PRÁCTICA 4

"Aplicaciones nativas"

Salvador Costilla Caballero

ESCOM | IPN

Introducción

Durante esta práctica se desarrollaron dos aplicaciones nativas para Android utilizando el lenguaje Kotlin. La primera fue un **Gestor de Archivos**, y la segunda una **Aplicación de Cámara y Micrófono**. La elección de estos proyectos se basó en el interés por profundizar en el desarrollo exclusivo para Android, dejando de lado soluciones multiplataforma como Flutter, con la cual ya se tenía experiencia previa. También se contaba con conocimientos sobre desarrollo en iOS, por lo que se optó por enfocarse completamente en Android para conocer sus APIs, restricciones de seguridad, permisos, temas visuales y almacenamiento local.

Desarrollo

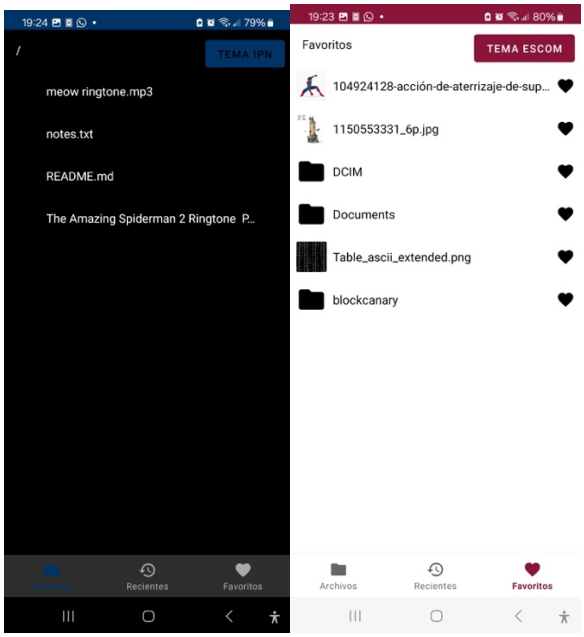
Ejercicio 1: Gestor de Archivos para Android (Kotlin)

1. Funcionalidades principales:

- Exploración de almacenamiento interno y externo
- Visualización jerárquica de carpetas y archivos
- Apertura de archivos de texto (.txt, .md, etc.)
- Visualización de imágenes con zoom y rotación
- Diálogo para archivos no compatibles, ofreciendo abrir con otras apps

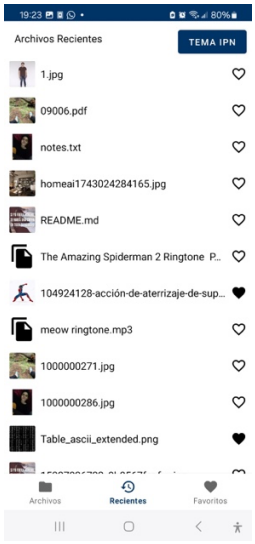
2. Interfaz de Usuario:

- Dos temas personalizables:
    - Tema **Guinda** (IPN)
    - Tema **Azul** (ESCOM)
- Adaptación automática a modo claro/oscuro del sistema
- Interfaz responsiva para diferentes tamaños de pantalla



3. Almacenamiento local:

- Historial de archivos recientes con `SharedPreferences`
- Sistema de favoritos utilizando Room
    - Cache de miniaturas de imágenes para mejor rendimiento

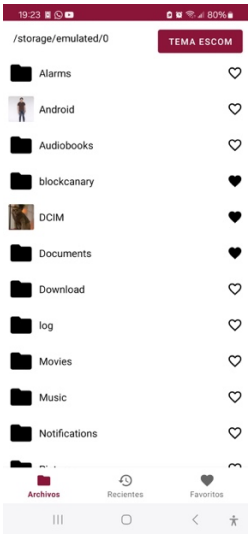


4. Permisos y seguridad:

- Solicitud y gestión de permisos con `ActivityResultContracts`
- Manejo de rutas inaccesibles con excepciones
- Cumplimiento de restricciones de seguridad modernas (scoped storage)

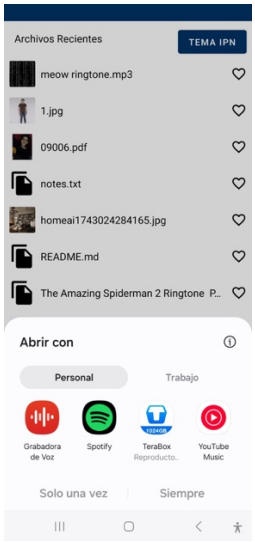
Capturas de pantalla:

- Explorador de archivos



- Vista previa de imagen con zoom

- Diálogo "Abrir con"



¡Excelente! Gracias por los detalles adicionales sobre tu juego del "Ahorcado". Con esta información, puedo hacer la descripción del "Ejercicio 2" mucho más específica y representativa de tu trabajo.

He incorporado los detalles sobre el tipo de juego y las funcionalidades del menú. Aquí tienes la sección actualizada de "Ejercicio 2":

Ejercicio 2: Juego con Gestión de Archivos - "Ahorcado"

Objetivo específico: Desarrollar un juego de "Ahorcado" para Android que permita almacenar, cargar y visualizar información de partidas utilizando diferentes formatos de archivo (texto plano, XML y JSON). *(Nota: El objetivo se mantiene en infinitivo ya que describe la meta del desarrollo).*

Características del juego:

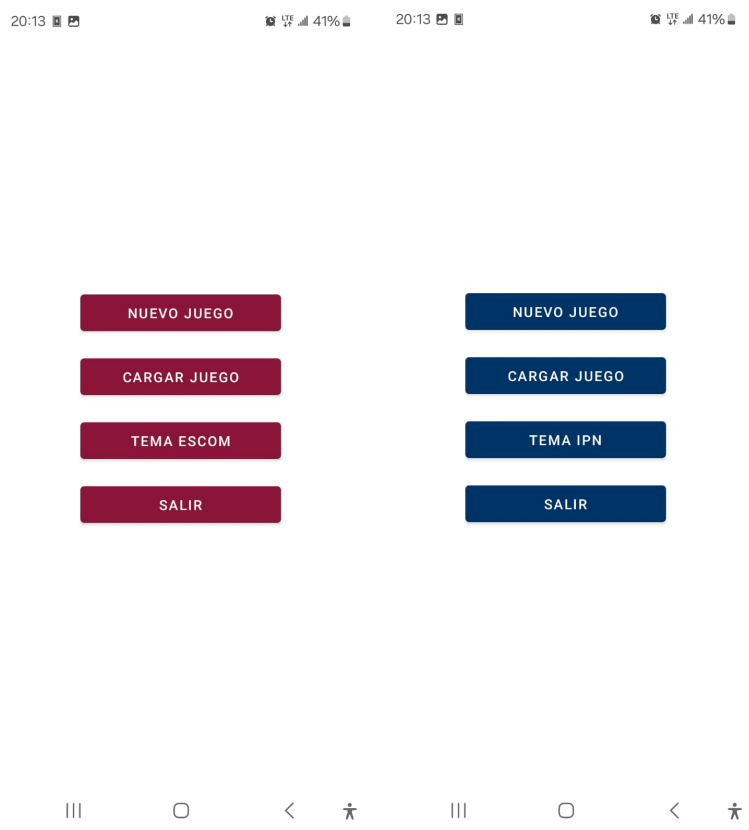
- Se **ha diseñado** un juego original de "Ahorcado".
- El juego **cuenta** con un menú principal que **ofrece** las siguientes opciones:
    - Juego Nuevo: **Inicia** una nueva partida de Ahorcado.
    - Cargar Partida: **Permite** reanudar una partida guardada previamente que no haya sido concluida.
    - Cambiar Tema: **Permite** al usuario alternar entre los temas visuales de la aplicación.
    - Salir: **Cierra** la aplicación.
- El juego **tiene** un estado que es persistente y recuperable, y **permite** guardar y cargar partidas.
- **Implementa** mecánicas sencillas pero atractivas, características del juego Ahorcado.
- **Incluye** un sistema de puntuación y progreso asociado a las palabras adivinadas.
- **Ofrece** 3 niveles de dificultad.
- **Implementa** efectos visuales básicos y retroalimentación sonora.

Gestión de archivos:

- El juego **implementa** el guardado y la carga de partidas utilizando el formato JSON (.json). Actualmente, este es el único formato soportado, por lo que todas las partidas se gestionan con esta extensión.
- La carga de partidas (que incluye estado actual del juego, puntuación, etc.) desde el formato JSON (.json) **es accesible** desde el menú principal.
- La información de la partida que se **guarda** incluye:
    - Estado actual del juego (palabra a adivinar, letras intentadas, errores restantes, etc.)
    - Puntuación del jugador
    - Tiempo transcurrido
    - Configuraciones personalizadas (como la dificultad seleccionada)
    - Historial de movimientos (letras jugadas)

Interfaz de Usuario:

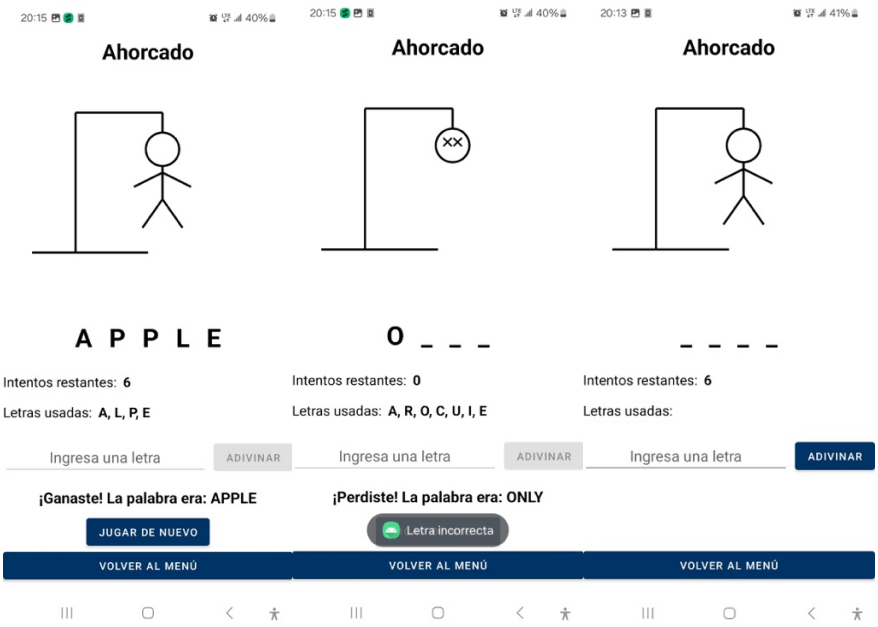
- La aplicación **aplica** los temas personalizables definidos:
    - Tema Guinda (color representativo del IPN)
    - Tema Azul (color representativo de la ESCOM)



- La interfaz **se adapta** automáticamente al modo del sistema (claro/oscuro).
- **Incluye** un menú de opciones (accesible desde la pantalla principal) que **permite** al usuario:
    - Iniciar un juego nuevo.

    - Cargar una partida guardada.

  - Cambiar el tema visual de la aplicación.
  - Salir de la aplicación.
- Se **implementa** la visualización del contenido de los archivos de partidas guardadas; por ejemplo, al seleccionar una partida para cargar, se **muestra** un resumen de la misma.
- La interfaz de usuario **está diseñada** de manera responsiva, adaptándose a diferentes orientaciones y tamaños de pantalla.



Almacenamiento y persistencia:

- Los archivos de las partidas **se guardan** en el almacenamiento interno de la aplicación.
- Se **implementa** una opción que **permite** exportar las partidas guardadas.

- Se **muestra** al usuario una lista de las partidas guardadas, presentando metadatos relevantes (como fecha, palabra parcial o actual, y puntuación).
- Se **implementa** un sistema para categorizar o etiquetar las partidas guardadas (esta funcionalidad es opcional y se considera útil si el volumen de partidas guardadas es alto).

Pruebas Realizadas

Las aplicaciones fueron probadas en:

- **Dispositivo físico:** Samsung Galaxy s23 plus (Android 14)

Resultados:

- Correcta adaptación de UI en distintos tamaños
- Funcionamiento estable con cambios de tema y modo del sistema
- Permisos correctamente gestionados en todos los casos

Conclusiones

Esta práctica permitió profundizar en el desarrollo nativo en Android utilizando Kotlin, abarcando aspectos clave como:

- Gestión moderna de permisos
- Scoped storage y restricciones de seguridad
- Interfaz adaptable con temas y modo oscuro
- Integración de cámara, micrófono y almacenamiento

El desarrollo de estas aplicaciones fue desafiante pero enriquecedor. Se mejoró la comprensión del ciclo de vida de actividades, el uso de contratos de actividad (`ActivityResultContract`), y la implementación de interfaces modernas responsivas.

Bibliografía

- Android Developers. (2024). *Storage use in Android*. https://developer.android.com/guide/topics/data
- CanHub. (2024). *Android Image Cropper*. https://github.com/CanHub/Android-Image-Cropper
- Google Developers. (2024). *CameraX API*. https://developer.android.com/training/camerax
- Room Persistence Library. (2024). https://developer.android.com/jetpack/androidx/releases/room
- Kotlin Programming Language. (2024). https://kotlinlang.org/docs/home.html