### **Lab 14-** **Create Custom Button in Swift**

To create a reusable button style using a custom class in Swift, you can subclass UIButton and customize the appearance in this subclass. This way, you can easily create multiple buttons with the same style without duplicating code.

Here’s how to do it:

**Step 1: Create a Custom Button Class**

* Create a new Swift file named CustomButton.swift.
* Define the custom button class by subclassing UIButton.

import UIKit

class CustomButton: UIButton {

// Custom initializer

override init(frame: CGRect) {

super.init(frame: frame)

setupButton()

}

required init?(coder: NSCoder) {

super.init(coder: coder)

setupButton()

}

// Custom setup for the button

private func setupButton() {

// Set button title for different states

setTitle("Tap me", for: .normal)

setTitle("Tapped", for: .highlighted)

// Set title color for different states

setTitleColor(.white, for: .normal)

setTitleColor(.gray, for: .highlighted)

// Set button font

titleLabel?.font = UIFont.systemFont(ofSize: 18, weight: .bold)

// Set button background color

backgroundColor = UIColor.systemBlue

// Set button corner radius

layer.cornerRadius = 10

// Set button border

layer.borderWidth = 2

layer.borderColor = UIColor.white.cgColor

// Add shadow

layer.shadowColor = UIColor.black.cgColor

layer.shadowOpacity = 0.5

layer.shadowOffset = CGSize(width: 2, height: 2)

layer.shadowRadius = 5

}

}

**Step 2: Use the Custom Button Class in a View Controller**

* Modify your view controller to use the CustomButton class.

import UIKit

class ViewController: UIViewController {

override func viewDidLoad() {

super.viewDidLoad()

// Create an instance of the custom button

let customButton = CustomButton(frame: CGRect(x: 100, y: 100, width: 200, height: 50))

// Add the button to the view

self.view.addSubview(customButton)

// Add target-action pair for the button

customButton.addTarget(self, action: #selector(buttonTapped), for: .touchUpInside)

}

// Define the action method

@objc func buttonTapped() {

print("Button was tapped!")

}

}

**Benefits**

* Reusability: The custom button class can be used anywhere in the project, ensuring a consistent style without repeating code.
* Maintainability: Any style changes need to be made only in the custom class, and they will be reflected across the entire project.
* Readability: The view controller code is cleaner and more focused on layout and logic rather than styling details.

By using this approach, you can easily maintain a consistent button style throughout your app and make your codebase more organized and maintainable..