iOS App Extensions

Hands-On Challenges

iOS App Extensions Hands-On Challenges

Challenge #3: Shared Settings

You're using Core Data to share data between your app and extension, which is great! But do you remember how to share simpler things like settings?

In this challenge, you'll save the user's favorite category from the app to the shared `NSUserDefaults`. Then, you'll load this from the extension and display the most recent post from that category specifically rather than the most recent post across the entire site.

Getting Started

There are three steps to get through this challenge:

1. Save the favorite category to the shared container from the app.

2. Load the favorite category from the shared container from the extension.

3. Fetch the most recent post by category from Core Data.

Let's get started!

Saving Data

First, you'll need to save the favorite category! The default is "No Favorite" which leads to the current behavior of showing the most recent post regardless of category.

Open **SettingsTableViewController.swift** and make sure the data is being saved to the shared container. Find this line of code:

```
let defaults = NSUserDefaults.standardUserDefaults()
```

And replace it with the following:

```
let defaults = NSUserDefaults(suiteName:
 "group.com.razeware.rwtracker")!
```

Note that you'll need to change the exact app group ID to match your own.

Loading Data

Open **TodayViewController.swift** and add the same property to the class to access the shared defaults:

```
let defaults = NSUserDefaults(suiteName:
 "group.com.razeware.rwtracker")!
```

Again, replace the ID here with your own app group ID.

Next, let's refactor the code so the fetching from Core Data is separate from updating the interface. Add the following method:

```
func updateLabelFromPost(post: NSManagedObject?) {
  if let title = post?.valueForKey("title") as? String {
    postTitle.text = title
  }

  if let timestamp = post?.valueForKey("timestamp") as? NSDate {
    postDate.text = dateFormatter.stringFromDate(timestamp)
  }
}
```

The code here is from the existing `updateLabels()` method. Given an `NSManagedObject` from Core Data, it will update the two labels in the interface.

Now you can change `updateLabels()` to call the Core Data fetch and hand the results to `updateLabelFromPost()`. Replace `updateLabels()` with the following implementation:

```
func updateLabels() {
  let favoriteCategories = [
    "No Favorite",
    "Announcements",
    "Podcast Episodes",
    "Tutorials",
    "Video Tutorials"
  ]

  let favoriteCategoryIndex =
    defaults.integerForKey("favoriteCategory")
  let favoriteCategory = favoriteCategories[favoriteCategoryIndex]

  var post: NSManagedObject?

  if favoriteCategoryIndex > 0 {
    post =
     coreDataStack.mostRecentPostForCategory(favoriteCategory)
  }

  if post == nil {
    post = coreDataStack.mostRecentPost()
  }

  updateLabelFromPost(post)
}
```

First up is an array to map a category index to a category name. The app stores integers 0-4 rather than category names, so you'll need this array to look up the name from the index.

If there is a favorite category set, then call `mostRecentPostForCategory()` to get that category's most recent post.

It's possible that the category doesn't have any posts so as a fallback, the method will call `mostRecentPost()` both if the user didn't select a favorite, or if there was nothing available in the favorite category.

That's it! Run the app and select a favorite category. While you're there, take note of the most recent post in the category. Then when you open notification center, make sure the extension now shows that post.