iCloud

Daten und Dokumente in der Wolke

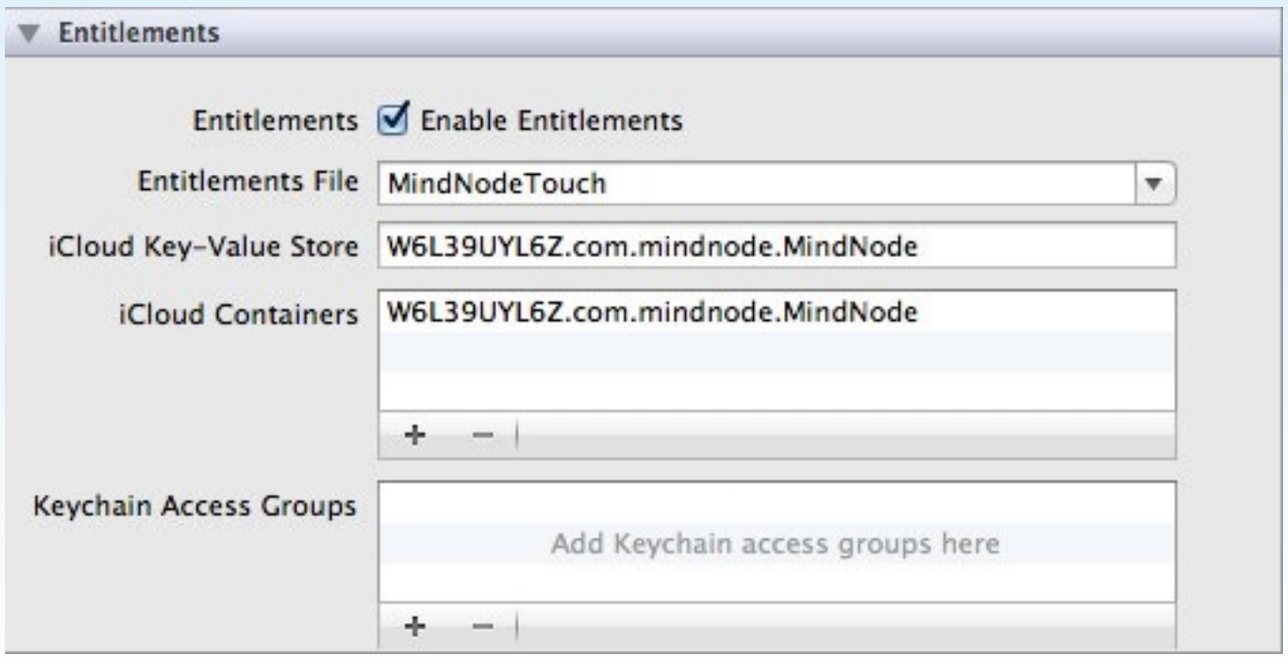
Markus Müller
IdeasOnCanvas GmbH

Agenda

- Key Value Store
- Documents in the Cloud
- iCloud auf dem Mac

Provisioning

- Provisioning Portal
- Entitlements
  - Key Value Store
    `com.apple.developer.ubiquity-kvstore-identifier`
  - Documents in the Cloud
    `com.apple.developer.ubiquity-container-identifiers`



Key Value Store

Key Value Store

- ✓ Einfach zu implementieren
- ✗ Nur “PropertyList” Objekte
- ✗ Maximal 64 KB an Daten
- ✗ Simples Konfliktmanagement

Initialisieren

```
NSUbiquitousKeyValueStore* store = [NSUbiquitousKeyValueStore defaultStore];
```

```
BOOL success = [store synchronize];
```

Initialisieren

```
NSUbiquitousKeyValueStore* store = [NSUbiquitousKeyValueStore defaultStore];

[[NSNotificationCenter defaultCenter] addObserver: self
      selector: @selector(updateKVStoreItems:)
      name: NSUbiquitousKeyValueStoreDidChangeExternallyNotification
      object: store];

BOOL success = [store synchronize];
```

Wert Setzen

```
[store setBool:flag forKey:@”myKey”];
```

Wert Lesen

```
BOOL flag = [store boolForKey:@”myKey”];
```

Documents in the Cloud

Documents in the Cloud

- ✓ Beliebige Dokumente
- ✓ Kein Limit bei der Dateigröße
- ✓ Flexibles Konfliktmanagement
- ✗ Aufwendig zu implementieren
- ✗ Asynchron

APIs

- NSFileManager
- NSMetadataQuery
- UIDocument/NSDocument
- NSFileVersion
- NSFileCoordinator
- NSFilePresenter

iCloud aktivieren

```
dispatch_queue_t q_default;
q_default = dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0);

dispatch_async(q_default, ^{

    NSFileManager *fm = [[NSFileManager alloc] init];
    if (![fm URLForUbiquityContainerIdentifier:nil]) {
       NSLog(@"iCloud not available");
    }

});
```

Dokumente finden (1)

```
NSMetadataQuery *query = [[NSMetadataQuery alloc] init];

NSArray *scopes = [NSArray
    arrayWithObject:NSMetadataQueryUbiquitousDocumentsScope];
[query setSearchScopes: scopes];

[query setPredicate:
     [NSPredicate predicateWithFormat:@"%K like '*'",
     NSMetadataItemFSNameKey]];
```

Dokumente finden (2)

```
NSNotificationCenter *nc = [NSNotificationCenter defaultCenter];

[nc addObserver:self
    selector:@selector(metadataQueryDidFinishGatheringNotifiction:)
    name:NSMetadataQueryDidFinishGatheringNotification
    object:query];

[nc addObserver:self
    selector:@selector(metadataQueryDidUpdateNotifiction:)
    name:NSMetadataQueryDidUpdateNotification
    object:query];

[query startQuery];
```

NSMetadataItem

- Document URL
- Metadaten
- Download Status
- Upload Status
- Konflikte

Downloading

```
NSFileManager *fm = [[NSFileManager alloc] init];

[fm startDownloadingUbiquitousItemAtURL:self.fileURL error:NULL];
```

Neues Dokument anlegen

- Zuerst lokal erstellen
- Dann auf iCloud laden

Dokumente auf iCloud laden

```
NSURL *ubiquitousDocumentsURL = [[self class]
ubiquitousDocumentsURL];

NSURL *destinationURL = [ubiquitousDocumentsURL
    URLByAppendingPathComponent:[sourceURL lastPathComponent]
    isDirectory:NO];

dispatch_async(q_default, ^(void) {
    NSFileManager *fileManager = [[NSFileManager alloc] init];
    NSError *error = nil;

    BOOL success = [fileManager setUbiquitous:YES
        itemAtURL:sourceURL
        destinationURL:destinationURL
        error:&error];

};
```

Dokumente von iCloud entfernen

```
BOOL success = [fileManager setUbiquitous:NO
    itemAtURL:sourceURL
    destinationURL:destinationURL
    error:&error];
```

UIDocument

- Asynchrones Laden und Speichern
- Automatisches Speichern
- iCloud Integration

Konflikte

- UIDocumentStateChangedNotification
- NSMetadataUbiquitousItemHasUnresolved ConflictsKey
- NSURLUbiquitousItemHasUnresolvedConfl ictsKey

Strategien

- Automatisch mergen
- Neuestes Dokument
- Benutzer fragen
- UIManagedDocument

NSFileVersion

- Aktuelle Version
- Versionen die im Konflikt stehen
- Versionen verwerfen

Koordinierter Dateizugriff

- iCloud Prozess aktualisiert laufend Dateien
- Saveless user model

Beispiel Umbenennen

```
NSFileCoordinator *fileCoordinator;
fileCoordinator = [[NSFileCoordinator alloc] initWithFilePresenter:nil];

[fileCoordinator coordinateWritingItemAtURL: sourceURL
     options: NSFileCoordinatorWritingForMoving
     writingItemAtURL: destinationURL
     options: NSFileCoordinatorWritingForReplacing
     error: &writeError
     byAccessor: ^(NSURL *newURL1, NSURL *newURL2) {

          NSFileManager* fileManager = [[NSFileManager alloc] init];
          [fileManager moveItemAtURL:sourceURL toURL:destinationURL error:NULL];

}];
```

Meine Implementierung

https://gist.github.com/1438664

MNDocumentReference

- Metadaten des Dokuments
- iCloud Status des Dokuments
- Implementiert NSFilePresenter

MNDocumentController

- Verwaltet Set von MNDocumentReferences
- Erzeugt neue Dokumente
- Dupliziert Dokumente
- iCloud Verwaltung

Hinweise

- Kein iCloud Support im iPhone Simulator
- Dokumente können sich jederzeit ändern

iCloud

Mac OS X

Thomas Zöchling

File Management

- iCloud nicht applikationsweit aktiv
- “Move to/from iCloud” Menü Eintrag
- API gleich (NSFileManager)

```
[fm setUbiquitous:YES/NO itemAtURL:source
destinationURL:destination error:&error]
```

Dokument Architektur

- NSDocument
  - implementiert NSFilePresenter
  - verwendet NSFileCoordinator
- "Document-Based App Programming Guide"

Dokument Architektur

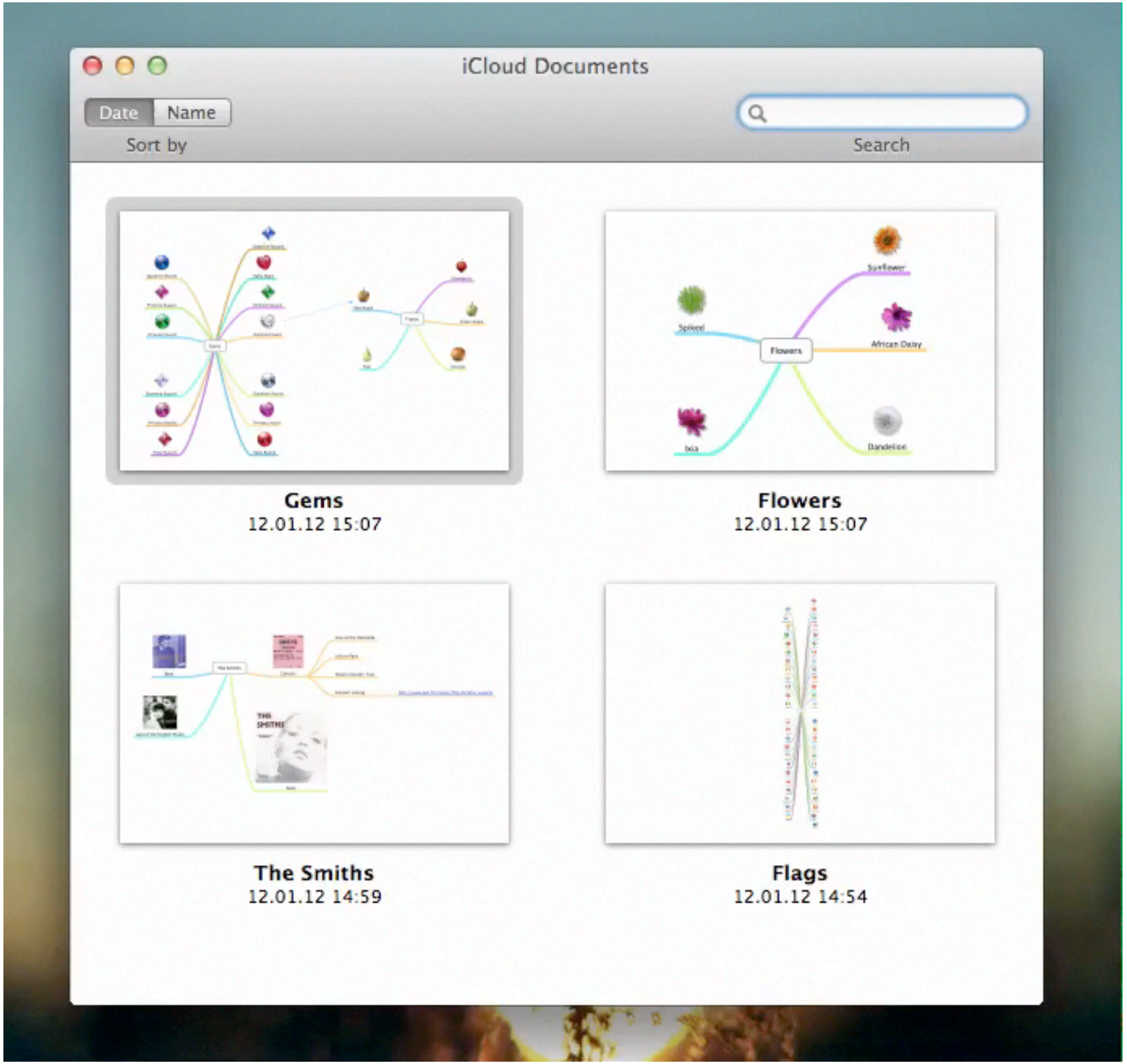
- 10.7 NSDocument Features implementieren

```
+ (BOOL)autosavesInPlace {return YES;}
```

  - Autosaving
  - Document Revisions
- Grundlage für Versions & iCloud
- Conflict Resolution via Versions UI

iCloud UI - Mac

- noch keine iCloud fähige App von Apple
- keine Standardkomponenten
- keine Integration in den Finder
- 3rd Party Apps verwenden “Recent Items”


iCloud Documents
Date
Name
Sort by
Search
Gems
12.01.12 15:07
Flowers
12.01.12 15:07
The Smiths
12.01.12 14:59
Flags
12.01.12 14:54

iCloud Abfragen

- NSMetadataQuery
- Shared instance:

```
@property(retain) NSSet* queryResult;
```

- NSSet (`minusSet:`, `intersectSet:`, ...)
- NSMetadataItem → in ein KVO Objekt
- ermöglicht KV Observing & Bindings

NSCollectionView

- hat seit 10.5 viel dazugelernt
- Kann mittels NSArrayController über Bindings gesteuert werden:
  contentArray→queryResult.arrangedObjects
- Lokales Sortieren & Filtern
- filterPredicate des Array Controllers binden

Tipps

- 10.7 NSDocument Features implementieren
- iCloud Document Sharing (NSFileManager)

```
[fm
URLForPublishingUbiquitousItemAtURL:destinationURL
expirationDate:&expirationDate
error:&error];
```

- Nicht versuchen Sandboxing & iCloud gleichzeitig zu implementieren

Ressourcen

- WWDC Sessions
  - 501 - iCloud Storage Overview
  - 116 - Storing Documents in iCloud Using iOS 5
  - 107 - AutoSave and Versions in Mac OS X 10.7 Lion
- Blogs & Dokumentation
  - Michael Jurewitz on iCloud Storage

    http://oleb.net/blog/2011/11/ios5-tech-talk-michael-jurewitz-on-icloud-storage/
  - iOS Developer Library - About Document-Based Applications in iOS