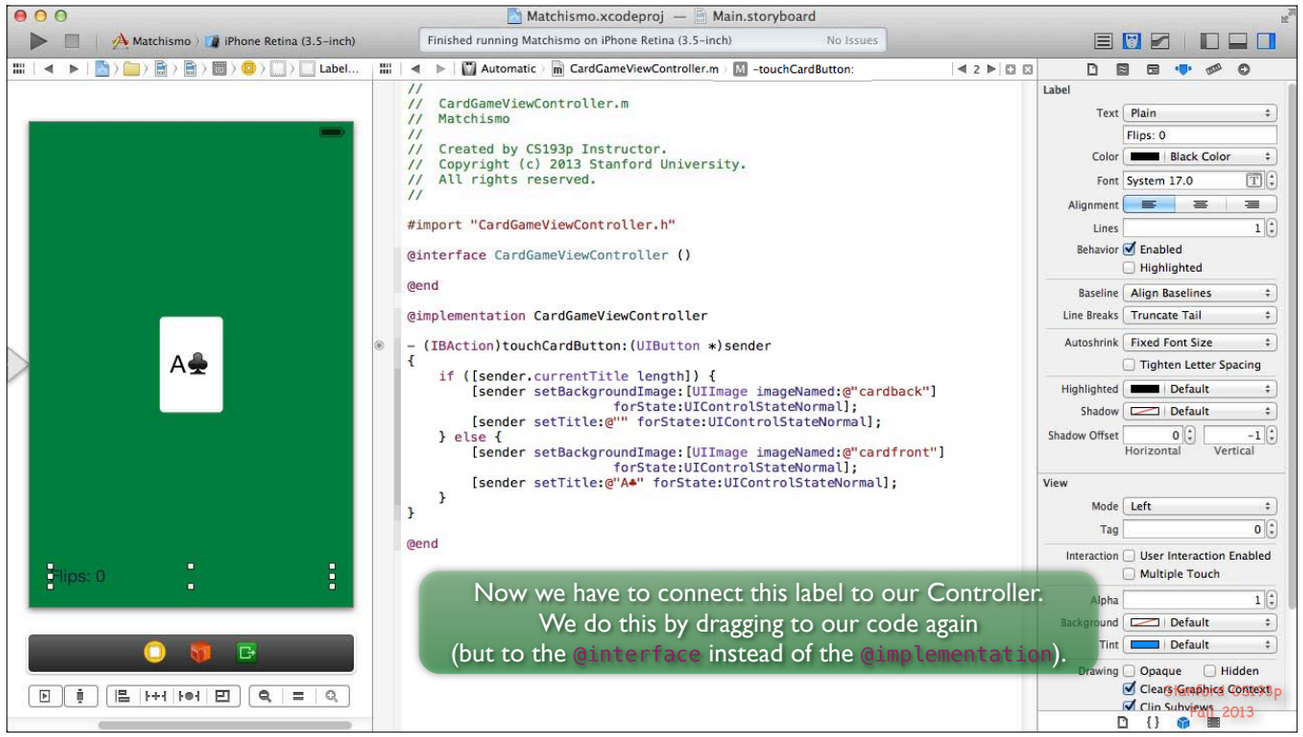

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Button...
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
A♣
Label
View
Drag one into the lower left corner of the View.
Use the blue guidelines to place it.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
directly available in Interface...
Label Label – A variably sized amount of static text.
Button Button – Intercepts touch events and sends an action message to a target object when it's tapped.
1 2 Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

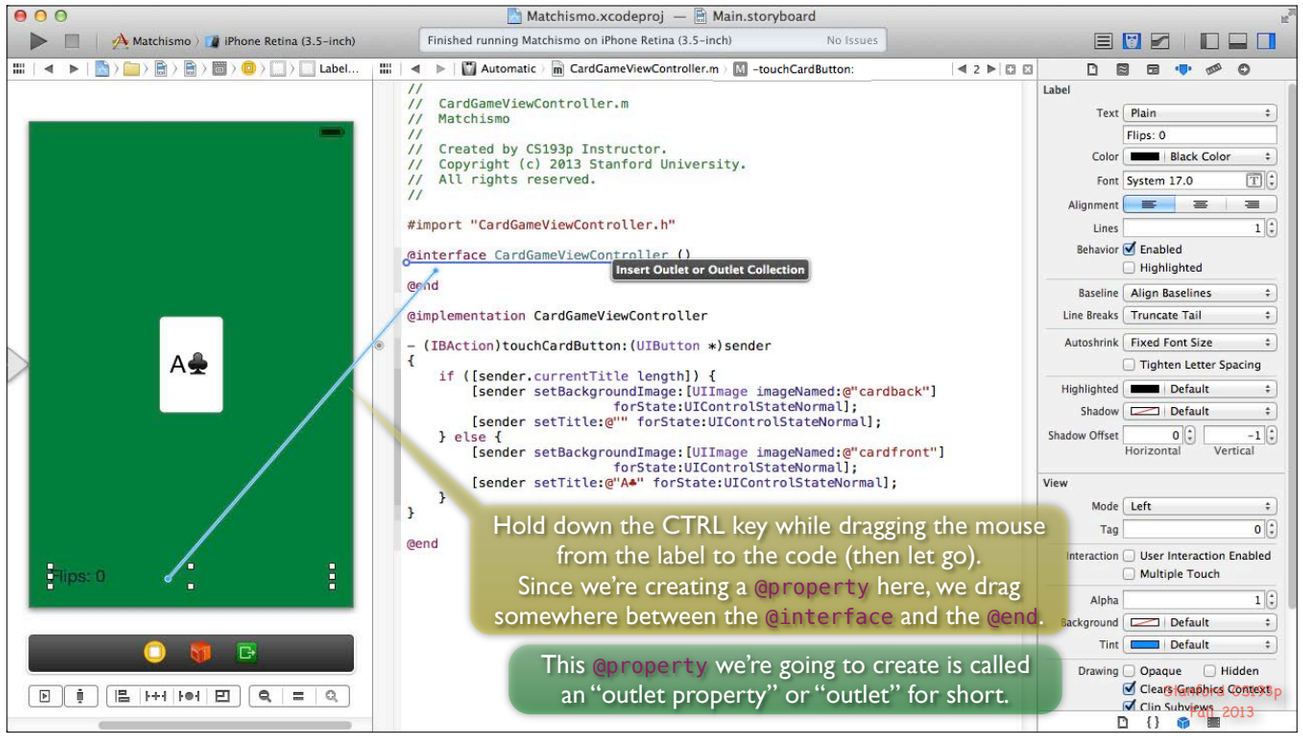

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
The Attributes Inspector has now changed to show attributes of the Label.
Put this away by dragging it down.
Label
Text Plain
Label
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
directly available in Interface...
Label – A variably sized amount of static text.
Button – Intercepts touch events and sends an action message to a target object when it's tapped.
Segmented Control – Displays multiple segments, each of which functions as a discrete button.
Text Field – Displays editable text and sends an action message to a target object when Return...
Slider – Displays a continuous
Stanford CS193p
Fall 2013

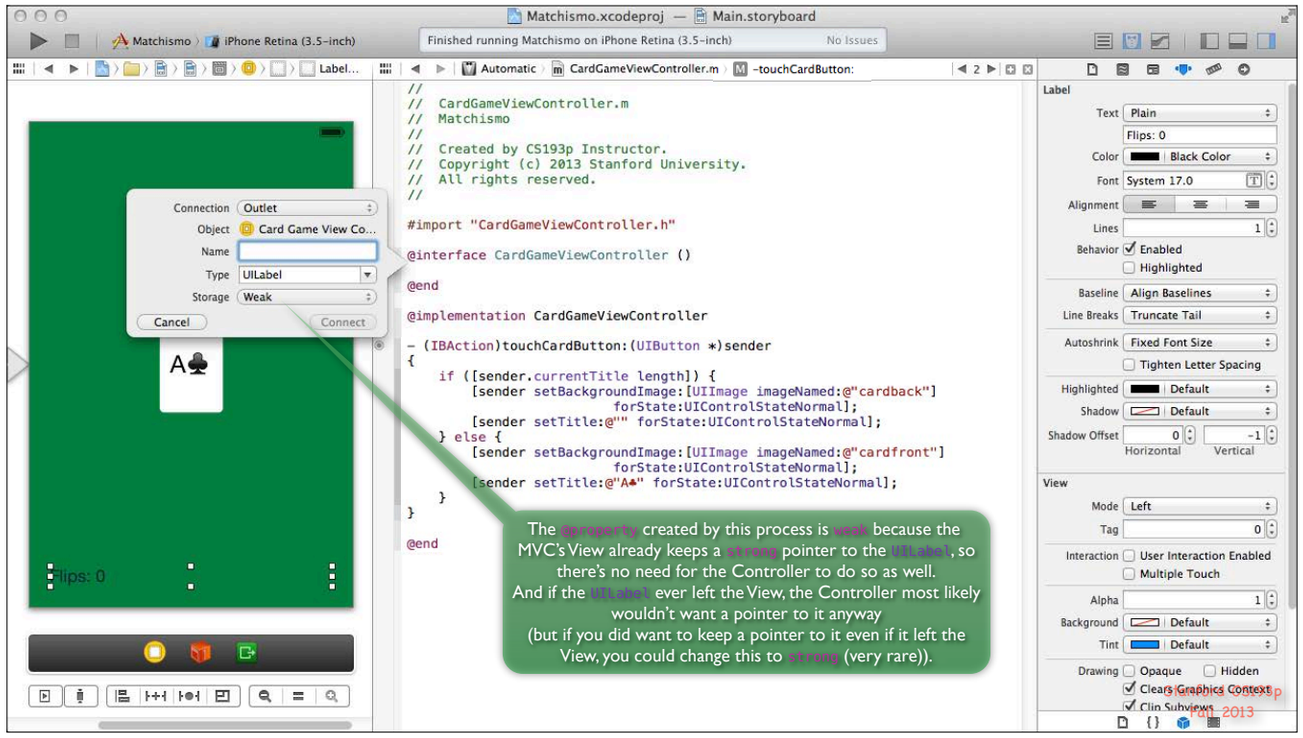

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Label...
Automatic
CardGameViewController.m
-touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
A♣
abel
Grab a resize handle and make this UILabel very wide (again, use the blue guidelines).
Label
Text Plain
Label
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

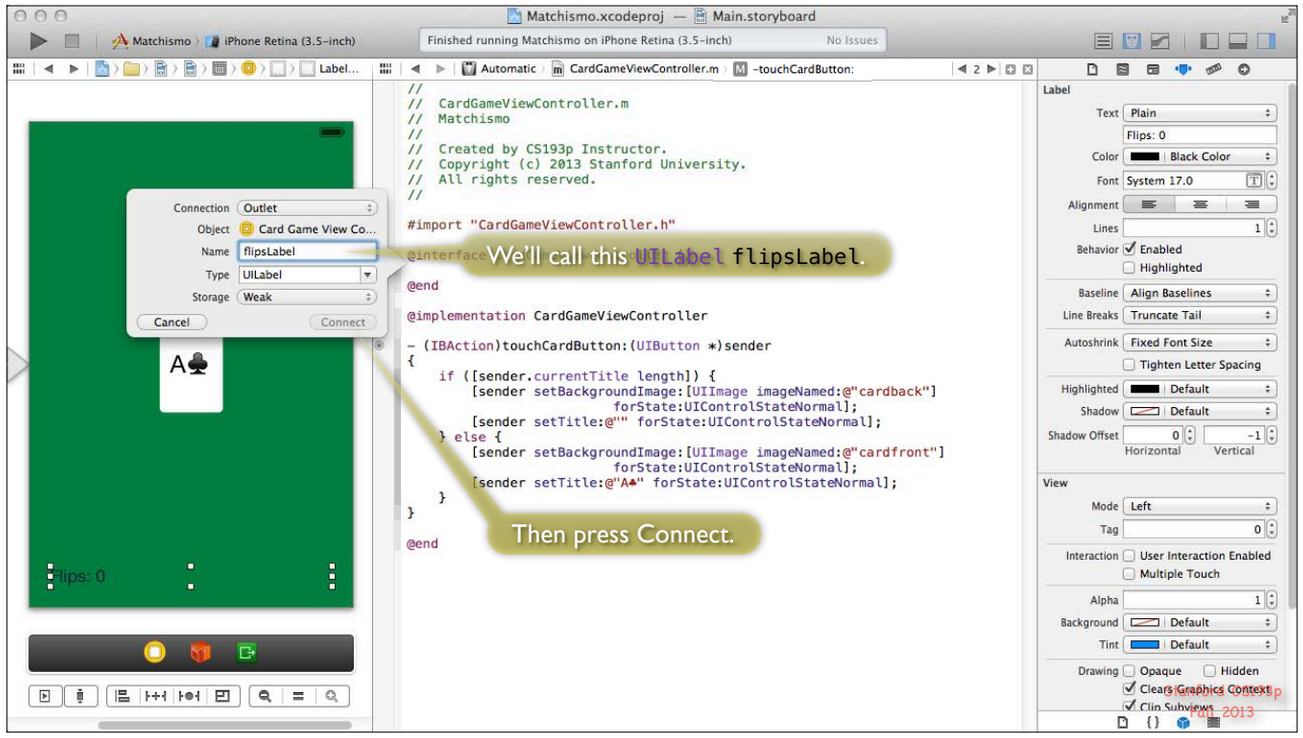

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Label...
Automatic › CardGameViewController.m › -touchCardButton:
A♣
Label
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Now double-click on it to change the text ...
Label
Text Plain
Label
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

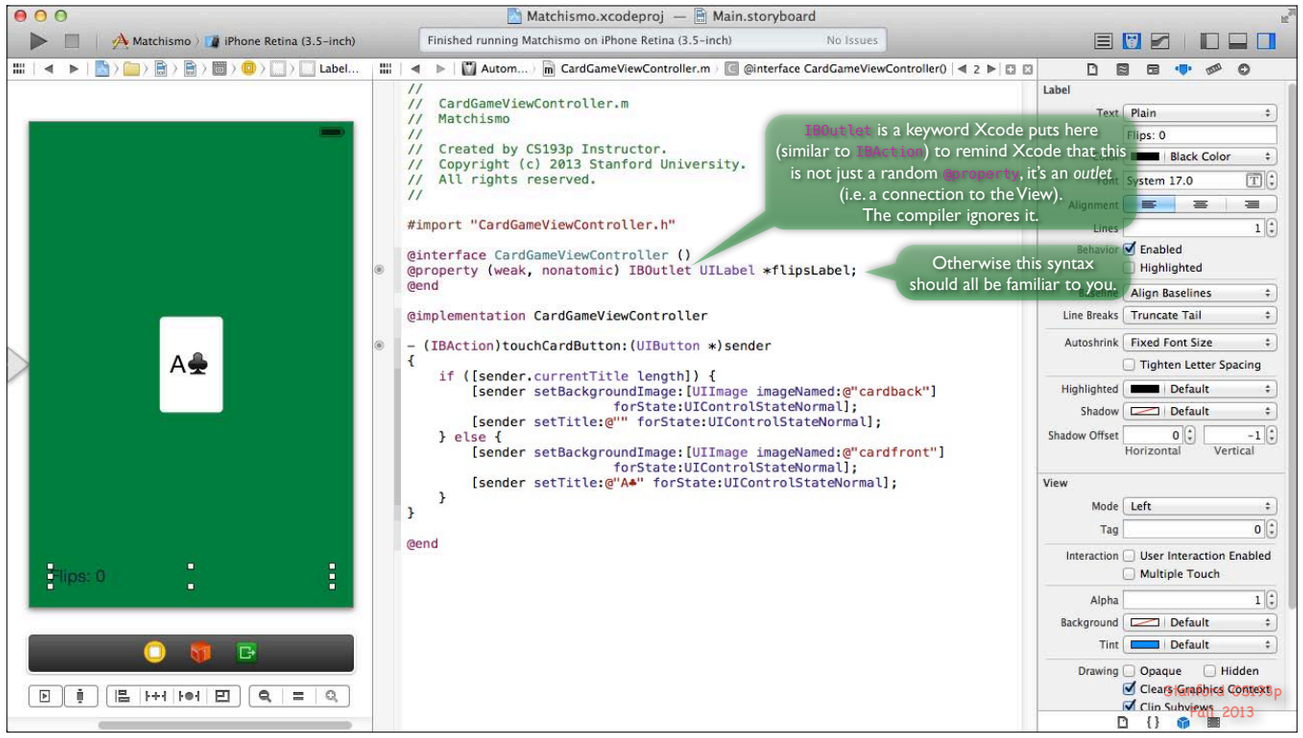

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
A♣
Flips: 0
... to "Flips: 0".
We're actually going to set this text entirely from our code, but "Flips: 0" is what is going to appear when our application first launches.
Label
Text Plain
Label
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

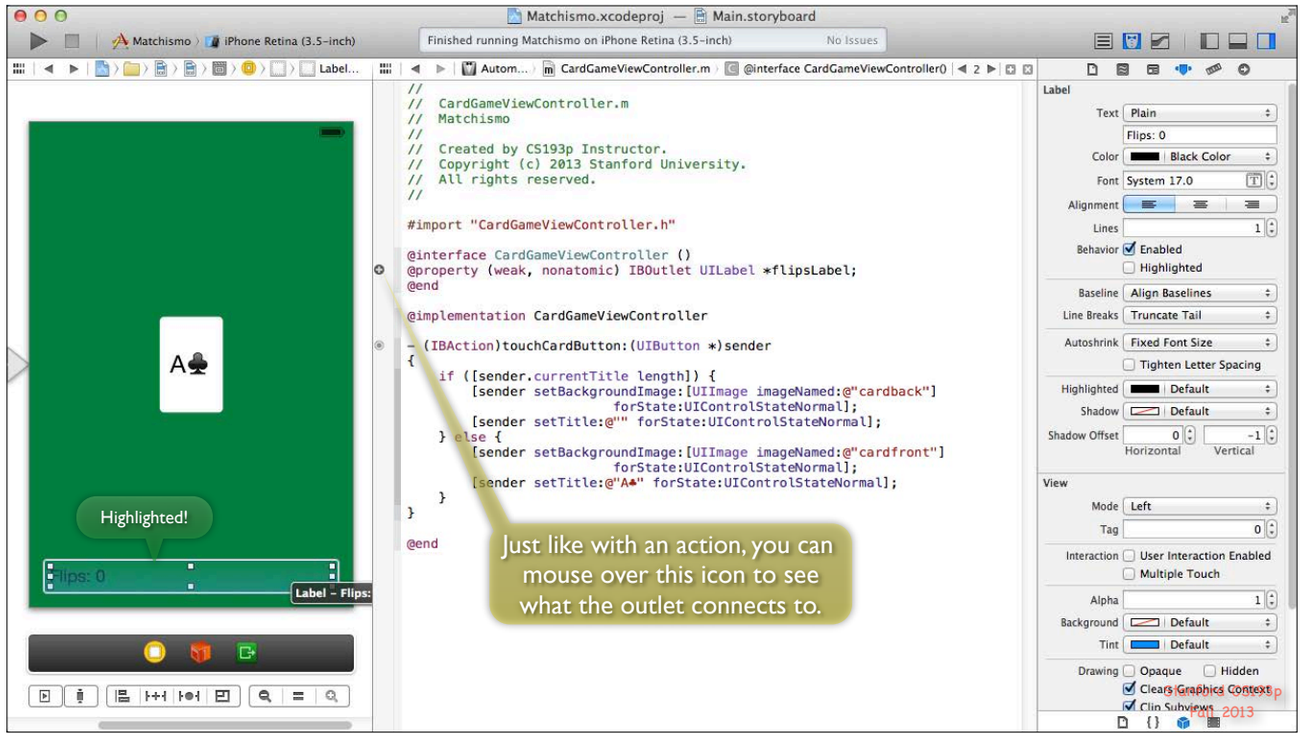

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
A♣
ips: 0
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Now we have to connect this label to our Controller.
We do this by dragging to our code again
(but to the @interface instead of the @implementation).

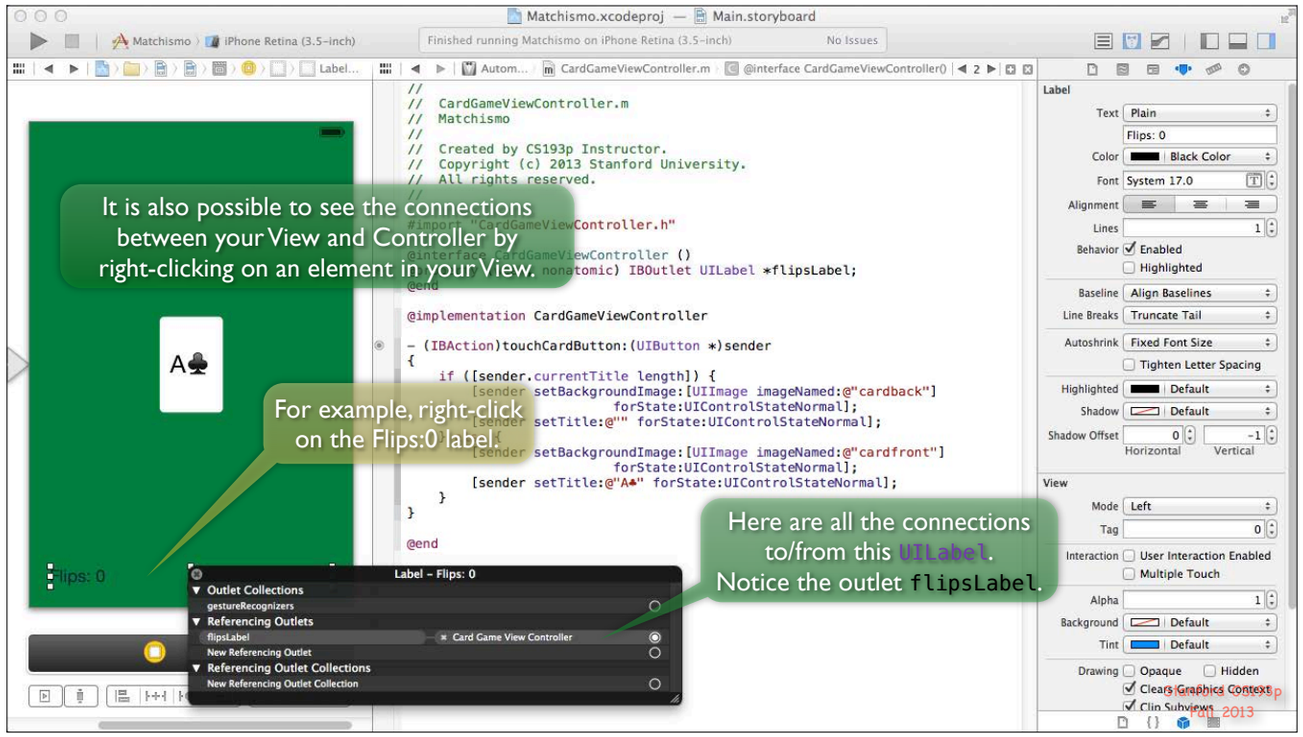

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
Insert Outlet or Outlet Collection
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
A♣
ips: 0
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Hold down the CTRL key while dragging the mouse from the label to the code (then let go).
Since we're creating a @property here, we drag somewhere between the @interface and the @end.
This @property we're going to create is called an "outlet property" or "outlet" for short.
Stanford CS193p Fall 2013

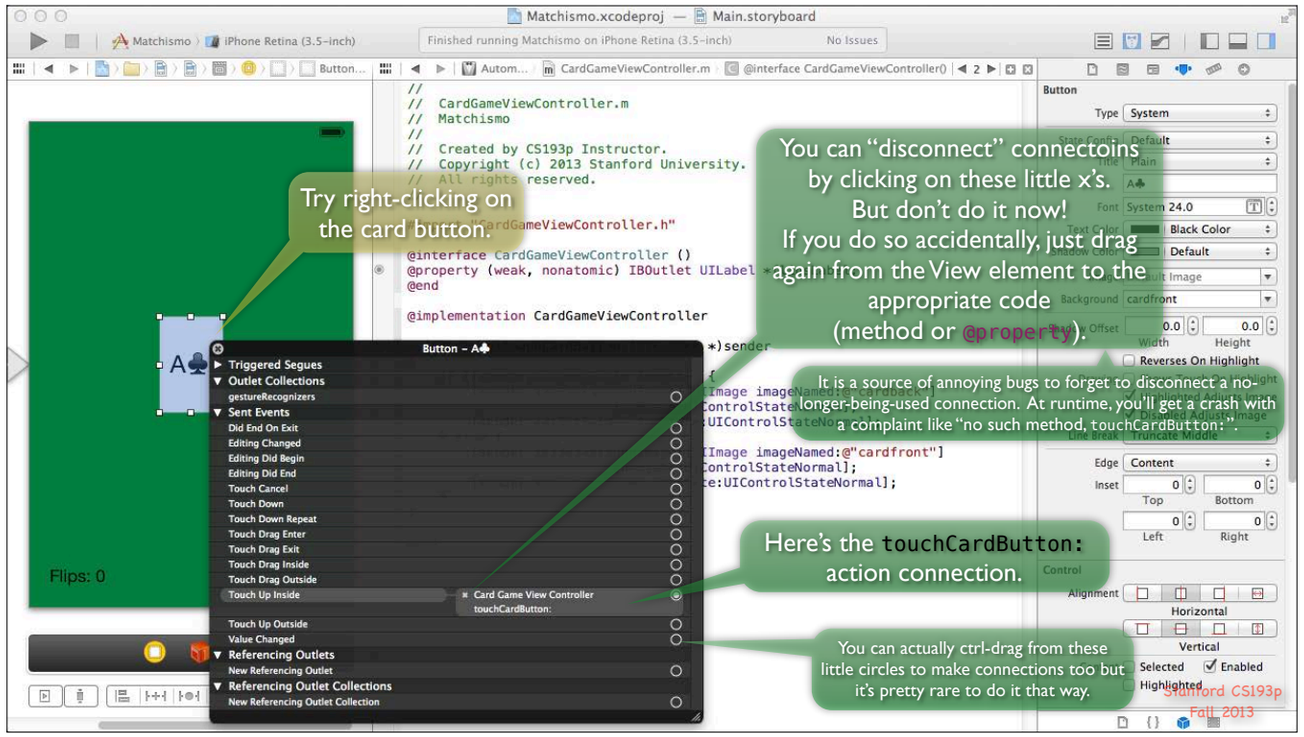

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
Connection Outlet
Object Card Game View Co...
Name
Type UILabel
Storage Weak
Cancel
Connect
A♣
ips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
The @property created by this process is weak because the MVC's View already keeps a strong pointer to the UILabel, so there's no need for the Controller to do so as well.
And if the UILabel ever left the View, the Controller most likely wouldn't want a pointer to it anyway
(but if you did want to keep a pointer to it even if it left the View, you could change this to strong (very rare)).
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

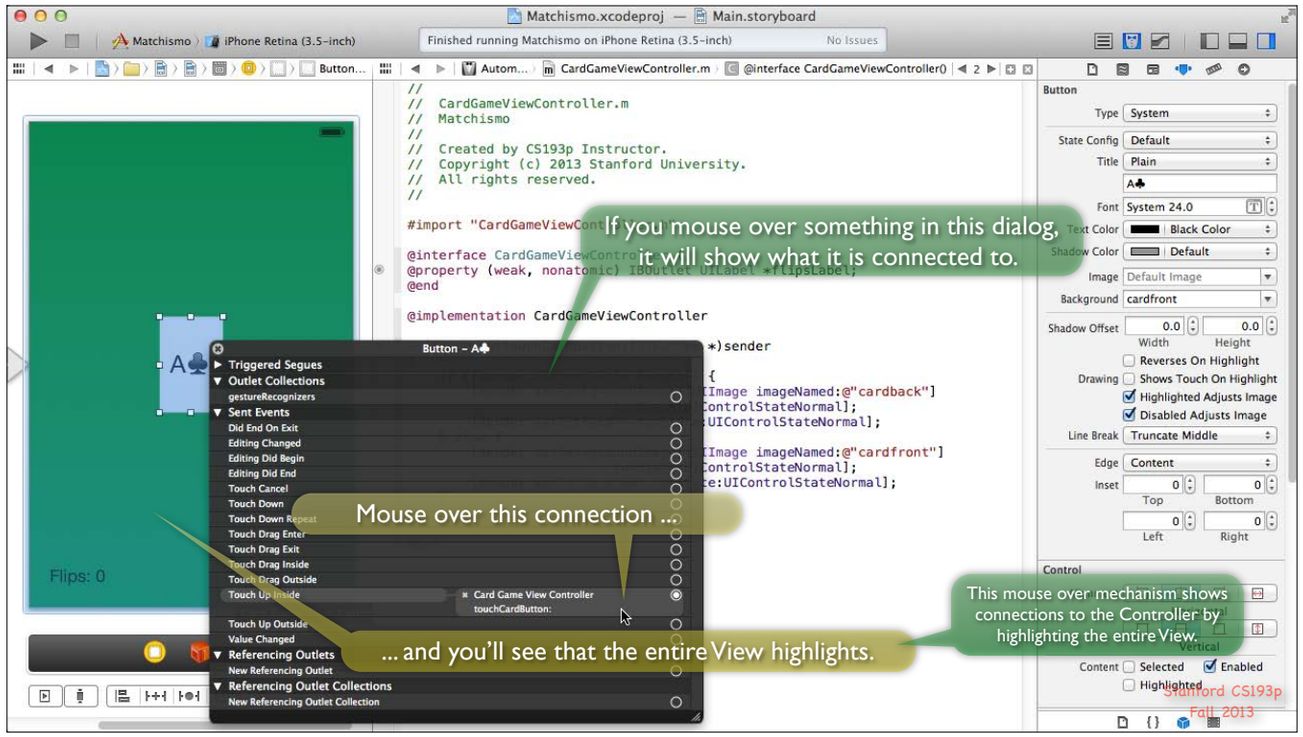

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -touchCardButton:
Connection Outlet
Object Card Game View Co...
Name flipsLabel
Type UILabel
Storage Weak
Cancel
Connect
We'll call this UILabel flipsLabel.
Then press Connect.
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

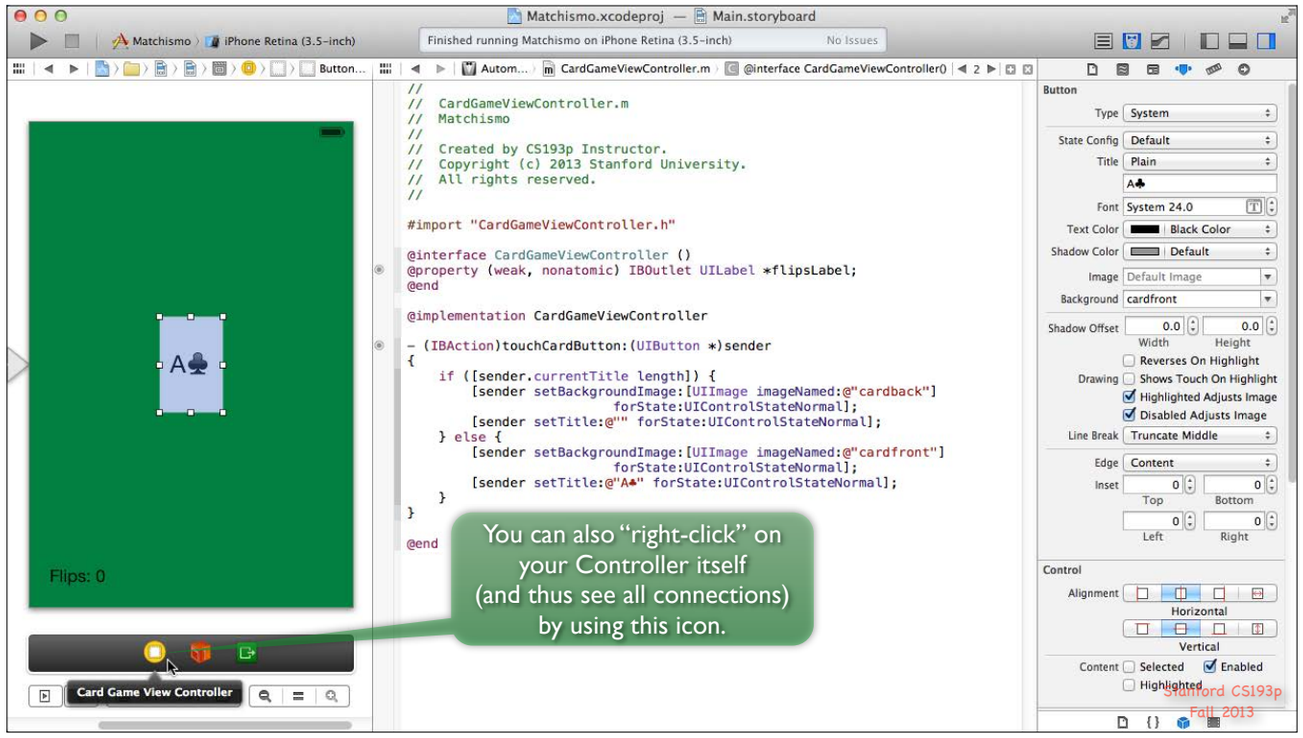

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
IBOutlet is a keyword Xcode puts here (similar to IBAction) to remind Xcode that this is not just a random @property, it's an outlet (i.e. a connection to the View). The compiler ignores it.
Otherwise this syntax should all be familiar to you.
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

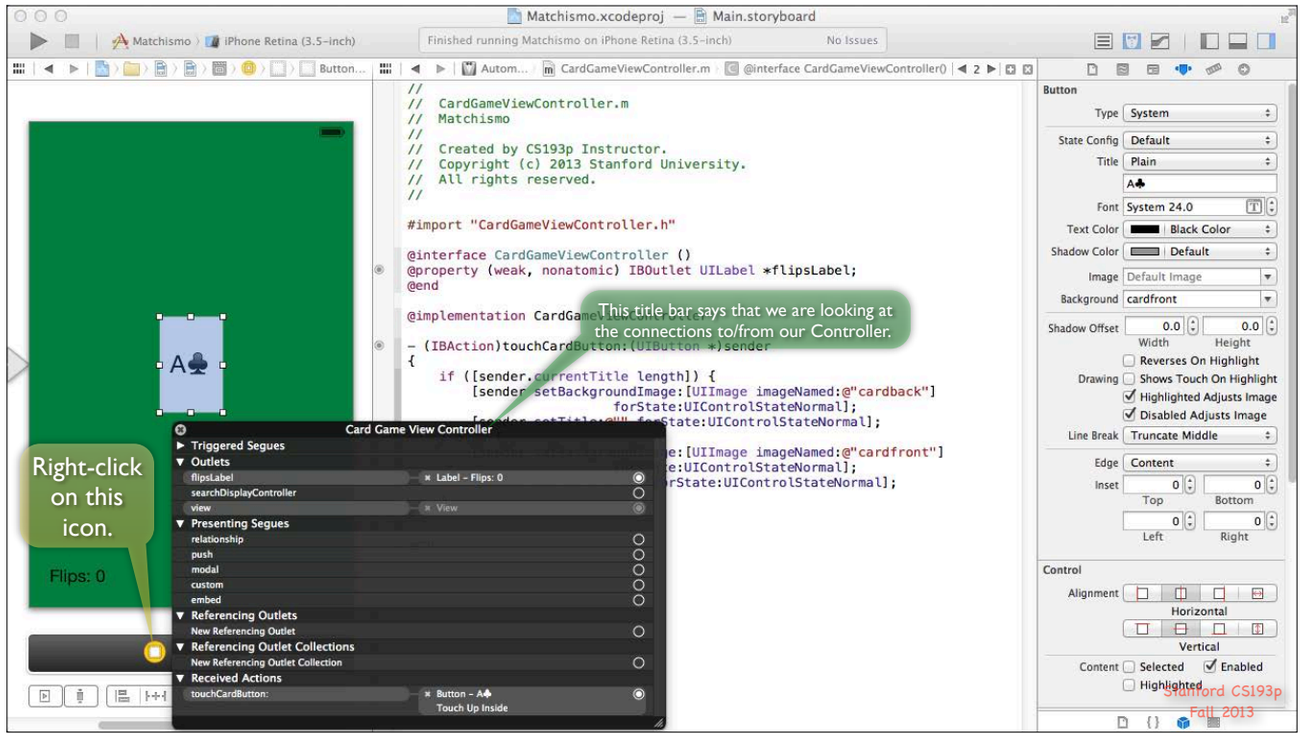

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Highlighted!
Label – Flips:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Just like with an action, you can mouse over this icon to see what the outlet connects to.
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

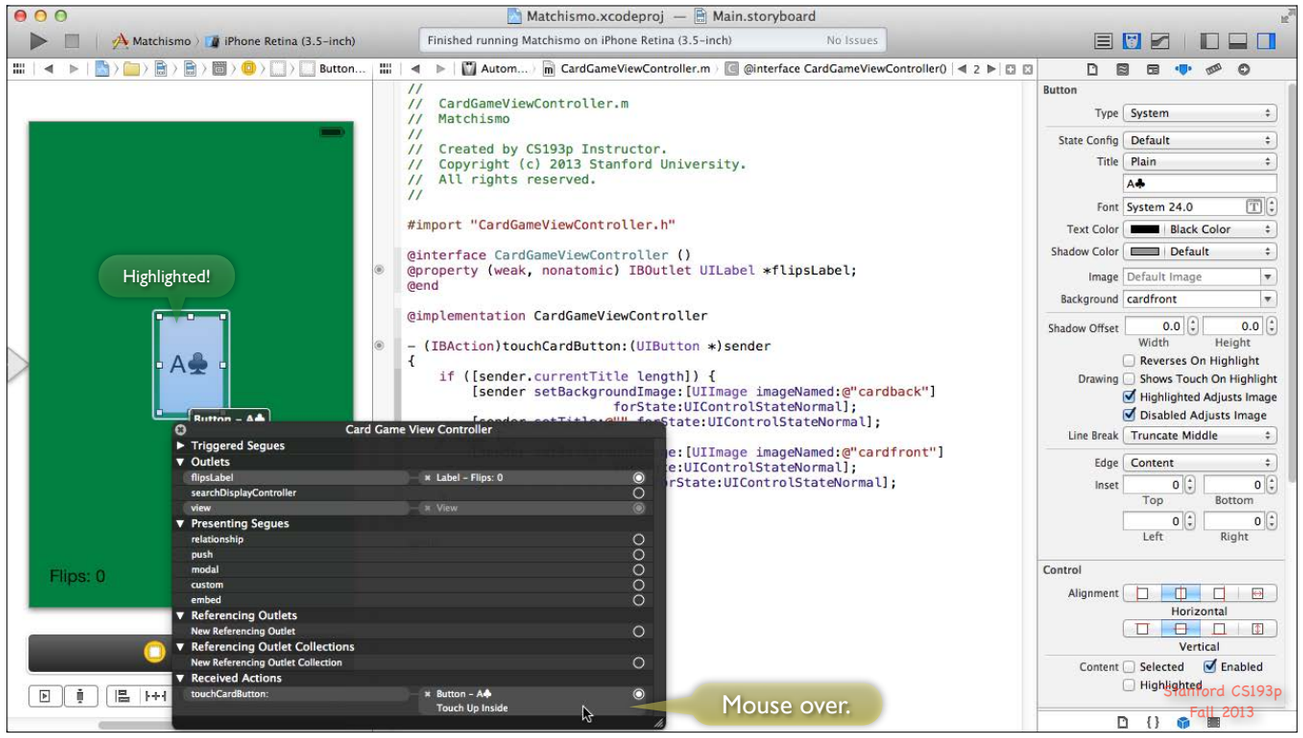

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
It is also possible to see the connections between your View and Controller by right-clicking on an element in your View.
For example, right-click on the Flips:0 label.
Label – Flips: 0
Outlet Collections
gestureRecognizers
Referencing Outlets
flipsLabel
Card Game View Controller
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Here are all the connections to/from this UILabel. Notice the outlet flipsLabel.
Label
Text Plain
Flips: 0
Color Black Color
Font System 17.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Breaks Truncate Tail
Autoshrink Fixed Font Size
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Horizontal Vertical
View
Mode Left
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Tint Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Stanford CS193p
Fall 2013

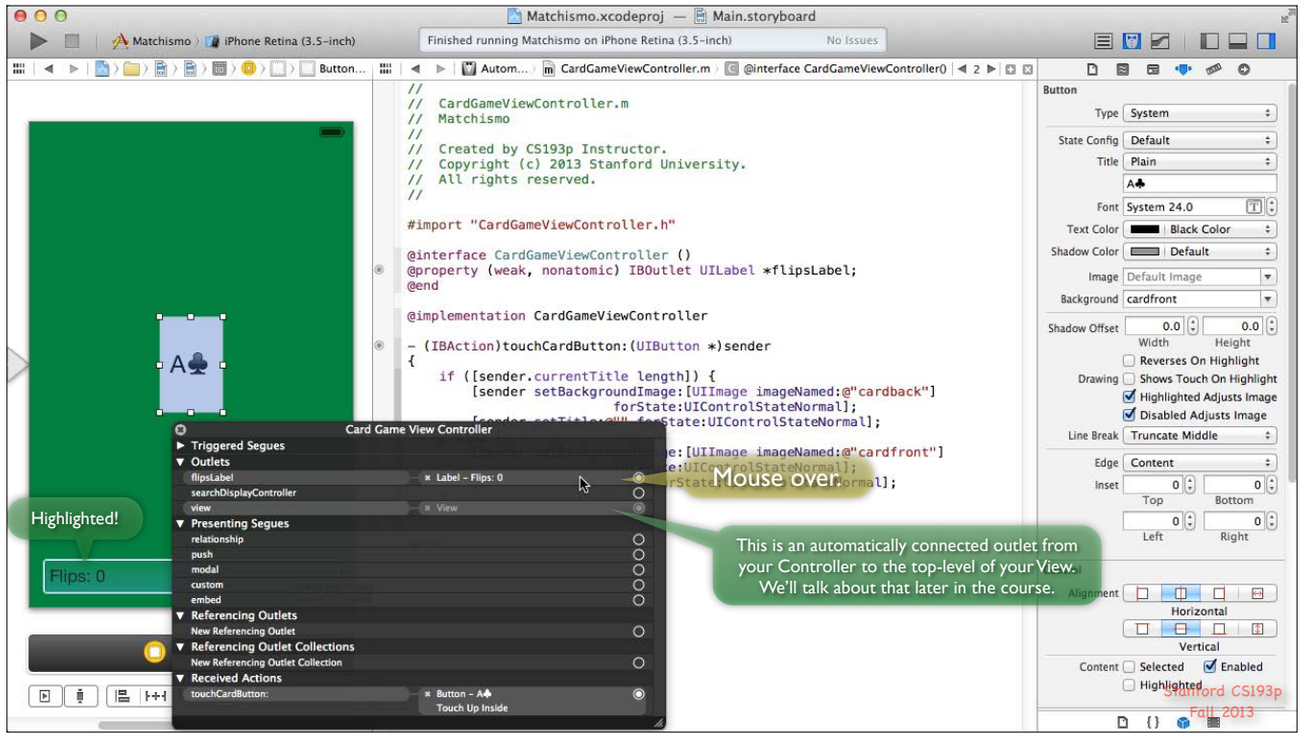

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel
@end
@implementation CardGameViewController
Flips: 0
Button – A♣
Triggered Segues
Outlet Collections
gestureRecognizers
Sent Events
Did End On Exit
Editing Changed
Editing Did Begin
Editing Did End
Touch Cancel
Touch Down
Touch Down Repeat
Touch Drag Enter
Touch Drag Exit
Touch Drag Inside
Touch Drag Outside
Touch Up Inside
Card Game View Controller
touchCardButton:
Touch Up Outside
Value Changed
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Button
Type System
Font System 24.0
Text Color Black Color
Background cardfront
Width
Height
Reverses On Highlight
Edge Content
Inset
Top
Bottom
Left
Right
Control
Alignment
Horizontal
Vertical
Selected
Enabled
Highlighted
Try right-clicking on the card button.
You can "disconnect" connectoins by clicking on these little x's. But don't do it now! If you do so accidentally, just drag again from the View element to the appropriate code (method or @property).
It is a source of annoying bugs to forget to disconnect a no-longer-being-used connection. At runtime, you'll get a crash with a complaint like "no such method, touchCardButton:".
Here's the touchCardButton: action connection.
You can actually ctrl-drag from these little circles to make connections too but it's pretty rare to do it that way.
Stanford CS193p Fall 2013

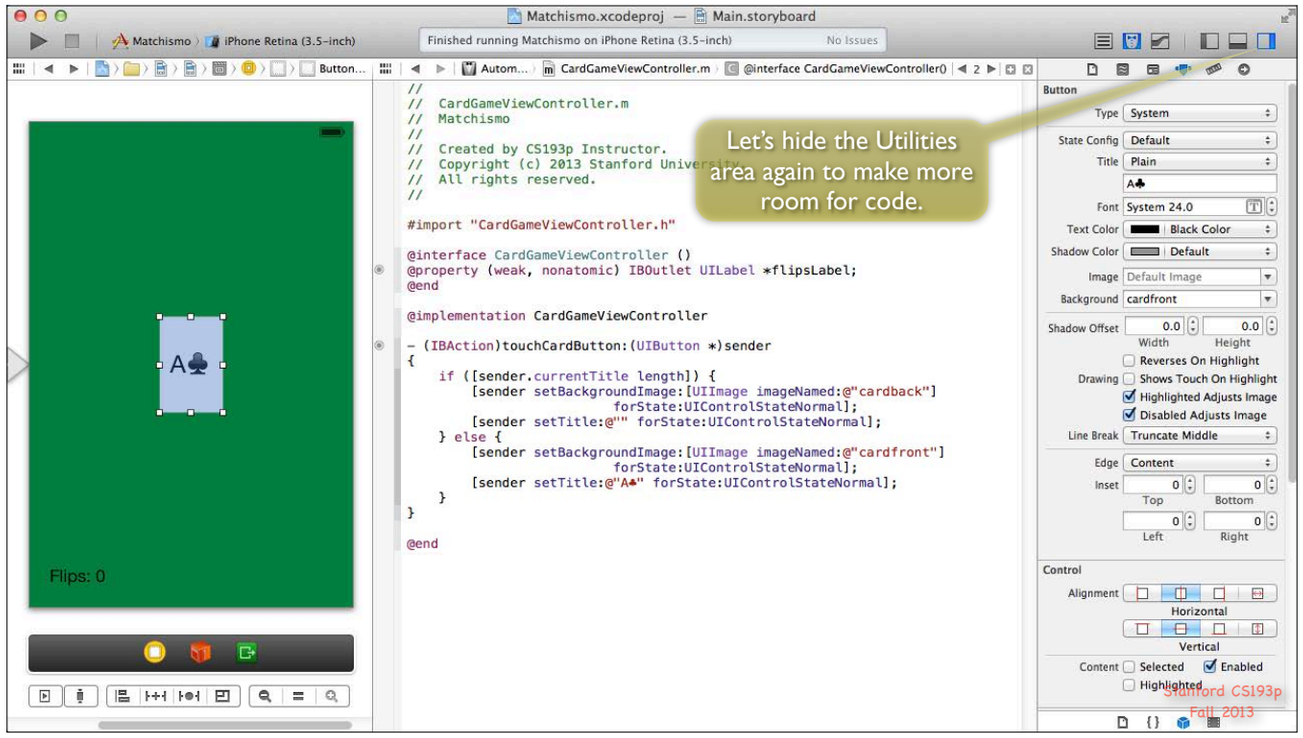

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewCont
@interface CardGameViewContro
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
*)sender
{
Image imageNamed:@"cardback"]
ControlStateNormal];
UIControlStateNormal];
Image imageNamed:@"cardfront"]
ControlStateNormal];
e:UIControlStateNormal];
Flips: 0
Button – A♣
Triggered Segues
Outlet Collections
gestureRecognizers
Sent Events
Did End On Exit
Editing Changed
Editing Did Begin
Editing Did End
Touch Cancel
Touch Down
Touch Down Repeat
Touch Drag Enter
Touch Drag Exit
Touch Drag Inside
Touch Drag Outside
Touch Up Inside
Card Game View Controller
touchCardButton:
Touch Up Outside
Value Changed
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
If you mouse over something in this dialog, it will show what it is connected to.
Mouse over this connection ...
... and you'll see that the entire View highlights.
This mouse over mechanism shows connections to the Controller by highlighting the entire View.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

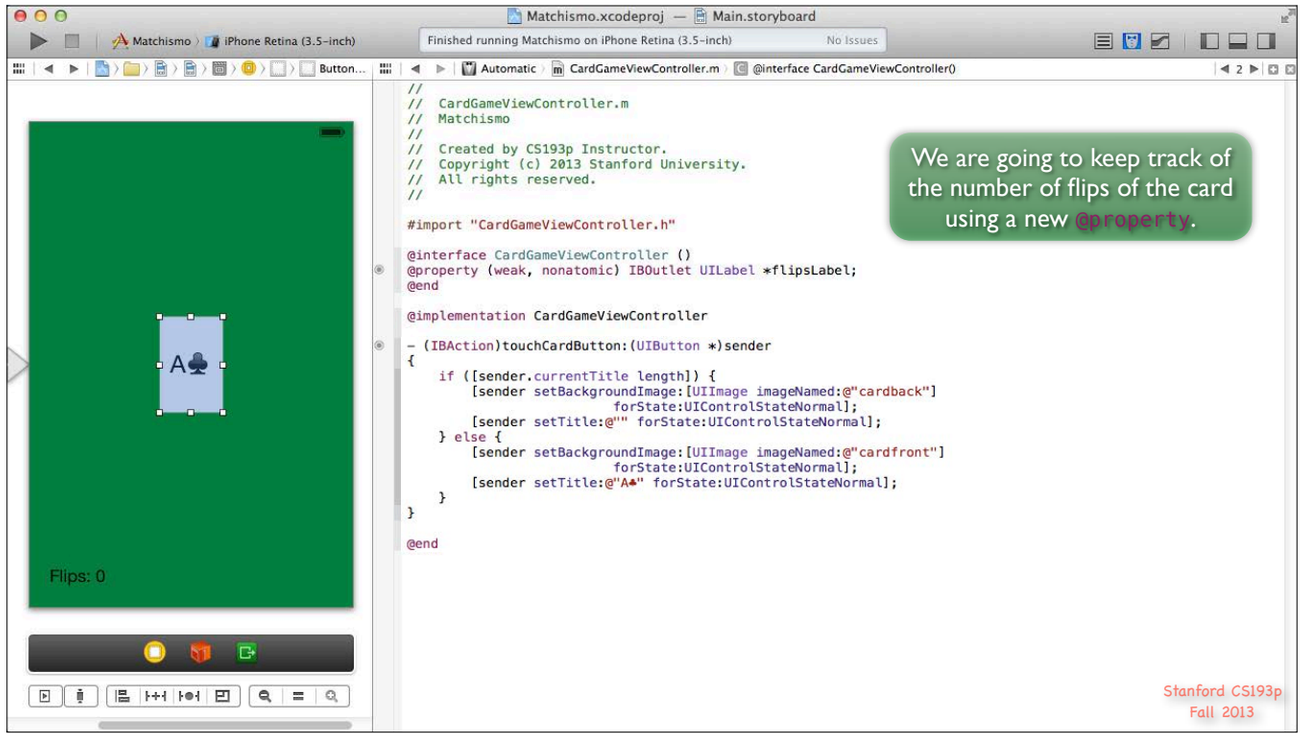

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
Flips: 0
Card Game View Controller
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
You can also "right-click" on your Controller itself (and thus see all connections) by using this icon.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

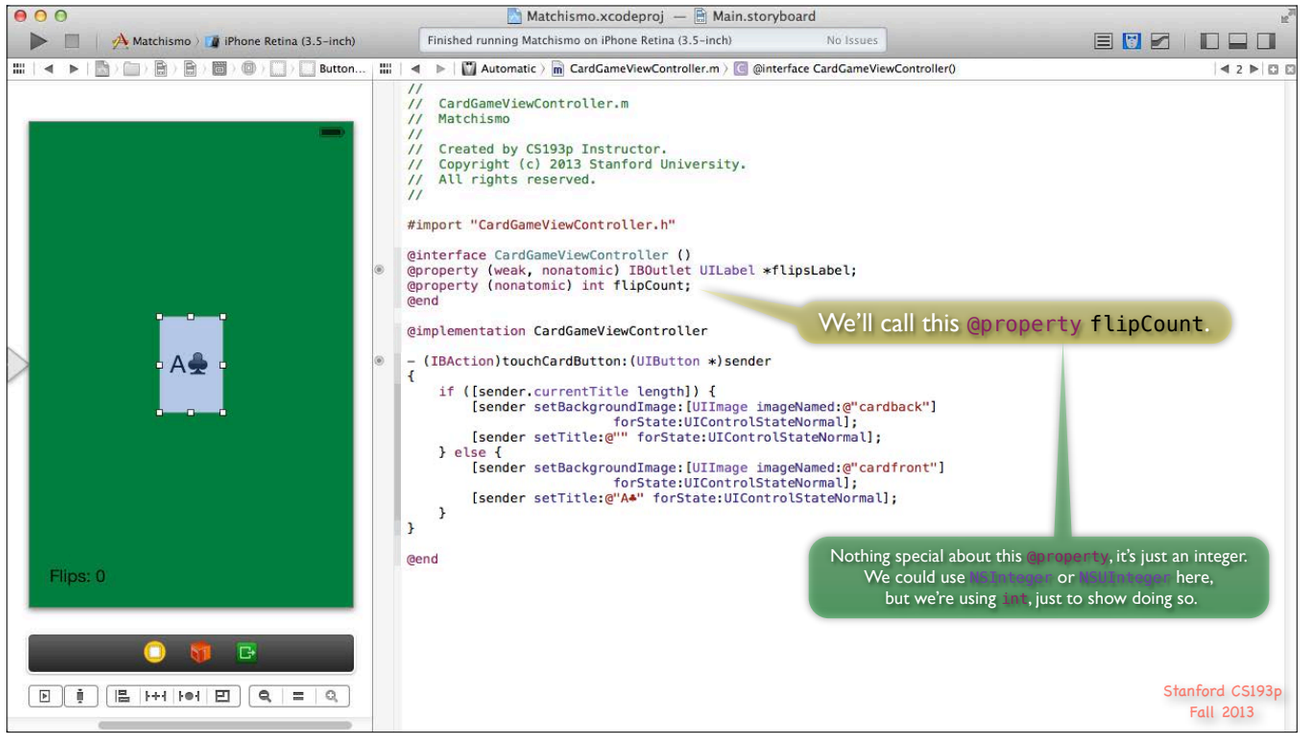

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
This title bar says that we are looking at the connections to/from our Controller.
Right-click on this icon.
Flips: 0
Card Game View Controller
Triggered Segues
Outlets
flipsLabel
Label – Flips: 0
searchDisplayController
view
View
Presenting Segues
relationship
push
modal
custom
embed
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Received Actions
touchCardButton:
Button – A♣
Touch Up Inside
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

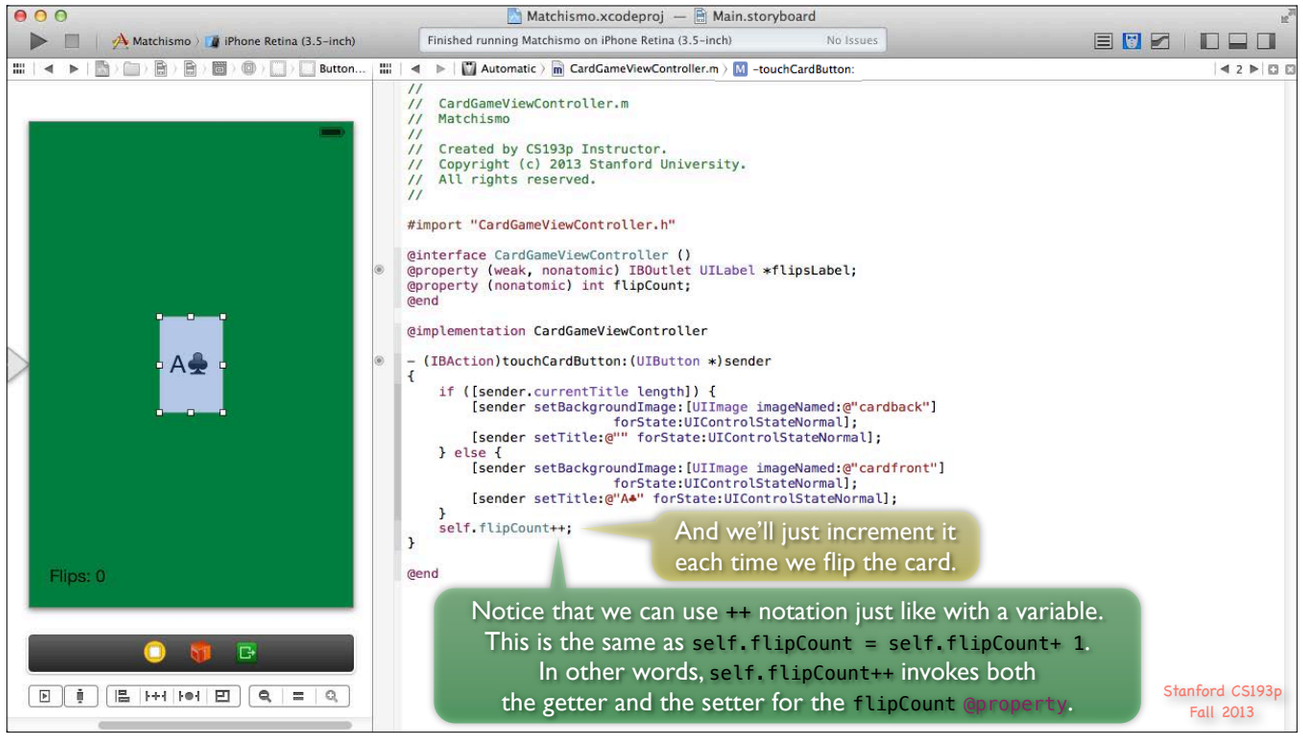

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Highlighted!
Flips: 0
Card Game View Controller
Triggered Segues
Outlets
flipsLabel
Label – Flips: 0
searchDisplayController
view
View
Presenting Segues
relationship
push
modal
custom
embed
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Received Actions
touchCardButton:
Button – A♣
Touch Up Inside
Mouse over.
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

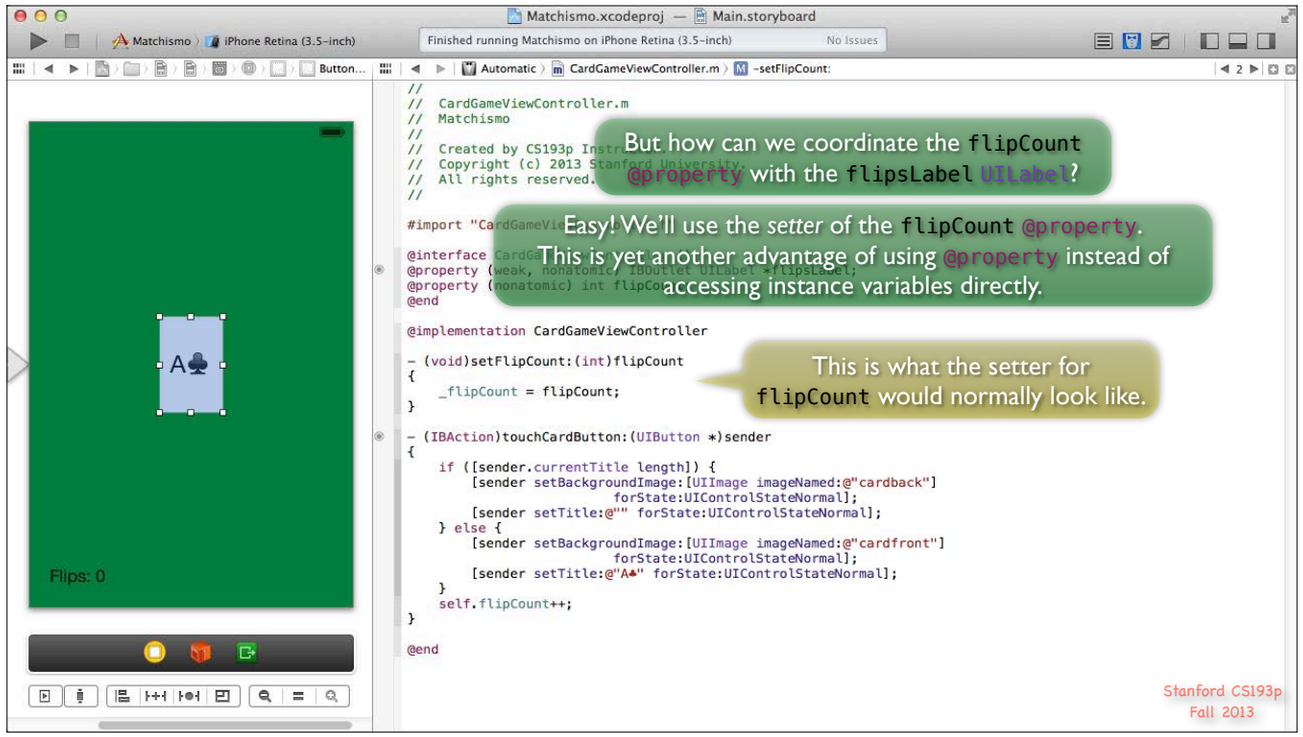

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
CardGameViewController.m
@interface CardGameViewController()
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
A♣
Highlighted!
Flips: 0
Card Game View Controller
Triggered Segues
Outlets
flipsLabel
Label – Flips: 0
searchDisplayController
view
View
Presenting Segues
relationship
push
modal
custom
embed
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Received Actions
touchCardButton:
Button – A♣
Touch Up Inside
Mouse over.
This is an automatically connected outlet from your Controller to the top-level of your View. We'll talk about that later in the course.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Drawing Reverses On Highlight
Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

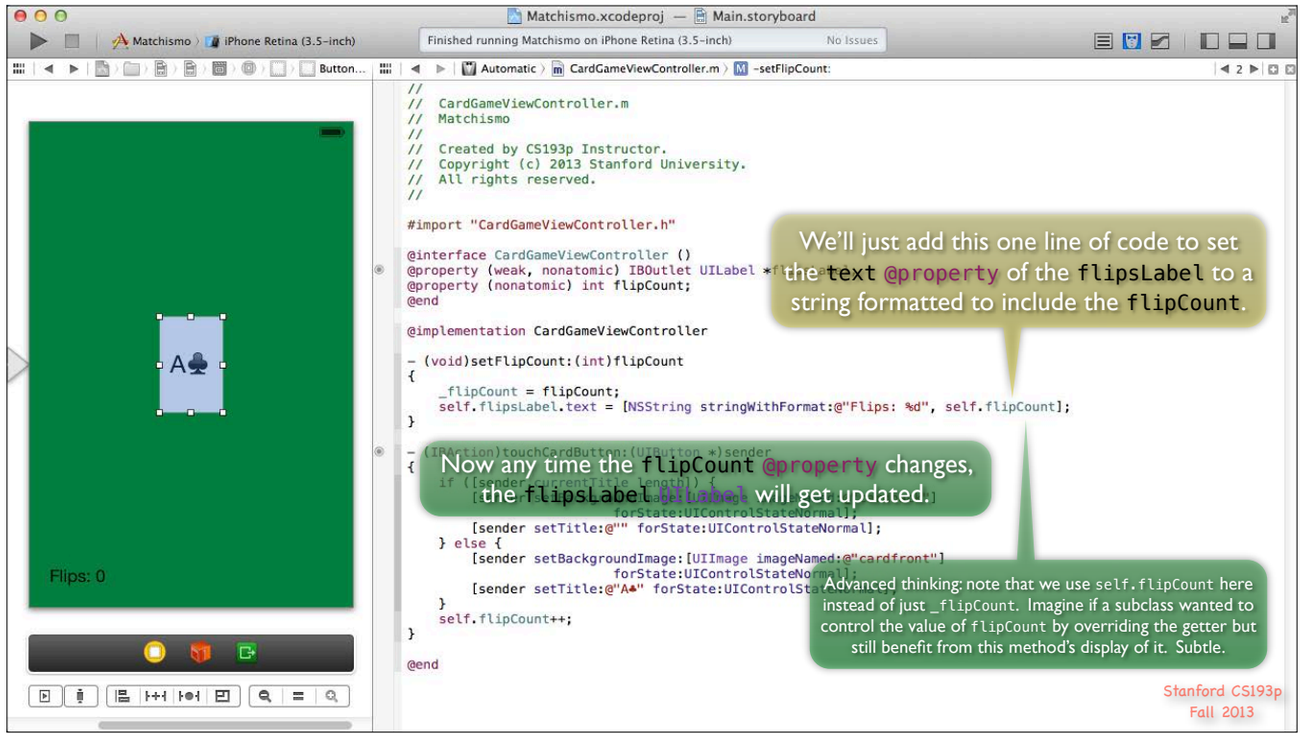

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Button...
Autom... › CardGameViewController.m › @interface CardGameViewController() › 2
A♣
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
Let's hide the Utilities area again to make more room for code.
Button
Type System
State Config Default
Title Plain
A♣
Font System 24.0
Text Color Black Color
Shadow Color Default
Image Default Image
Background cardfront
Shadow Offset 0.0 0.0
Width Height
Reverses On Highlight
Drawing Shows Touch On Highlight
Highlighted Adjusts Image
Disabled Adjusts Image
Line Break Truncate Middle
Edge Content
Inset 0 0
Top Bottom
0 0
Left Right
Control
Alignment
Horizontal
Vertical
Content Selected Enabled
Highlighted
Stanford CS193p
Fall 2013

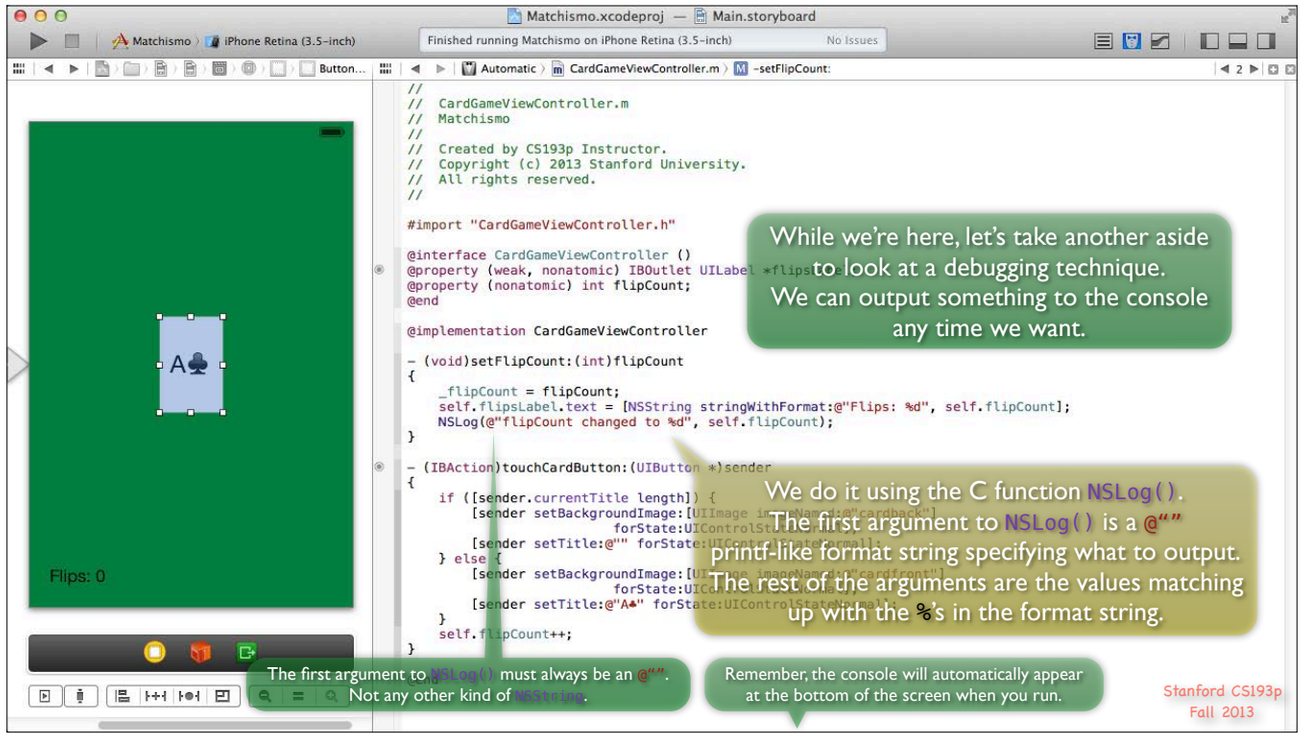

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › @interface CardGameViewController()
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
We are going to keep track of the number of flips of the card using a new @property.
Stanford CS193p
Fall 2013

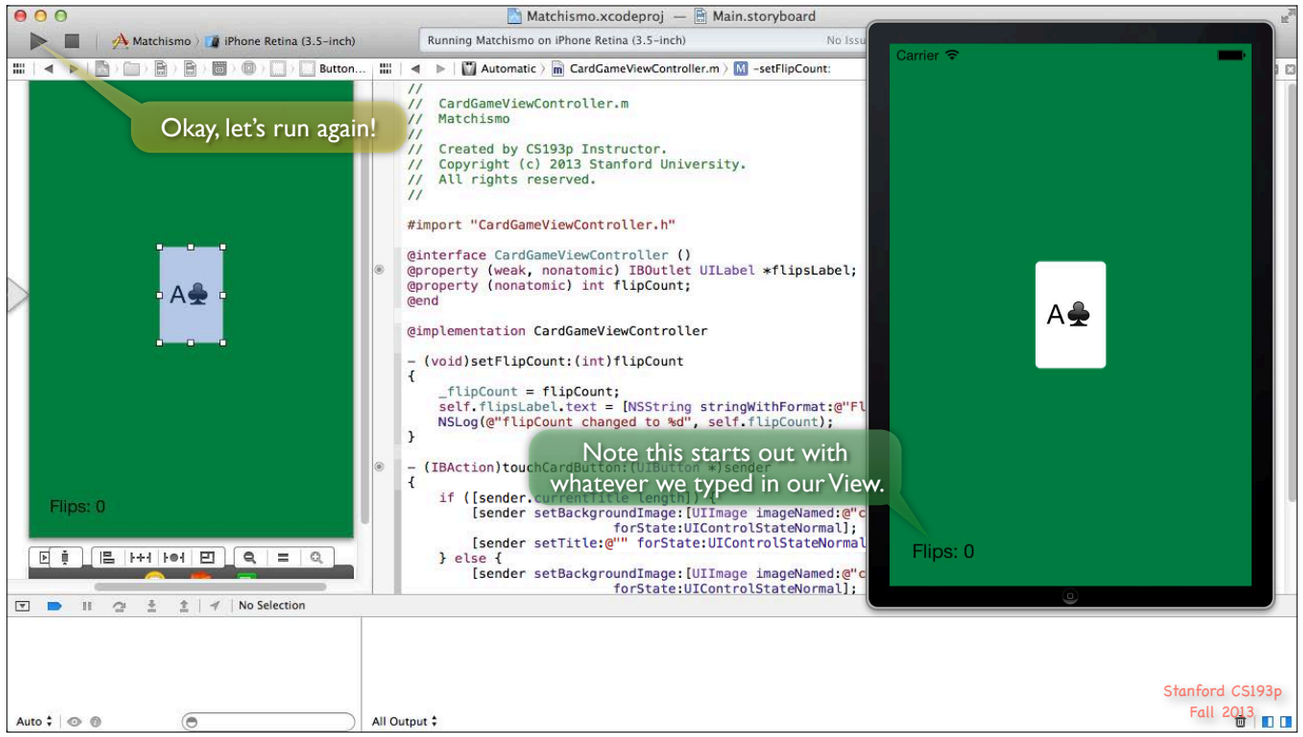

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › @interface CardGameViewController()
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
}
@end
We'll call this @property flipCount.
Nothing special about this @property, it's just an integer.
We could use NSInteger or NSUInteger here,
but we're using int, just to show doing so.
Stanford CS193p
Fall 2013

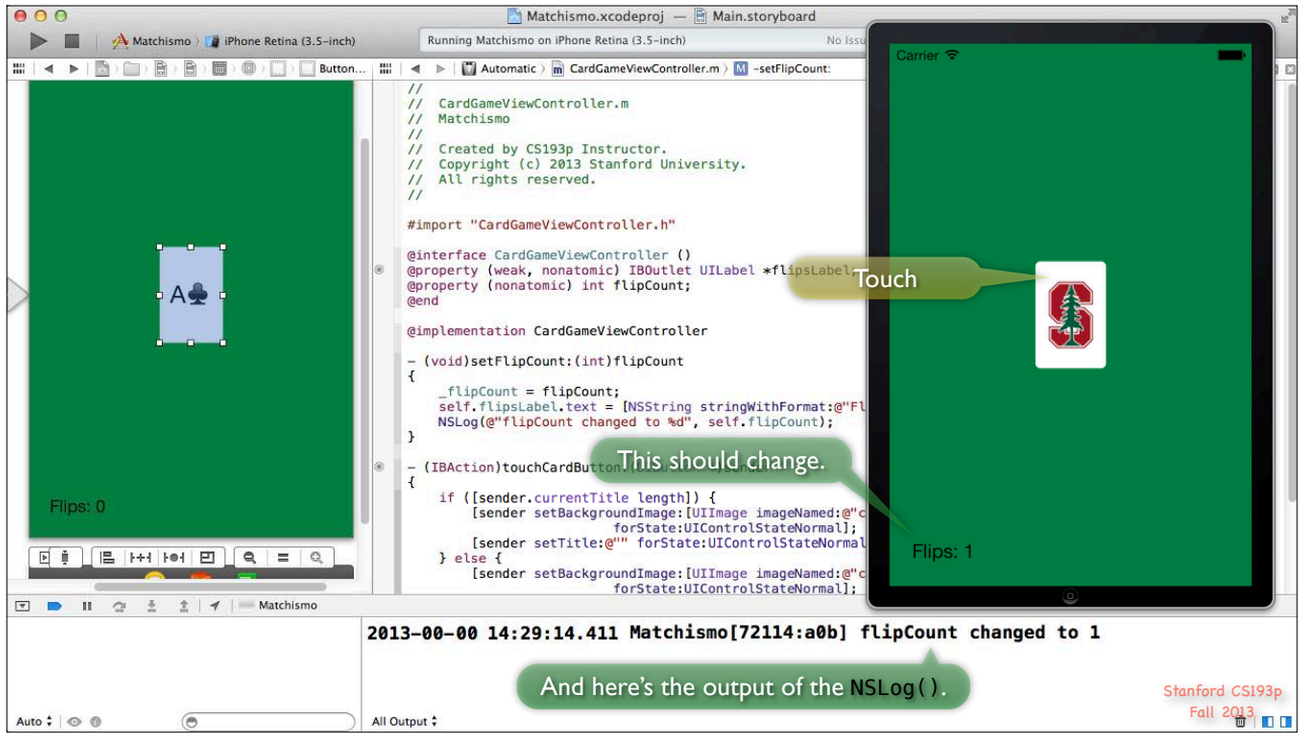

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Button...
Automatic › CardGameViewController.m › -touchCardButton:
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
self.flipCount++;
}
@end
And we'll just increment it each time we flip the card.
Notice that we can use ++ notation just like with a variable. This is the same as self.flipCount = self.flipCount+ 1. In other words, self.flipCount++ invokes both the getter and the setter for the flipCount @property.
Stanford CS193p
Fall 2013

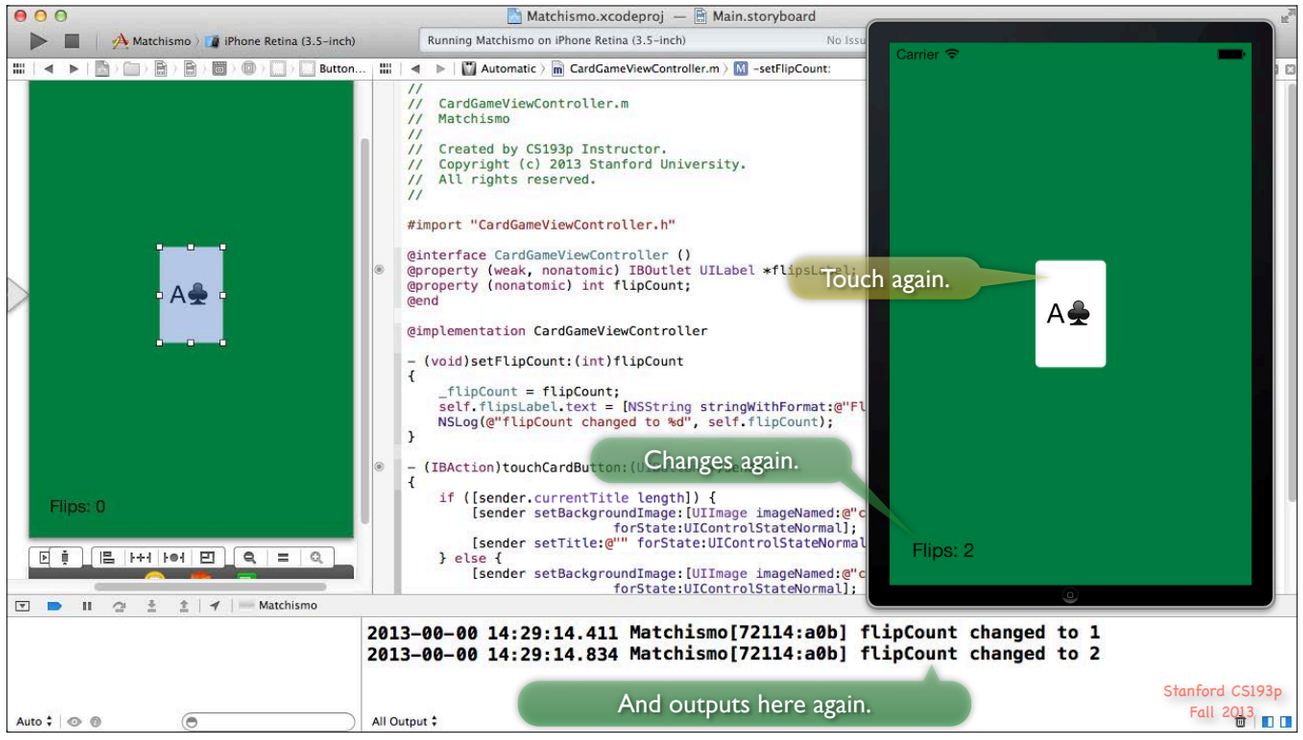

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -setFlipCount:
Flips: 0
But how can we coordinate the flipCount @property with the flipsLabel UILabel?
Easy! We'll use the setter of the flipCount @property. This is yet another advantage of using @property instead of accessing instance variables directly.
This is what the setter for flipCount would normally look like.
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
}
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
self.flipCount++;
}
@end
Stanford CS193p
Fall 2013

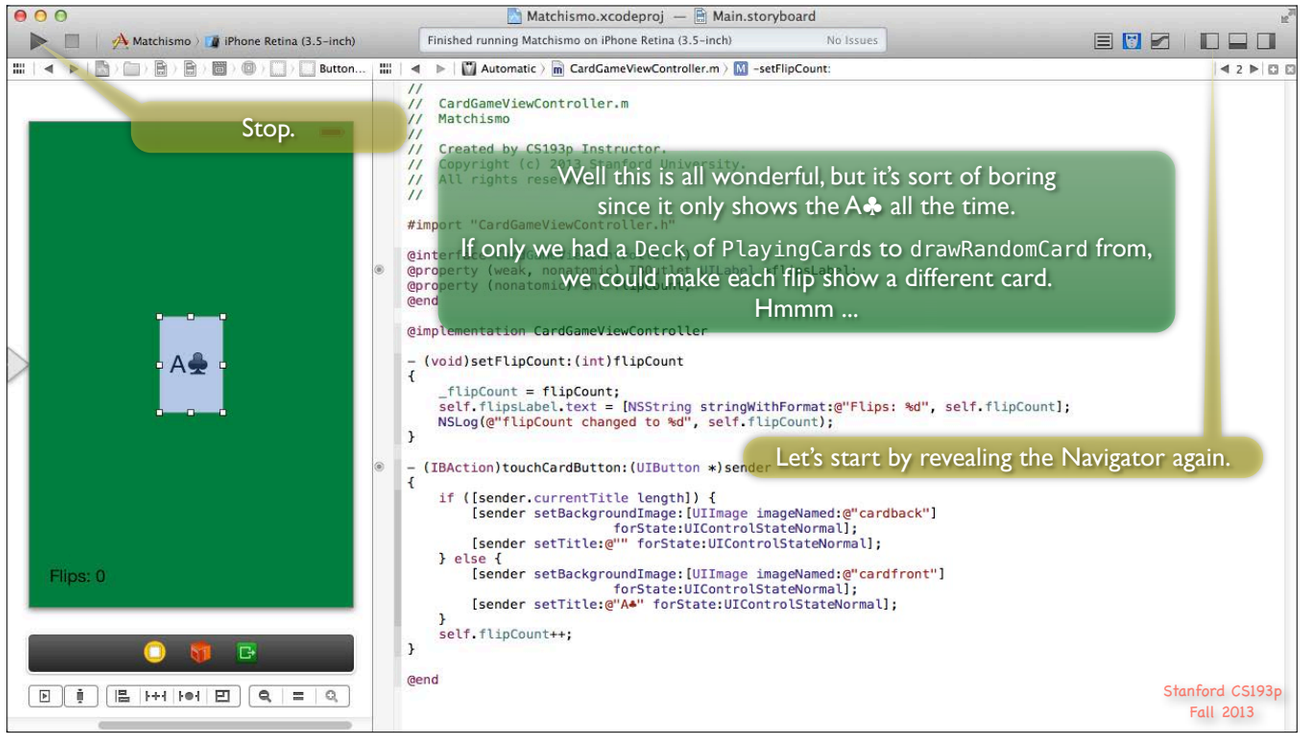

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -setFlipCount:
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Flips: %d", self.flipCount];
}
- (IBAction)touchCardButton:(UIButton *)sender
{
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
}
self.flipCount++;
}
@end
We'll just add this one line of code to set the text @property of the flipsLabel to a string formatted to include the flipCount.
Now any time the flipCount @property changes, the flipsLabel will get updated.
Advanced thinking: note that we use self.flipCount here instead of just _flipCount. Imagine if a subclass wanted to control the value of flipCount by overriding the getter but still benefit from this method's display of it. Subtle.
Stanford CS193p
Fall 2013

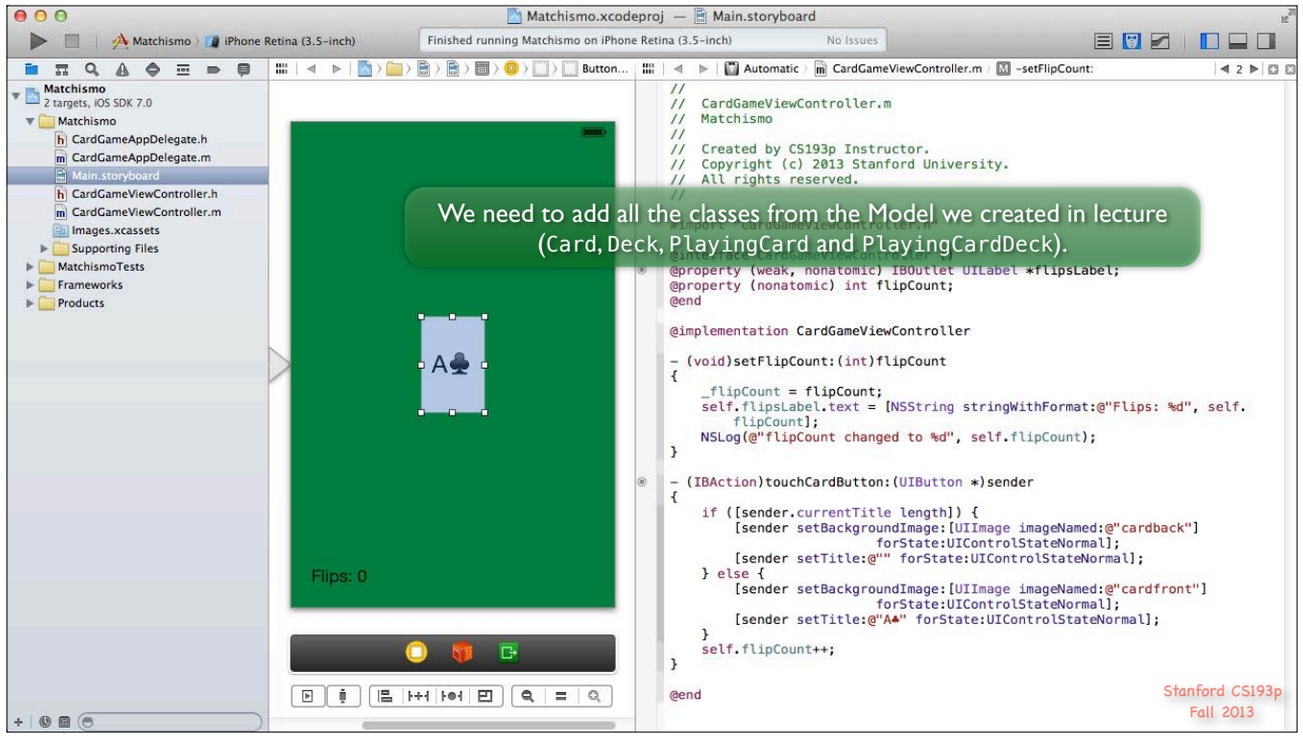

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -setFlipCount:
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Flips: %d", self.flipCount];
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
self.flipCount++;
}
While we're here, let's take another aside to look at a debugging technique. We can output something to the console any time we want.
We do it using the C function NSLog(). The first argument to NSLog() is a @"" printf-like format string specifying what to output. The rest of the arguments are the values matching up with the %'s in the format string.
The first argument to NSLog() must always be an @"". Not any other kind of NSString.
Remember, the console will automatically appear at the bottom of the screen when you run.
Stanford CS193p
Fall 2013

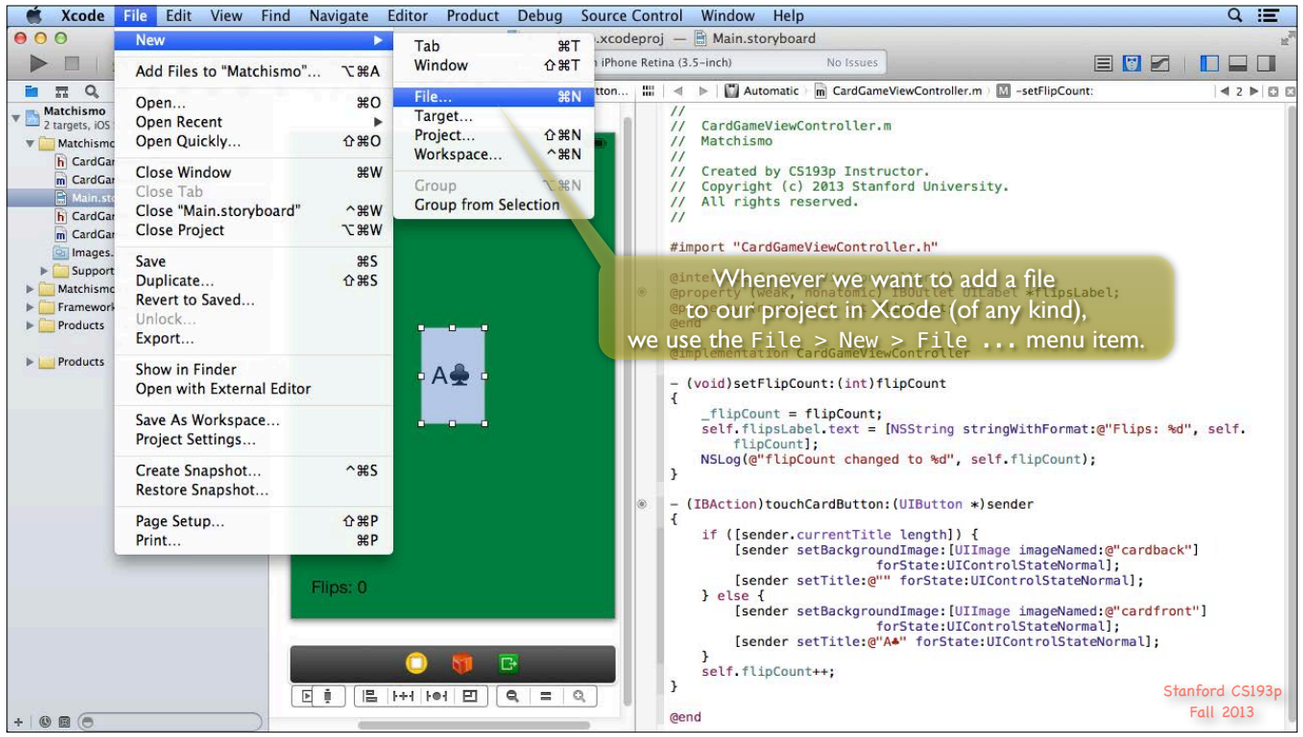

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
Okay, let's run again!
Flips: 0
Automatic › CardGameViewController.m › -setFlipCount:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Fl
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"c
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"c
forState:UIControlStateNormal];
Carrier
Note this starts out with whatever we typed in our View.
Flips: 0
No Selection
Auto
All Output
Stanford CS193p
Fall 2013

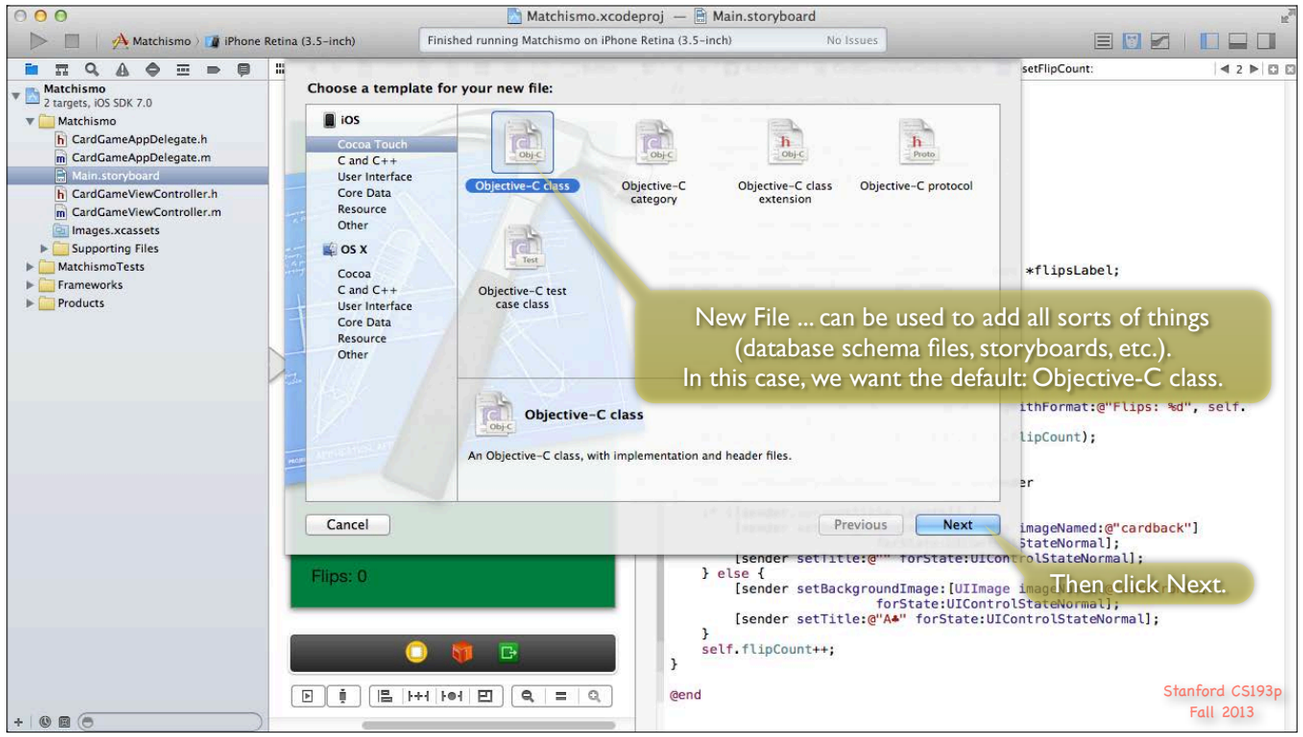

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -setFlipCount:
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Fl
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"c
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"c
forState:UIControlStateNormal];
Flips: 0
Carrier
Touch
This should change.
Flips: 1
Matchismo
2013-00-00 14:29:14.411 Matchismo[72114:a0b] flipCount changed to 1
And here's the output of the NSLog().
Auto
All Output
Stanford CS193p
Fall 2013

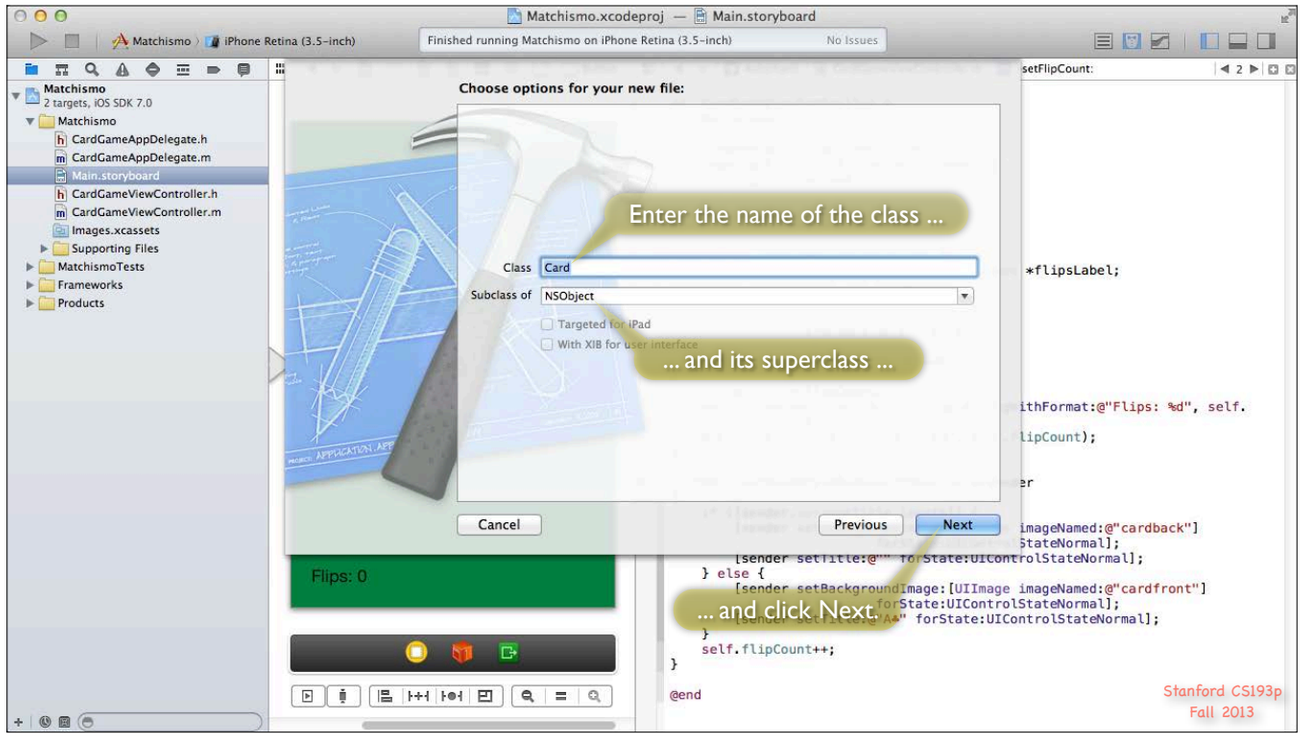

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Running Matchismo on iPhone Retina (3.5-inch)
Automatic › CardGameViewController.m › -setFlipCount:
Flips: 0
Carrier
Touch again.
Changes again.
Flips: 2
2013-00-00 14:29:14.411 Matchismo[72114:a0b] flipCount changed to 1
2013-00-00 14:29:14.834 Matchismo[72114:a0b] flipCount changed to 2
And outputs here again.
Stanford CS193p
Fall 2013

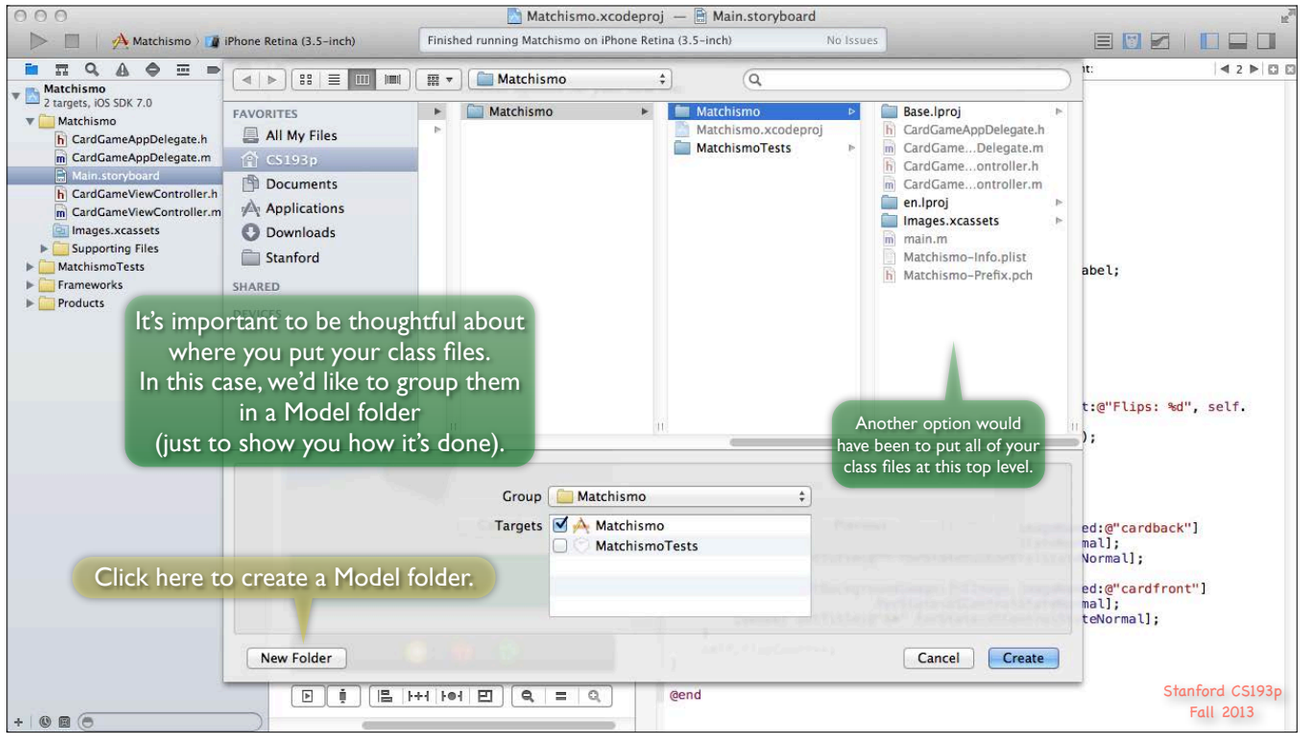

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Automatic › CardGameViewController.m › -setFlipCount:
Stop.
Well this is all wonderful, but it's sort of boring since it only shows the A♣ all the time.
If only we had a Deck of PlayingCards to drawRandomCard from, we could make each flip show a different card.
Hmmm ...
Let's start by revealing the Navigator again.
A♣
Flips: 0
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Flips: %d", self.flipCount];
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"]
forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"]
forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
self.flipCount++;
}
@end
Stanford CS193p
Fall 2013

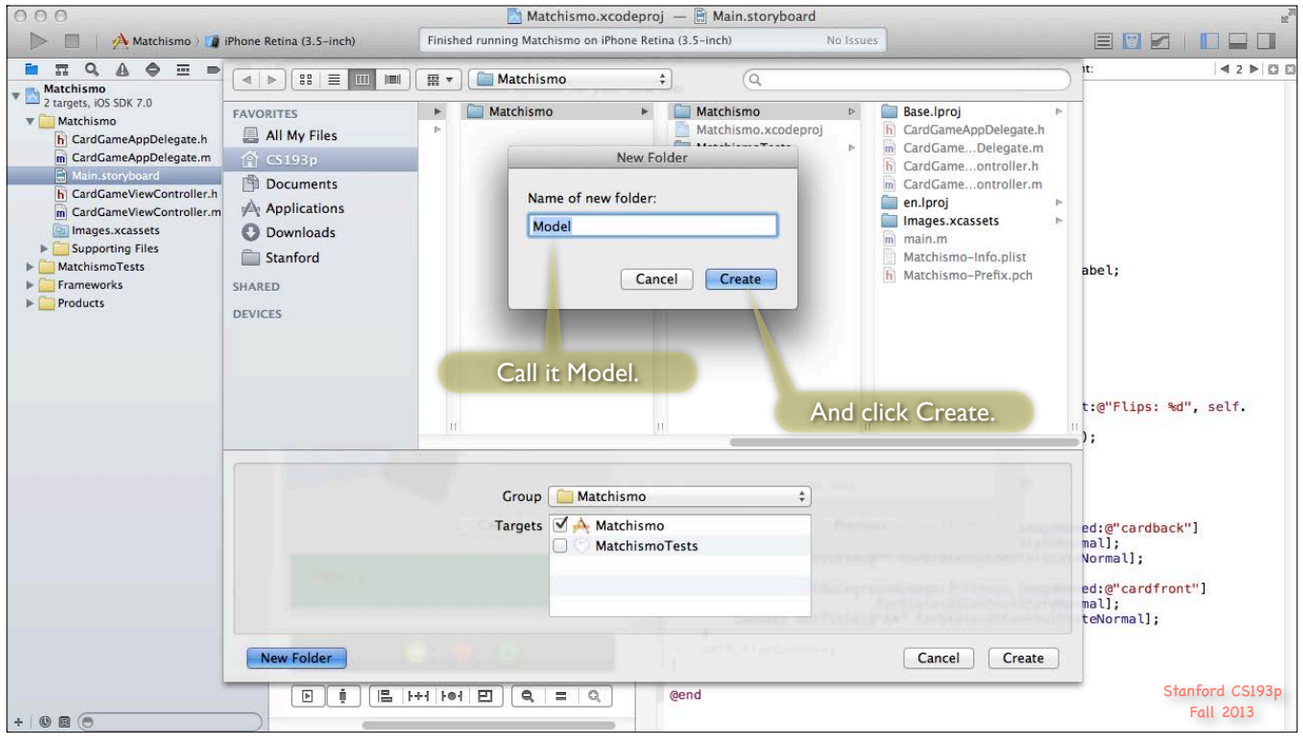

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Flips: 0
We need to add all the classes from the Model we created in lecture (Card, Deck, PlayingCard and PlayingCardDeck).
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Flips: %d", self.flipCount];
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton:(UIButton *)sender
{
if ([sender.currentTitle length]) {
[sender setBackgroundImage:[UIImage imageNamed:@"cardback"] forState:UIControlStateNormal];
[sender setTitle:@"" forState:UIControlStateNormal];
} else {
[sender setBackgroundImage:[UIImage imageNamed:@"cardfront"] forState:UIControlStateNormal];
[sender setTitle:@"A♣" forState:UIControlStateNormal];
}
self.flipCount++;
}
@end
Stanford CS193p
Fall 2013

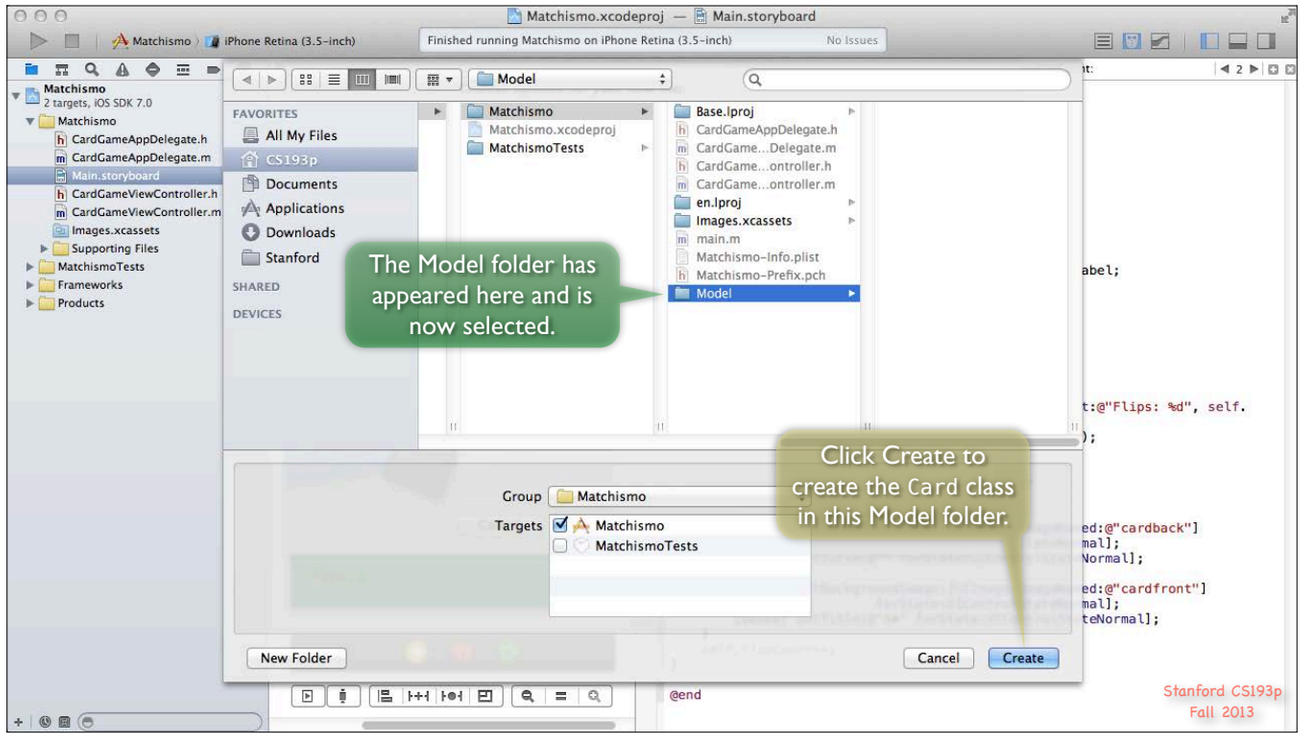

Whenever we want to add a file
to our project in Xcode (of any kind),
we use the File > New > File ... menu item.
Stanford CS193p
Fall 2013

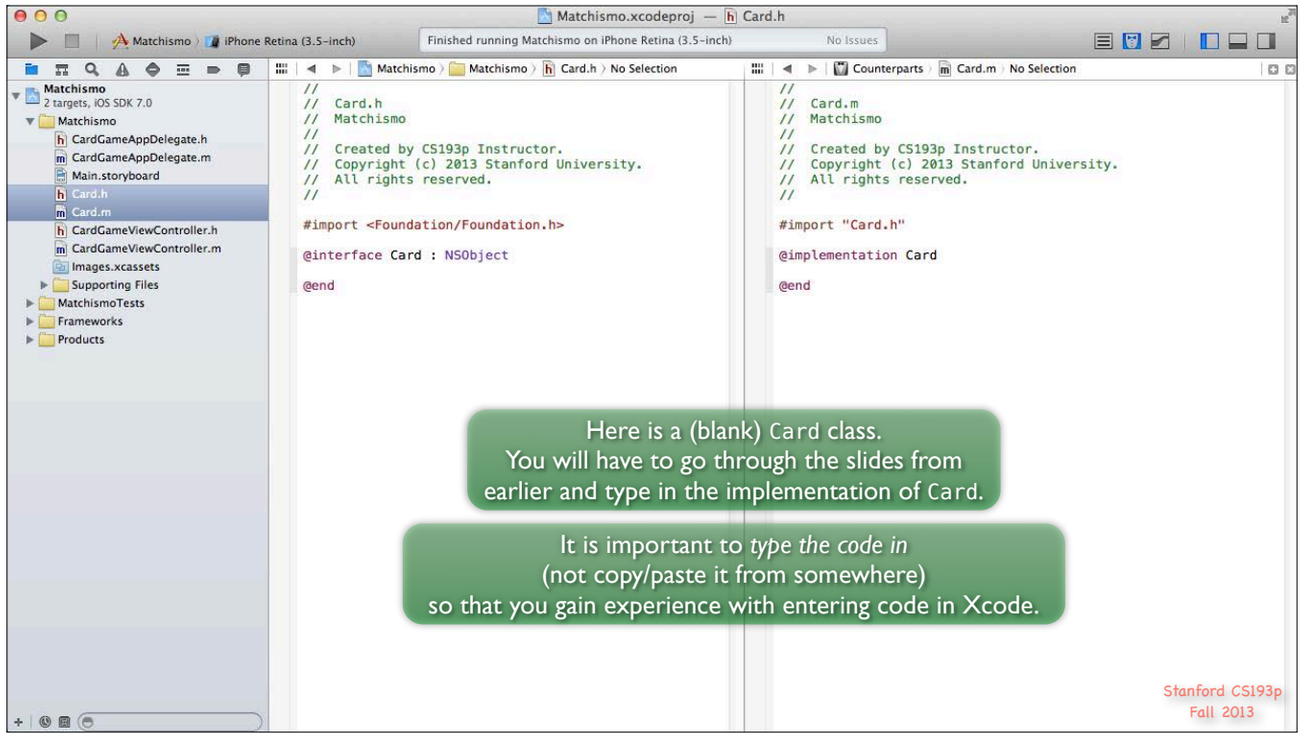

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Choose a template for your new file:
iOS
Cocoa Touch
C and C++
User Interface
Core Data
Resource
Other
OS X
Cocoa
C and C++
User Interface
Core Data
Resource
Other
Objective-C class
Objective-C category
Objective-C class extension
Objective-C protocol
Objective-C test case class
Objective-C class
An Objective-C class, with implementation and header files.
Cancel
Previous
Next
Flips: 0
setFlipCount:
*flipsLabel;
ithFormat:@"Flips: %d", self.
lipCount);
imageNamed:@"cardback"]
StateNormal];
forState:UIControlStateNormal];
forState:UIControlStateNormal];
forState:UIControlStateNormal];
self.flipCount++;
@end
New File ... can be used to add all sorts of things
(database schema files, storyboards, etc.).
In this case, we want the default: Objective-C class.
Then click Next.
Stanford CS193p
Fall 2013

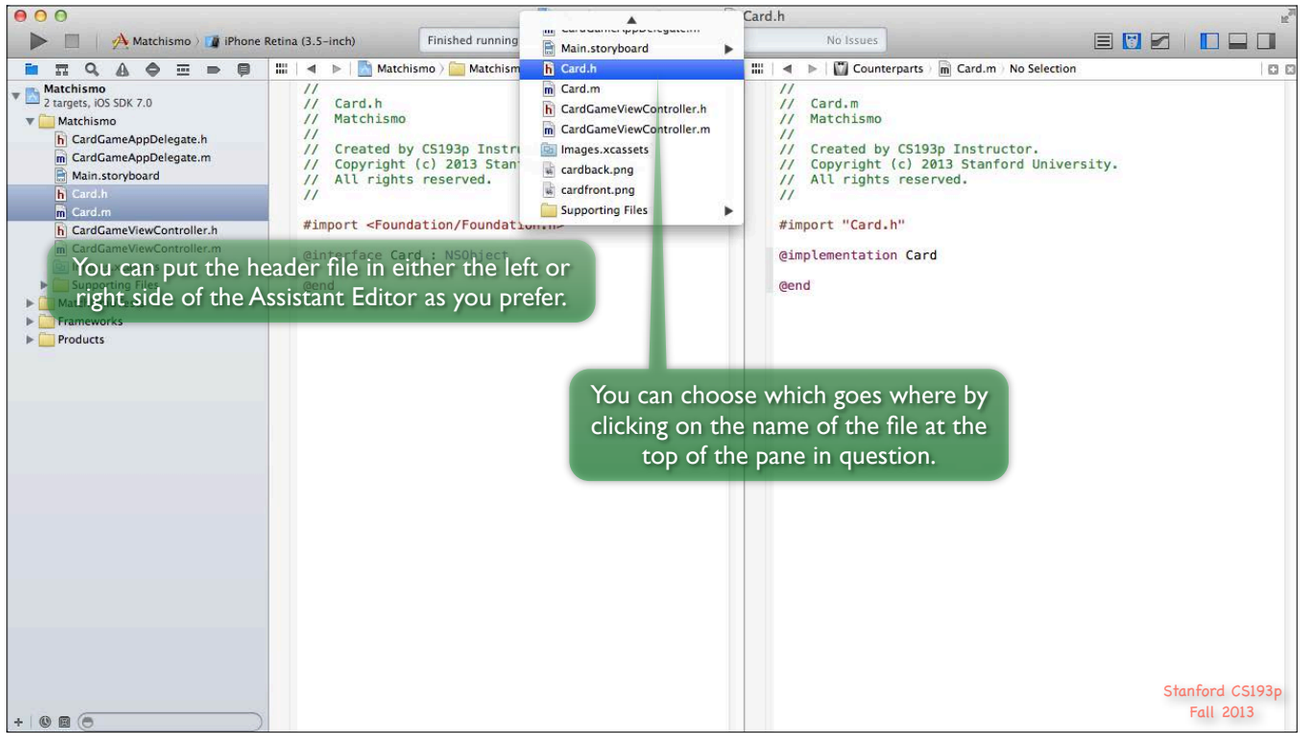

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Choose options for your new file:
Enter the name of the class ...
Class
Card
Subclass of
NSObject
Targeted for iPad
With XIB for user interface
... and its superclass ...
Cancel
Previous
Next
... and click Next.
Flips: 0
setFlipCount:
*flipsLabel;
ithFormat:@"Flips: %d", self.
lipCount);
imageNamed:@"cardback"]
StateNormal];
} else {
forState:UIControlStateNormal];
forState:UIControlStateNormal];
self.flipCount++;
}
@end
Stanford CS193p
Fall 2013

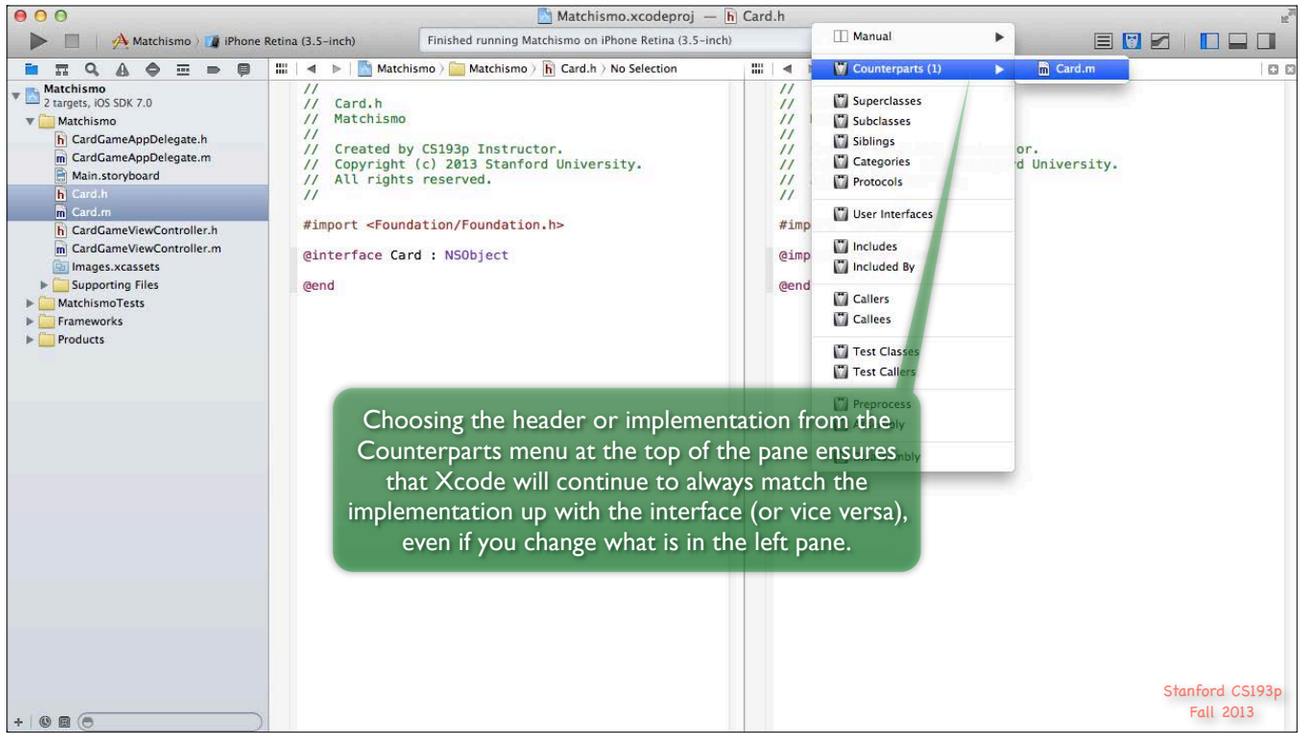

Matchismo.xcodeproj — Main.storyboard
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
FAVORITES
All My Files
CS193p
Documents
Applications
Downloads
Stanford
SHARED
Base.lproj
Matchismo-Info.plist
Matchismo-Prefix.pch
main.m
en.lproj
It's important to be thoughtful about where you put your class files. In this case, we'd like to group them in a Model folder (just to show you how it's done).
Another option would have been to put all of your class files at this top level.
Group
Targets
MatchismoTests
Click here to create a Model folder.
New Folder
Cancel
Create
@end
Stanford CS193p
Fall 2013

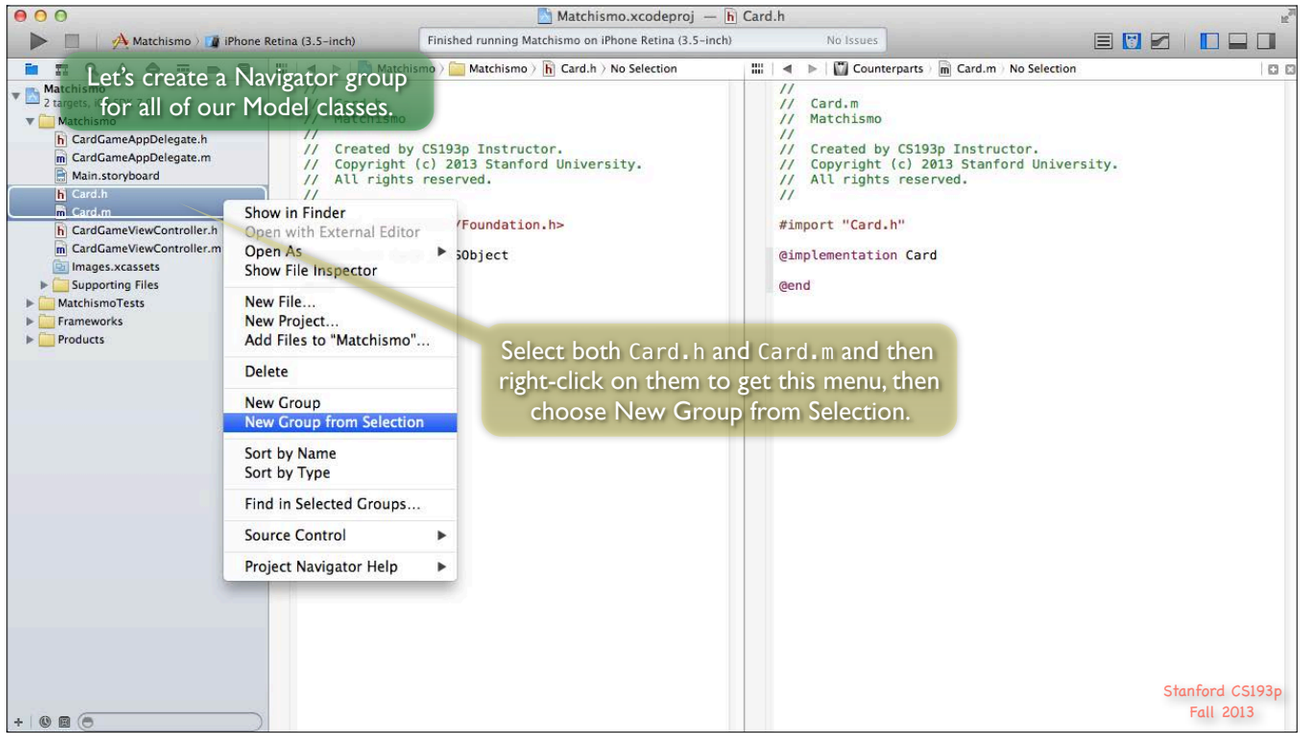

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
FAVORITES
All My Files
CS193p
Documents
Applications
Downloads
Stanford
SHARED
DEVICES
Matchismo.xcodeproj
Base.lproj
CardGameAppDelegate.h
CardGame...Delegate.m
CardGame...ontroller.h
CardGame...ontroller.m
en.lproj
Images.xcassets
main.m
Matchismo-Info.plist
Matchismo-Prefix.pch
New Folder
Name of new folder:
Model
Cancel
Create
Call it Model.
And click Create.
Group
Targets
MatchismoTests
New Folder
Cancel
Create
@end
Stanford CS193p
Fall 2013

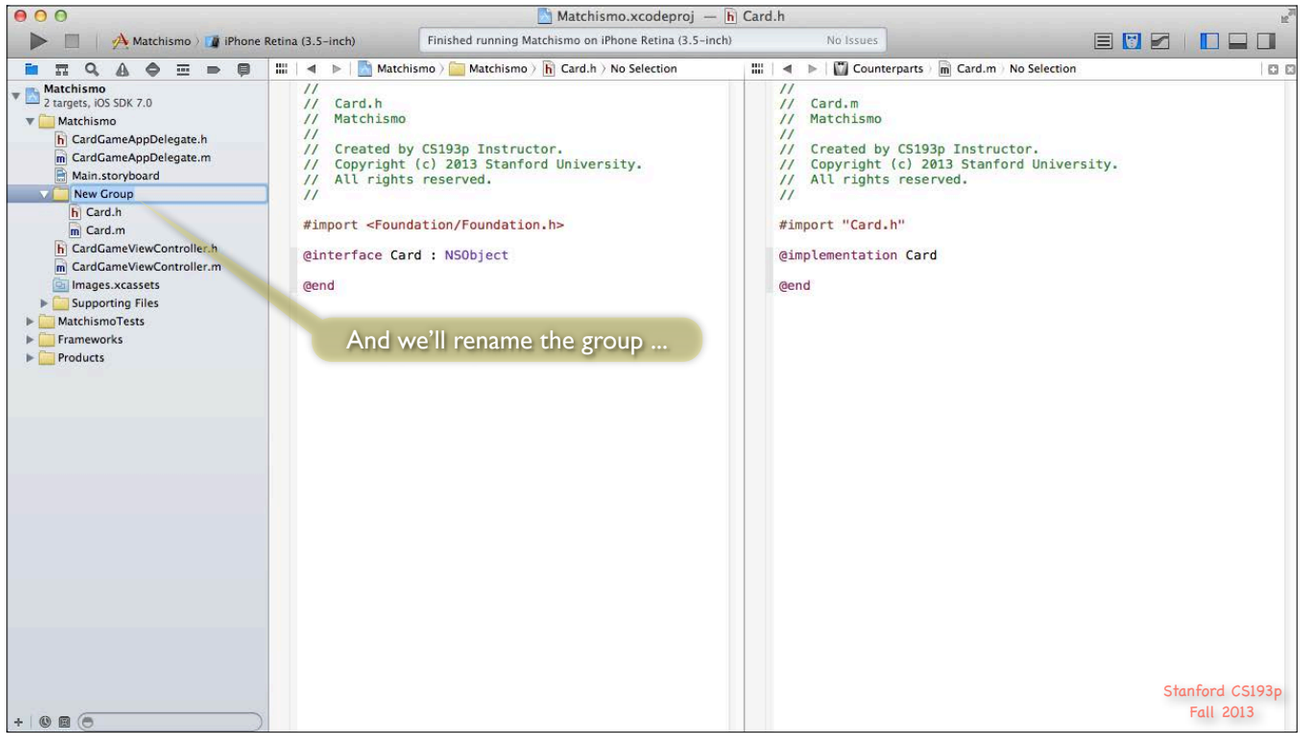

Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Model
FAVORITES
All My Files
CS193p
Documents
Applications
Downloads
Stanford
SHARED
DEVICES
Matchismo
Matchismo.xcodeproj
MatchismoTests
Base.lproj
CardGameAppDelegate.h
CardGame...Delegate.m
CardGame...ontroller.h
CardGame...ontroller.m
en.lproj
Images.xcassets
main.m
Matchismo-Info.plist
Matchismo-Prefix.pch
Model
The Model folder has appeared here and is now selected.
Group
Matchismo
Targets
Matchismo
MatchismoTests
Click Create to create the Card class in this Model folder.
New Folder
Cancel
Create
@end
Stanford CS193p
Fall 2013

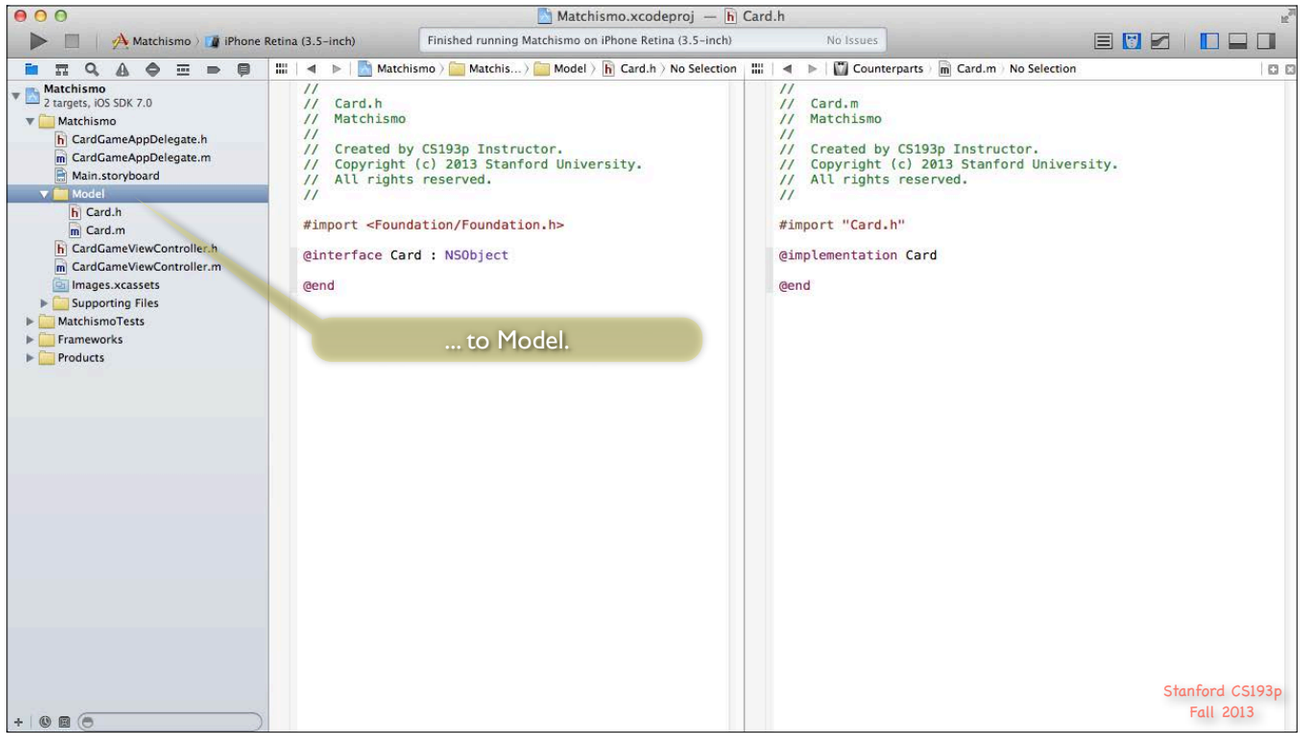

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
Here is a (blank) Card class.
You will have to go through the slides from
earlier and type in the implementation of Card.
It is important to type the code in
(not copy/paste it from somewhere)
so that you gain experience with entering code in Xcode.
Stanford CS193p
Fall 2013

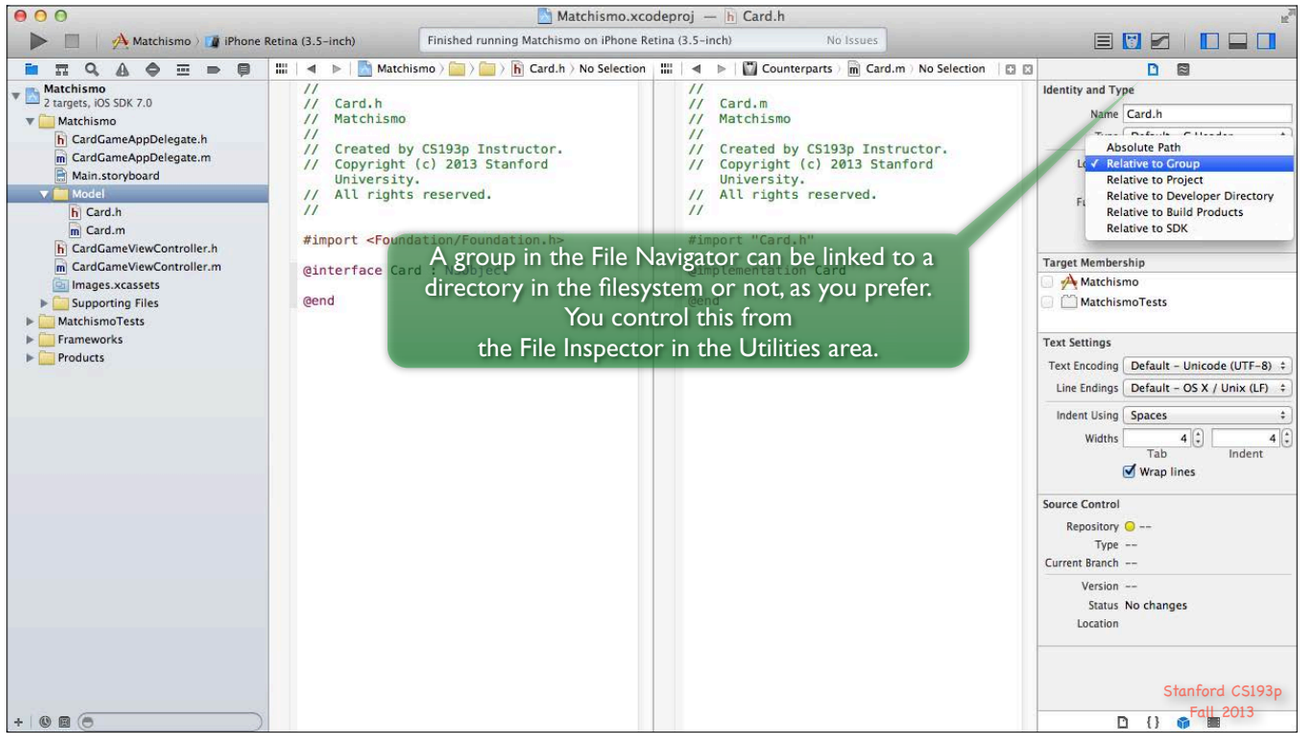

Matchismo
iPhone Retina (3.5-inch)
Finished running
No Issues
Card.h
Matchismo
2 targets, iOS SDK 7.0
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Supporting Files
Frameworks
Products
Images.xcassets
cardback.png
cardfront.png
Counterparts
No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
You can put the header file in either the left or right side of the Assistant Editor as you prefer.
You can choose which goes where by clicking on the name of the file at the top of the pane in question.
Stanford CS193p
Fall 2013

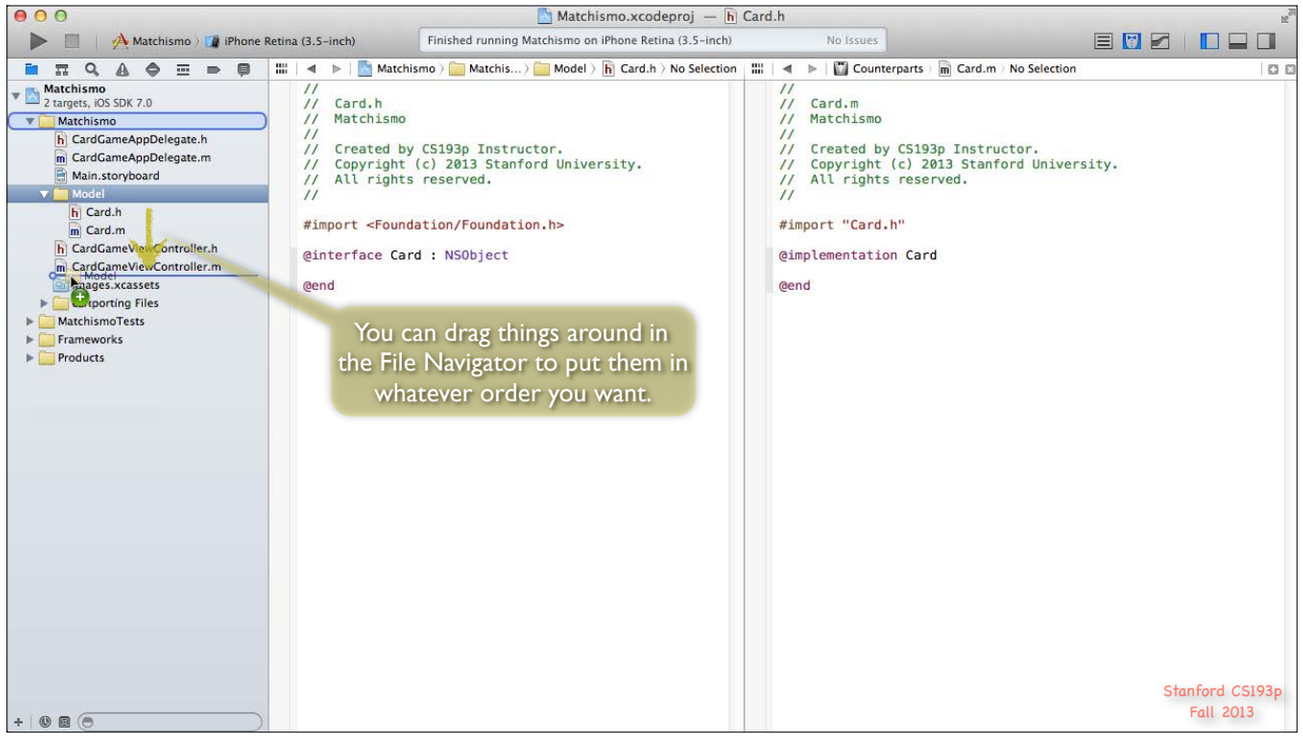

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Manual
Counterparts (1)
Card.m
Superclasses
Subclasses
Siblings
Categories
Protocols
User Interfaces
Includes
Included By
Callers
Callees
Test Classes
Test Callers
Preprocess
Choosing the header or implementation from the Counterparts menu at the top of the pane ensures that Xcode will continue to always match the implementation up with the interface (or vice versa), even if you change what is in the left pane.
Stanford CS193p
Fall 2013

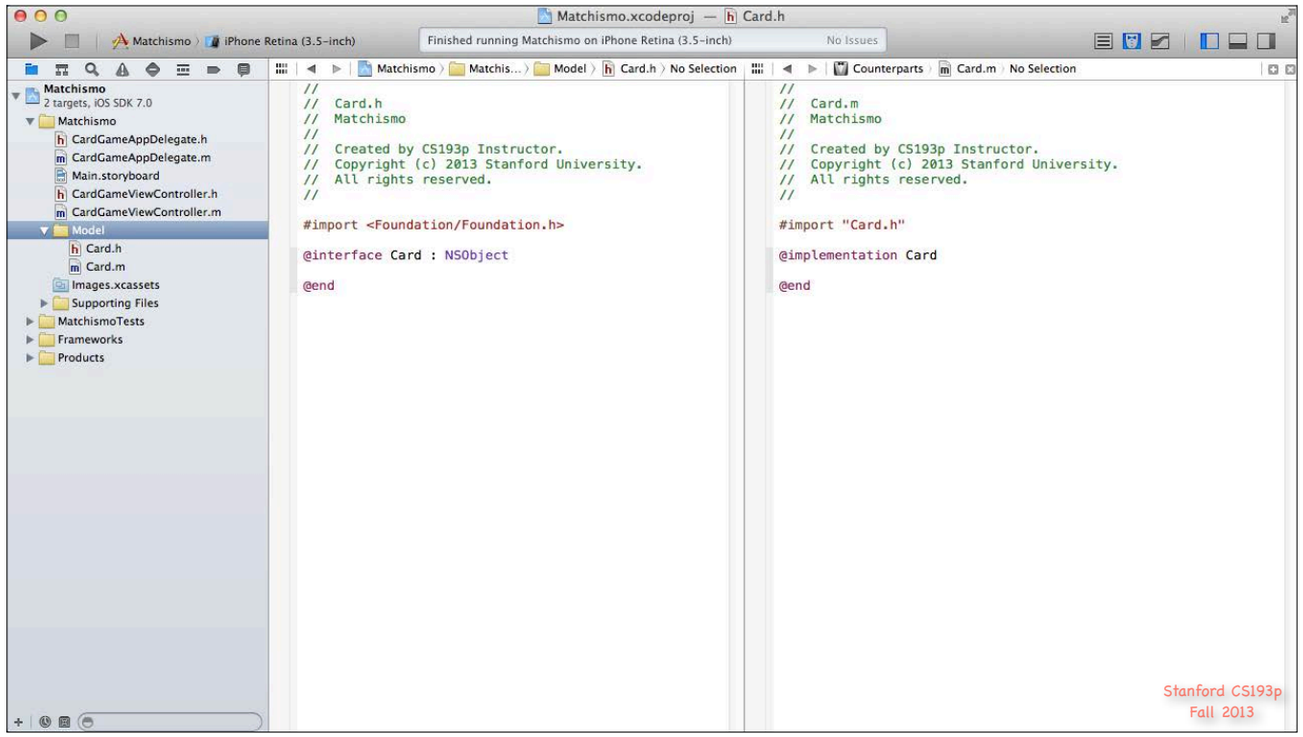

Let's create a Navigator group for all of our Model classes.
Select both Card.h and Card.m and then right-click on them to get this menu, then choose New Group from Selection.
Show in Finder
Open with External Editor
Open As
Show File Inspector
New File...
New Project...
Add Files to "Matchismo"...
Delete
New Group
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help
Stanford CS193p
Fall 2013

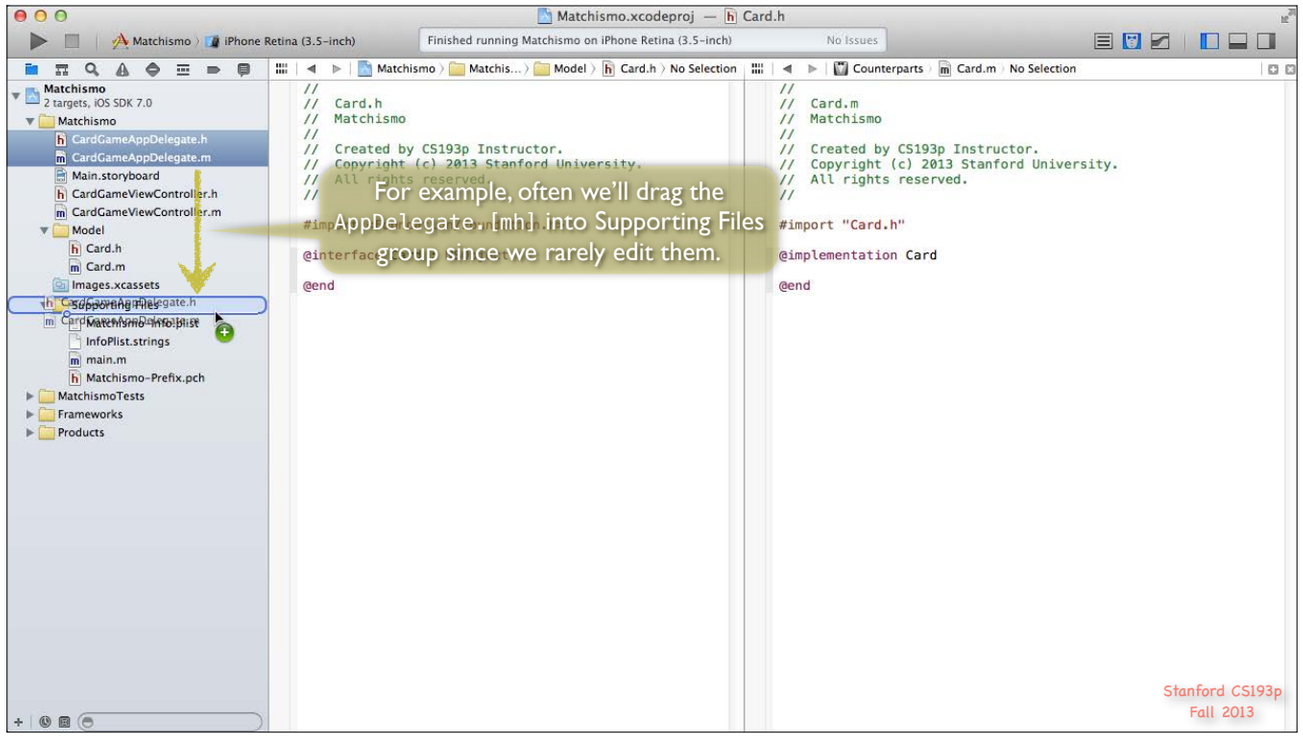

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
New Group
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchismo › Card.h › No Selection
Counterparts › Card.m › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
And we'll rename the group ...
Stanford CS193p
Fall 2013

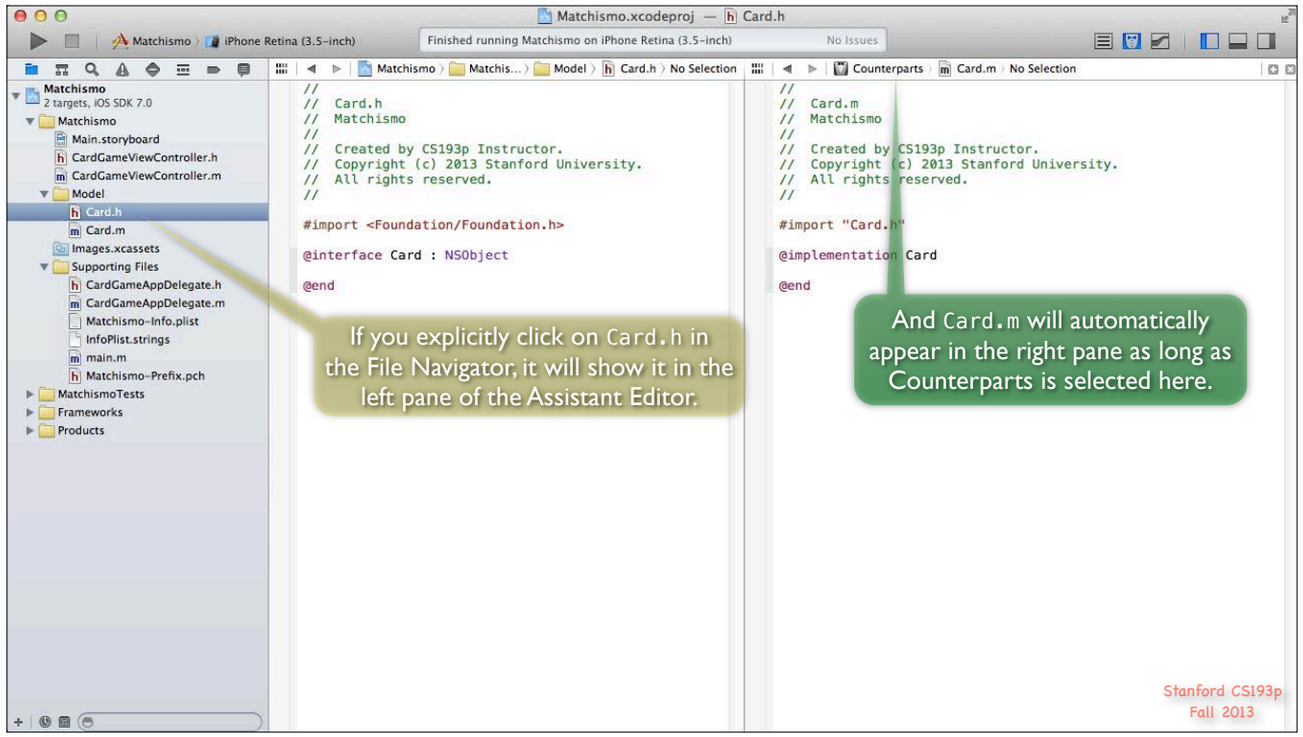

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Model
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchis... › Model › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
... to Model.
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
Stanford CS193p
Fall 2013

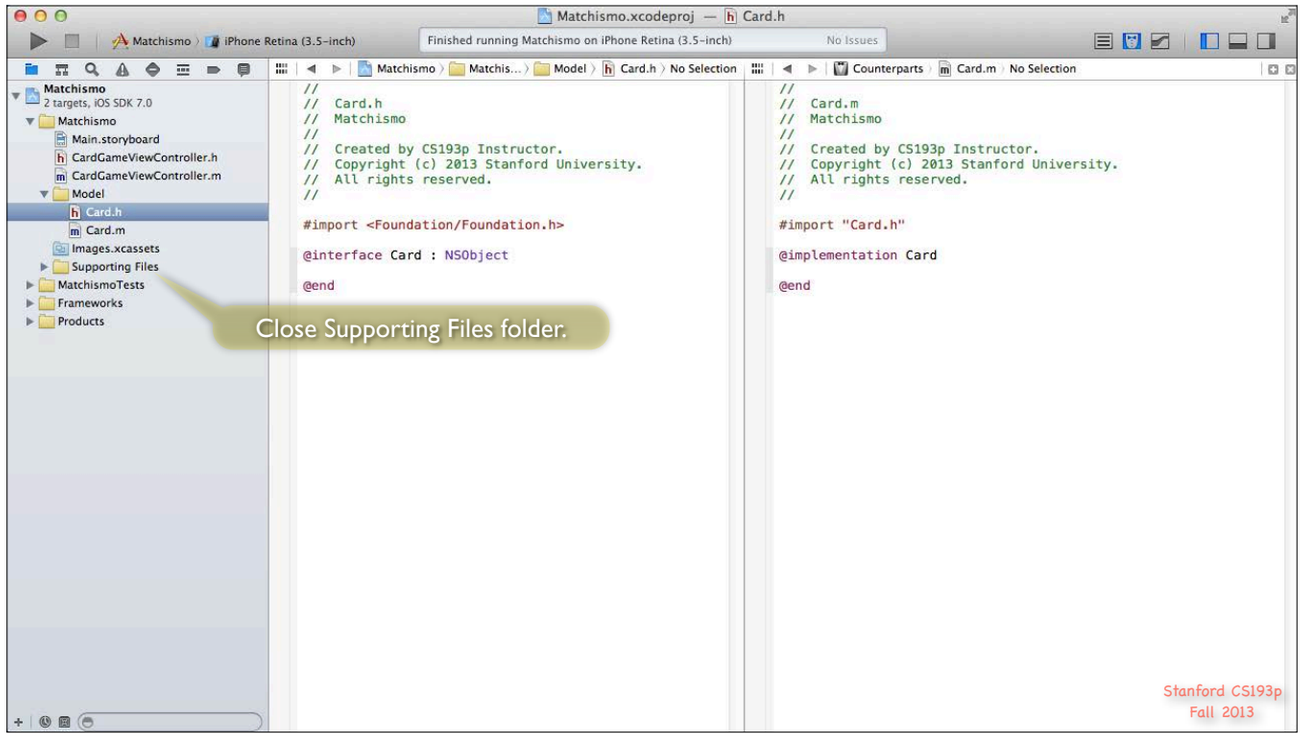

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Model
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Card.h › No Selection
Counterparts › Card.m › No Selection
// Card.h // Matchismo // // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
// Card.m // Matchismo // // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import "Card.h"
@implementation Card
@end
A group in the File Navigator can be linked to a directory in the filesystem or not, as you prefer. You control this from the File Inspector in the Utilities area.
Identity and Type
Name Card.h
Absolute Path
Relative to Group
Relative to Project
Relative to Developer Directory
Relative to Build Products
Relative to SDK
Target Membership
Matchismo
MatchismoTests
Text Settings
Text Encoding Default - Unicode (UTF-8)
Line Endings Default - OS X / Unix (LF)
Indent Using Spaces
Widths 4 4
Tab Indent
Wrap lines
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
Model
Card.h
Card.m
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchis... › Model › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
You can drag things around in the File Navigator to put them in whatever order you want.
Stanford CS193p
Fall 2013

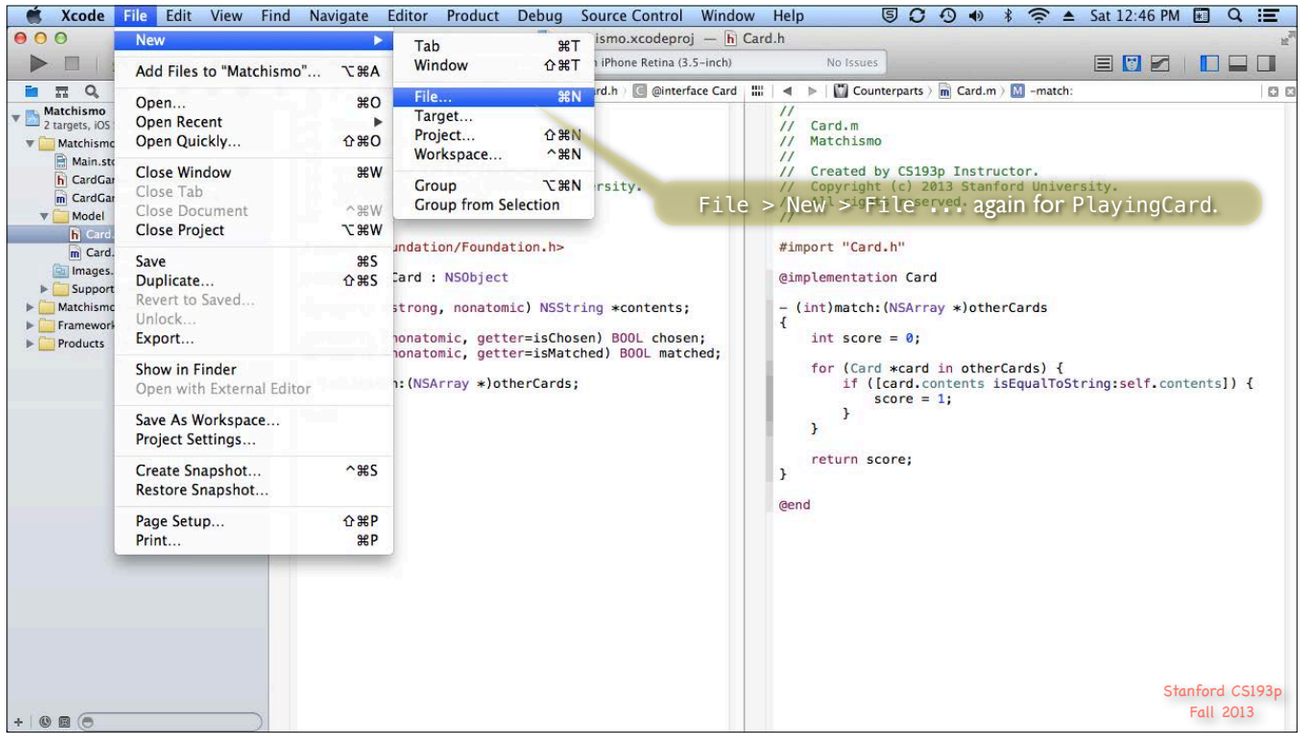

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › Matchis... › Model › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
Stanford CS193p
Fall 2013

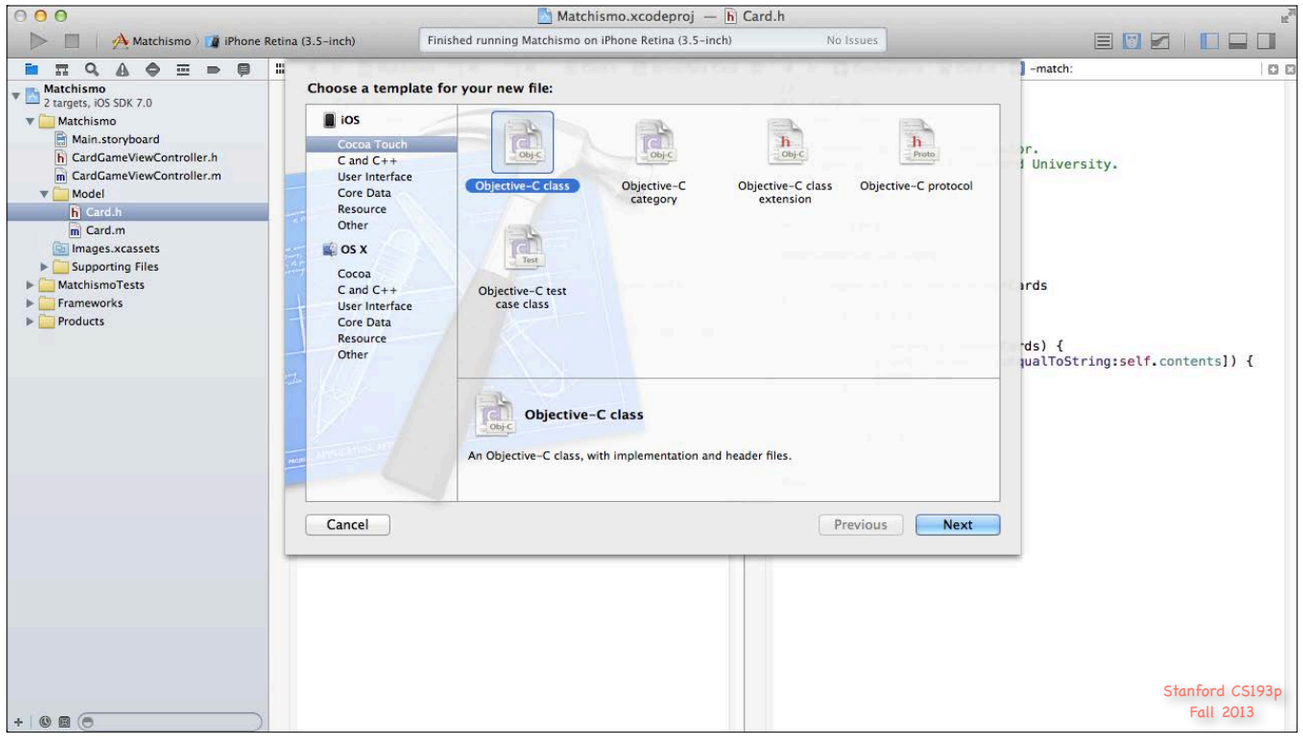

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
InfoPlist.strings
main.m
Matchismo-Prefix.pch
MatchismoTests
Frameworks
Products
Matchismo › Matchis... › Model › Card.h › No Selection
Counterparts › Card.m › No Selection
// Card.h // Matchismo // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
@end
// Card.m // Matchismo // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import "Card.h"
@implementation Card
@end
For example, often we'll drag the AppDelegate.[mh] into Supporting Files group since we rarely edit them.
Stanford CS193p
Fall 2013

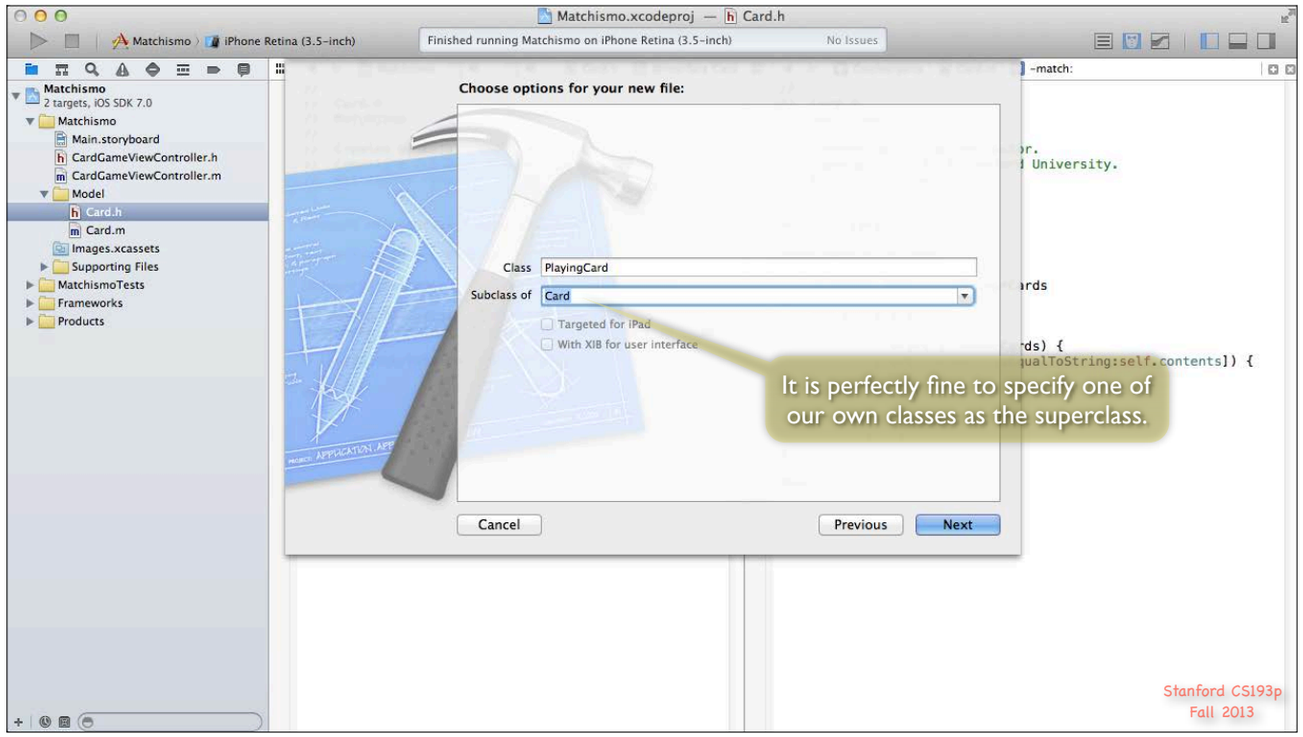

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
CardGameAppDelegate.h
CardGameAppDelegate.m
Matchismo-Info.plist
InfoPlist.strings
main.m
Matchismo-Prefix.pch
MatchismoTests
Frameworks
Products
Matchismo › Matchis... › Model › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
If you explicitly click on Card.h in the File Navigator, it will show it in the left pane of the Assistant Editor.
And Card.m will automatically appear in the right pane as long as Counterparts is selected here.
Stanford CS193p
Fall 2013

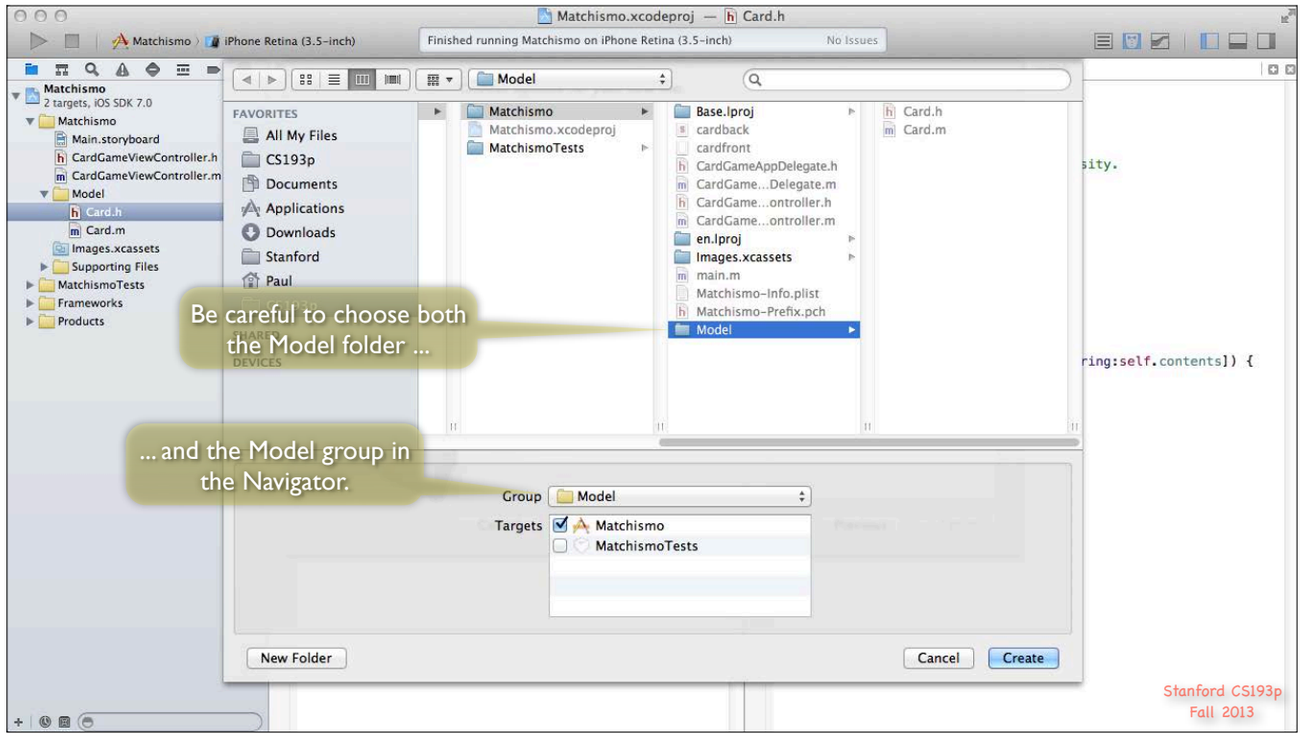

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Close Supporting Files folder.
Matchismo › Matchis... › Model › Card.h › No Selection
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@end
Counterparts › Card.m › No Selection
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
@end
Stanford CS193p
Fall 2013

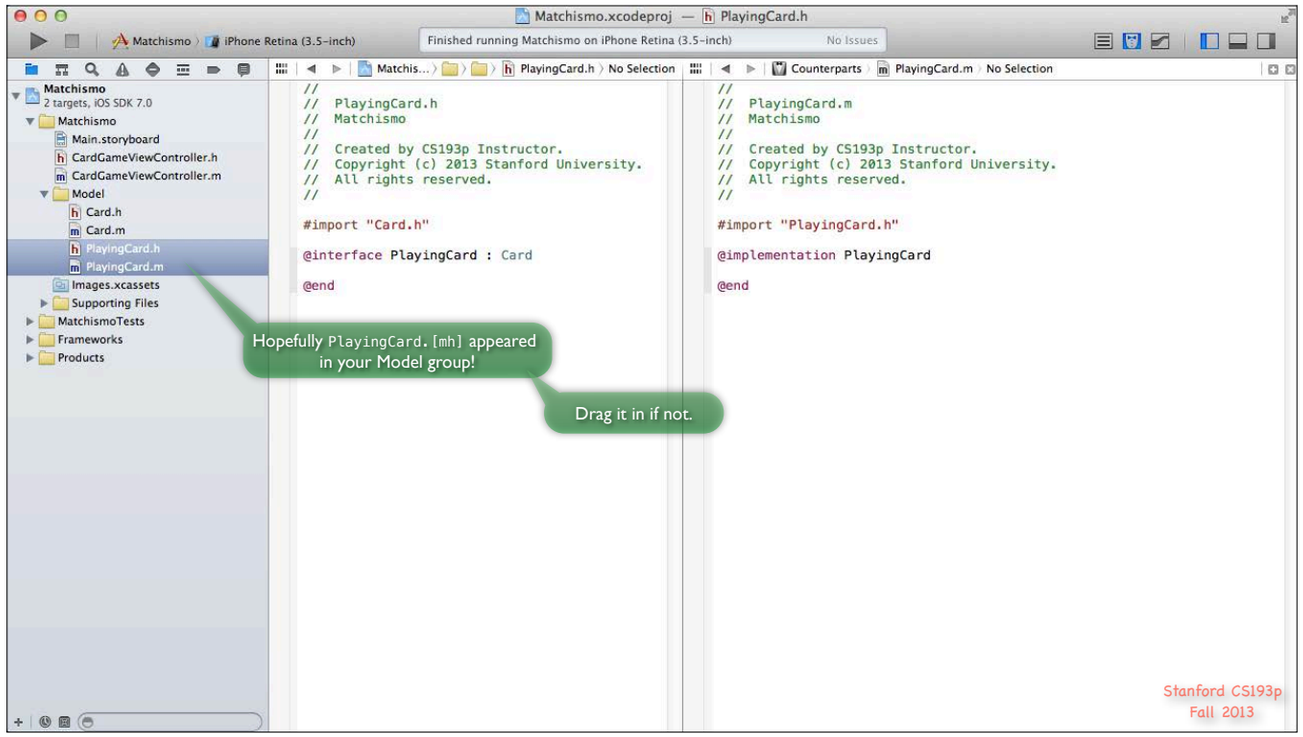

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchismo › M... › M... › Card.h › @interface Card
//
// Card.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import <Foundation/Foundation.h>
@interface Card : NSObject
@property (strong, nonatomic) NSString *contents;
@property (nonatomic, getter=isChosen) BOOL chosen;
@property (nonatomic, getter=isMatched) BOOL matched;
- (int)match:(NSArray *)otherCards;
@end
Counterparts › Card.m › -match:
//
// Card.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@implementation Card
- (int)match:(NSArray *)otherCards
{
int score = 0;
for (Card *card in otherCards) {
if ([card.contents isEqualToString:self.contents]) {
score = 1;
}
}
return score;
}
@end
Type in the code for Card.[mh].
Now let's move on to creating templates for the Deck, PlayingCard and PlayingCardDeck classes.
Stanford CS193p
Fall 2013

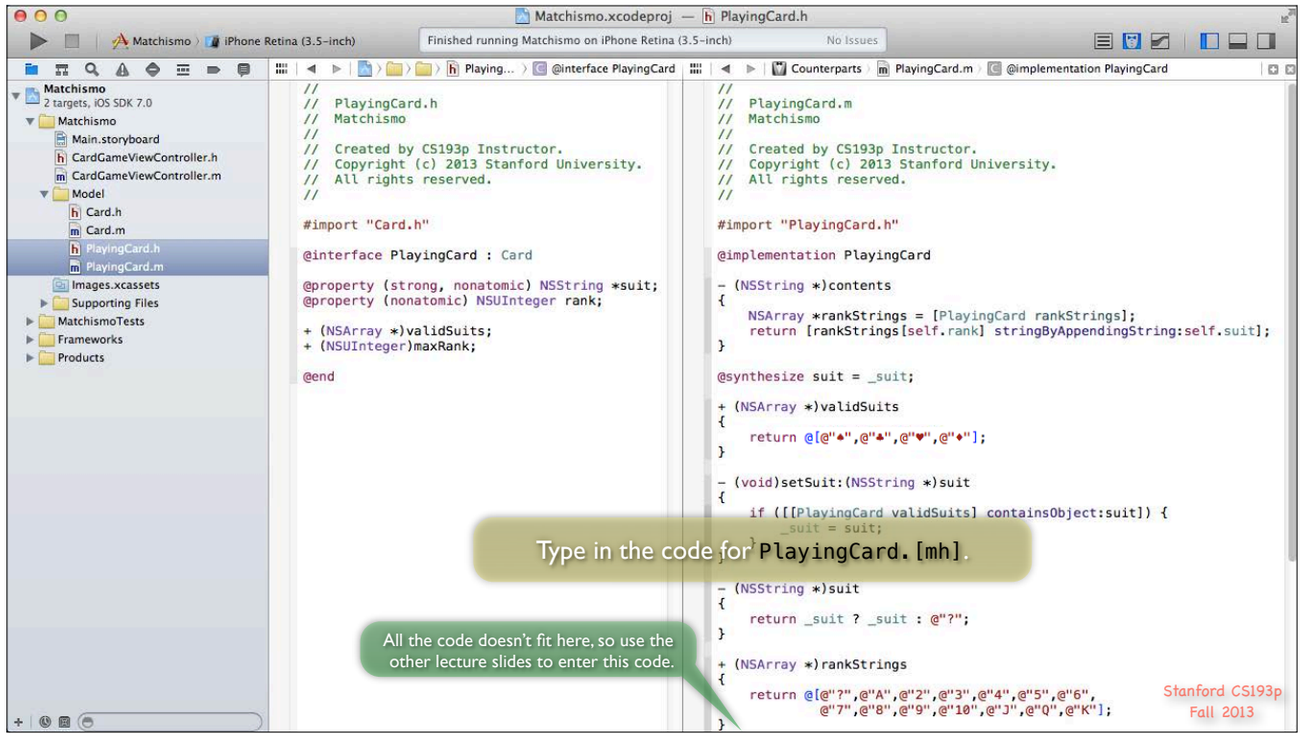

File > New > File ... again for PlayingCard.
Stanford CS193p
Fall 2013

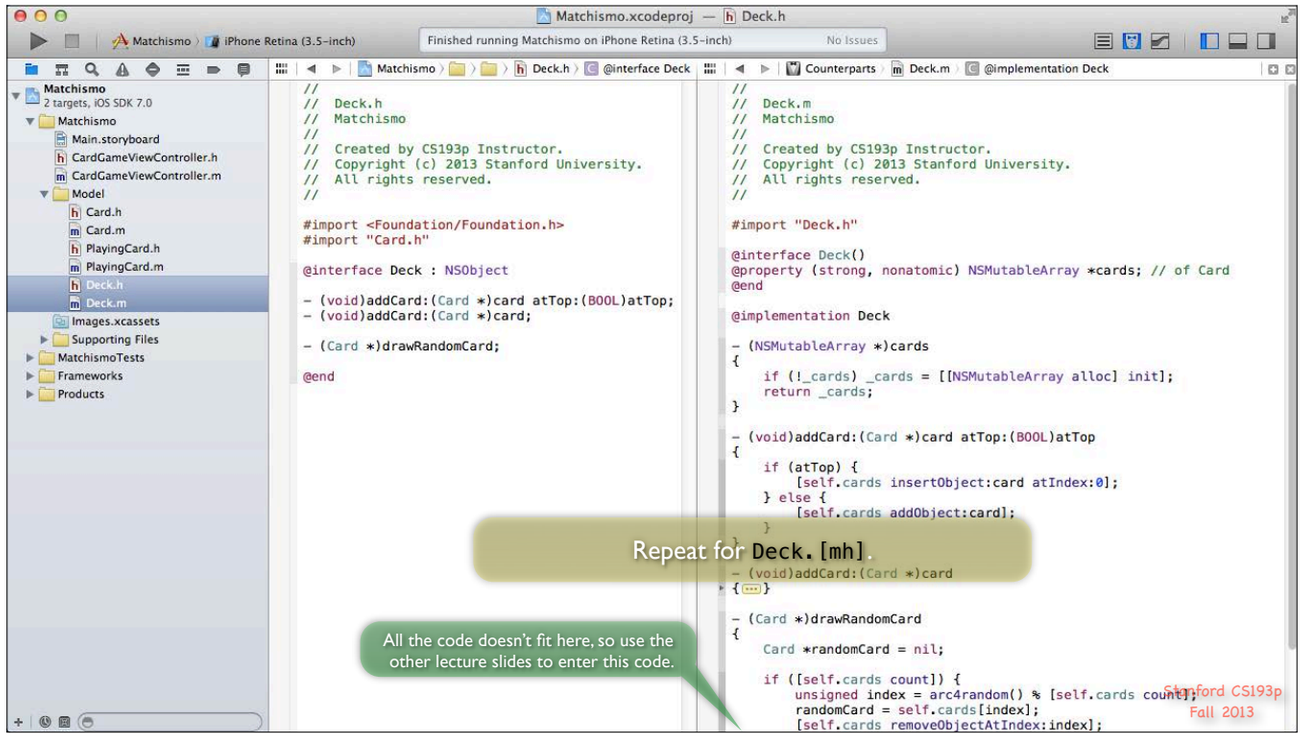

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Choose a template for your new file:
iOS
Cocoa Touch
C and C++
User Interface
Core Data
Resource
Other
OS X
Cocoa
C and C++
User Interface
Core Data
Resource
Other
Objective-C class
Objective-C category
Objective-C class extension
Objective-C protocol
Objective-C test case class
Objective-C class
An Objective-C class, with implementation and header files.
Cancel
Previous
Next
-match:
University.
rds
rds) {
qualToString:self.contents]) {
Stanford CS193p
Fall 2013

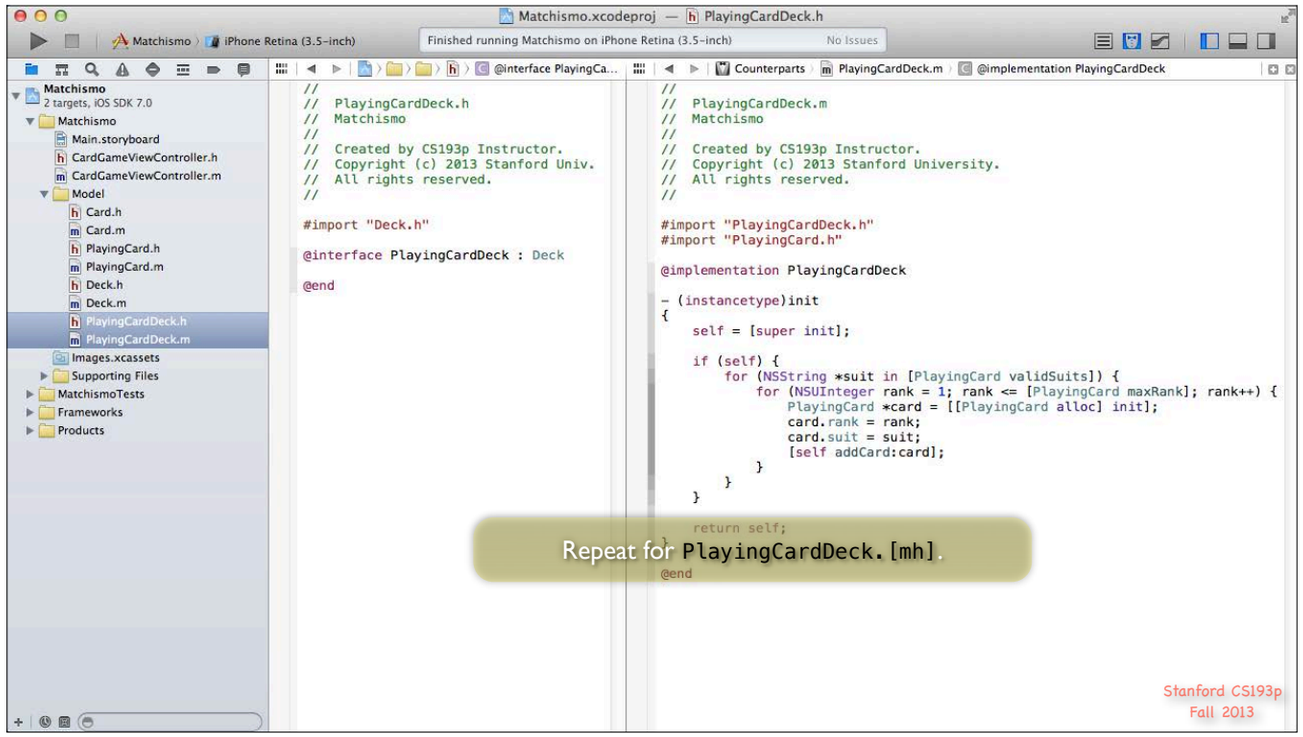

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Choose options for your new file:
Class
PlayingCard
Subclass of
Card
Targeted for iPad
With XIB for user interface
Cancel
Previous
Next
It is perfectly fine to specify one of our own classes as the superclass.
Stanford CS193p
Fall 2013

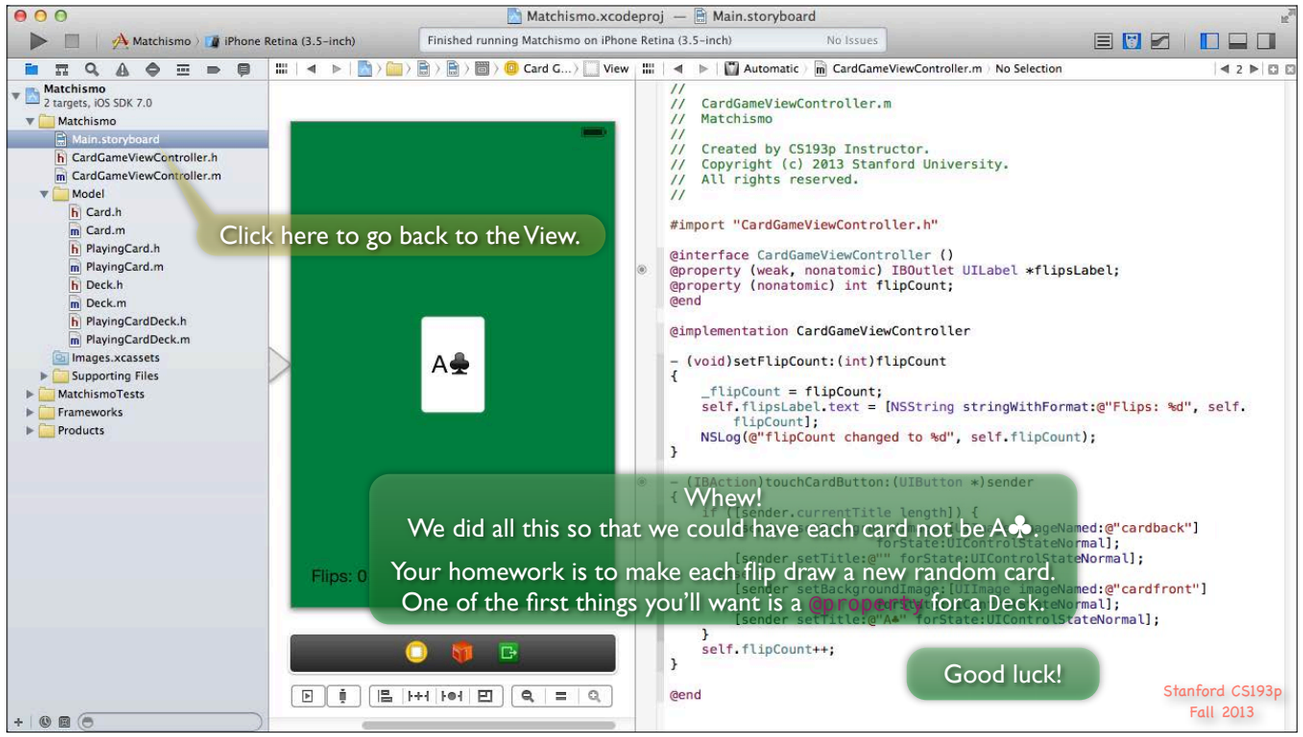

Matchismo.xcodeproj — Card.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Model
FAVORITES
All My Files
CS193p
Documents
Applications
Downloads
Stanford
Paul
DEVICES
Matchismo
Matchismo.xcodeproj
MatchismoTests
Base.lproj
cardback
cardfront
CardGameAppDelegate.h
CardGame...Delegate.m
CardGame...ontroller.h
CardGame...ontroller.m
en.lproj
Images.xcassets
main.m
Matchismo-Info.plist
Matchismo-Prefix.pch
Model
Card.h
Card.m
Be careful to choose both the Model folder ...
... and the Model group in the Navigator.
Group
Model
Targets
Matchismo
MatchismoTests
New Folder
Cancel
Create
sity.
ring:self.contents]) {
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — PlayingCard.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
PlayingCard.h
PlayingCard.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Matchis… › PlayingCard.h › No Selection
Counterparts › PlayingCard.m › No Selection
// PlayingCard.h // Matchismo // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import "Card.h"
@interface PlayingCard : Card
@end
// PlayingCard.m // Matchismo // Created by CS193p Instructor. // Copyright (c) 2013 Stanford University. // All rights reserved. //
#import "PlayingCard.h"
@implementation PlayingCard
@end
Hopefully PlayingCard.[mh] appeared in your Model group!
Drag it in if not.
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — PlayingCard.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
PlayingCard.h
PlayingCard.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Playing... › @interface PlayingCard
Counterparts › PlayingCard.m › @implementation PlayingCard
//
// PlayingCard.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "Card.h"
@interface PlayingCard : Card
@property (strong, nonatomic) NSString *suit;
@property (nonatomic) NSUInteger rank;
+ (NSArray *)validSuits;
+ (NSUInteger)maxRank;
@end
//
// PlayingCard.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "PlayingCard.h"
@implementation PlayingCard
- (NSString *)contents
{
NSArray *rankStrings = [PlayingCard rankStrings];
return [rankStrings[self.rank] stringByAppendingString:self.suit];
}
@synthesize suit = _suit;
+ (NSArray *)validSuits
{
return @[@"♠",@"♣",@"♥",@"♦"];
}
- (void)setSuit:(NSString *)suit
{
if ([[PlayingCard validSuits] containsObject:suit]) {
_suit = suit;
}
}
- (NSString *)suit
{
return _suit ? _suit : @"?";
}
+ (NSArray *)rankStrings
{
return @[@"?",@"A",@"2",@"3",@"4",@"5",@"6",
@"7",@"8",@"9",@"10",@"J",@"Q",@"K"];
}
Type in the code for PlayingCard.[mh].
All the code doesn't fit here, so use the other lecture slides to enter this code.
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — Deck.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
PlayingCard.h
PlayingCard.m
Deck.h
Deck.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Deck.h › @interface Deck
Counterparts › Deck.m › @implementation Deck
Repeat for Deck.[mh].
All the code doesn't fit here, so use the other lecture slides to enter this code.
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — PlayingCardDeck.h
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Matchismo
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
PlayingCard.h
PlayingCard.m
Deck.h
Deck.m
PlayingCardDeck.h
PlayingCardDeck.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
@interface PlayingCa...
Counterparts
PlayingCardDeck.m
@implementation PlayingCardDeck
//
// PlayingCardDeck.h
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford Univ.
// All rights reserved.
//
#import "Deck.h"
@interface PlayingCardDeck : Deck
@end
//
// PlayingCardDeck.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "PlayingCardDeck.h"
#import "PlayingCard.h"
@implementation PlayingCardDeck
- (instancetype)init
{
self = [super init];
if (self) {
for (NSString *suit in [PlayingCard validSuits]) {
for (NSUInteger rank = 1; rank <= [PlayingCard maxRank]; rank++) {
PlayingCard *card = [[PlayingCard alloc] init];
card.rank = rank;
card.suit = suit;
[self addCard:card];
}
}
}
return self;
}
@end
Repeat for PlayingCardDeck.[mh].
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — Main.storyboard
Matchismo › iPhone Retina (3.5-inch)
Finished running Matchismo on iPhone Retina (3.5-inch)
No Issues
Matchismo
2 targets, iOS SDK 7.0
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Model
Card.h
Card.m
PlayingCard.h
PlayingCard.m
Deck.h
Deck.m
PlayingCardDeck.h
PlayingCardDeck.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
Click here to go back to the View.
A♣
Flips: 0
Whew!
We did all this so that we could have each card not be A♣.
Your homework is to make each flip draw a new random card.
One of the first things you'll want is a @property for a Deck.
Good luck!
Stanford CS193p
Fall 2013
//
// CardGameViewController.m
// Matchismo
//
// Created by CS193p Instructor.
// Copyright (c) 2013 Stanford University.
// All rights reserved.
//
#import "CardGameViewController.h"
@interface CardGameViewController ()
@property (weak, nonatomic) IBOutlet UILabel *flipsLabel;
@property (nonatomic) int flipCount;
@end
@implementation CardGameViewController
- (void)setFlipCount:(int)flipCount
{
_flipCount = flipCount;
self.flipsLabel.text = [NSString stringWithFormat:@"Flips: %d", self.flipCount];
NSLog(@"flipCount changed to %d", self.flipCount);
}
- (IBAction)touchCardButton:(UIButton *)sender
{
self.flipCount++;
}
@end