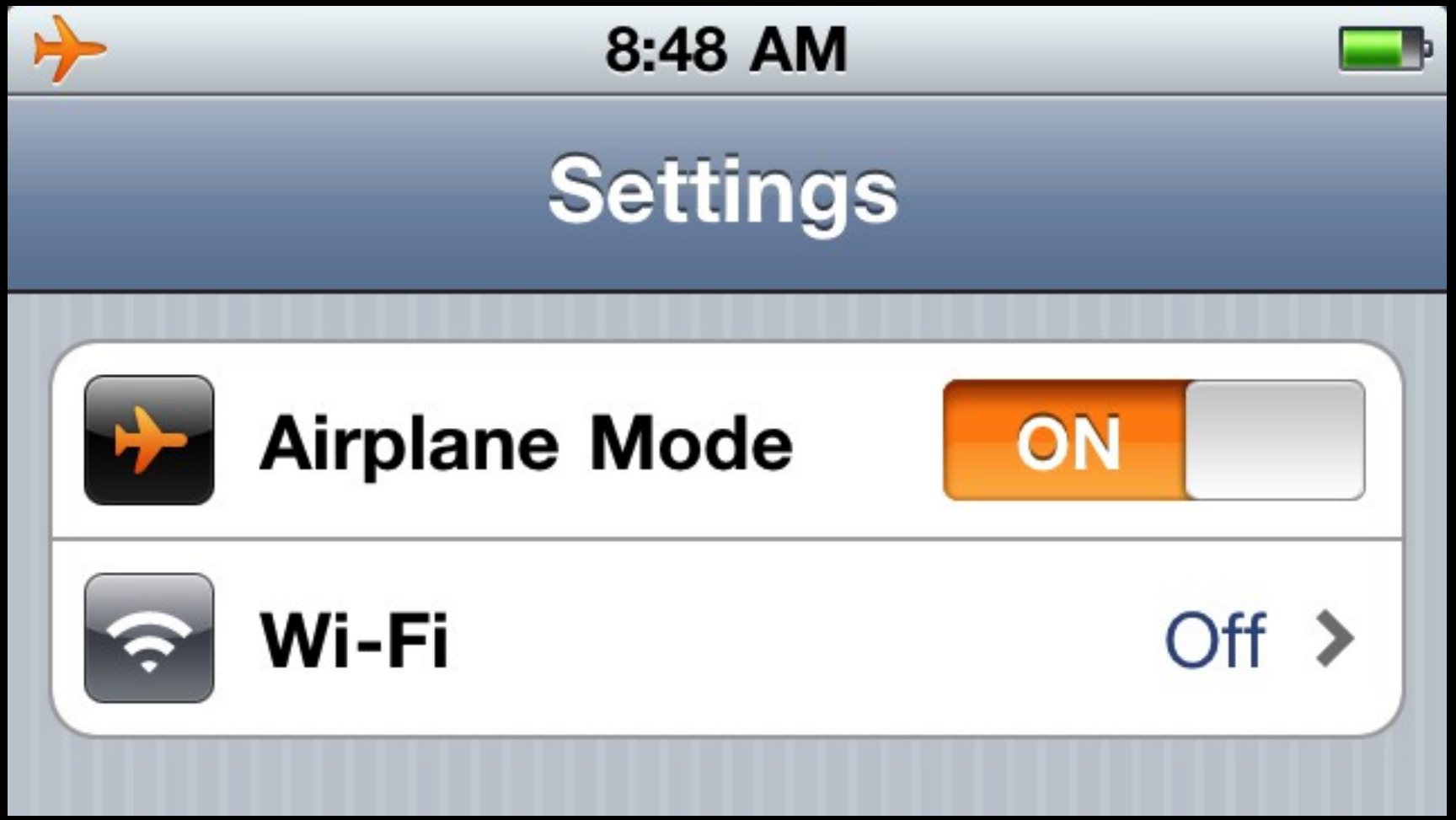
iAsync
Functional Programming
in Objective-C

Alexander Dodatko
2013


TETRIS
HBO GO
CNN
epi
GUITAR HERO
elp

Over 900,000 apps

(as of June 2013)


8:48 AM
Settings
Airplane Mode
ON
Wi-Fi
Off


NOKIA
Меню
19:08
Календарь
Контакты
Музыка
Интернет
Сообщения
Фото
Магазин
Чат
Параметры
Диспт. SIM
Подарок
Прилож.
Функции
Выбрать
Выйти
Calendar
Clock
Videos
Notes
Reminders
Compass
Phone
Music
Camera
Photos
Settings
Contacts
Calculator
Voice Memos

```
NSData* jsonData = nil;
jsonData = [ NSData dataWithContentsOfURL: RESOURSE ];
```


That's not good


45
38

Asynchronous Programming

```
@protocol NSURLConnectionDataDelegate
<NSURLConnectionDelegate>

@optional
- (void)connection:  didReceiveResponse:

- (void)connection:   didReceiveData:

- (void)connectionDidFinishLoading:

- (void)connection: didFailWithError:

@end
```

Too Much Code

Objectice-C Blocks

```
NS_AVAILABLE(10_7, 5_0);

+ (void)sendAsynchronousRequest:
                          queue:
                completionHandler:
```

```
void (^HANDLER_BLOCK)(
    NSURLResponse* response,
    NSData* data,
    NSError* connectionError)
```


45
38

Progress

Cancel


AFN


```
NSURL *url = [NSURL URLWithString:@"https://...."];
NSURLRequest *request = [NSURLRequest requestWithURL:url];

id onSuccess = ^(NSURLRequest *, NSHTTPURLResponse *, id JSON)
{
      NSLog( @"App.net Global Stream: %@", JSON );
 };

AFJSONRequestOperation *operation =
[ AFJSONRequestOperation JSONRequestOperationWithRequest: request
                                                 success: onSuccess
                                                   failure: nil ];
 [ operation start ];
```

Dependencies

When using methods which return blocks they can be very convenient. However, when you have to string a few of them together it gets messy really quickly

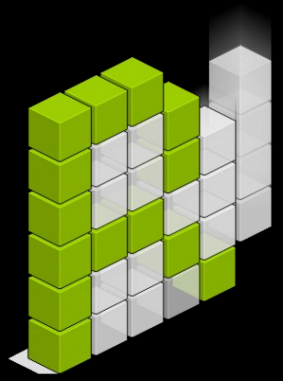
```
[remoteAPIWithURL:url1 success:^(int status){
    [remoteAPIWithURL:url2 success:^(int status){
        [remoteAPIWithURL:url3 success:^(int status){
            [remoteAPIWithURL:url2 success:^(int status){
            //success!!!
            }];
        }];
    }];
}];
```

It was not uncommon when using AFNetworking in Gowalla to have calls chained together in success blocks.
My advice would be to factor the network requests and serializations as best you can into class methods in your model. Then, for requests that need to make sub-requets, you can call those methods in the success block.

Callback Hell

```
- (void)suspendURLIfDownloading:(NSURL *)url automaticallyResumeLater:(BOOL)autoResume
{
    [_session getTasksWithCompletionHandler:^(NSArray *dataTasks, NSArray *uploadTasks, NSArray *downloadTasks) {
        [downloadTasks enumerateObjectsUsingBlock:^(id obj, NSUInteger idx, BOOL *stop) {
            NSURLSessionDownloadTask *task = (NSURLSessionDownloadTask *)obj;
            if ([task.originalRequest.URL.absoluteString isEqualToString:url.absoluteString]) {
                *stop = YES;
                [task cancelByProducingResumeData:^(NSData *resumeData) {
                    [_queue addOperationWithBlock:^{
                        [_db executeUpdate:
                            @"UPDATE FCDownloadQueue SET state = ?, resumeData = ? WHERE url = ?",
                            @( autoResume ? FCDownloadStatePausedAutoResume : FCDownloadStatePausedDoNotAutoResume ),
                            resumeData ? resumeData : [NSNull null],
                            url.absoluteString
                        ];
                        [self.delegate downloadQueue:self didPauseDownloadingURL:url userInfo:[self userInfoForURL:url]];
                        [self ensureEnoughDownloadsAreRunning];
                    }];
                }];
            }
        }];
    }];
}
```

AFURLConnectionOperation



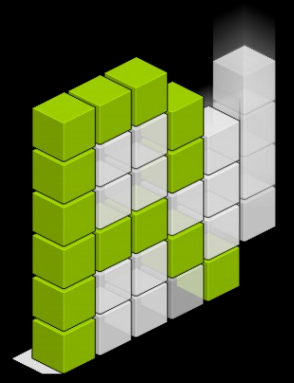
Working with independent data sets in parallel and then combining them into a final result is non-trivial in Cocoa, and often involves a lot of synchronization

```
__block NSArray *databaseObjects, fileContents;
NSOperationQueue *backgroundQueue = [[NSOperationQueue alloc] init];
NSBlockOperation *databaseOperation = [NSBlockOperation blockOperationWithBlock:^{
  databaseObjects = [databaseClient fetchObjectsMatchingPredicate:predicate];
}];

NSBlockOperation *filesOperation = [NSBlockOperation blockOperationWithBlock:^{
  NSMutableArray *filesInProgress = [NSMutableArray array];
  for (NSString *path in files) {
    [filesInProgress addObject:[NSData dataWithContentsOfFile:path]];
  }

  fileContents = [filesInProgress copy];
}];

NSBlockOperation *finishOperation = [NSBlockOperation blockOperationWithBlock:^{
  [self finishProcessingDatabaseObjects:databaseObjects fileContents:fileContents];
  NSLog(@"Done processing");
}];
[finishOperation addDependency:databaseOperation];
[finishOperation addDependency:filesOperation];
[backgroundQueue addOperation:databaseOperation];
[backgroundQueue addOperation:filesOperation];
[backgroundQueue addOperation:finishOperation];
```



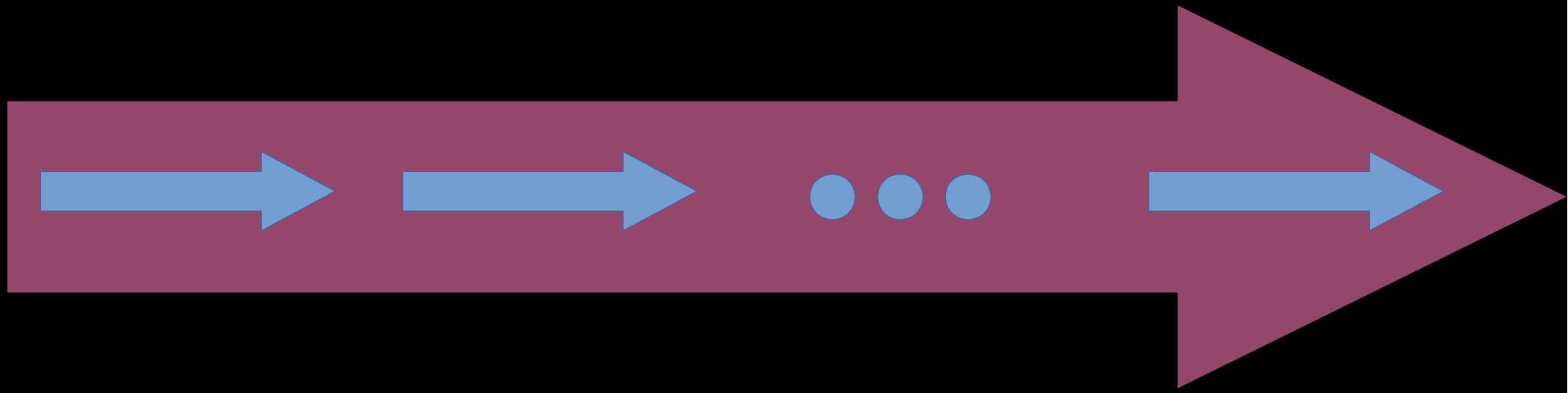
Rather than using mutable variables which are replaced and modified in-place, RAC provides signals (represented by RACSignal) that capture present and future values.

When Design Patterns are NOT enough

Meet the Async Operation

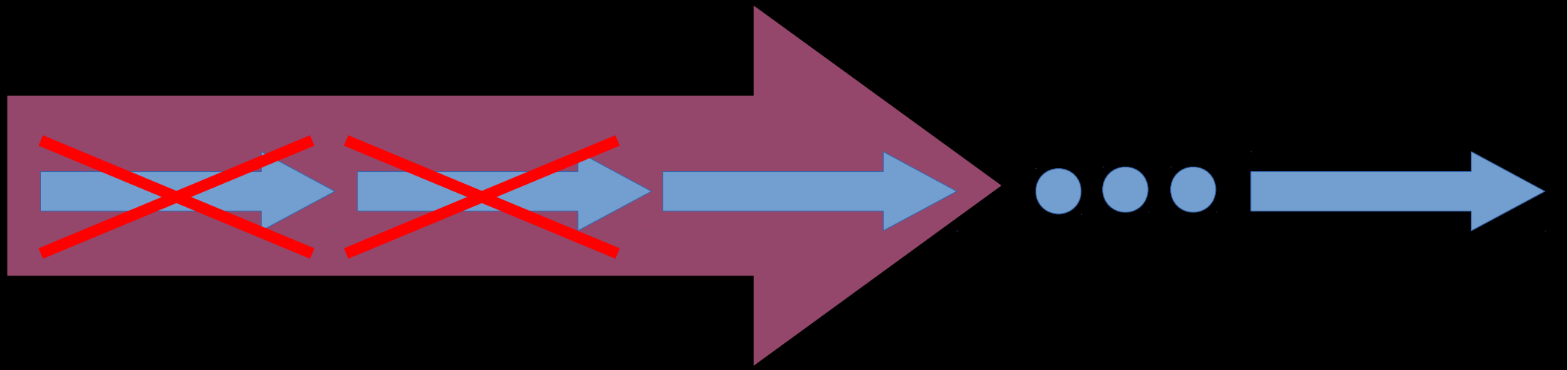
```
CancelBlock (^AsyncOperation)
    ( ProgressCallback,
      CancelCallback,
      CompletionCallback ) {...};
```

Sequence



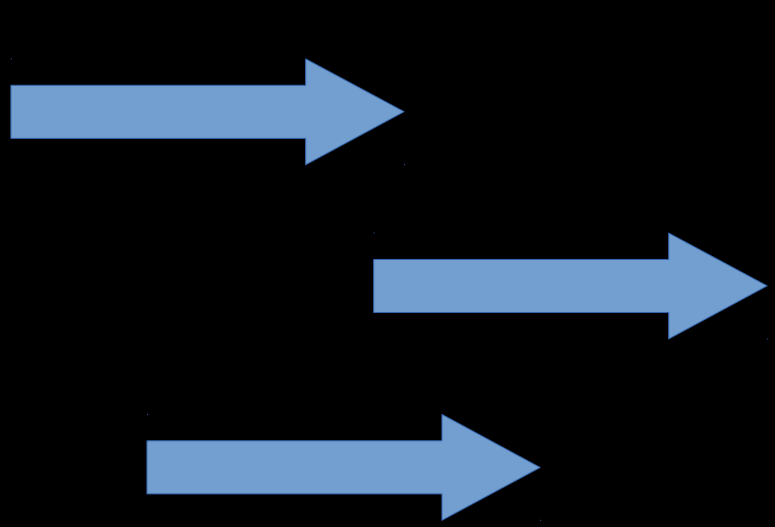
```
JFFAsyncOperation sequence =
    sequenceOfAsyncOperations(
        @[ op1, op2, opN ]  );
```

Sequence of Attempts



```
JFFAsyncOperation sequence =
    trySequenceOfAsyncOperations(
        @[ op1, op2, opN ]  );
```

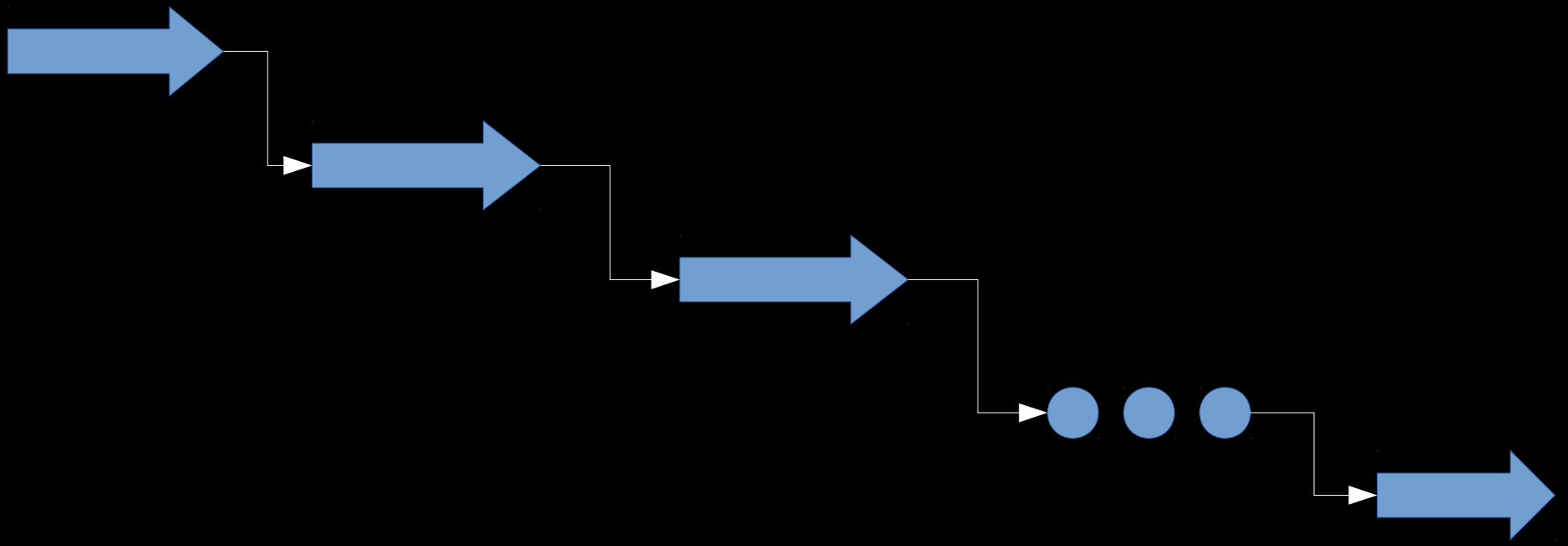
Parallel execution



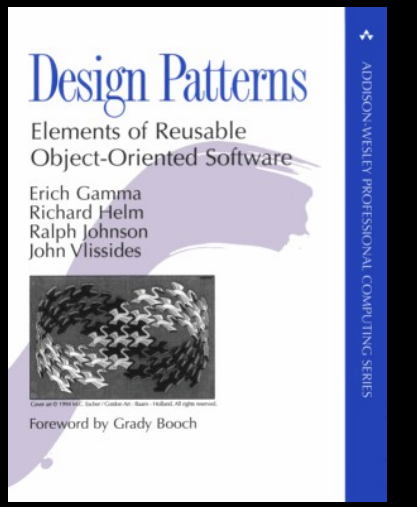
CompletionCallback

```
JFFAsyncOperation sequence =
    groupOfAsyncOperations(
        @[ op1, op2, opN ]  );
```

Waterfall



```
JFFAsyncOperation sequence =
    bindSequenceOfAsyncOperations(
        op1, @[ b1, b2, bN ]  );
```



Function = Command + Composite

Any Complexity Tasks

```
loadFromNet = sequenceOfAsyncOperations(
@[ login, loadFromNet ] );

loadRawData = trySequenceOfAsyncOperations(
@[ loadFromFSCache, loadFromNet  ] );

JFFAsyncOperation getData =
bindSequenceOfAsyncOperations(
loadRawData, @[ parseBinder, filterBinder ] );
```

iAsync

bit.ly/1bLdN59

Does anyone use it?

Wishdates

Sitecore

Pitfalls

Large codebase

No PodSpec

No Documentation

No Community

No Versioning policy

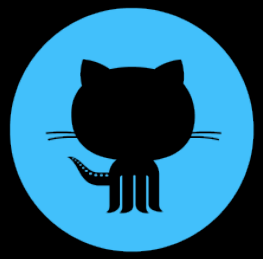

DEVELOPERS
DEVELOPERS
DEVELOPERS
DEVELOPERS

iAsync Author

Vladimir Gorbenko

Gorbenko.Vova@gmail.com

Vova.Gorbenko.mac



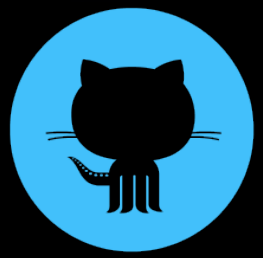
github.com/volodg

Oleksandr Dodatko

dodikk88.tutor@gmail.com

alexander.dodatko.work

@dodikk88



github.com/dodikk