Swift & Objective-C

Swift & Objective-C

- Utilisation des API Cocoa Touch
- Utilisation d’Objective-C dans un projet Swift
- Utilisation de Swift dans un projet Objective-C

Utilisation des API Cocoa Touch

Généralités

- Swift a été conçu pour s’interfacer avec Cocoa Touch
- Toutes les API Cocoa Touch sont accessibles depuis Swift
- Des audits des API ont été effectués pour optimiser ces API pour Swift
- Pour utiliser un framework, il faut l’importer

Import

```
import UIKit
import CoreLocation
import CoreData
```

Méthodes

Objective-C

```
UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectZero style:UITableViewStyleGrouped];
[aView insertSubview:anotherView atIndex:5];
```

Swift

```
let tableView = UITableView(frame: CGRect.zero, style: UITableViewStyle.grouped)
aView.insertSubview(anotherView, at: 5)
```

Méthodes

Objective-C

```
UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectZero style:UITableViewStyleGrouped];
[aView insertSubview:anotherView atIndex:5];
```

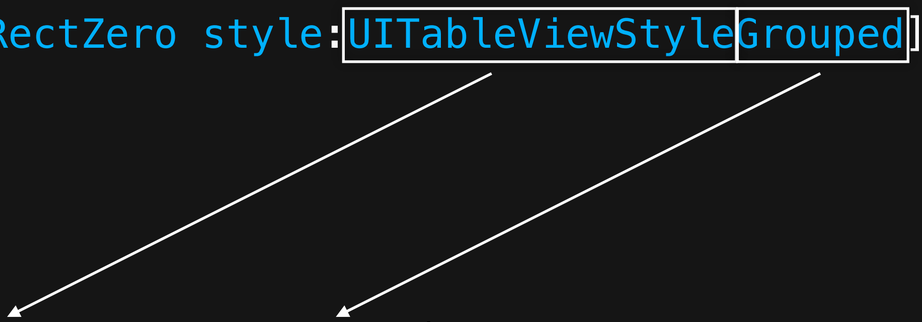
Swift

```
let tableView = UITableView(frame: CGRect.zero, style: UITableViewStyle.grouped)
aView.insertSubview(anotherView, at: 5)
```

Méthodes

Objective-C

```
UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectZero style:UITableViewStyleGrouped];
[aView insertSubview:anotherView atIndex:5];
```



Swift

```
let tableView = UITableView(frame: CGRect.zero, style: UITableViewStyle.grouped)
aView.insertSubview(anotherView, at: 5)
```

Méthodes

Objective-C

```
UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectZero style:UITableViewStyleGrouped];
[aView insertSubview:anotherView atIndex:5];
```

Swift

```
let tableView = UITableView(frame: CGRect.zero, style: .grouped)
aView.insertSubview(anotherView, at: 5)
```

Correspondance de types

- Swift *bridge* automatiquement certains types Objective-C sur ses types
  - `id -> Any`
  - `NSString -> String`
  - `NSArray -> Array<Any>`
  - `NSDictionary -> Dictionary<AnyHashable : Any>`
  - `NSSet -> Set<Any>`
- De nombreux types de Foundation sont également bridgés sur des types *valeur* en Swift.
  - `NSURL -> URL`
  - `NSDate -> Date`
  - `NSIndexPath -> IndexPath`
  - …

Correspondance de types

```
var hello = "hello, world!"
```

Correspondance de types

```
import Foundation

var hello = "hello, world!"
let capitalizedHello = hello.capitalized
```

Méthode de NSString

Any

- Protocole auquel se conforment implicitement toutes les instances Swift
- Un objet générique Objective-C est mappé comme Any en Swift
- Utiliser Any n’est pas sans risque
  - Comme pour id, pas de vérification possible du type réel avant l’exécution
  - Il est préférable de réaliser un transtypage avant l’utilisation

Blocs

- Les blocs Objective-C sont convertis en clôtures
- Différences sur la façon de capturer les variables
  - En Swift les variables ne sont pas copiées mais "référencées"
  - Même fonctionnement qu'en utilisant des `__block` en Objective-C

Utilisation d'Objective-C dans un projet Swift

Généralités

- L'utilisation des bonnes pratiques de développement Objective-C modernes est indispensable pour un usage "agréable" depuis Swift
  - Annotation de nullabilité : bon affichage des optionnels
  - Génériques sur les collections : tapage des collections Swift
  - Utilisation de NS_ENUM() : bon import de l'enum en Swift

Généralités

- Certains patterns Objective-C sont converties en patterns Swift
  - Gestion des erreurs par passage de NSError* en gestion d'erreur Swift

```
- (BOOL)removeItemAtURL:(NSURL*)url error:(NSError **)error;
```

```
func removeItem(at: URL) throws
```

Généralités

- On peut également utiliser nos propres classes Objective-C depuis un projet en Swift
- Il faut spécifier les classes qui seront exposées au runtime Swift
- Lors de l’ajout de classes Objective-C dans un projet Swift, Xcode propose automatiquement l’ajout d’un ***bridging header***

Bridging Header

- NomDuProjet-Bridging-Header.h
- Fichier qui permet d’importer nos classes Objective-C afin de les exposer à Swift

```
//
//  Use this file to import your target's public headers that you would like to expose to Swift.
//

#import "ObjCTableViewCell.h"
#import <AFNetworking.h>
#import <UIImageView+AFNetworking.h>
```

```
let aCell = ObjCTableViewCell()
aCell.cellImageView.image = UIImage(named: "Swift")
```

Utilisation de Swift dans un projet Objective-C

Généralités

- On peut également utiliser nos classes Swift dans un projet Objective-C
- Une classe doit hériter de NSObject pour être accessible depuis l'Objective-C
- Pour être accessible depuis de l’Objective-C, les éléments Swift doivent être marqués avec *@objc*.
- Si tous les éléments doivent être accessibles, on peut marquer la classe avec *@objcMembers.*

Import

- Pour exploiter une classe Swift depuis une classe Objective-C, il faut importer le header autogénéré dans la classe Objective-C

```
#import "NomDuModule-Swift.h"
```

Utilisation

```swift
class ACustomSwiftClass: NSObject {

    var name = "A name"

    @objc class func customInstance() -> ACustomSwiftClass {

        return ACustomSwiftClass()
    }

     @objc init() {

    }
}
```

```objc
ACustomSwiftClass *test2 = [ACustomSwiftClass customInstance];
ACustomSwiftClass *test = [[ACustomSwiftClass alloc] init];
```

Utilisation

```swift
@objcMembers class ACustomSwiftClass: NSObject {

    var name = "A name"

    init(name: String) {

        self.name = name
    }
}
```

```objc
ACustomSwiftClass *test = [[ACustomSwiftClass alloc]initWithName:@"A name"];

test.name = @"";
```

Limitations

- Les fonctionnalités spécifiques au Swift ne sont pas disponibles depuis l'Objective-C
  - Énumérations non Int
  - Structures
  - Tuples
  - Variables globales
  - Fonctions de premier niveau
  - Types imbriqués
  - …

Bon à savoir

- L’ajout d’une seule classe Swift dans un projet Objective-C augmente sensiblement le poids d’une application.
  - Projet Single View
    - Taille App Store en Objective-C : 200Ko
    - Taille App Store en Objective-C + Swift : 8Mo

Bon à savoir

libswiftCore.dylib	27 février 2015 01:18	6,9 Mo
libswiftCoreGraphics.dylib	27 février 2015 01:18	385 Ko
libswiftCoreImage.dylib	27 février 2015 01:18	59 Ko
libswiftDarwin.dylib	27 février 2015 01:18	150 Ko
libswiftDispatch.dylib	27 février 2015 01:18	78 Ko
libswiftFoundation.dylib	27 février 2015 01:18	1,1 Mo
libswiftObjectiveC.dylib	27 février 2015 01:18	116 Ko
libswiftSecurity.dylib	27 février 2015 01:18	58 Ko
libswiftUIKit.dylib	27 février 2015 01:18	99 Ko

- L’ajout d’une seule classe Swift dans un projet Objective-C augmente sensiblement le poids d’une application.
  - Projet Single View
    - Taille App Store en Objective-C : 200Ko
    - Taille App Store en Objective-C + Swift : 8Mo

Pour aller plus loin…

- Using Swift with Cocoa and Objective-C