

Moving Your iOS App from Parse to CloudKit

Genady Okrain

Sugar So Studio

Co-Founder

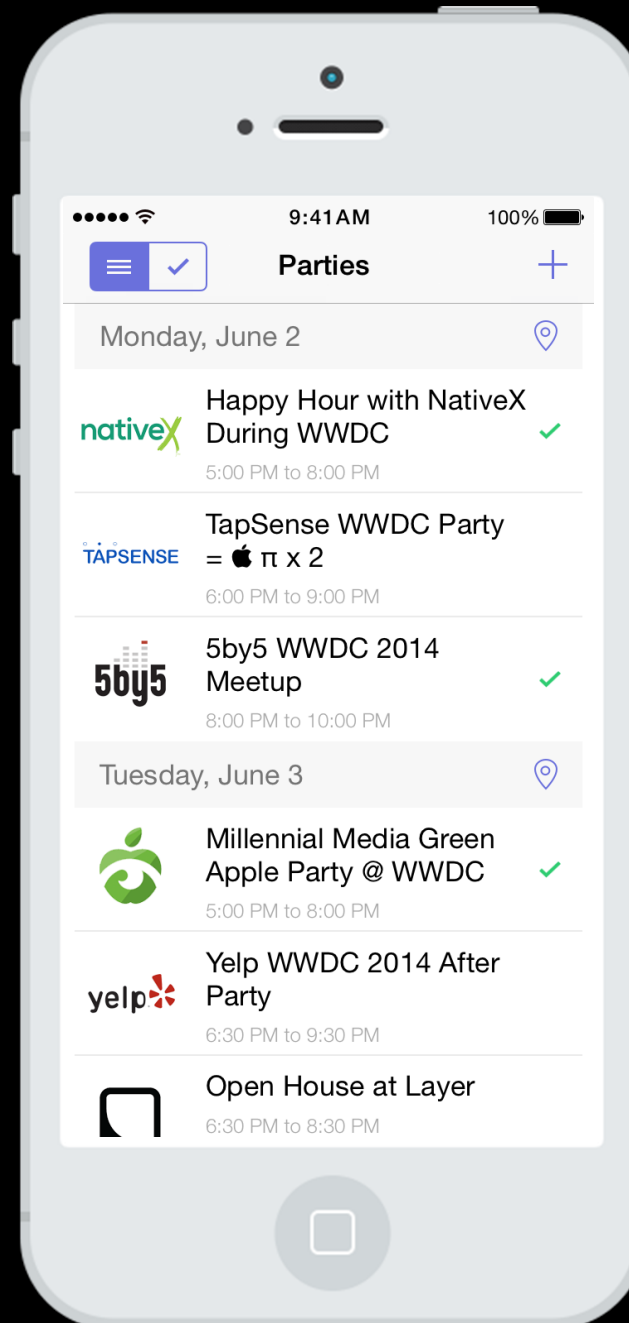
WWDC 2014

- My first WWDC
- 5000 Developers and 1000 Apple Engineers
- 50 Organized Parties (Twitter, Dropbox, etc.)



Parties for WWDC

Parties for WWDC



<https://medium.com/@genadyo/86d13d4cad7d>

Parties for WWDC

2,436
Downloads

Parties for WWDC

2,436

Downloads

8,870

Sessions

Why did I use Parse?


- Backend as a service (BAAS)
- Fast prototyping
- Lots of client side work is avoided
- Enforces best practices

What is CloudKit?

- Access to iCloud servers
- Supported on OS X and iOS
- Uses iCloud accounts
- Public and private databases
- Structured and bulk data
- Transport, not local persistence

From Parse To CloudKit

Project Capabilities

▼  iCloud

ON ☐

Services:

☐ Key-value storage

☐ iCloud Documents

☒ CloudKit

Containers:

☒ Use default container

☐ Specify custom containers

☒ iCloud.so.sugar.SFParties iCloud.\$(CFBundleIdentifier)

+ ↺

CloudKit Dashboard

CloudKit Dashboard

SFParties ▾

SCHEMA

Record Types

Security Roles

Subscription Types

PUBLIC DATA

User Records

Default Zone

PRIVATE DATA

Default Zone

For genady@okrain.com

ADMIN

Team

Environment: **DEVELOPMENT**

Record Types

Sort by Name ▾

Notification

10 Records

Party

50 Records

Users

1 Record

🗑️

+

Genady Okrain ▾

Beta

Party

Created:
Jun 21 2014 12:50 AM

Modified:
Jun 21 2014 12:51 AM

Security:
Roles ▾

Attribute Name	Attribute Type
address1	String <div>✕</div>
address2	String
address3	String
details	String
endDate	Date/Time
icon	Asset
location	Location

<https://icloud.developer.apple.com/dashboard/>

Moving data from Parse to CloudKit

```
PFQuery *query = [PFQuery queryWithClassName:@"WDCParty"];
NSArray *objects = [query findObjects];

for (PFObject *party in objects) {
    CKRecord *ckParty =
        [[CKRecord alloc] initWithRecordType:@"Party"];
    ckParty[@"title"] = party[@"title"];
    ckParty[@"address1"] = party[@"address1"];
    ...
    [[party[@"icon"] getData] writeToURL:iconURL
                                atomically:YES];
    ckParty[@"icon"] = [[CKAsset alloc]
                        initWithFileURL:iconURL];
    ...
    [publicDatabase saveRecord:ckParty completionHandler:nil];
}
```

Query Operation

```
NSPredicate *predicate =  
    [NSPredicate predicateWithFormat:@"show = 1"];  
  
CKQuery *query =  
    [[CKQuery alloc] initWithRecordType:@"Party"  
        predicate:predicate];  
  
CKQueryOperation *queryOperation =  
    [[CKQueryOperation alloc] initWithQuery:query];  
  
queryOperation.desiredKeys = @[@"title", @"address1", ...];  
  
...  
  
[publicDatabase addOperation:queryOperation];
```


Fetching Assets

```
queryOperation.recordFetchedBlock = ^(CKRecord *record) {  
    CKFetchRecordsOperation *fetchRecordsIconOperation =  
        [[CKFetchRecordsOperation alloc]  
         initWithRecordIDs:@[record.recordID]];  
  
    fetchRecordsIconOperation.desiredKeys = @[@"logo"];  
  
    fetchRecordsIconOperation.perRecordCompletionBlock =  
        ^(CKRecord *record, ...) {  
            CKAsset *logo = record[@"logo"];  
        };  
  
    [publicDatabase addOperation:fetchRecordsIconOperation];  
};
```

Records Caching

- NSUserDefaults
- CoreData
- SQLite

Records Caching

- UserDefaults
- CoreData 
- SQLite

NSUserDefaults

```
NSData *data =  
    [userDefaults objectForKey:@"parties"];  
NSArray *parties =  
    [NSKeyedUnarchiver unarchiveObjectWithData:data];  
  
NSData *data =  
    [NSKeyedArchiver archivedDataWithRootObject:parties];  
[userDefaults setObject:data forKey:@"parties"];  
[userDefaults synchronize];
```

Assets Caching

- Local Storage
- NSCache
- TMCache ([GitHub](#))

TMCache

```
UIImage *img = [[UIImage alloc] initWithData:data  
                scale:[[UIScreen mainScreen] scale]];  
[[TMCache sharedCache] setObject:img forKey:@"image"];  
  
UIImage *img = [[TMCache sharedCache] objectForKey:@"image"];
```

Subscriptions

```
UIUserNotificationSettings *settings =  
    [UIUserNotificationSettings  
        settingsForTypes:UIUserNotificationTypeAlert|  
                        UIUserNotificationTypeBadge|  
                        UIUserNotificationTypeSound categories:nil];  
[application registerUserNotificationSettings:settings];  
[application registerForRemoteNotifications];  
  
NSPredicate *predicate =  
    [NSPredicate predicateWithFormat:@"push = 1"];  
CKSubscription *subscription =  
    [[CKSubscription alloc] initWithRecordType:@"Notification"  
        predicate:predicate  
        options:CKSubscriptionOptionsFiresOnRecordCreation];  
  
[publicDatabase saveSubscription:subscription completionHandler:nil];
```

Pricing & Storage

CloudKit Storage and Data Transfers

	Start with	Grow with every user	Free up to
Storage for assets	5GB	100MB	1PB
Storage for database	50MB	1MB	10TB
Data Transfer for assets	25MB/day	0.5MB/user	5TB/day
Data Transfer for database	250KB/day	5KB/user	50GB/day

Parse Pricing

	File Storage	Database Storage	Data Transfer	Background Jobs	Request limit	Parse Push	Parse Analytics
Free	20GB	20GB	2TB	1	30 req/s	1 million unique recipients	Free
Paid	\$0.03/GB extra		\$0.10/GB extra	100\$-1700\$/month 2-10 Background Jobs 40-200 req/s		\$0.05 per 1,000 recipients extra	

Advantages and Disadvantages

Advantages

- Seamless integration
- CloudKit user accounts
- Separate private and public databases
- Push notifications subscriptions
- Free

Disadvantages

- No support for cross platform apps
- No way to export/import databases
- No cacheing
- No way to monitor storage and data transfer
- Free

Conclusions

- CloudKit is good for you
- Don't use it for you need a cross platform app
- Try it :)

Fork now!

<http://github.com/sugarso/WWDC>

Resources

CloudKit Resources

- <https://developer.apple.com/icloud/>
- Introducing CloudKit
- Advanced CloudKit

Parse Resources

- <https://parse.com>
- <https://parse.com/docs/>
- <https://parse.com/tutorials>

Sugar So Studio



@genadyo



@mvxlr



@hvost

<http://sugar.so>

Thank You!

Follow me on Twitter



[@genadyo](https://twitter.com/genadyo)