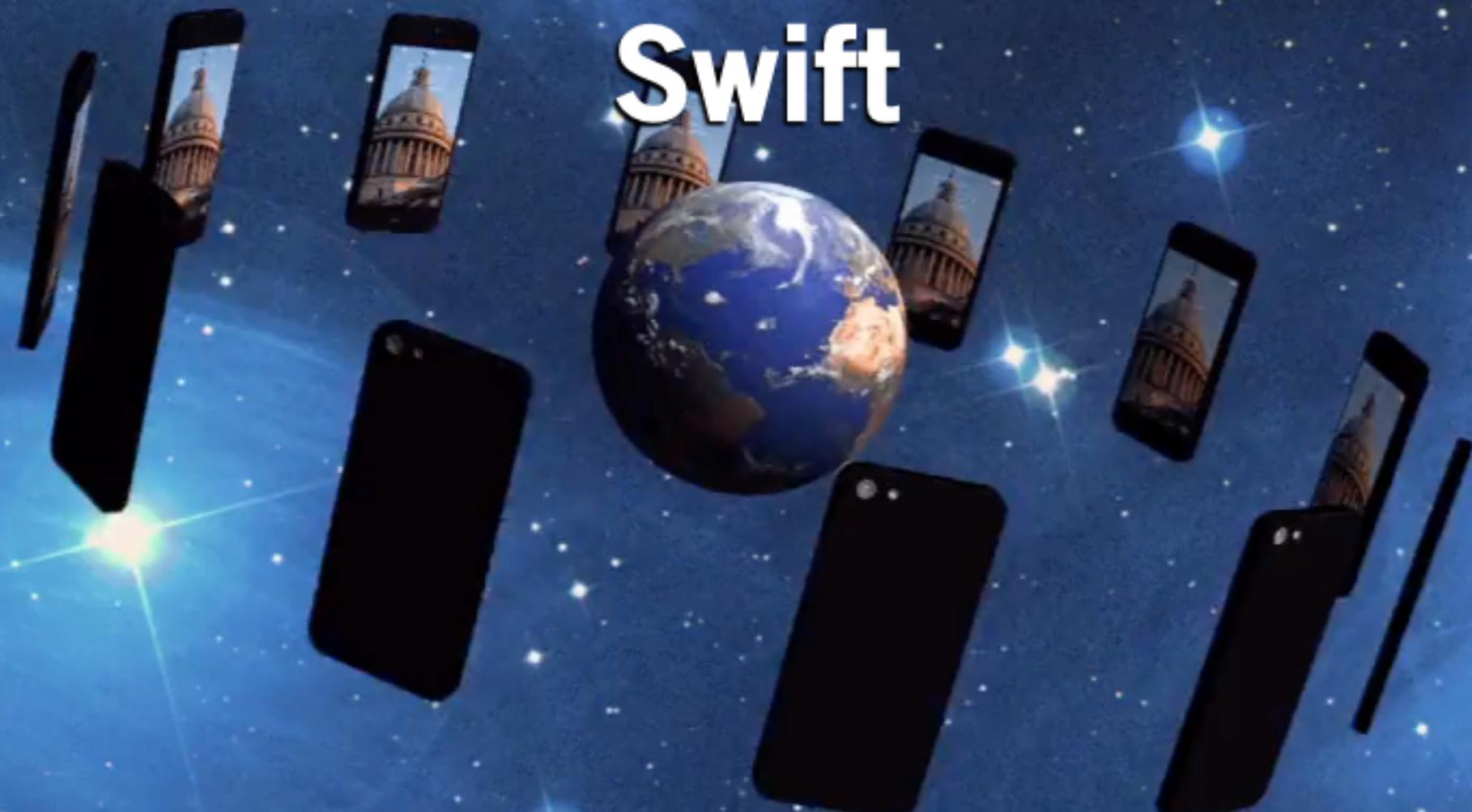


Guide minimum de survie en **Swift**



Fabrice.Kordon@lip6.fr

En guise d'introduction...

Historique

- Conçu depuis 2010 (en secret)
- Lancé en fanfare le 2 juin 2014 (WWDC)
- Version 1.0 stabilisée le 9 septembre 2014



Nous aborderons...

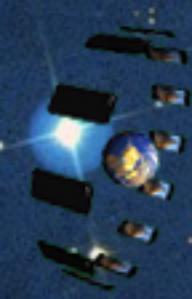
- Principes du langage
- Principaux éléments de syntaxe
- Les chaînes de caractères

► Vous en aurez rapidement besoin



Swift, concepts de base

► Classes et appel de méthodes (super basique)



Principes de Swift

4

- | **Langage Fonctionnel et Objet**
- | **Intègre les dernières techniques de compilation**

- «Facile» (compact à écrire)
 - ▶ Plein d'implicite!
 - ▶ Tout est déduit de vos expressions
 - ▶ Sera-ce lisible?
- Fortement typé (inférence de type)

- | **S'inspire d'Objective-C**

- Mais intègre certains mécanismes
 - ▶ ARC

- | **S'appuie sur les mêmes bibliothèques**

- Donc vous vous y retrouverez...

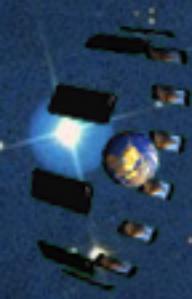
Méthodes de classes*

Contrairement à Objective-C, un seul type de méthodes (sur les instances)

```
class Compteur {  
    var compte: Int = 0  
    func incremente() {  
        compte++  
    }  
    func ajoute(n: Int) {  
        compte += n  
    }  
    func afficheCompte() {  
        println("Compte : \(compte)")  
    }  
    func val () -> Int {  
        return compte  
    }  
}
```

*Usage régulier
dans les API d'iOS*

Plus sur les classe plus tard



Appel de méthodes

Syntaxe pointée (plus «classique»)

```
var monCompte = Compteur()  
monCompte.incremente()  
monCompte.ajoute(2)  
monCompte.afficheCompte()  
println(monCompte.val())
```

```
class Compteur {  
    var compte: Int = 0  
    func incremente() {  
        compte++  
    }  
    func ajoute(n: Int) {  
        compte += n  
    }  
    func afficheCompte() {  
        println("Compte : \(compte)")  
    }  
    func val () -> Int {  
        return compte  
    }  
}
```

Premiers éléments utiles

- ▶ **print / println**
- ▶ **Gestion des chaînes de caractères**

Générer une trace sur la console...

8

Instruction println et print

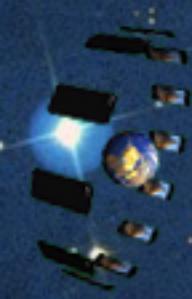
- Prend une chaîne de caractères en paramètre
- Concaténation/substitutions gérées en natif (cf plus loin)

Exemple

```
println ("Toto est content")
```

- Quelques observations...





Les chaînes de caractères

9

Le type string existe mais se déduit (merci le typage dynamique)

- Support d'Unicode intégral
- Comme pour toutes les variables, deux modes
 - ▶ **Immutable ou modifiable**

Exemple

- En C: "Hello World"
- En Objective-C: @"Hello World"

```
let immuable = "Toto l'immobile"
var variable = "Je peux changer"
let duchinois = "Hànzi 漢字"
```

Manipulation des chaînes de caractères

10

Les variables déclarées «let» ne sont pas modifiables

- Il faut les déclarer en «var»

```
let maChaine = "Hello World"
```

```
let meteo = "beau"  
let msg = "Il fait "+meteo
```

```
var logParis = meteoclass()  
println("Le vent souffre à \$(logParis.vent) km/m: ",  
      "\$(logParis.equivalentBeaufort(logParis.vent))",  
      " sur l'échelle de beaufort")
```

```
let myString = "Hello"  
var fullString : String  
fullString = myString+" World!"
```



Pour le reste...

En guise de conclusion...

11

■ Vous disposez (presque 😊) des bases pour construire votre première application en Swift



