iOS / Twitter Integration

Nicolas Seriot

{{ softshake }}

incubator

{{ cocktail of computing backgrounds }}

Geneva

2013 OCTOBER

24&25@HEPIA

third edition

Abstract

Cette présentation explique le fonctionnement et les subtilités de l'**API Twitter** ainsi que les différentes manières d'y accéder **depuis une application iOS**.

Dans le cas le plus simple, le développeur iOS peut utiliser les comptes indiqués dans les réglages de l'appareil pour envoyer un tweet. Il peut également, même si c'est plus compliqué, accéder à l'API complète pour, par exemple, **suivre un compte** ou **récupérer une timeline**.

Si aucun compte Twitter n'est paramétré dans l'appareil, le développeur peut demander à l'utilisateur ses identifiants puis créer lui-même des requêtes. Le développeur doit alors gérer l'authentification avec le protocole **OAuth** et ses différentes variantes telles que **xAuth**.

Le développeur peut aussi proposer à l'utilisateur de **s'authentifier dans Safari** puis récupérer les jetons d'accès de l'utilisateur.

Le développeur peut encore demander à l'utilisateur d'autoriser une application Twitter tierce à utiliser le compte qu'il a configuré dans l'appareil (**reverse auth.**). Pratique par exemple pour authentifier l'utilisateur auprès d'un service tiers ou pour pouvoir accéder à ses "**direct messages**".

Dans le cas où l'utilisateur n'a pas de compte Twitter, une application peut tout de même accéder à un sous-ensemble de l'API Twitter, pour peu qu'elle s'authentifie avec le protocole dit "**App Only**".

J'expliquerai simplement ces différents cas d'utilisation et montrerai pour chacun une **implémentation en Objective-C**. Vous découvrirez la **librairie STTwitter**, capable de gérer les différents modes d'authentification et qui fournit des méthodes Objective-C pour chaque point de l'API Twitter.

AppSec 2012


Twitter relies on OAuth to control 3rd party clients.
Bad idea! secret tokens are easy to extract.
Leaked consumer secrets can result in API abuses, DoS and session fixation attacks.

HITB 2013 Amsterdam

HITB, April 10th

April 23rd, 2013

AppSec 2013

request_key

Create Twitter accounts in batch!

access_token

✗ Users

My.app Server

Remove ads from Twitter.app.

#APPSEC #YPARC #YVERDON 15-10-2013 (cc) by @Bizcom

Abusing Twitter API

Table of Contents

http://seriot.ch/abusing_twitter_api.php

Side Effect

- Knowledge of Twitter API
- Knowledge of different flows
- Development of an Objective-C library

Agenda

1. **Access Twitter API**
2. Twitter API
3. STTwitter

1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

1.6 Reverse Auth

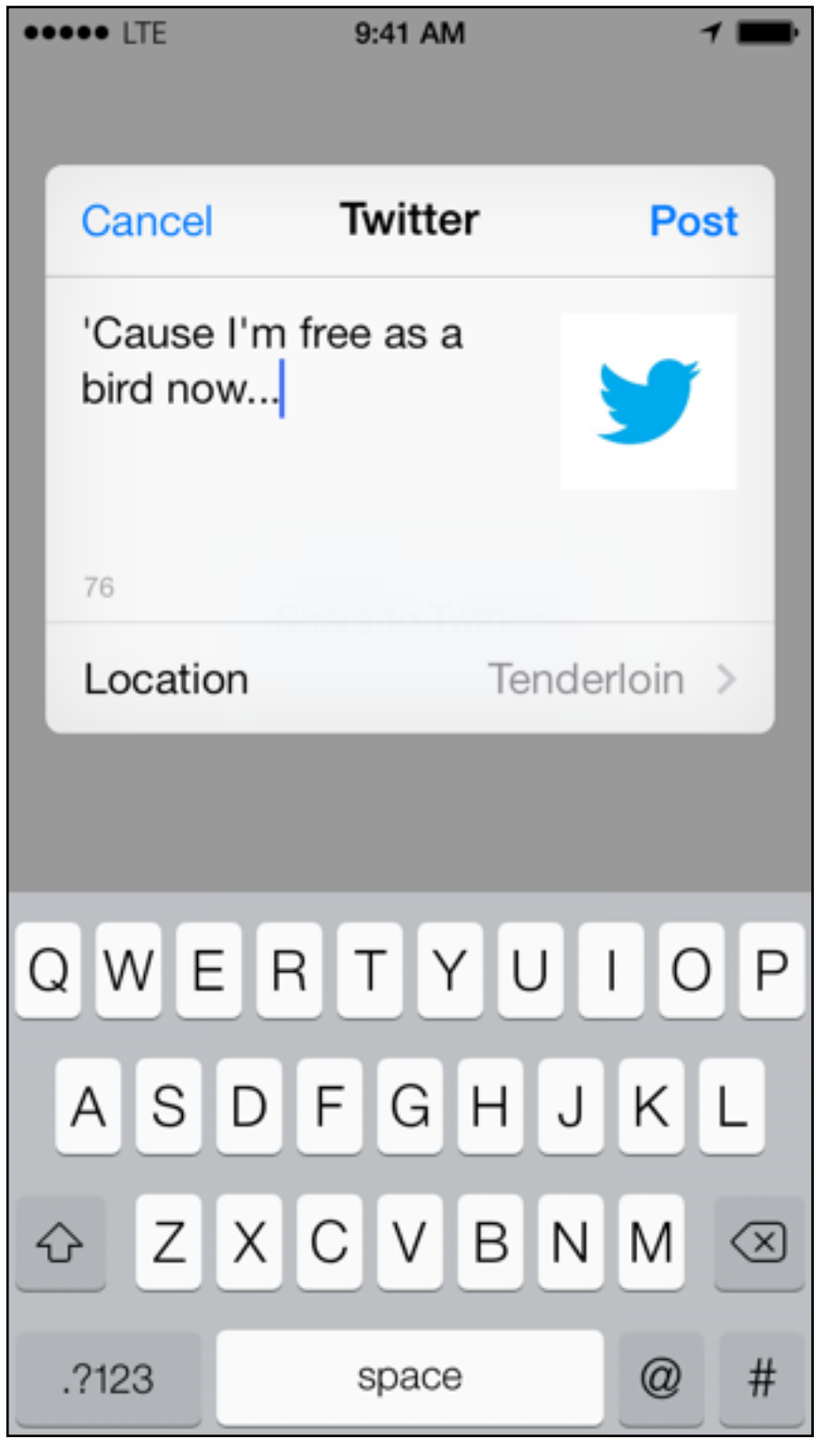
1.7 App Only

2. Twitter API

3. STTwitter

Tweet Sheet

- most simple use case
- when your user want to share contents
- compose a tweet using Setting's account



```objc
// check Twitter availablility
if ([SLComposeViewController isAvailableForServiceType:SLServiceTypeTwitter] == NO)
    return;

// instantiate composer
SLComposeViewController *composeVC =
[SLComposeViewController composeViewControllerForServiceType:SLServiceTypeTwitter];

// add contents
[composeVC setInitialText:@"Test"];
[composeVC addImage:myImage];
[composeVC addURL:myURL];

// set completion handler
[composeVC setCompletionHandler:^(SLComposeViewControllerResult result) {
    if(result == SLComposeViewControllerResultCancelled)
        NSLog(@"post cancelled");
    if(result == SLComposeViewControllerResultDone)
        NSLog(@"post successful");
}];

// present tweet sheet
[self presentViewController:composeVC animated:YES completion:nil];
```

1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

1.6 Reverse Auth

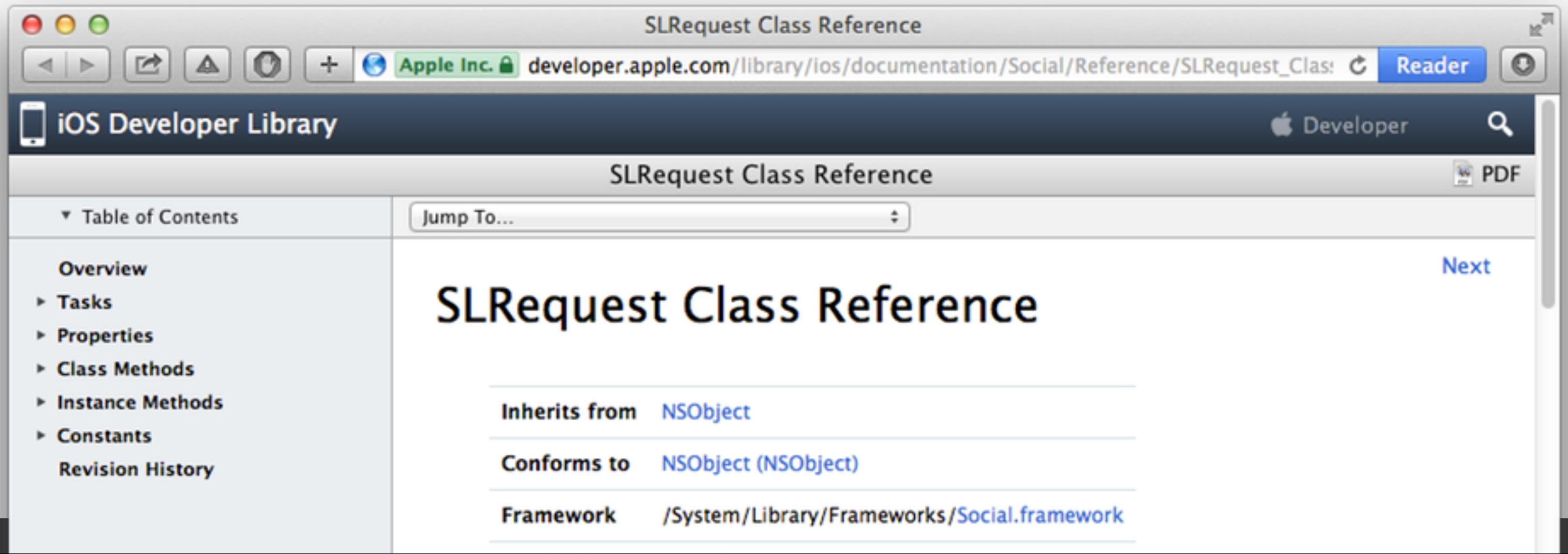
1.7 App Only

2. Twitter API

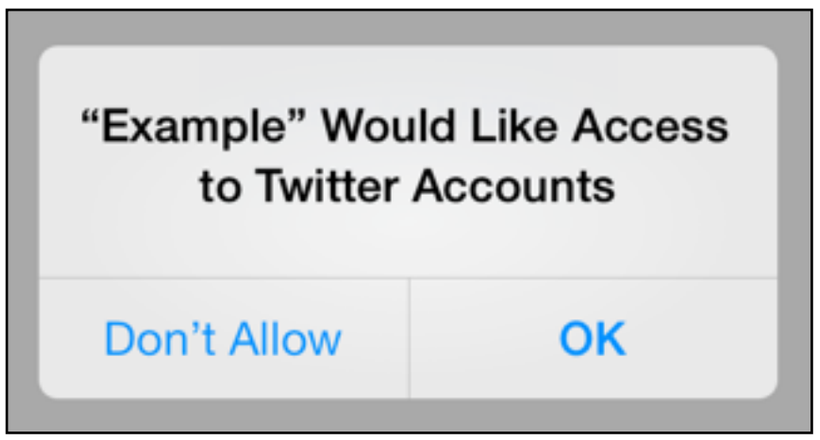
3. STTwitter

SLRequest

- access the full Twitter API
- create / customize a SLRequest instance
- access the endpoint you need



```objc
// 1. check that the user has a local Twitter account
if([SLComposeViewController isAvailableForServiceType:SLServiceTypeTwitter] == NO) return;

ACAccountStore *accountStore = [[ACAccountStore alloc] init];
ACAccountType *twitterAccountType =
    [accountStore accountTypeWithAccountTypeIdentifier:ACAccountTypeIdentifierTwitter];

// 2. ask the user permission to access his account
[accountStore requestAccessToAccountsWithType:twitterAccountType
                                      options:nil
                                   completion:^(BOOL granted, NSError *error) {

    if(granted == NO) {
        NSLog(@"-- error: %@", [error localizedDescription]);
        return;
    };

    // create and execute a request (next slide)
}];
```



```objc
// 3. create a request
NSURL *url = [NSURL URLWithString:@"https://api.twitter.com/1.1/statuses/user_timeline.json"];

SLRequest *request = [SLRequest requestForServiceType:SLServiceTypeTwitter
                                        requestMethod:SLRequestMethodGET
                                                  URL:url
                                           parameters:@{@"screen_name":@"nst021"}];

// 4. attach an account to the request
NSArray *twitterAccounts = [accountStore accountsWithAccountType:twitterAccountType];
[request setAccount:[twitterAccounts lastObject]];

// 5. execute the request
[request performRequestWithHandler:^(NSData *responseData,
                                     NSHTTPURLResponse *urlResponse,
                                     NSError *error) {

   NSError *jsonError = nil;
   NSDictionary *json = [NSJSONSerialization JSONObjectWithData:responseData
                                                        options:NSJSONReadingAllowFragments
                                                          error:&jsonError];

  // caution we're on an arbitrary queue
  [[NSOperationQueue mainQueue] addOperationWithBlock:^{
    NSLog(@"-- json: %@", json);
  }];

}];
```

Why SLRequest?

- SLRequest signs requests with OAuth
- Signed with both:
  - `consumer_secret` (app.)
  - `access_token_secret` (user on app.)
- SLRequest instances signed by Accounts.frameworks, with consumer tokens from OS X / iOS

Typical OAuth Request

```
GET https://api.twitter.com/1.1/statuses/home_timeline.json

Authorization = OAuth
  # consumer key for Echofon
  oauth_consumer_key="yqoymTNrS9ZDGsBnlFhIuw",
  # unique, random string
  oauth_nonce="15D07944-3625-4945-820B-E58CC33C",
  # always “HMAC-SHA1”
  oauth_signature_method="HMAC-SHA1",
  # epoch timestamp of the request
  oauth_timestamp="1381053556",
  # always “1.0”
  oauth_version="1.0",
  # the access key
  oauth_token="1294332967-M8UHF8i99sOEumljptPQ5BgdC6fho5BM3dLXsUn",
  # HMAC_SHA1(request_parameters, signing_key)
  # where signing_key is consumer_secret&access_secret
  oauth_signature="HONipNo6dJsQoSGZljccDfliNnY%3D"
```

STTwitter

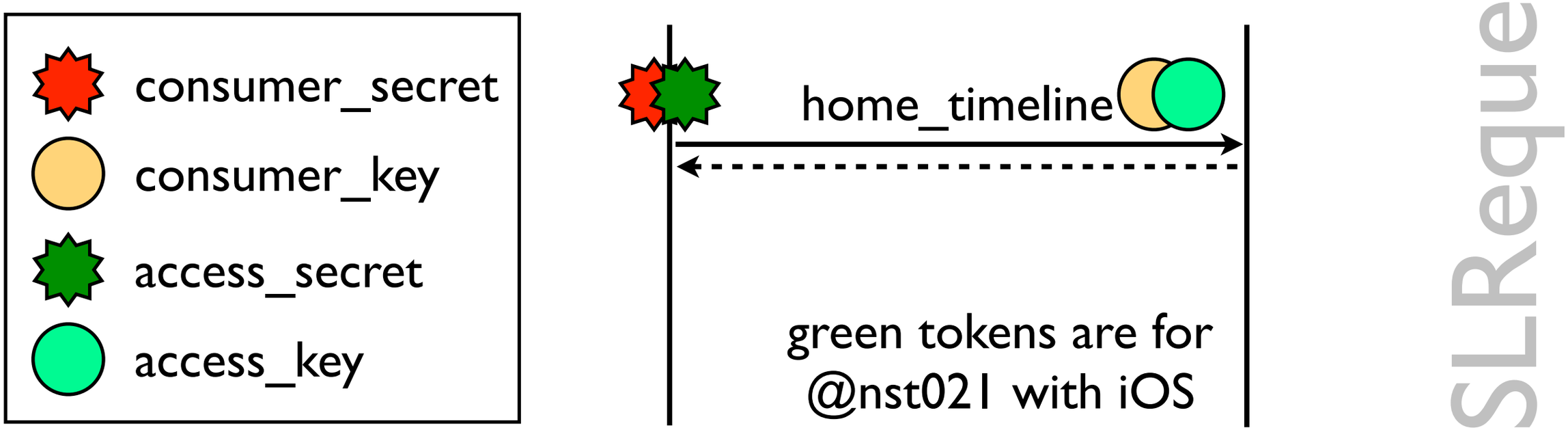
- Open-source Objective-C library
- Can sign OAuth requests
- Provides Obj-C method for each endpoint

https://github.com/nst/STTwitter

SLRequest with STTwitter

```
STTwitterAPI *t = [STTwitterAPI twitterAPIOSWithFirstAccount];

[t verifyCredentialsWithSuccessBlock:^(NSString *username) {

    [t getUserTimelineWithScreenName:@"nst021"
                               count:20
                        successBlock:^(NSArray *statuses) {
                            NSLog(@"-- %@", statuses);
                        } errorBlock:^(NSError *error) {
                            NSLog(@"-- %@", [error localizedDescription]);
                        }];

} errorBlock:^(NSError *error) {
    NSLog(@"-- %@", [error localizedDescription]);
}];
```

Notation



Now, how do we get
all these tokens in the first place?

TwitHunter

Mac OS X desktop Twitter client with innovative features, such as scoring rules and baysian filters.
http://seriot.ch

Organization

Information about the organization or company associated with your application. This information is optional.

Organization	seriot.ch
Organization website	http://seriot.ch

OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read, write, and direct messages About the application permission model
Consumer key	`YgP55HPexGdwxd6maVg1Q`
Consumer secret	[redacted]

1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

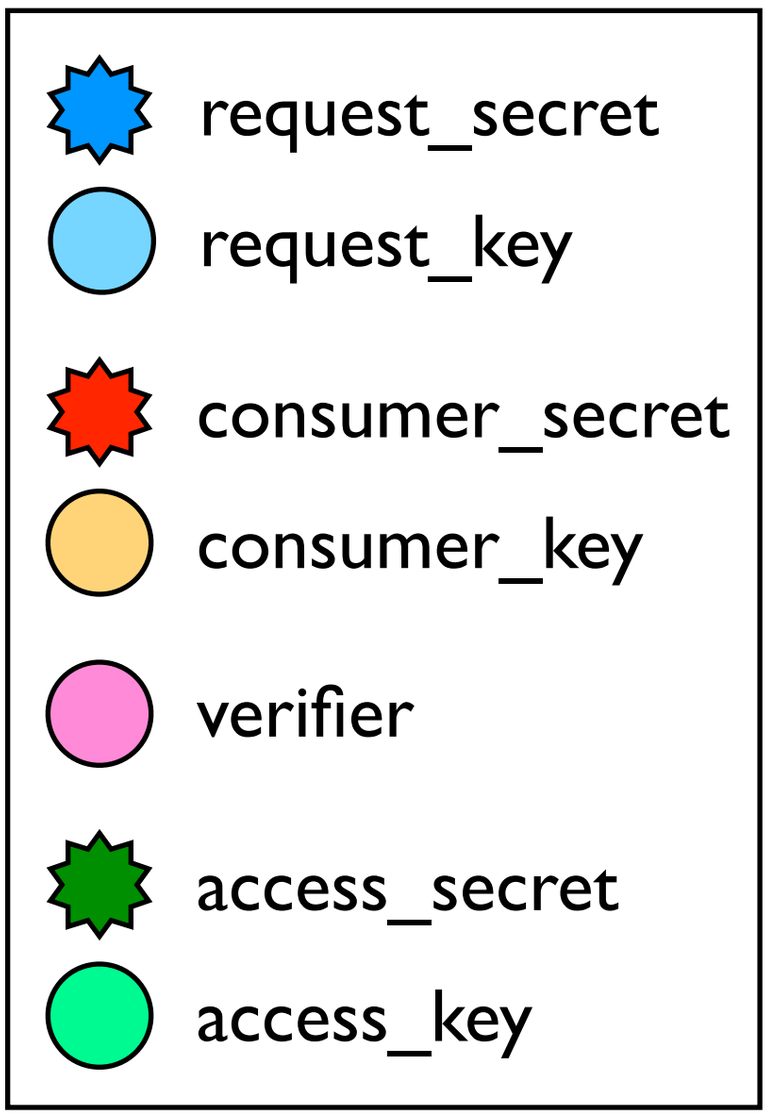
1.6 Reverse Auth

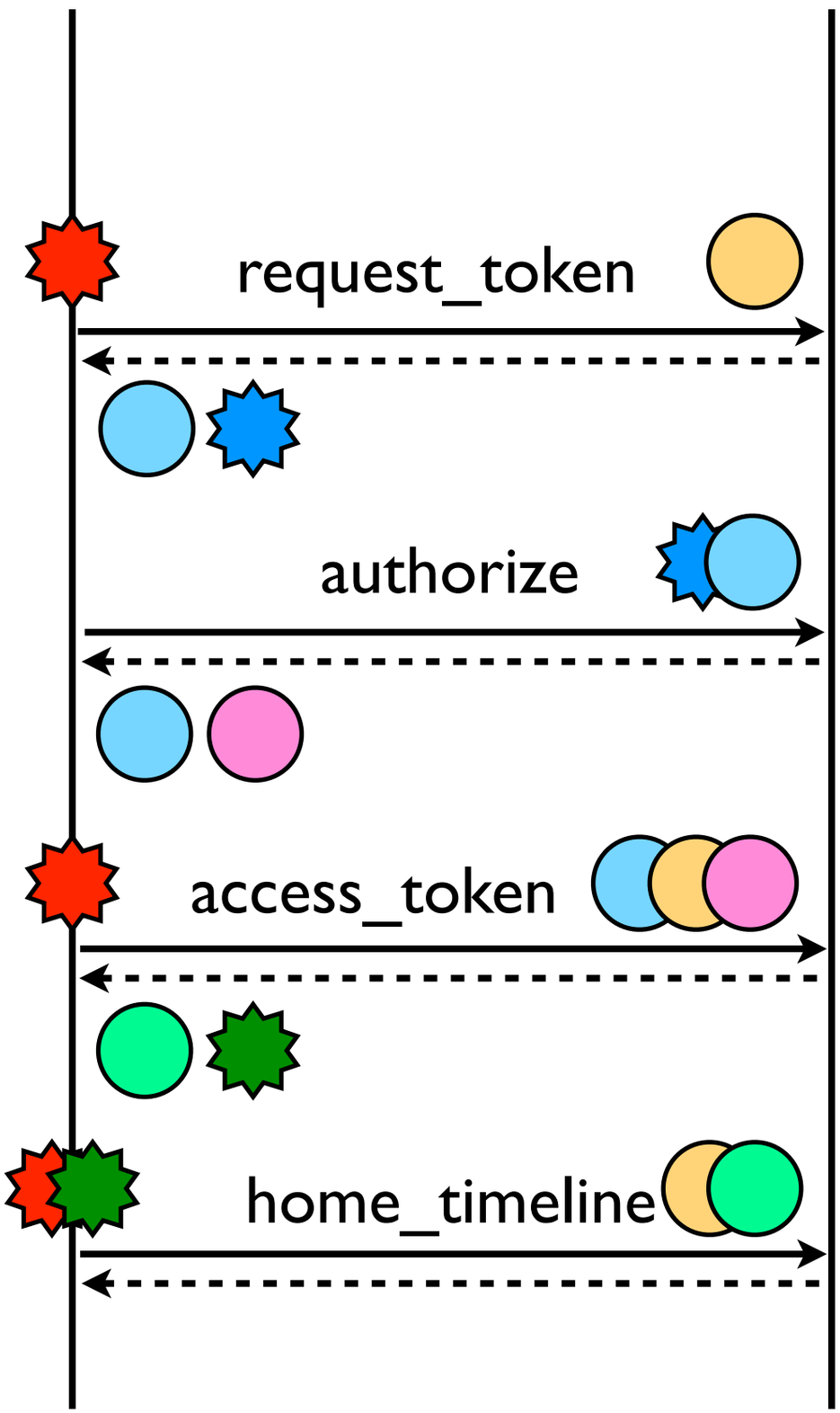
1.7 App Only

2. Twitter API

3. STTwitter

Get access tokens for App/User



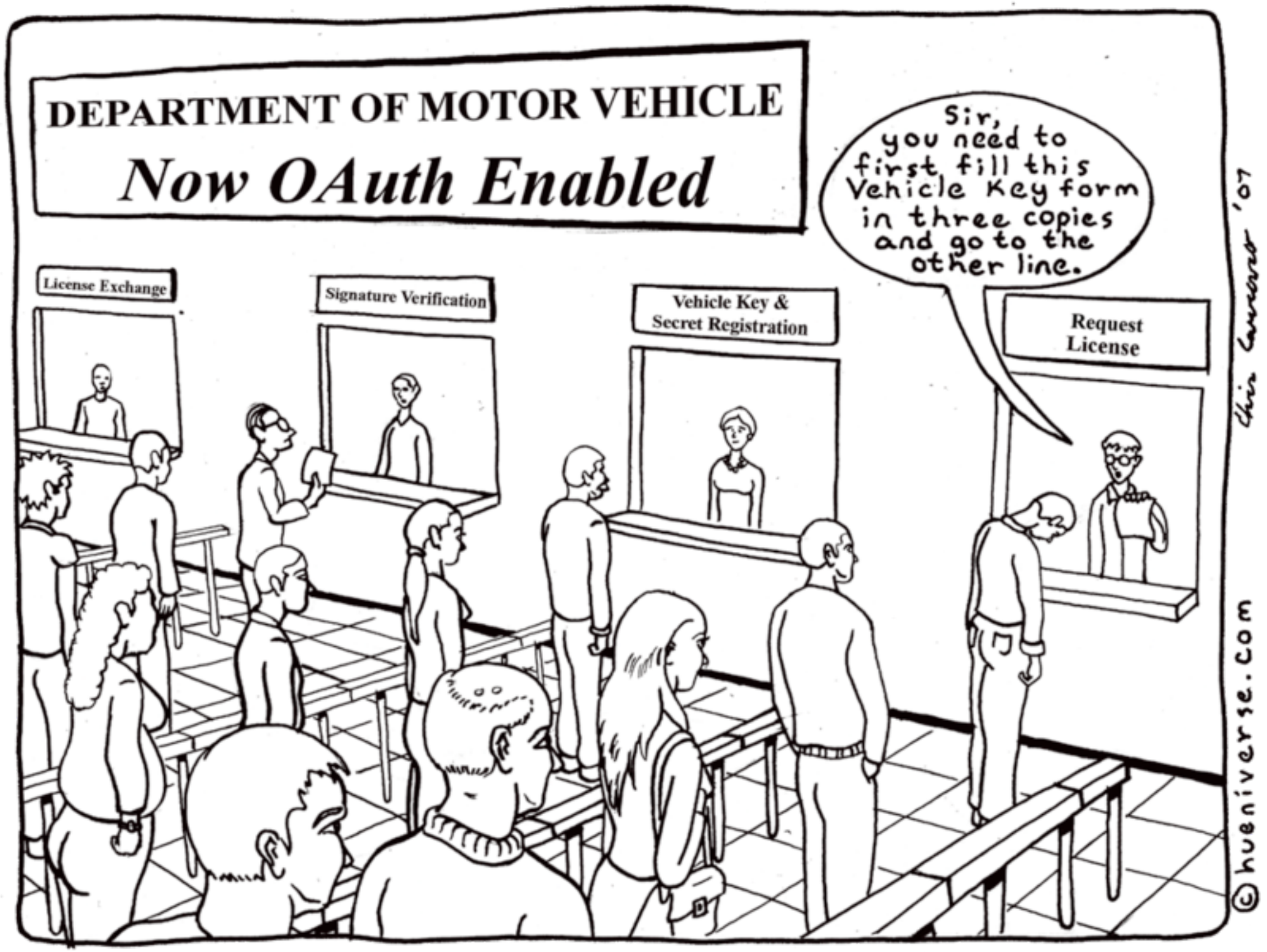


3 phases Auth.
Desktop

green tokens
valid for @nst021
on My.app

OAuth with PIN

```
STTwitterAPI *t = [STTwitterAPI twitterAPIWithOAuthConsumerKey:@""
                                                     consumerSecret:@""];

[t postTokenRequest:^(NSURL *url, NSString *oauthToken) {

    // open url, authorize application, get the pin

    [t postAccessTokenRequestWithPIN:pin
                        successBlock:^(NSString *oauthToken,
                                       NSString *oauthTokenSecret,
                                       NSString *userID,
                                       NSString *screenName) {

                        } errorBlock:^(NSError *error) {

                        }];

} oauthCallback:nil errorBlock:^(NSError *error) {

}];
```



http://hueniverse.com/2007/09/oauth-isnt-always-the-solution/

1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

1.6 Reverse Auth

1.7 App Only

2. Twitter API

3. STTwitter

XAuth

- Simplification of OAuth 3 steps auth.
- Exchange username and password for access_tokens

consumer_secret

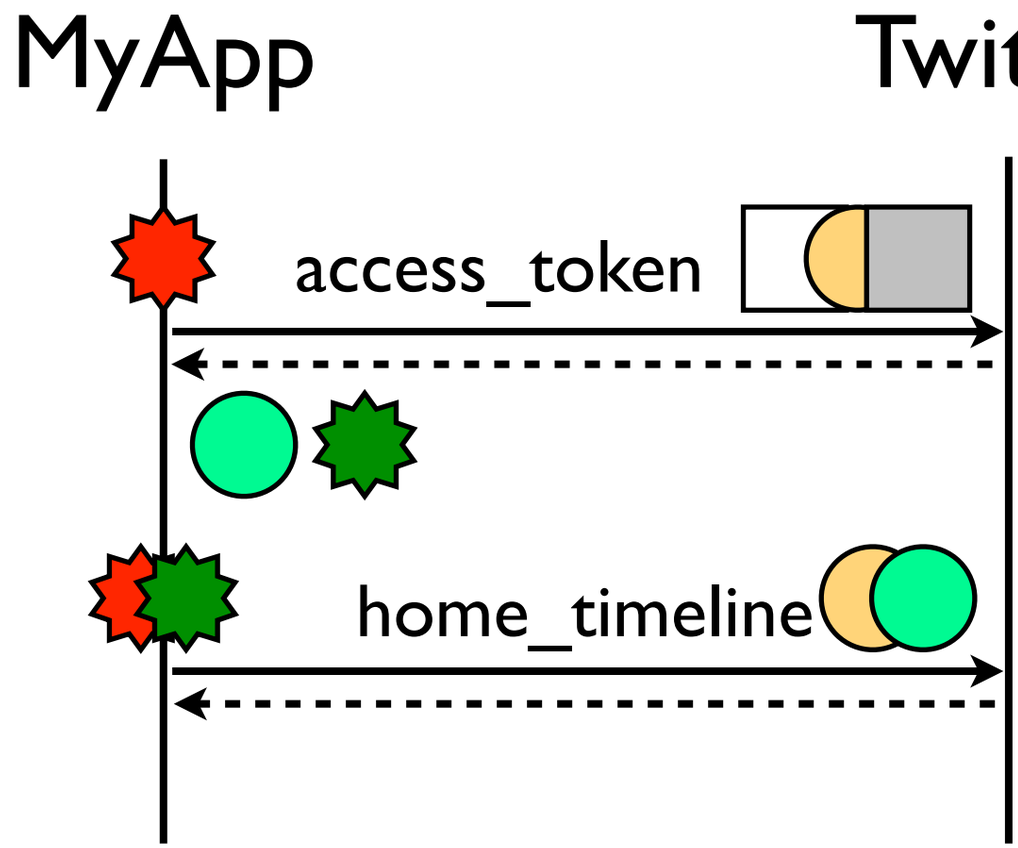
consumer_key

access_secret

access_key

username

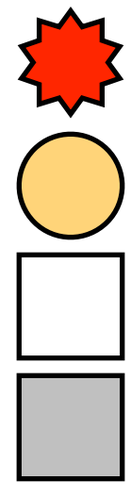
password



green tokens are for @nst021 with iOS

xAuth: Authentication phase

```objc
STTwitterAPI *t = [STTwitterAPI twitterAPIWithOAuthConsumerKey:@""
                                                consumerSecret:@""
                                                      username:@""
                                                      password:@""];

[t verifyCredentialsWithSuccessBlock:^(NSString *username) {

    NSLog(@"-- %@", t.oauthAccessToken);
    NSLog(@"-- %@", t.oauthAccessTokenSecret);

    // open url, authorize application, get the pin

} errorBlock:^(NSError *error) {
    NSLog(@"-- %@", [error localizedDescription]);
}];
```



xAuth

OAuth

View What links here

Updated on Wed, 2013-05-22 12:40

API version 1 API version 1.1

About xAuth

xAuth is still OAuth. You still need to master how to send signed requests to Twitter.

The xAuth process will only yield read-only or read-write access tokens. Direct message read access is not provided with xAuth. If your application requires access to a user's direct messages, you will need to use the full OAuth flow.

xAuth provides a way for desktop and mobile applications to exchange a username and password for an OAuth access token. Once the access token is retrieved, xAuth-enabled developers should dispose of the login and password corresponding to the user.

xAuth access is restricted to approved applications. If your application is a desktop or mobile application that has no other recourse but to utilize xAuth, send a detailed request to https://support.twitter.com/forms/platform. Include the name of your application, the consumer key, the application ID (if available), and a summary of how xAuth is best-suited for your application.

xAuth requires that you use header-based OAuth authentication against an SSL access token end point, using the POST HTTP method.

xAuth also supports Twitter's Login Verification feature.

Using xAuth

To use xAuth, first request access by filling out this form with plenty of details about your application and why xAuth is the best choice for it. xAuth is only granted in circumstances where the standard OAuth flow is not possible.

Related Questions

- How long do I have to wait when requesting xAuth?

Tags

- OAuth (178)
- Auth (31)
- xAuth (11)
- login verification (1)

1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

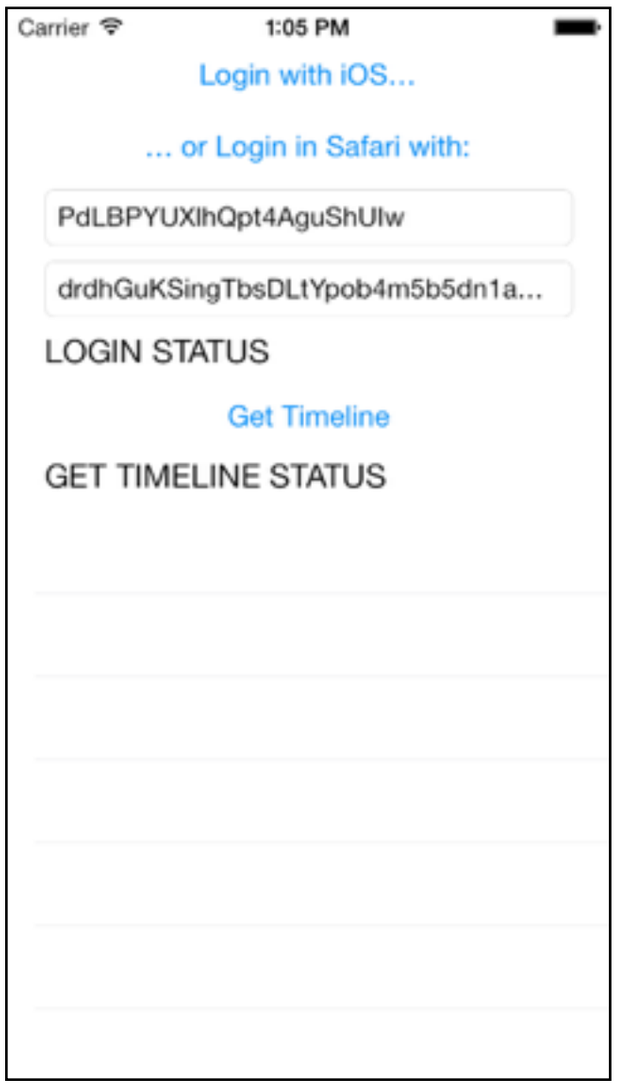
1.5 Safari

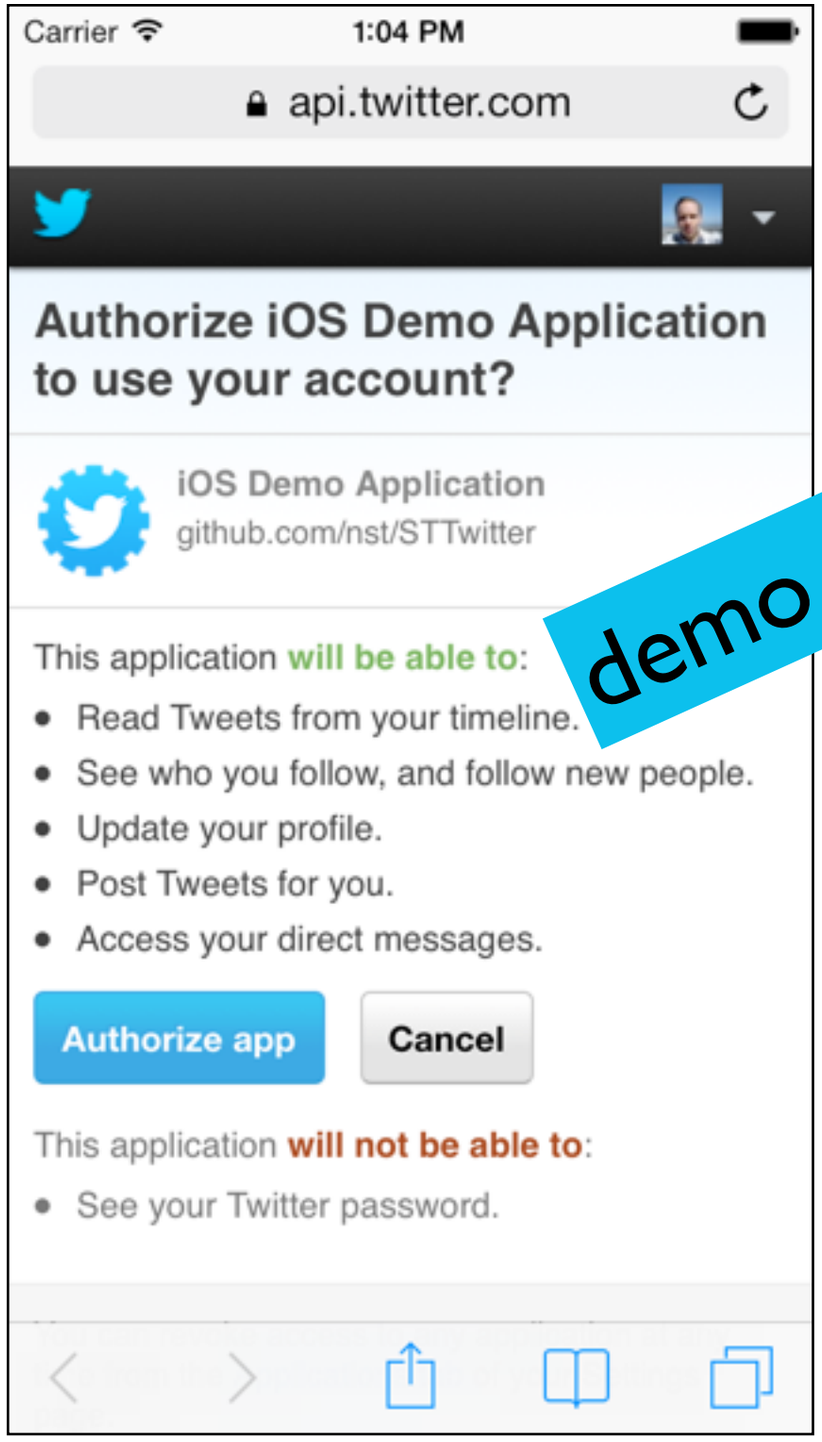
1.6 Reverse Auth

1.7 App Only

2. Twitter API

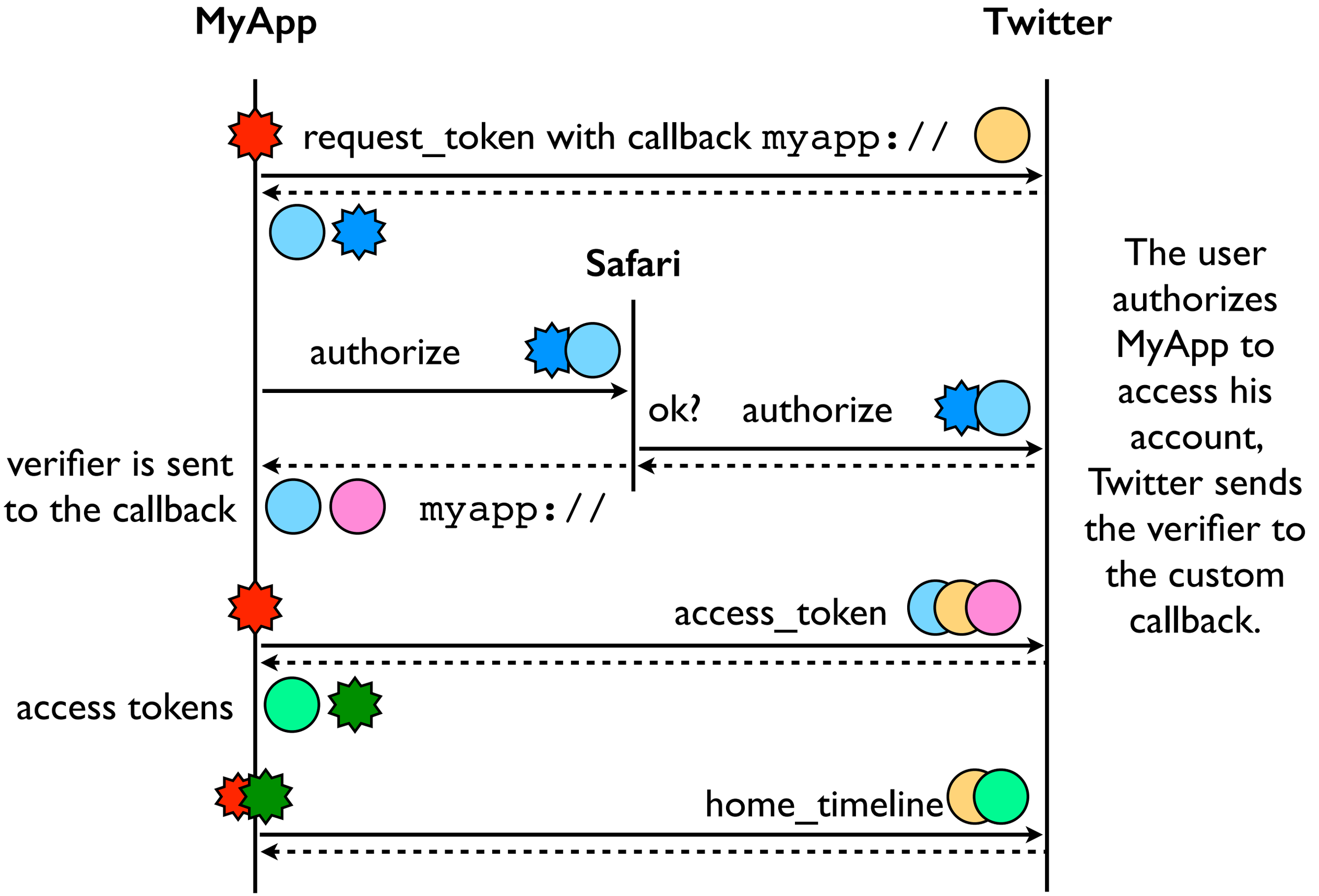
3. STTwitter



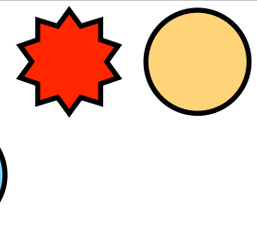


demo iOS

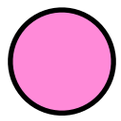
Benefit: trust! user doesn't give you her password.


MyApp
Twitter
request_token with callback myapp://
Safari
authorize
ok?
authorize
The user authorizes MyApp to access his account, Twitter sends the verifier to the custom callback.
verifier is sent to the callback
myapp://
access_token
access tokens
home_timeline

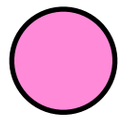
```
STTwitterAPI *t = [STTwitterAPI twitterAPIWithOAuthConsumerKey:@""
                                                    consumerSecret:@""];

[t postTokenRequest:^(NSURL *url, NSString *oauthToken) {

    [[UIApplication sharedApplication] openURL:url];

} oauthCallback:@"myapp://twitter_access_tokens/"

                errorBlock:^(NSError *error) {

                }];
```



```
- (BOOL)application:(UIApplication *)application
            openURL:(NSURL *)url
  sourceApplication:(NSString *)sourceApplication
         annotation:(id)annotation {

    if ([[url scheme] isEqualToString:@"myapp"] == NO) return NO;

    NSString *query = [url query];

    // read 'oauth_verifier' in query

    return YES;
}
```



```
[t postAccessTokenRequestWithPIN:verifier successBlock:^(NSString *oauthToken,
                                                         NSString *oauthTokenSecret,
                                                         NSString *userID,
                                                         NSString *screenName) {

    NSLog(@"-- screenName: %@", screenName);

} errorBlock:^(NSError *error) {

}];
```



1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

1.6 Reverse Auth

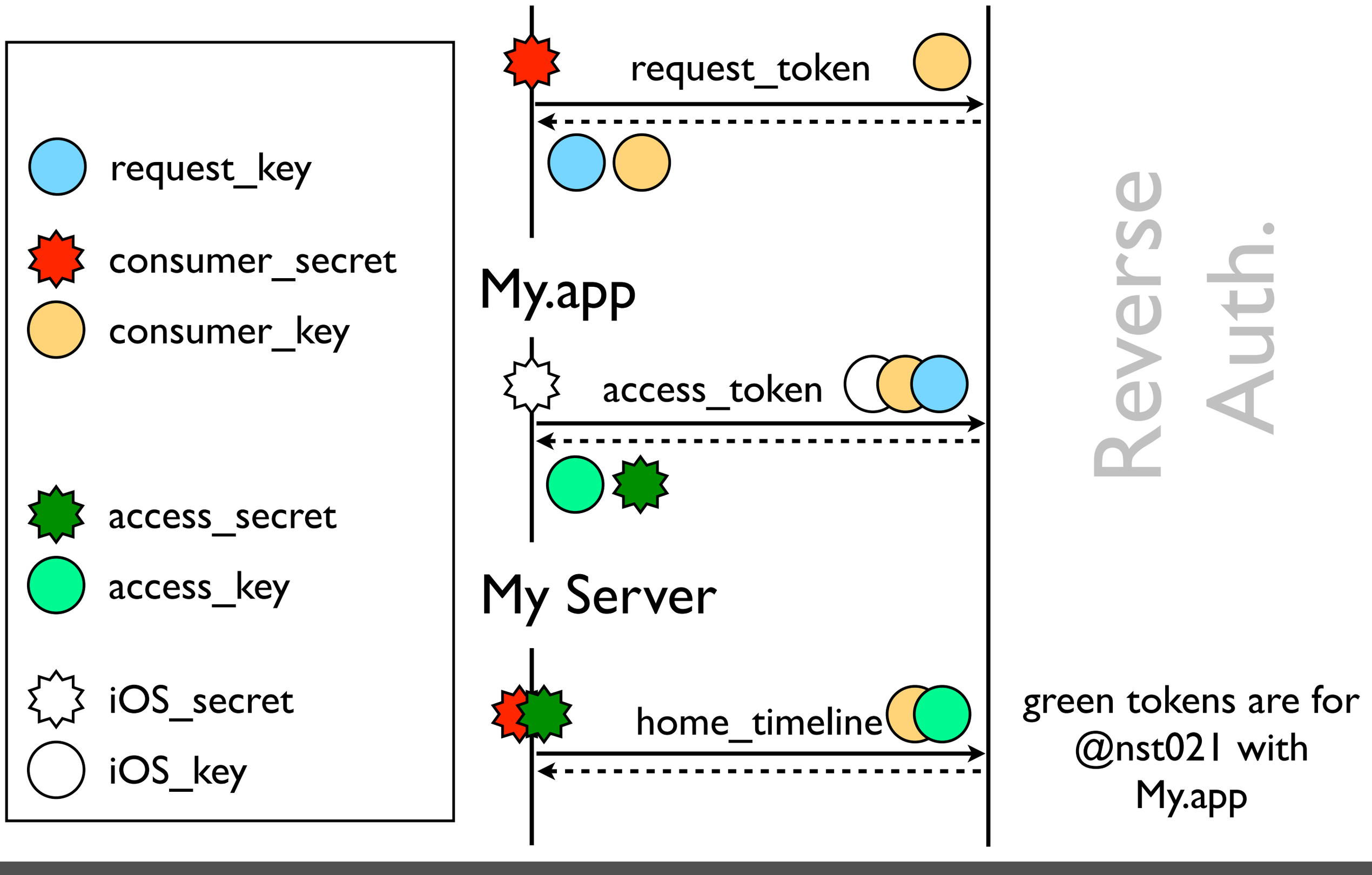
1.7 App Only

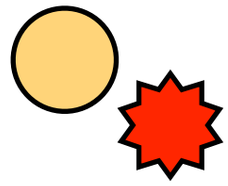
2. Twitter API

3. STTwitter

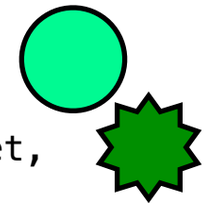
Reverse Auth

- Use case: U on A1 (iOS) => U/A1 allows A2 to get access tokens for U/ A2
- Benefits: don't ship CS, identify user with her Twitter account, access user's account from your server, access direct messages


My App or My Server
Twitter
request_key
consumer_secret
consumer_key
access_secret
access_key
iOS_secret
iOS_key
request_token
My.app
access_token
My Server
home_timeline
green tokens are for
@nst021 with
My.app



```objc
STTwitterAPI *twitter = [STTwitterAPI twitterAPIWithOAuthConsumerKey:@""
                                                      consumerSecret:@""];

[twitter postReverseOAuthTokenRequest:^(NSString *authenticationHeader) {

    STTwitterAPI *twitterAPIOS = [STTwitterAPI twitterAPIOSWithFirstAccount];

    [twitterAPIOS verifyCredentialsWithSuccessBlock:^(NSString *username) {

        [twitterAPIOS postReverseAuthAccessTokenWithAuthenticationHeader:authenticationHeader
                                                            successBlock:^(NSString *oAuthToken,
                                                                           NSString *oAuthTokenSecret,
                                                                           NSString *userID,
                                                                           NSString *screenName) {

            // use the tokens...

        } errorBlock:^(NSError *error) {
            // ...
        }];

    } errorBlock:^(NSError *error) {
        // ...
    }];

}];
```



1. Access Twitter API

1.1 Tweet Sheet

1.2 SLRequest

1.3 OAuth

1.4 XAuth

1.5 Safari

1.6 Reverse Auth

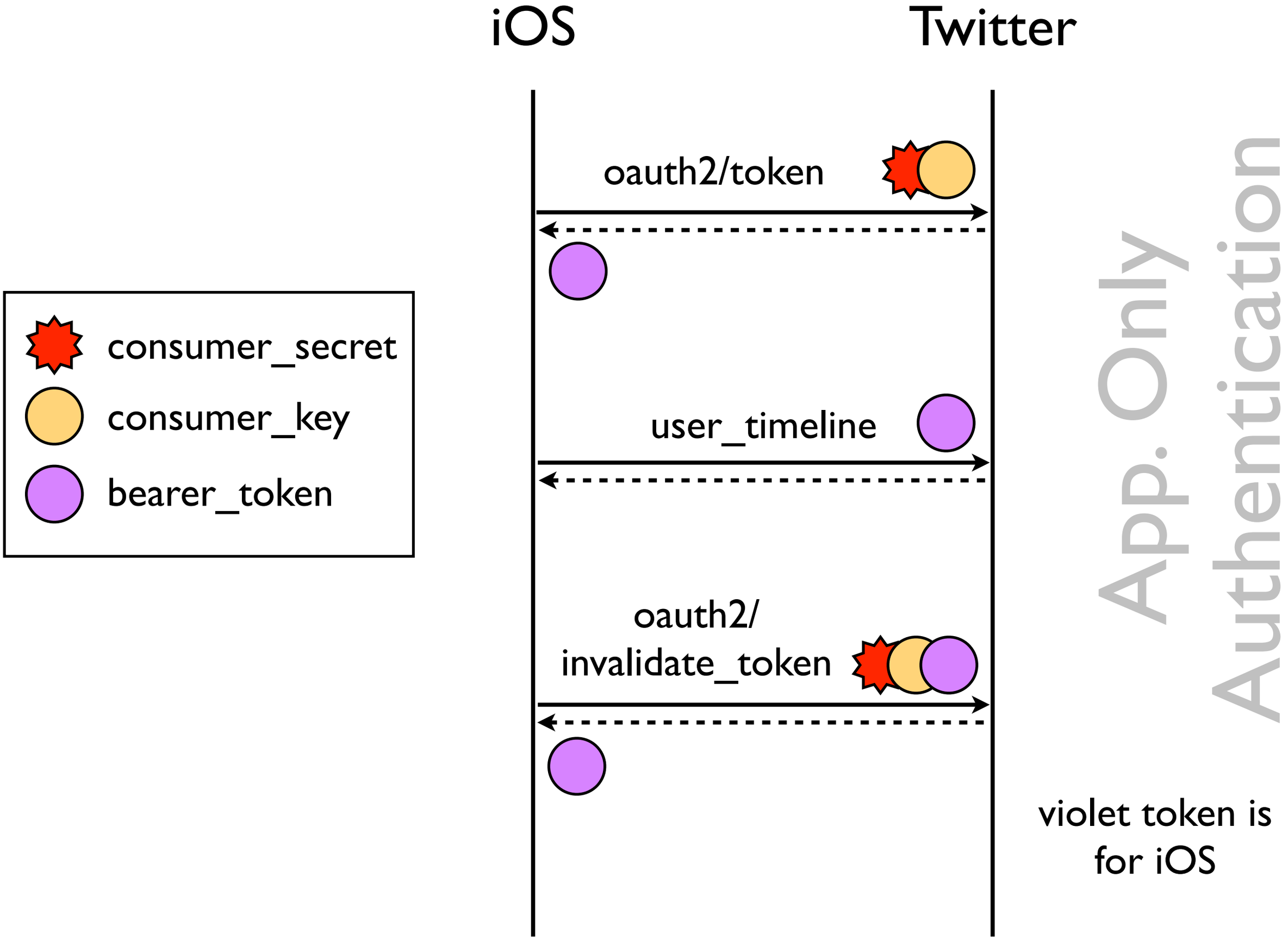
1.7 App Only

2. Twitter API

3. STTwitter

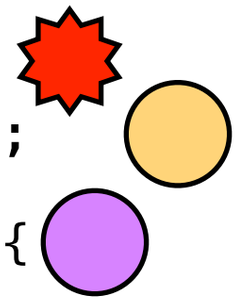
App Only

- Use case: **no user's context**
- Eg. display your company's timeline in your iOS application
- Benefit: you can use Twitter API without users having a Twitter account


iOS
Twitter
oauth2/token
user_timeline
oauth2/
invalidate_token
consumer_secret
consumer_key
bearer_token
App. Only
Authentication
violet token is
for iOS

demo OS X

App Only



```
STTwitterAPI *t = [STTwitterAPI twitterAPIAppOnlyWithConsumerKey:@""
                                                  consumerSecret:@""];

[_twitter verifyCredentialsWithSuccessBlock:^(NSString *bearerToken) {

    // use API without user's context

    // eg. display timeline for @BarackObama

    // but cannot post or favorite a tweet because no user context

} errorBlock:^(NSError *error) {

    //

}];
```

1. Access Twitter API

2. Twitter API

3. STTwitter

Twitter API

https://dev.twitter.com/docs/api/1.1

Timeline,
Retweet,
Search,
Direct Messages
…

Follow / Unfollow
Favorites
Lists
Trends
…

Rate Limits in Headers

```
{
    "x-rate-limit-limit" = 15;
    "x-rate-limit-remaining" = 14;
    "x-rate-limit-reset" = 1381054457;
}
```

API Changes

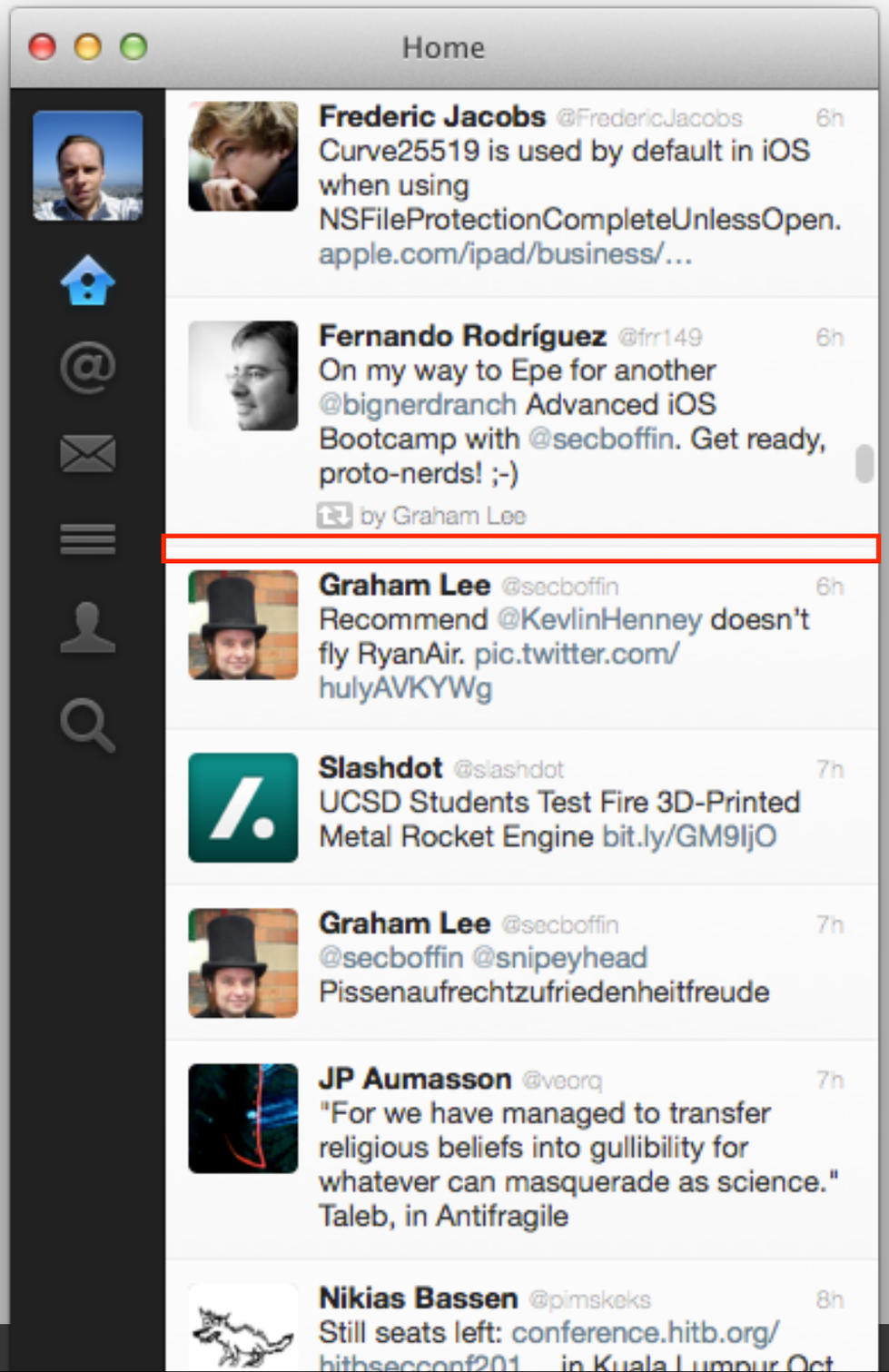
https://dev.twitter.com/blog
https://dev.twitter.com/docs/recent
https://dev.twitter.com/calendar
https://twitter.com/twitterapi

Undocumented API?

```
$ strings Twitter
...
activity/about_me.json
activity/by_friends.json
conversation/show.json
discover/highlight.json
discover/universal.json
statuses/:id/activity/summary.json
statuses/media_timeline.json
statuses/mentions_timeline.json
timeline/home.json
trends/available.json
users/recommendations.json
...
```

Promoted Contents

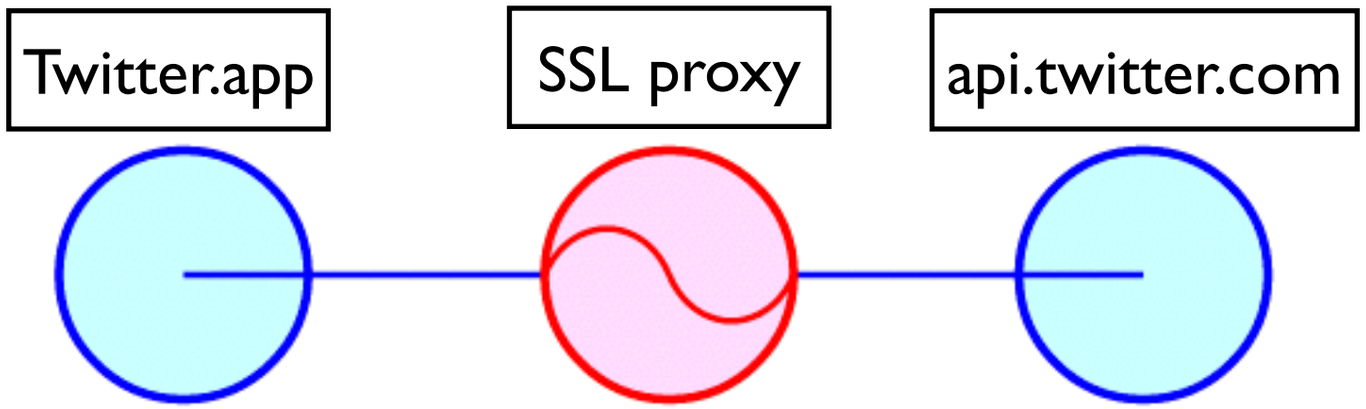
Ad



No Ad

Looking for Ads

- Still no clues about promoted contents. Let's start our favorite SSL proxy!

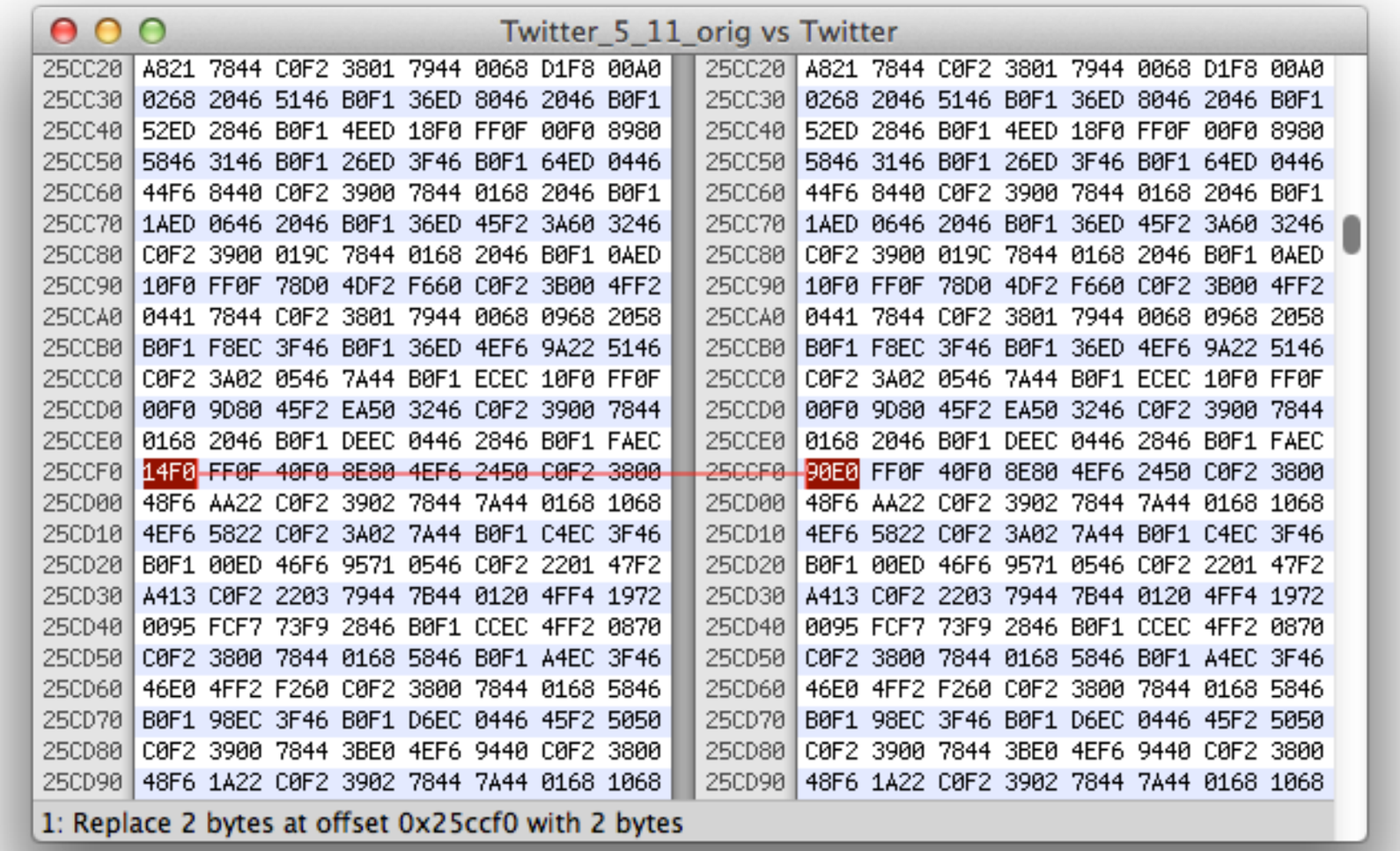


- Does't work because of certificate pinning.
- Binary patching FTW!

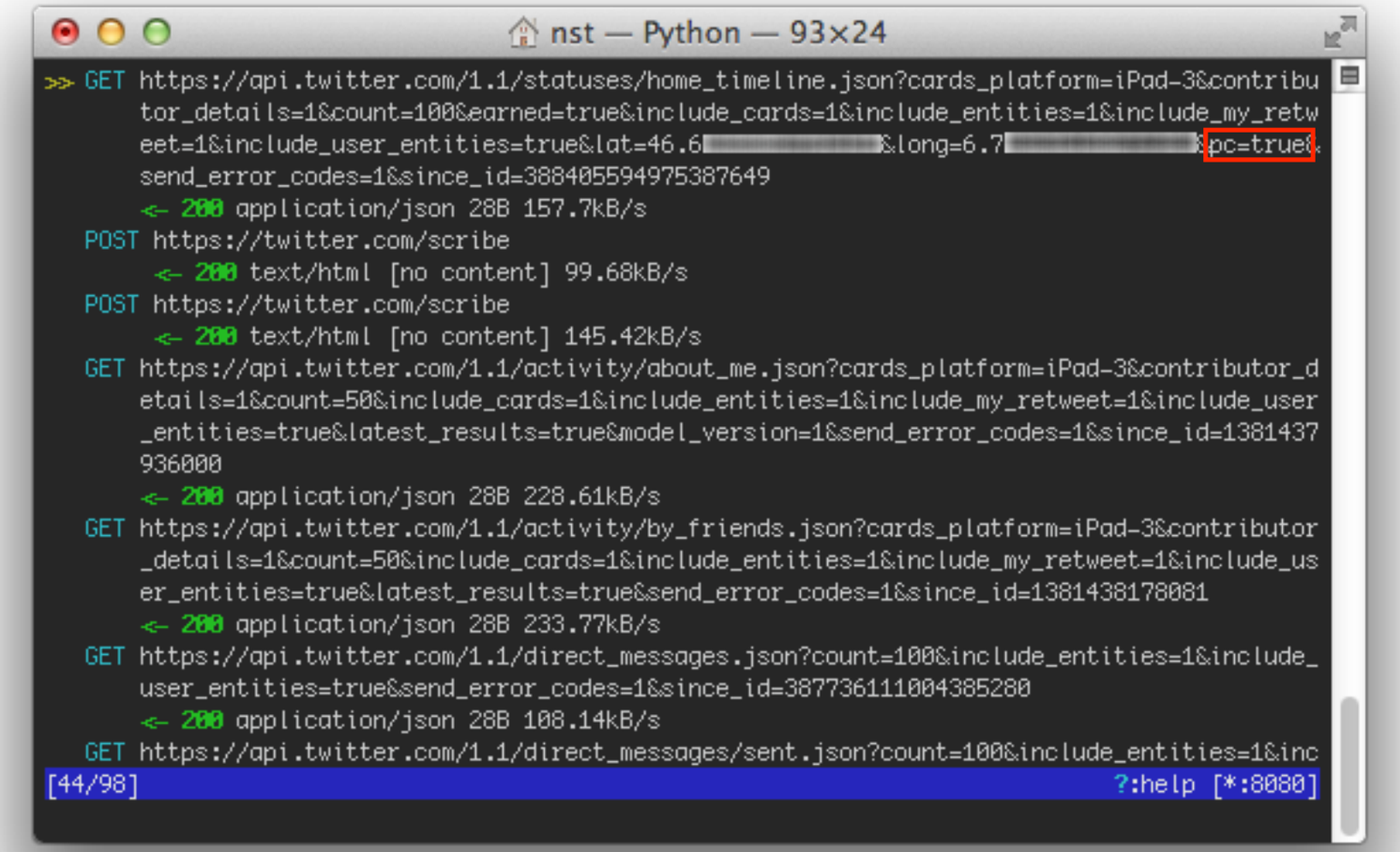
```
-[ABHTTPRequest connection:willSendRequestForAuthenticationChallenge:]
  ...
  0x00260cd4 45F2EA50    movw     r0, #0x55ea
  0x00260cd8 3246        mov      r2, r6
  0x00260cda C0F23900    movt     r0, #0x39
  0x00260cde 7844        add      r0, pc          ; 0x5f62cc
  0x00260ce0 0168        ldr      r1, [r0]        ; @selector(_isPinnedCertificateChain:)
  0x00260ce2 2046        mov      r0, r4
  0x00260ce4 B0F1DEEC    blx      imp___picsymbolstub4__objc_msgSend

  ; BOOL isPinned = [self _isPinnedCertificateChain:object];

  0x00260ce8 0446        mov      r4, r0
  0x00260cea 2846        mov      r0, r5
  0x00260cec B0F1FAEC    blx      imp___picsymbolstub4__objc_release

- 0x00260cf0 14F0FF0F    tst.w    r4, #0xff       ; Z = (r4 & 0xff) == 0 ; Z = (r4 == 0)
+ 0x00260cf0 90E0FF0F    b        0x260e14        ; jump to happy path (FF0F is unused)
  0x00260cf4 40F08E80    bne.w    0x260e14        ; never reached
  ...
```

```
; if(isPinned) {                    ; goto 0x260e14;
;     goto 0x260e14;                ;
; } else {                    =>    ;
;     return error;                 ; return error;
; }                                 ;
```

My 2 Bytes


Twitter_5_11_orig vs Twitter
25CC20 A821 7844 C0F2 3801 7944 0068 D1F8 00A0 | 25CC20 A821 7844 C0F2 3801 7944 0068 D1F8 00A0
25CC30 0268 2046 5146 B0F1 36ED 8046 2046 B0F1 | 25CC30 0268 2046 5146 B0F1 36ED 8046 2046 B0F1
25CC40 52ED 2846 B0F1 4EED 18F0 FF0F 00F0 8980 | 25CC40 52ED 2846 B0F1 4EED 18F0 FF0F 00F0 8980
25CC50 5846 3146 B0F1 26ED 3F46 B0F1 64ED 0446 | 25CC50 5846 3146 B0F1 26ED 3F46 B0F1 64ED 0446
25CC60 44F6 8440 C0F2 3900 7844 0168 2046 B0F1 | 25CC60 44F6 8440 C0F2 3900 7844 0168 2046 B0F1
25CC70 1AED 0646 2046 B0F1 36ED 45F2 3A60 3246 | 25CC70 1AED 0646 2046 B0F1 36ED 45F2 3A60 3246
25CC80 C0F2 3900 019C 7844 0168 2046 B0F1 0AED | 25CC80 C0F2 3900 019C 7844 0168 2046 B0F1 0AED
25CC90 10F0 FF0F 78D0 4DF2 F660 C0F2 3B00 4FF2 | 25CC90 10F0 FF0F 78D0 4DF2 F660 C0F2 3B00 4FF2
25CCA0 0441 7844 C0F2 3801 7944 0068 0968 2058 | 25CCA0 0441 7844 C0F2 3801 7944 0068 0968 2058
25CCB0 B0F1 F8EC 3F46 B0F1 36ED 4EF6 9A22 5146 | 25CCB0 B0F1 F8EC 3F46 B0F1 36ED 4EF6 9A22 5146
25CCC0 C0F2 3A02 0546 7A44 B0F1 ECEC 10F0 FF0F | 25CCC0 C0F2 3A02 0546 7A44 B0F1 ECEC 10F0 FF0F
25CCD0 00F0 9D80 45F2 EA50 3246 C0F2 3900 7844 | 25CCD0 00F0 9D80 45F2 EA50 3246 C0F2 3900 7844
25CCE0 0168 2046 B0F1 DEEC 0446 2846 B0F1 FAEC | 25CCE0 0168 2046 B0F1 DEEC 0446 2846 B0F1 FAEC
25CCF0 14F0 FF0F 40F0 8E80 4EF6 2450 C0F2 3800 | 25CCF0 90E0 FF0F 40F0 8E80 4EF6 2450 C0F2 3800
25CD00 48F6 AA22 C0F2 3902 7844 7A44 0168 1068 | 25CD00 48F6 AA22 C0F2 3902 7844 7A44 0168 1068
25CD10 4EF6 5822 C0F2 3A02 7A44 B0F1 C4EC 3F46 | 25CD10 4EF6 5822 C0F2 3A02 7A44 B0F1 C4EC 3F46
25CD20 B0F1 00ED 46F6 9571 0546 C0F2 2201 47F2 | 25CD20 B0F1 00ED 46F6 9571 0546 C0F2 2201 47F2
25CD30 A413 C0F2 2203 7944 7B44 0120 4FF4 1972 | 25CD30 A413 C0F2 2203 7944 7B44 0120 4FF4 1972
25CD40 0095 FCF7 73F9 2846 B0F1 CCEC 4FF2 0870 | 25CD40 0095 FCF7 73F9 2846 B0F1 CCEC 4FF2 0870
25CD50 C0F2 3800 7844 0168 5846 B0F1 A4EC 3F46 | 25CD50 C0F2 3800 7844 0168 5846 B0F1 A4EC 3F46
25CD60 46E0 4FF2 F260 C0F2 3800 7844 0168 5846 | 25CD60 46E0 4FF2 F260 C0F2 3800 7844 0168 5846
25CD70 B0F1 98EC 3F46 B0F1 D6EC 0446 45F2 5050 | 25CD70 B0F1 98EC 3F46 B0F1 D6EC 0446 45F2 5050
25CD80 C0F2 3900 7844 3BE0 4EF6 9440 C0F2 3800 | 25CD80 C0F2 3900 7844 3BE0 4EF6 9440 C0F2 3800
25CD90 48F6 1A22 C0F2 3902 7844 7A44 0168 1068 | 25CD90 48F6 1A22 C0F2 3902 7844 7A44 0168 1068
1: Replace 2 bytes at offset 0x25ccf0 with 2 bytes

Promoted Contents Secret



&pc=true

1. Using Twitter API
2. Twitter API
3. **STTwitter**

PUBLIC nst / STTwitter

Unwatch 20 | Unstar 235 | Fork 33

A comprehensive Objective-C library for Twitter REST API 1.1 — Edit

439 commits | 1 branch | 5 releases | 6 contributors

branch: master | STTwitter /

typo

nst authored an hour ago — latest commit fdf67b70e8

Art	updated screenshot	4 days ago
STTwitter	updated STHTTPRequest	a day ago
Tests	show request headers in OS X demo app	5 days ago
demo_ios	typo	an hour ago
demo_osx	updated screenshot	4 days ago
LICENSE.txt	improved README, preparing tag 0.0.5	11 days ago
README.md	updated timestamp of OS X demo application archive	4 hours ago
STTwitter.podspec	fixed syntax error	10 days ago

Code
Issues 0
Pull Requests 0
Wiki
Pulse
Graphs
Network
Settings

HTTPS clone URL

https://github.com/n

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

README.md

STTwitter

A comprehensive Objective-C library for Twitter REST API 1.1

- -getMentionsTimelineSinceID:count:successBlock:errorBlock:
- -getStatusesUserTimelineForUserID:screenName:sinceID:count:maxID:trimUser:excludeReplies:contributorDetails:includeRetweets:successBlock:errorBlock:
- -getUserTimelineWithScreenName:sinceID:maxID:count:successBlock:errorBlock:
- -getUserTimelineWithScreenName:count:successBlock:errorBlock:
- -getUserTimelineWithScreenName:successBlock:errorBlock:
- -getStatusesHomeTimelineWithCount:sinceID:maxID:trimUser:excludeReplies:contributorDetails:includeEntities:successBlock:errorBlock:
- -getHomeTimelineSinceID:count:successBlock:errorBlock:
- -getStatusesRetweetsOfMeWithCount:sinceID:maxID:trimUser:includeEntities:includeUserEntities:successBlock:errorBlock:
- -getStatusesRetweetsOfMeWithSuccessBlock:errorBlock:

Tweets

- -getStatusesRetweetsForID:count:trimUser:successBlock:errorBlock:
- -getStatusesShowID:trimUser:includeMyRetweet:includeEntities:successBlock:errorBlock:
- -postStatusesDestroy:trimUser:successBlock:errorBlock:
- -postStatusUpdate:inReplyToStatusID:latitude:longitude:placeID:displayCoordinates:trimUser:successBlock:errorBlock:
- -postStatusRetweetWithID:trimUser:successBlock:errorBlock:
- -postStatusRetweetWithID:successBlock:errorBlock:
- -postStatusUpdate:mediaDataArray:possiblySensitive:inReplyToStatusID:latitude:longitude:placeID:displayCoordinates:successBlock:errorBlock:
- -postStatusUpdate:inReplyToStatusID:mediaURL:placeID:latitude:longitude:successBlock:errorBlock:
- -getStatusesOEmbedForStatusID:urlString:maxWidth:hideMedia:hideThread:omitScript:align:related:lang:successBlock:errorBlock:
- -getStatusesRetweetersIDsForStatusID:cursor:successBlock:errorBlock:

Search

- -getSearchTweetsWithQuery:geocode:lang:locale:resultType:count:until:sinceID:maxID:includeEntities:callback:successBlock:errorBlock:
- -getSearchTweetsWithQuery:successBlock:errorBlock:

Streaming

- -postStatusesFilterUserIDs:keywordsToTrack:locationBoundingBoxes:delimited:stallWarnings:progressBlock:stallWarningBlock:errorBlock:
- -postStatusesFilterKeyword:progressBlock:errorBlock:
- -getStatusesSampleDelimited:stallWarnings:progressBlock:stallWarningBlock:errorBlock:
- -getStatusesFirehoseWithCount:delimited:stallWarnings:progressBlock:stallWarningBlock:errorBlock:
- -getUserStreamDelimited:stallWarnings:includeMessagesFromFollowedAccounts:includeReplies:keywordsToTrack:locationBoundingBoxes:progressBlock:stallWarningBlock:errorBlock:
- -getSiteStreamForUserIDs:delimited:stallWarnings:restrictToUserMessages:includeReplies:progressBlock:stallWarningBlock:errorBlock:

Direct Messages

- -getDirectMessagesSinceID:maxID:count:includeEntities:skipStatus:successBlock:errorBlock:
- -getDirectMessagesSinceID:count:successBlock:errorBlock:
- -getDirectMessagesSinceID:maxID:count:page:includeEntities:successBlock:errorBlock:
- -getDirectMessagesShowWithID:successBlock:errorBlock:
- -postDestroyDirectMessageWithID:includeEntities:successBlock:errorBlock:
- -postDirectMessage:to:successBlock:errorBlock:

Friends & Followers

- -getFriendshipNoRetweetsIDsWithSuccessBlock:errorBlock:
- -getFriendsIDsForUserID:orScreenName:cursor:count:successBlock:errorBlock:

UNDOCUMENTED APIs

- M -_getActivityAboutMeSinceID:count:includeCards:modelVersion:sendErrorCodes:contributorDetails:includeEntities:includeMyRetweet:successBlock:errorBlock:
- M -_getActivityByFriendsSinceID:count:contributorDetails:includeCards:includeEntities:includeMyRetweets:includeUserEntites:latestResults:sendErrorCodes:successBlock:errorBlock:
- M -_getStatusesActivitySummaryForStatusID:successBlock:errorBlock:
- M -_getConversationShowForStatusID:successBlock:errorBlock:
- M -_getDiscoverHighlightWithSuccessBlock:errorBlock:
- M -_getDiscoverUniversalWithSuccessBlock:errorBlock:
- M -_getMediaTimelineWithSuccessBlock:errorBlock:
- M -_getUsersRecommendationsWithSuccessBlock:errorBlock:
- M -_getTimelineHomeWithSuccessBlock:errorBlock:
- M -_getStatusesMentionsTimelineWithCount:contributorsDetails:includeEntities:includeMyRetweet:successBlock:errorBlock:
- M -_getTrendsAvailableWithSuccessBlock:errorBlock:
- M -_postUsersReportSpamForTweetID:reportAs:blockUser:successBlock:errorBlock:
- M -_postAccountGenerateWithADC:discoverableByEmail:email:geoEnabled:language:name:password:screenName:sendErrorCode:timeZone:successBlock:errorBlock:

Architecture

Your Application

STTwitterAPI

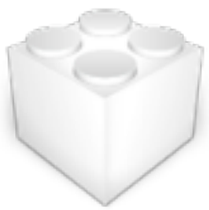
STTwitterProtocol

STTwitterOS

STTwitterOAuth

STTwitterAppOnly

STHTTPRequest



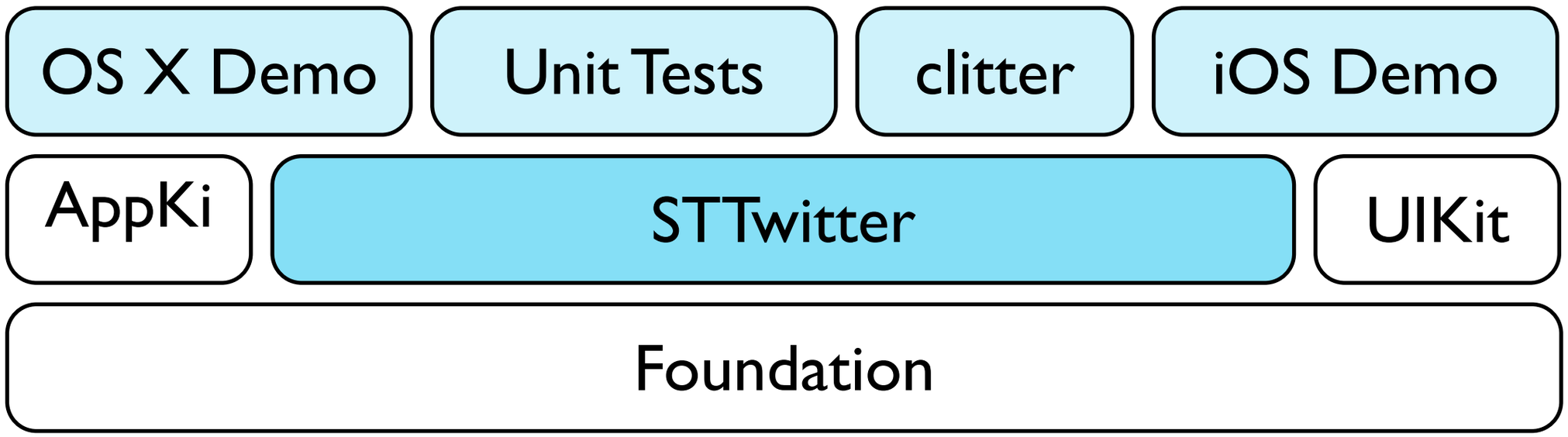
Account.framework
Social.framework

STHTTPRequest

```objc
STHTTPRequest *r = [STHTTPRequest requestWithURLString:@"http://twitter.com"];

r.completionBlock = ^(NSDictionary *headers, NSString *body) {
    // ...
};

r.errorBlock = ^(NSError *error) {
    // ...
};

[r startAsynchronous];
```

https://github.com/nst/STHTTPRequest

Subprojects



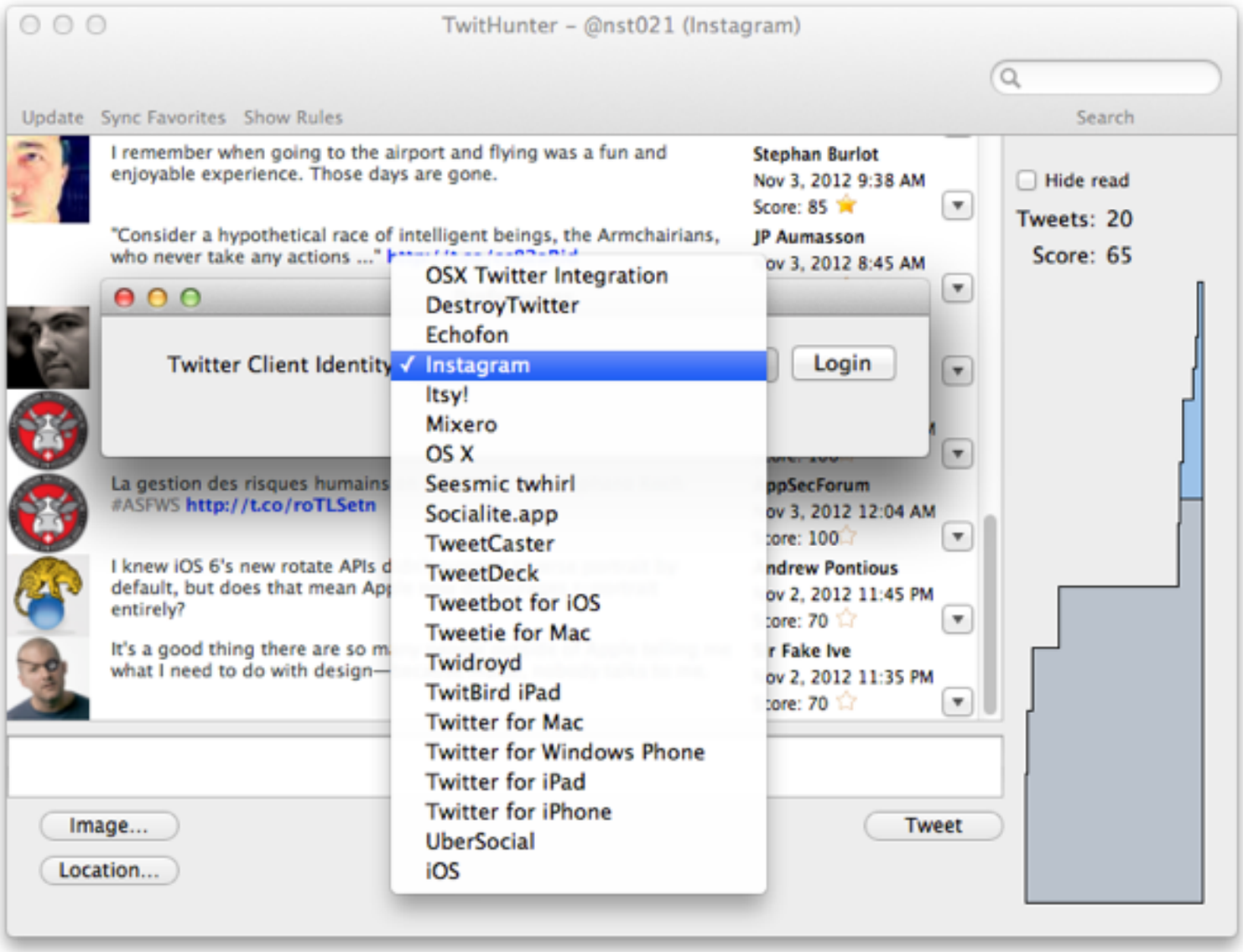
demo OS X

pwning demo

Debug

```
nst:Debug nst$ ./clitter
----------
Sun Oct 20 18:22:44 +0000 2013 @0xced favorited:
[F 1]   [R 0]   @nuthatch       Hmm. I was using SimPholders but now I've discovered Simulator Folders from @john_nye
----------
Sun Oct 20 17:51:02 +0000 2013 @FredericJacobs favorited:
[F 1]   [R 0]   @noneinhere     you know how kids are always complaining about school around here? Kids in #Syria love school and those who don't go to school would die to.
[F 2]   [R 0]   @JZdziarski     If you haven't pulled apart router firmware bundles before, binwalk is a great tool for identifying different components in a .bin.
[F 2]   [R 4]   @noneinhere     "Those who make peaceful revolution impossible will make violent revolution inevitable" #FreeSyria #Syria #act2EndAssadsWar
[F 1]   [R 1]   @lifewinning    Serious question: are people who don't work at NGOs actually going to the #stopwatchingus rally?
[F 2]   [R 0]   @whispersystems @joshualund The only secrets are in the mind of @corbett. There is a secret RedPhone repo though.
[F 1]   [R 0]   @pblilou        @FredericJacobs Yup. Though what I especially like about @AbiHaworth 's article is how it distanciates itself from weird japanese sex stuff.
[F 3]   [R 13]  @AmalHanano     Many people in #Syria this year are not slaughtering lambs for Eid. Meat has become a dream commodity to millions.
[F 1]   [R 2]   @briannoguchi   "In theory, theory and practice are the same. In practice, they are not." — Albert Einstein
[F 4]   [R 3]   @Bedouinocracy  Palestinian prisoners freed from zionist torture chambers and syrian prisoners freed from the Assad gulags. Thats when I'll celebrate.
[F 4]   [R 3]   @Ugariti_Homsi  Until now I haven't lost hope that I'll see Homs again at some point!
[F 52]  [R 129] @iamdevloper    What's long and hard?

                Objective-C method names.
[F 2]   [R 1]   @TelecomixSyria blip blip, reboot sucessful! #system
[F 4]   [R 0]   @Dymaxion       (Also, hello Switzerland! It's been months since I met a new country.)
[F 2]   [R 0]   @MariamGhazalaa I just wanna go back to Syria
----------
Sun Oct 20 17:09:51 +0000 2013 @ESP_09 favorited:
[F 1]   [R 1]   @tedfeed        ted video HD TED: Hetain Patel: Who am I? Think again - Hetain Patel / Yuyu Rau (2013): How do we decide who w... http://t.co/u1DMOX8KnI
[F 156] [R 710] @KimDotcom      Fact: Gmail, iCloud, Skype, etc. have to provide (by law) secret &amp;amp; untraceable NSA backdoors to all your data. #GetOutNow
[F 64]  [R 35]  @r_netsec       Penetration Testing Practice Labs .. LOTS of them! - http://t.co/mNEaXGRCAy
[F 36]  [R 12]  @Kailichi       @TheBloggess do you own these shoes?  If not you need them . They fit your chicken fetish. http://t.co/oBzR4FAlds
[F 3]   [R 5]   @MrsYisWhy      My presentation from @BsidesDC: Homunculus: Why You Will Lose the Battle of BYOD&amp;gt; @slideshare http://t.co/g3tQu9GbdF
[F 242] [R 68]  @lifehacker     Looking to learn a new language? We have five great ways to pick one up: http://t.co/lBuTqvgJGw
[F 11]  [R 3]   @r_netsec       TinySHell — Ported to SCTP (code + video inside) - http://t.co/PwjD4X8CGf
[F 18]  [R 34]  @donmcmillan    BIG DATA Truths: Coffee price at Starbucks is directly related to Engineer's starting salary @EngineersHumor http://t.co/L1kOsyPdSc
[F 1]   [R 3]   @irongeek_adc   NSA Wiretaps are Legal and Other Annoying Facts - @cowboysfaninky Branden Miller for @Hack3rcon 4 http://t.co/4cUJhlWy6M
[F 25]  [R 31]  @SecurityTube   Decrypt SSL Traffic using private key files : http://t.co/k3VFhSYD9w
[F 19]  [R 10]  @DavidJBianco   Slides for my "Enterprise Security Monitoring" talk at #BSidesDC this morning can be found at http://t.co/9AStWOQvhG. Feedback encouraged!
[F 1]   [R 1]   @CompuSecure    Dick Cheney Altered Heart Implant to Thwart Wireless Hackers http://t.co/E6hol0gp9U
[F 23]  [R 4]   @r_netsec       Watch How NOT to do a Penetration Test | SOURCE Barcelona 2011 Episodes | Videos - http://t.co/MR4ipvRE9r
[F 11]  [R 9]   @r_netsec       Facebook CSRF leading to full account takeover - http://t.co/IMpHjyCo5v
[F 26]  [R 62]  @hdmoore        Source code for the backdoored GoAhead web server in Tenda is on GitHub: https://t.co/0yFq3JwjQ8 (found by Preston)
----------
Sun Oct 20 14:18:38 +0000 2013 @taoeffect favorited:
[F 146] [R 517] @_wirepair      Congratulations on your device purchase!
                Would you like:
                A. A US Backdoor.
                B. A Chinese Backdoor.
                C. A Vendor Backdoor.
----------
```

TwitHunter



https://github.com/nst/TwitHunter

Integration Tips

- STTwitterAPI is not a singleton :-) allows access to several accounts simultaneously. Simplifies reverse auth.
- Does not depend on UIKit or AppKit! Easy to implement a command-line client. Also, easy to test.

```
// use default values for URL shortening
-[NSString numberOfCharactersInATweet];
```

Use Categories

STTwitter

```
/*
 GET    lists/subscribers/show

 Check if the specified user is a subscriber of the specified list.
 Returns the user if they are subscriber.
 */

- (void)getListsSubscribersShowForSlug:(NSString *)slug
                       ownerScreenName:(NSString *)ownerScreenName
                           orOwnerID:(NSString *)ownerID
                              userID:(NSString *)userID
                        orScreenName:(NSString *)screenName
                     includeEntities:(NSNumber *)includeEntities // @(YES)
                          skipStatus:(NSNumber *)skipStatus      // @(NO)
                        successBlock:(void(^)())successBlock
                          errorBlock:(void(^)(NSError *error))errorBlock;
```

STTwitter+YourApp

```
- (void)getListsSubscribersForListName:(NSString *)listName
                          successBlock:(void(^)())successBlock
                            errorBlock:(void(^)(NSError *error))errorBlock;
```

Deeper, Generic Fetch Method

```
- (NSString *)fetchResource:(NSString *)resource
                 HTTPMethod:(NSString *)HTTPMethod
              baseURLString:(NSString *)baseURLString
                 parameters:(NSDictionary *)params
              progressBlock:(void(^)(NSString *requestID,
                                     id response))progressBlock
               successBlock:(void(^)(NSString *requestID,
                                     NSDictionary *requestHeaders,
                                     NSDictionary *responseHeaders,
                                     id response))successBlock
                 errorBlock:(void(^)(NSString *requestID,
                                     NSDictionary *requestHeaders,
                                     NSDictionary *responseHeaders,
                                     NSError *error))errorBlock;
```

Contributions

- Some dude with ‘&’ in his password :-)
- API endpoints, ARC, CocoaPods, …
- Plistorious JSON NSOutlineView @bavarious
- Hopefully yours!

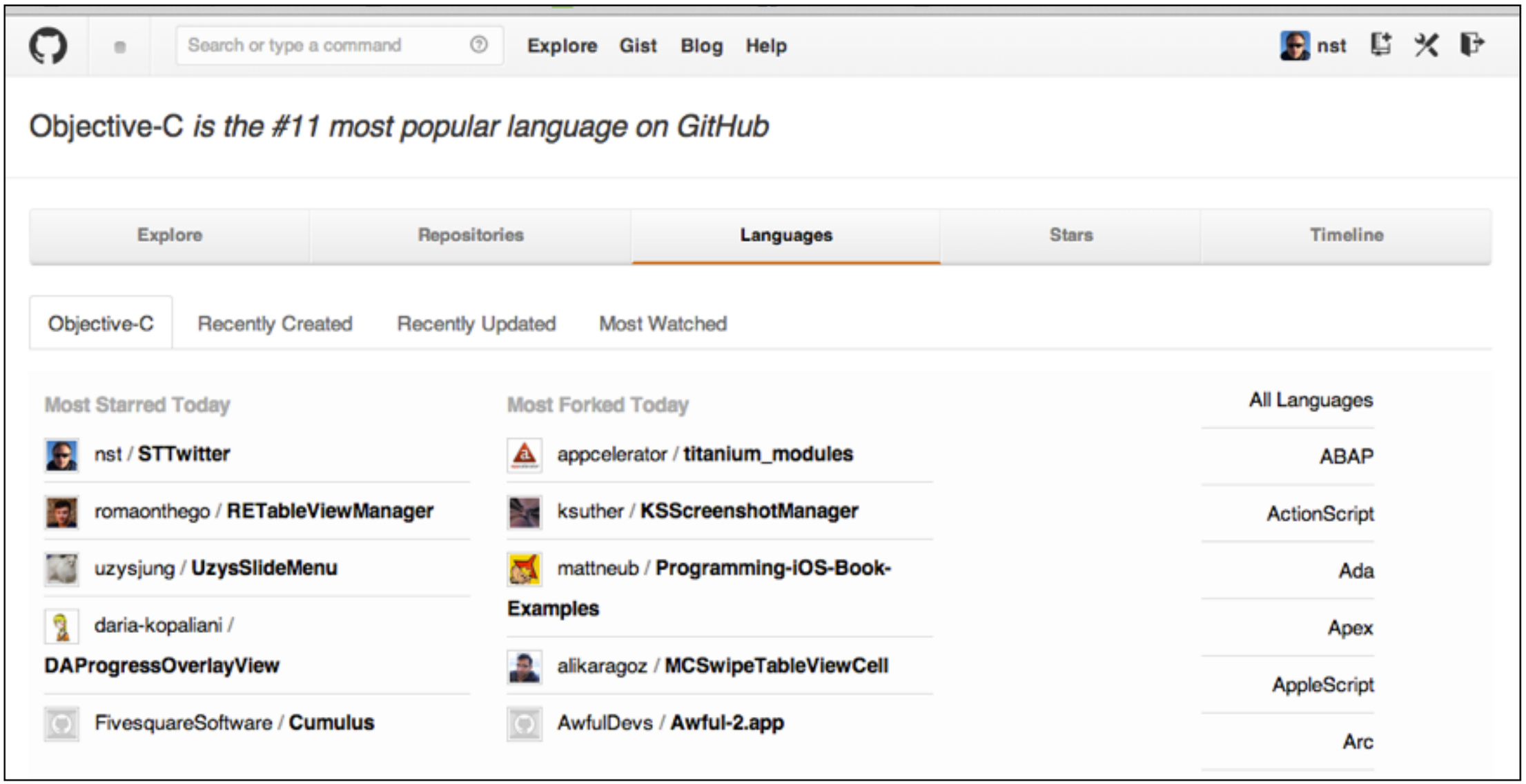
Enthusiatic Users

https://twitter.com/nilsou/status/392364862472736768

August 10th, 2013

https://twitter.com/mattgemmell/status/366203308219695108

August 11th, 2013



Spreading the Word

Adium 1.5.7

stackoverflow

Recap

- Security issues: HITBAMS2013, ASFWS12/13
- SLComposeViewController and SLRequest
- STTwitter unlocks more flows:
  XAuth, Reverse auth, Web auth, App only, …
- STTwitter maps endpoints / Obj-C methods
- Stories on writing an Objective-C library

SWISSQUOTE

Vous avez du talent ?
Swissquote recrute

{{ softshake }}