
GROUP
PROJECT

TODAY'S AGENDA

1. Introduction
2. The team
3. Project overview
4. Images of app
5. Difficulties

THE TEAM

Samadjon

-connection calculator buttons to actions & updates the displayed expression (workings)

Ahliyor

-interface design and helper functions

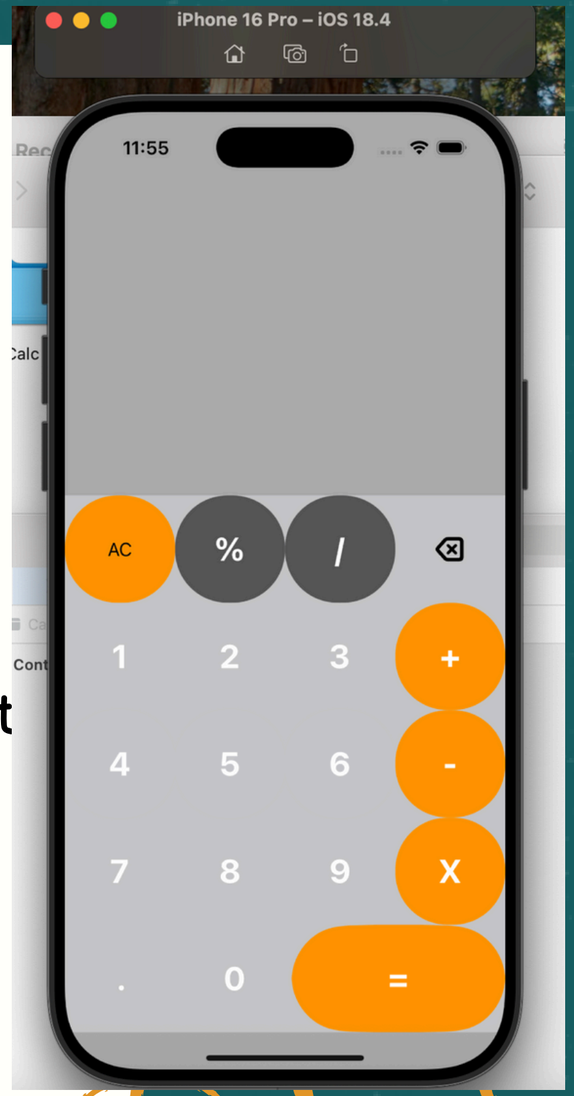
-

Shohboz

- button actions for a calculator app

INTRODUCTION

The Swift Calculator App is a lightweight and user-friendly iOS application developed using Swift and UIKit . It is designed to handle basic arithmetic operations with a clean, responsive, and modern user interface.



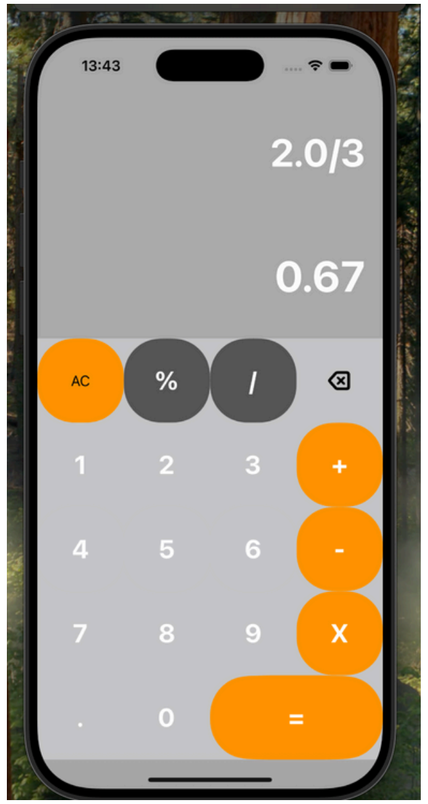
```swift
@IBOutlet weak var q: UIButton!
@IBOutlet weak var w: UIButton!
@IBOutlet weak var e: UIButton!
@IBOutlet weak var r: UIButton!
@IBOutlet weak var t: UIButton!
@IBOutlet weak var y: UIButton!
@IBOutlet weak var u: UIButton!
@IBOutlet weak var i: UIButton!
@IBOutlet weak var o: UIButton!
@IBOutlet weak var p: UIButton!
@IBOutlet weak var a: UIButton!
@IBOutlet weak var s: UIButton!
@IBOutlet weak var d: UIButton!
@IBOutlet weak var f: UIButton!
@IBOutlet weak var g: UIButton!
@IBOutlet weak var h: UIButton!
@IBOutlet weak var j: UIButton!
@IBOutlet weak var k: UIButton!

@IBOutlet weak var button: UIBut
var workings:String = ""
```

```swift
@IBAction func equalsTap(_ sender: Any)
{
    if(validInput())
    {
        let checkedWorkingsForPercent = workings.replacingOccurrences(of: "%", with: "*0.01")
        let expression = NSExpression(format: checkedWorkingsForPercent)
        let result = expression.expressionValue(with: nil, context: nil) as! Double
        let resultString = formatResult(result: result)
        calculatorResults.text = resultString
    }
    else
    {
        let alert = UIAlertController(
            title: "Invalid Input",
            message: "Calculator unable to do math based on input",
            preferredStyle: .alert)
        alert.addAction(UIAlertAction(title: "Okay", style: .default))
        self.present(alert, animated: true, completion: nil)
    }
}
```

```swift
func validInput() ->Bool
{
    var count = 0
    var funcCharIndexes = [Int]()

    for char in workings
    {
        if(specialCharacter(char: char))
        {
            funcCharIndexes.append(count)
        }
        count += 1
    }

    var previous: Int = -1

    for index in funcCharIndexes
    {
        if(index == 0)
        {
            return false
        }

        if(index == workings.count - 1)
        {
            return false
        }

        if (previous != -1)
        {
            if(index - previous == 1)
            {
                return false
            }
        }
        previous = index
    }

    return true
}
```

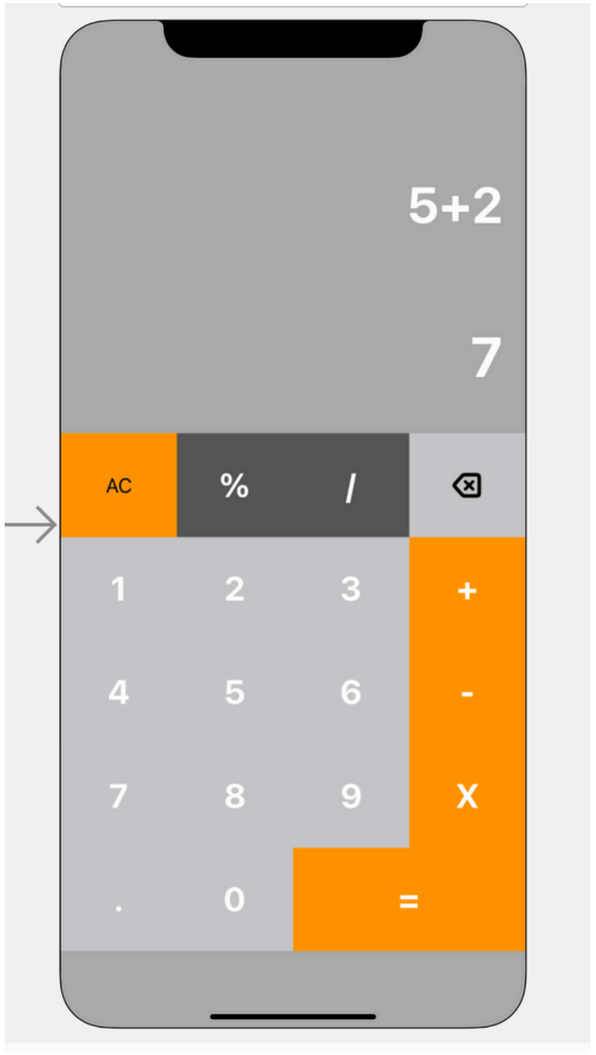
```swift
func specialCharacter (char: Character) -> Bool
{
    if(char == "*")
    {
        return true
    }
    if(char == "/")
    {
        return true
    }
    if(char == "+")
    {
        return true
    }
    return false
}
```

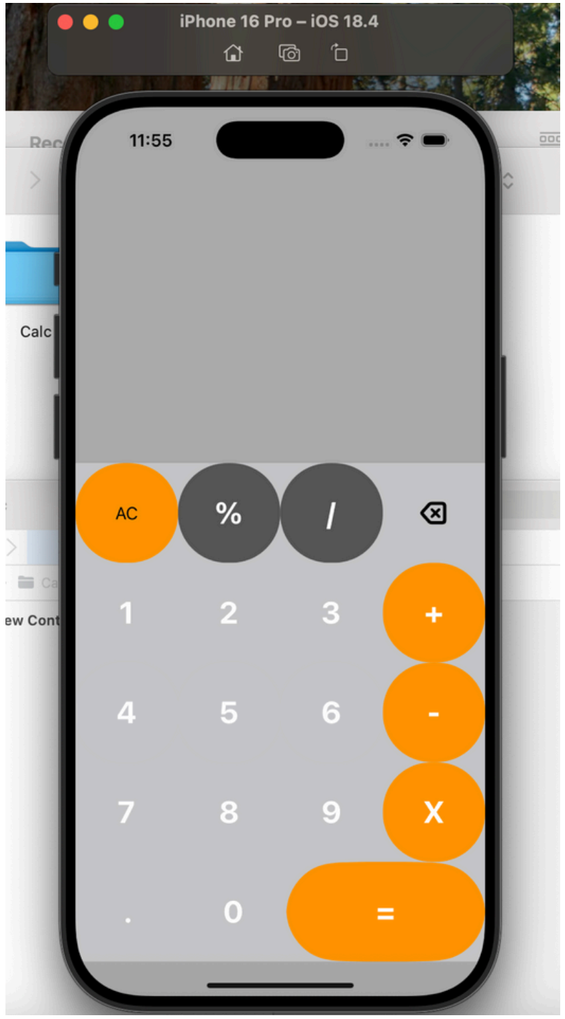
```
func handleOperator(_ op: String) {
    if let lastChar = workings.last, "%+-*/.".contains(lastChar) {
        workings.removeLast()
    }
    workings += op
    calculatorWorkings.text = workings
}
```



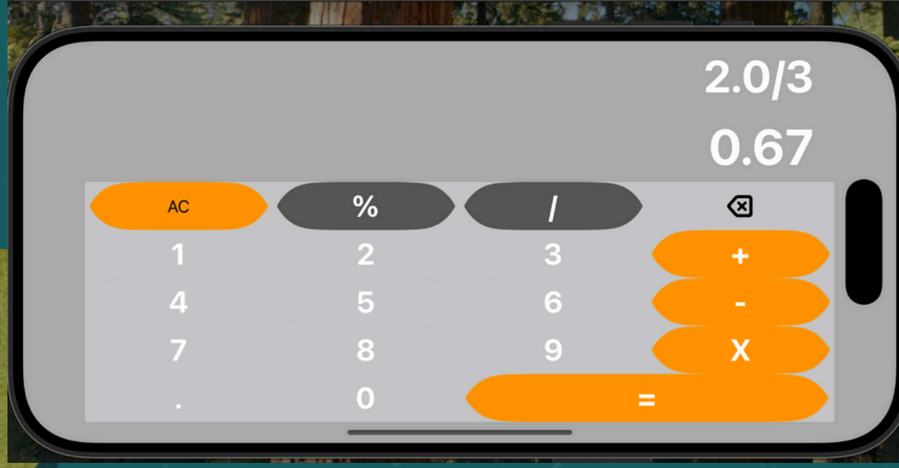
```swift
func addToWorkings(value: String)
{
    workings = workings + value
    calculatorWorkings.text = workings
}

@IBAction func percentTap(_ sender: Any)
{
    handleOperator("%")
}

@IBAction func divideTap(_ sender: Any)
{
    handleOperator("/")
}

@IBAction func timesTap(_ sender: Any)
{
    handleOperator("*")
}

@IBAction func minusTap(_ sender: Any)
{
    handleOperator("-")
}

@IBAction func plusTap(_ sender: Any)
{
    handleOperator("+")
}

@IBAction func decimalTap(_ sender: Any)
{
    handleOperator(".")
}
```

```swift
@IBAction func zeroTap(_ sender: Any)
{
    addToWorkings(value: "0")
}

@IBAction func oneTap(_ sender: Any)
{
    addToWorkings(value: "1")
}

@IBAction func twoTap(_ sender: Any)
{
    addToWorkings(value: "2")
}

@IBAction func threeTap(_ sender: Any)
{
    addToWorkings(value: "3")
}

@IBAction func fourTap(_ sender: Any)
{
    addToWorkings(value: "4")
}

@IBAction func fiveTap(_ sender: Any)
{
    addToWorkings(value: "5")
}

@IBAction func sixTap(_ sender: Any)
{
    addToWorkings(value: "6")
}

@IBAction func sevenTap(_ sender: Any)
{
    addToWorkings(value: "7")
}

@IBAction func eightTap(_ sender: Any)
{
    addToWorkings(value: "8")
```


5+2
7
AC
%
/
1
2
3
+
4
5
6
-
7
8
9
X
.
0
=


iPhone 16 Pro – iOS 18.4
11:55
Calc
AC
%
/
1
2
3
+
4
5
6
-
7
8
9
X
.
0
=

PROJECT DIFFICULTIES



During development, We faced challenges with operator handling, ensuring valid input without repeated or misplaced symbols. Auto Layout required careful constraint adjustments for different screen sizes, especially on iPad. Configuring Stack Views to maintain equal spacing and circular buttons was tricky, but improved my understanding of dynamic UI design in UIKit.

THANK YOU!