



Piscine iOS Swift - Day 00

Calculatrice

Résumé: Ce document contient le sujet du Day 00 de la piscine iOS Swift de [42](#)

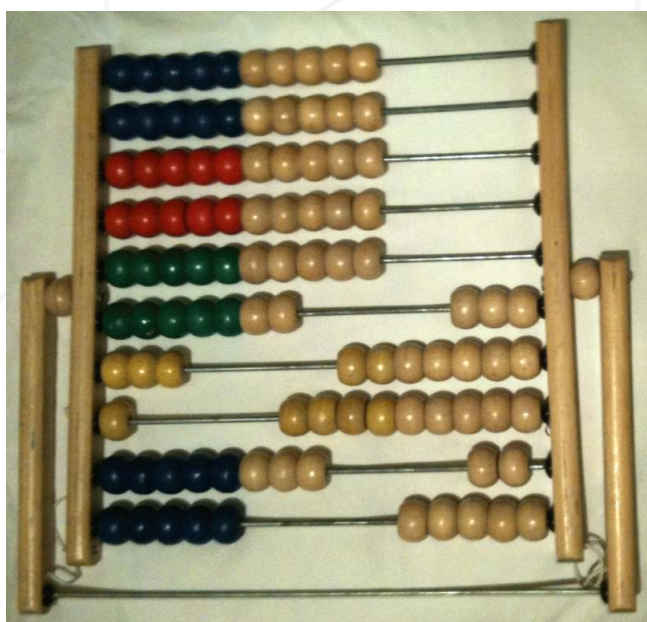
Table des matières

I	Préambule	2
I.1	Catégories	2
I.2	Fonctions	3
I.3	Historique	3
I.4	Utilisation contemporaine	3
I.5	Lecture d'un nombre	4
II	Consignes	5
III	Introduction	6
IV	Exercice 00 : Hello World	7
V	Exercice 01 : Supersize me	8
VI	Exercice 02 : Moar buttons !	9
VII	Exercice 03 : Make some code !	10
VIII	Exercice 04 : Overflows	11

Chapitre I

Préambule

Voici ce que Wikipedia nous raconte sur le Boulier :



Le boulier est un abaque (outil servant à calculer) formé d'un cadre rectangulaire muni de tiges sur lesquelles coulisent des boules.

I.1 Catégories

Le boulier est lié au système de numération décimale, mais il existe deux grandes catégories de bouliers.

Les bouliers en base 10, pour lesquels chaque boule représente, selon la tige sur laquelle elle se trouve, une unité, une dizaine, une centaine... Ces bouliers se rencontrent essentiellement en Europe occidentale et de l'Est. Les décimales peuvent aussi être représentées sur la première tige. Et les bouliers en base alternée (5, 2) pour lesquels chaque tige comprend deux parties : une partie supérieure sur laquelle les boules valent 5 unités (ou 5 dizaines, 5 centaines... selon la position de la tige) et une partie inférieure sur laquelle les boules valent 1 unité (ou 1 dizaine, 1 centaine... selon la position de la tige). Ces bouliers se rencontrent essentiellement en Asie.

I.2 Fonctions

Les bouliers permettent d'effectuer le calcul des opérations élémentaires : additions, soustractions, multiplications et divisions. Dans des mains expertes, il est cependant possible de réaliser d'autres opérations comme le calcul de racines énièmes ou la conversion entre différentes bases.

I.3 Historique

« Cet instrument était utilisé par des peuples très largement séparés comme les étrusques, les Grecs, les égyptiens, les Indiens, les Chinois et les Mexicains et l'on peut penser qu'il a été inventé indépendamment dans différents endroits¹. » En conséquence la datation des découvertes reste aléatoire. Le boulier est sans doute un des plus anciens instruments d'aide au calcul de l'histoire de l'humanité. Les Grecs utilisaient des tablettes recouvertes de sable ou de poussière, les « abaqes ». Le boulier chinois ou suan pan. Il semble dater du XIII^e siècle voire plus tôt (on en trouve une illustration probable sur un ouvrage datant du XII^e siècle) mais sa véritable diffusion date du XV^e siècle³. Sa ressemblance avec le boulier romain peut laisser penser qu'il dérive de celui-ci⁴ mais il est plus probablement dérivé de l'ancien système de calcul chinois avec baguettes³. Sur chaque tige, on trouve cinq boules représentant une unité et deux boules représentant cinq unités, séparées par une barre centrale. Le boulier japonais ou soroban. Il a progressivement perdu, par rapport au boulier chinois, deux boules (une boule de valeur 1 et une boule de valeur 5). Le boulier dit russe ou Stchoty, utilisé également en Iran sous le nom de Tchortkeh et en Turquie sous le nom de coulba, est formé de tiges portant dix boules de valeur 1. Le boulier-compteur ou d'école a été utilisé dans les des écoles enfantines françaises jusqu'au XVIII^e siècle, variante probable de l'instrument russe. Dans le monde entier, les bouliers ont été utilisés dans les écoles maternelles et primaires comme une aide à l'enseignement de l'arithmétique. Dans les pays occidentaux, un cadre de perles semblables au boulier de Russie, avec un cadre vertical (voir image du boulier d'école). Il est constitué de dix perles de bois sur dix tiges. Ce type de boulier est utilisé pour représenter des nombres sans utiliser la valeur de position. Chaque perle et chaque tige horizontale a la même valeur, et utilisées de cette façon, il peut représenter des nombres entiers de 0 à 100. En utilisant les valeurs de position comme montré dans l'image, il peut aussi représenter des nombres entiers de 0 à 11 111 111 111 110, ou bien des nombres avec trois décimales après la virgule, de 0 à 11 111 111 111,110.

I.4 Utilisation contemporaine

Même si la calculatrice électronique est très puissante, le boulier est courant dans toute l'Asie. Par exemple, des commerçants russes, iraniens et asiatiques utilisent une calculatrice, puis vérifient le résultat à l'aide du boulier. En 1945, un match opposant un comptable japonais muni d'un soroban et un opérateur de calculatrice électrique a été gagné par le Japonais par un score de 4 à 15.

I.5 Lecture d'un nombre

Chaque colonne représente en partant de la droite, les unités, les dizaines, les centaines etc. Les cinq boules en dessous de la barre valent chacune un, et les deux boules situées au-dessus de la barre valent chacune cinq. On ne prend en compte dans le calcul du nombre représenté que les boules activées, c'est-à-dire déplacées près de la barre centrale horizontale.

Chapitre II

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Lisez attentivement l'intégralité du sujet avant de commencer.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Vos exercices seront corrigés par vos camarades de piscine.
- Le sujet fait foi, ne vous fiez pas toujours à la lettre aux demos qui peuvent contenir des ajouts supplémentaires non demandés.
- Vous devrez rendre une app par jour (sauf pour le Day 01) sur votre depot git, rendez le dossier du projet Xcode.
- Voici le manuel officiel de [Swift](#) et de [Swift Standard Library](#)
- Il est interdit d'utiliser d'autres librairies, packages, pods... avant le Day 07
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Pensez à discuter sur le forum Piscine de votre Intra !
- Réfléchissez. Par pitié, par Odin ! Nom d'une pipe.



Les videos sur l'intra ont été tournées avant Swift 3. Enlevez le prefix "NS" que vous voyez devant les class/struct/function dans le code des videos pour les utiliser en Swift 3.



L'intra indique la date et l'heure de fermeture de vos dépôts. Cette date et heure correspond également au début de la période de peer-evaluation pour le jour de piscine correspondant. Cette période de peer-evaluation dure exactement 24h. Une fois ces 24h passées, vos notes peer manquantes seront complétées par des 0.

Chapitre III

Introduction

Pour bien commencer le développement d'une application mobile iOS en Swift, il est important de comprendre comment fonctionne Xcode.

Xcode est un [IDE](#) développé par Apple qui permet de créer des applications pour Mac OS X, iOS, watchOS et tvOS.


Swift est un langage de programmation open source, multi-paradigmes aussi développé par Apple. Il est extrêmement récent puisque sa première version est apparue le 2 juin 2014.

Dans ce premier jour de piscine vous allez apprendre à utiliser Xcode et découvrir le Swift en réalisant une petite application de calculatrice.

Cette application vous permettra de faire un premier pas dans le monde du développement mobile en vous faisant découvrir comment créer des liens entre la vue et le code. Cette application ne prendra en compte que les entiers et les opérations de base.

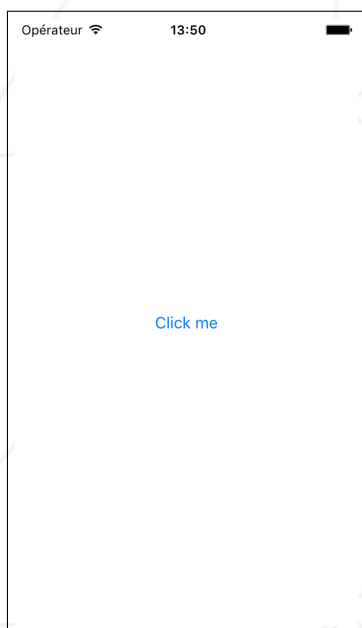
Chapitre IV

Exercice 00 : Hello World

	Exercice : 00
Hello World	
Fichiers à rendre : .xcodeproj et tous les fichiers nécessaires	
Fonctions Autorisées : Swift Standard Library, UIKit	
Remarques : n/a	


Pour ce premier exercice vous devez créer votre premier projet Xcode, pour iOS en langage Swift... Oui car, jusqu'à preuve du contraire, ceci n'est pas une piscine d'Objective-C fort heureusement.

Créez sur la vue principale, un **UIButton** qui, lorsque l'on clique dessus, affiche un message **QUELCONQUE** dans la console de debug deXcode.



Chapitre V

Exercice 01 : Supersize me

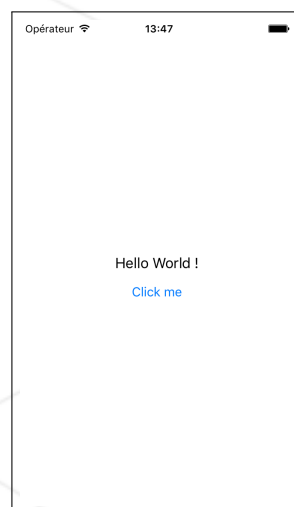
	Exercice : 01
Supersize me	
Fichiers à rendre : .xcodeproj et tous les fichiers nécessaires	
Fonctions Autorisées : Swift Standard Library, UIKit	
Remarques : n/a	

Ajoutez à ce projet un **UILabel** dans votre vue principale qui, lorsqu'on clique sur le **UIButton**, change le texte du Label. Vous devez aussi gérer l'**AutoLayout**.

Le UIButton et le UILabel doivent être centrés horizontalement sur tous les devices, même en mode paysage.




Utiliser une StackView peut vous être utile pour l'autolayout.



Chapitre VI

Exercice 02 : Moar buttons !

	Exercice : 02
Moar buttons !	
Fichiers à rendre : .xcodeproj et tous les fichiers nécessaires	
Fonctions Autorisées : Swift Standard Library, UIKit	
Remarques : n/a	

Il manque des touches, vous ne trouvez pas ?
Vous allez maintenant y ajouter toutes les touches d'une calculatrice simple, bien entendu en tant que **UIButton** :

- Les chiffres de 0 à 9
- 'AC' qui réinitialisera la calculatrice
- '=' qui exécutera l'opération avec les 2 opérandes
- Les opérateurs '+', '-', '/', '*'
- 'NEG' qui prendra l'opposé du nombre en cours


Une fois tous les **UIButton**s placés correctement, assurez vous que l'**AutoLayout** est toujours géré (soit sur tous les devices et tous les modes).

Les touches assignées aux chiffres doivent pouvoir modifier le **UILabel** en changeant le nombre affiché dessus, mais les autres touches ne seront pas à programmer pour le moment.

Vous en profiterez pour rajouter un peu de debug. A chaque appui sur un **UIButton**, son action doit être écrite dans la console de debug (le format est libre).

Chapitre VII

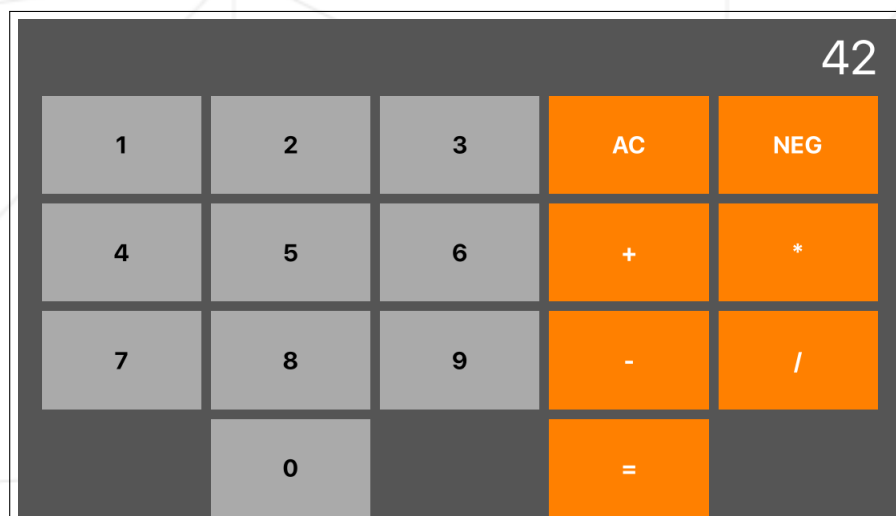
Exercice 03 : Make some code !

	Exercice : 03
Make some code !	
Fichiers à rendre : .xcodeproj et tous les fichiers nécessaires	
Fonctions Autorisées : Swift Standard Library, UIKit	
Remarques : n/a	

A présent, vous allez coder les actions à exécuter lorsqu'une opération est demandée. Le **UILabel** doit pouvoir afficher le résultat de l'opération et on doit pouvoir enchaîner les opérations. De même, l'**AutoLayout** doit être toujours géré.




Attention à la division par 0 !



Chapitre VIII

Exercice 04 : Overflows

	Exercice : 04
Overflows	
Fichiers à rendre : .xcodeproj et tous les fichiers nécessaires	
Fonctions Autorisées : Swift Standard Library, UIKit	
Remarques : n/a	

Si vous avez bien fait vos tests dans l'exercice précédent, vous avez peut être remarqué que votre application crashe lorsque les nombres deviennent trop grands (les positifs comme les négatifs). Vous faites ce que l'on appelle un **overflow**.

La negligence des **overflows** peut causer de [lourds degats](#) !

Dans cet exercice, vous allez palier à ce problème en incorporant la gestion des **overflows**.



Votre application ne doit pas crasher, sous aucun pretexte !