SWIFT - 3

Ahmed Yossef

Agenda

- MVC
- Main Components
- Window, View and View Controller
- Echo App

MVC
Model View Controller

MVC - Model View Controller

Why MVC

- Same interfacing and isolation concepts
- It makes you able to change business rules without affecting the GUI
- You can also change the GUI and keep the logic the same

Model

- It is what your application is, Not how your application look like
- Model is not connected directly to your UI
- Model may stay the same across different UIs

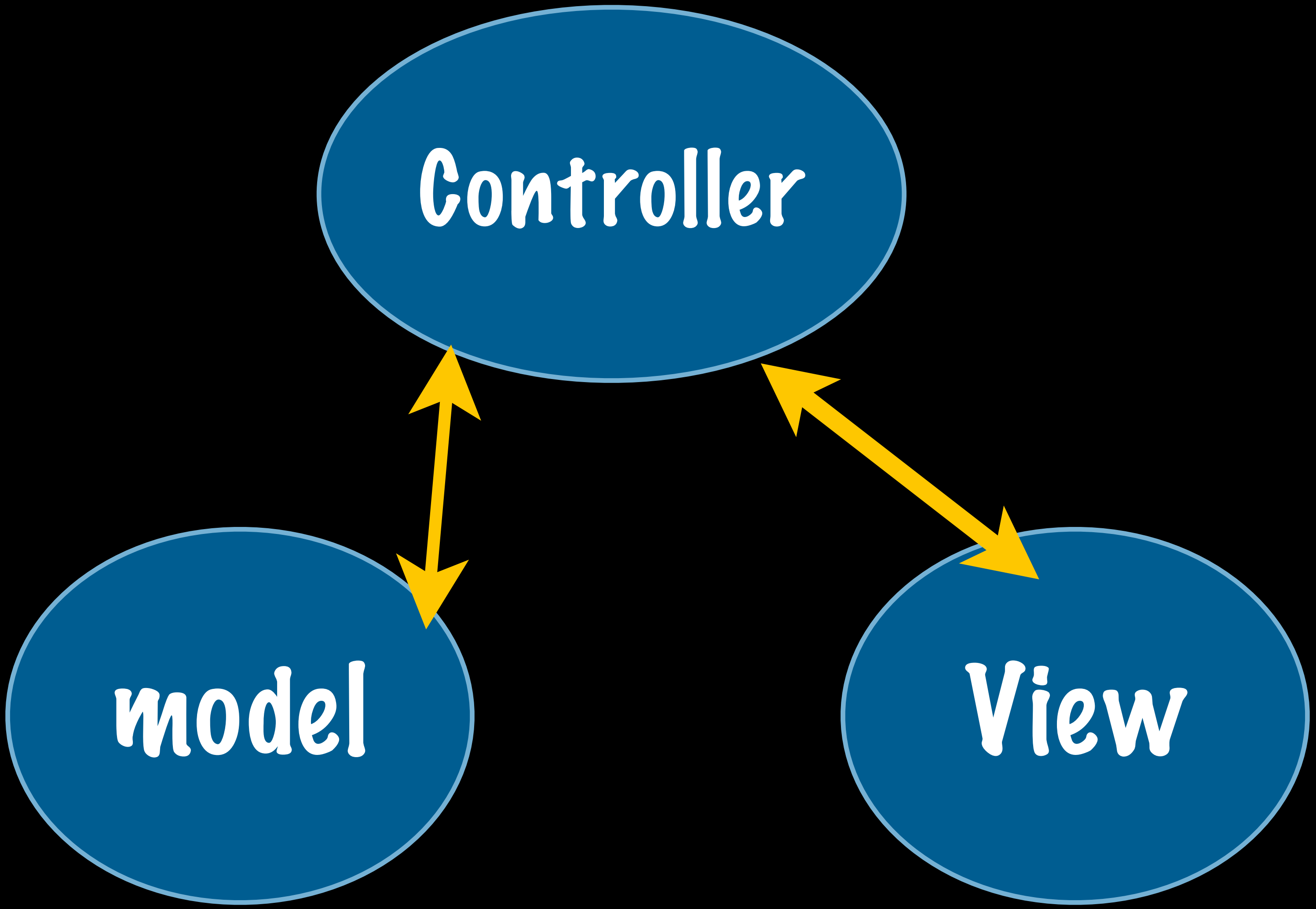
View

- It is how the data represented
- It enables user to interact with data
- No data should be stored here, except cache
- It the only layer that should be affected when you are trying to migrate a command line app to GUI

Controller

- It is the communicator between the model and view
- Updates model when view requires
- Updates view when model changes
- It translate user’s actions on the view to a system function and ask the model to execute it

Allowed Intercommunications



GAD Example

1. Casher: is your view
2. Receptor: is your controller
3. Chef: is your model

MVC Example

- Calculator example

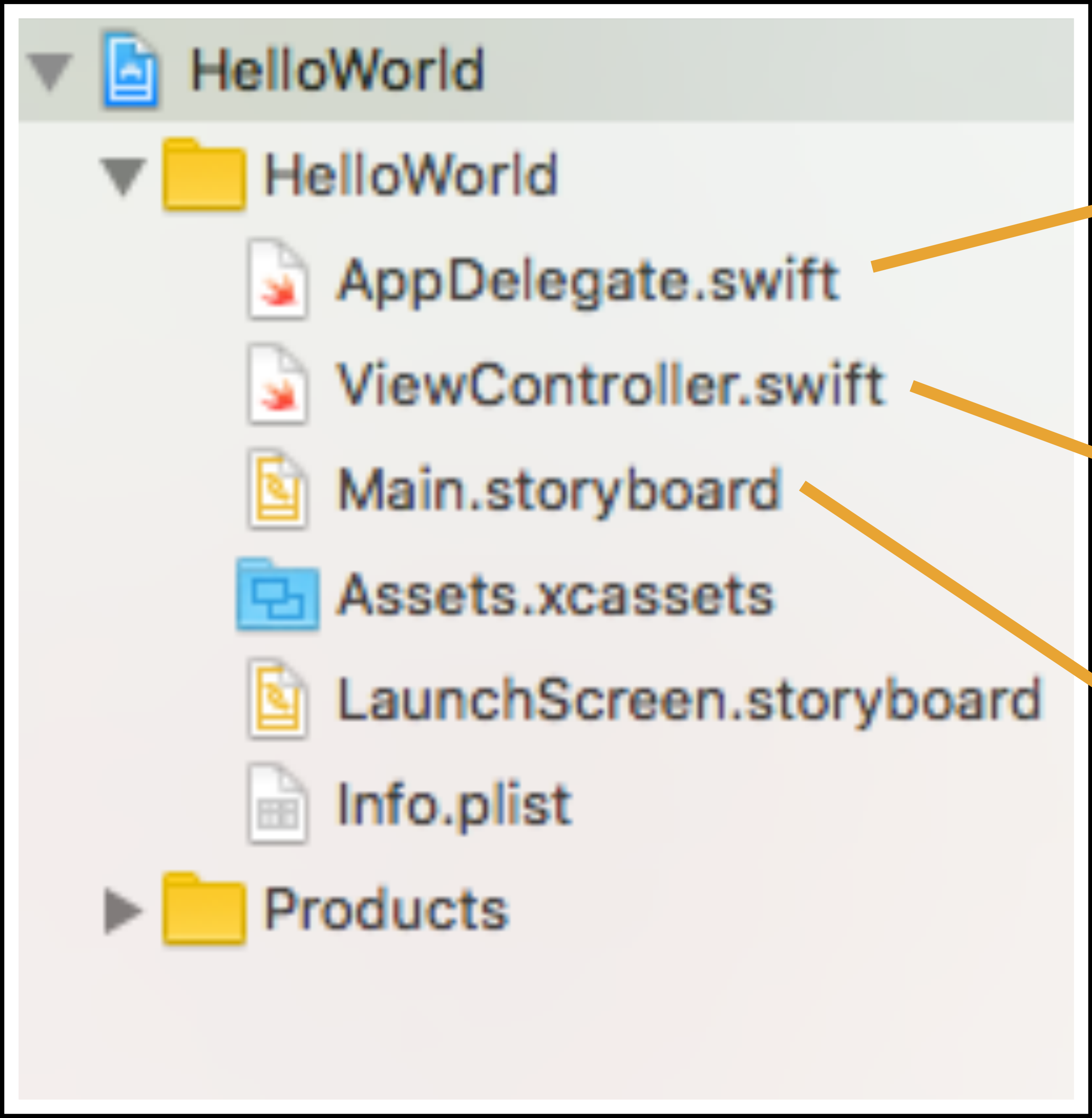
MVC in iOS Development

- Nib files, which represents the View, is used to design and implement the UI

- Nib files are used to LINK the designed view with the variables on the corresponding controller

Getting started

XCode



- HelloWorld
  - HelloWorld
    - AppDelegate.swift
    - ViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - Products

Main class of the app
used for initialisation and
interaction with OS

Sub class of UIViewController
It is a controller of certain View

Where all UI design goes

Basic Concepts

Screens are designed on the storyboard file

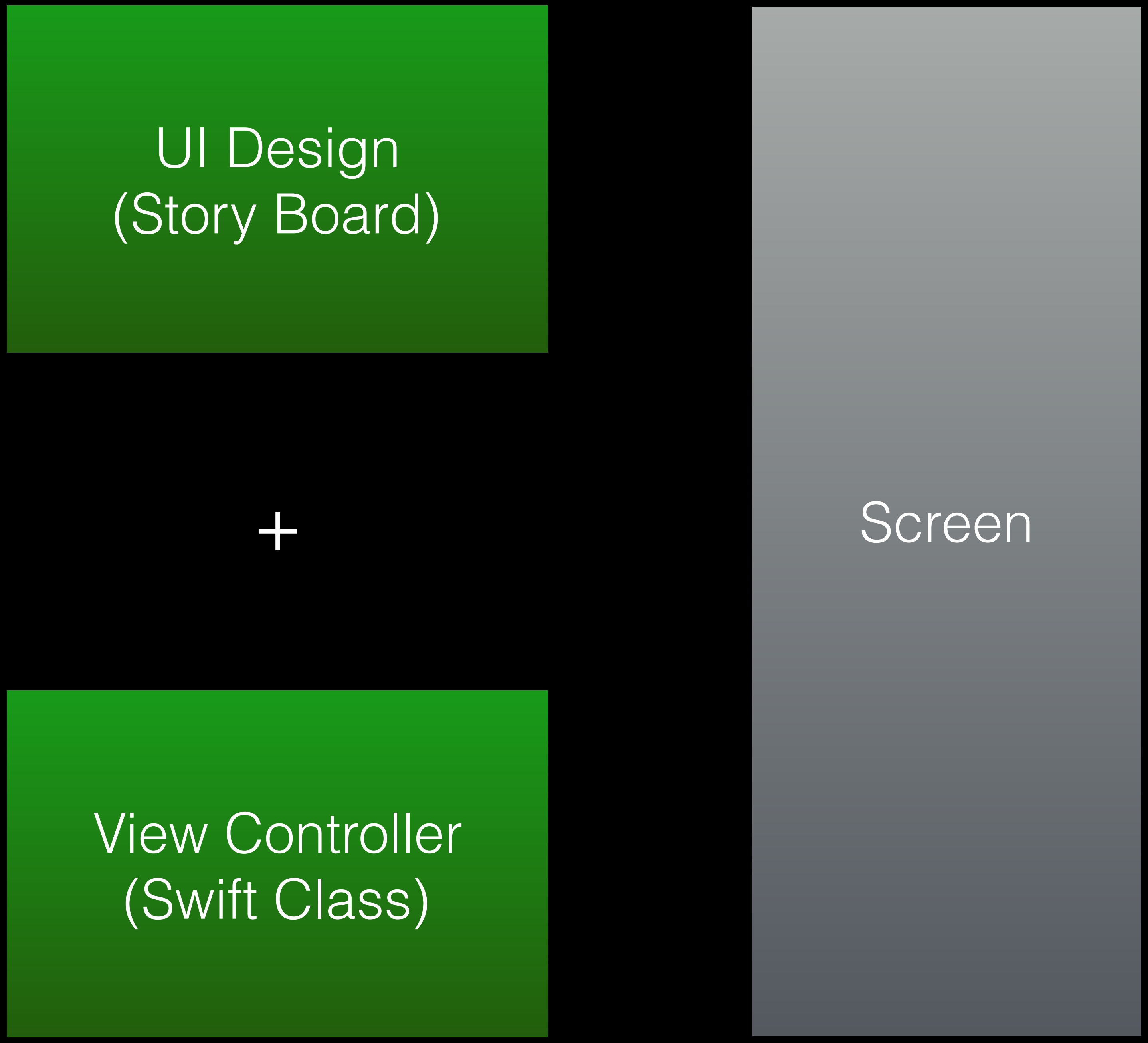
Each screen has an **owner** (subclass of UIViewController)

Owner contains:

Variables attached to GUI elements

Methods linked to event sources

App Structure - User



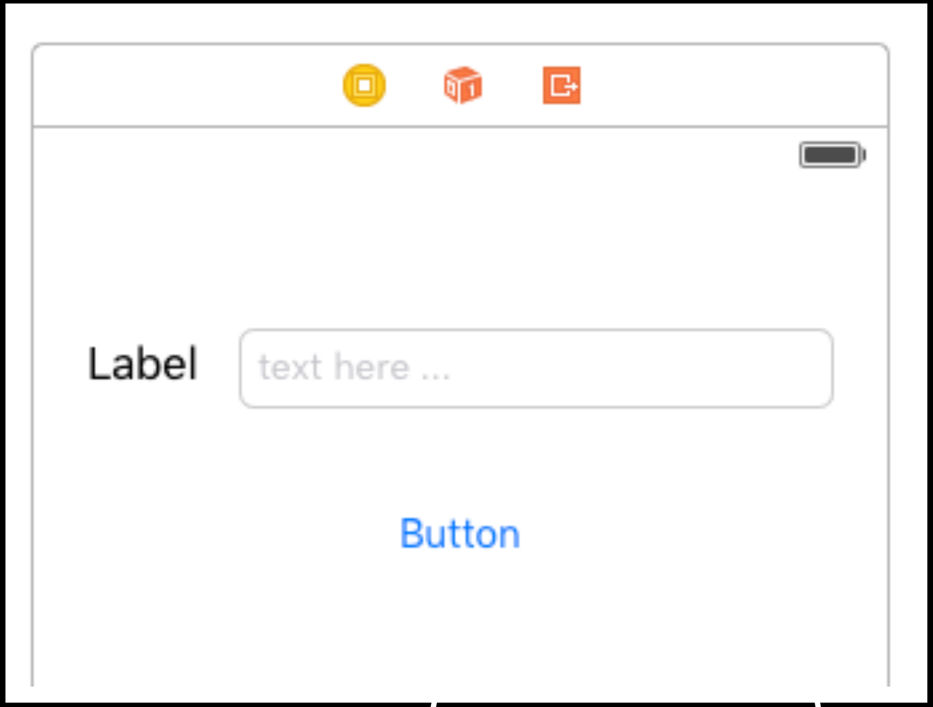
UI Design
(Story Board)

+

View Controller
(Swift Class)

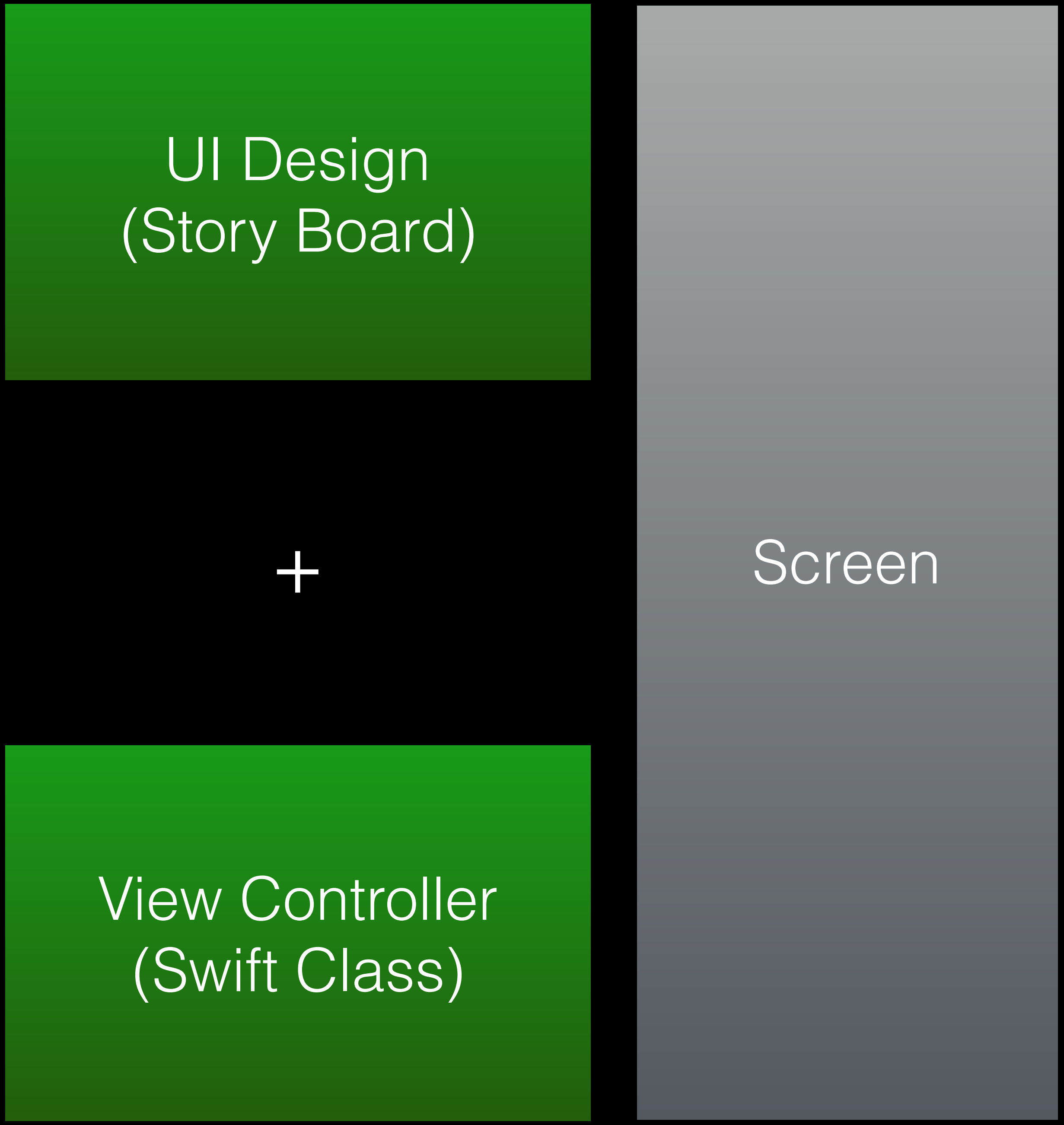
Screen

App Structure - User



```
@IBAction func
buttonAction(){}
```

```
@IBOutlet weak var
userInput:UITextField
```



UI Design
(Story Board)

+

View Controller
(Swift Class)

Screen

main Components

Files

- AppDelegate.swift
- ViewController.swift
- Main.storyboard

App Delegate

- It is a class with call back methods that
- Let iOS interact with application like
  - didFinishLaunchingWithOptions
  - applicationDidEnterBackground
  - applicationWillTerminate

View Controller

- Controlling the view
- Contains the action methods
- Contains the UI components objects

Main Storyboard

- UI part
- Contains the design
- Holds more than one screen

IBOutlet and IBAction

IBOutlet

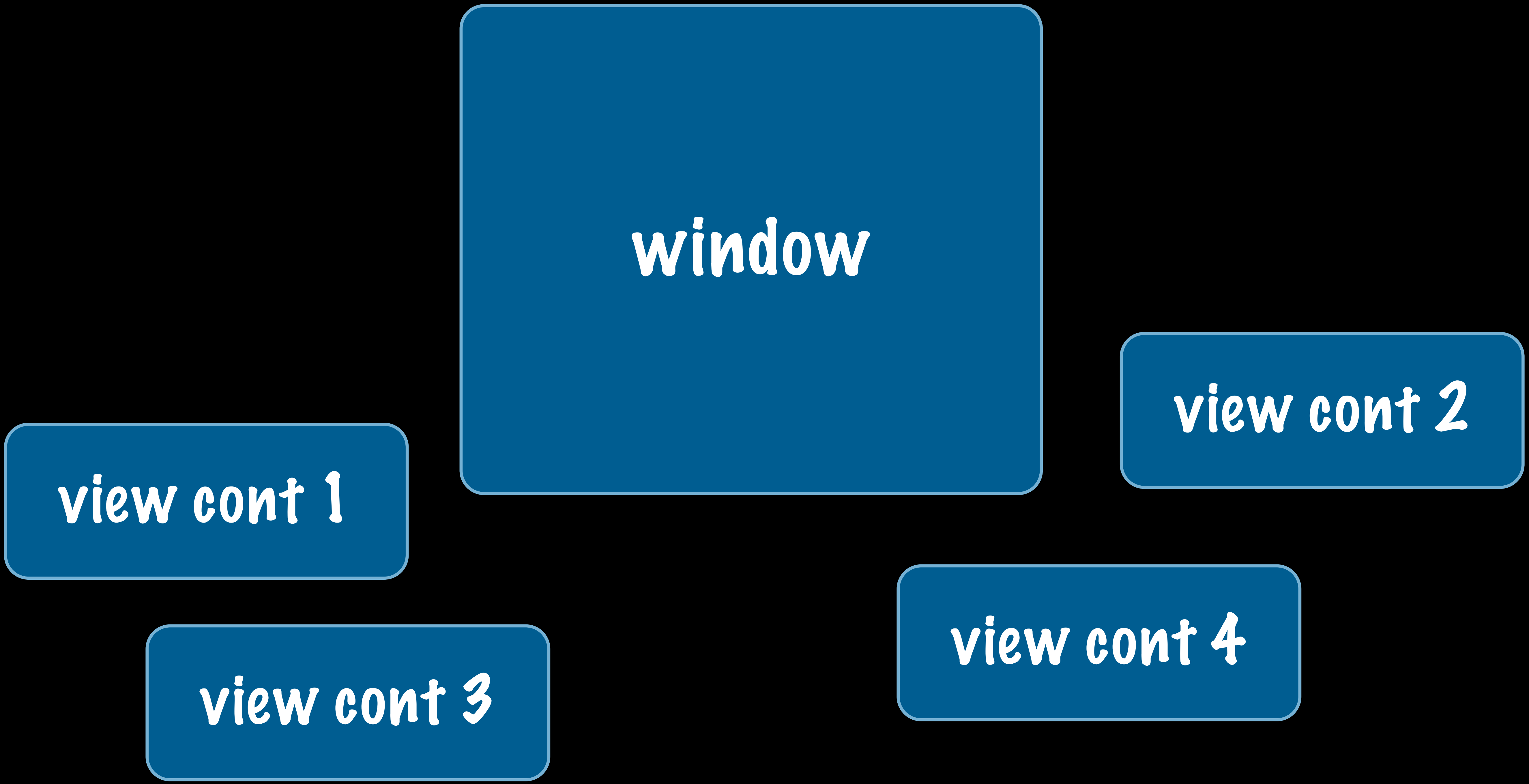
It is used to identify that this element should appear in the interface builder for **LINKING** with some elements

IBAction

It is used to identify that this method is a call back method, called by the system upon GUI event. It makes this method appear in the interface builder for **LINKING** with some events

UIKit

GUI



window

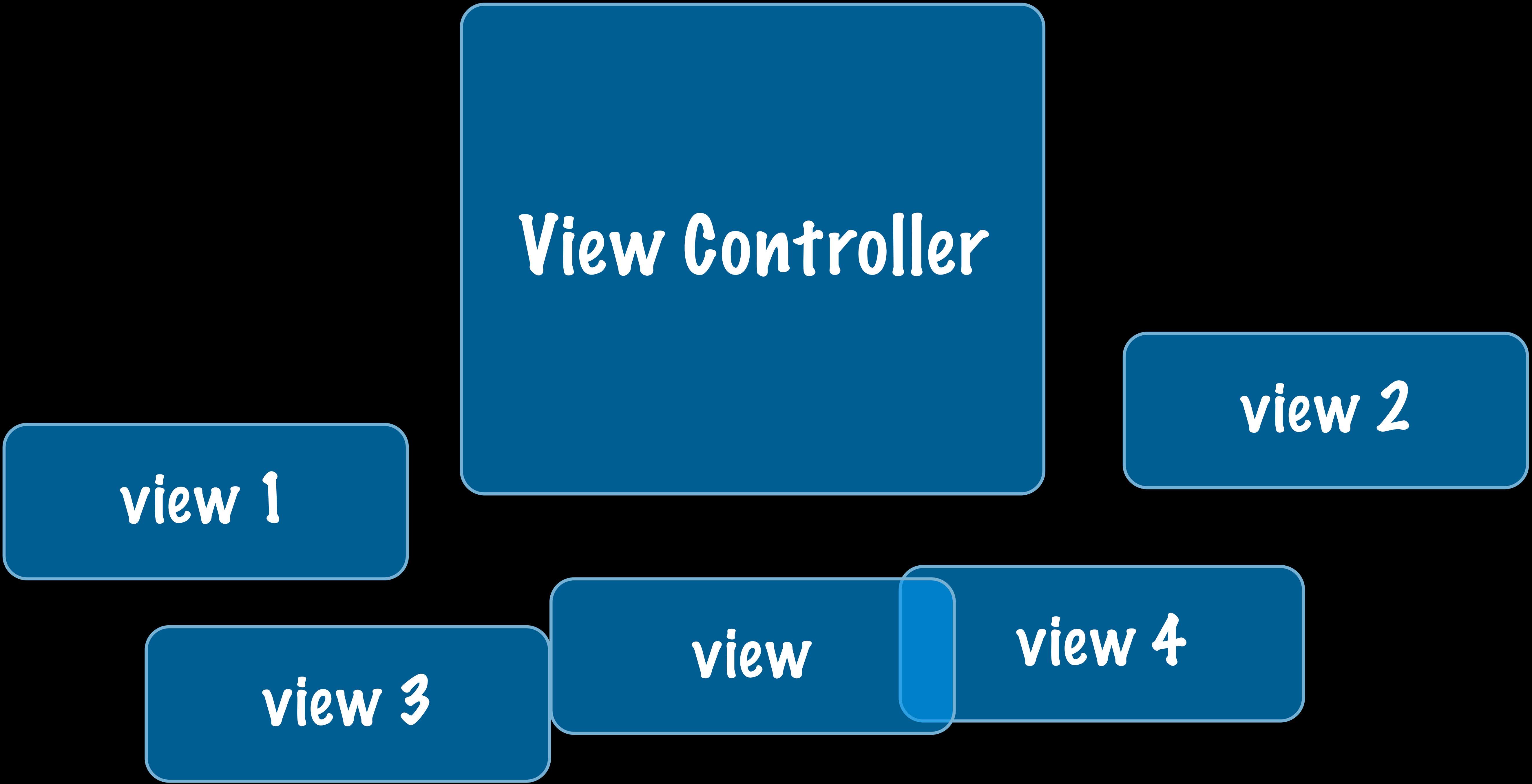
view cont 2

view cont 1

view cont 4

view cont 3

GUI



View Controller

view 1

view 2

view 3

view

view 4

More Data

NSArray

It is an array object

It is fixed (Immutable) - must initialise

Main Methods

addObject

objectAtIndex

NSMutableArray

It is an array object

It is changeable (mutable)

Main Methods

addObject

objectAtIndex

NSDictionary

It is a hashmap object

It is fixed (Immutable) - must initialise

Main Methods

setValue ForKey

valueForKey

NSMutableDictionary

It is a hash map object

It is changeable (Mutable)

Main Methods

setValue ForKey

valueForKey

Optional ?/!

Basic Concepts

Optional is a variable that might not has a value (nil)

String? is not the same as String

To make s:String? is a String .. use !

? defines uncertainty

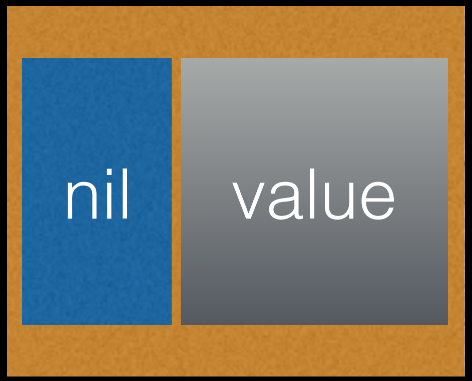
! defines exact value

Optional

```
var sExact:String
```

value

```
var sOptional:String?
```



nil

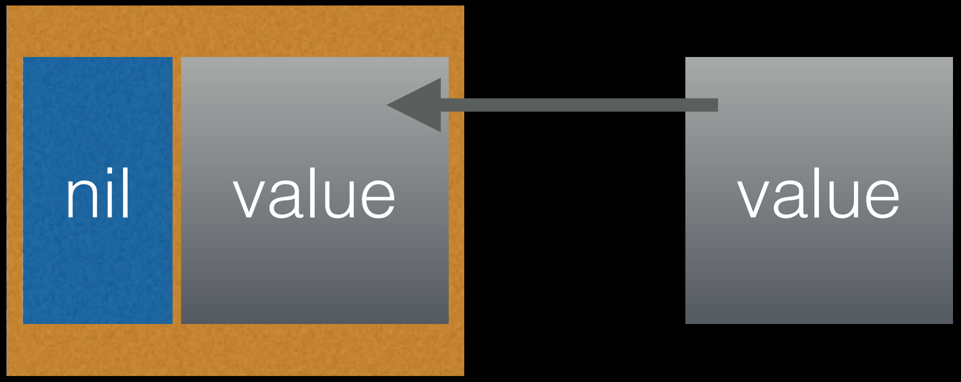
value

```
sExact = sOptional // wrong

sExact = sOptional ! // OK
```

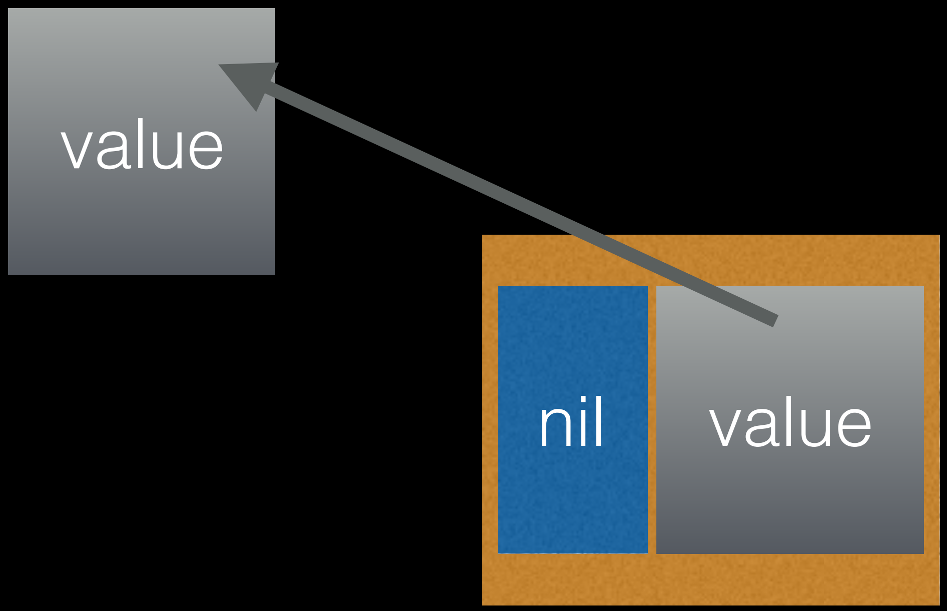
Optional

sOptional = sExact

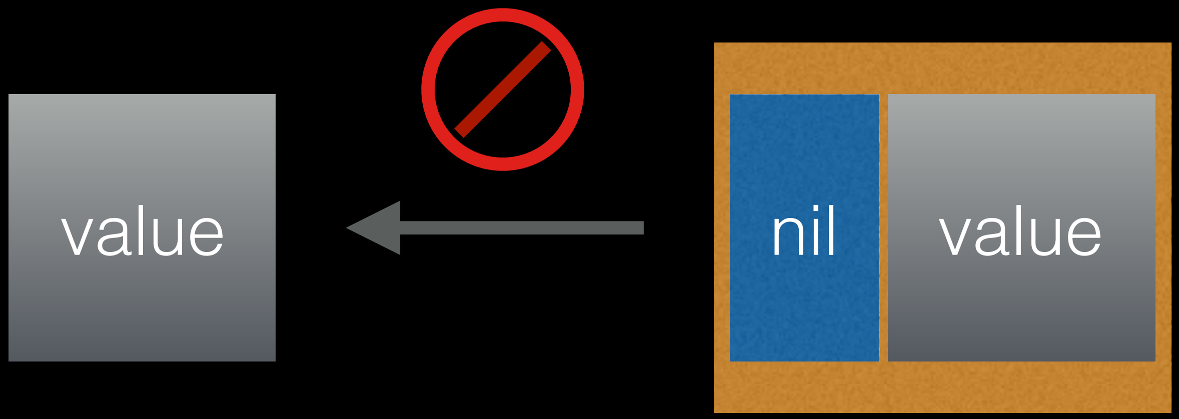


Optional

sExact = sOptional !



sExact = sOptional


value
nil
value