You must have a Mac running OS X Lion 10.7.3 or a device running iOS 5 to utilize the iCloud Storage API for Documents.
iCloud

Application Design Changes

Determine if iCloud is enabled
• Incorporate file presenters
• Explicitly move files to iCloud
• Be prepared to handle version conflicts
• Make use of searches to locate files in iCloud
• Be prepared to provide user feedback
• Using Core Data for live databases
• Common document formats for OS X and iOS
iCloud

Readying Your Project

• Enable Entitlements
• Provisioning Portal
  • Registering App ID
  • Generate Provisioning Profile
• Update Project (Code Sign)
iCloud & iOS Development

iCloud (functionality) doesn’t exist in the iOS simulator!

You need to debug on the device
iCloud
2 Types of Storage

Key / Value Pairs

NSUbiquitousKeyValueStore

Containers

NS/UIDocument (typically)

Property Lists

Files & Directories
iCloud

File Coordinators and Presenters

- File Coordinators *(NSFileCoordinator)*
  - Within single process
  - Synchronization / Locking
- File Presenters *(NSFilePresenter)*
  - Notifications
  - Keeps internal data structure integrity
iCloud
User Interface Updates

When a user-generated document must be downloaded before it can be used.
(getResourceValue:forKey:error:)

- When there is a version conflict that the user must resolve. (NSFileVersion)

- When you want to give the user the option to enable or disable iCloud usage entirely for your app.
iCloud

Further Considerations

• Incorporating Search into Your Infrastructure (*NSMetadataQuery*)
• Handling Cases in Which iCloud Is Unavailable (*URLForUbiquityContainerIdentifier:*)
• Improving the Document Presentation Interface
• Supporting the Dynamic Naming of Documents
• Supporting the Key-Value Store (*NSUbiquitousKeyValueStore*)
• Using Core Data with iCloud
• Learning More About File Coordinators
iCloud

Further Considerations

• Incorporating Search into Your Infrastructure (NSMetadataQuery)
• Handling Cases in Which iCloud Is Unavailable (URLForUbiquityContainerIdentifier:)
• Improving the Document Presentation Interface
• Supporting the Key-Value Store (NSUbiquitousKeyValueStore)

• **Using Core Data with iCloud**

• Learning More About File Coordinators
iCloud & Core Data

- Two primary application choices
  1. Type of application
     - "Library" style (E.G. Music and Photos)
     - Document based (E.G. Keynote and Numbers)
  2. Type of persistent store
     - SQL(ite) database
     - Atomic (binary) or custom `NSAtomicStore`
iCloud & Core Data

SQLite Store

• Efficient
• Record by record
• Stored by transaction log
• Must inform iCloud where to store the logs
• Must delete the log when store is removed
• The SQLite store itself must not be synced by iCloud
iCloud & Core Data

Atomic Store

• Not as efficient
• Notified when the store has changed
• You accept or reject changes wholesale
• Simpler to implement, more work to manage conflicts
iCloud & Core Data

“Library” Types of Applications

- **Single**: Core Data stack and persistent store with coordinator
- **SQLite** *(recommended)*
  - You provide name and place for the store itself
  - Store is either a local application sandbox or in a folder with the suffix: ".nosync"
- **Atomic**
  - Store goes in a “ubiquity” (iCloud sync’d) container
  - You provide suitable *NSFilePresenter* methods
iCloud & Core Data

Document-based Applications

- **NSManagedObject** or **UIManagedDocument**
  - Aggregation and synchronization is handled for you
  - You subclass this for your application
  - Atomic stores are recommended
  - SQLite stores have some limitations (see “references” at end)
    - ...and a bit more complex:

```swift
NSPersistentStoreDidImportUbiquitousContentChangesNotification
```
iCloud

References

- iCloud for Developers
- Design Considerations for iCloud Apps
- iOS Provisioning Portal
- iCloud Storage (iOS) iCloud Storage (OS X)
- Using Core Data with iCloud Release Notes
iCloud
(optional) Labs

- **Tutorial:** *Your Third iOS App: iCloud*
- **Tutorial (K/V):** *PrefsInCloud (iOS)*

**Notes**

- The API is the same for OS X, so the iOS tutorials are just as helpful
- You will need to submit a request for the App provisioning profile
- The "*Second iOS App*" will also teach you about storyboards!