iOS App Extensions

Hands-On Challenges

iOS App Extensions Hands-On Challenges

Challenge #1: Backwards Compatibility

As you improve your extension and release updates to the App Store, you'll want to make sure that edits made with previous versions of the extension are handled properly.

In this short challenge, you'll add support for version 1.0 of the Hipsterize extension from the first video tutorial on photo extensions.

Getting Started

First, you'll need a photo to work with! Take your best selfie, and then load the final project from the first photo extensions video tutorial. Fire up the extension, hipsterize the photo, and tap the **Done** button to close the extension, and then again to close the photo editor.

Now, load the project from the resources for this video tutorial. That will be the code for version 2.0 of the extension.

Run the extension and then select the same photo again. When the extension interface comes up, you'll see some strange behavior:

Double Hipsterization! The extension is incorrectly processing an already-processed image with disastrous results.

There will be two parts to fix this problem:

1. Update the extension so it will accept adjustment data from version 1.0 of the extension.
2. Process the photo differently depending on whether it's coming from version 1.0 or version 2.0.

Accepting Adjustment Data

First, you'll need to signal to iOS that you can handle version 1.0 adjustment data.

In that version of the extension, the adjustment data's `NSData` is just filled with some filler data that doesn't mean anything. Even if you won't actually *use* the data, you still need to let iOS know that you can accept incoming version 1.0 data.

Open **PhotoEditingViewController.swift** and find `canHandleAdjustmentData()`. Update the implementation to the following:

```
func canHandleAdjustmentData(adjustmentData: PHAdjustmentData?)
 -> Bool {
  if let adjustmentData = adjustmentData {
    return adjustmentData.formatIdentifier ==
       "com.razeware.hipsterize" &&
     (adjustmentData.formatVersion == "1.0" ||
      adjustmentData.formatVersion == "2.0") &&
     adjustmentData.data != nil
   }
  return false
}
```

The method will now return true if:

1. The `formatIdentifier` matches.

2. The `formatVersion` is either 1.0 or 2.0.

3. The adjustment data's NSData property has some data.

With this change, iOS will now pass along the *unprocessed* image rather than the already hipsterized image. You're one step closer to restoring hipster balance!

Processing the Photo

Now it's just a matter of adding another code path to handle incoming version 1.0 photos and data.

The photo processing logic is in `viewDidAppear()`. In that method, find the code near the top that deals with adjustment data:

```
if let data = input?.adjustmentData?.data {
  if let unarchivedElements =
   NSKeyedUnarchiver.unarchiveObjectWithData(data) as?
   [HipsterElement] {
    self.hipsterElements = unarchivedElements
  }
}
```

Again, you won't actually need to unarchive the data for version 1.0 since there's nothing useful there. Instead, you just want to run the auto-hipsterize algorithm as if it were a fresh photo.

Change the above block of code to the following:

```
if let data = input?.adjustmentData?.data {
  if input!.adjustmentData.formatVersion == "1.0" {
```

```
      // do nothing
    } else if input!.adjustmentData.formatVersion == "2.0" {
      if let unarchivedElements =
       NSKeyedUnarchiver.unarchiveObjectWithData(data) as?
       [HipsterElement] {
        self.hipsterElements = unarchivedElements
      }
    }
  }
```

In your own extensions, you might have very different logic between versions of the extension. In this case, you can just do nothing for version 1.0 since that means the `hipsterElements` array will stay empty. That will trigger the auto-hipsterize algorithm to run.

Now, run the extension again and re-hipsterize your photo. You should get just a single set of hat + glasses. And remember: the photo is now in version 2.0 of the extension! Feel free to upsize that hat; you've earned it!