UNIVERSITAS SCIENTIARUM SZEGEDIENSIS
UNIVERSITY OF SZEGED
Department of Software Engineering

Okosóra, Okostelefon és OkosTV - Apple Swift alapú alkalmazás fejlesztés

Dr. Bilicki Vilmos

Szoftverfejlesztés Tanszék

MVC

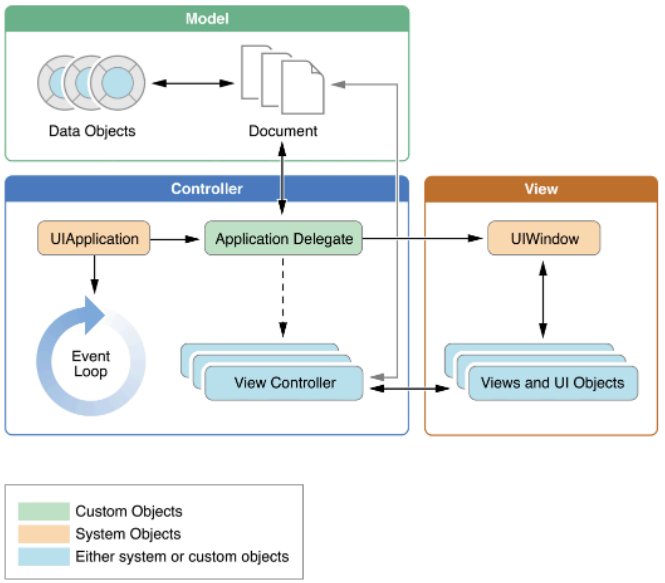
- MVC modell bemutatása
- Model, View, Controll egységek funkciója (data, logic, view)
- Az objektumok osztályozása (mely egységhez tartoznak)
- A MVC modellen belüli kommunikáció (Model-Controller, Controller-View)
- A kód újrafelhasználhatósága
- A források strukturálása

IOS alkalmazás

- Alkalmazás csomag (Xcode kezeli)
  - Futtatható alkalmazás (Alkalmazásnév.app)
  - Információ tulajdonság fájl (Info.plist)
    - Igények (WiFi,…)
    - Típus (Újságos stand,..)
  - Ikonon
    - Alkalmazás
    - Indító
    - Ad-hoc
  - Szorti fájlok (nib fájlok)
  - Beállítások fájl
  - A helyi nyelveket támogató lokalizált erőforrások könyvtárai (en.lproj, hu.lproj)

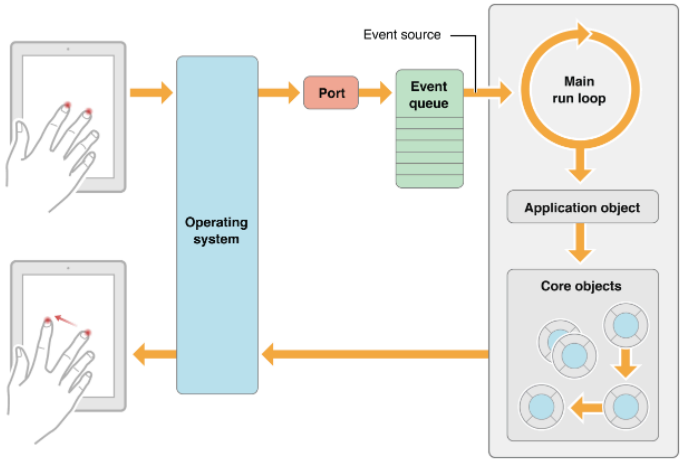
IOS alkalmazás

- Indításkor UIApplicationMain függvény inicializálja a fontos objektumokat és elindítja az alkalmazást (UI, Story board, init,…)
  - UIApplication objektum
    - App delegate
    - Dokumentum és adat modell objektumok
    - View controller objektumok
    - UIWindow objektum
    - View és kontroll objektumok



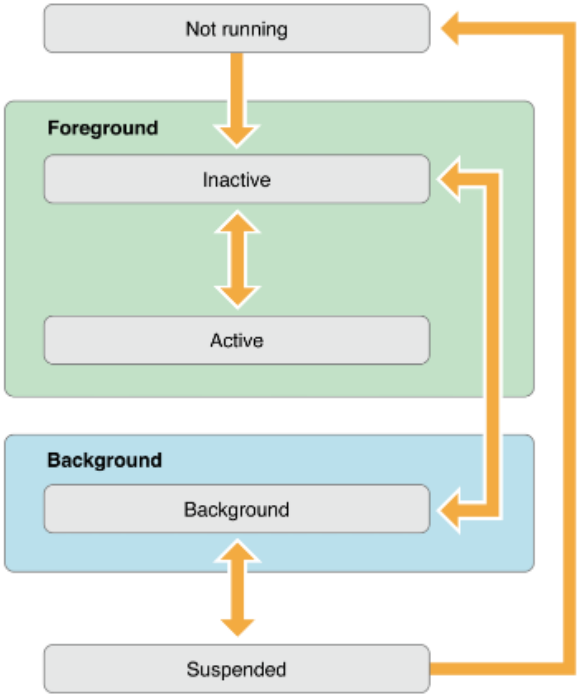
Fő ciklus

- Minden felhasználói esemény kezelője
  - Érintés (Responder object)
  - Rázás
  - Gyorsuáls, magnetométer
  - Helyszín
  - Újrarajzolás
  - ….
- UIApplication indítja el
- A fő szálban fut



Alkalmazás életciklus

- Az operációs rendszer az események hatására megváltoztatja a program állapotát
  - Nem futó
  - Inaktív – nincs view
  - Aktív – Eseményeket kezel
  - Háttér – exta időt kérhet az itt maradáshoz
  - Felfüggesztett – memóriában marad de nem fut, ha kevés a memória akkor kidobhatja

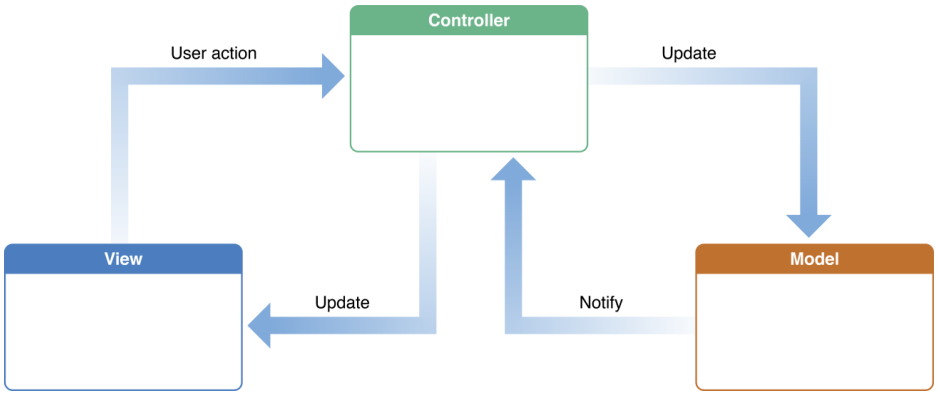


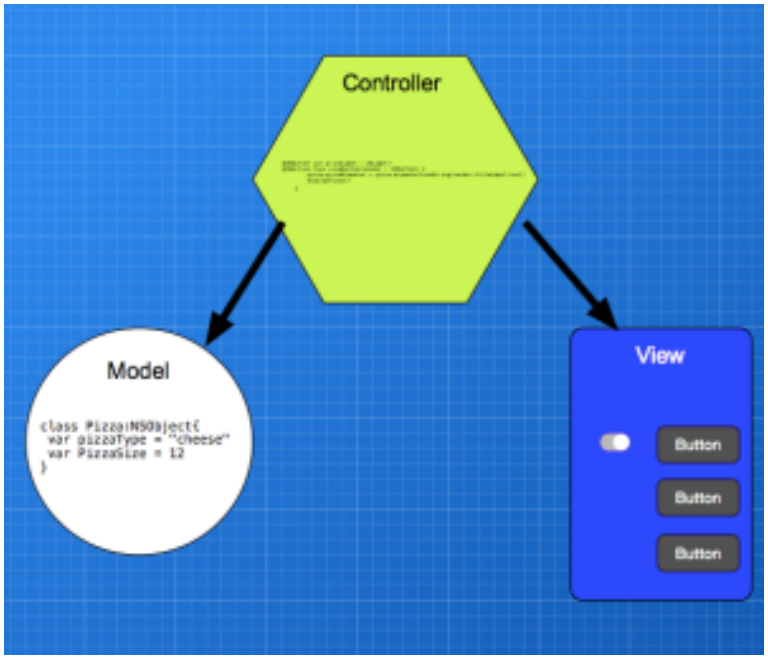
Delegate minden állapotátmenethez

- application:willFinishLaunchingWithOptions
- application:didFinishLaunchingWithOptions
- applicationDidBecomeActive
- applicationWillResignActive
- applicationDidEnterBackground
- applicationWillEnterForeground
- applicationWillTerminate

MVC

- A Model-View-Controller (MVC) tervezési minta három szerepkört különböztet meg
  - Model
  - View
  - Controller
- Nem csak a szerepköröket hanem a kommunikáció módját is megadja
- Absztrakt határokkal szeparáltak
- Időnként rétegnek is nevezzük a három entitást



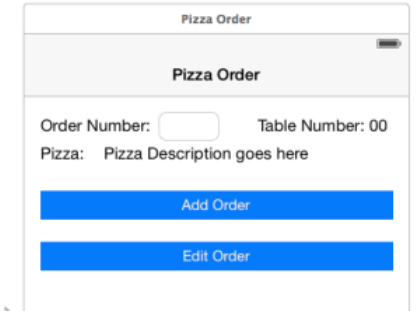


Model

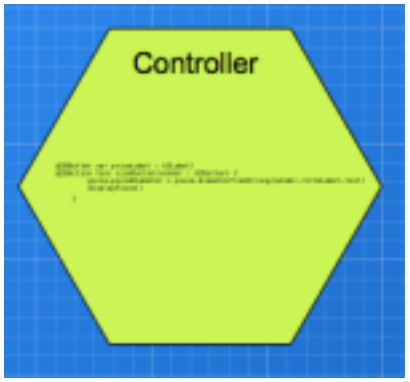
- Az adatot és a hozzá tartozó logikát kezeli. Kapcsolatban lehet más modell objektumokkal.
- Leggyakrabban a perzisztenciával foglalkozik.
- Adott terület szakmai tudását ábrázolja így azon a területen belül újrahszanosítható
- Nincs közvetlen kapcsolatban a View-val
- Kommunikáció: A felhasználói események amelyek a View-ben keletkeznek a Conroller segítségével jutnak ide. Amikor az adat változik akkor értesíti a vezérlőt.

View

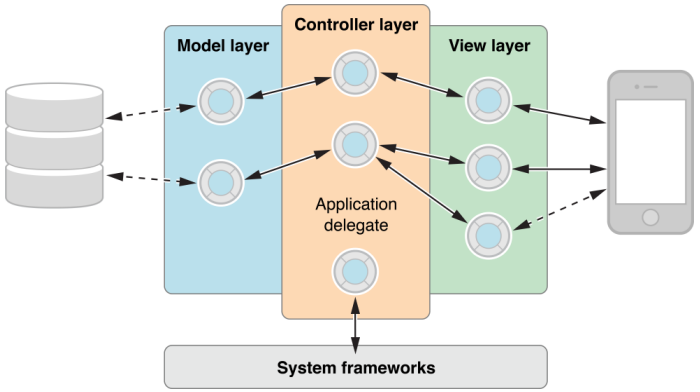
- A felhasználó által látott objektum. Tudja hogyan jelenítse meg magát és hogyan válaszoljon a felhasználónak.
- A View objektumok alkalmazásokon átívelő konzisztenciát biztosítanak. (UIKit, AppKit)
- **Kommunikáció**: Az adat változásokról a Controller-től hallanak. Az itt keletkező eseményeket a Controllernek küldik

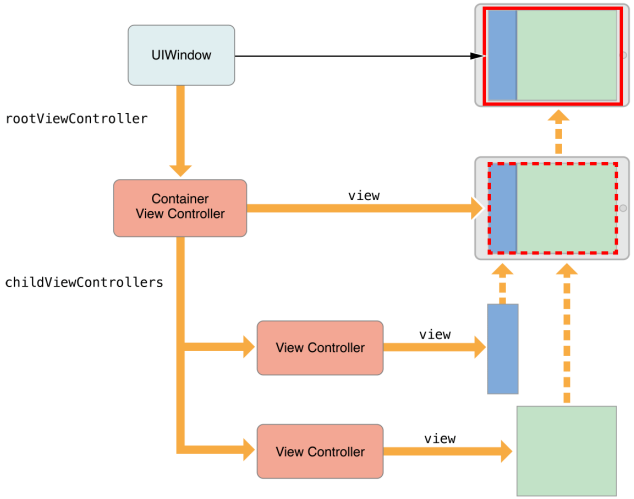


Controller



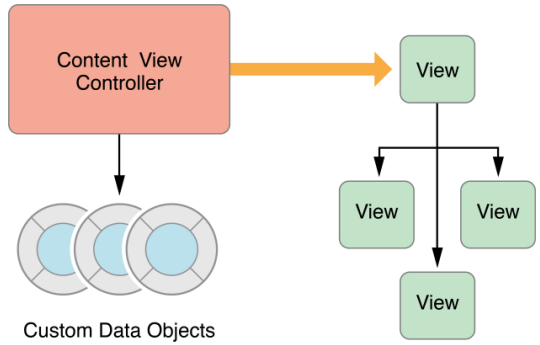
- Közvetítő szerepet lát el. Egy vagy több View és Model objektum között. Inicializálással és az alkalmazás életciklusának kezelésével is foglalkozik.
- **Kommunikáció**: A felhasználói eseményeket értelmezi és a model felé közvetíti. A model változásokat közvetíti a felhasználónak.

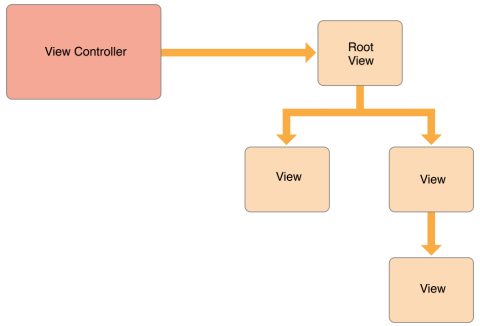


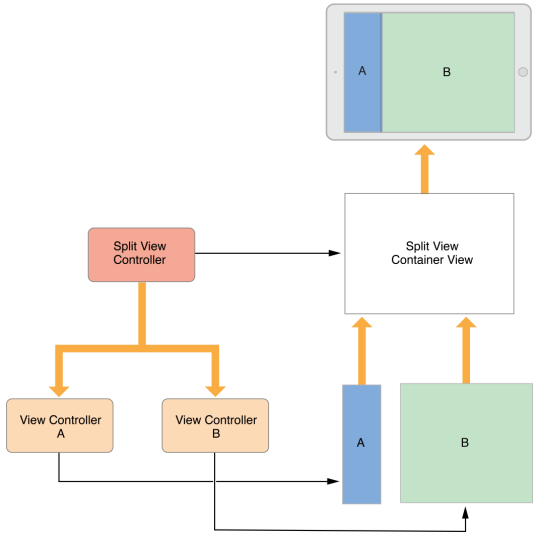


ViewController

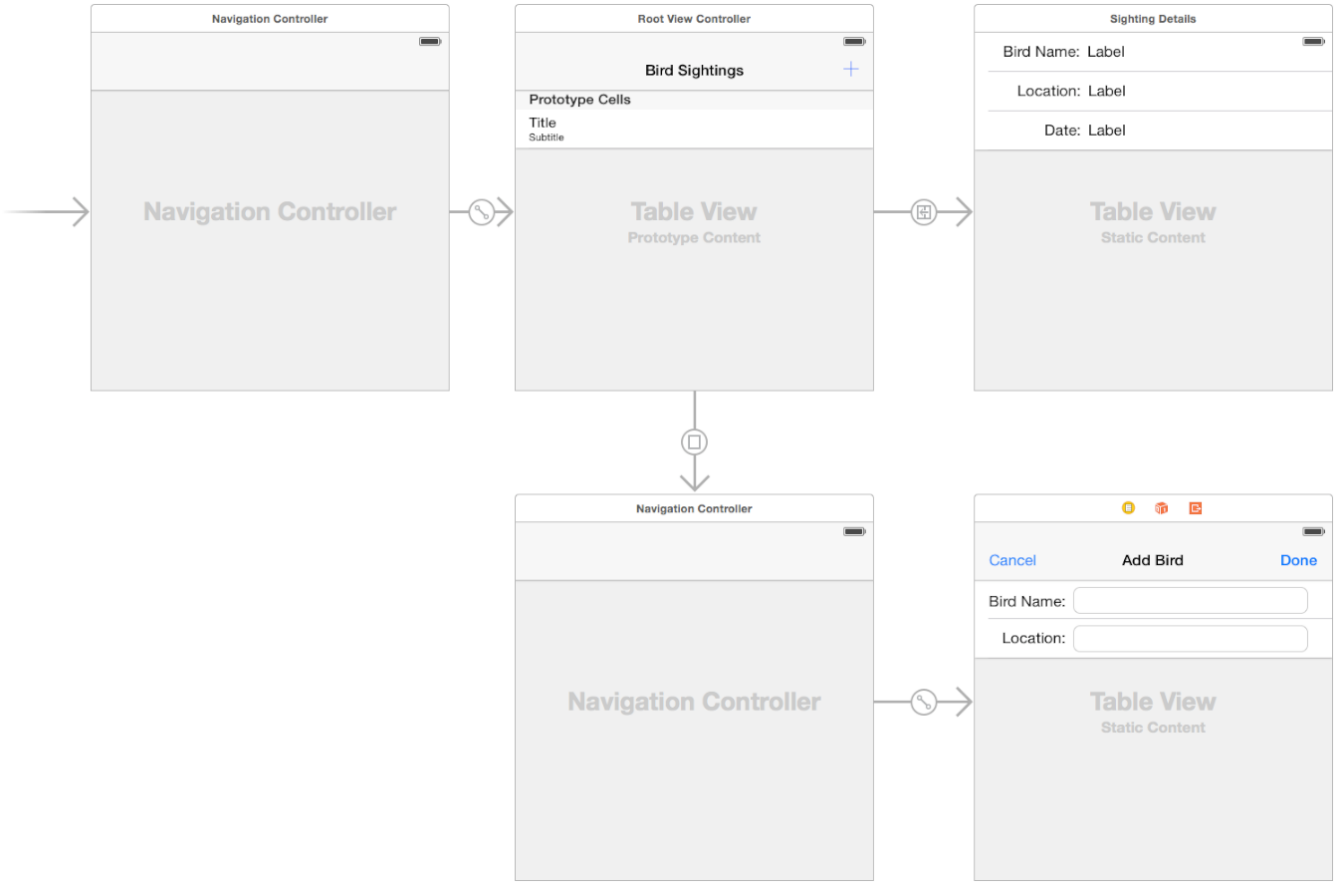
- Az alkalmazás belső struktúrájának alapja
- Típusai
  - Tartalom ViewController
    - Az alkalmazás egy adott részére fókuszál
  - Konténer ViewController
    - Több másik ViewControllerrel kommunikál







Storyboard

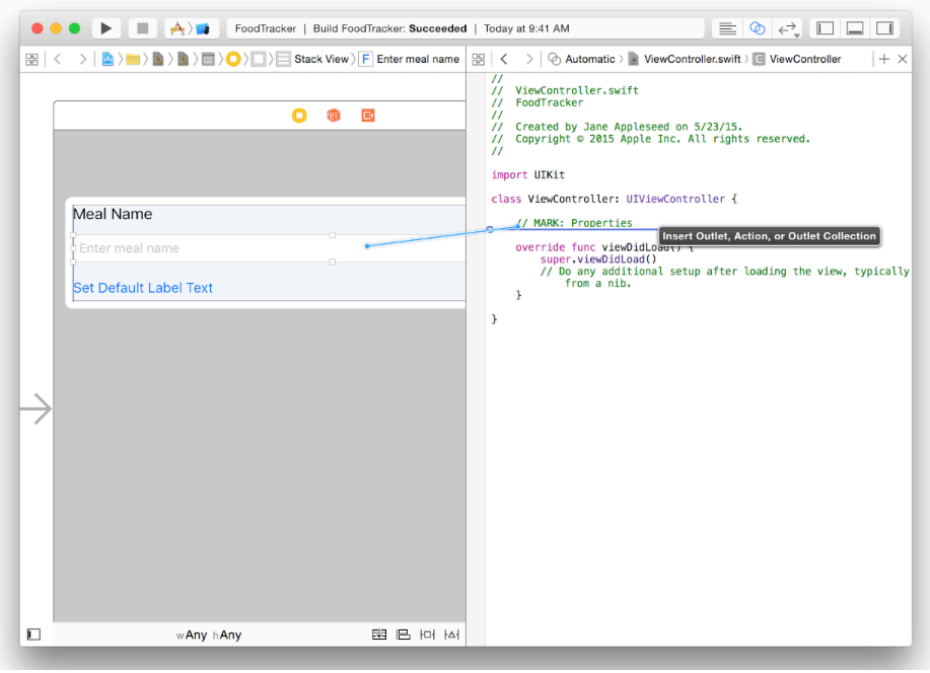

Navigation Controller
Navigation Controller
Root View Controller
Bird Sightings
Prototype Cells
Title
Subtitle
Table View
Prototype Content
Sighting Details
Bird Name: Label
Location: Label
Date: Label
Table View
Static Content
Navigation Controller
Navigation Controller
Cancel
Add Bird
Done
Bird Name:
Location:
Table View
Static Content

Kommunikációs módok

- Controller – Model
  - Hozzáfér
- Controller – View
  - Outlet
- View – Controller
  - Target action (ID action), View meghívja ezt
  - Delegate
    - Scroll view, zoom, …: should, will, did
    - View tulajdonság (protokoll)
  - Delegate - datasource
    - Data, at, count
- Model – Controller
  - Notification & KVO (új adat, adat változott)

Outlet

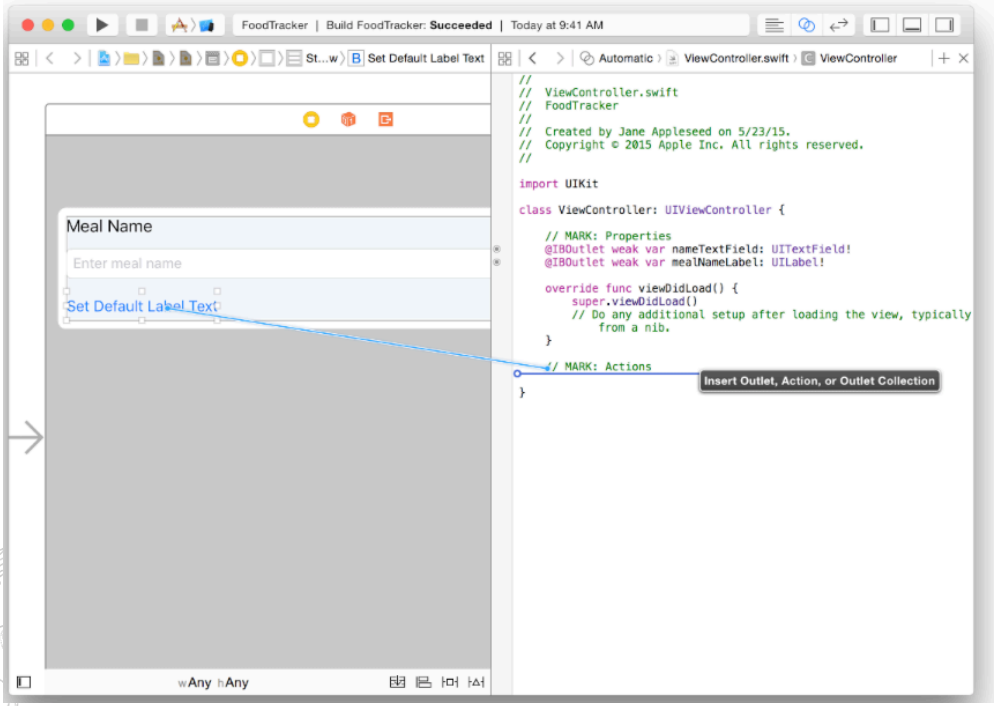
- Interfész (View) objektumokhoz férünk hozzá a Controller objektumból




```
@IBOutlet weak var nameTextField: UITextField!
```

Action

- Eseményhez kötött kód részlet




```
1  @IBAction func setDefaultLabelText(sender: UIButton) {
2  }
```

Delegate - UI

- Kisegítő objektum, adott funkció ide van kiszervezve
- Pl. szöveg mező állapotok kezelése
  - Protokoll: UITextFieldDelegate

```
class ViewController: UIViewController, UITextFieldDelegate {

 override func viewDidLoad() {
     super.viewDidLoad()

     // Handle the text field's user input through delegate callbacks.
     nameTextField.delegate = self
 }
 func textFieldShouldReturn(textField: UITextField) -> Bool
 func textFieldDidEndEditing(textField: UITextField)
```

Delegate - Datasource

- Az adott GUI elemet látja el adattal
  - Pl.: protokoll: UITableViewDataSource

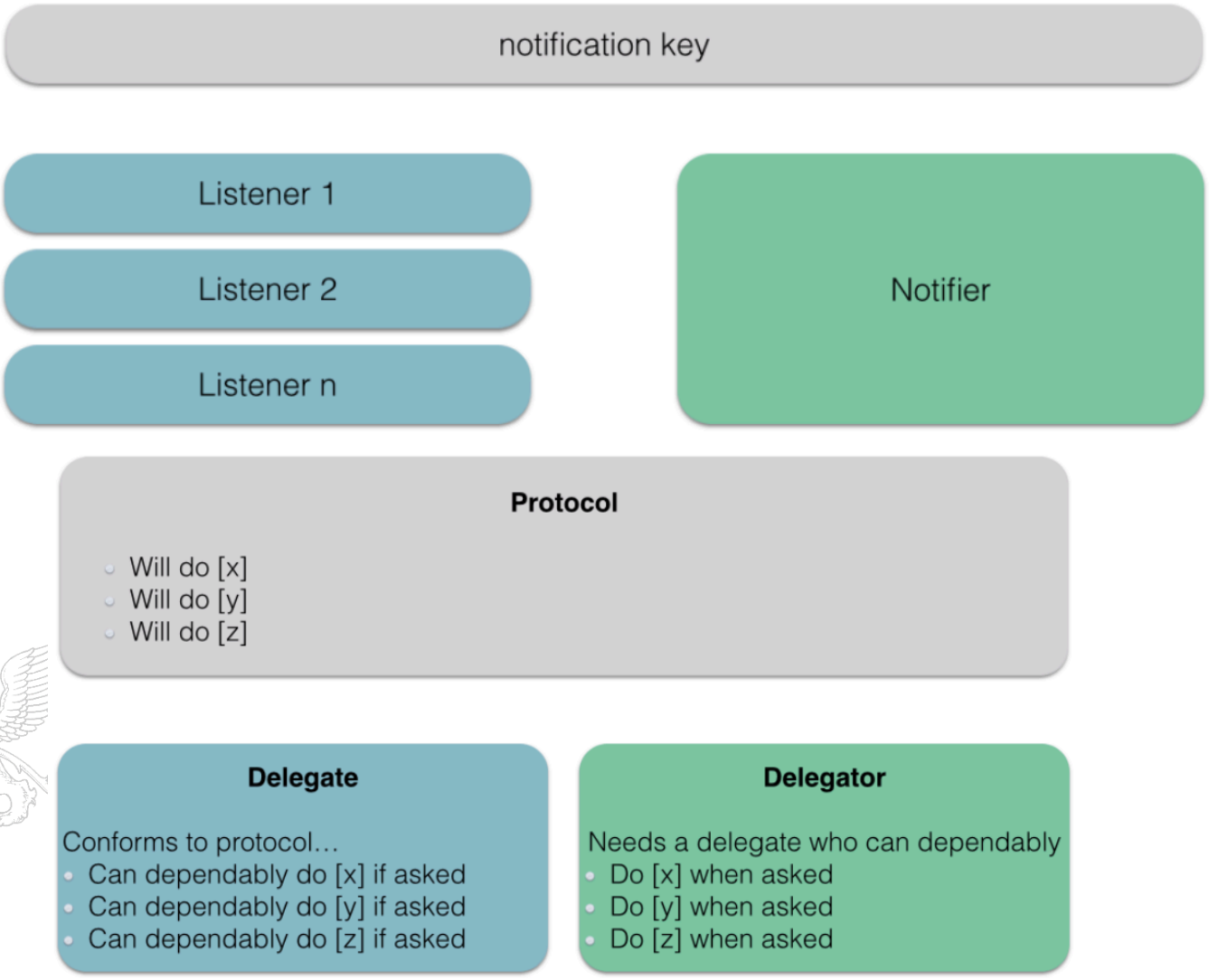
```
func numberOfSectionsInTableView(tableView: UITableView) -> Int
func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int
func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath:
  NSIndexPath) -> UITableViewCell
```

Notification

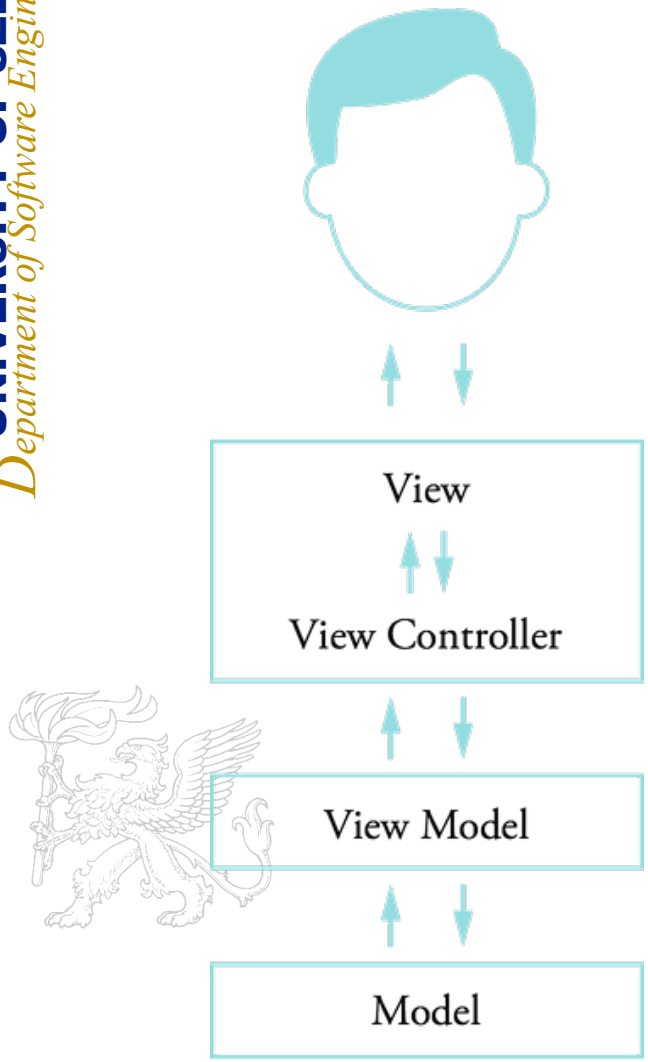
- NSNotificationCenter – üzenetközpont

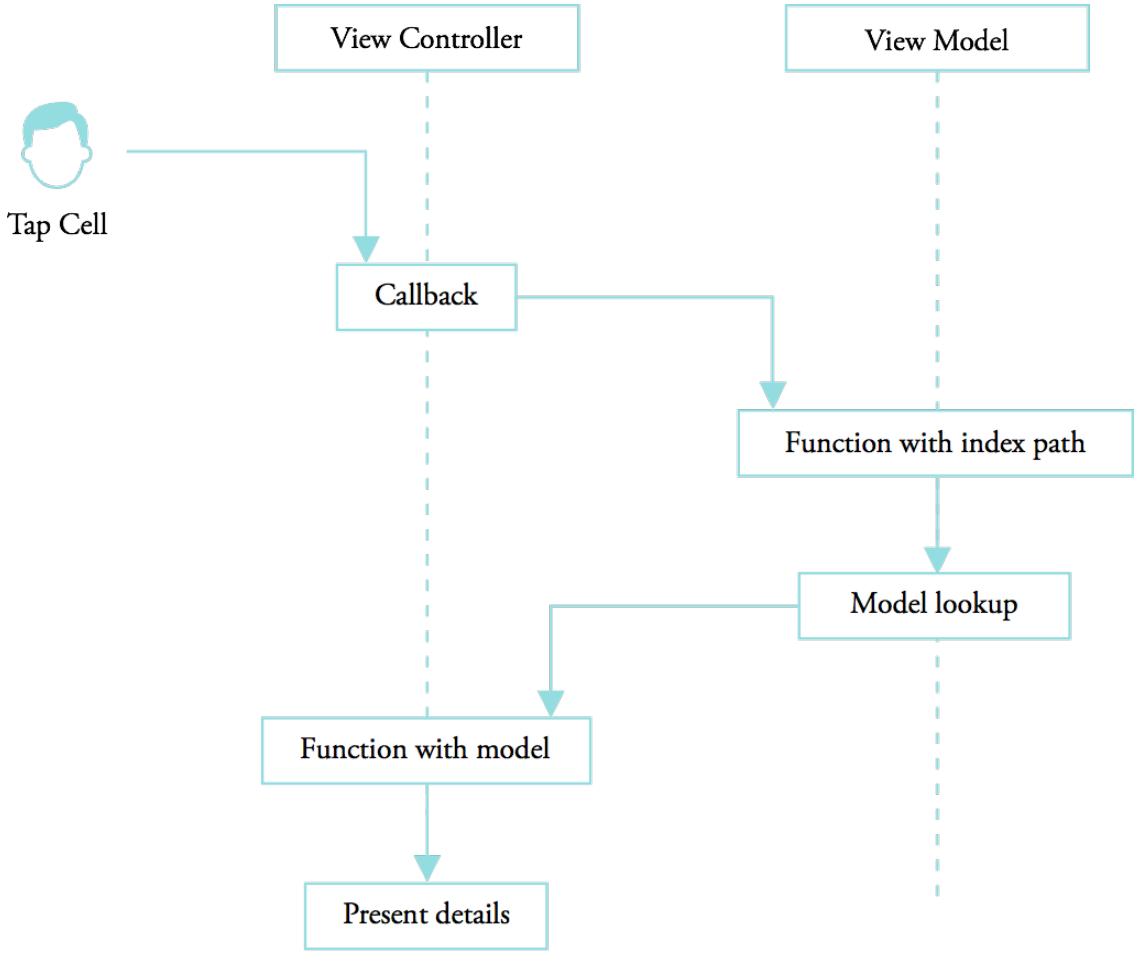
```swift
let mySpecialNotificationKey = "com.andrewcbancroft.specialNotificationKey"

class FirstViewController: UIViewController {
    @IBOutlet weak var sentNotificationLabel: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()
        NotificationCenter.default.addObserver(self, selector: #selector(FirstViewController.updateNotificationSentLabel), name: NSNotification.Name(rawValue: mySpecialNotificationKey), object: nil)
    }

    // 2. Post notification using "special notification key"
    @IBAction func notify() {
        NotificationCenter.default.post(name: Notification.Name(rawValue: mySpecialNotificationKey), object: self)
    }

    func updateNotificationSentLabel() {
        self.sentNotificationLabel.text = "Notification sent!"
    }
}
```

Delegate vs Notification



MVVM


View
View Controller
View Model
Model


View Controller
View Model
Tap Cell
Callback
Function with index path
Model lookup
Function with model
Present details

MVC

- MVC modell bemutatása
- Model, View, Controll egységek funkciója (data, logic, view)
- Az objektumok osztályozása (mely egységhez tartoznak)
- A MVC modellen belüli kommunikáció (Model-Controller, Controller-View)
- A kód újrafelhasználhatósága
- A források strukturálása

A következő előadás tartalma

- Cocoa Touch Class
- UIViewController
- Autolayout
- Storyboard belépési pont (Initial View Controller)
- Koordináta-rendszer, frame, view, bound
- View elemek használata
- Elemek csoportosítása (Stack View)- Outlet, Action- viewDidLoad()