iOS DESIGN PATTERNS

HANDS-ON CHALLENGES

iOS Design Patterns

Joshua Greene

Table of Contents: Overview

Table of Contents: Extended

3 MVC-N - Challenge

By Joshua Greene

There's currently two view controllers that are nearly identical: **BusinessProductsViewController** and **HomeProductsViewController**.

"Another developer" on the project (ahem) justified this because the networking logic was different.

Since you're now familiar with MVC-N, you know networking code doesn't actually belong in the controllers at all. You're now ready to fix the duplicate controllers once and for all!

Challenge

Extract the commonalities from **BusinessProductsViewController** and **HomeProductsViewController** into a new class called **ProductsViewController**, which will show any type of **Product**.

Create a new injected property of type **Product.ProductType!** on this class, which will determine whether or not "business" or "home" products should be fetched and also the title to display.

Delete **BusinessProductsViewController.swift** and **HomeProductsViewController.swift**.

Update **CleaningServices.storyboard** to replace the previous references to **BusinessProductsViewController** and **HomeProductsViewController** to **ProductsViewController** instead.

Set segue **identifiers** for both the "Business" and "Home" segues.

Lastly, implement **prepare(for segue: UIStoryboardSegue, sender: Any?)** on **CleaningServicesTableViewController** to determine which **Product.ProductType** needs to be injected based on the **segue.identifier**.

Challenge Solution

Create a new Swift file called **ProductsViewController.swift** within **Cleaning Services \ Controllers**. Replace its contents with the following:

```
import UIKit

public class ProductsViewController: UIViewController {

  // MARK: - Injections
  internal var networkClient = NetworkClient.shared

  internal var productType: Product.ProductType! {
    didSet {
      title = productType.title
    }
  }

  // MARK: - Instance Properties
  internal var products: [Product] = []

  // MARK: - Outlets
  @IBOutlet internal var collectionView: UICollectionView! {
    didSet {
      let refreshControl = UIRefreshControl()
      refreshControl.addTarget(self,
                               action: #selector(loadProducts),
                               for: .valueChanged)
      collectionView.refreshControl = refreshControl
      let layout = collectionView.collectionViewLayout
        as! UICollectionViewFlowLayout
      collectionView.collectionViewLayout =
        CollectionViewCenterFlowLayout(layout: layout)
    }
  }

  internal func loadProducts() {
    collectionView.refreshControl?.beginRefreshing()
    networkClient.getProducts(
      forType: productType,
      success: { [weak self] products in
        guard let strongSelf = self else { return }
        strongSelf.products = products
        strongSelf.collectionView.reloadData()
        strongSelf.collectionView.refreshControl?.endRefreshing()


      }, failure: { [weak self] error in
        print("Product download failed: \(error)")
        guard let strongSelf = self else { return }
        strongSelf.collectionView.refreshControl?.endRefreshing()
    })
  }

  // MARK: - View Lifecycle
  public override func viewDidLoad() {
    super.viewDidLoad()
    loadProducts()
  }

  public override func viewWillDisappear(_ animated: Bool) {
    super.viewWillDisappear(animated)

    guard let selectedItem = collectionView.indexPathsForSelectedItems
      else { return }
    selectedItem.forEach { collectionView.deselectItem(
      at: $0, animated: false)
    }
  }

  // MARK: - Segue
  public override func prepare(for segue: UIStoryboardSegue,
                               sender: Any?) {
    guard let viewController = segue.destination
      as? ProductDetailsViewController else { return }
    let indexPath = collectionView.indexPathsForSelectedItems!.first!
    let product = products[indexPath.row]
    viewController.product = product
  }
}

// MARK: - UICollectionViewDataSource
extension ProductsViewController: UICollectionViewDataSource {

  public func collectionView(
    _ collectionView: UICollectionView,
    numberOfItemsInSection section: Int) -> Int {
    return products.count
  }

  public func collectionView(
    _ collectionView: UICollectionView,
    cellForItemAt indexPath: IndexPath)
    -> UICollectionViewCell {

    let cellIdentifier = "ProductCell"

    let product = products[indexPath.row]
    let cell = collectionView.dequeueReusableCell(
      withReuseIdentifier: cellIdentifier,
      for: indexPath) as! ProductCollectionViewCell
    cell.label.text = product.title
    cell.imageView.rw_setImage(url: product.imageURL)
    return cell
  }
}
```

Delete **BusinessProductsViewController.swift** and **HomeProductsViewController.swift**.

Change the class identifier for the "Business" and "Home" scenes to **ProductsViewController** on the **CleaningServices.storyboard**.

Set **business** for the **segue identifier** to the "Business" scene, and set **home** for the **segue identifier** to the "Home" scene.

Add the following to **CleaningServicesTableViewController** right before **// MARK: - UITableViewDelegate**:

```
private struct SegueIdentifiers {
  static let business = "business"
  static let home = "home"
}

public override func prepare(for segue: UIStoryboardSegue, sender: Any?)
{
  guard let viewController = segue.destination as? ProductsViewController
else { return }

  if segue.identifier == SegueIdentifiers.business {
    viewController.productType = .business

  } else if segue.identifier == SegueIdentifiers.home {
    viewController.productType = .home

  } else {
    fatalError("Unknown ProductsViewController segue identifier: " +
      "\(String(describing: segue.identifier))")
  }
}
```

Build and run; navigate to both the "Business" and "Home" screens; and verify they work as expected.

Über challenge

There's one last piece of duplication: the "Business" and "Home" **storyboard scenes**. Fix it! ;]

Hint: you'll need to make *two* segues (the **business** and **home** segues) to the *same* storyboard scene and delete the other.

If you get stuck, check out the completed challenge in the resources for this video.