UNIVERSIDADE DA CORUÑA

PROGRAMACIÓN DE SISTEMA 23/24 Q2

Simon Says para android

Autores:
Belén Domínguez Álvarez: belen.domingueza
Adrian Ojea Gonçalves: a.ojea.goncalves
Alejandro López Fabeiro: alejandro.lopez.fabeiro
Fecha:*A Coruña, 15 Marzo 2024*

Índice

Capítulos **Página**

1. Introducción **1**
1.1. Objetivos 1
1.2. Motivación 1
1.3. Trabajo relacionado 1

2. Análisis de requisitos **2**
2.1. Funcionalidades 2
2.2. Prioridades 3

3. Planificación inicial **4**
3.1. Iteraciones 4
3.2. Responsabilidades 4
3.3. Hitos 5
3.4. Incidencias 5

4. Diseño **6**
4.1. Arquitectura 6
4.2. Persistencia 7
4.3. Vistas 8
4.4. Comunicaciones 8

5. Bibliografía **9**

Cuadro 1: Tabla de versiones.

Versión	Fecha	Autor
x	y	
x	y	
x	y	

1. Introducción

1.1. Objetivos

El objetivo de este proyecto es desarrollar una aplicación de android interactiva que aplique las reglas del juego "Simón dice" o "Simon says". Este juego pone a prueba la memoria y la capacidad de atención atención de los jugadores. Para ello se hace uso de una secuencia de luces y sonidos generada por el juego, de manera aleatoria o por un jugador.

En su modo más simple el juego consiste en una serie de rondas en las que Simón (la máquina) emite una secuencia creciente que los jugadores deben repetir en exactamente en el mismo orden. Cada vez que los jugadores repliquen la secuencia correctamente esta se alarga, agregando un nuevo elemento y sumándose a su puntuación total. Si el jugador comete un error al repetir la secuencia, el juego termina, devolviendo la puntuación obtenida hasta ese momento.

Sumado a este objetivo, también se busca que el juego tenga un sistema de multijugador, de tal manera que puedas competir con amigos para ver quién consigue mayor puntuación.

Como un objetivo secundario se busca encontrar nuevas acciones con las que interactuar en el juego a mayores de la simple pulsación de un botón en la pantalla, como podría ser el giro del teléfono mediante uso del giroscopio.

1.2. Motivación

Al ser el "Simon Saysün juego muy conocido y sencillo, al menos en sus mecánicas, resulta mucho más sencillo centrar el esfuerzo en el desarrollo del sistema multijugador y añadir posibles nuevas mecánicas, algo que resulta parte fundamental del proyecto.

Por otro lado el juego, como se ha mencionado en el párrafo anterior, a pesar de contar con mecánicas simples, puede adquirir un alto nivel de dificultad, pues las secuencias se hacen cada vez más largas y complejas por lo que aumenta la necesidad por parte de los jugadores de prestar atención y la capacidad de concentración de los mismos. Por lo que puede ayudar a ejercitar habilidades cognitivas básicas.

Además este tipo de proyecto servirá para adquirir conocimientos prácticos en partes básicas de la interacción con un móvil, como son los sensores, interacciones extra con la pantalla y accesos de android.

1.3. Trabajo relacionado

El juego "Simón dice" es un juego bastante extendido y popular. Ha parte de la versión física, que normalmente consiste en un disco con 4 luces de colores, también existen numerosas implementaciones: ya sea en consolas como un juego único o siendo un "minijuego.[en] algún juego más extenso, en plataformas online de juegos, etc.

El mismo también cuenta con multitud de variaciones, hay algunos que utilizan palabras, otros que utilizan gestos o posturas, e incluso combinaciones.

Comparados con estos proyectos, nuestro objetivo es hacer un juego bastante más parecido al original, con cuatro posibles colores. (Añadir requisitos)

2. Análisis de requisitos

2.1. Funcionalidades

- Modos de Juego
  - Modo clásico: El modo tradicional de Simon Says donde los jugadores repiten secuencias de colores cada vez más largas.
  - Otros modos.
- Funcionalidades básicas:
  - Modo multijugador: La aplicación debe permitir a múltiples jugadores conectarse y jugar simultáneamente.
  - Sistema de emparejamiento: Un sistema que permita emparejar a los jugadores de manera rápida. (Salas de juego o emparejamiento)
  - Sincronización en tiempo real: La sincronización precisa entre los dispositivos de todos los jugadores para asegurar que las secuencias de colores y las acciones se reproduzcan correctamente.
  - Mecanismo de turnos: Un sistema que gestione los turnos de los jugadores de manera justa, asegurando que cada uno tenga la oportunidad de repetir la secuencia correctamente.
  - Sistema de puntuación: Un sistema que registre y muestre las puntuaciones de los jugadores, ya sea basado en la precisión de la repetición de las secuencias o en otros criterios.
- Funcionalidades del Menú
  - Control de volumen: Permitir a los jugadores ajustar el volumen de la música de fondo, los efectos de sonido y las notificaciones de la aplicación.
  - Configuración de idioma: Ofrecer la posibilidad de elegir el idioma de la interfaz de la aplicación para adaptarse a las preferencias de los jugadores.
  - Tutorial o ayuda: Incluir un tutorial o sección de ayuda en el menú de ajustes para que los jugadores puedan revisar las instrucciones del juego, consejos para mejorar, solución de problemas, etc.
- Funcionalidades Extra

- Animaciones y efectos visuales: Animaciones claras y efectos visuales que muestren las secuencias de colores y las respuestas de los jugadores.
- Musica de fondo y efectos de sonidos: Musica simple de fondo que no distraiga al jugador y efectos de sonido que ayuden a reforzar los estímulos visuales.

2.2. Prioridades

- Funcionalidades Core:
  - Funcionalidades Básicas
  - Modo Clásico de los Módulos de juego
- Funcionalidades Accesorio:
  - Otros módulos de juego
  - Funcionalidades del menú
  - Funcionalidades Extra

Una vez se tienen definidas tanto las funcionalidades core, como las funcionalidades accesorio. debemos establecer el orden más coherente para la implementación del mismo.

1. Modo básico del Simon Says.
2. Sistema de puntuación
3. Modo Multijugador.
4. Sincronización en tiempo real.
5. Mecanismo de turnos

Con esto se tendría la implementación de las funcionalidades Core del mismo. A excepción del sistema de puntuación, que podría implementarse en cualquier momento, se considera que una funcionalidad no se puede implementar sin haberse implementado la anterior en la lista.

1. Animaciones y efectos visuales
2. Musica y efectos de Sonido
3. Control de volumen
4. Tutorial
5. Configuración de Idiomas

La implementación de las funcionalidades accesorio podrían realizarse en casi cualquier punto del proyecto, sin embargo se considera que primero se deberían implementar las Animaciones y efectos visuales junto a la Musica y efectos de sonido. El control de volumen se podría implementar al mismo tiempo que la parte auditiva. Tanto el Tutorial como la configuración de idioma se podrían realizar en paralelo a la implementación de las funcionalidades de las que dependen.

3. Planificación inicial

3.1. Iteraciones

- **Iteración 1**: Funcionalidades Básicas
  - Implementar el modo clásico del juego.
  - Configuración de idioma y control de volumen.
  - Tutorial o sección de ayuda.
- **Iteración 2**: Multijugador y Emparejamiento
  - Desarrollar el modo multijugador.
  - Implementar el sistema de emparejamiento.
  - Realizar pruebas de sincronización en tiempo real.
- **Iteración 3**: Turnos y Puntuación
  - Integrar el mecanismo de turnos.
  - Diseñar y añadir el sistema de puntuación.
- **Iteración 4**: Funcionalidades Extra
  - Incluir música de fondo y efectos de sonido.
  - Realizar pruebas de integración

3.2. Responsabilidades

- Belén: Diseño de interfaz y gestión de documentación.
- Adrián: Diseño de bajo nivel y gestión de documentación.
- Alejandro: Diseño de alto nivel y gestión de documentación.

3.3. Hitos

- **Hito 1: Fin de Iteración 1**
  - Entregable: Aplicación con modo clásico, configuración de idioma y control de volumen funcional.
- **Hito 2: Fin de Iteración 2**
  - Entregable: Aplicación con modo multijugador, sistema de emparejamiento y pruebas de sincronización.
- **Hito 3: Fin de Iteración 3**
  - Entregable: Aplicación con mecanismo de turnos, sistema de puntuación y funcionalidades multijugador mejoradas.
- **Hito 4: Fin de Iteración 4**
  - Entregable: Aplicación con todas las funcionalidades básicas y extra implementadas y probadas.

3.4. Incidencias

- **Incidencia: Problemas de sincronización en tiempo real.**
  - Plan de Contingencia: Investigar y utilizar bibliotecas o tecnologías probadas para la sincronización en tiempo real.
- **Incidencia: Dificultades en la implementación del sistema de puntuación.**
  - Plan de Contingencia: Simplificar el sistema de puntuación inicialmente y luego iterar para mejorarlo según sea necesario.
- **Incidencia: Problemas de compatibilidad con diferentes dispositivos Android.**
  - Plan de Contingencia: Realizar pruebas exhaustivas en una variedad de dispositivos Android populares y solucionar cualquier problema de compatibilidad que surja. Utilizar las herramientas de desarrollo de Android para identificar y resolver problemas de rendimiento específicos del dispositivo.

4. Diseño

4.1. Arquitectura

Se ha intentado crear una arquitectura basica que pueda usarse para implementar el juego de Simon Says de una forma básica pero que pueda entenderse facilmente y no resulte demasiado compleja.

- Actividades:
  - Actividad Principal (MainActivity): Esta actividad gestionaría la interfaz de usuario principal de la aplicación, mostrando opciones para jugar, acceder al menú de ajustes, ver puntuaciones, etc.
  - Actividad de Juego (GameActivity): Encargada de mostrar la secuencia de colores y permitir a los jugadores repetirla en modo multijugador.
  - Actividad de Configuración (SettingsActivity): Donde los usuarios pueden personalizar sus preferencias, como sonido, idioma, notificaciones, etc.
- Servicios:
  - Servicio de Multijugador (MultiplayerService): Maneja la lógica de juego multijugador, gestiona la comunicación en tiempo real entre los jugadores y sincroniza las acciones en todas las instancias del juego.
  - Servicio de Notificaciones (NotificationService): Encargado de gestionar las notificaciones, como invitaciones de juego, recordatorios, etc.
- Receptores de difusión:
  - Receptor de Notificaciones (NotificationReceiver): Escucha las notificaciones del sistema y las procesa, por ejemplo, mostrando un mensaje emergente cuando llega una invitación de juego.
- Hilos:
  - Hilo de la Interfaz de Usuario (UI Thread): Responsable de manejar la interfaz de usuario y responder a las interacciones del usuario.
  - Hilos de Trabajo en Segundo Plano (Background Threads): Se utilizan para llevar a cabo tareas que no deben bloquear el hilo de la interfaz de usuario, como la comunicación de red, el procesamiento de datos y la lógica del juego.
- Base de Datos:
  - Base de Datos Local (Local Database): Almacena información localmente, como puntuaciones, configuraciones de usuario, etc.

- Red:
    - Capa de Comunicación en Red (Network Communication Layer): Gestiona la comunicación entre los dispositivos de los jugadores en tiempo real
- Componentes de Interfaz de Usuario:
    - Botones, vistas de texto, animaciones y otros elementos de la interfaz de usuario utilizados para interactuar con el juego y mostrar información al usuario.

4.2. Persistencia

Para la persistencia en una aplicación de Simon Says con modo multijugador, necesitaremos almacenar varios tipos de datos para proporcionar una experiencia completa y funcional.

- Puntuaciones (Scores):
    - Datos: Las puntuaciones de los jugadores en diferentes partidas.
    - Gestión: Se pueden almacenar en una base de datos local, como SQLite, utilizando una tabla con campos para el nombre del jugador, la puntuación y la fecha/hora de la partida. Se pueden proporcionar métodos para guardar nuevas puntuaciones, recuperar las puntuaciones más altas y borrar registros antiguos si es necesario.
- Configuración de Usuario (User Settings):
    - Datos: Preferencias personalizadas de los usuarios, como idioma, sonido, notificaciones, etc.
    - Gestión: Estos datos se pueden almacenar en Shared Preferences para un acceso rápido y sencillo. Se puede proporcionar una clase de utilidad para acceder y modificar estas preferencias según las elecciones del usuario.
- Estado del Juego (Game State):
    - Datos: Información sobre el estado actual del juego en curso, como la secuencia actual de colores, los jugadores participantes, las rondas completadas, etc.
    - Gestión: Se puede utilizar una clase de utilidad para guardar y cargar el estado del juego.

Para diseñar las vistas de una aplicación de Simon Says con modo multijugador, necesitaríamos considerar varios elementos de la interfaz de usuario, como actividades, fragmentos, notificaciones y la navegación entre ellos. A continuacion se expondra una posible propuesta de cómo podrían estructurarse:

4.3. Vistas

- Actividades (Activities):
  - Actividad Principal (MainActivity): Esta actividad sería la pantalla de inicio de la aplicación. Podría mostrar opciones para jugar en modo multijugador, acceder a configuraciones, ver puntuaciones, etc.
  - Actividad de Juego (GameActivity): Aquí es donde se lleva a cabo el juego real. Mostraría la secuencia de colores y permitiría a los jugadores repetirla en modo multijugador.
  - Actividad de Configuración (SettingsActivity): Los jugadores podrían acceder a esta actividad para personalizar su experiencia de juego, como ajustes de sonido, idioma, notificaciones, etc.
- Fragmentos (Fragments):
  - Fragmento de Puntuaciones (ScoresFragment): Muestra las puntuaciones más altas de los jugadores. Los jugadores podrían ver sus propias puntuaciones y compararlas con las de otros jugadores.
  - Fragmento de Ajustes (SettingsFragment): Presenta las opciones de configuración de la aplicación de manera más detallada, permitiendo a los jugadores ajustar diferentes aspectos de la aplicación.
- Notificaciones (Notifications):
  - Notificación de Invitación de Juego: Cuando un jugador recibe una invitación de juego, se mostraría una notificación emergente para informarle y permitirle aceptar o rechazar la invitación.
- Navegación (Navigation):
  - Para la navegación entre las diferentes actividades y fragmentos, se podría utilizar el componente Navigation de Android Jetpack. Esto facilita la implementación de un flujo de navegación coherente y predecible en la aplicación, con animaciones y transiciones fluidas entre las pantallas.
  - También se podrían incluir botones y menús de navegación en la interfaz de usuario para permitir a los usuarios moverse fácilmente entre las distintas secciones de la aplicación.

4.4. Comunicaciones

- Servidor de Juego (Game Server):
  - Se necesitaría un servidor de juegos para gestionar las partidas multijugador. Este servidor sería responsable de coordinar las acciones entre los jugadores, sincronizar el estado del juego, y manejar la lógica del juego.

- Protocolo de Comunicación
    - Se necesitaría un protocolo de comunicación para que los dispositivos de los jugadores se comuniquen con el servidor de juegos. Esto podría ser implementado utilizando HTTP para solicitudes RESTful o WebSockets para una comunicación en tiempo real.
- Intercambio de Datos:
    - Los datos intercambiados entre los dispositivos de los jugadores y el servidor de juegos podrían incluir información sobre el estado del juego, como la secuencia de colores actual, las acciones de los jugadores, las puntuaciones, etc. Estos datos podrían ser enviados y recibidos en formato JSON u otro formato estructurado para facilitar su procesamiento tanto en el cliente como en el servidor.

5. Bibliografía

- Documentacion de Android proporcionada por Android Developers
- Cómo desarrollar tu juego en Android Studio
- Beginner Kotlin and Android Studio Tutorial Build a Simon Says Game App
- Best practices for making an Android game
- How to build a game on android studio / (Example)
- How to Build a Tic Tac Toe Game with Both Offline and Online Mode in Android?