Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Aplicación móvil de reloj de ajedrez para personas con discapacidad visual

Chess clock mobile app for visually impaired people

David Hernández Suárez

La Laguna, 6 de septiembre de 2021

D. **David Abreu Rodríguez**, con N.I.F. 78568919-E profesor contratado laboral interino, adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

Dña. **Vanesa Muñoz Cruz**, con N.I.F. 78698687-R profesora Titular de Universidad, adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutora

C E R T I F I C A (N)

Que la presente memoria titulada:

"Aplicación móvil de reloj de ajedrez para personas con discapacidad visual"

ha sido realizada bajo su dirección por **D. David Hernández Suárez**,

con N.I.F. 42240107-D.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 6 de septiembre de 2021

Agradecimientos

Quisiera agradecer a mi tutor, David Abreu Rodríguez, y cotutora, Vanesa Muñoz Cruz, por su entera disposición y por prestarme su ayuda y asesorarme durante la realización de este proyecto.

Agradezco también su labor al profesorado que me ha formado durante estos años, y que han colaborado para hacerme disfrutar esta etapa universitaria.

Por último, agradezco a mi familia y amigos, por haber estado junto a mí en los momentos más críticos de la carrera y haberme apoyado en todo momento.

Licencia

© Esta obra está bajo una licencia de Creative Commons Reconocimiento 4.0 Internacional.

Resumen

El objetivo de este proyecto ha sido el desarrollo de una aplicación para dispositivos Android, que sirva de reloj de ajedrez para personas ciegas o con algún tipo de discapacidad visual. De esta forma, el reloj está completamente adaptado para poder ser utilizado por usuarios con estas características. Se ha utilizado el Entorno de Desarrollo Integrado Android Studio para la implementación de la aplicación, y el código fuente está desarrollado en Java.

A continuación, se detallan las principales tecnologías utilizadas en el desarrollo de la aplicación:

- *Text To Speech*, para crear un asistente de voz.
- *Speech To Text*, para crear un sistema de reconocimiento de voz.
- Protocolo Bluetooth, para partidas en diferentes dispositivos.
- Protocolo Internet con *Scaledrone*, para partidas online en diferentes dispositivos.

Palabras clave: android, reloj, discapacidad visual, ajedrez, java.

Abstract

The objective of this project has been the development of an application for Android devices, which works as a chess clock for people who are blind or visually impaired. In this way, the clock is completely adapted to be used by users of these characteristics. The Android Studio Integrated Development Environment has been used for the implementation of the application, and the source code is implemented in Java.

The main technologies used in the development of the application are detailed below:

- *Text To Speech*, to create a voice assistant.
- *Speech To Text*, to create a speech recognition system.
- Bluetooth protocol.
- Internet Protocol with *Scaledrone*.

Keywords: android, clock, visual impairment, chess, java.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

1.1 Descripción general del proyecto

Este proyecto contempla la creación de una aplicación para dispositivos móviles Android, que cumpla las funciones de un reloj especializado para partidas de ajedrez, con la particularidad de que éste debe estar adaptado para usuarios con discapacidad visual, ceguera o algún tipo de problema de visión. En este sentido, el usuario debe disponer de diferentes señales sonoras que le guíen, de forma similar al funcionamiento de la tecnología *Talkback* disponible en los dispositivos Android. De forma adicional, se busca que la aplicación incluya comandos de voz o reconocimiento de gestos, de modo que el usuario pueda acceder a las distintas funcionalidades de la herramienta sin necesidad de accionar manualmente ningún botón o acceso directo. Por último, la aplicación debe incluir diferentes protocolos de conexión, de modo que se permita la interacción entre dos dispositivos; de esta forma, cada usuario podría jugar la partida de ajedrez y utilizar su propio dispositivo móvil. Así, se podrá realizar la conexión tanto por Bluetooth para partidas presenciales, como por internet para partidas online.

1.2 Antecedentes

El ajedrez es el deporte más practicado por personas con discapacidad visual. Tanto es así, que organizaciones como la **ONCE** han invertido en una completa adaptación del juego para personas ciegas. Esto es así principalmente por la facilidad de adaptación del ajedrez a personas ciegas, necesitando realizar únicamente unos pequeños cambios en el material utilizado para jugar. Así, se organizan torneos en los que estos usuarios utilizan los medios adaptados para realizar sus partidas de ajedrez de forma normal. Sin embargo, algunos de los componentes adaptados del juego no son accesibles para todos, debido a su elevado costo. Si bien para los torneos oficiales es la ONCE quien presta los dispositivos y el material adaptado, si un usuario desea jugar en otro ámbito no oficial, se verá obligado a comprar estos elementos. Éste es el principal inconveniente que se presenta, y el que ha motivado la creación de este proyecto.

A continuación se describen algunas de las adaptaciones que se han realizado en el juego del ajedrez:

1.2.1 Tablero y fichas

El tablero y las fichas de ajedrez son el principal obstáculo que se puede encontrar a la hora de adaptar el juego a las personas con discapacidad visual. Sin embargo, éste es un problema que ya se encuentra resuelto. En primer lugar, las fichas son fácilmente reconocibles al tacto; simplemente basta con añadir una pequeña pieza adicional en la cabeza de todas las fichas negras, de modo que el usuario pueda saber de qué color es la pieza que está tocando. Por otro lado, para evitar que las piezas se caigan o se muevan cada vez que uno de los jugadores palpa el tablero para situarse, el tablero incluye en el centro de cada casilla un pequeño orificio donde las piezas serán insertadas.

Por otro lado, para poder diferenciar las casillas negras de las blancas, las primeras son unos dos milímetros más altas. Con estas pequeñas pero efectivas modificaciones, el juego ya puede desempeñarse con normalidad. Cuando le toque mover a un jugador, éste irá tocando el tablero para saber dónde está cada una de sus piezas y las de su oponente; asimismo, utilizará la misma técnica para mover la ficha que desee a la posición oportuna. Además, también es posible realizar la partida de ajedrez con dos tableros, uno para cada jugador, en el que se replica cada jugada; de esta forma, el jugador puede tocar el tablero todas las veces que necesite, sin interferir con las manos del oponente. Sin duda, jugar de esta forma requiere un mayor tiempo para cada jugada, así como una buena memoria para recordar las posiciones exploradas.

1.2.2 Movimientos

A la hora de realizar un movimiento, el jugador no solo moverá su ficha, sino que además cantará la jugada. De esta forma, el adversario sabrá desde ese mismo momento cuál es el movimiento que ha ejecutado su rival. Para anunciar la jugada, el usuario debe indicar en primer lugar el nombre de la pieza que va a mover, seguido de la posición a la que mueve, en notación algebraica. Una vez escuche el movimiento de su oponente, el jugador deberá replicar esa misma jugada en su tablero, en caso de estar realizando la partida con dos tableros.

De forma opcional, los jugadores podrán anotar cada jugada que se va realizando; para ello, pueden utilizar notación en braille o simplemente un sistema de grabación.

1.2.3 Reloj

En partidas de ajedrez comunes, se puede jugar utilizando un reloj especializado, de modo que la partida tenga un límite de tiempo. Esta es una práctica habitual en torneos y otros eventos oficiales de ajedrez. Este reloj está conectado a un pulsador, un mecanismo que se debe pulsar cada vez que un jugador termina su turno. Al inicio de la partida, se configura el reloj para otorgar el mismo tiempo a cada usuario. Cuando un usuario mueve ficha, posteriormente debe accionar el pulsador; es entonces cuando su tiempo se detiene, y empieza a correr el del adversario, que hasta ese momento permanecía pausado. La partida finaliza cuando alguno de los jugadores haga jaque mate, o bien cuando a alguno de ellos se le agote su tiempo.

La creación de este reloj es relativamente simple. Se trata de realizar un mecanismo con dos relojes y un pulsador a modo de interruptor en el centro; mientras un reloj esté en marcha, el otro permanecerá detenido. Este tipo de dispositivos permite, además, que cada usuario pueda ver en cada momento de la partida no solo el tiempo que le queda disponible a él, sino también el que le resta a su adversario. Además, el dispositivo permite parar ambos relojes en caso de que se pause la partida, y modificar alguno de los tiempos con la partida ya iniciada; esto es útil para poder imponer una penalización de tiempo causada por alguna infracción realizada en el juego.

Otro de los aspectos fundamentales del reloj de ajedrez es el incremento. Existen muchas modalidades de ajedrez en las que el tiempo de juego varía considerablemente, y en las que entra en juego un incremento de varios segundos. El incremento en las partidas de ajedrez funciona de la siguiente manera: cada vez que un jugador mueve ficha y acciona el pulsador, su tiempo no solo se detiene, sino que se añade al mismo un incremento de varios segundos. De esta forma, cada vez que un jugador realice una jugada, gastará un tiempo variable pero se le sumará un número de segundos fijo, establecido antes de la partida. Esto permite que cuando la partida está finalizando y, por tanto, ambos contrincantes se están quedando sin tiempo, puedan recuperar algunos segundos si mueven sus fichas con rapidez.

Teniendo todo esto en cuenta, nos planteamos cómo habría que modificar este reloj para que pudiera ser utilizado por personas con discapacidad visual. El funcionamiento del reloj es exactamente el mismo, solo que se incluyen una serie de botones. Así, cada jugador dispone de dos botones, uno que indica de forma sonora el tiempo que le queda disponible, y otro que indica el

tiempo que le resta al contrincante. Cada usuario cuenta con unos auriculares a través de los cuales recibe la información del tiempo que haya solicitado; como es lógico, cada jugador podrá consultar ambos tiempos en cualquier momento de la partida.

Figura 1. Reloj de ajedrez adaptado

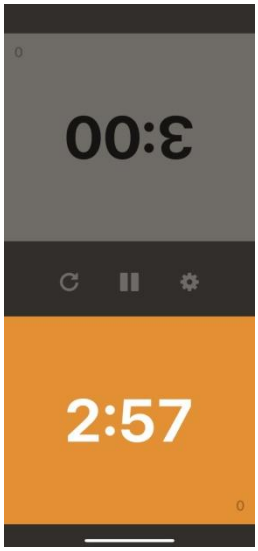
En la imagen de la izquierda se puede apreciar un reloj de ajedrez adaptado para personas con discapacidad visual. Incluye dos pequeñas pantallas en las que se muestra el tiempo de cada jugador, así como cuatro botones (dos por jugador) que indican el tiempo de cada uno. Por último, vemos que el pulsador funciona como un interruptor, dando paso al siguiente jugador una vez el primero haya finalizado su turno.

1.3 Creación de un reloj propio

Aunque ya existe un reloj de ajedrez adaptado para personas con discapacidad visual, son pocos los usuarios que realmente se pueden permitir su adquisición. Todos y cada uno de los elementos de ajedrez adaptados tienen un precio bastante elevado, pero especialmente el reloj de ajedrez, por el que se piden varios cientos de euros. Principalmente debido a este factor, se ha propuesto la creación de una aplicación para dispositivos móviles Android que realice las mismas funciones que un reloj de ajedrez adaptado. De esta forma, los usuarios podrían optar por descargar una app gratuita y evitar así comprar el reloj. La versión final de la aplicación no solo incluirá toda la funcionalidad del reloj convencional, sino que además añadirá nuevas características que mejoren la experiencia de usuario, y faciliten aún más la inclusión de la población ciega al ajedrez.

1.4 Estado del arte

Como hemos visto anteriormente, realmente ya existe un reloj de ajedrez que cumple con los requisitos funcionales que se le exigen. No obstante, nuestro objetivo ahora recae en la transformación de este dispositivo en una aplicación móvil, lo que otorga no solo una mayor comodidad, sino también una mayor accesibilidad. Previo al desarrollo, el primer paso que se ha realizado es una búsqueda de las aplicaciones gratuitas disponibles actualmente para dispositivos móviles Android. Después de terminar esta pequeña fase de exploración, no fue posible encontrar ninguna aplicación móvil con tales características. Bien es cierto que sí existen aplicaciones de relojes de ajedrez, pero ninguno que esté adaptado a personas con problemas de visión. No obstante, se ha estudiado alguna de estas aplicaciones para ver cómo distribuir los distintos elementos de la aplicación y para decidir cuál va a ser el diseño de la misma. Deberá ser un diseño que ofrezca una interfaz cómoda y práctica, cuyo uso sea factible para una persona con discapacidad visual.



En la imagen se puede ver la pantalla principal de la aplicación *ChessClock*. En esta, tenemos dos relojes, como en el caso del reloj convencional, pero no vemos ningún pulsador. En este ejemplo hay dos pulsadores, uno para cada jugador, que coinciden con los relojes. De esta forma, cuando un jugador termina su turno, pulsa sobre su propio reloj, éste se para y empieza a correr el del adversario. Realmente esta implementación se podría variar para que hubiera un único pulsador, y ser así más fiel al reloj original; sin embargo, realmente resulta más cómodo que cada usuario tenga su propio pulsador. Por otro lado, contamos con un pequeño botón dentro de cada reloj para establecer el tiempo, así como tres botones centrales para pausar, resetear y abrir los ajustes de la app. Utilizaremos el diseño de esta aplicación como modelo para nuestro proyecto. Así, crearemos una pantalla principal bastante similar, añadiendo ciertas características que destacaremos más adelante.

Figura 2. Captura de la aplicación ChessClock

https://apps.apple.com/us/app/chess-clock-by-chess-com/id858039162

Capítulo 2
Tecnologías

En este capítulo se expondrán detalladamente las tecnologías que han sido utilizadas a la hora de desarrollar la aplicación. Entre otras, comentaremos el framework de desarrollo utilizado, así como el lenguaje de programación y las librerías empleadas. Se hará especial hincapié en las tecnologías que permiten la adaptación del reloj a personas con problemas de visión.

2.1 Android Studio

Android Studio es un Entorno de Desarrollo Integrado, ampliamente utilizado para la creación de aplicaciones para dispositivos Android. Este framework de desarrollo permite no solo codificar la aplicación, sino además modificar las vistas de la app. Así, podemos ver en tiempo real los cambios realizados programáticamente, o hacerlos directamente de forma gráfica. Ésta es una ventaja que, aunque parezca básica, no se encuentra en todos los IDEs; por ejemplo, en *Eclipse* necesitamos un programa externo para poder obtener esta funcionalidad.

Por otra parte, para ejecutar la aplicación que se está desarrollando, Android Studio nos proporciona varias opciones. Por un lado, podemos directamente conectar un dispositivo android; en este caso, cada vez que ejecutemos la app, ésta se instalará en el dispositivo y se ejecutará en el mismo, pudiendo utilizar la app tal y como será utilizada cuando esté finalizada. Por otro lado, podemos crear un dispositivo virtual que simula perfectamente el comportamiento del físico. En este caso, la aplicación se ejecutaría en este dispositivo virtual, que se abre en una nueva ventana de Android Studio. Hemos optado por la primera opción, puesto que la segunda ralentiza mucho más el funcionamiento del programa. Además, como en esta aplicación vamos a necesitar solicitar diferentes permisos, al tener que acceder al micrófono del dispositivo y al Bluetooth, entre otras herramientas, la ejecución en un dispositivo virtual acabaría dando más problemas que ventajas.

En cuanto a algunas herramientas externas, Android Studio está directamente vinculado a GitHub, de modo que la utilización de este sistema para el control de versiones resulta muy cómoda. Podemos vincular nuestro proyecto a un repositorio en GitHub, y la propia interfaz del programa nos permite fácilmente hacer un *commit* y un *push*, crear ramas, etc. Además de esto, Android Studio también incluye una interfaz para utilizar ***Firebase*** en el proyecto, aunque en este caso no ha sido necesario, puesto que se ha utilizado ***Scaledrone***.

Además, se ha elegido este framework de desarrollo debido a que es el IDE oficial para la plataforma Android. Esto implica que todas las aplicaciones que se implementen en este programa solo estarán disponibles para dispositivos Android, y no para dispositivos IOS. Existen otros frameworks de desarrollo que sí permiten la creación de aplicaciones para dispositivos IOS, como **React Native**. En una primera instancia se estudiaron ambas posibilidades para comprobar qué herramienta se ajustaría más al propósito del proyecto. Aunque React Native presenta una interfaz bastante similar a Android Studio, y además soporta la creación de aplicaciones IOS, finalmente se optó por utilizar Android Studio como plataforma para crear la app. Esta decisión se basa principalmente en que React Native exige un conocimiento más avanzado de desarrollo web y JavaScript, además de la utilización de Node. Por el contrario, la utilización de Java en Android Studio parecía, a priori, más simple. Además, la limitación de tiempo para realizar el proyecto ayudó a tomar la decisión.

2.2 GitHub

GitHub es un servicio de alojamiento y de desarrollo colaborativo basado en el control de versiones de Git. Permite alojar el código fuente, documentación, tests, y todo tipo de ficheros en repositorios. Hemos utilizado esta plataforma en este proyecto para ir guardando las diferentes versiones realizadas. De esta forma, en cada *commit* podemos acceder a los cambios realizados entre una versión y otra, y retroceder si es necesario. Como se dijo anteriormente, la conexión entre GitHub y Android Studio viene predefinida en este último. Simplemente basta con crear un repositorio GitHub en la web y asociar el proyecto actual al mismo. Para ello, primero es necesario vincular Android Studio con nuestra cuenta de GitHub, operación que se realizó utilizando un *Personal Access Token*. Este sistema de control de versiones es especialmente útil, además, para la colaboración en un proyecto. Sin embargo, al tratarse de un proyecto individual, esta ventaja no fue aprovechada en esta ocasión.

Por otro lado, Android Studio cuenta con una consola para Git. En ella podemos ver las diferentes ramas creadas en el proyecto, así como todos y cada uno de los *commits* realizados. Asimismo, pinchando sobre cada *commit* podemos observar qué ficheros fueron modificados en esa nueva versión, qué usuario hizo el *commit* y la fecha del mismo. A continuación, se adjunta una imagen con la información de la consola de Git:

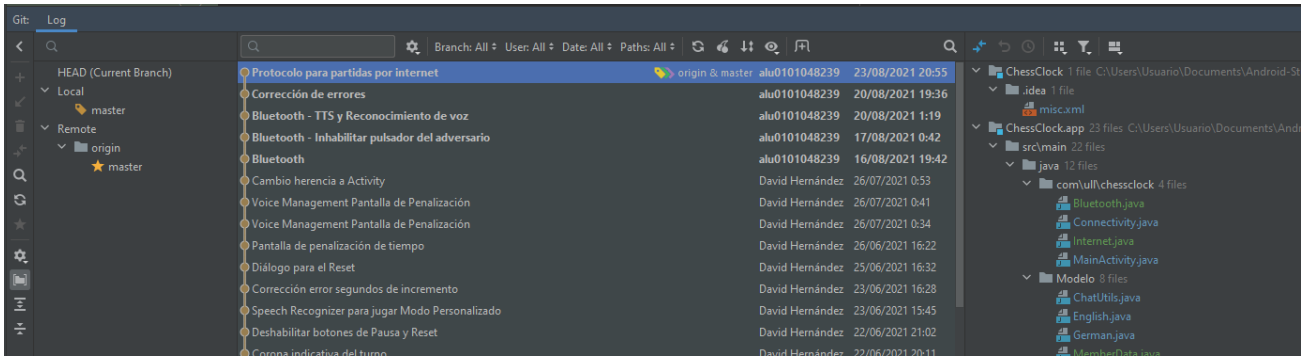


Figura 3. Consola de Git en Android Studio

2.3 Java

Java es un lenguaje de programación multiplataforma y orientado a objetos, y es el lenguaje utilizado en este proyecto. Android Studio soporta la codificación en dos lenguajes: Java y Kotlin. Se ha elegido Java como lenguaje principalmente por ser relativamente antiguo, lo que implica que haya más documentación del mismo. Mientras que Kotlin es un lenguaje muy nuevo, será mucho más fácil encontrar referencias, información, ejemplos de código y foros sobre Java. Además, Java era un lenguaje en el que el autor de la presente memoria tenía experiencia previa, por lo que la curva de aprendizaje sería mucho menor.

En especial, se ha hecho uso de clases, herencia y polimorfismo. Un ejemplo claro de ello es el tema de los idiomas. La app está disponible en varios idiomas, de modo que utilizamos el polimorfismo para definir cuál es el idioma de la app. Posteriormente, sea cual sea el lenguaje seleccionado, se accederá a los mismos métodos.

Por otro lado, se ha utilizado programación orientada a eventos para controlar las diferentes acciones que van teniendo lugar a la hora de interactuar con la aplicación. Este es un apartado muy importante, puesto que ha sido necesario ajustar la aplicación para poder invocar el mismo evento de formas diferentes. Esto permite que un usuario pueda acceder a un apartado de la app pulsando sobre un botón, diciendo verbalmente el nombre del apartado, realizando un gesto con los dedos, etc.

2.3.1 Modelo Vista Controlador (MVC)

Por otro lado, para mantener una organización adecuada, se ha hecho uso del Modelo Vista Controlador (MVC). Este es un patrón de arquitectura del software que separa los datos de la aplicación de su interfaz gráfica, diferenciando tres partes:

- **Vista**: es la interfaz de usuario, es decir, la parte visible por el cliente de la aplicación. Como dijimos anteriormente, la vista puede ser modificada tanto programática como gráficamente, y se encuentra diseñada en lenguaje **xml**.
- **Modelo**: contiene toda la parte lógica del programa. Es en esta parte donde creamos las clases, guardamos los diferentes elementos del reloj, las opciones, etc.
- **Controlador**: es el nexo entre las dos partes anteriores. Contiene el código necesario para gestionar los eventos lanzados desde la interfaz de usuario.

Llevando estos conceptos a la práctica, en la aplicación cada una de las diferentes pantallas la llamaremos **Actividad**, y será un controlador. Cada actividad accederá al modelo para realizar las operaciones oportunas y reflejarlas en la vista. Por tanto, la app tendrá tantos controladores como pantallas diseñadas. Hay que tener en cuenta que, aunque las actividades son también clases, estas son clases diferentes, puesto que incluyen métodos predefinidos que deben implementarse obligatoriamente para el correcto funcionamiento de la aplicación.

2.4 Text To Speech

Uno de los elementos fundamentales en este proyecto es la creación de un sistema que permite servir de guía sonora para los usuarios. De esta manera, cada vez que el usuario realice una acción en la app, tendrá una respuesta sonora que le indique qué ha hecho, a qué pantalla está accediendo, qué ajuste ha activado… Para conseguir este objetivo, hemos utilizado diferentes librerías de android. En una primera instancia, se planteó la posibilidad de no utilizar la base de datos que nos proporciona esta librería de android, sino añadir explícitamente los diferentes audios que fueran necesarios. No obstante, realizar esta tarea suponía grabar una cantidad considerable de audios; además, sería más complicada una posible ampliación de la app con nuevos idiomas. Por tanto, finalmente se utilizaron las librerías que se comentan en los siguientes apartados.

2.4.1 android.speech.tts.TextToSpeech

Esta es la librería más importante que se ha utilizado en el proyecto. Nos permite reproducir sonoramente el texto que queramos. Así, dependiendo del botón o el elemento que accione el usuario, indicaremos al Text To Speech que reproduzca unas palabras determinadas. Esta librería que nos ofrece android incluye una larga lista de métodos. A continuación, vamos a comentar algunos de los métodos que hemos utilizado:

- ***speak***: este es el método más importante de la librería, puesto que es la instrucción que reproduce el sonido. Recibe una serie de parámetros, entre ellos un String con el texto que se quiere leer.
- ***setSpeechRate***: este método permite cambiar la velocidad de nuestro speaker. Esto es una opción muy útil, puesto que la idea es que el usuario pueda modificar este valor, para que el asistente de voz hable más rápido o más lento. Recibe un flotante que indica la velocidad, siendo el valor 1 el ritmo normal por defecto.

- ***setPitch***: este método permite cambiar el tono de voz, haciéndolo más grave o más agudo. De nuevo, esta es una opción que el usuario podrá cambiar para adaptarlo a su gusto.
- ***setLanguage***: éste es otro de los métodos más importantes, que nos permite cambiar el idioma del asistente. El usuario podrá cambiar el idioma a español, inglés o alemán. Sin embargo, la librería incluye una base de datos con una gran cantidad de idiomas: italiano, francés, portugués, ruso, etcétera. Cambiar el idioma del asistente es algo trivial, puesto que simplemente debemos llamar a este método indicando el idioma que queremos. Se ha implementado cada idioma en una clase distinta, que hereda de un padre común; así, si en un futuro se quiere ampliar la app para que soporte nuevos idiomas, solo sería necesario crear una nueva clase.

  El cambio del idioma es importante principalmente por la pronunciación del texto que se quiere leer. Si introducimos el texto en inglés, pero el idioma establecido es el español, el texto será leído de igual forma; no obstante, lo leerá en español, de modo que no se entenderá el mensaje. Es algo similar a lo que ocurre con el traductor de Google, si le indicamos que pronuncie un texto en un idioma distinto al establecido.
- ***setVoice***: este método permite cambiar la voz del asistente. Para ello, es necesario pasarle como parámetro un objeto de la clase android.speech.tts.Voice, que comentaremos en el siguiente apartado.

Por otro lado, a la hora de crear el objeto Text To Speech, es muy importante indicar de qué base de datos se van a obtener las voces. Si en el constructor de la clase indicamos *null*, luego tendremos problemas para cambiar las voces del asistente. Por tanto, pasaremos la opción *com.google.android.tts* para acceder así a la base de datos de voces de Google.

2.4.2 android.speech.tts.Voice

Esta es una librería secundaria para la creación del asistente de voz. En este caso, esta librería la utilizamos únicamente para seleccionar la voz de nuestro asistente. Así, para cada idioma hemos habilitado una serie de voces, masculinas y femeninas, de modo que el usuario pueda elegir la voz que prefiera. Para ello, el método que hemos utilizado es *setVoice*, comentado anteriormente. Sin embargo, este método recibe un objeto, de la clase *Voice*, que crearemos previamente. En este objeto indicamos el idioma de la voz, así como su identificador único. Un paso intermedio consiste en crear un objeto *Locale* en el que se indica no solo el idioma que se busca, sino además el país del que procede; de esta manera, la librería distingue el inglés británico del americano, por ejemplo, o el castellano del latino.

2.5 Speech Recognizer

El otro pilar fundamental de la aplicación es el reconocimiento de voz. Recordemos que la idea es que el usuario pueda acceder a la aplicación, cambiar sus ajustes o realizar cualquier acción sin necesidad de tocar específicamente un botón. Para conseguir este propósito, se barajaron dos opciones. Por un lado, la detección de gestos, de modo que tocando cualquier punto de la pantalla siguiendo un patrón determinado se accediera a algún punto de la app. Por otro lado, el reconocimiento de voz, de modo que el usuario simplemente hablara, indicando qué quería hacer. Finalmente, se optó por esta última alternativa; el usuario deberá indicar un comando de voz para poder acceder a ciertas funcionalidades de la aplicación. No obstante, aunque esta tecnología resulta más sencilla para el usuario con discapacidad visual, igualmente están habilitados los accesos comunes, de modo que también se puede acceder a las mismas prestaciones pulsando un botón, por ejemplo.

En los siguientes apartados se comentarán las librerías que se han utilizado para el reconocimiento de voz.

2.5.1 android.speech.SpeechRecognizer

Esta es la librería más importante, que nos permite crear el objeto necesario para iniciar el reconocedor de voz. Entre sus métodos más importantes podemos destacar los siguientes:

- ***createSpeechRecognizer**:* crea el objeto de la clase *SpeechRecognizer*.
- ***setRecognitionListener**:* crea el escuchador del *SpeechRecognizer*, creando sus métodos predefinidos.
- ***startListening**:* activa el escuchador, de modo que el reconocedor de voz empieza a escuchar lo que dice el usuario.

2.5.2 android.speech.RecognitionListener

Esta librería nos permite implementar en la clase un *listener*, que permanecerá escuchando en espera de un comando de voz. Un *listener* es un sistema que se encarga de controlar eventos, esperar a que estos se produzcan y realizar una serie de acciones en consecuencia. Guardaremos en un atributo de la clase el contenido de ese comando de voz que ha escuchado, y posteriormente lo procesaremos para ver si el mensaje obtenido se corresponde con alguno de los mensajes de voz disponibles en la aplicación. El *listener* incluye varios métodos por defecto, pero el único que utilizamos es *onResults*, que se invoca automáticamente cuando se obtiene el mensaje de voz.

Hay que aclarar que el escuchador no estará activo constantemente; para evitar que esté escuchando cuando sea innecesario, se ha aplicado un mecanismo para activarlo. Así, el usuario deberá pulsar un botón físico del dispositivo (en este caso, el botón de subir volumen) y acto seguido indicar el comando de voz.

2.5.3 android.speech.RecognizerIntent

Esta es simplemente una librería que funciona como intermediaria, siendo necesaria para la creación del reconocedor, en concreto para establecer el idioma en que va a funcionar.

2.6 Bluetooth

Además del asistente y del reconocimiento de voz, otra de las tecnologías que incluye el proyecto es un protocolo de conexión por bluetooth. De esta forma, conseguimos utilizar la aplicación con dos dispositivos diferentes, de modo que cada jugador puede utilizar su propio smartphone para realizar la partida. Para establecer la conexión por bluetooth, será necesario implementar dos códigos; uno para el dispositivo que actúe como servidor, y otro para el que actúe como cliente. De esta forma, tendremos varios hilos de ejecución. Se han utilizado principalmente las librerías que se mostrarán en los siguientes apartados.

2.6.1 android.bluetooth.BluetoothAdapter

Esta es la librería principal para la incorporación de la tecnología Bluetooth en la aplicación. Crearemos un objeto de esta clase, y con él accederemos a los distintos métodos que necesitemos. A

continuación, se exponen los métodos de esta clase que hemos utilizado:

- ***getDefaultAdapter***: método necesario para la inicialización del objeto *BluetoothAdapter*.
- ***getBondedDevices***: método que accede a la lista de aparatos que han sido emparejados con el dispositivo previamente.
- ***startDiscovery***: comienza a realizar la búsqueda de los dispositivos cercanos disponibles, que no estén emparejados.
- ***cancelDiscovery***: cancela la búsqueda anterior, para ahorrar recursos.

2.6.2 android.bluetooth.BluetoothDevice

Esta librería es necesaria para guardar el conjunto de dispositivos bluetooth disponibles. Cada dispositivo obtenido será un objeto de esta clase, guardando algunas de sus propiedades más importantes. Los métodos principales de esta librería son los siguientes:

- ***getBondState***: indica si el dispositivo está vinculado o no.
- ***getName***: indica el nombre del dispositivo en cuestión.
- ***getAddress***: indica la dirección del dispositivo, un String con la siguiente forma: 00:11:22:AA:BB:CC.

2.6.3 android.bluetooth.BluetoothServerSocket

Esta librería es la necesaria para implementar la parte del código relacionada con la gestión del servidor. Simplemente utilizaremos los dos métodos siguientes:

- ***accept:*** espera a que se establezca una conexión, y devuelve un objeto *BluetoothSocket* si se ha conectado correctamente.
- ***close:*** desactiva el *ServerSocket.*

2.6.4 android.bluetooth.BluetoothSocket

Esta clase es necesaria para la implementación del código referente a la programación multihilo. Con ella, creamos los sockets necesarios para que la comunicación vía bluetooth funcione correctamente.

- ***connect:*** se inicia la conexión a un dispositivo remoto.
- ***close:*** desactiva el *socket*.
- ***getRemoteDevice:*** obtiene el dispositivo remoto al que el *socket* está conectado.
- ***getInputStream:*** obtiene el flujo de entrada asociado con el *socket*.
- ***getOutputStream:*** obtiene el flujo de salida asociado con el *socket*.

2.7 Internet

Finalmente, la última tecnología de la que dispone la aplicación es de un protocolo de conexión por internet. Esto permite poder utilizar la app en dos dispositivos distintos, pudiendo realizar partidas online. Para lograr este objetivo, es necesario utilizar alguna herramienta externa como

Firebase; sin embargo, aunque ésta está directamente vinculada a Android Studio, hemos optado por utilizar *Scaledrone*, puesto que para nuestro caso es más sencillo y directo.

2.8 Scaledrone

Scaledrone es una plataforma y un servicio de mensajería en tiempo real. Permite crear salas de chat a las que se pueden unir múltiples usuarios, e interactuar entre ellos. Se ha utilizado esta herramienta para poder establecer la conexión entre los dos jugadores. Cada uno de ellos entrará en la misma sala; una vez estén dentro los dos, estarán conectados. De esta forma, podrán intercambiar mensajes entre ellos. No obstante, no está habilitada la opción de compartir mensajes; realmente, lo que hacemos es enviar internamente señales, indicando qué acción ha realizado el jugador 1 para que al jugador 2 se le refleje ese movimiento en su dispositivo.

Para empezar a usar Scaledrone en nuestro proyecto, una vez creemos la cuenta debemos crear un canal. A continuación, se asignará a dicho canal un identificador y una clave secreta; estos dos datos serán necesarios posteriormente, de modo que debemos copiarlos e incluirlos en el código de la aplicación. Es importante aclarar que es la app la que se encarga de crear la sala a la que se conectarán los dispositivos; el usuario no tendrá que realizar esta labor.

Además de instalar la dependencia *com.scaledrone:scaledrone-java:0.6.0*, debemos usar las librerías que se comentarán en los siguientes apartados.

2.8.1 com.scaledrone.lib.Scaledrone

Es la librería principal de Scaledrone, y la que nos va a permitir crear el objeto Scaledrone. Para crearlo, es necesario indicar el identificador del canal, así como la información del usuario (en este caso, el nombre). La información del usuario viene dada por el objeto de la clase *MemberData*, que debemos implementar previamente. Los métodos principales de esta librería son:

- ***connect:*** crea el *listener* del objeto *scaledrone*.
- ***subscribe:*** enlaza el *listener* creado con la sala.
- ***publish:*** publica el mensaje en la sala, de modo que éste es enviado al otro dispositivo.
- ***getClientID:*** obtiene el identificador del dispositivo actual.

2.8.2 com.scaledrone.lib.Listener

Esta librería es necesaria para implementar el escuchador necesario para inicializar la conexión de Scaledrone. Cuando llamamos al método *connect* del apartado anterior, este recibe un escuchador que es invocado cuando se abre la conexión a Scaledrone. Los métodos por defecto que incluye el escuchador son los siguientes:

- ***onOpen:*** se invoca cuando el dispositivo está conectado a la sala.
- ***onOpenFailure:*** se llama si ha habido un error al conectarse a la sala.
- ***onFailure:*** se invoca cuando ha ocurrido algún error.
- ***onClosed:*** invocado cuando se cierra la conexión.

2.8.3 com.scaledrone.lib.Room

Esta librería es utilizada para representar la sala donde se conectan los usuarios. Existen varios métodos, que comentaremos en el siguiente apartado, que reciben un objeto de la clase *Room*,

indicando la información necesaria sobre la sala que hemos creado previamente, y a la que se unirán los dos jugadores.

2.8.4 com.scaledrone.lib.RoomListener

La clase en la que implementamos todo este protocolo de conexión (en este caso, *MainActivity*) debe implementar un *RoomListener*. De esta forma, el sistema se quedará escuchando hasta que reciba un mensaje. Los métodos por defecto que incluye la librería, y que hemos utilizado para el manejo de los mensajes, son los siguientes:

- ***onOpen**:* invocado si el dispositivo se ha conectado correctamente a la sala.
- ***onOpenFailure**:* se invoca si ha habido algún error a la hora de conectarse a la sala.
- ***onMessage**:* es llamado cuando se envía y se recibe un mensaje. Simplemente comprobaremos quién fue el emisor y actuaremos en consecuencia.

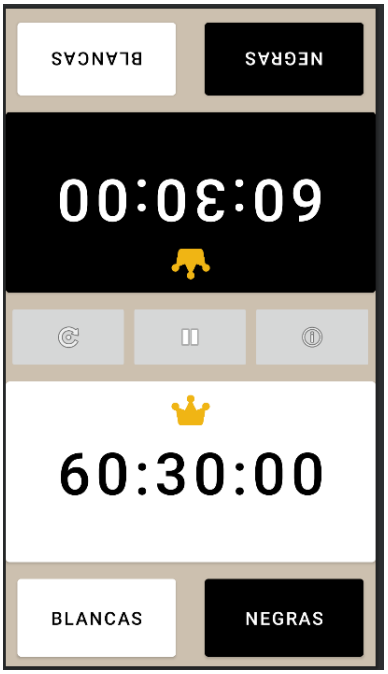
Este último es el método más importante, puesto que en él se incluye el código necesario para replicar los movimientos del emisor en el receptor.

Capítulo 3
Desarrollo

3.1 Pantalla principal

En esta aplicación, al igual que en otras similares, se ha establecido la pantalla principal como aquella en la que se desarrolla el juego. Se ha construido de esta forma principalmente para ofrecer una mayor facilidad de uso y accesibilidad. De esta forma, si un usuario quiere empezar a jugar, no necesita seleccionar ninguna opción o apartado; simplemente abrirá la app y ya podrá comenzar la partida directamente. La pantalla principal está diseñada de tal manera que el dispositivo donde se encuentra la app se sitúe en el centro del espacio de juego; así, ambos jugadores compartirán un único dispositivo para la partida. Además, se ha intentado seguir un diseño en el que se maximice el contraste, para los usuarios que sí cuenten con algo de visión.

3.1.1 Diseño



Como vemos en la imagen, contamos con dos relojes, uno para cada jugador. El reloj de la parte superior se corresponde con el tiempo del jugador con las fichas negras, mientras que el otro sería el reloj de las blancas. En el centro de cada uno de los relojes se muestra el tiempo de juego restante; en este caso, se ha puesto como tiempo por defecto 60 minutos, con un incremento de 30 segundos. Asimismo, podemos observar que encima de cada reloj encontramos una pequeña corona. Una vez inicie la partida, esta corona aparecerá únicamente en aquel reloj cuyo tiempo esté en marcha; de esta forma, indicamos de qué jugador es el turno en cada momento.

Por otro lado, contamos en la parte superior con un botón de *BLANCAS* y otro de *NEGRAS*. Estos botones se repiten en la parte inferior, porque están destinados al uso del otro jugador. De esta forma, cuando comience la partida, el jugador que elija negras utilizará los dos botones de la parte superior, mientras que el jugador de blancas usará los botones inferiores. La funcionalidad de estos botones es la misma, pero entraremos en más detalle en el siguiente apartado.

Figura 4. Pantalla principal

Por otro lado, teniendo en cuenta que ambos jugadores se sentarán uno enfrente de otro, las letras de los botones superiores, así como el tiempo de las negras, están invertidos. De esta forma, ambos

jugadores podrán leer correctamente estos datos, aunque para ellos el dispositivo se encuentre situado al revés. Si bien en principio la app está diseñada para personas con discapacidad visual, sí que puede haber algunos usuarios que cuenten con algo de visión; es por ello que se ha realizado esta pequeña inversión de las letras. Además, tanto esta pantalla como las siguientes han sido desarrolladas creando botones relativamente grandes, y letras con un tamaño considerable. La idea es que el usuario que pueda utilizar mínimamente su visión lo haga de la forma más cómoda posible. Como es lógico, no nos interesa disponer de un diseño en el que haya muchos botones, estén todos muy juntos y casi no se lea su contenido.

Finalmente, en la parte central de la pantalla podemos encontrar tres botones: *PAUSA*, *RESET* y *AJUSTES*. A continuación, se detalla la funcionalidad de cada uno de los botones de esta pantalla principal.

3.1.2 Funcionalidad

Para empezar, podemos comprobar que no existe ningún pulsador. Esto es porque realmente hay dos pulsadores, que coinciden con los relojes. De esta forma, el sistema de juego es muy sencillo:

- Al comienzo de la partida, el pulsador de las blancas está desactivado, pero el de las negras no. Esto es así porque siempre empezarán a mover las blancas.
- Cuando accionamos el pulsador de las negras, ocurren una serie de cosas. En primer lugar, este pulsador que acabamos de accionar se desactiva, y el de las blancas se activa. En segundo lugar, el tiempo de las negras se detiene, y empieza a correr el de las blancas. En tercer lugar, la corona desaparece del pulsador de las negras y aparece en el de las blancas. En esencia, lo que hemos hecho es cambiar el turno, y ahora mueven las blancas. Además, si el modo de juego establecido incluye incrementos, su suma se realiza al cambiar el turno. Así, cuando se presiona el pulsador de las negras, no solo se detiene su tiempo, sino que aparecerá el tiempo después de haberle sumado el incremento de segundos correspondiente.
- Una vez que las blancas hayan completado su movimiento, el jugador pulsará sobre su propio tiempo; éste se detendrá, y se reanudará el de su oponente, repitiendo el mismo proceso de antes.

En cuanto a los botones superiores e inferiores, cada pareja realiza el mismo trabajo. Cuando el botón de *BLANCAS* es pulsado, el asistente de voz indica de forma sonora el tiempo restante que le queda a las blancas. Cuando pulsamos *NEGRAS*, ocurre lo contrario; el asistente indica el tiempo restante de las negras. Así, cada uno de los jugadores tiene acceso en todo momento a conocer su tiempo y el de su rival.

Por otro lado, nos encontramos con los botones centrales. El situado más a la izquierda, *RESET*, nos permite resetear el juego. Si lo presionamos, aparecerá un cuadro de diálogo indicándonos si estamos seguros de que queremos resetear el juego. Asimismo, el mensaje de este cuadro de diálogo será automáticamente leído por el asistente de voz. Si aceptamos, el juego se reseteará. El botón central, *PAUSE*, nos permite pausar el juego. Cuando lo pulsamos, ambos relojes se detienen, y la corona permanece en aquel jugador en cuyo turno se detuvo la partida. Mientras el juego permanece pausado, los dos pulsadores estarán inhabilitados; si queremos reanudar la partida, bastará con volver a presionar el botón de *PAUSE*, y se reanudará el reloj del jugador que continúa su turno. Finalmente, encontramos un último botón, *AJUSTES*, que nos llevará a una nueva pantalla que comentaremos más adelante. Cuando accionamos este botón, la partida se pausa automáticamente. Así, cuando regresemos de vuelta a la pantalla principal, los tiempos quedan guardados, y se podrá continuar la partida. Al principio de la partida, cuando todavía no ha empezado a andar ningún reloj, los botones de pausa y reseteo estarán desactivados, puesto que su uso es inútil.

Debemos recordar que el juego puede terminar cuando se dé alguna de estas dos circunstancias: cuando uno de los jugadores haga jaque mate o cuando a alguno de los jugadores se le agote el

tiempo. Esta segunda condición es la que tenemos en cuenta en la aplicación. Cuando alguno de los dos relojes llega a 0, nos saldrá una ventana emergente indicando *JUEGO FINALIZADO*. Como con el resto de funcionalidades, el asistente de voz indicará en ese momento el mismo mensaje, de forma sonora.

Por último, otra de las funcionalidades que presenta esta primera pantalla es la de establecer una penalización de tiempo a cualquiera de los jugadores. Se puede dar el caso de que, mientras se desarrolla la partida, uno de los jugadores cometa una infracción o una jugada ilegal; en ese caso, el árbitro le impondría una penalización, quitándole tiempo de su reloj o añadiéndole tiempo al adversario. Para poder establecer esta penalización, se accederá de una forma muy concreta, pero sencilla. Si queremos añadir tiempo a las negras, deberemos dejar pulsado su reloj unos segundos, y nos aparecerá la ventana de penalización, que comentaremos más adelante. Del mismo modo, si queremos añadir tiempo a las blancas, haremos lo propio con su reloj. No obstante, esta funcionalidad no es accesible en todo momento. Mientras alguno de los relojes esté en marcha, no podremos acceder a esta pantalla, ni para un jugador ni para otro. Si queremos aplicar una penalización, primero deberemos pausar la partida. Entonces, ambos pulsadores están inhabilitados, pero si los dejamos pulsados podremos acceder a esta ventana.

3.2 Pantalla de Penalización

Esta pantalla es accesible únicamente desde la pantalla principal y, como comentamos anteriormente, es utilizada para aplicar un aumento en el tiempo de cualquiera de los jugadores.

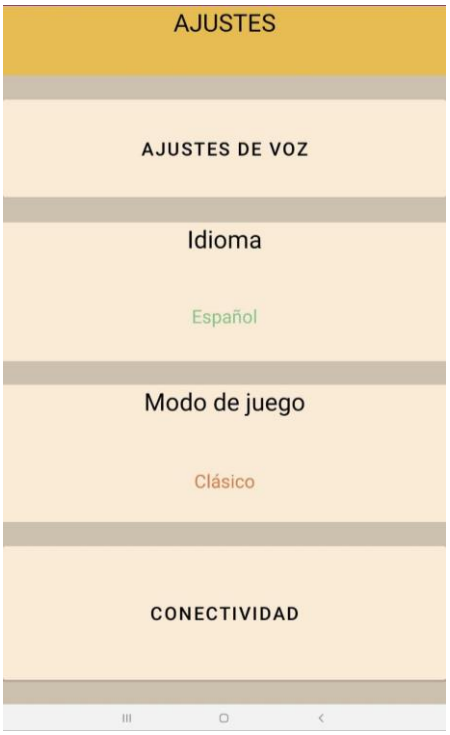
Figura 5. Pantalla de Penalización

Como podemos observar, esta pantalla es mucho más simple. En realidad, será el árbitro quien impondrá las penalizaciones y, puesto que este tiene visión normalizada, no es necesario adaptar la pantalla a las personas con discapacidad visual. Sin embargo, igualmente se ha procurado seguir un diseño adaptado, y de nuevo podremos usar comandos de voz para establecer el tiempo de penalización.

Únicamente encontramos dos pequeñas secciones en las que elegiremos la cantidad de minutos y la cantidad de segundos que se añadirán al tiempo del jugador. Una vez tengamos seleccionado el tiempo, volvemos a la pantalla principal y ya veremos el tiempo actualizado. Si no indicamos ningún tiempo, este por defecto se quedará en 0, de modo que no se le sumará nada al tiempo del jugador.

Tanto si accedemos desde el pulsador de las blancas como desde el de las negras, la pantalla que nos encontramos es la misma. No obstante, dependiendo del pulsador que hayamos usado, el tiempo se añadirá a un jugador o a otro. Hemos optado por realizar esta pantalla así, puesto que cuantos menos botones, secciones y apartados tenga la pantalla, más fácil será de leer y por tanto, mayor facilidad tendrá el usuario para utilizarla. De otra manera, habríamos tenido que duplicar los elementos que se ven en la imagen, distinguiendo los minutos y segundos de las blancas, y los minutos y segundos de las negras.

3.3 Pantalla de Ajustes



Esta pantalla es accesible desde la pantalla principal, pulsando el botón *AJUSTES* situado en la parte central derecha. El cometido principal de ésta es modificar algunos aspectos de la aplicación: el modo de juego, el idioma, los ajustes del asistente de voz y la conectividad. A continuación, iremos explicando detalladamente cada uno de estos apartados, su diseño y su funcionalidad.

Como se puede comprobar, se ha procurado realizar un diseño lo más claro y simple posible, separando cada una de las opciones y creando unos botones relativamente grandes. No obstante, el usuario siempre tendrá la opción de acceder a los distintos apartados mediante el uso de los comandos de voz.

Figura 6. Pantalla de Ajustes

3.3.1 Ajustes de voz

Si pulsamos sobre esta opción, se nos abrirá una nueva ventana de *Ajustes de voz*, destinada a cambiar diferentes aspectos del asistente de voz. Sin embargo, entraremos en más detalle cuando pasemos a comentar esa pantalla.

3.3.2 Idioma

Esta sección nos permite cambiar el idioma de la aplicación. Recordemos que la app soporta tres idiomas: español, inglés y alemán. Si pulsamos sobre esta opción se nos abrirá un menú con los tres idiomas. Una vez cambiemos el idioma, se sucederán una serie de cambios, tanto internamente como en la interfaz. En primer lugar, cambiarán todos los campos de texto, estableciéndose en el idioma seleccionado. En segundo lugar, el asistente de voz cambiará su idioma, de forma que toda la guía sonora que realice se haga en el lenguaje correspondiente. Por último, también se modifica el lenguaje del reconocedor de voz, de modo que entienda los comandos de voz que proporcione el usuario.

Figura 7. Selección de idioma

A la hora de implementar la aplicación en los diferentes idiomas, lo que se ha hecho es crear una clase ***Language***. A continuación, crearemos una clase por cada idioma de la aplicación, que herede de *Language*; por tanto, creamos las clases ***Spanish, English*** y ***German***. Dentro de cada clase, guardaremos tres tipos de datos. En primer lugar, las diferentes voces disponibles en ese idioma. Así, cuando el usuario vaya a cambiar la voz del asistente, accederá a este conjunto de voces. En segundo lugar, un conjunto de frases que dirá el asistente en algún momento. Por último, un

conjunto de etiquetas que utilizaremos para nombrar a los diferentes botones o secciones dentro de cada pantalla. Crearemos exactamente las mismas etiquetas y frases en cada uno de los idiomas, y las guardaremos en un hash. De esta manera, la clave del hash para una etiqueta será la misma para los tres idiomas; no obstante, dependiendo del idioma que esté seleccionado, se accederá al valor de esa clave almacenado en la clase *Spanish*, en la clase *German* o en la clase *English*. De esta forma conseguimos automatizar todo el proceso de cambio de idioma, simplificándolo. Asimismo, si en un futuro se quisiera ampliar la app para que soportara más idiomas, no habría que cambiar muchos aspectos del código, más allá de añadir una nueva clase hija de *Language*.

3.3.3 Modo de juego

En el ajedrez existen una gran cantidad de modos de juego. En cada uno de ellos, el tiempo de juego varía, así como los segundos de incremento. En esta aplicación hemos implementado tres modos predefinidos de juego, que son los más conocidos y extendidos por los jugadores: **Ajedrez Clásico**, **Ajedrez Rápido** y **Ajedrez Blitz**. Cada uno de estos modos de juego presenta un tiempo total y un incremento determinados; de esta forma, el usuario simplemente seleccionará el modo de juego y los dos relojes se establecerán al tiempo determinado. Los tiempos de cada uno de estos tres modos de juego suelen variar mínimamente; no obstante, se ha optado por establecer los tiempos más utilizados para cada uno. Así, para el Ajedrez Clásico contamos con un tiempo de juego de 60 minutos, y un incremento de 30 segundos. Esto significa que cada vez que un jugador presione el pulsador, se le añadirá a un tiempo medio minuto más. Este modo de juego está pensado para partidas largas, en las que los participantes disponen de mucho tiempo para pensar el movimiento que van a realizar. Por otro lado, el Ajedrez Rápido es una modalidad en la que el tiempo de juego varía entre los 10 y los 15 minutos. En este caso, hemos establecido un tiempo de 15 minutos, con un incremento de 10 segundos. Al contrario que en el ajedrez clásico, en este tipo de partidas se busca que el jugador piense mucho más rápido, ya que dispone de un tiempo de juego mucho menor. Por último, en el Ajedrez Blitz se dispone de un tiempo que será siempre inferior a los 10 minutos. Las distintas modalidades utilizan un tiempo determinado u otro, y en este caso el tiempo que hemos definido es de 3 minutos, con un incremento de 2 segundos. Esta modalidad, también llamada Ajedrez Relámpago, es muy utilizada en torneos. Las partidas son muy cortas, y el jugador tiene que mover muy rápido, sin detenerse demasiado a pensar su jugada. Cuando la partida está finalizando, se realizan movimientos muy rápidos para poder recuperar tiempo gracias al incremento.

Figura 8. Selección del modo de juego

El usuario pulsará el modo de juego que desee, y cuando retorne a la pantalla principal, observará que los tiempos de los relojes han cambiado, sustituyéndose por los nuevos tiempos. Como vemos en la imagen, hemos añadido una cuarta opción en los modos de juego, ***Personalizar***. Con esta opción el usuario podrá establecer el tiempo de juego que quiera; elegirá las horas de juego, los minutos, los segundos y el incremento. De esta forma, podrá realizar una partida con un tiempo y un incremento totalmente personalizados. Cuando el usuario presiona este botón *Personalizar*, se abre una nueva ventana emergente, que se muestra a continuación:

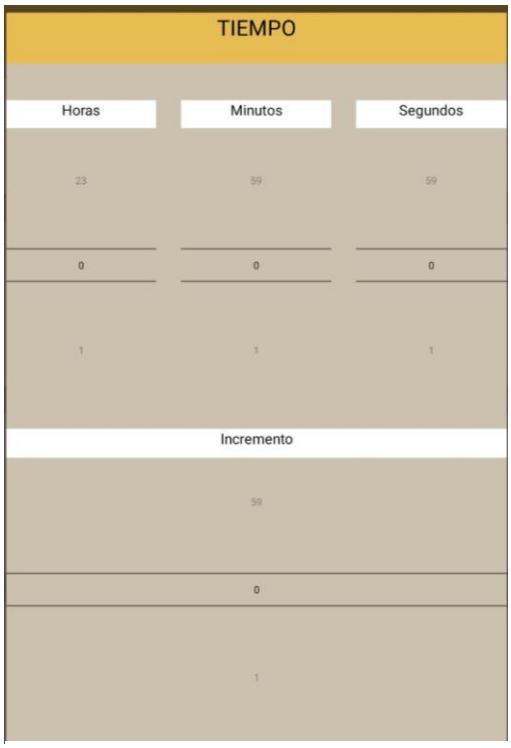


Figura 9. Pantalla de Personalización

En esta pantalla, el usuario cambiará el tiempo que durará la partida, estableciendo el número de horas, minutos y segundos. En cuanto a las horas, hemos puesto un máximo de 23. Es poco probable que algún usuario utilice esta opción, pero se ha preferido poner un límite alto, para que el usuario tenga total libertad a la hora de establecer un tiempo.

Por otro lado, en la parte inferior de la pantalla encontramos el incremento. En este caso, únicamente se ha habilitado la opción de que el tiempo de incremento sean segundos, con un máximo de 59, principalmente porque el máximo tiempo que se suele utilizar para un incremento es de 30 segundos, de modo que ampliarlo más de 1 minuto no resultaría muy útil. Sin embargo, en un futuro esta opción se podría modificar sin mayor problema.

Hay que tener en cuenta que, si no se establece ningún tiempo y se retrocede, el sistema usa por defecto el 0; por tanto, cuando se vaya a empezar la partida, esta finalizará al instante, puesto que el tiempo es de 0 horas, 0 minutos y 0 segundos, con un incremento de 0 segundos.

3.3.4 Conectividad

Esta es una de las últimas funcionalidades que se añadió a la aplicación, y está relacionada con la conexión de la aplicación vía bluetooth o internet. Cuando el usuario presiona este botón, se le abrirá una nueva pantalla, que se comentará en detalle más adelante, en la que el usuario elegirá qué modo de conexión desea activar. El desempeño de esta funcionalidad busca como objetivo principal una mayor comodidad a la hora de jugar, puesto que evita que los dos jugadores utilicen un mismo dispositivo para jugar, lo que puede provocar alguna incomodidad o situaciones de fallos.

3.4 Pantalla de Ajustes de Voz

Figura 10. Pantalla de ajustes de voz

Esta pantalla es accesible desde la pantalla de *Ajustes*, siendo la primera de las opciones que encontramos en ella. Su cometido principal es cambiar algunas de las características del asistente de voz, con el fin de adaptarlo lo máximo posible al gusto del usuario. La idea es que se personalice el asistente, de modo que la experiencia de usuario sea mucho más fructífera.

Como vemos en la imagen, tenemos cuatro partes diferenciadas, en las que ajustaremos algunos de los detalles de la voz del asistente. A continuación, se expone cada uno de estos campos, su diseño y funcionalidad. Cuando hayamos modificado todas las características que queramos, veremos que cuando retrocedamos a la pantalla de *Ajustes*, los cambios ya estarán presentes. Cuando volvamos a la pantalla de inicio, el asistente nos indicará el tiempo de juego de la forma que se ha programado.

3.4.1 Cambio de voz

Como se comentó en apartados anteriores, en cada una de las clases que se realizaron para representar los idiomas de la aplicación, hay un objeto en el que se guardan los identificadores de las distintas voces disponibles para cada idioma. Cuando pulsamos este botón de *CAMBIAR VOZ*, lo que hacemos es acceder a este objeto, que no es más que un *array* que contiene las diferentes voces. Así, cada vez que se presiona esta opción, se accede a la siguiente posición del *array*, cambiando así la voz. Para que el usuario pueda saber instantáneamente si le gusta la voz actual, cada vez que se cambia la voz el asistente dice "Cambiando voz". Con esta pequeña frase, el usuario habrá escuchado la nueva voz y valorará si quiere cambiarla de nuevo o no.

3.4.2 Cambio de velocidad

Éste es quizás uno de los apartados más útiles dentro de los ajustes de voz. Con esta opción el usuario podrá hacer que el asistente hable más rápido o más lento. El objetivo principal del cambio de velocidad es que el jugador pierda el menor tiempo posible mientras juega. Lo ideal es que el asistente le indique a la mayor velocidad posible el tiempo que le queda restante. No obstante, en caso de que el usuario necesite que el asistente hable más despacio para poder entenderle mejor, siempre tiene la opción de bajarle la velocidad de habla. A la hora de cambiar la velocidad, se ha intentado realizar un diseño lo más básico y sencillo posible; si se presiona sobre el botón verde "+", la velocidad aumenta, y si pulsamos el rojo "-", disminuye.

3.4.3 Cambio de tono

Con este apartado damos la posibilidad al usuario de que cambie el tono de voz del asistente, de modo que la agrave o la agudice. De nuevo, contamos con dos botones bien diferenciados; uno para hacer más aguda la voz, y otro para hacerla más grave. Si, por ejemplo, se ha puesto la voz muy grave y se quiere probar cómo sonaría en agudo, hay que deshacer los movimientos que hemos hecho, uno a uno, accionando el mismo número de veces el botón opuesto. Hay que tener en cuenta que cada vez que hayamos agravado, deberemos agudizar para volver al tono anterior. Lo mismo ocurre con el cambio de velocidad.

3.4.4 Activación del asistente de voz

Por último, contamos con un ajuste muy importante relacionado con la habilitación del asistente de voz. Por defecto, cuando inicias la aplicación se encuentra activado, puesto que se espera que el usuario que utilice la app sea alguien que lo necesite. No obstante, si un jugador quiere utilizar la aplicación y no necesita este asistente, lo puede desactivar en esta opción. Hay que aclarar que con esta inhabilitación lo único que se desactiva es el asistente de voz; es decir, ya no se contará con un ayudante sonoro que guíe los pasos en la aplicación. Sin embargo, el reconocedor de voz seguirá funcionando, de modo que podremos seguir haciendo uso de los comandos de voz.

3.5 Pantalla de Conectividad

Si bien el diseño de esta pantalla es muy simple, toda la tecnología que lleva detrás es relativamente compleja. Esta parte de la aplicación permite al usuario establecer una conexión vía bluetooth o vía internet, para quedar enlazado con otro dispositivo. Como vemos, el usuario únicamente dispone de dos botones: ***BLUETOOTH*** e ***INTERNET***.

Hay que tener en cuenta que esta opción se debe utilizar cuando los dos jugadores disponen de su propio dispositivo y ambos tienen instalada la aplicación. De lo contrario, cuando se vaya a realizar la conexión, simplemente esta no se llegará a establecer. Tanto para un tipo de tecnología como para otro, en la pantalla principal, debajo del nombre de la app, habrá escrito un pequeño texto que nos indica si el dispositivo está actualmente conectado con otro. Si no hemos utilizado la opción de conectividad, no saldrá nada escrito. Si estamos usando bluetooth, nos podrán salir los siguientes mensajes:

- *BT - Connected*: el dispositivo ha sido conectado correctamente con el otro, vía bluetooth.
- *BT - Not connected*: ha habido algún error y no se han enlazado los dispositivos.
- *BT - Connecting*: la conexión se está estableciendo.

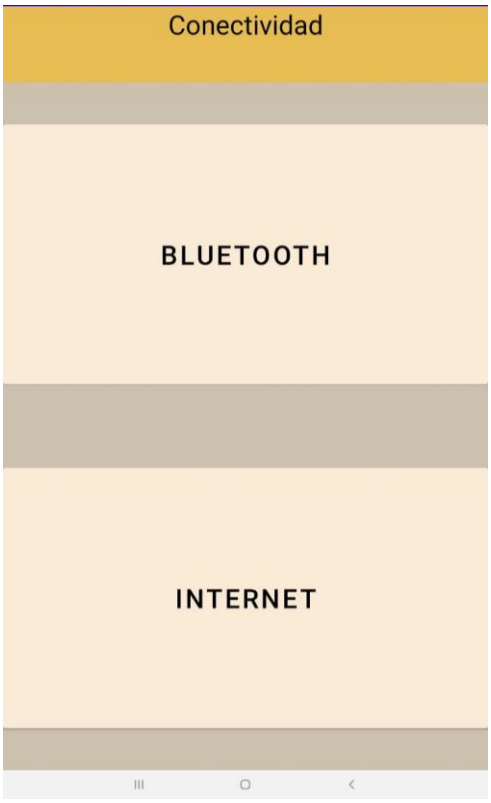


Figura 11. Pantalla de Conectividad

Si, por el contrario, estamos utilizando la opción de internet, cuando ambos dispositivos estén conectados a la sala, aparecerá el mensaje *Online – Connected*. A continuación, estudiaremos cada una de estas opciones por separado, indicando el funcionamiento de cada una de las conexiones.

3.5.1 Bluetooth

Como ya hemos explicado anteriormente el funcionamiento del bluetooth, así como las librerías y los métodos que hemos utilizado, simplemente nos centraremos en la usabilidad de esta funcionalidad. Si los dos jugadores desean utilizar su propio dispositivo para jugar, y tienen la intención de realizar la conexión por bluetooth, deberán pulsar **ambos** el botón ***BLUETOOTH***, que les llevará a la siguiente pantalla:

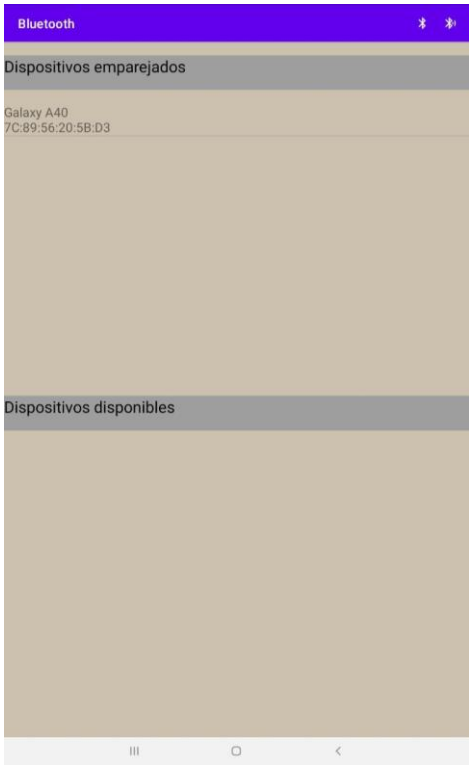


Figura 12. Pantalla de Bluetooth

Podemos observar dos zonas diferenciadas; en una, se nos muestra una lista con los dispositivos emparejados con el dispositivo. Esto significa que aquí aparecerán únicamente aquellos aparatos que han sido enlazados alguna vez con el dispositivo y que, por tanto, están ya guardados en él. Por otro lado, contamos con una lista de dispositivos disponibles, en la que nos aparecerán aquellos aparatos que están disponibles, pero que nunca han sido enlazados con el dispositivo en cuestión y, por tanto, no están emparejados con el mismo. Para que un dispositivo esté disponible y aparezca en la lista, este debe tener activado el bluetooth y estar situado en una zona cercana al otro dispositivo.

La primera vez que accedemos a esta pantalla, únicamente nos aparecerán los dispositivos emparejados. Hay que tener en cuenta que esta lista saldrá siempre; es decir, nos podrán aparecer dispositivos que han sido enlazados previamente pero que actualmente no se encuentran disponibles, cercanos o no tienen el bluetooth activado.

Por otro lado, podemos observar que hay dos pequeños botones en la parte superior de la pantalla. Ambos serán útiles en el caso de que nos queramos conectar con un dispositivo que no salga en la lista de emparejados. El botón que está situado más a la izquierda permite activar la visibilidad del dispositivo. Esto es importante, ya que permite que el dispositivo le pueda aparecer al otro jugador. Normalmente, esta opción no es necesario activarla, ya que el dispositivo suele encontrarse con facilidad; no obstante, si hay problemas a la hora de encontrar el dispositivo, ambos jugadores deben aplicar esta opción. En cuanto al botón de la derecha, este permite realizar un escaneo para localizar los dispositivos cercanos disponibles.

Para poder realizar la conexión, es indispensable que a ambos jugadores les aparezca el dispositivo del otro, ya sea en la lista de emparejados como en la de disponibles. Una vez consigamos esto, lo único que deberán hacer es pulsar sobre ese dispositivo. De nuevo, es importante que **ambos jugadores** lo hagan. Realmente, lo que conseguimos con esto es copiar la dirección del dispositivo con el que queremos conectarnos, pero la conexión aún no está realizada. Ésta se iniciará cuando volvamos a la pantalla principal. Una vez ambos dispositivos retornan a esta pantalla, si todo ha salido correctamente, les aparecerá a ambos el mensaje *BT – Connected*, indicando que ya están conectados vía bluetooth.

Ahora que los dispositivos están conectados, el modo de juego cambia ligeramente. En esta ocasión, al usuario que comience a jugar se le asignan las blancas, y al otro las negras. Desde ese momento, el jugador de las negras tendrá inhabilitado de forma permanente el botón de las blancas, y viceversa. En cuanto a los botones centrales, cuando la partida se realiza por bluetooth, solo el jugador cuyo turno está en marcha está capacitado para pausar la partida y para resetearla; el otro jugador tendrá estos botones desactivados, hasta que sea su turno. Así, evitamos que un jugador interfiera en el turno de su rival. A la hora de pausar la partida, como es lógico, ésta se parará en ambos dispositivos, y lo mismo ocurre para resetearla. Si después de establecer la conexión por bluetooth accedemos a los ajustes de la app con cualquiera de los dispositivos, la conexión seguirá establecida; no obstante, no es aconsejable realizar esta acción, puesto que es posible que la conexión se pierda.

La primera vez que se utilice la app y, en concreto esta función, el usuario deberá aceptar una serie de permisos, necesarios para que la aplicación, entre otras cosas, acceda a la ubicación actual. Esto es necesario para poder detectar los dispositivos cercanos cuando utilizamos el botón de escaneo.

3.5.2 Internet

Al igual que con la tecnología bluetooth, en este apartado nos centraremos únicamente en el funcionamiento a nivel usuario, y no en la tecnología que utiliza internamente, ya que ésta se explicó en el apartado 2.7 del Capítulo 2. En este caso, el proceso de conexión es mucho más sencillo que en el caso de la tecnología bluetooth. Lo único que debemos hacer es pulsar el botón ***INTERNET***, y este se pondrá de color verde. En este momento, el jugador ya ha entrado en la sala de Scaledrone. Al igual que ocurría con la otra alternativa de conexión, serán los dos jugadores quienes realicen esta acción. Solo entonces ambos estarán dentro de la misma sala, y podrán interactuar. Cuando regresen a la pantalla principal, a ambos usuarios les deberá aparecer el texto *Online – Connected*. Al contrario de lo que ocurría con el bluetooth, en este caso aunque solo haya un jugador en la sala, a éste le saldrá *Online – Connected*. Por eso, es importante que ambos jugadores comprueben que les aparece este mensaje. Con el bluetooth ocurría algo diferente; desde que uno de los jugadores no estuviera conectado, a ambos les aparecía que están desconectados, de modo que era más sencillo realizar la comprobación de conexión.

Al contrario de lo que ocurría con el bluetooth, desde que el usuario pulsa el botón de *INTERNET*, éste se conecta directamente a la sala, ya que no necesita del otro dispositivo para hacerlo. Es por ello que la usabilidad de esta tecnología resulta más sencilla para el usuario. Si el usuario desea desconectarse de esta opción, simplemente deberá pulsar de nuevo el botón y éste recuperará su color original. Asimismo, como ocurre en todo momento, el asistente de voz nos indicará que hemos conectado o desconectado internet.

Capítulo 4
Accesibilidad y guía de usuario

En este apartado se tratan algunos aspectos relacionados con el uso de la aplicación, enfocándonos especialmente en la adaptación para personas con discapacidad visual. Como se viene diciendo desde el principio, todas y cada una de las pantallas de la app cuentan con una serie de comandos de voz que permiten al usuario acceder a la funcionalidad de la aplicación de una forma mucho más sencilla. En este apartado comentaremos, en primer lugar, los reconocimientos de voz comunes para todas las pantallas, e iremos ampliando con todos y cada uno de los comandos de voz. Finalmente, haremos una pequeña tabla a modo de resumen, en la que se refleje toda la funcionalidad de esta tecnología adaptativa.

4.1 Aspectos generales

El reconocedor de voz no es más que un escuchador que permanece a la espera de que el usuario indique una frase o una palabra. Como es lógico, esta funcionalidad requiere que el usuario proporcione los permisos necesarios a la app para que esta tenga libre acceso al micrófono del dispositivo. Estos permisos serán solicitados únicamente la primera vez que ejecutamos la app, tras instalarla. A la hora de utilizar el escuchador, hay que tener en cuenta de qué manera funciona. Una vez comencemos a hablar, el reconocedor registrará toda la frase que digamos, hasta que hagamos una pausa prolongada; el sistema seguirá oyéndonos hasta que detecte un silencio, interpretando que hemos terminado. Es por ello que es conveniente indicar los comandos de voz de forma clara y continuada. Para evitar problemas, todos los comandos de voz registrados son de corta longitud, con frases de un máximo de tres palabras.

A la hora de utilizar el reconocedor, existían diferentes variantes. Una de ellas consistía en que, una vez se acceda a una nueva pantalla, se activara automáticamente el escuchador, de modo que el sistema permaneciera constantemente escuchando. Sin embargo, esta opción fue descartada por varios motivos. En primer lugar, por una cuestión de ahorro de recursos; la app ya utiliza ciertas tecnologías como el bluetooth y el protocolo de conexión por internet, que consumen recursos, de modo que queremos evitar un exceso en el consumo de recursos de la app. En segundo lugar, esta alternativa supondría que no solo ambos jugadores permanecieran en completo silencio durante toda la partida, sino que su entorno también lo hiciera; de lo contrario, cualquier palabra sería detectada por el reconocedor y procesada. Evidentemente, si la frase detectada no está dentro de la base de datos de comandos de voz, no se ejecutará ninguna acción, pero esto no evita que la app procese el mensaje y lo busque en el registro. Además, cuando la frase detectada no se encuentra dentro de las disponibles, el asistente de voz nos indica que repitamos la orden; por ello, si la app está constantemente escuchando, el asistente permanecerá continuamente sugiriéndonos esta acción. Por estos motivos, lo que hemos hecho es establecer una forma de conectar el escuchador. En este caso, cada vez que pulsamos el botón de Subir Volumen del dispositivo, el escuchador se activa, y se queda a la espera de que el usuario indique el comando de voz. Por tanto, cada vez que un jugador quiera realizar alguna acción utilizando el reconocimiento de voz, simplemente tendrá que pulsar este botón y acto seguido indicar el mensaje deseado.

El primer paso a la hora de realizar un reconocedor de voz es establecer una base de datos en la que se almacenen los comandos de voz soportados. Para ello, lo que hemos hecho es utilizar *hashes* en cada una de las clases que representan los idiomas de la app. Así, cada vez que creamos un nuevo comando de voz lo añadimos en las tres clases *Spanish*, *English* y *German*. A continuación se muestra un ejemplo para el comando de voz utilizado para pausar el juego:

dictado.put("pausar", "Juego pausado");

dictado.put("pausar", "Game paused");

dictado.put("pausar", "Spiel gestoppt");

Cada una de estas líneas ha sido implementada en su clase correspondiente. De esta manera, en el hash ***dictado*** de la clase *Spanish* añadimos el valor "Juego pausado" con la clave "pausar". Seguimos el mismo procedimiento en las dos clases restantes, utilizando siempre la misma clave. Como el lenguaje seleccionado siempre será uno y solo uno de estos tres, no habrá claves repetidas; accederemos al *hash dictado* de la clase *Language* (la clase padre de los tres idiomas), y este se encargará de acceder a la clave "pausar" del lenguaje que corresponda.

Una vez creada la base de datos con los distintos comandos de voz, solo queda procesar el mensaje de entrada. Para ello, el procedimiento es muy sencillo. El escuchador tiene una serie de métodos por defecto, y en concreto el que nos interesa es ***onResults***. Cada vez que el escuchador procesa una palabra o una frase, este método se encarga de guardarla en un String. A partir de ahí, comenzamos a hacer el procesamiento de esta variable. En cada actividad crearemos un método *VoiceManagement*, que se encarga de procesar el contenido de la variable que almacena el comando de voz. Así, lo único que haremos en este método será un *switch* en el que comprobemos si el mensaje obtenido se corresponde con ciertos comandos de voz guardados en el *hash* del idioma correspondiente. En caso afirmativo, se llevarán a cabo las acciones pertinentes. Es importante entender que cada una de las actividades tendrá su propio método *VoiceManagement*, ya que en cada pantalla tendremos habilitados unos comandos de voz determinados, y otros no. Por ejemplo, el comando de voz del ejemplo anterior, utilizado para pausar los relojes, queremos que esté habilitado únicamente cuando nos encontramos en la pantalla principal; en el resto de pantallas no interesa que esté habilitado. Todas las actividades heredan de la clase *SuperActivity*. Es en esta clase donde implementamos el escuchador, y donde creamos el método *VoiceManagement*. De esta forma, desde la clase padre se accederá al método *VoiceManagement* de la actividad correspondiente.

Solo existe un comando de voz que se repite en cada actividad. Este es el comando "atrás" que, al ser utilizado, retrocede a la pantalla anterior. Nos interesa tener este comando activo en cada pantalla, aunque en la pantalla principal sí que lo inhabilitaremos. Por otro lado, en cada actividad el asistente de voz está preparado para indicarnos que repitamos el mensaje si este no concuerda con ningún comando de la base de datos. Como indicamos anteriormente, todas las acciones que se pueden hacer manualmente, también están soportadas para realizarse mediante el reconocedor de voz. A continuación se muestra una guía de usuario, pantalla por pantalla, indicando los comandos de voz disponibles en cada una.

4.2 Pantalla principal

Recordemos que esta es la pantalla más importante de la aplicación, puesto que es en la que se desarrollará el juego. Los comandos de voz habilitados para esta pantalla se comentan a continuación.

- **"Ajustes"**: abre la pantalla de *Ajustes*.

- **“Pausa”**: pausa la partida.
- **“Parar”**: resetea la partida.
- **“Blancas”**: indica el tiempo restante de las blancas.
- **“Negras”**: indica el tiempo restante de las negras.
- **“Ya”**: cambia de turno. Es la alternativa a accionar el pulsador.
- **“Salir”**: sale de la aplicación.
- **“Penalización blancas”**: accede a la pantalla de penalización de las blancas.
- **“Penalización negras”**: accede a la pantalla de penalización de las negras.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
ajustes	settings	einstellungen	Abrir Ajustes
pausa	pause	pause	Pausar partida
parar	stop	halt	Resetear partida
blancas	white	weiß	Tiempo de las blancas
negras	black	schwarz	Tiempo de las negras
ya	go	gleich	Pulsador
salir	exit	beenden	Salir de la app
penalización blancas	white penalization	Weiß bestrafung	Penalización a las blancas
penalización negras	black penalization	Schwarz bestrafung	Penalización a las negras

Tabla 1. Pantalla principal

4.3 Pantalla de Ajustes

En esta pantalla tendremos otros comandos de voz diferentes a los creados en la pantalla principal, puesto que la funcionalidad entre ambas actividades es completamente diferente.

- **“Ajustes de voz”**: abre la pantalla de *Ajustes de Voz*.
- **“Idioma”**: indica el idioma en que está la aplicación en ese momento.
- **“Español”**: cambia el idioma a español.
- **“English”**: cambia el idioma a inglés.
- **“Deutsche”**: cambia el idioma a alemán.
- **“Ajedrez clásico”**: cambia el modo de juego a Ajedrez Clásico.
- **“Ajedrez rápido”**: cambia el modo de juego a Ajedrez Rápido.
- **“Ajedrez blitz”**: cambia el modo de juego a Ajedrez Blitz.
- **“Personalizar”**: abre la ventana de personalización del juego.
- **“Atrás”**: retrocede a la actividad anterior (pantalla principal).

- **“Salir”**: sale de la aplicación.
- **“Conectividad”**: abre la pantalla de *Conectividad*.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
ajustes de voz	voice settings	spracheinstellungen	Abrir Ajustes de voz
idioma	language	sprache	Indicar idioma actual
español	español	español	Establecer idioma español
english	english	english	Establecer idioma inglés
deutsche	deutsche	deutsche	Establecer idioma alemán
ajedrez clásico	classical chess	Klassisches schach	Modo Clásico
ajedrez rápido	rapid chess	Schnellschach	Modo Rápido
ajedrez blitz	blitz chess	blitzschach	Modo Blitz
personalizar	personalize	personifizieren	Modo Personalizar
atrás	back	rückkehr	Retroceder
salir	exit	beenden	Salir de la app
conectividad	connectivity	konnektivität	Abrir Conectividad

Tabla 2. Pantalla de Ajustes

4.4 Pantalla de Ajustes de voz

- **“Cambiar voz”**: cambia la voz del asistente.
- **“Tono” / “Cambiar tono”**: indica la frase “Tono actual” para comprobar el tono que hay establecido.
- **“Más grave”**: cambia el tono de voz del asistente, haciéndolo más grave.
- **“Más agudo”**: cambia el tono de voz del asistente, haciéndolo más agudo.
- **“Velocidad”**: indica la frase “Velocidad actual” para comprobar la velocidad establecida del asistente de voz.
- **“Más rápido”**: cambia la velocidad del asistente de voz, haciendo que hable más deprisa.
- **“Más lento”**: cambia la velocidad del asistente de voz, haciendo que hable más despacio.
- **“Desactivar asistente”**: desactiva el asistente de voz, en caso de estar activado.
- **“Activar asistente”**: activa el asistente de voz, en caso de estar desactivado.
- **“Atrás”**: retrocede a la actividad anterior (pantalla de Ajustes).
- **“Salir”**: sale de la aplicación.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
cambiar voz	change voice	stimme ändern	Cambiar la voz
tono / cambiar tono	pitch / change pitch	Ton / ton wechseln	Indica el tono actual
más grave	deeper	tiefer	Voz más grave
más agudo	higher	akuter	Voz más aguda
velocidad	speed	geschwindigkeit	Indica la velocidad actual
más rápido	faster	schneller	Subir velocidad
más lento	slower	langsamer	Reducir velocidad
desactivar asistente	disable assistant	sprachassistent deaktiviert	Desactivar asistente de voz
activar asistente	enable assistant	sprachassistent aktiviert	Activar asistente de voz
atrás	back	rückkehr	Retroceder a Ajustes
salir	exit	beenden	Salir de la app

Tabla 3. Pantalla de Ajustes de Voz

4.5 Pantalla de Personalizar

En este caso, el usuario podrá indicar por comandos de voz el número de horas, minutos y segundos que tendrá el juego, así como los segundos de incremento. Por lo tanto, la gestión del reconocimiento de voz se vuelve mucho más compleja. Así, hay que comprobar en primer lugar el número de palabras que ha dicho el usuario. Si solo ha dicho una, el único comando de voz permitido será "atrás"; de lo contrario, se solicitará la repetición de la instrucción. En caso de que haya dicho dos palabras, entonces debemos comprobar el formato de la frase. A continuación se detalla el formato que deberán tener las instrucciones.

- **Horas**: un número entre 0 y 23, seguido de la palabra "horas".
- **Minutos***:* un número entre 0 y 59, seguido de la palabra "minutos".
- **Segundos:** un número entre 0 y 59, seguido de la palabra "segundos".
- **Incremento:** la palabra "incremento" seguida de un número entre 0 y 59.
- **"Atrás"**: retrocede a la actividad anterior (pantalla de Ajustes).
- **"Salir"**: sale de la aplicación.

Para realizar este reconocimiento de forma más sencilla, se han utilizado expresiones regulares que comprueban si se sigue el formato establecido. Se comprueba, en primer lugar, que alguna de las dos palabras sea un número, y que esté en el rango establecido. En segundo lugar, se comprueba que se haya continuado la orden de forma correcta. Se muestran a continuación las dos expresiones

regulares utilizadas, una para el tiempo de la partida, y la otra para los segundos de incremento.

```
String pattern = "(([1-9])|([1-5][0-9]))\\s(minuto|hora|segundo)s?";
String pattern = "incremento\\s(([0-9])|([1-5][0-9]))";
```

Después de comprobar que las instrucciones casan con estas expresiones, simplemente se comprobará cuál de las palabras se indicó, para establecer el tiempo correctamente.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
[0-23] horas	[0-23] hours	[0-23] stunden	Número de horas
[0-59] minutos	[0-59] minutes	[0-59] minuten	Número de minutos
[0-59] segundos	[0-59] seconds	[0-59] sekunden	Número de segundos
incremento [0-59]	increment [0-59]	zuwachs [0-59]	Segundos de incremento
atrás	back	rückkehr	Retroceder a Ajustes
salir	exit	beenden	Salir de la app

Tabla 4. Pantalla de Personalización

4.6 Pantalla de Conectividad

Esta pantalla, al ser muy básica, cuenta con muy pocos comandos de voz, que se detallan a continuación.

- **"Bluetooth"**: abre la pantalla de *Bluetooth*.
- **"Internet"**: activa la opción de internet y cambia el botón de color.
- **"Atrás"**: retrocede a la actividad anterior (pantalla de Ajustes).
- **"Salir"**: sale de la aplicación.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
bluetooth	bluetooth	bluetooth	Abrir pantalla Bluetooth
internet	internet	internet	Conectar internet
atrás	back	rückkehr	Retroceder a Ajustes
salir	exit	beenden	Salir de la app

Tabla 5. Pantalla de Conectividad

4.7 Pantalla de Penalización

La gestión del reconocimiento de voz de esta pantalla es prácticamente igual a la de la pantalla de *Personalizar*. De nuevo, utilizamos expresiones regulares para comprobar que el comando de

voz cumple con el formato establecido. El usuario utilizará el reconocimiento de voz para indicar el número de minutos y de segundos que se penalizará al jugador.

- **Minutos***:* un número entre 0 y 59, seguido de la palabra "minutos".
- **Segundos:** un número entre 0 y 59, seguido de la palabra "segundos".
- **"Atrás"**: retrocede a la actividad anterior (pantalla principal).
- **"Salir"**: sale de la aplicación.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
[0-59] minutos	[0-59] minutes	[0-59] minuten	Número de minutos
[0-59] segundos	[0-59] seconds	[0-59] sekunden	Número de segundos
atrás	back	rückkehr	Retroceder a Ajustes
salir	exit	beenden	Salir de la app

Tabla 6. Pantalla de Penalización

4.8 Pantalla de Bluetooth

En esta pantalla, el reconocimiento de voz es muy importante, ya que con él podemos acceder a ciertas prestaciones que de forma manual no podemos obtener. Así, se detallan a continuación los comandos de voz establecidos, explicando brevemente su cometido y el resultado obtenido.

- **"Emparejados"**: lee la lista de dispositivos emparejados. El asistente indica *"Dispositivos emparejados:"* y comienza a leer la lista, diciendo únicamente el nombre del dispositivo, y obviando la dirección. Si no hay ningún dispositivo en esta lista, el asistente dirá *"Dispositivos emparejados: ninguno"*.
- **"Disponibles"**: lee la lista de dispositivos disponibles. El asistente indica *"Dispositivos disponibles:"* y comienza a leer la lista, diciendo de nuevo únicamente el nombre del dispositivo. Si la lista está vacía, el asistente dirá *"Dispositivos disponibles: ninguno"*.
- **"Atrás"**: retrocede a la actividad anterior (pantalla principal).
- **"Salir"**: sale de la aplicación.

ESPAÑOL	INGLÉS	ALEMÁN	ACCIÓN
emparejados	paired	gekoppelte	Dispositivos emparejados
disponibles	available	verfügbare	Dispositivos disponibles
atrás	back	rückkehr	Retroceder a Ajustes
salir	exit	beenden	Salir de la app

Tabla 7. Pantalla de Bluetooth

Capítulo 5
Conclusiones y líneas futuras

5.1 Conclusiones

La aplicación desarrollada cumple con los mismos objetivos que un reloj de ajedrez adaptado para personas con discapacidad visual. Además, ofrece una funcionalidad significativamente mayor, ya que el jugador podrá modificar las características del asistente de voz y mejorar así la experiencia de usuario. Por otro lado, la app cuenta con un reconocedor de voz, de modo que los usuarios podrían utilizar todas y cada una de las prestaciones de la aplicación sin necesidad de mantener un contacto físico con el dispositivo. Por todo ello, esta aplicación supone una interesante alternativa al reloj de ajedrez convencional. Además, hay que tener en cuenta que su acceso es gratuito, y que resulta una forma mucho más cómoda y accesible de utilizar un reloj de ajedrez. El jugador únicamente deberá contar con un dispositivo android relativamente nuevo, y podrá utilizar el reloj sin ningún problema ni coste adicional.

5.2 Líneas futuras

A partir de este momento, la aplicación queda abierta a un conjunto de posibles modificaciones que puedan mejorar las prestaciones actuales o añadir nuevas funcionalidades. Sin embargo, antes de realizar cualquier cambio, un primer paso obligatorio que deberá tener lugar es el testeo de la aplicación por parte de un usuario con discapacidad visual. Si bien la aplicación ha sido probada por distintos usuarios, la opinión más relevante para el desarrollador de la app es la de aquel usuario que verdaderamente necesite utilizar el asistente de voz, el reconocimiento de voz y los medios adaptados. Será entonces cuando se conozcan con exactitud los puntos débiles de la aplicación y se puedan tomar las medidas oportunas. No obstante, a continuación se detallarán algunos de los aspectos que, aun sin haberse podido llevar a cabo el testeo mencionado, se ha comprobado que necesitan algunas mejoras.

- **Añadir detección de gestos**. Como explicamos anteriormente, se prefirió desarrollar un reconocedor de voz para evitar el contacto excesivo con el dispositivo, pero una vez implementada esta tecnología, se podrían añadir también ciertos gestos que puedan facilitar el uso de la aplicación. Además, podría resultar más rápido que el reconocimiento de voz.
- **Ampliar la base de datos de los comandos de voz**. En este sentido, se busca que el reconocedor de voz se vuelva lo más flexible posible. De esta forma, se podría ampliar la base de datos de cada idioma, de modo que se pueda acceder a la misma funcionalidad indicándolo de varias formas distintas. Esto permitiría que el reconocedor no fuera tan estricto, y que el usuario no necesitara memorizar de forma exacta tantos comandos de voz. En este sentido, se podría incluso aplicar la Inteligencia Artificial para mejorar este

aspecto, concretamente algún algoritmo de Procesamiento de Lenguaje Natural.

- **Añadir nuevos idiomas**. Éste es quizás el punto con más posibilidad de ampliación de la aplicación. Como ya se ha comentado, la app solo está disponible en tres idiomas, pero tanto el reconocedor de voz como el asistente utilizan unas bases de datos que soportan una gran cantidad de idiomas. Añadir nuevos idiomas a la aplicación es relativamente sencillo, aunque habría que traducir todos los comandos de voz existentes.

- **Mejorar la traducción al alemán**. El total desconocimiento de este idioma por parte del desarrollador implica que probablemente existan algunas traducciones que sean erróneas. En este sentido, bastaría con que un conocedor del alemán y del español o el inglés comprobara si los términos de ajedrez utilizados están traducidos de forma correcta.

- **Añadir más modos de juego**. En la versión actual de la aplicación únicamente se han añadido los tres modos de juego más utilizados. Sin embargo, existe una gran cantidad de variantes que se pueden añadir a la lista de modos de juego disponibles.

- **Mejorar el tiempo de respuesta**. Este es uno de los problemas más importantes con los que cuenta la aplicación. En un reloj de ajedrez convencional, el pulsador es un dispositivo mecánico, de modo que la respuesta de los relojes es prácticamente inmediata. En la aplicación, este sistema funciona con prácticamente la misma velocidad; no obstante, presenta algunas deficiencias cuando la partida se realiza por bluetooth o por internet. Así, cuando se cambia el turno en una partida con cualquiera de estas conexiones, el dispositivo donde se cambia el turno realiza los cambios en los relojes de forma inmediata, pero en el otro dispositivo se tarda algunas décimas de segundo en aplicarse los cambios. Esto implica que a la larga, se cuente con un pequeño desfase que implica diferencias más significativas en los tiempos de cada jugador. Para mejorar este aspecto, habría que estudiar en profundidad el código referente a la parte de programación multihilo, e investigar la manera de sincronizar al máximo ambos dispositivos, sobre todo en el caso de las partidas por internet; en bluetooth, el desfase temporal es menos pronunciado.

- **Ampliar las voces disponibles**. La base de datos de voces que se ha utilizado cuenta con un amplio número de voces diferentes, de las cuales se han escogido solo unas pocas. Siempre se pueden añadir más voces a la aplicación, para que el usuario tenga más repertorio donde elegir.

- **Mejorar la interfaz**. El diseño de la aplicación será un apartado que siempre se podrá mejorar, intentando siempre maximizar la facilidad de uso y la adaptabilidad de los usuarios con discapacidad visual.

Capítulo 6
Summary and Conclusions

6.1 Conclusions

The developed application achieves the same objectives as a chess clock adapted for visually impaired people. In addition, it offers significantly more functionality, since players will be able to modify the characteristics of the voice assistant and thus improve user experience. Moreover, the app has a voice recognizer, so that users could use each and every feature of the application without having to maintain physical contact with the device. Therefore, this application is an interesting alternative to the conventional chess clock. In addition, it must be taken into account that its access is free, and that it is a much more comfortable and accessible way of using a chess clock. The player will only need to have a relatively new android device, and will be able to use the clock without any problem and at no cost.

6.2 Future development

From now on, the application is open to a set of possible modifications that may improve current features or add new functionalities. However, before making any changes, a mandatory first step that must be taken is the testing of the application by a visually impaired user. Although the application has been tested by different users, the most relevant opinion for the app developer is the one of the user who really needs to use the voice assistant, voice recognition and adapted media. Then, the weak points of the application will be known exactly and the appropriate measures can be taken. However, some of the aspects that, even without having been able to carry out the aforementioned test, have been found to need some improvements will be detailed below.

- **Add gesture detection**. As we explained previously, it was preferred to develop a voice recognizer to avoid excessive contact with the device, but once this technology was implemented, certain gestures could also be added in order to facilitate the use of the application. It also could be faster than speech recognition.
- **Expand the database of voice commands**. In this sense, it is sought that the voice recognizer becomes as flexible as possible. In this way, the database for each language could be expanded, so that the same functionality can be accessed by indicating it in several different ways. This could allow that the recognizer was not so strict, and that user did not need to memorize exactly as many voice commands. In this sense, Artificial Intelligence could even be applied to improve this aspect, specifically some Natural Language Processing algorithm.
- **Add new languages.** This is perhaps the point with the most capacity of expansion for the application. As already mentioned, the app is only available in three languages, but both the voice recognizer and the assistant use databases which support a large number of languages. Adding new languages to the application is relatively straightforward,

although all existing voice commands would have to be translated.

- **Improve the German translation.** The ignorance of this language by the developer implies that there are probably some translations that are wrong. In this sense, it would be enough for a connoisseur of German and Spanish or English to check whether the chess terms used are translated correctly.
- **Add more game modes.** In the current version of the application, only the three most used game modes have been added. However, there are a large number of variants that can be added to the list of available game modes.
- **Improve response time.** This is one of the most important problems the application has. In a conventional chess clock, the pusher is a mechanical device, so the response of the clocks is practically immediate. In the application, this system works with practically the same speed; however, it has some shortcomings when the game is made via Bluetooth or via internet. Thus, when the turn is changed in a game with any of these connections, the device where the turn is changed makes the changes to the clocks immediately, but it takes a few tenths of a second for the other device to apply the changes. This implies that in the long run, there is a small lag that provokes more significant differences in the times of each player. In order to improve this aspect, it would be necessary to make a deep study of the code referring to the multithreaded programming part, and investigate how to synchronize both devices as much as possible, especially in the case of internet games; in Bluetooth, the time lag is less pronounced.
- **Expand the available voices.** The voice database that has been used has a large number of different voices, from which only a few have been chosen. You can always add more voices to the app, so that user has more repertoires to choose from.
- **Improve the interface.** The design of the application will be a section that can always be improved, always trying to maximize the ease of use and adaptability for users with visual disabilities.

Capítulo 7
Presupuesto

En este capítulo se establece el presupuesto estimado del proyecto, teniendo en cuenta los costes materiales del mismo, así como el desarrollo de la aplicación en base a las horas invertidas.

7.1 Coste material

Tipos	Descripción	Cometido	Precio
Smartphone	Sistema Operativo Android, versión 11	Testeo de la app	200€
Ordenador	Sistema Operativo Windows 10	Creación de la app	1200€
Tablet	Sistema Operativo Android, versión 11	Testeo de bluetooth e internet	180€
TOTAL	-	-	**1580**€

Tabla 8. Resumen de tipos

7.2 Recursos humanos

Tarea	Horas empleadas	Coste por hora	Coste total
Investigación sobre relojes de ajedrez	6	15€	90€
Investigación sobre modos de juego y penalizaciones	10	15€	150€
Búsqueda de documentación, códigos y ejemplos	90	20€	1800€

Implementación del código	180	20€	3600€
Testeo de la aplicación	14	20€	280€
TOTAL	**300**	-	**5920€**

Tabla 9. Recursos humanos

7.3 Presupuesto final

Recurso	**Coste total**
Coste material	1580€
Recursos humanos	5920€
TOTAL	**7500€**

Tabla 10. Presupuesto final

Bibliografía

[1] Android Speech Recognizer, Speech Recognition
[2] Android Text To Speech, Text To Speech
[3] Android TTS Voice, Text To Speech Voice
[4] Scaledrone, Scaledrone
[5] Bluetooth overview, Introducción general a Bluetooth
[6] Bluetooth Chat Example en GitHub, Bluetooth Chat
[7] Android In-App Messaging, Android Chat Messaging Tutorial
[8] How to create an Android Chat App, Android Chat App using Firebase
[9] Android Chat Tutorial, Android Chat with Scaledrone
[10] How to build an Android Chat app, Android chat in Java
[11] Stack Overflow, StackOverflow
[12] Connect Bluetooth Devices, Android Studio Bluetooth Devices
[13] Ejemplo de Bluetooth Chat Service, Bluetooth Chat Service
[14] Aplicación de Chat por Bluetooth, Simple Bluetooth Chat Application
[15] Ejemplos Bluetooth en Stack Overflow, Bluetooth Examples for Android
[16] Android Bluetooth, Developing an Android Mobile Bluetooth Chat Messenger
[17] Leyes del Ajedrez, Wikipedia
[18] Penalizaciones del ajedrez, Penalizaciones
[19] Jugadas ilegales del ajedrez, Jugada ilegal
[20] Penalización por volcar piezas, Penalizaciones ajedrez
[21] Ajedrez ONCE, Ajedrez ONCE Sevilla
[22] Relojes, Relojes de ajedrez
[23] Android Studio, Android Studio
[24] Android TalkBack, TalkBack
[25] Ejemplo de reloj de ajedrez, Repositorio GitHub
[26] Código de la aplicación, Repositorio GitHub
[27] Aplicación ChessClock, ChessClock