

Projet 7 : CountOnMe

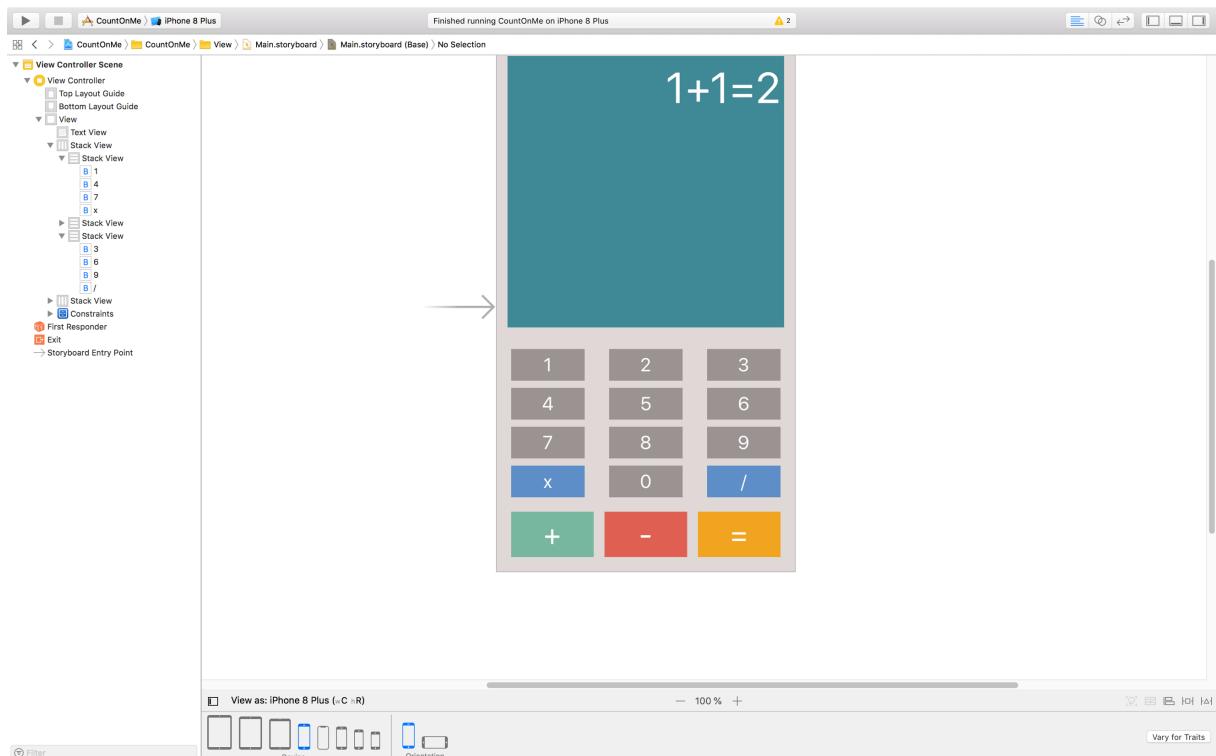
Présentation :

Dans le cadre de la formation Openclassrooms dans le parcours Développeur d'applications IOS. Je présente la feature créer en bonus pour le Projet 7 CountOnMe.

Explication de ma feature :

Multiplier et Diviser :

J'ai tout d'abord implémenté les 2 boutons (le Multiplier et le Diviser) à notre Interface dans le Main Storyboard. Puis venir ajouter une action à chacun dans le ViewController.



En revenant dans ma class Operations, j'ai dans ma fonction priority calcul dans calculate total. Qui ajoute plusieurs conditions qui vont vérifier lorsque premièrement si notre tableau de calculs contient bien le multiplier et diviser les calculer en priorité, faire correspondre le calcul de chacun et les additionner ou soustraire si c'est un calcul de plusieurs opérateurs.

```
gate.swift 92 //Calculate result total
93 func calculateTotal() -> String {
94     if !isExpressionCorrect {
95         return "0"
96     }
97     priorityCalcul()
98     var total: Double = 0
99     for (i, stringNumber) in stringNumbers.enumerated() {
100         if let number = Double(stringNumber) {
101             if operators[i] == "*" {
102                 total += number
103             } else if operators[i] == "/" {
104                 total -= number
105             }
106         }
107     }
108     let result = String(format: "%.2f", total)
109     clear()
110     return String(result)
111 }
112
113 //Check if calcul contains multiply or divider for priority calcul
114 private func priorityCalcul() {
115     var result: Double = 0
116     let priorityOperators = "x/"
117     for (index, stringNumber) in stringNumbers.enumerated().reversed() {
118         if let number = Double(stringNumber) {
119             if priorityOperators.contains(operators[index]) {
120                 if let currentNumber = Double(stringNumbers[index-1]) {
121                     if operators[index] == "*" {
122                         result = Double(currentNumber * number)
123                     } else if operators[index] == "/" {
124                         if number == 0 {
125                             showAlertDelegate?.showAlert(title: "Error", message: "Erreur calcul")
126                         } else {
127                             result = Double(currentNumber / number)
128                         }
129                     }
130                 }
131                 stringNumbers[index-1] = String(result)
132                 stringNumbers.remove(at: index)
133                 operators.remove(at: index)
134             }
135         }
136     }
137 }
138 }
```

Ensuite pour chacun je créer une fonction pour chaque opérateur lorsque celui-ci sera appuyé par l'utilisateur, qui va vérifier que le calcul soit possible, puis l'effectuer et renvoyer la valeur du résultat à notre fonction.

```

35     return true
36 }
37
38 var canAddOperator: Bool {
39     if let stringNumber = stringNumbers.last {
40         if stringNumber.isEmpty {
41             showAlertDelegate?.showAlert(title: "Zéro!", message: "Expression incorrecte!")
42             return false
43         }
44     }
45     return true
46 }
47
48 // MARK: - Methods
49
50 func plus() -> String {
51     if canAddOperator {
52         operators.append("+")
53         stringNumbers.append("")
54     }
55     return updateDisplay()
56 }
57
58 func minus() -> String {
59     if canAddOperator {
60         operators.append("-")
61         stringNumbers.append("")
62     }
63     return updateDisplay()
64 }
65
66 func multiply() -> String {
67     if canAddOperator {
68         operators.append("X")
69         stringNumbers.append("")
70     }
71     return updateDisplay()
72 }
73
74 func divider() -> String {
75     if canAddOperator {
76         operators.append("/")
77         stringNumbers.append("")
78     }
79     return updateDisplay()
80 }
81
82

```

Et pour finir, j’instancie la class Opérations dans le ViewController pour venir appeler chaque fonction dans chacune des actions du contrôleur correspondant à son bouton.

```

17
18 // MARK: - Properties
19 var operations = Operations()
20
21 override func viewDidLoad() {
22     operations.displayAlertDelegate = self
23 }
24
25 // MARK: - Action
26
27 @IBAction func tappedNumberButton(_ sender: UIButton) {
28     for (i, numberButton) in numberButtons.enumerated() {
29         if sender == numberButton {
30             operations.addNewNumber(i)
31         }
32     }
33     textView.text = operations.updateDisplay()
34 }
35
36 @IBAction func plus() {
37     textView.text = operations.plus()
38 }
39
40 @IBAction func minus() {
41     textView.text = operations.minus()
42 }
43
44 @IBAction func equal() {
45     textView.text = operations.calculateTotal()
46 }
47
48 @IBAction func multiply() {
49     textView.text = operations.multiply()
50 }
51
52 @IBAction func divider() {
53     textView.text = operations.diviser()
54 }
55 }
56
57 extension ViewController: DisplayAlert {
58     func showAlert(title: String, message: String) {
59         let alertVC = UIAlertController(title: title, message: message, preferredStyle: .alert)
60         alertVC.addAction(UIAlertAction(title: "OK", style: .cancel, handler: nil))
61         self.present(alertVC, animated: true, completion: nil)
62     }
63 }

```