State Preservation & Restoration on iOS

Sebastian Hagedorn, iosphere GmbH

Presented at CocoaHeads Dresden, 09.07.2014
Updated/translated to English, 08.08.2015

Table of Contents

Motivation

- ☞ iOS terminates background apps at (rather) random times
- ☞ State restoration: Trying to hide from the user that the app had been terminated
- ☞ Goal: Re-opening an app that was terminated by the system should equal the experience of switching to an app that awakes from the background
  - ☞ **Note:** Full restoration is not always possible or even desirable, e.g., temporary alerts

Implementation

- ☞ Roadmap:
  - ☞ Opting into state restoration
  - ☞ Restoring view controllers (and their hierarchy)
  - ☞ Restoring fine-grained state

Implementation

- ☞ Sample project:

github.com/shagedorn/StateRestorationDemo

  - ☞ Start tag:
    `NO_STATE_RESTORATION`

- ☞ **Note:** Swift has evolved quickly, so only `master` will build with the latest tools, not the intermediary tags.

Implementation: Opting In

- ☞ Opt-In in the app delegate:

```
func application(_:shouldSaveApplicationState:) -> Bool
```

- ☞ A restoration archive file named `data.data` with global app information (version, timestamp, interface idiom,...) will be created

```
func application(_:shouldRestoreApplicationState:) -> Bool
```

- ☞ Consider returning `false` in `shouldRestoreApplicationState` after app updates

Implementation: Opting In

- ☞ iOS takes a snapshot before the app goes into the background
  - ☞ Snapshot replaces `Default.png` (or launch Nibs/Storyboards) if at least one view controller supports state restoration
- ☞ Tag: `STATE_RESTORATION_OPT_IN`

Implementation: View Controller

- ☞ Per-controller opt-in: set the `restorationIdentifier` property
  - ☞ Either in code, or Storyboard/Nib
- ☞ **Done: State preservation**
  - ☞ Per view controller, a path of restoration identifiers will be saved
- ☞ **`TODO`: State restoration**
- ☞ Tag: `RESTORATION_IDENTIFIERS`

Implementation: View Controller

☞ View controller restoration (options):

① Controller sets `restorationClass` to a class that implements `UIViewControllerRestoration`

② App delegate instantiates controllers on demand

③ Implicit: Controller has already been created before state restoration begins

④ Implicit: Preserved controller had been instantiated from a storyboard

Implementation: View Controller

☞ **Note:** State restoration is attempted in the order defined above, but in the following slides, the various mechanisms have been re-ordered by increasing complexity, and to follow the sample app.

Implementation: View Controller

- ☞ Restoration option:
  - ☞ **Implicit: Controller has already been created before state restoration began**
  - ☞ Call order:
    1. `application(_:willFinishLaunchingWithOptions:)`
    2. State restoration
    3. `application(_:didFinishLaunchingWithOptions:)`

Implementation: View Controller

- ☞ Demo app's state at `BASIC_STATE_RESTORATION` tag:
  - ☞ Initial controllers (tabs one and two) are restored implicitly, including tab selection (we get that for free from `UITabBarController`)
  - ☞ View controllers that are created later (city detail) are not restored
  - ☞ State of the controllers (e.g., slider value) is not restored

Implementation: View Controller

- ☞ Restoration option:
  - ☞ **Set a `restorationClass`**
  - ☞ E.g., set the controller as its own restoration class
  - ☞ Implement `UIViewControllerRestoration` protocol, and create the controller on demand:

```
class func viewControllerWithRestorationIdentifierPath(_:coder:)
-> UIViewController?
```

Implementation: View Controller

- ☞ Restoration option: Set a `restorationClass` (continued)
  - ☞ Opt-out/abort: return `nil` from `viewControllerWithRestorationIdentifierPath(_:coder:)`
  - ☞ `coder` parameter allows informed decision about whether or not to restore
  - ☞ Tag: `RESTORATION_CLASS`

Implementation: View Controller

- ☞ Restoration option:
  - ☞ **App delegate creates controllers on demand**, similar to `UIViewControllerRestoration` protocol at controller level

```
func application(_:viewControllerWithRestorationIdentifierPath:
coder:) -> UIViewController?
```

- ☞ `return nil`: In contrast to `UIViewControllerRestoration`, this does not cancel implicit restoration
- ☞ Not used in the sample app

Implementation: State

☞ Restoring fine-grained state: `UIStateRestoring` protocol

```
override func encodeRestorableStateWithCoder(coder: NSCoder)  {
    super.encodeRestorableStateWithCoder(coder)
    // Make sure not to access explicitly-unwrapped outlets directly:
    // May be called when the controller (e.g., in a tab) has never loaded its view
    coder.encodeFloat(slider?.value ?? 0.5, forKey: "encodingKeySlider")
}
override func decodeRestorableStateWithCoder(coder: NSCoder)  {
    super.decodeRestorableStateWithCoder(coder)
    slider.value = coder.decodeFloatForKey("encodingKeySlider")
}
```

☞ Encode/decode primitive values or other objects that implement state restoration

Implementation: State

- ☞ `decodeRestorableStateWithCoder(_)` is called after `viewDidLoad()`
- ☞ Restoration (of the view controller itself) cannot be cancelled at this point
- ☞ Any object can implement `UIStateRestoring`, but views and controllers do automatically, and are registered by the system (see `UIStateRestoring` documentation for details)
- ☞ Tag: `STATE_ENCODING`

Debugging

- ☞ State preservation is triggered when app enters the background
- ☞ Restoration archive is deleted when…
  - ☞ Restoration fails or the app crashes
  - ☞ User force-quits the app in the app switcher
    - ☞ Can be disabled with a developer mode profile
    - ☞ Simulator: Quitting the app from Xcode (when in the background) does not delete the archive

Debugging

- ☞ State information is stored in `data.data` files
  - ☞ Binary PLISTs: Use `plutil` or `restorationArchiveTool` to read & debug the archives
  - ☞ https://developer.apple.com/downloads/ → Log in and search for "restoration"
    - ☞ `restorationArchiveTool` and documentation
    - ☞ `Debug Logging` iOS profile
    - ☞ `Developer Mode` iOS profile

Debugging

- ☞ Tip: Save a Finder search for `data.data` within the simulator directory in your Finder sidebar
- ☞ Usage: `restorationArchiveTool path/to/data.data`

```
                UIApplicationStateRestorationSystemVersion........8.0
                UIApplicationStateRestorationTimestamp............2014-07-06 16:05:18 +0000
                UIApplicationStateRestorationUserInterfaceIdiom...iPhone


[View Controller] TabController (Class: UITabBarController)
------------------------------------------------------------

                kTabBarSelectedViewControllerKey...Object Identifier Proxy: TabController/:[1]:/NavigationController
                kViewControllerViewWasLoadedKey....Yes


[View Controller] TabController/:[0]:/_TtC20StateRestorationDemo19FirstViewController (Class: _TtC20StateRestorationDemo19FirstViewController)
------------------------------------------------------------------------------------------------------------------------------------------

                encodingKeySliderValue............0.8926057
                kViewControllerViewWasLoadedKey...Yes
```

Debugging

- ☞ Debug Logging profile: very verbose, including time profiling
- ☞ "`Warning: Unable to create restoration in progress marker file`"
  - ☞ `data.data` archive could not be found, i.e., state preservation failed

```
2014-07-07 22:24:50.412 StateRestorationDemo[1321:60b] void _restoreState(UIApplication
*, NSData *, NSObject<UIApplicationDelegate> *, NSURL *, NSString *,
UIStateRestorationRestoreStateBeginHandler): State restoration archive was saved with
major version 2, minor version 1. Current major version 2, current minor version 1.
2014-07-07 22:24:50.417 StateRestorationDemo[1321:60b] void _restoreState(UIApplication
*, NSData *, NSObject<UIApplicationDelegate> *, NSURL *, NSString *,
UIStateRestorationRestoreStateBeginHandler): Root restoration identifier paths are (
    TabController,
    "TabController/:[0]:/_TtC20StateRestorationDemo19FirstViewController",
    "TabController/:[1]:/NavigationController",
    "TabController/:[1]:/NavigationController/:[0]:/
_TtC20StateRestorationDemo20SecondViewController",
    "TabController/:[1]:/NavigationController/:[1]:/
_TtC20StateRestorationDemo18CityViewController"
)
2014-07-07 22:24:50.420 StateRestorationDemo[1321:60b] void _restoreState(UIApplication
*, NSData *, NSObject<UIApplicationDelegate> *, NSURL *, NSString *,
UIStateRestorationRestoreStateBeginHandler): Restoration Class map: {
    "TabController/:[1]:/NavigationController/:[1]:/
_TtC20StateRestorationDemo18CityViewController" =
"_TtC20StateRestorationDemo18CityViewController";
}
2014-07-07 22:24:50.422 StateRestorationDemo[1321:60b] void _restoreState(UIApplication
*, NSData *, NSObject<UIApplicationDelegate> *, NSURL *, NSString *,
UIStateRestorationRestoreStateBeginHandler): Object in root set for index [0] for
identifier path TabController: <UITabBarController: 0x1455cb50>
2014-07-07 22:24:50.425 StateRestorationDemo[1321:60b] void _restoreState(UIApplication
*, NSData *, NSObject<UIApplicationDelegate> *, NSURL *, NSString *,
```

Debugging

- ☞ `"Can't find Child View Controller at index 1 with identifier TabController/:[1]:/ NavigationController/:[1]:/ _TtC20StateRestorationDemo18CityViewController, truncating child array"`
  - ☞ Controller that was stored in the archive could not be restored

GOTCHAs/Tips

- ☞ "Setting" the `restorationIdentifier` by overwriting the getter instead of setting the variable property does not work
  - ☞ Enables state preservation: `data.data` is fine
  - ☞ Restoration fails
    - ☞ My guess: The object is registered for state restoration in `didSet`

GOTCHAs/Tips

- ☞ Not all subviews are included in the snapshot image, e.g., alerts
  - ☞ Strive for consistency between the snapshot and what you restore
  - ☞ Don't restore temporary errors that (hopefully) have become stale in the meantime, e.g., connection errors
  - ☞ Ignore snapshot:

```
UIApplication.sharedApplication().ignoreSnapshotOnNextApplicationLaunch()
```

Tips

- ☞ The main window does not have to be preserved explicitly, but doing so adds extra information, such as, size classes, to the archive

  - ☞ Tag: `SIZE_CLASSES`

Sources & Links

- ☞ Sample project source code and comments
- ☞ Docs: Preserving Your App’s Visual Appearance Across Launches
  - ☞ Detailed, but not 100% up-to-date
  - ☞ General availability since iOS 6, improvements in iOS 7
- ☞ WWDC 2013 Session 222: "What's New in State Restoration"
  - ☞ Detailed and up-to-date