HCNX SDK iOS 3.0.0

1. Integration **2**

2. SDK's configuration **2**

2.1 Obj-c 2

2.2 Swift 3

3. Media Games Solution **4**

3.1 Get MGS' games 4

3.1.1 Obj-C 4

3.1.2 Swift 4

3.2 Display the rules of the game 4

3.2.1 Obj-C 4

3.2.2 Swift 5

3.3 Play to a game 5

3.3.1 Obj-C 5

3.3.2 Swift 5

3.4 Object Description 5

3.5 Display Sample 7

3.6 Empty State 7

4. Hello Push Notification Solution **8**

4.1 Get segments 8

4.1.1 Obj-C 8

4.1.2 Swift 8

4.2 Manage segments 9

4.2.1 Obj-C 9

4.2.2 Swift 9

4.3 Get campaigns in app 9

4.3.1 Obj-C 9

4.3.2 Swift 10

4.4 Object description 11

4.5 Version 12

4.5.1 Obj-c 12

4.5.2 Swift 12

5. One Time Password solution **12**

5.1 Generate password 12

5.1.1 Obj-c 12

5.1.2 Swift 12

5.2 Authenticate 12

5.2.1 Obj-c 12

5.2.2 Swift 13

6. Testing Integration **13**

7. Migration Guide from 2.x.x to 3.0.0 **14**

MGS : 14

Display the rules : 14

Hello : 14

getSegments : 14

manageSegments : 14

1. Integration

CocoaPods is a dependency manager for Objective-C, which automates and simplifies external library integration processes in your project. To learn how to use CocoaPods in your project, visit the official CocoaPods website.

```
platform :ios, '8.0'
pod "com.hcnx.hcnx"
```

2. SDK's configuration

set On to push Notifications capabilities

2.1 Obj-c

- Open your file AppDelegate.m
- Add the following lines at the top of the file below your own imports :

```
#import <HCNXFramework/HCNXFramework.h>
```

- Search the function application:didFinishLaunchingWithOptions: and add the following line :

```
/* You have to fill the API_KEY and the HMAC_KEY provide by HighConnexion */
    [[HCNX sharedInstance] configureWithAPIKey:@"HCNX_API_KEY"
andHmacKey:@"HCNX_HMAC_KEY"];

/* Ask the push notification authorisation to the user */
    [[HCNX sharedInstance].hello askPermission];
```

- Search the function application:didRegisterForRemoteNotificationsWithDeviceToken: and add the following lines :

```
[[HCNX sharedInstance].hello setTokenDevice:         ];
```

- Search the function application:didRegisterUserNotificationSettings and add the following line :

```
    [application registerForRemoteNotifications];
```

2.2 Swift

- Open your file AppDelegate.swift
- Add the following lines at the top of the file below your own imports :

```
import HCNXFramework
```

- Search the function application(application: UIApplication, didFinishLaunchingWithOptions launchOptions:[NSObject: AnyObject]?) -> Bool and add the following lines :

```
/* You have to fill the API_KEY and the HMAC_KEY provide by HighConnexion */
    HCNX.sharedInstance().configureWithAPIKey("HCNX_API_KEY","HCNX_HMAC_KEY");
/* Ask the push notification authorisation to the use */
    HCNX.sharedInstance().hello.askPermission();
```

- Search the function application:didRegisterForRemoteNotificationsWithDeviceToken: and add the following lines :

```
HCNX.sharedInstance().hello.setTokenDevice(deviceToken)
```

3. Media Games Solution

The HCNX SDK provides methods to use our Media Games solutions (https://app.mgs.media).

3.1 Get MGS' games

Get the mgsGames apply for configured games from the platform MGS.

3.1.1 Obj-C

```
[[HCNX sharedInstance].mgs getGames:^(NSArray *mgsGames, NSError *error) {
}];
```

3.1.2 Swift

```
HCNX.sharedInstance().mgs.getGames { ([Any]?, Error?) in }
```

It's possible to get some games in the list with a date in the past. If you don't want to display them, it's your call to hide them.

3.2 Display the rules of the game

Display the rules of the game in a webview.

3.2.1 Obj-C

Deprecated :

```
[[HCNX sharedInstance].mgs openRules:[NSNumber
numberWithInterger:mgsGame.mgs_id]]
```

New version :

```
[[HCNX sharedInstance].mgs showRules:mgsGame];
```

3.2.2 Swift

Deprecated :

```
HCNX.sharedInstance().mgs.openRules(pIdGame: NSNumber!)
```

New version :

```
HCNX.sharedInstance().mgs.showRules(pGame: MGSGame!)
```

mgsGame (an object of the list mgsGames) *the current game*

3.3 Play to a game

Launch the flow of a game :

3.3.1 Obj-C

```
[[HCNX sharedInstance].mgs play:[NSNumber numberWithInteger:mgsGame.mgs_id]
andAnswer:smsContent_answer];
```

3.3.2 Swift

```
HCNX.sharedInstance().mgs.play(pGame: NSNumber!, andAnswer: String!)
```

mgsGame (an object of the list mgsGames) is the current game.
smsContent_answer maybe null if there isn't an answer to the current game otherwise it matches to an answer choose by the user in the list of the answers of sms Content of the current mgsGame.

3.4 Object Description

1. MGSGame

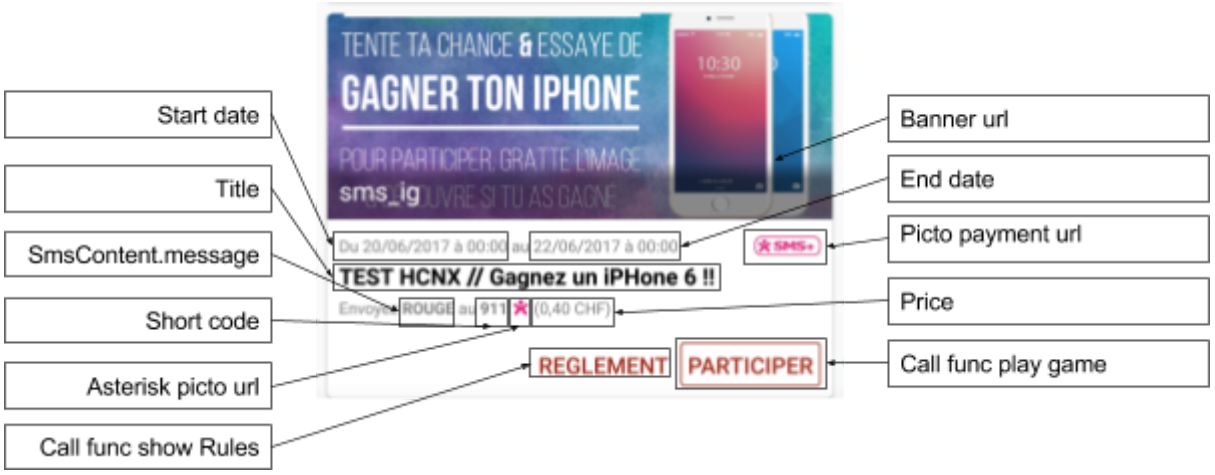
Type	Name	Description	Visibility on the application	Field filled on MGSGame
NSString	pictoAsteriskUrl	url of asterisk image	mandatory	optional
NSString	pictoPayment	url of picto payment image	mandatory	optional
CGFloat	price	price of the game	mandatory	optional

NSString	pdfGameRules	url of the game cgu, a button have to be displayed to be able to call showRules for display it	mandatory	optional
NSString	bannerUrl	url of the main image displayed as a banner	optional	optional
NSString	gameFormat	one of sms, form, audiotel	optional	mandatory
NSString	mgs_id	id of the game Not unique. To have a unique ID for each game, combine mgs_id and gameType.	optional	mandatory
NSString	title	title of the game	optional	optional
NSDate	startDate	game start date	optional	mandatory
NSDate	endDate	game end date	optional	mandatory
SmsContent	smsContent	sms content	optional	optional
NSString	phoneNumber	phone number, can be displayed for audiotel game	optional	optional
NSString	shortcode	shortcode, can be displayed for sms game	optional	optional
NSInteger	position	position in the list		optional
NSString	pictoGameTypeUrl	not used		
NSInteger	highwinId	not used		
NSString	gameType	not used		
NSString	paymentMethod	not used		
NSString	subtitle	not used		
NSString	cgu	not used		
NSString	formId	not used		
NSString	url	not used		
NSDictionary	raw_data	not used		

2 . SmsContent

Type	Name	Description	Visibility on the application	Field filled on MGSGame
NSString	message	message to display in the sms	mandatory	optional
NSString	answers	If you want to display 2 buttons instead of the “participate” button (for example for a quiz game) you can add this function. It’s adapted for cinematics with 2 SMS sent instead of 3 (in the 1st SMS it will pass the SHORTCODE + the answer corresponding to to the button selected) answers of the game.	mandatory	optional

3.5 Display Sample



3.6 Empty State

For a better user experience, we recommend you to have an empty state in the games list.

4. Hello Push Notification Solution

Segments (or channels) are used to categorize the pushes, and can be used, if you wish, to let your users manage which push they wish to receive or not.

4.1 Get segments

Get your segments (Channels) configured from the Hello! platform to manage your populations Opt'in / Opt'out

4.1.1 Obj-C

Deprecated :

```
[[HCNX sharedInstance].hello getChannels:^(NSArray *segments, NSError *error) {
}];
```

The NSArray segments is an array of dictionnary

New version :

```
[[HCNX sharedInstance].hello getSegments:^(NSArray *segments, NSError *error)
{}];
```

The NSArray segments is an array of HelloObject

4.1.2 Swift

Deprecated :

```
HCNX.sharedInstance().hello.getChannels { ([Any]?, Error?) in }
```

The NSArray segments is an array of dictionnary

New version :

```
HCNX.sharedInstance().hello.getSegments { ([Any]?, Error?) in  }
```

The NSArray segments is an array of HelloObject

4.2 Manage segments

The manageSegments method is used to let the user enable or disable pushs for each segment.

4.2.1 Obj-C

Deprecated:

```
[[HCNX sharedInstance].hello manageChannels :subscribeSegments
andUnSubSegmentList:unSubscribeSegments completion:^(NSError *error) { }];
```

New version :

```
[[HCNX sharedInstance].hello manageSegments:subscribeSegments
andUnSubSegmentList:unSubscribeSegments completion:^(NSError *error) {
}];
```

4.2.2 Swift

Deprecated :

```
HCNX.sharedInstance().hello.manageChannels(subChannelList: [Any]!,
andUnSubChannelList: [Any]!) { (Error?) in }
```

New version :

```
HCNX.sharedInstance().hello.manageSegments(subSegmentList: [Any]!,
andUnSubSegmentList: [Any]!) { (Error?) in }
```

subscribeSegments and unSubscribeSegments are arrays of segments ids.

The action of unsubscribe takes over the action of subscribe if two IDs are in both array.

One of the two arrays can be empty.

4.3 Get campaigns in app

4.3.1 Obj-C

```
[[HCNX sharedInstance].hello getCampaignsInApp:^(NSArray *campaigns, NSError
*error) {
}];
```

4.3.2 Swift

```
HCNX.sharedInstance().hello.getCampaignsInApp { ([Any]?, Error?) in }
```

4.4 Object description

HelloObject object :

Type	Name	Description
NSInteger	channel_category_order	Display order of the category of the channel
NSInteger	channel_order	Display order of channel inside its category
NSString	channel_color	Display color of channel in #RRGGBB format
Boolean	channel_is_default	true if channel is subscribed by default by new consumers, false otherwise
Boolean	channel_is_visible	true if channel should be shown to consumer, false otherwise
NSString	channel_name	Display name
NSString	channel_code	Hidden name, used to refer this channel in other webservices.
NSString	channel_category	Display name of channel category
Boolean	channel_subscribed	true if device_token is set and the corresponding customer is subscribed, false otherwise

campaign object :

Type	Name	Description
NSInteger	cpg_id	id of the campaign
NSString	cpg_title	Title of the campaign
NSString	cpg_message	Message of the campaign
NSString	cpg_startdate	Date of the campaign in format : YYYY-MM-DD HH:II:00
NSInteger	cpg_action_type	1, 2 : Application starting ; 5 : Pre-filled SMS; 7 : Application URL scheme opening
NSString	cpg_action_sms_to	the SMS addressee
NSString	cpg_action_urs	the SMS's content

NSString	cpg_action_sms_tex	the app's URL scheme

4.5 Version

4.5.1 Obj-c

```
[[HCNX sharedInstance].version checkVersion:^(NSDictionnary *versionInfo,
NSError *error) {
}];
```

4.5.2 Swift

```
HCNX.sharedInstance().version.check { ([AnyHashable : Any]?, Error?) in }
```

5. One Time Password solution

OTP sdk can be used to check the phone number validity or to add a security layer to your app.

5.1 Generate password

5.1.1 Obj-c

```
[[HCNX sharedInstance].otp generate:phoneNumber  completion^(NSError *error)
{}];
```

5.1.2 Swift

```
HCNX.sharedInstance().otp.generate(phone: String!) { (Error?) in  }
```

5.2 Authenticate

5.2.1 Obj-c

```
[[HCNX sharedInstance].otp authenticate:challenge  completion^(NSString *
userid, NSString token, NSError *error) {}];
```

5.2.2 Swift

```
HCNX.sharedInstance().otp.authenticate(challenge: String!) { (String?,
String>, Error?) in }
```

6. Testing Integration

In order to be able to test MGS and HELLO, you must call enableTestEnvironment method after calling the configure method.
By activating the test environment, the getGame method of MGS will return an example of games and Hello will display your last Push, your hn_id, token and hn_code.

Objective C :

```
[[HCNX sharedInstance] enableTestEnvironment];
```

Swift :

```
HCNX.sharedInstance().enableTestEnvironment()
```

WARNING : these function must be deleted on the build you deliver to High Connexion tests.
WARNING : these function must be deleted before the STORE publication build.

7. Migration Guide from 2.x.x to 3.0.0

MGS :

Display the rules :

Previous call :
[[HCNX sharedInstance].mgs openRules:[NSNumber numberWithInterger:mgsGame.mgs_id]];

openRules is now tagged as deprecated. You can still use this function but it will be deleted in a futur version of the SDK. We recommend to change to getSegments for your next integration.

Current call :
Now you have to pass as parameter the entire object MGSGame instead of the id of the game
[[HCNX sharedInstance].mgs showRules:mgsGame];

Hello :

getSegments :

Previous call :
getChannels:^(NSArray *segments, NSError *error) { }];

getChannels is now tagged as deprecated. You can still use this function but it will be deleted in a futur version of the SDK. We recommend to change to getSegments for your next integration.

Current call :

getSegments:^(NSArray *segments, NSError *error) {}];

manageSegments :

Previous call :

[[HCNX sharedInstance].hello manageChannels :subscribeSegments

andUnSubSegmentList:unSubscribeSegments completion:^(NSError *error) { }];

manageChannels is now tagged as deprecated. You can still use this function but it will be deleted in a futur version of the SDK. We recommend to change to manageSegments for your next integration.

Current call :

```
[[HCNX sharedInstance].hello manageSegments:subscribeSegments
andUnSubSegmentList:unSubscribeSegments completion:^(NSError *error) {
}];
```