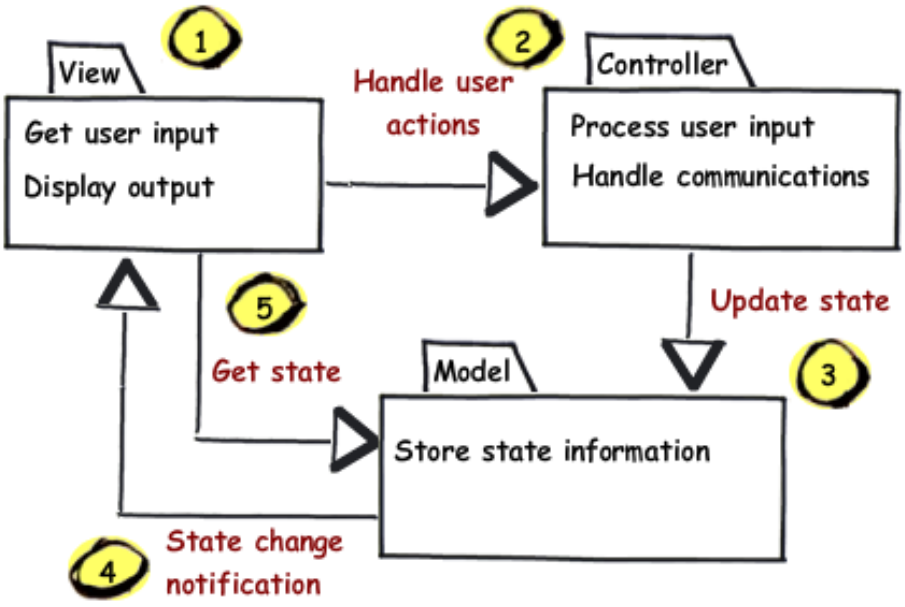
iPhone Programming
Model-View-Controller

Tim Gegg-Harrison

Model-View-Controller



Model-View-Controller

- **Controller** provides an interface between the **model** (program logic) and the **view** (user interactions)

- Simple View Controller and View
  - `UIViewController`
  - `UIView`

Model-View-Controller …

- App Delegate (`UIApplicationDelegate`)
  - Contains several methods that are called when key events for the app occur
    - Finished launching – the following method is called when the app has finished launching. We create the window and view controller in this method.

```
optional func application(application: UIApplication,
didFinishLaunchingWithOptions launchOptions:
[NSObject : AnyObject?) -> Bool
```

    - About to be terminated – the following method is called when the app is about to terminate. We need to save relevant information about the app's state in this method.

```
optional func applicationWillTerminate(application:
UIApplication)
```

Model-View-Controller …

- App Delegate (`UIApplicationDelegate`)
  - Contains several methods that are called when key events for the app occur
    - Becoming active – the following method is called when the app moves from the inactive to active state. Restart tasks that were paused while the app was inactive in this method.
      `applicationDidBecomeActive()`
    - Becoming inactive – the following method is called when the app moves from the active to inactive state. Pause tasks (timers, etc.) in this method.
      `applicationWillResignActive()`

Model-View-Controller …

- App Delegate (`UIApplicationDelegate`)
  - Contains several methods that are called when key events for the app occur
    - Entering foreground – the following method is called when the app enters the foreground.
      `applicationWillEnterForeground()`
    - Entering background – the following method is called when the app has entered the background.
      `applicationDidEnterBackground()`

Model-View-Controller …

- App Delegate (`UIApplicationDelegate`)
  - Contains an instance of `UIWindow`

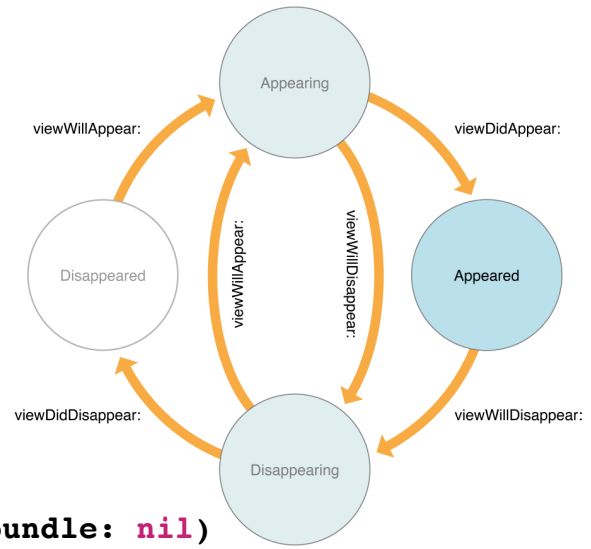
```
window = UIWindow(frame:
          UIScreen.mainScreen().bounds)
window?.makeKeyAndVisible()
```

  - We will also also add an instance of `UIViewController`

```
window?.rootViewController =
  ViewController(nibName: nil, bundle: nil)
```

Model-View-Controller …

- `UIViewController`
- `UIView`
- Creating a view for the view controller:




```
init() {
  super.init(nibName: nil, bundle: nil)
  self.view = UIView(frame:
              UIScreen.mainScreen().bounds)
}
```

Model-View-Controller …

```
let viewRect: CGRect = CGRectMake(10, 20, 100, 200)
let myView: UIView = UIView(frame: viewRect)
```

```
CGRect

CGPoint
    x
    y

CGSize
    width
    height

CGFloat
```

Model-View-Controller …

- Device orientation:
  - `UIInterfaceOrientationPortrait`
  - `UIInterfaceOrientationPortraitUpsideDown`
  - `UIInterfaceOrientationLandscapeLeft`
  - `UIInterfaceOrientationLandscapeRight`

Views

- Rectangular area on the screen
- Responsible for drawing and handling events in its defined rectangle
- Hierarchical
  - One superview
  - Many (or zero) subviews
    - Subview order matters – later ones are placed on top of earlier ones
- **`UIWindow`**
  - **`UIView`** at the top of the view hierarchy
  - Only one **`UIWindow`** in an iPhone application

Views …

- **`UIView`**
  - **`frame`**
    - `CGRect`
      - `CGRectMake(x, y, width, height)`
    - location (origin) and dimension of the `UIView` inside another `UIView`
  - **`bounds`**
    - `CGRect`
    - location (origin) and dimension of the `UIView` in its own coordinate system
  - **`center`**
    - `CGPoint`
      - Pair of `CGFloat`s (floating point numbers) for x, y coordinates
    - location of the center of the `UIView`
    - Note that `CGSize` is also a pair of `CGFloat`s for width and

Views …

- `UIScreen`
  - Class that provides dimensions of device
  - iPad device screen dimension is 768 x 1024
    - Aspect ratio 4:3
  - iPhone 4 device screen dimension is 320 x 480
    - Aspect ratio 3:2
  - iPhone 5 device screen dimension is 320 x 568
    - Aspect ration 16:9
  - iPhone 6/7 device screen dimension is 375 x 667
  - iPhone 6+/7+ device screen dimension is 414 x 736

- Retrieve the size of the device's screen (`CGSize`):

```
UIScreen.mainScreen().bounds.size
```

View Hierarchy

- Superview:

```
var superview: UIView? { get }
```

- Subviews:

```
var subviews: [AnyObject] { get }
```

- To create a new view `mySubview` and add it as a subview to the main view:

```
let frame: CGRect = UIScreen.mainScreen().bounds
let mySubview: UIView = UIView(frame: frame)
self.view.addSubview(mySubview)
```

View Hierarchy …

- Removing a view from its superview:

```
removeFromSuperview()
```

- Inserting a view in specified place in view hierarchy:

```
insertSubview(view: UIView, at: Int)
insertSubview(view: UIView, aboveSubview: UIView)
insertSubview(view: UIView, belowSubview: UIView)
```

- Lower views will only show through views on top if they are transparent, otherwise they are covered up

```
bringSubview(toFront: UIView)
sendSubview(toBack: UIView)
```