Universidad Nacional Autónoma de México

Matemáticas Aplicadas y Computación

Temas Selectos de Computacion

Práctica "Piedra, Papel Tjeras, Lagarto y Spock"

Profesor

Eduardo Eloy Loza Pacheco

Alumno

Hernández Espinoza Gabriel Emanuel

Indice

Objetivo

Desarrollar una aplicación interactiva en iOS mediante Xcode que implemente el juego 'Piedra, Papel, Tijeras, Lagarto, Spock', utilizando estructuras de control, enumeraciones y lógica condicional para determinar el resultado entre el usuario y el oponente.

Materiales y Métodos

Para el desarrollo de esta práctica se empleó una computadora Mac con sistema operativo macOS y el entorno de desarrollo integrado Xcode. Se creó un proyecto de tipo 'App' para iPhone, en el cual se diseñó una interfaz gráfica que incluía etiquetas (UILabel) y botones (UIButton) para permitir la interacción del usuario con las diferentes opciones del juego. El código fue implementado en el lenguaje Swift, haciendo uso de enumeraciones (enum) para representar las cinco posibles elecciones: Piedra, Papel, Tijeras, Lagarto y Spock. Se utilizó una función principal para gestionar las jugadas, comparando la elección del usuario con una elección generada aleatoriamente por el sistema. La aplicación fue ejecutada en el simulador de iPhone para verificar la correcta funcionalidad del programa y los resultados de cada partida.

Resumen

En esta práctica se desarrolló una aplicación móvil para iOS que simula el juego 'Piedra, Papel, Tijeras, Lagarto, Spock', con el objetivo de fortalecer el dominio de las estructuras de control, funciones, enumeraciones y manejo de lógica condicional en el lenguaje Swift. El proyecto fue implementado en el entorno Xcode, donde se diseñó una interfaz que permite al usuario seleccionar una de las cinco opciones disponibles mediante botones interactivos.

El sistema genera una elección aleatoria para el oponente utilizando el método randomElement() aplicado a un conjunto de valores definidos dentro de una enumeración. Posteriormente, una función de comparación evalúa las condiciones que determinan el resultado de la partida: victoria, derrota o empate. La aplicación despliega en pantalla tanto la elección del usuario como la del oponente, junto con un mensaje indicando el resultado.

Durante la práctica se aplicaron conceptos fundamentales de la programación orientada a objetos, la modularidad del código y el uso eficiente de las estructuras de decisión. Además, se reforzó la comprensión del patrón de diseño Modelo-Vista-Controlador (MVC), mediante la integración de la lógica del juego con los componentes de la interfaz gráfica. El resultado fue una aplicación funcional, dinámica y didáctica que demuestra la capacidad de Swift para manejar la aleatoriedad y la interacción en tiempo real dentro de aplicaciones móviles.

Antecedentes Teóricos

El desarrollo de aplicaciones interactivas requiere el dominio de estructuras de control, funciones y modelos de organización de código que permitan una ejecución ordenada y eficiente. En Swift, las enumeraciones (enum) son tipos de datos que permiten agrupar valores relacionados bajo un mismo contexto, proporcionando un mecanismo eficaz para representar conjuntos finitos de opciones. Estas estructuras, además de ser más seguras y legibles, facilitan la implementación de juegos o sistemas de decisión, como el de 'Piedra, Papel, Tijeras, Lagarto, Spock' [1].

La lógica de control en juegos de comparación se basa en estructuras condicionales (if-else y switch) que evalúan las relaciones entre las elecciones de los jugadores. Swift ofrece un modelo seguro y eficiente para este propósito, evitando errores comunes como la comparación de tipos incompatibles. Por otra parte, el uso de funciones parametrizadas permite modularizar el código, haciendo que las operaciones de evaluación, inicialización y resultado sean independientes, favoreciendo la mantenibilidad del programa [2].

El método randomElement(), parte de la biblioteca estándar de Swift, permite seleccionar elementos de una colección de forma aleatoria, lo cual resulta esencial para simular la elección del oponente en el juego. Este proceso introduce la aleatoriedad como componente clave en el desarrollo de experiencias interactivas [3].

Desde la perspectiva del diseño de interfaz, Apple enfatiza en sus Human Interface Guidelines (HIG) la importancia de la retroalimentación inmediata, la claridad visual y la consistencia. En esta aplicación, los mensajes de resultado y las etiquetas dinámicas cumplen con estos principios al proporcionar información clara y comprensible al usuario [4].

Desarrollo

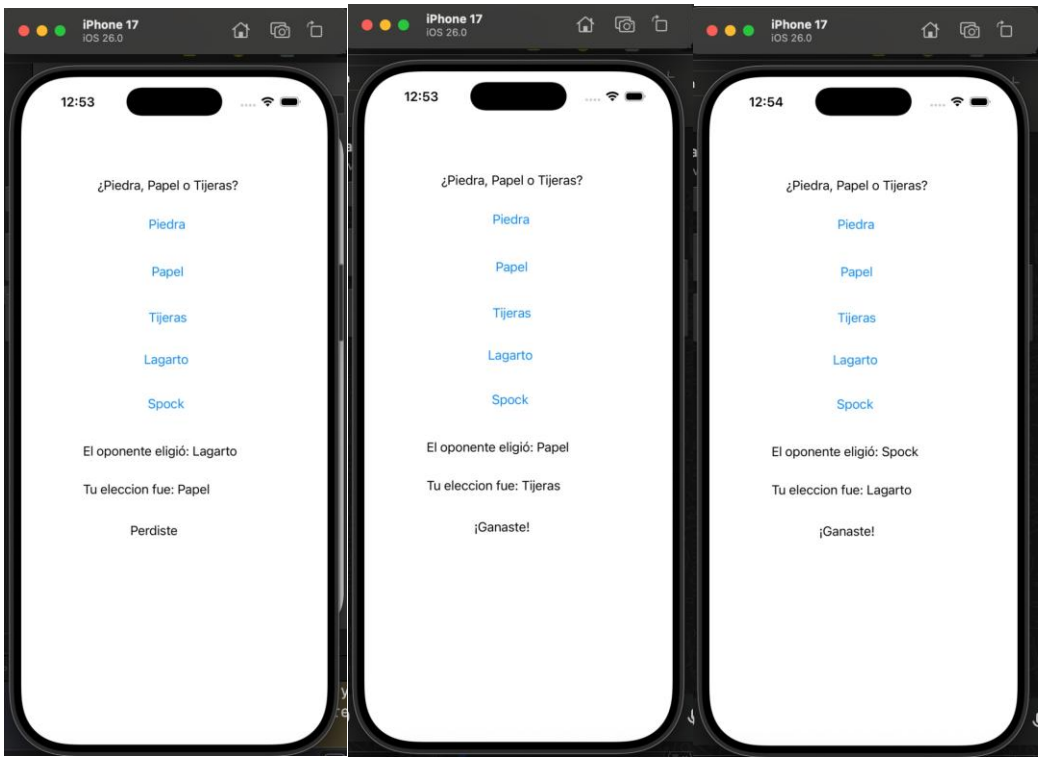
La práctica comenzó con la creación de un nuevo proyecto en Xcode bajo la plantilla 'App' para dispositivos iPhone. Dentro del storyboard se diseñó la interfaz de usuario, la cual incluyó tres etiquetas (UILabel) destinadas a mostrar la elección del jugador, la elección del oponente y el resultado de la partida. Asimismo, se añadieron cinco botones (UIButton), cada uno representando una de las opciones del juego: Piedra, Papel, Tijeras, Lagarto y Spock.

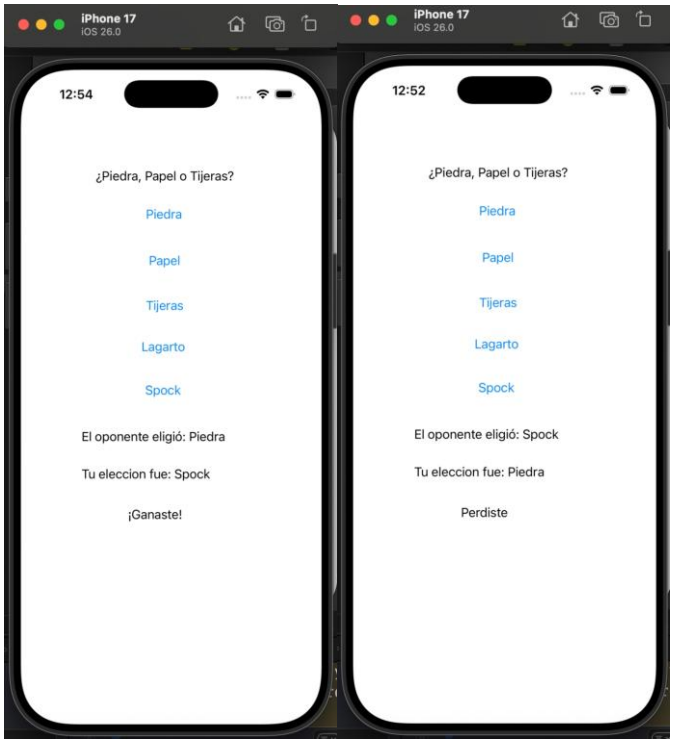
En el archivo ViewController.swift se implementó una enumeración denominada Eleccion, que agrupa los cinco posibles valores de jugada. Esta estructura, compatible con el protocolo CaseIterable, permitió generar selecciones aleatorias mediante el método allCases.randomElement(). Esta funcionalidad fue esencial para simular la jugada del oponente en cada turno.

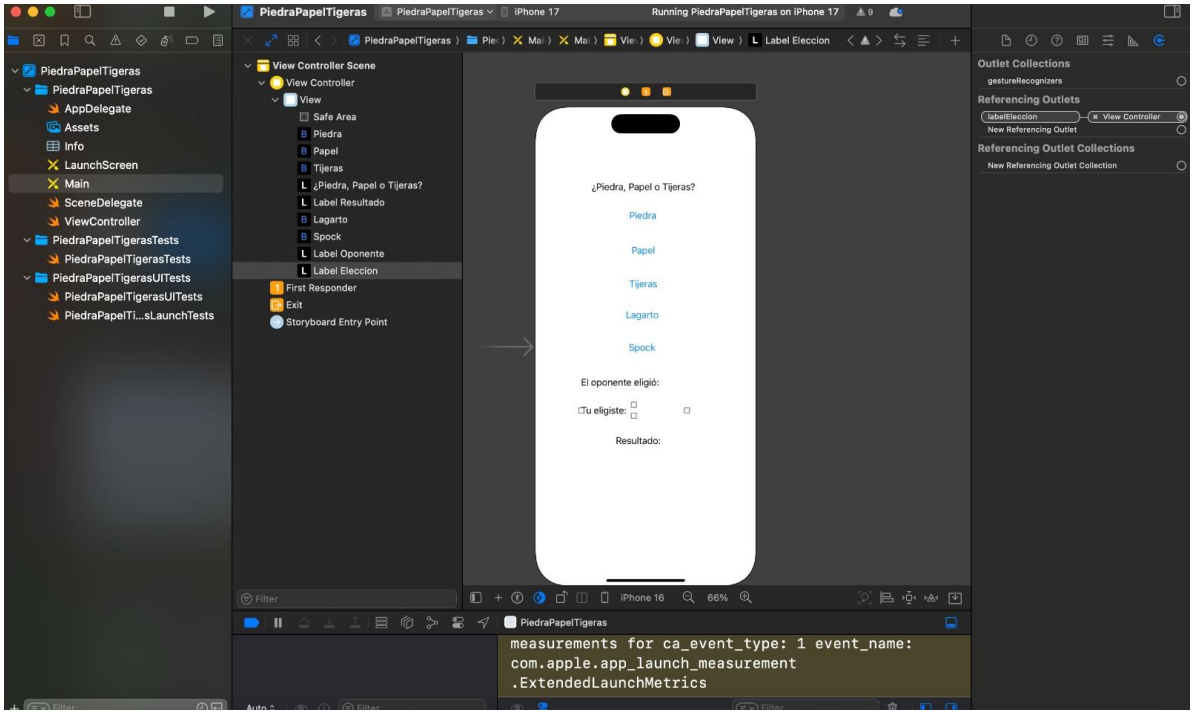
El núcleo lógico del programa se implementó en la función jugar(usuario:), que recibe como parámetro la elección del jugador. Dicha función obtiene la elección del oponente, actualiza las etiquetas correspondientes y determina el resultado de la partida. Para ello, invoca la función gana(jugador), que evalúa las combinaciones ganadoras mediante una estructura switch. Este método devuelve un valor booleano que indica si la elección del usuario vence a la del oponente según las reglas definidas.

Las reglas del juego se basan en la versión extendida de 'Piedra, Papel o Tijeras', que incluye dos nuevas opciones: Lagarto y Spock. En esta versión, Piedra vence a Tijeras y Lagarto; Papel vence a Piedra y Spock; Tijeras vence a Papel y Lagarto; Lagarto vence a Spock y Papel; y Spock vence a Tijeras y Piedra. Estas relaciones fueron codificadas explícitamente en la función gana().

Cada botón de la interfaz se vinculó con un método @IBAction que invoca la función jugar(usuario:) con el valor correspondiente de la enumeración. Esta modularidad permitió mantener un código limpio y evitar duplicidad de lógica. Además, se utilizaron etiquetas dinámicas para actualizar los textos de la interfaz en tiempo real, proporcionando una experiencia de usuario inmediata.


iPhone 17
iOS 26.0
12:53
¿Piedra, Papel o Tijeras?
Piedra
Papel
Tijeras
Lagarto
Spock
El oponente eligió: Lagarto
Tu eleccion fue: Papel
Perdiste
iPhone 17
iOS 26.0
12:53
¿Piedra, Papel o Tijeras?
Piedra
Papel
Tijeras
Lagarto
Spock
El oponente eligió: Papel
Tu eleccion fue: Tijeras
¡Ganaste!
iPhone 17
iOS 26.0
12:54
¿Piedra, Papel o Tijeras?
Piedra
Papel
Tijeras
Lagarto
Spock
El oponente eligió: Spock
Tu eleccion fue: Lagarto
¡Ganaste!


iPhone 17
iOS 26.0
12:54
¿Piedra, Papel o Tijeras?
Piedra
Papel
Tijeras
Lagarto
Spock
El oponente eligió: Piedra
Tu eleccion fue: Spock
¡Ganaste!
iPhone 17
iOS 26.0
12:52
¿Piedra, Papel o Tijeras?
Piedra
Papel
Tijeras
Lagarto
Spock
El oponente eligió: Spock
Tu eleccion fue: Piedra
Perdiste



Finalmente, la aplicación fue ejecutada en el simulador de iPhone para validar su funcionamiento. Las pruebas demostraron que la generación aleatoria, las comparaciones y la actualización visual de las etiquetas operaban correctamente. El resultado fue una aplicación completamente funcional, que simula con precisión el juego 'Piedra, Papel, Tijeras, Lagarto, Spock'.

Conclusiones

Esta práctica permitió consolidar el conocimiento en programación estructurada y orientada a objetos dentro del entorno Xcode, aplicando conceptos avanzados como enumeraciones, funciones parametrizadas y estructuras condicionales. A través de la implementación del juego 'Piedra, Papel, Tijeras, Lagarto, Spock', se comprendió cómo integrar lógica de negocio con elementos visuales para generar una aplicación interactiva y dinámica.

La utilización de enumeraciones facilitó la representación de las posibles elecciones y mejoró la legibilidad del código. Además, la generación aleatoria de la jugada del oponente permitió simular un comportamiento autónomo, añadiendo variabilidad a la aplicación. El empleo de la función gana() reforzó la comprensión de la estructura switch y de la programación basada en reglas.

Desde el punto de vista del diseño, se respetaron los principios de las Human Interface Guidelines de Apple, proporcionando una interfaz clara y una retroalimentación inmediata al usuario tras cada partida. El resultado fue un programa intuitivo, robusto y entretenido.

En conclusión, esta práctica representó un avance significativo en la comprensión del ciclo completo de desarrollo de una aplicación en iOS, integrando lógica, interfaz y experiencia de usuario. Los conocimientos adquiridos servirán como base para la creación de proyectos más complejos e interactivos en el futuro.

Referencias

[1] Apple, *Swift Programming Language Guide*. Apple Developer Documentation, 2023. [En línea]. Disponible en: https://docs.swift.org/swift-book/

[2] Apple, *Swift Standard Library*. Apple Developer Documentation, 2023. [En línea]. Disponible en: https://developer.apple.com/documentation/swift

[3] Apple, *Xcode Overview*. Apple Developer, 2023. [En línea]. Disponible en: https://developer.apple.com/xcode/

[4] Apple, *Human Interface Guidelines*. Apple Developer, 2022. [En línea]. Disponible en: https://developer.apple.com/design/human-interface-guidelines/