iOS development. Daily.

(RestKit, Storyboards, Massive View Controllers)

Stanislaw Pankevich
Provectus, Kazan
October 10, 2014

1. Everyday tools

- RestKit
- Storyboards
- Auto-layouts
- Schemes, Configurations
- Debugging

2. Massive View Controllers

- Code organization
- Lightweight View Controllers
- Anti-patterns

1. Everyday tools

RestKit

+

- Mapping JSON <-> CoreData
- 304
- Orphaned objects

-

- Still hardcoded to AFNetworking 1.3
- Ugly logging => AFNetworkingLogger
- Lack of support these days

Storyboards

RBStoryboardLink

https://github.com/rob-brown/RBStoryboardLink

Shared Xib-based UIViewControllers

Auto-layouts

Schemes, Configurations

Schemes

- Debug, Development (Staging), Production, Unit Tests, Integration Tests
- D- and P- TestFlight builds

Configurations

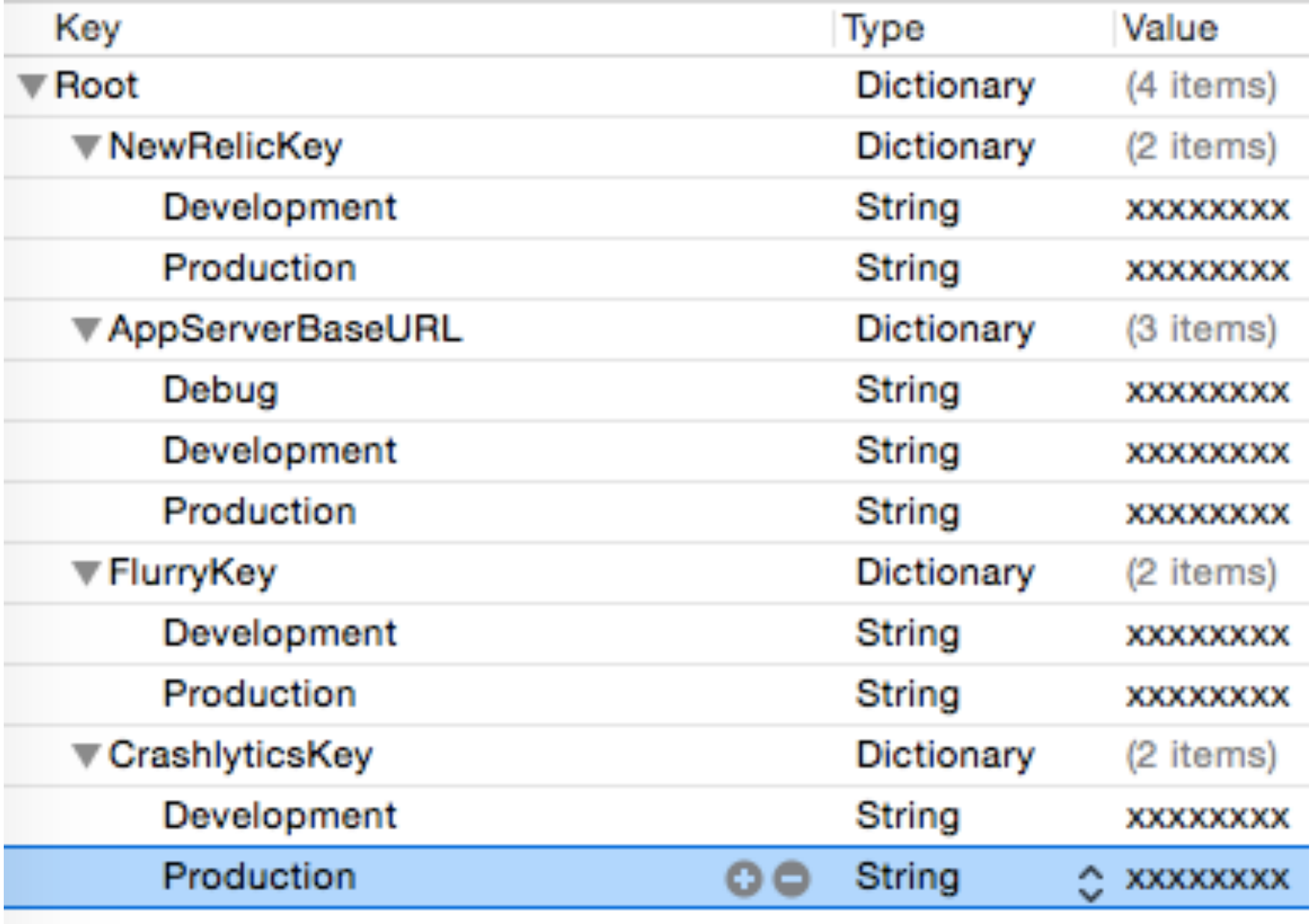
- Xcode Configurations
- $(CONFIGURATION)
- Plist
- -[AppConfiguration currentConfiguration]

http://code.tutsplus.com/tutorials/ios-quick-tip-managing-configurations-with-ease--mobile-18324


Configurations
Name
Based on Configuration File
Debug
2 Configurations Set
Development
2 Configurations Set
Production
2 Configurations Set
Use
Debug
for command-line builds

Custom iOS Target Properties

Key	Type	Value
Bundle versions string, short	String	0.4
Bundle identifier	String	com.app.${CONFIGURATION:lower}
InfoDictionary version	String	6.0
Main storyboard file base name	String	Auth
Bundle version	String	122
Bundle name	String	App
Executable file	String	${EXECUTABLE_NAME}
Application requires iPhone environment	Boolean	YES
▶ Fonts provided by application	Array	(1 item)
▶ Supported interface orientations	Array	(1 item)
Bundle display name	String	App D122
Bundle OS Type code	String	APPL
Bundle creator OS Type code	String	????
Status bar style	String	Gray style (default)
Configuration	String	${CONFIGURATION}
Localization native development region	String	en
▶ Required device capabilities	Array	(1 item)


Key
Type
Value
Root
Dictionary
(4 items)
NewRelicKey
Dictionary
(2 items)
Development
String
xxxxxxxx
Production
String
xxxxxxxx
AppServerBaseURL
Dictionary
(3 items)
Debug
String
xxxxxxxx
Development
String
xxxxxxxx
Production
String
xxxxxxxx
FlurryKey
Dictionary
(2 items)
Development
String
xxxxxxxx
Production
String
xxxxxxxx
CrashlyticsKey
Dictionary
(2 items)
Development
String
xxxxxxxx
Production
String
xxxxxxxx

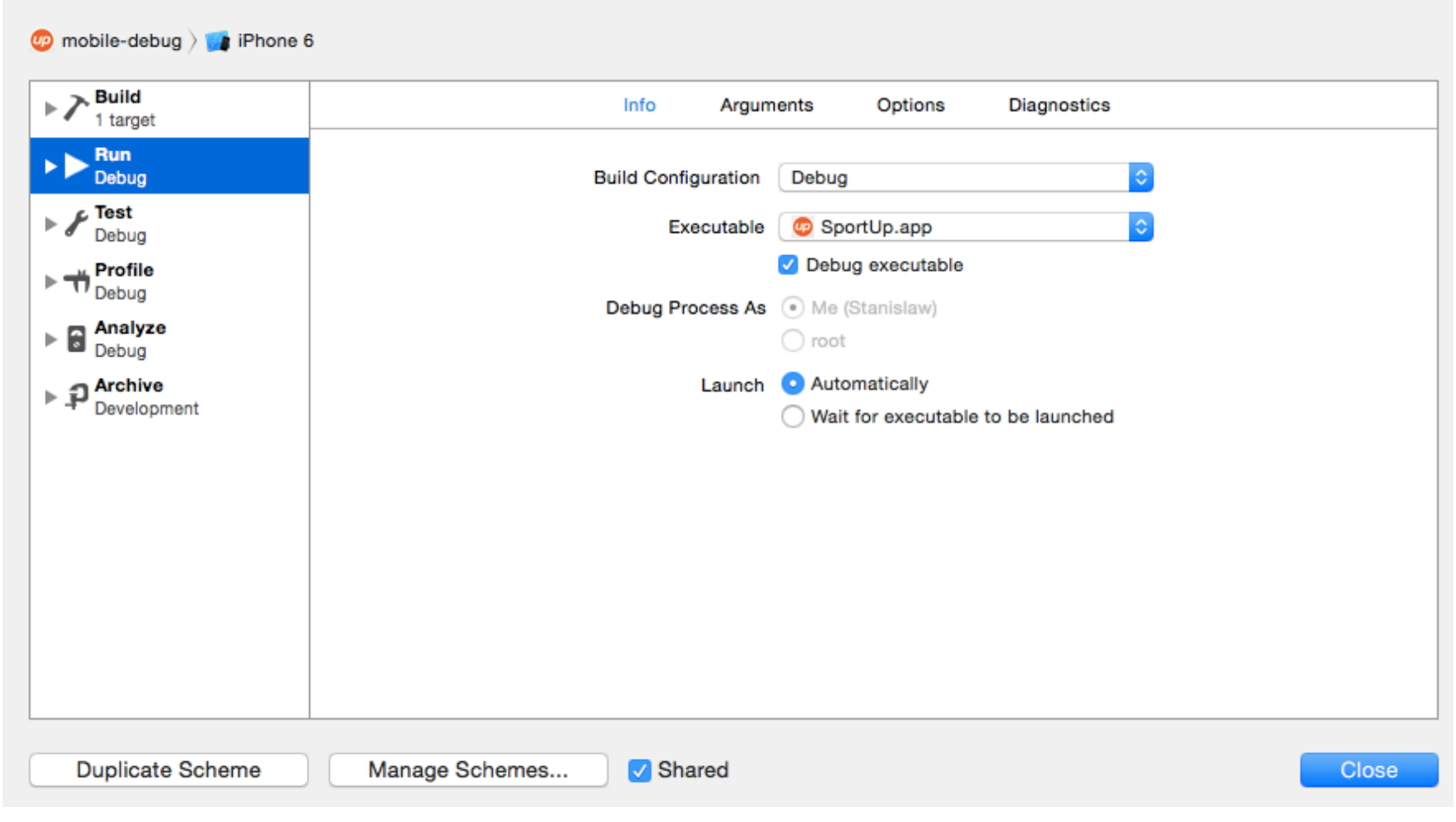
```objc
NSString *currentConfigurationName =
    [NSBundle bundleForClass:[self class]].infoDictionary[@"Configuration"];

NSString *configurationsPath = [[NSBundle bundleForClass:[self class]]
                                        pathForResource:@"AppConfigurations"
                                                 ofType:@"plist"];

NSDictionary *configurationsData =
    [NSDictionary dictionaryWithContentsOfFile:configurationsPath];

// Wrap into class AppConfiguration
// ...

// And use
NSURL *baseURL = [SportUpConfiguration currentConfiguration].yourAPI.endpointURL;
```


mobile-debug
iPhone 6
Build
1 target
Run
Debug
Test
Debug
Profile
Debug
Analyze
Debug
Archive
Development
Info
Arguments
Options
Diagnostics
Build Configuration
Debug
Executable
SportUp.app
Debug executable
Debug Process As
Me (Stanislaw)
root
Launch
Automatically
Wait for executable to be launched
Duplicate Scheme
Manage Schemes...
Shared
Close

Debugging

AFNetworkingLogger

https://github.com/stanislaw/AFNetworkingLogger

```
#import <AFNetworking/AFNetworking.h>
#import "AFNetworkingLogger.h"

// ...

[AFNetworkingLogger sharedLogger].level = AFNetworkingLoggerLevelVerbose;
[[AFNetworkingLogger sharedLogger] startLogging];
```

```
GET https://www.google.com.ua/search?
    client=firefox-a&
    q=W A Mozart&
    channel=fflb&
    gws_rd=cr&
    oe=utf-8&
    rls=org.mozilla:en-US:official&
    ie=utf-8&
    ei=Z_9DUubbM8rAhAent4DIBg


200 https://www.google.com.ua/search?
    client=firefox-a&
    q=W A Mozart&
    channel=fflb&
    gws_rd=cr&
    oe=utf-8&
    rls=org.mozilla:en-US:official&
    ie=utf-8&
    ei=Z_9DUubbM8rAhAent4DIBg

 -> 42890 bytes in 0.525s

    X-XSS-Protection   = 1; mode=block
    Expires            = -1
    X-Frame-Options    = SAMEORIGIN
    Content-Type       = text/html; charset=UTF-8
    P3P                = CP="This is not a P3P policy! See http://www.google.com/support/accounts/bin/
answer.py?hl=en&answer=151657 for more info."
    Server             = gws
    Alternate-Protocol = 443:quic
    Date               = Mon, 14 Oct 2013 21:22:40 GMT
    Cache-Control      = private, max-age=0
    Set-Cookie         = PREF=ID=49c7bea01fa399db:FF=0:TM=1381785760:LM=1381785760:S=6ImGFlMKbcaXwR2C;
expires=Wed, 14-Oct-2015 21:22:40 GMT; path=/; domain=.google.com.ua, NID=67=e8tXQXY-
LksUhq1ErdgBmeAj3U_nFbY9Z3YDKxUXUiMIKdEs1k5QjlWMam2SfRwT9sKnglv2jLW79RNdgUrfHyVc42ygnXBagtZLx3RGcRg_yjb
E5gJLWdV86r-cB_H9; expires=Tue, 15-Apr-2014 21:22:40 GMT; path=/; domain=.google.com.ua; HttpOnly
    Transfer-Encoding  = Identity

    <!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"><head><meta http-
equiv="Content-Type" content="text/html; charset=UTF-8"><meta itemprop="image" content="/images/
google_favicon_128.png"><title>W A Mozart - Пошук Google</title><style>#gbar,#guser{font-size:
13px;padding-top: ...TRUNCATED...
```

EchoLogger

https://github.com/stanislaw/EchoLogger

```objc
#import <EchoLogger/EchoLogger.h>

// ...

int intNumber = 10;
double doubleNumber = 5.55;
BOOL boolValue = YES;
NSUInteger unsignedIntNumber = 7;
NSNumber *number = @(18);
NSString *string = @"I'am the string!";
CGRect frame = (CGRect){0, 0, 200, 200};
CGSize size = (CGSize){200, 200};

L(intNumber);
L(doubleNumber);
L(boolValue);
L(unsignedIntNumber);
L(number);
L(string);

L(intNumber, doubleNumber, boolValue, unsignedIntNumber, number, string);
L(frame, size);

LL(intNumber, doubleNumber, boolValue, unsignedIntNumber, number, string);
LL(frame, size);
```

```
EL> intNumber
-> (int)10
EL> doubleNumber
-> (double)5.550000
EL> boolValue
-> YES
EL> unsignedIntNumber
-> 7
EL> number
-> @(18)
EL> string
-> (__NSCFConstantString)I'am the string!
EL> intNumber, doubleNumber, boolValue, unsignedIntNumber, number, string
-> (int)10; (double)5.550000; YES; 7; @(18); (__NSCFConstantString)I'am the string!
EL> frame, size
-> (CGRect){0.000000, 0.000000, 200.000000, 200.000000}; (CGSize){200.000000, 200.000000}
EL: AppDelegate.m:45; -[AppDelegate application:didFinishLaunchingWithOptions:]; com.apple.main-thread>
intNumber, doubleNumber, boolValue, unsignedIntNumber, number, string
-> (int)10; (double)5.550000; YES; 7; @(18); (__NSCFConstantString)I'am the string!
EL: AppDelegate.m:46; -[AppDelegate application:didFinishLaunchingWithOptions:]; com.apple.main-thread> frame,
size
-> (CGRect){0.000000, 0.000000, 200.000000, 200.000000}; (CGSize){200.000000, 200.000000}
```

Xtrace

https://github.com/johnno1962/Xtrace

[myUILabel xtrace]

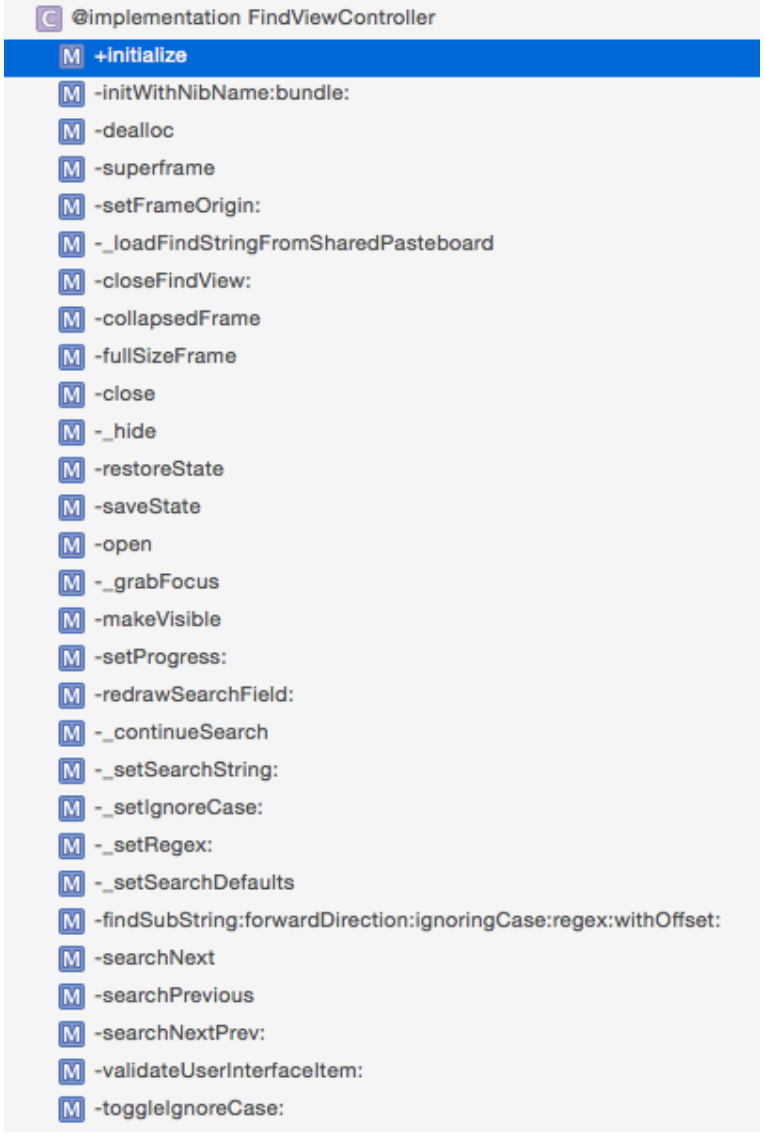
```
  [<UILabel 0x8d4f170> setCenter:{240, 160}] v16@0:4{CGPoint=ff}8
   [<UILabel 0x8d4f170> actionForLayer:<CALayer 0x8d69410> forKey:<__NSCFString 0x8a53
    [<UILabel 0x8d4f170> _shouldAnimatePropertyWithKey:<__NSCFString 0x8a535e0>] c12@0
    -> 1 (_shouldAnimatePropertyWithKey:)
   -> <NSNull 0x194d068> (actionForLayer:forKey:)
 [<UILabel 0x8d4f170> window] @8@0:4
 -> <UIWindow 0x8a69920> (window)
 [<UILabel 0x8d4f170> _isAncestorOfFirstResponder] c8@0:4
 -> 0 (_isAncestorOfFirstResponder)
[<UILabel 0x8d4f170> layoutSublayersOfLayer:<CALayer 0x8d69410>] v12@0:4@8
 [<UILabel 0x8d4f170> _viewControllerToNotifyOnLayoutSubviews] @8@0:4
  [<UILabel 0x8d4f170> _viewDelegate] @8@0:4
  -> <nil 0x0> (_viewDelegate)
```

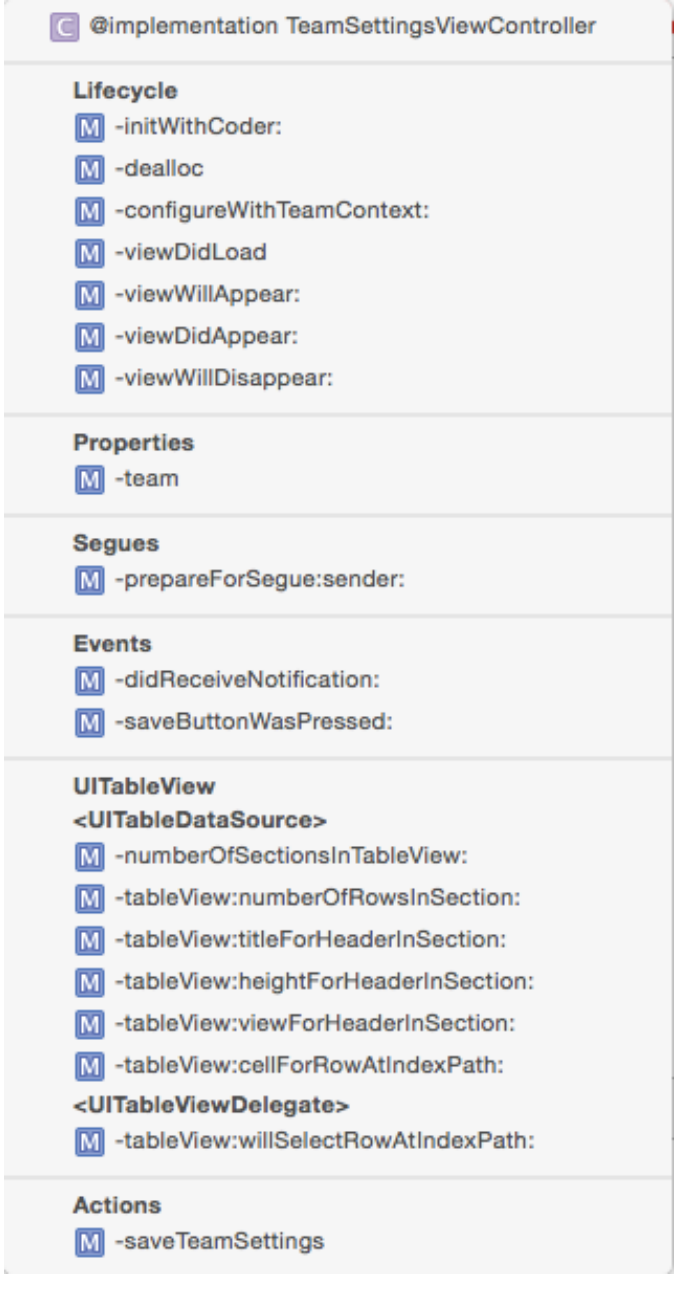
Massive View Controllers

Pragmas

```
#pragma mark - <UITableViewDataSource>
```

Somewhere in iTerm2...




@implementation TeamSettingsViewController
Lifecycle
-initWithCoder:
-dealloc
-configureWithTeamContext:
-viewDidLoad
-viewWillAppear:
-viewDidAppear:
-viewWillDisappear:
Properties
-team
Segues
-prepareForSegue:sender:
Events
-didReceiveNotification:
-saveButtonWasPressed:
UITableView
<UITableDataSource>
-numberOfSectionsInTableView:
-tableView:numberOfRowsInSection:
-tableView:titleForHeaderInSection:
-tableView:heightForHeaderInSection:
-tableView:viewForHeaderInSection:
-tableView:cellForRowAtIndexPath:
<UITableViewDelegate>
-tableView:willSelectRowAtIndexPath:
Actions
-saveTeamSettings

UIViewController Lifecycle

- Minimize logic within UIViewController template methods
- Do not rely on viewDidLoad
- As much as possible Storyboards

Entry point

- Dependency injection
- Prefer method injection to property injection
- Configure methods

UIViewController.h

- No IBOutlets (should go private)
- No IBActions (should go private)
- No @properties (should go private)
- Use forward declarations: @class, @protocol. http://railsware.com/blog/2013/08/09/using-forward-declaration-in-your-objective-c-projects
- Empty or entry point only

Before we get to anti-patterns...

- Citizenship
- Relationship
- Anonimity
- Explicit vs Implicit
- Declarative vs Imperative
- Gravitation

```objc
- (IBAction)nextButtonPressed:(id)sender {
    [[AnalyticsManager sharedManager] logEvent:@"Team/Create/Step1/Next"];

    [self.view endEditing:YES];

    self.errorRows = [[NSMutableArray alloc] init];

    NSMutableArray *errors = [[NSMutableArray alloc] init];

    if (NSStringIsStringAndNotEmpty(self.team.model.name) == NO) {
        [errors addObject:@"Please enter team name"];
        [self addErrorRow:SectionTeamInfoRowName];
    }
    if (!self.team.model.sportId || self.team.model.sportId.integerValue == 0) {
        [errors addObject:@"Please select sport"];
        [self addErrorRow:SectionTeamInfoRowSport];
    }
    if (!self.team.model.seasonId || self.team.model.seasonId.integerValue == 0) {
        [errors addObject:@"Please select season"];
        [self addErrorRow:SectionTeamInfoRowSeason];
    }
    if (!self.team.model.ageId || self.team.model.ageId.integerValue == 0) {
        [errors addObject:@"Please select age group"];
        [self addErrorRow:SectionTeamInfoRowAgeGroup];
    }

    [self.tableView reloadData];

    if (errors.count > 0) {
        [self showErrors:errors];
        return;
    }

    [self hideErrors];

    [self performSegueWithIdentifier:@"CreateTeamFirstStage->CreateTeamSecondStage" sender:nil];
}
```

```objc
- (IBAction)saveBtnPressed:(id)sender {
    [self.view endEditing:YES];

    self.errorRows = [[NSMutableArray alloc] init];

    NSMutableArray *errors = [[NSMutableArray alloc] init];
    if (!self.team.name || [self.team.name isEqualToString:@""]) {
        [errors addObject:@"Please enter team name"];
        [self addErrorRow:SectionTeamInfoRowName];
    }
    [self.tableView reloadData];
    if (errors.count > 0) {
        [self showErrors:errors];
        [self.tableView reloadData];
        return;
    }
    [self hideErrors];

    NSNumber *oldTimeZoneId = self.team.currentPersistedTimeZoneId;

    void (^handler)(id response, NSError *error) = ^(id team, NSError *error) {
        [HFUtils hideRequestHud];

        if (team) {
            self.teamContext.team = team;

            [self.team updateAllEventsToHaveCorrectUTCDatesIfNeeded:oldTimeZoneId];

            [self setCalendarSyncEnabledForThisTeam:self.calendarSyncEnabledForThisTeamLocally];
            [[NSNotificationCenter defaultCenter] postNotificationName:SUStartCalendarSynchronizationRequestNotification
                                                                object:self];

            [[AnalyticsManager sharedManager] logEvent:@"Team/Settings/Saved" withParameters:@{ @"teamId": self.team.id_ }];
        } else {
            [self processServerErrors:error
                          withFields:self.fieldsCriticalToValidations];
        }
    };

    [HFUtils showRequestProgressHud];

    UIImage *updatedImage = self.avatarChanged ? self.avatarView.stretchedAvatarImage : nil;

    [[ApiClient instance] updateTeam:self.team
                           withImage:updatedImage
                         withHandler:handler];
}
```

UIViewController.m

- No networking
- No reachability checks
- Say it again: No logic inside viewDidLoad!
- Minimize logic inside viewWillAppear
- Ideal - UIViewController without asynchronicity

Anti-patterns

- Private methods level > 1 are evil
- Singletons are evil
- Always prefer @property to ivars (never access ivar unless you're in init or dealloc (c) Github Conventions)
- Do not create abstract View Controller classes or handle with care. Factory methods are evil, difficult to manager 'super' within methods of lifecycle. Two-level inheritance is evil!
- Avoid umbrella utils classes like Utils, Tools, Helpers, Methods
- Avoid using Objective-C Categories for implicit, anonymous things without any known citizenship :)
- Blocks may be evil

Clever SRP applied to UIViewControllers

"Single Responsibility Principle & iOS"

http://bendyworks.com/single-responsibility-principle-ios/

Intentions. An experiment in Ultralight View Controllers

http://chris.eidhof.nl/posts/intentions.html

Lighter View Controllers

http://www.objc.io/issue-1/lighter-view-controllers.html

One might think, “I already write my View Controllers in that way!” Well, to be sure we’re on the same page, allow me propose rules that must be followed in order to adhere to this heuristic:

The IBAction macro must not be used in a View Controller

The @interface block in a View Controller’s header file must be blank.

A View Controller may not implement any extra *Delegate or *DataSource protocols except for observing Model changes.

A View Controller may only do work in viewDidLoad (or awakeFromNib), viewDidAppear, viewDidDisappear, and in response to an observed change in the model layer.

To abide by these rules is to respect the Single Responsibility Principle in View Controllers.

http://bendyworks.com/single-responsibility-principle-ios/

Just-Controllers

- NSFetchResultsController, CLClusteringController etc
- Session (data) and SessionController (logic)
- Controller as context where models can meet
- Data Controllers

Data Controllers

```objc
typedef NS_ENUM(NSInteger, DataControllerRefreshStatus) {
    DataControllerRefreshStatusError = -1,
    DataControllerRefreshStatusOK = 1,
};

typedef NS_ENUM(NSInteger, DataControllerRefreshPolicy) {
    DataSourceRefreshPolicyDefault             = 0,

    DataControllerRefreshPolicyLocalElseRemote = 1,
    DataControllerRefreshPolicyRemoteThenLocal = 2,
    DataControllerRefreshPolicyLocalOnly       = 3
};

@class DataController;

@protocol DataControllerDelegate <NSObject>

- (void)dataController:(DataController *)dataController
  didRefreshWithPolicy:(DataControllerRefreshPolicy)refreshPolicy
                status:(DataControllerRefreshStatus)refreshStatus;

@end

@interface DataController : NSObject

- (id)initWithDelegate:(id <DataControllerDelegate>)delegate;

- (void)refreshUsingPolicy:(DataControllerRefreshPolicy)refreshPolicy;

@end
```

```objc
#import "TeamListViewController.h"
#import "TeamsDataController.h"

@interface TeamListViewController () <DataControllerDelegate>
@property (strong, nonatomic) TeamsDataController *teamsDataController;
@end

@implementation TeamListViewController

- (id)initWithCoder:(NSCoder *)aDecoder {
    self = [super initWithCoder:aDecoder];
    _teamsDataController = [[TeamsDataController alloc] initWithDelegate:self];
    return self;
}

- (void)viewWillAppear:(BOOL)animated {
    [super viewWillAppear:animated];
    [self.teamsDataController refreshUsingPolicy:DataControllerRefreshPolicyLocalElseRemote];
}

- (void)refreshControlWasPulled:(id)sender {
    [self.teamsDataController refreshUsingPolicy:DataControllerRefreshPolicyRemoteThenLocal];
}

#pragma mark - <DataController>
- (void)dataController:(DataController *)dataController
  didRefreshWithPolicy:(DataControllerRefreshPolicy)refreshPolicy
                status:(DataControllerRefreshStatus)refreshStatus {
    // ...
}

#pragma mark - UITableView
// ... UITableView that uses data controller as its data source

@end
```

- Structure is more complex. I have many classes, and quite a few dependencies. It is worth noting that in practice, most of those classes would be 5 to 10 lines long. Easy to write (bug-free), easy to test, and easy to understand. But the overall shape may not be easy. Here is where I see the value of a diagram as a communication tool.
- The centralized controller is easier for the junior guy. Which is another way to say Worse is Better. Yeah, well, it's true. Beginners are usually better at understanding complex logic than complex structure and interactions. There is also some evidence that a diagram can help there. But overall, it's a non-trivial choice about the kind of software you want to write (for beginners or for expert programmers).
- The process is no longer "visible". This is true, as there is no longer a central piece of software encoding the entire process. It has been scattered among cooperating objects. This is another facet of the same problem: for those who don't get OO, this shape is harder to understand. For those who do, it provides a number of non-functional benefits.
- It's easier to add stuff in the centralized controller. Again, this is a facet of the same problem, with a different slant. Every once in a while, someone tells me that when he's working with a sophisticated structure, he needs to actually think before adding stuff. Where do I put this logic without breaking the conceptual integrity of the whole? When you have little or no structure to begin with, you just don't care, and you can take the path of least resistance.
- Of course we know where this leads: to the big ball of mud, the natural destiny of those who surrender to gravity. Depending on your business model and professionalism, you may still not care.

taken from "Life without a controller, case 1" by Carlo Pescio

More ideas

- UIViewControllers should not rule i.e. drive
- Looking for decent Routing implementation
- Storyboard as State-machine
- WADL, WADL2OBJC
- IO
- Easy testing of UIViewControllers

Questions?

Stanislaw Pankevich

email: s.pankevich@gmail.com
https://github.com/stanislaw
https://twitter.com/sbpankevich
https://www.linkedin.com/in/stanislawpankevich