



## Actividad 3 Programa 2 Parte 2

---

### Desarrollo de Aplicaciones Móviles IV

Ingeniería en Desarrollo de Software



TUTOR: Marco Alonso Rodríguez Tapia

ALUMNO: Homero Ramirez Hurtado

FECHA: 10 de Abril del 2025

Índice.

. Introducción.

. Descripción.

. Justificación.

. Desarrollo.

- Codificación.
- Prueba del Programa.

. Conclusión.

. Referencias.

## Introducción.

La segunda fase de este proyecto de banca en línea para el Banco Mexicano se centrará en la optimización y expansión de funcionalidades clave, garantizando una mayor seguridad y eficiencia en las operaciones financieras. Después de establecer una base sólida con Swift, se implementarán mejoras en el sistema, asegurando que los usuarios puedan interactuar con la plataforma de manera intuitiva y sin inconvenientes.

Uno de los principales objetivos en esta etapa será fortalecer la seguridad del sistema, integrando mecanismos de verificación adicionales y optimizando la gestión de datos. Esto garantizará que cada transacción, ya sea depósito, retiro o consulta de saldo, se realice con total confianza y protección. Además, se mejorará la interfaz de usuario para que la navegación sea más fluida y accesible.

Otro aspecto clave será la compatibilidad con futuras actualizaciones y nuevas funciones, permitiendo que la aplicación evolucione según las necesidades del banco y sus clientes. Con estas mejoras, el Banco Mexicano reforzará su presencia digital, ofreciendo una solución bancaria moderna y eficiente que se alinea con las exigencias del mercado financiero actual.

## Descripción.

En esta fase del desarrollo del programa de banca en línea del Banco Mexicano, se completarán las funciones restantes del menú principal: "Saldo" y "Salir". Estas opciones garantizarán una experiencia de usuario más completa y permitirán a los clientes gestionar sus cuentas con mayor facilidad.

Cuando el usuario seleccione la opción "Saldo", el sistema mostrará el monto disponible en la cuenta, brindando una visión clara de sus finanzas. Además, se preguntará si desea realizar otra operación, ofreciendo una navegación fluida y evitando la necesidad de regresar manualmente al menú principal.

Por otro lado, la opción "Salir" permitirá a los usuarios cerrar sesión de manera segura. Al elegir esta opción, el programa mostrará un mensaje de confirmación indicando que la sesión ha finalizado correctamente. Esto garantizará la protección de la información del cliente y evitará accesos no autorizados.

Con la implementación de estas funciones, el programa de banca en línea mejorará su utilidad y seguridad, brindando a los usuarios una plataforma intuitiva y eficiente para gestionar sus transacciones de manera segura y organizada.

Justificación.

Replit es un entorno de desarrollo integrado (IDE) en línea que ofrece una plataforma versátil y accesible para la creación de aplicaciones en diferentes lenguajes de programación, incluyendo Swift. Para el desarrollo de este programa de banca en línea del Banco Mexicano, Replit es una opción ideal debido a su facilidad de uso, compatibilidad y herramientas integradas.

Uno de los principales beneficios de utilizar Replit es que permite escribir y ejecutar código directamente desde el navegador, eliminando la necesidad de instalar software adicional en el dispositivo del usuario. Esto facilita la colaboración entre desarrolladores, ya que múltiples personas pueden trabajar en el mismo proyecto en tiempo real, agilizando el proceso de desarrollo.

Además, Replit ofrece ejecución en tiempo real, permitiendo identificar y corregir errores de manera rápida y eficiente. Su integración con herramientas de control de versiones, como Git, es clave para mantener un registro organizado de los cambios realizados en el código.

Gracias a su accesibilidad y funcionalidad, Replit es una excelente opción para este proyecto, proporcionando un entorno confiable donde los desarrolladores pueden trabajar de manera eficiente en la creación de la banca en línea del Banco Mexicano.

Desarrollo.

Codificación.

```
import Foundation
```

```
class BancoMexicano {  
    var saldo: Double = 0.0  
    var primeraVez: Bool = true  
  
    func iniciar() {  
        while true {  
            print("\nBienvenido al Banco Mexicano")  
            print("1. Depósito")  
            print("2. Retiro")  
            print("3. Saldo")  
            print("4. Salir")  
            print("Ingrese una opción:")  
  
            guard let opcion = readLine(), let opcionInt = Int(opcion) else {
```

```

        print("Entrada no válida. Intente de nuevo.")
        continue
    }

    switch opcionInt {
    case 1:
        realizarDeposito()
    case 2:
        realizarRetiro()
    case 3:
        consultarSaldo()
    case 4:
        print("Sesión cerrada. Gracias por usar Banco Mexicano.")
        return
    default:
        print("Opción no válida. Intente de nuevo.")
    }
}
}

```

```

func realizarDeposito() {
    while true {
        print("Ingrese la cantidad a depositar:")

        guard let cantidad = readLine(), let cantidadDouble = Double(cantidad), cantidadDouble > 0
    else {
        print("Cantidad no válida. Intente de nuevo.")
        continue
    }
}

```

```

    saldo += cantidadDouble

    print("Depósito exitoso. Su saldo actual es: \"(saldo)\")

    print("¿Desea realizar otro depósito? (Sí/No)")

    if let respuesta = readLine(), respuesta.lowercased() == "no" {
        preguntarOtraOperacion()
        break
    }
}

func realizarRetiro() {
    if primeraVez {
        print("No cuenta con saldo en su primera operación.")
        primeraVez = false
        preguntarOtraOperacion()
        return
    }

    while true {
        print("Ingrese la cantidad a retirar:")

        guard let cantidad = readLine(), let cantidadDouble = Double(cantidad), cantidadDouble > 0
else {
        print("Cantidad no válida. Intente de nuevo.")
        continue
    }

    if cantidadDouble > saldo {
        print("Saldo insuficiente. Su saldo actual es: \"(saldo)\")

```

```

    } else {
        saldo -= cantidadDouble
        print("Retiro exitoso. Su saldo actual es: \$(saldo)")
    }

    print("¿Desea realizar otro retiro? (Sí/No)")
    if let respuesta = readLine(), respuesta.lowercased() == "no" {
        preguntarOtraOperacion()
        break
    }
}

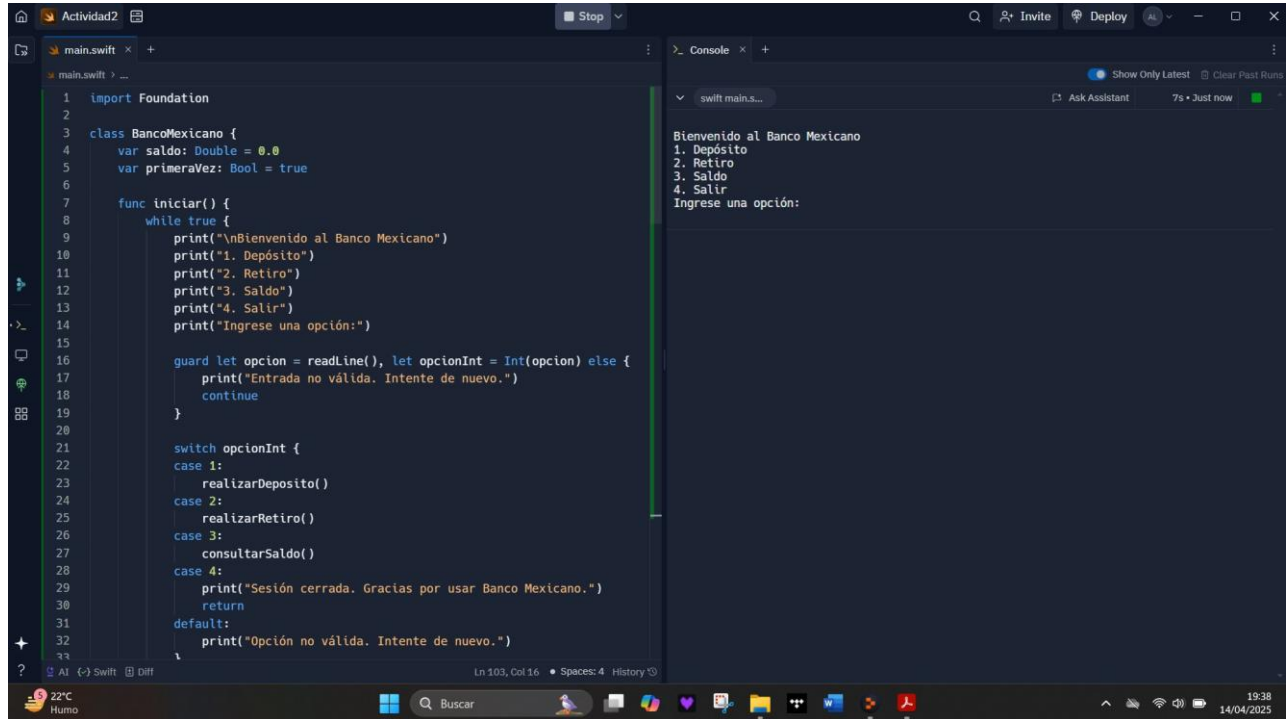
func consultarSaldo() {
    print("Su saldo actual es: \$(saldo)")
    preguntarOtraOperacion()
}

func preguntarOtraOperacion() {
    print("¿Desea realizar otra operación? (Sí/No)")
    if let respuesta = readLine(), respuesta.lowercased() == "no" {
        print("Sesión cerrada. Gracias por usar Banco Mexicano.")
        exit(0)
    }
}

let banco = BancoMexicano()
banco.iniciar()

```

## Prueba del Programa.



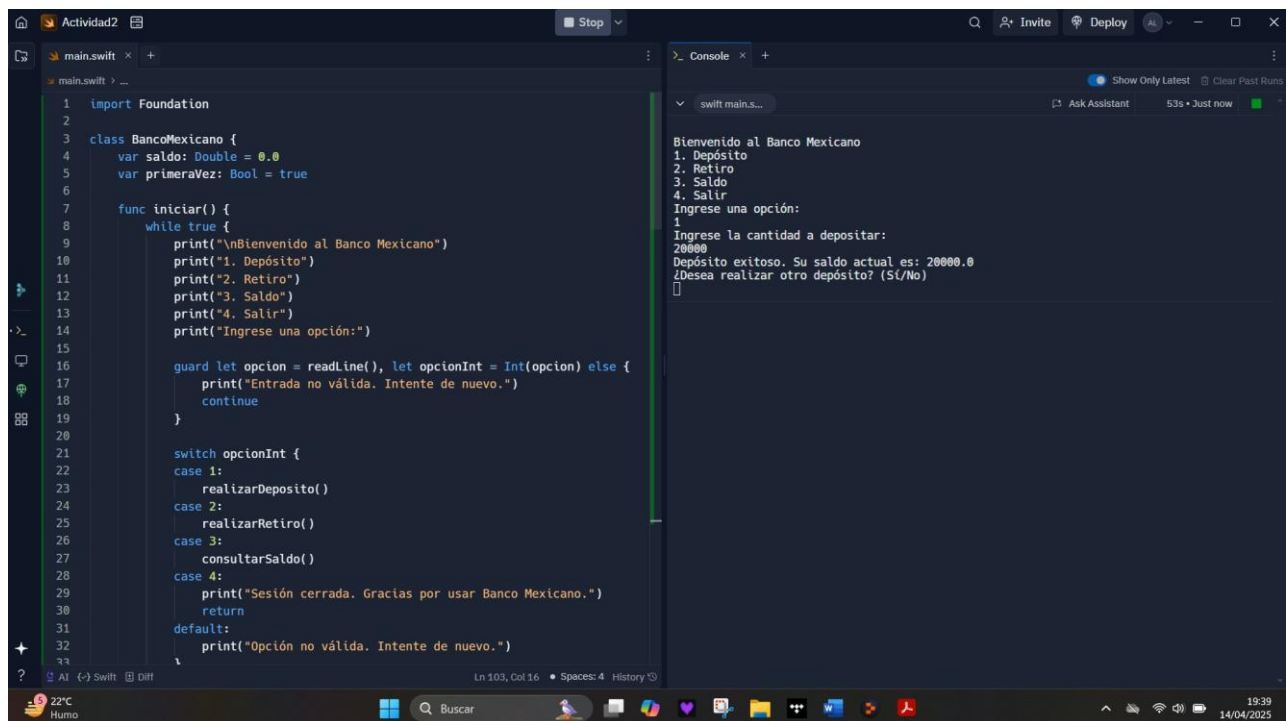
The screenshot shows the Xcode IDE with a Swift file named 'main.swift' and a console window. The code defines a 'BancoMexicano' class with a balance of 0.0 and a 'primeravez' flag. The 'iniciar()' function enters a loop that prints a welcome message and a menu with four options: 1. Depósito, 2. Retiro, 3. Saldo, and 4. Salir. It prompts the user to 'Ingrese una opción:'. The console output shows the program running and displaying the menu.

```
1 import Foundation
2
3 class BancoMexicano {
4     var saldo: Double = 0.0
5     var primeraVez: Bool = true
6
7     func iniciar() {
8         while true {
9             print("\nBienvenido al Banco Mexicano")
10            print("1. Depósito")
11            print("2. Retiro")
12            print("3. Saldo")
13            print("4. Salir")
14            print("Ingrese una opción:")
15
16            guard let opcion = readline(), let opcionInt = Int(opcion) else {
17                print("Entrada no válida. Intente de nuevo.")
18                continue
19            }
20
21            switch opcionInt {
22            case 1:
23                realizarDeposito()
24            case 2:
25                realizarRetiro()
26            case 3:
27                consultarSaldo()
28            case 4:
29                print("Sesión cerrada. Gracias por usar Banco Mexicano.")
30                return
31            default:
32                print("Opción no válida. Intente de nuevo.")
33            }
34        }
35    }
36
37    func realizarDeposito() {
38        // Implementación de depósito
39    }
40
41    func realizarRetiro() {
42        // Implementación de retiro
43    }
44
45    func consultarSaldo() {
46        // Implementación de consulta de saldo
47    }
48
49    func cerrarSesion() {
50        // Implementación de cierre de sesión
51    }
52}
```

Console Output:

```
Bienvenido al Banco Mexicano
1. Depósito
2. Retiro
3. Saldo
4. Salir
Ingrese una opción:
```

## Menú.



The screenshot shows the same Xcode IDE, but the console output now includes the result of a deposit operation. The user has entered '1' for 'Depósito', and the program prompts for the amount to deposit. The user enters '20000', and the program confirms the deposit and shows the current balance of 20000.0. It then asks if the user wants to perform another deposit.

```
1 import Foundation
2
3 class BancoMexicano {
4     var saldo: Double = 0.0
5     var primeraVez: Bool = true
6
7     func iniciar() {
8         while true {
9             print("\nBienvenido al Banco Mexicano")
10            print("1. Depósito")
11            print("2. Retiro")
12            print("3. Saldo")
13            print("4. Salir")
14            print("Ingrese una opción:")
15
16            guard let opcion = readline(), let opcionInt = Int(opcion) else {
17                print("Entrada no válida. Intente de nuevo.")
18                continue
19            }
20
21            switch opcionInt {
22            case 1:
23                realizarDeposito()
24            case 2:
25                realizarRetiro()
26            case 3:
27                consultarSaldo()
28            case 4:
29                print("Sesión cerrada. Gracias por usar Banco Mexicano.")
30                return
31            default:
32                print("Opción no válida. Intente de nuevo.")
33            }
34        }
35    }
36
37    func realizarDeposito() {
38        // Implementación de depósito
39    }
40
41    func realizarRetiro() {
42        // Implementación de retiro
43    }
44
45    func consultarSaldo() {
46        // Implementación de consulta de saldo
47    }
48
49    func cerrarSesion() {
50        // Implementación de cierre de sesión
51    }
52}
```

Console Output:

```
Bienvenido al Banco Mexicano
1. Depósito
2. Retiro
3. Saldo
4. Salir
Ingrese una opción:
1
Ingrese la cantidad a depositar:
20000
Depósito exitoso. Su saldo actual es: 20000.0
¿Desea realizar otro depósito? (S/N)

```

## Deposito.





The screenshot shows a Swift IDE with a file named 'main.swift'. The code implements a banking application with the following logic:

- It starts with a 'continue' statement in a loop.
- It calculates the new balance: `saldo += cantidadDouble`.
- It prints the updated balance: `print("Depósito exitoso. Su saldo actual es: \(saldo)")`.
- It asks if the user wants to perform another deposit: `print("¿Desea realizar otro depósito? (S/No)")`.
- It reads the user's response and checks if it's 'no'. If 'no', it calls `preguntarOtraOperacion()` and breaks the loop.
- It defines a function `realizarRetiro()` that checks if it's the first time using the app. If so, it prints a message and sets `primeraVez = false`. It then calls `preguntarOtraOperacion()` and returns.
- It enters a `while true` loop for withdrawals.
- Inside the loop, it prints 'Ingrese la cantidad a retirar:'.
- It uses a guard statement to validate the withdrawal amount: `guard let cantidad = readLine(), let cantidadDouble = Double(cantidad), cantidadDouble > 0 else {`. If invalid, it prints 'Cantidad no válida. Intente de nuevo.' and continues the loop.
- It checks if the withdrawal amount is greater than the current balance: `if cantidadDouble > saldo {`.

The console output shows the following sequence of events:

```
Bienvenido al Banco Mexicano
1. Depósito
2. Retiro
3. Saldo
4. Salir
Ingrese una opción:
1
Ingrese la cantidad a depositar:
15000
Depósito exitoso. Su saldo actual es: 15000.0
¿Desea realizar otro depósito? (S/No)
no
¿Desea realizar otra operación? (S/No)
si
Bienvenido al Banco Mexicano
1. Depósito
2. Retiro
3. Saldo
4. Salir
Ingrese una opción:
4
Sesión cerrada. Gracias por usar Banco Mexicano.
```

Salida.

Conclusión.

Finalizar este proyecto y ver cómo las cuatro opciones—**Depósito, Retiro, Saldo y Salir**—funcionan correctamente ha sido una gran satisfacción. Lo que comenzó como una idea se convirtió en un sistema funcional, con lógica estructurada y flujos bien definidos para ofrecer una experiencia bancaria fluida. Cada paso en la implementación fue una oportunidad de aprendizaje, desde el manejo de condiciones hasta la validación de operaciones, asegurando que la aplicación responda de manera precisa y segura.

Más allá del código escrito, este proceso ha sido un recordatorio de lo valioso que es construir algo desde cero y perfeccionarlo con cada prueba y ajuste. Resolver desafíos, depurar errores y ver cómo cada función cobra vida ha hecho que este desarrollo sea más que solo programación; ha sido una experiencia de crecimiento y creatividad.

Este proyecto representa no solo el logro de una meta técnica, sino también el esfuerzo y dedicación detrás de cada línea de código. Y con esta última función integrada, la aplicación ahora tiene una base sólida para seguir mejorando y evolucionando.

Referencias.

Tutoría 3.

Copilot.