
**Universitatea din Bucuresti**
Facultatea de Matematica si Informatica

# AppleLab @ UniBuc
# Dezvoltare aplicatii iOS cu Swift 3

## - CURS 3 -

## Functii, Closure, Protocoale si Delegati, ViewControllers, Segues, NavigationController

22.03.2017

# iOS Course - 3

Noțiuni necesare pentru a continua dezvoltarea de aplicații sub Swift

- Functii (Swift este un limbaj functional)
- Closure (functie transmisa ca parametru)
- Protocol (mijloc de legatura intre obiecte)
- Delegat (implementeaza functionalitatile unui protocol)
- View (ceea ce este desenat pe ecran) *dumb
- Subviews (view-uri din interiorul unui alt View) *dumb
- ViewController (controleaza partea de View) *smart
- Segue ("legatura" dintre doua ViewControllere)
- NavigationController (controleaza mai multe ViewControllere)

# iOS Course - 3

FUNCTII

- Fara parametri

```
func printStaticMessage() {
    print("it works")
}
```

- Cu parametri si return

```
func sum(extA a: Int, __ b: Int) -> Int {
    return a + b
}
print(sum(extA: 5, 10)) // Afiseaza 15
```

- Function types

```
var addition: (Int, Int) -> Int = sum
print("Result of sum is: \(addition(31, 19))") // Afiseaza 50
```

# iOS Course - 3

FUNCTII

- Nested functions

```
func calcDecrement(forDecrement total: Int) -> () -> Int {
    var overallDecrement = 39 // Ce tip e?
    func decrementer() -> Int {
        overallDecrement -= total
        return overallDecrement
    }
    return decrementer // Ce se intampla aici?
}

let decrem = calcDecrement(forDecrement: 30)
print(decrem())  // Afiseaza 9
```

# iOS Course - 3

Closures

| Global Functions | Nested Functions | Closure Expressions |
| --- | --- | --- |
| Have a name. Do not capture any values | Have a name. Capture values from enclosing function | Unnamed Closures capture values from the adjacent blocks |

**Sintaxa:**

```
{ (parametri) -> tip_returnat in
    // do anything
}
```

# iOS Course - 3

Closures

```
let sum = { (a: Int, b: Int) -> Int in
    return a + b
}
let digits = sum(10, 20)
print(digits) // Afiseaza 30

Ex 2:
var sum: (Int, Int) -> Int
sum = { (a: Int, b: Int) -> Int in
    return a + b
}
print(sum(15, 20)) // Afiseaza 35
```

# iOS Course - 3

## Closures

**Culese din practica:**

```
var students = ["Mihai", "Ana", "Andrei", "Maria"]
students.map ( { (student: String) in
        print("\(student) are mere")
} ) // Ce va afisa?
```

**Exemplu 2 (tot din practica) :**

```
var result = students.map ( ) {
      "\($0) este student"
} // Ce tip de date este result ?
print(result) // Ce va afisa?
```

# iOS Course - 3

## Protocoale & Delegati

**Sintaxa:**

```
protocol nume_protocol {
    var computed_var: Int? { get }
    func metoda()
}
```

- **Un protocol nu poate contine stored property-uri**
- **Un protocol doar defineste functionalitatea, nu o si implementeaza**

```
protocol Demo {
    func myDemo() {                    Protocol methods may not have bodies
        print("it works")
    }
}
```

# iOS Course - 3

Protocoale & Delegati

- Clasa (delegat) care implementeaza protocolul se ocupa de definirea body-ului functiilor / implementarea metodelor definite in protocol

- ESTE OBLIGATORIE IMPLEMENTAREA TUTUROR METODELOR PROTOCOLULUI

- Delegat este o instanta a clasei **(mai exact, a celei care implementeaza functionalitatea delegatului)**, aflata in interiorul altei clase

```swift
protocol MamaDelegate {
    func cumparaMere(_ nrMere: Int) -> Int
}

class Mama {
    var delegate: MamaDelegate?

    func vreauMere() {
        let numarMereCumparate = delegate?.cumparaMere(10)
        print("Mama: Vad ca ai cumparat \(numarMereCumparate!) mere")
    }
}
```

```swift
class Copil: MamaDelegate {
    func cumparaMere(_ nrMere: Int) -> Int {
        print("Marius: M-a trimis mama sa cumpar \(nrMere) mere")
        return nrMere / 2
    }
}

var mariana = Mama()
var marius = Copil()

mariana.delegate = marius
mariana.vreauMere()
```

# iOS Course - 3

## View / UIView

**Un view / UIView reprezinta o zona dreptunghiulara care:**

- **Are definite coordonate in spatiu (pe ecran)**
- **Poate fi folosita pentru a desena in interiorul ei**
- **Poate contine Subviews ( alte UIView-uri )**
- **Poate "suporta" evenimente de touch ( Gestures )**

**Ierarhie:**

- **Un view poate contine UN SINGUR Superview**
- **Contine un array de Subviews (self.subviews)**
- **Ordinea in acest array conteaza, elementele de la final sunt afisate pe ecran "mai sus"**

**exista UIWindow: Un singur UIWindow in toata aplicatia**

# iOS Course - 3

## View / UIView

**Contine metode predefinite pentru UIView:**

- **func addSubview(_ view: UIView)**
- **func removeFromSuperview()**

**Dar si proprietati...**

- **var superview: UIView?**
- **var subviews: [UIView]**

**Exemple de view-uri:**

- **UIButton, UILabel, UITextField**
- **UITableView, UIScrollView**

# iOS Course - 3

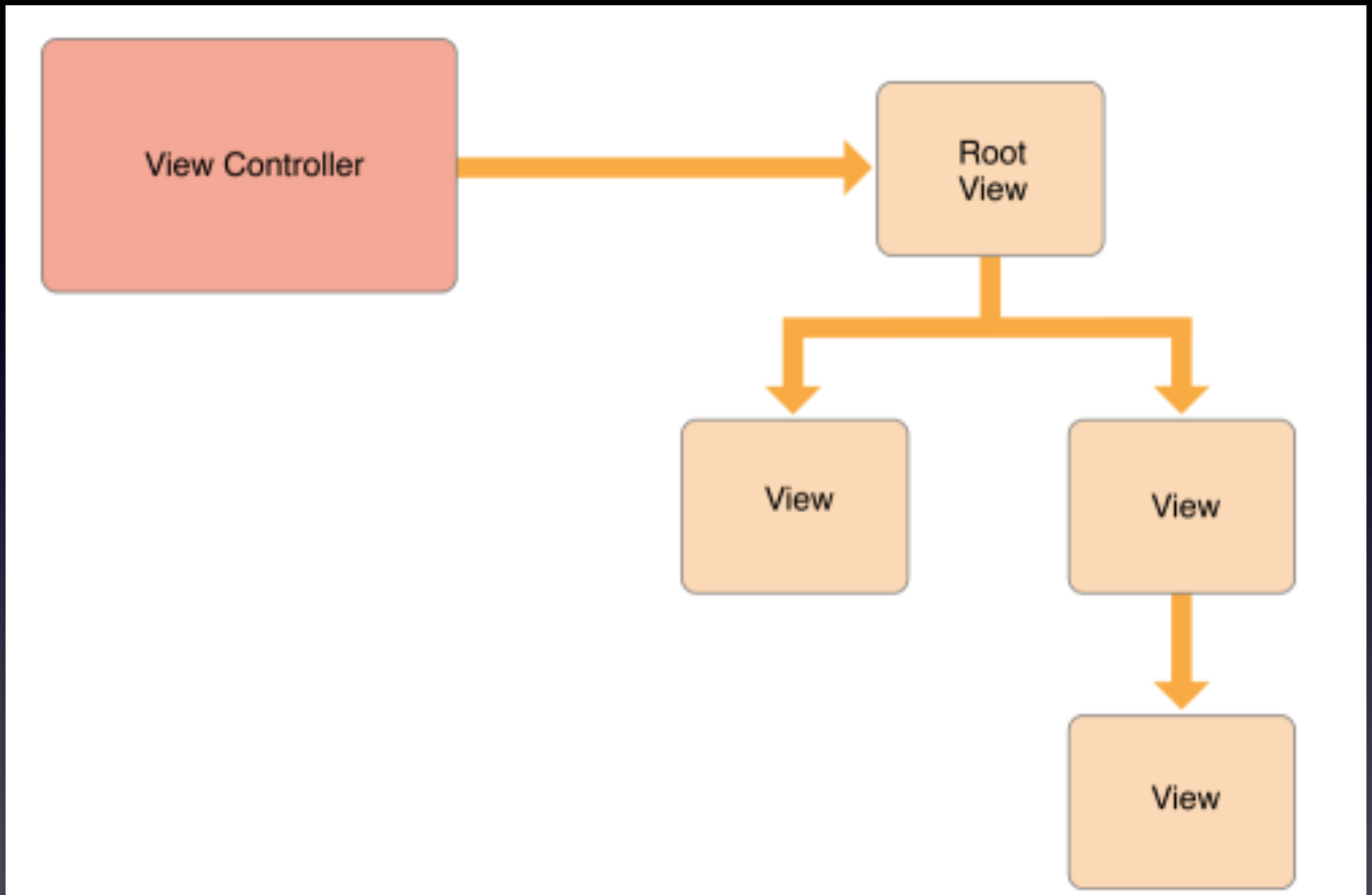
## ViewController / UIViewController

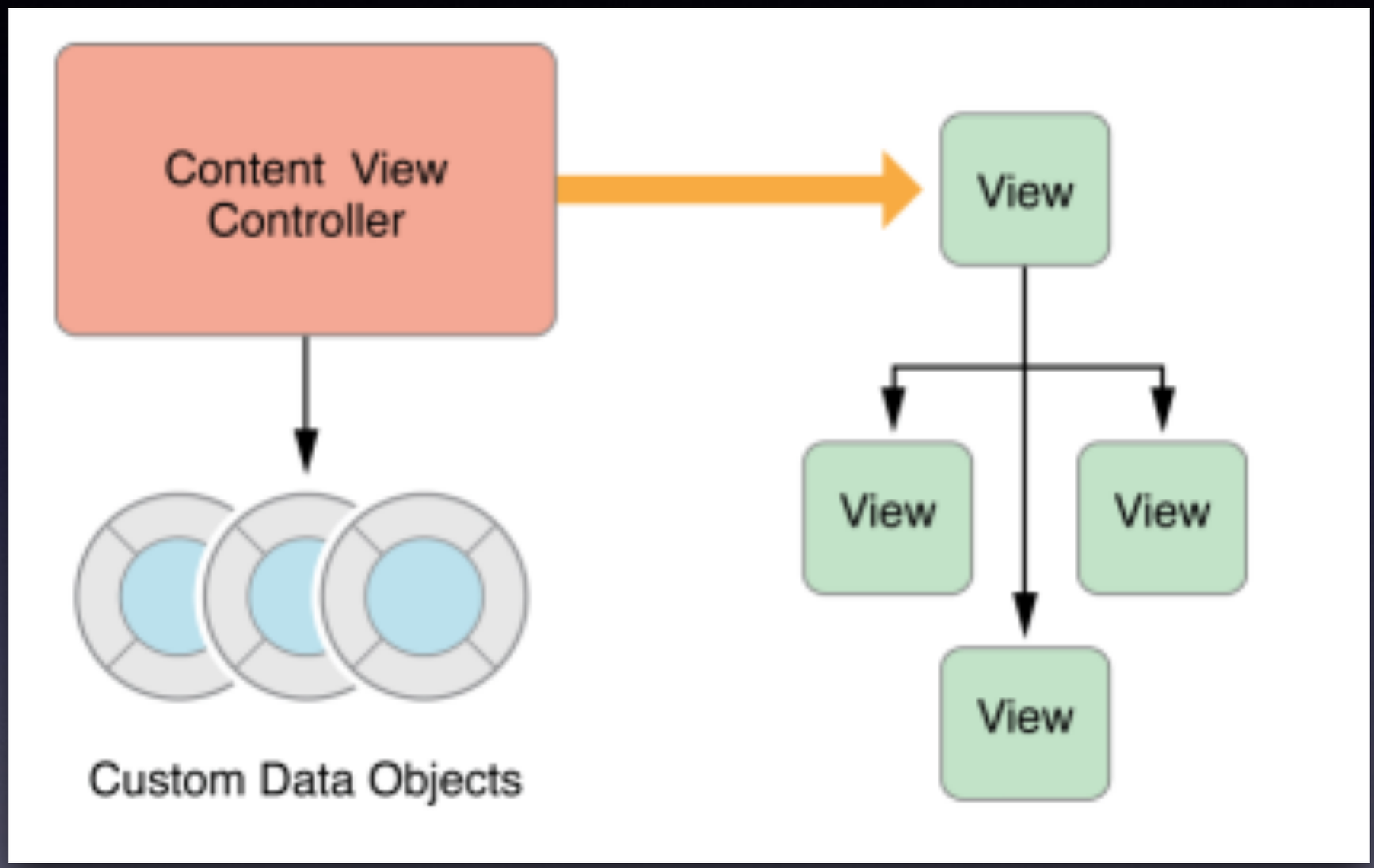
**Un ViewController este responsabil sa afiseze si sa "controleze" unul sau mai multe UIView-uri. Fara el nu s-ar intampla NIMIC in aplicatie**

- Are originea in partea din stanga sus
- Unitatea de masura a coordonatelor si dimensiunilor se masoara in puncte, nu pixeli

Metode si proprietati predefinite:

- var bounds: CGRect // (CoreGraphics Rectangle)
- var center: CGPoint // (CoreGraphics Point)
- var frame: CGRect // (CoreGraphics Rectangle)
- var contentScaleFactor: CGFloat // cati pixeli / punct
- Metodele si proprietatile derivate din UIView

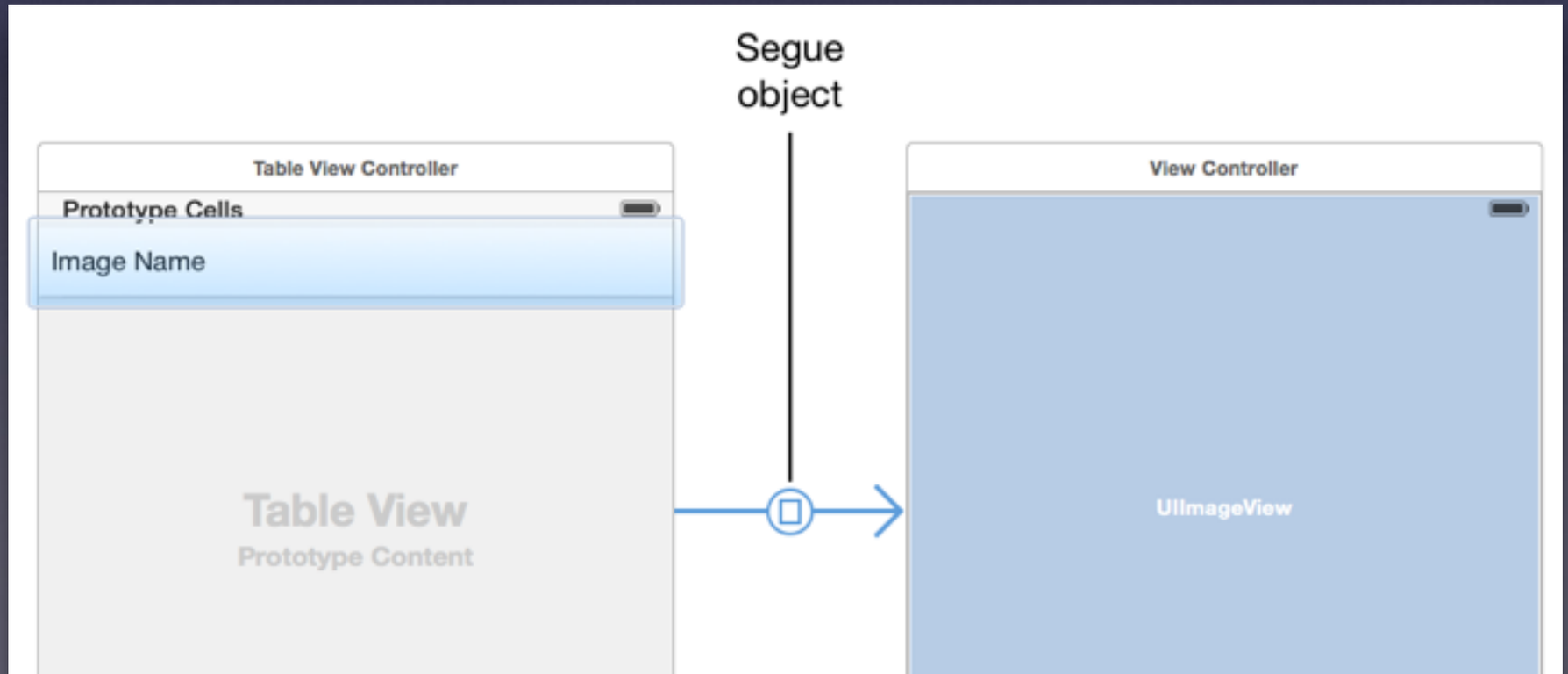

View Controller
Root View
View
View
View


Content View Controller
View
View
View
View
Custom Data Objects

# iOS Course - 3

Segues

- Un SEGUE defineste o tranzitie intre doua ViewControllere
- Un SEGUE este pus in functiune de apasarea unui buton, a unui rand intr-un tabel, deci de un gest (ex: tap)
- Punctul final al unui SEGUE este ViewController-ul pe care vrei sa il afisezi pe ecran



UnwindSegue.xcodeproj — MainStoryboard.storyboard

iPhone 6.1 Simulator

Scheme

Breakpoints

Xcode

No Issues

Editor

View

Organiz

UnwindSegue 〉 UnwindSegue 〉 MainStoryboard.storyboard 〉 MainStoryboard.storyboard (English) 〉 No Selection

ntroller

A View Controller

Add B

B View Controller

Add C

# iOS Course - 3

**NavigationController**

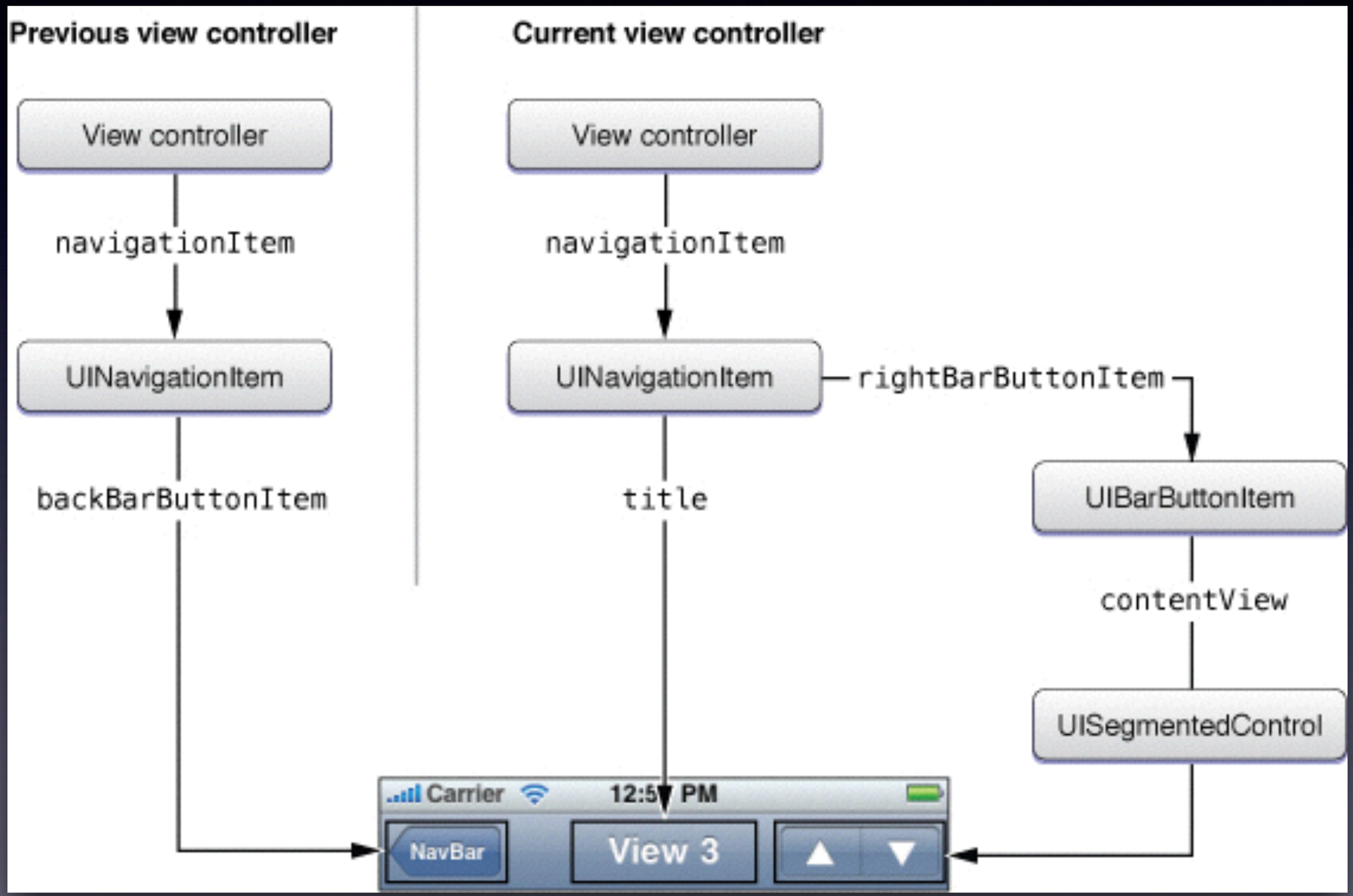
- Contine o stiva de unul sau mai multe ViewControllere (NavigationStack) si o bara in partea de sus (NavigationBar)

- Fiecare ViewController se ocupa de administrarea unui View iar NavigationController-ul se ocupa de administrarea fiecarui ViewController

# iOS Course - 3


Assembled views
Window
Navigation view
Edit
World Clock
Cupertino
Today
New York
Today, 3 hours ahead
Moscow
Tomorrow, 11 hours ahead
iPod
2:23 PM
World Clock
Stopwatch
Timer

# iOS Course - 3


Previous view controller
Current view controller
View controller
navigationItem
UINavigationItem
backBarButtonItem
View controller
navigationItem
UINavigationItem
rightBarButtonItem
title
UIBarButtonItem
contentView
UISegmentedControl
Carrier
12:5 PM
NavBar
View 3

# Demo

*Continuare aplicatie* ***NrPrim***

# Proiect Xcode:

*https://github.com/mariusjcb/iOS10_CS3*