DEVELOPPEMENT APPLICATION CALCULATRICE

I. INTRODUCTION

Pour débuter le projet, nous devions développer une application de calculatrice sur iOS et Android dans le but de s'initier au développement d'application.

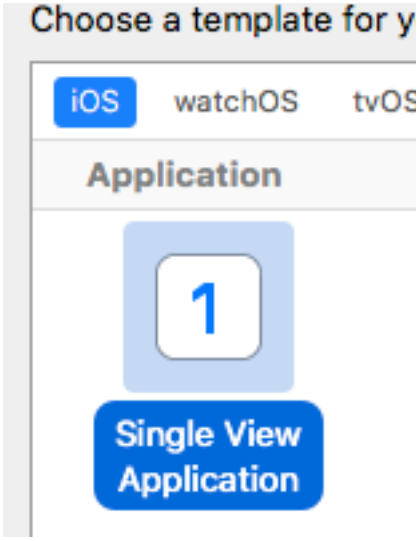
II. XCODE/SWIFT

Xcode est le logiciel de développement de MacOS, il permet de développer pour iOS sur iPad iPhone mais aussi pour l'Apple TV, l'Apple Watch, etc..

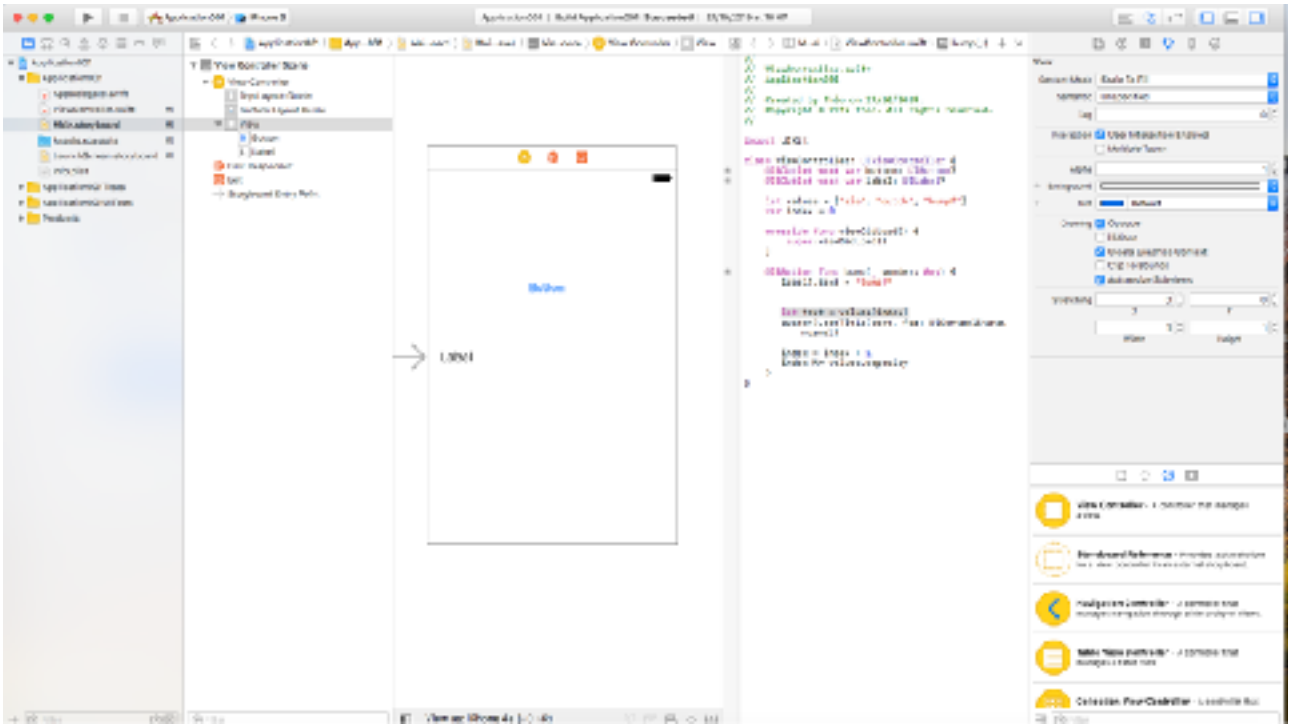
Pour développer sur iOS, j'ai eu le choix entre Objective C et Swift comme langage de programmation, j'ai commencé avec Swift, car il était plus récent et recommandé

1. Découverte de Xcode et de Swift

Pour developper une calculatrice, une application simple a une seule vue suffis, donc j'ai sélectionné un projet en « Single View » .



Voici l'interface de Xcode :



On peut voir que sur l'interface, j'ai intégré deux fenêtre, une pour le contrôleur (a gauche) une pour le code (a droite).

Avec le menu déroulant sur le bas de la droite j'ai placé un bouton et un label, que qui sont configurable sur le menu du haut de la droite.

Puis avec le bouton ctrl et le clique gauche, avec la souris j'ai créé une liaison du bouton vers le code.

```
    @IBOutlet weak var button: UIButton?
    @IBOutlet weak var label: UILabel?
```

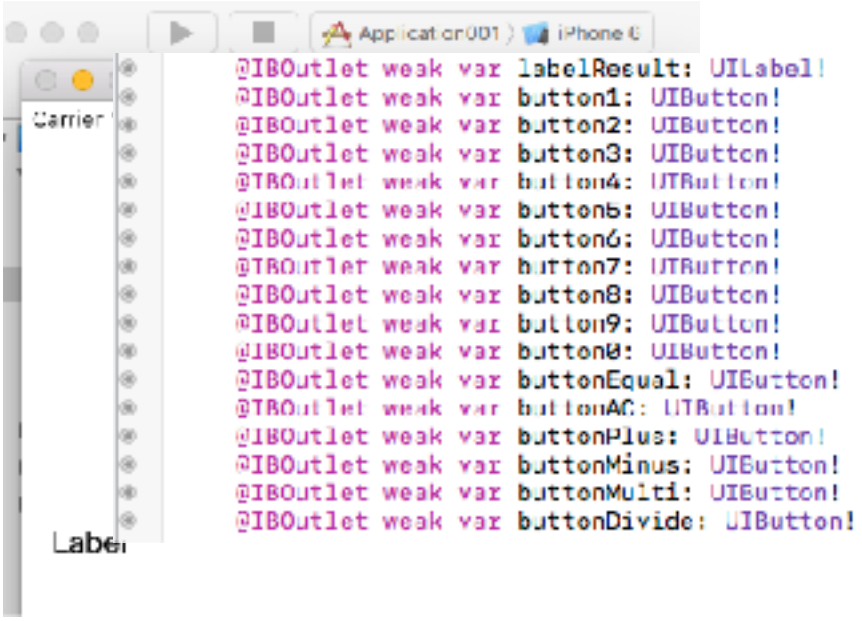
On crée donc des outlets qui vont permettre d'intégrer ces objets dans le code sous forme de variable.

```
@IBAction func bump(_ sender: Any) {
```

On peut utiliser la même manipulation pour intégrer l'action d'un objet sous forme de fonction, pour enclencher un code.

```
@IBAction func buttonClick(_ sender: UIButton) {
    if sender == button0 {
```

On peut aussi lier plusieurs actions de plusieurs bouton a la même fonction, sélectionnera notre bouton activé avec une condition pour le bouton défini.

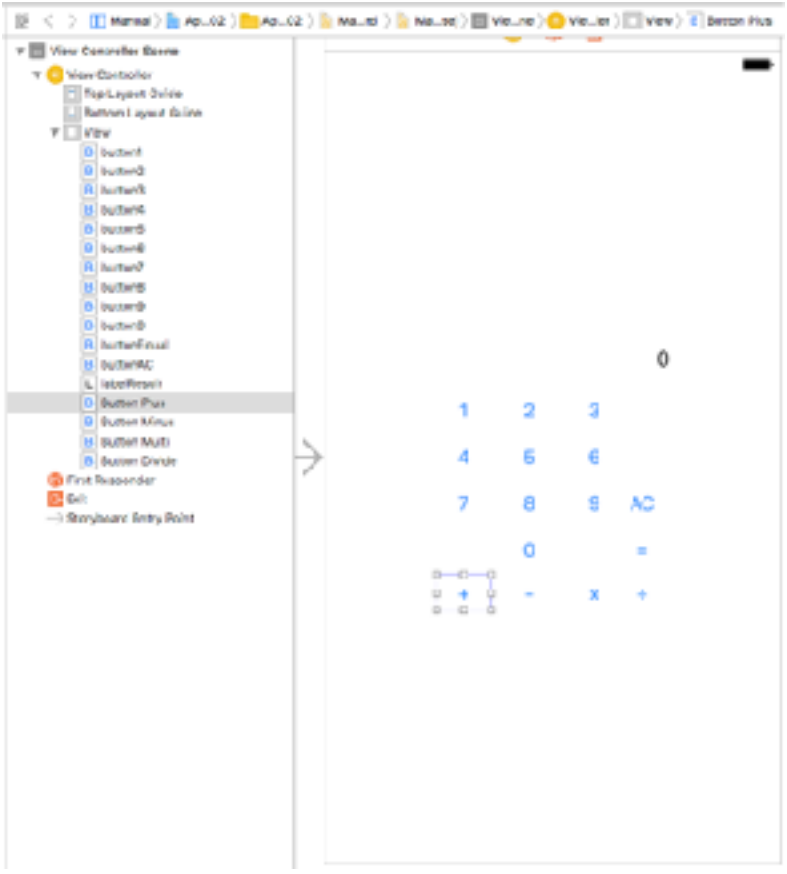


Je peux aussi simuler un iPhone avec Xcode pour tester mon application.

---

2. Développement de la Calculatrice

Ayant acquis quelques notion de base en Xcode et en Swift, je commence le développement de la calculatrice



Je commence d'abords par placer les boutons de la calculatrice.

```
else if sender == buttonAC {
    affichage = "0";
    firstNum = 0
    SecondNum = 0
    operation = ""
    reserve = ""
}
```

Puis je les lie au code.

```
@IBAction func buttonClick(_ sender: UIButton) {
```

Ensuite je lie tout les actions des boutons a une fonction qui va s'enclencher lors du click

---

Je commence donc le développement d'un système de calculatrice

Je crée des variable qui me serviront :

```
var affichage = "0"
var operation = ""
var firstNum = 0
var reserve = ""
var SecondNum = 0
var resultat = 0
```

Une pour l'affichage (à 0 au début)
Une qui stockera l'opération
Une pour le premier chiffre de l'opération
Une « réserve » nécessaire pour mon système
Une pour le deuxième chiffre
Et une pour le résultat de l'opération

Puis je commence à écrire dans la fonction

```
else if sender == button1 {
    if affichage == "0" {
        affichage = "1";
    }
    else {
        affichage += "1";
    }
    if firstNum != 0 {
        reserve += "1";
    }
}
```

Je met une condition if pour chaque bouton (si j'appui sur tel bouton), puis je lui dis si l'affichage est a 0, il change le nombre pour le chiffre du bouton, sinon il rajoute le chiffre du bouton après le précédant.
La condition suivante sert a déterminer si le premier nombre a été rentré (plus précisément après l'appui sur un bouton d'un signe d'opération), si c'est le cas, il place le nombre dans une « réserve ».

Pour le bouton AC, on réinitialise tout à son état de base.

```
else if sender == buttonPlus {
    operation = "+"
    firstNum = Int(affichage)!
```

Pour un bouton de signe d'opération, en plus de l'affichage, nous allons stocker le signe dans une variable, et prendre le chiffre d'affichage pour l'insérer dans une autre variable dédiée (nous le convertissons en INT pour pouvoir l'utiliser dans une opération, car la variable par défaut est en STRING).

```
else if sender == buttonEqual {
    affichage += " = ";
    SecondNum = Int(reserve)!
    if operation == "+" {
        resultat = firstNum+SecondNum
    }
    if operation == "-" {
        resultat = firstNum-SecondNum
    }
    if operation == "*" {
        resultat = firstNum*SecondNum
    }
    if operation == "/" {
        resultat = firstNum/SecondNum
    }
    affichage += "\(resultat)"

}
```

Pour le bouton égal, on ajoute la réserve dans une variable INT dédiée (pour avoir deux valeurs), et avec une condition if déterminant quel signe d'opérateur il s'agit il effectue le calcul puis ajoute le résultat a l'affichage.

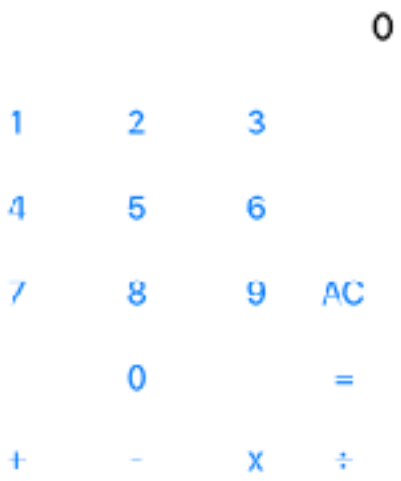
```
labelResult.text = "\(affichage)"
```

Bien sur il ne faut pas oublier d'afficher la variable d'affichage sur le label pour voir l'opération et le résultat.

---

3. Test de la calculatrice

Je teste donc la calculatrice si elle fonctionne, l'opération fonctionne donc bien 24x12 font bien 288.



Et lorsque j'appuie sur le bouton AC l'affichage affiche 0 et réinitialise le tout.

Je passe donc au développement Android.