GENERAL ASSEMBLY

MOBILE DEVELOPMENT

LESSON 09
DELEGATION REVIEW AND NOTIFICATIONS

Tedi Konda
VP, Engineering and Technology, RepEquity

LEARNING OBJECTIVES

LEARNING OBJECTIVES

- Recap: delegation pattern and table view controllers
- Notification pattern
- Singleton pattern
- Homework 3: Finalize and submit before class Wednesday

DESIGN PATTERNS

- A design pattern is a concept that revolves around having reusable code to solve common tasks in programming.

- There are about ~24 design patterns. We will learn about:
  - Delegation
  - Notifications
  - Singletons

DELEGATION

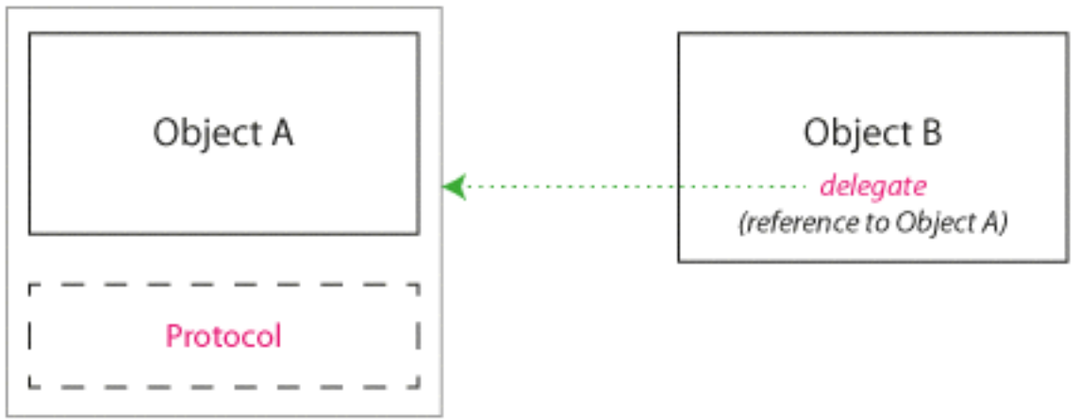
GETTING STARTED

DELEGATION

- In Real Life:
  - A person sent or authorized to represent others, in particular an elected representative sent to a conference.
- In Cocoa Touch:
  - Delegation is a simple and powerful pattern in which one object (the delegate) in a program acts on behalf of, or in coordination with, another object. The delegating object keeps a reference to the other object—the delegate—and at the appropriate time sends a message to it.
    - Official Source: Apple

DELEGATION EXAMPLE

- ObjectA is some object
- ObjectA also defines a protocol
- ObjectB implements ObjectA's protocol, and becomes a delegate of ObjectA
- ObjectA sends a message to the delegate (ObjectB) via the protocol methods



DELEGATION IN-CLASS EXERCISE

NEW: SINGLETON

GETTING STARTED

SINGLETON PATTERN

- A design pattern that restricts the initialization of a class to one object.
- That object is then accessible by any other class in your project.
- When is it useful?
    - Managing Preferences
    - Managing Themes/Stylization
    - Logging
    - Notifications
        - NSNotificationCenter.defaultCenter()

NEW: NOTIFICATIONS

GETTING STARTED

NOTIFICATIONS (OBSERVER PATTERN)

- Notifications allow for one**-to-many** communication.
- The NSNotificationCenter class is used for all communication
- An object uses NSNotificationCenter to post notifications
- An object uses NSNotificationCenter to subscribe to notifications
- Posted notifications are sent from the transmitter to the observers
- Notifications are identified using strings.

- Two important methods
  - addObserver
  - removeObserver
  - postNotification(_:)

IN-CLASS ASSIGNMENT

EXERCISE

KEY OBJECTIVE(S)

You will create a **to do app**. The app will have two main screens:

1. Main screen: this will be a TableViewController listing all the tasks
2. Add screen: this will be a UIViewController that will add a to-do task and pass it back to the main screen
3. Make the background of the text boxes in the second view controller blue when the keyboard comes up, and red when it goes down. Start with UIKeyboardWillShowNotification and NSNotificationCenter.
4. Bonus: add the ability to complete/delete tasks
5. Bonus 2: add the ability to view completed/deleted tasks in another TableViewController.

TIMING

45 min 1. Code with partner

5 min 2. Debrief

HOMEWORK

- Review UITableViewController using free tutorials on Ray Wenderlich's website:
  - http://www.raywenderlich.com/tag/uitableview
- Design Patterns
  - http://mobbook.generalassemb.ly/week04/intro.html
- NSNotificationCenter Tutorials
  - http://www.andrewcbancroft.com/2014/10/08/fundamentals-of-nsnotificationcenter-in-swift/
  - https://www.youtube.com/watch?v=yOouiCNIGE0

- Continue Week 3 Homework due Wednesday before class