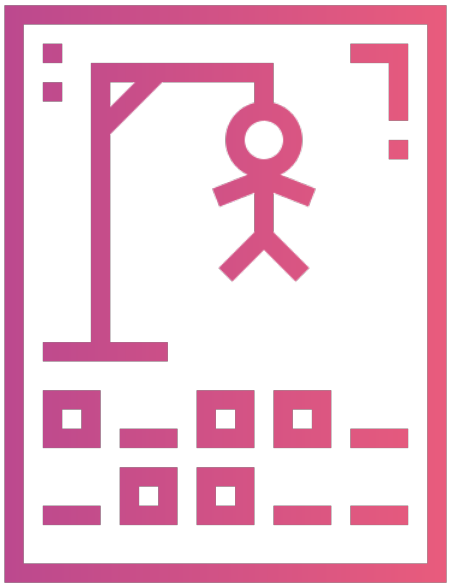
Facultade de Informática
UNIVERSIDADE DA CORUÑA

PROGRAMACIÓN DE SISTEMAS
Primeiro Cuadrimestre 2021/22

XOGO DO FORCADO PARA ANDROID

ForcApp



Estudante: Xico Fernández Lozano (xico.fernandez)
Estudante: Mario Páez Marcote (mario.paez)
Estudante: Andrés Filipe Oliveira da Silva (andres.oliveira)

A Coruña, xaneiro de 2022. Versión 1.0

Versión	Data
0.1	04/10/2021
0.2	08/11/2021
0.3	13/01/2021
0.4	15/01/2022
0.5	18/01/2022
0.6	19/01/2022
1.0	20/01/2022

Táboa 1: Táboa coas versións do proxecto.

Índice Xeral

Capítulo 1

Introdución

1.1 Obxetivos

Este proxecto consiste en elaborar un xogo para o sistema operativo **Android** [1] [2] [3] [4] basado no tradicional xogo do **Forcado**. O obxetivo da aplicación é poder xogar de maneira **individual** ou **multixogador** tratando de adiviñar a palabra secreta antes de que se forme por completo o forcado.

A aplicación estará enfocada ao xogo en local, é dicir, desde un único dispositivo aínda que contaremos cunha versión multixogador online ao final do proxecto. Implementarase un dicionario ao que o usuario lle pode engadir palabras e serían candidatas a ser seleccionadas aleatoriamente para ter que adiviñalas nunha partida.

1.2 Motivación

Visto o auxe do xénero do videoxogo na vida cotiá e a utilización de dispositivos móbiles estendidos cada vez máis en distintos rangos de idades dos usuarios decidimos decantarnos por implementar un videoxogo para dispositivo móbil sinxelo, fácil de utilizar, entendible e apto para todo tipo de público.

1.3 Traballo relacionado

Hai diversas aplicacións na Play Store nas que desenvolveron o mesmo xogo. As dúas con maior impacto a nivel de descargas (> 10 millóns). Ambas aplicacións amosan unha interface familiar e intuitiva, podemos tomar como comparativa estas dúas aplicacións para partir do deseño, xa que este videoxogo en concreto non ofrecen moitas posibilidades neste aspecto. Nas que nos estamos baseando, son as seguintes:

Ahorcado desarrollado por Senior Games:

Ahorcado

Senior Games Puzles 387.262

Para todos

Contiene anuncios · Ofrece compras en la aplicación

Esta aplicación está disponible para algunos de tus dispositivos

Añadir a la lista de deseos Instalar

Ahorcado desarrollado por Quarzo Apps:

El Ahorcado - en español

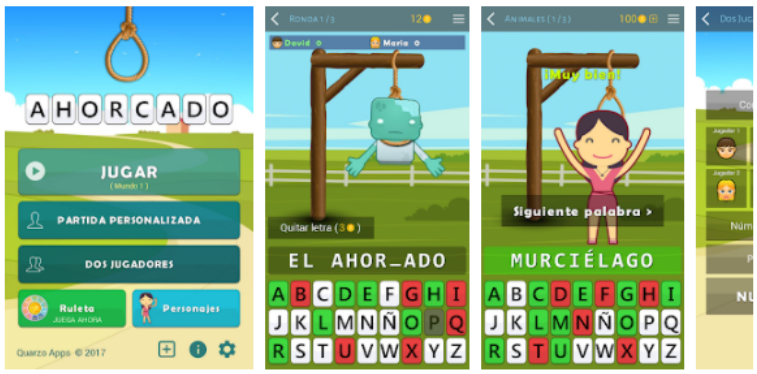
Quarzo Apps Puzles 58.145

Para todos

Contiene anuncios · Ofrece compras en la aplicación

Esta aplicación está disponible para algunos de tus dispositivos

Añadir a la lista de deseos Instalar



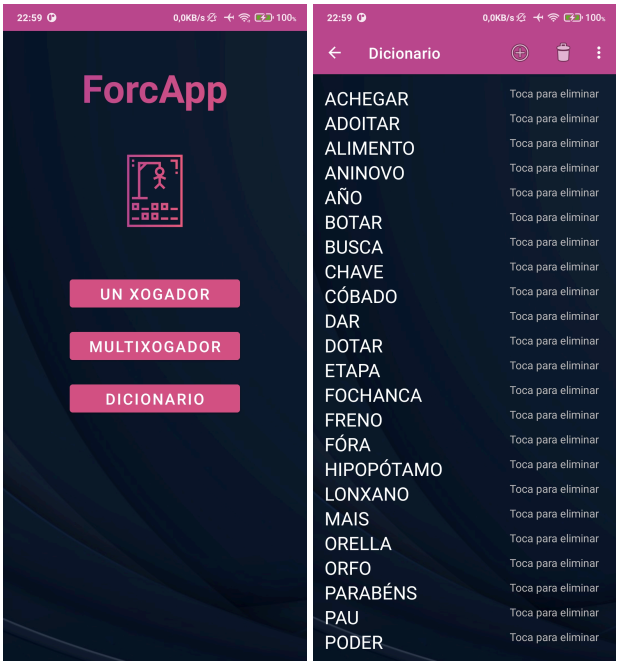
Capítulo 2

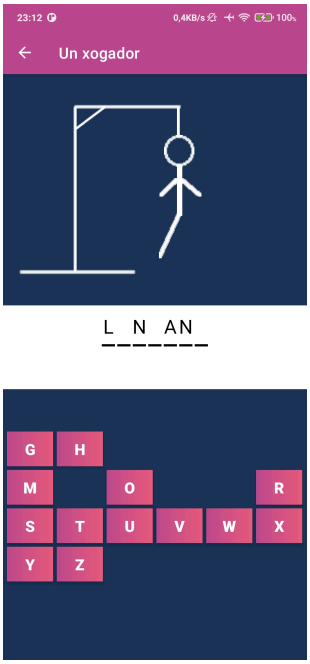
Análisis de requisitos

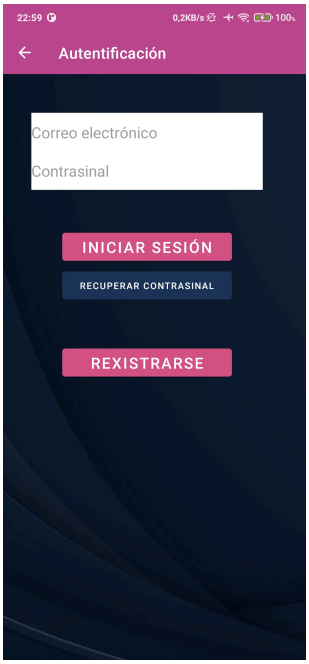
2.1 Funcionalidades

Esta aplicación dispoñerá de catro funcionalidades principais:

- **Menú principal**: Funcionalidade ”de enlace” co resto das funcionalidades. É a base da aplicación.
- **Dicionario**: Apoiarase nunha base de datos deseñada con *Room* para gardar as palabras e poder seleccionar unha aleatoriamente á hora de comezar unha partida. Poderanse eliminar e engadir palabras. Non se permitirá utilizar o dicionario baleiro no resto de funcionalidades. Non se poderá introducir unha palabra maior a 17 caracteres por cuestión de espazo. Haberá comprobación dos caracteres introducidos.
- **Modo un xogador**: Esta funcionalidade consistirá en permitir ao usuario poder xogar de maneira individual. Neste caso, a aplicación escollerá unha palabra aleatoria dende un dicionario (podendo ser este ampliable polos usuarios) situado nos propios arquivos da aplicación e presentaralla ao usuario por pantalla para que trate de adiviñala. Ademais a aplicación amosará un teclado ’artificial’ sen necesidade de usar un do sistema. Ao seleccionar unha letra dese teclado amosarase sombreada e non será posible volver a seleccionala.
- **Modo multixogador**: Nesta funcionalidade permitirase a dous usuarios xogar un contra outro. O funcionamento segue o mesmo modo que o dun xogador (hereda a súa vista) pero cun contador no que se amosa o tempo restante para elexir unha letra. Se este tempo expira, o xogador será eliminado. Gañará o xogador que adiviñe antes a palabra para evitar situacións de empate ou o que quede sen ser eliminado. Esta funcionalidade encárgase da parte de autencicación de usuarios e comunicacións en liña. Será necesario engadir todo o proceso de autentificación e unha sala de espera virtual para iniciar a partida cando os dous xogadores se presenten como listos. [5] [6] [7] [8]







A prioridade de implementación das funcionalidades seguirá a orde descrita anteriormente, sendo o menú principal a primeira funcionalidade en ser implementada, o modo multixogador o último.

2.2 Traballos futuros

En traballos futuros pódense engadir novas funcionalidades como levar a conta de partidas gañadas e perdidas (pódense calcular estatísticas como porcentaxes de éxito ou número de intentos medios para obter unha victoria) en calquera dos dous modos, para isto faría falta outra base de datos a maiores, por último conviría desenvolver un multixogador en liña para máis de dous xogadores.

Plantexamos elaborar tamén un proceso de testeo da aplicación coa axuda de usuarios e distintos tipos de probas de validación do software. Se se chegase a distribuír a aplicación cumpriría mantela con parches e actualizacións.

Tamén se considera a idea de seguir un modelo para o software da aplicación como pode ser o MVC que separa a vista do modelo mediante un controlador. Sería importante engadir soporte para diferentes idiomas e dispositivos en función da súa pantalla e do tamaño de fonte definido polo usuario a nivel de sistema operativo. De momento solo está pensado para estes parámetros en modo predeterminado.

Capítulo 3

Planificación Inicial

3.1 Iteracións

O proxecto fragmentarase na iteración de presentación do proxecto (*brainstorming*, preparación dos documentos necesarios), as tres primeiras funcionalidades *offline* a o multixogador *online* por últimno. Como iteracións complementarias están os traballos futuros e as probas de validación do software.

3.2 Responsabilidades

Por dispoñibilidade temporal, a idea é que todos os integrantes do grupo implementen as funcionalidades xuntos xa que é improbable que se solapen os traballos de dous recursos diferentes.

Tódolos membros do grupo terán a mesma porcentaxe de responsabilidade aínda que o esperable é que os traballos se vaian asignando dinamicamente e sobre a marcha do proxecto segundo sexa necesario.

Botaremos man do control de versións *git* utilizando *GitFlow* definindo as diferentes ramas para facilitar o traballo. Crearase unha rama *latex* exclusiva para gardar o código fonte da elaboración deste *PDF* e o documento en si. Obviaremos a rama *release* ao tratarse dun proxecto pequeno.

3.3 Fitos e entregables

Haberá un total de 2 fitos, un para cada unha das funcionalidades a realizar cada unha cun mes de duración. Primeiro farase o modo dun xogador e todo o que conleva (dicionario e

menú principal), facer todas as vistas e código para que se poida xogar con normalidade. Este punto marcará o fin do primeiro fito.

No segundo , partiremos do traballo xa relaizado para elaborar a lóxica e as vistas do modo multixogador, ao estar rematado darase por finalizado o segundo fito.

3.4 Incidencias e plans de continxencia

O principal risco que se vai ter en conta en conta debido á súa probable materialización son os contratempos á hora do desenvolvemento *software* e resolución de *bugs*. Este risco está presente en case todos os proxectos *software* e aínda máis cando se trata dun proxecto non profesional e con falta de experiencia na metodoloxía.

A solución aplicada para isto é tratar de adiantar o proxecto máximo posible coa mentalidade pesimista de que chegará algún contratempo que retrase o proceso.

Para a solución de *bugs* na interface intentaranse forzar as funcionalidades da aplicación o máximo posible e intentar solucionalos modificando o código. Para a solución de excepcións da linguaxe (*Java* e *xml*) botaremos man do *debugger* para ver os valores dos datos da aplicación en cada momento e ver o que sucede en cada intre para distintos valores. Para verificar o correcto funcionamento da base de datos utilizarase a funcionalidade *App Inspector* incorporada en *Android Studio*.

Como incidencias destacables ganaron importancia a xestión do cambio de tema (entre escuro e claro) do sistema operativo xa que daba lugar a basantes *bugs* e a concurrencia á hora de acceder a un resultado que se necesitaba de inmediato durante unha consulta a *Firebase*. Tamén un erro de programación non detectado ata case finalizado o proxecto que non incluía a letra 'Ñ' no teclado entón as palabras con esa letra nunca podían ser adiviñadas.

Capítulo 4

Deseño

4.1 Arquitectura proposta

A arquitectura proposta aísla o código de desenvolvemento da base de datos e todas as funcións relacionadas con *Room* do dicionario do resto da aplicación (modelo).

Propúxose unha interface para implementar os métodos en común entre os dous modos de xogo. Plantexouse elaborar unha clase abstracta que herdasen os dous modos de xogo á hora de xerar a partida pero as dúas clases xa extendían a *AppCompatActivity* entón decantámonos pola interfaz.

Seguimos unha organizaciónn sinxela na que cada pantalla corresponde a unha actividade en *Android* e unha clase en *Java*, ademais destas clases son necesarias outras para a configuración de *Room* e de *Firebase* e outra oara o adaptador da lista de palabras amosada no dicionario.

4.2 Persistencia

Almacenarase un dicionario coas palabras introducidas nunha base de datos. Non será posible abandoar o dicionario deixándoo baleiro e se se destrúe a aplicación restablécese.

Coa axuda de *Firebase* almacenaranse os usuarios rexistrados na aplicación co seu correo e a súa identificación.

Plantéxase a idea de gardar tamén un historial das partidas para poder elaborar un cadro estatístico de cada perfil e poder ser consultado, pero esta idea quedará para traballos futuros.

4.3 Vista

Para o comezo a impementación da vista axudámonos dun *wireframe* .dispoñible no repositorio do proxecto, que proporcionaríanos unha idea a alto nivel de como sería o deseño da aplicación

Valoráronse facer catro funcionalidades principais distintas coas súas correspondentes vistas. A primeira delas para o menú principal, constará de tres botóns. Un para o modo de un xogador, outra para o modo multixogador e outro para o acceso ao dicionario da aplicación. Amosaranse tamén o logo da aplicación e o nome.

Seguindo a orde dos anteriores botóns, será necesaria unha vista para a partida de un solo xogador que constará dun debuxo do forcado [9] con diferentes partes que irán modificando a súa visibilidade con cada fallo dinamicamente, un recadro para os ocos das letras da palabra a adiviñar e un botón por cada letra deseñado programaticamente no que ao pulsar unha letra fanse as correspondentes comprobacións e desactívase o botón pulsado. Habilitaríase o botón de atrás na *ActionBar*.

Para o multixogador serían necesarias máis vistas. Unha para o xogo en si, similar á anterior pero con algún detalle como un contador [10] entre pulsacións e outra con dous botóns na que se poida crear unha sala para a partida ou unirse a unha mediante un código; se o usuario está autenticado amósase tamén un botón de pechar sesión. E para a parte de autenticación (soamente funcional dentro do multixogador) serían necesarias tres vistas principais. A primeira delas unha pantalla na que se amosan tres botóns para inicar sesión, rexistrarse e recuperar o contrasinal ademais de dous *EditText* para o correo e o contrasinal. Se se preme no botón de recuperar o contrasinal só quedaría o *EditText* do correo e un novo botón de acción. Por outro lado ao premer o botón de rexistrarse quedaría un botón de acción cos *EditText* pertinentes para o rexistro dos datos.

Por último, para o dicionario utilizarase un *RecyclerView* coas palabras, que ao ser pulsadas poderán ser eliminadas, e un menú de opcións coa posibilidade de eliminar todas as palabras, restablecer o dicionario ou engadir unha palabra.

Cabe destacar que en case tódalas actividades axudarémonos dalgún *Toast* e bastantes *AlertDialog* para mellorar a experiencia do usuario.

4.4 Comunicacións

As comunicacións implementadas no modo de un xogador baséanse en consultas a unha base de datos. [11] [12] [13]

Por outro lado, no modo multixogador precisaranse comunicacións entre dous dispositivos distintos a través de internet. Este método baséase no típico modelo no que un xogador crea unha sala virtual cun código e o outro xogadore únese introducindo ese código. Neste aspecto axudarémonos de *Firebase* para elaborar unha autenticación dos usuarios e as comunicacións entre eles durante a partida. [14]

4.5 Sensores

Para a utilización desta aplicación non é necesario ningún hardware específico do dispositivo utilizado exceptuando os sensores WiFi para obter conexión a internet no modo multixogador (que a aplicación seguiría sendo funcional co modo dun xogador sen o funcionamento do sensor).
Necesitaranse os permisos adecuados no manifesto da aplicación, non se necesitarán máis permisos para respectar a privacidade do usuario e a experiencia (non resulta cómodo ter que estar concedendo permisos a unha aplicación de terceiros).

4.6 Traballo en background

No modo multixogador plantéxase facer un thread para o contador das partidas multixogador entre as seleccións de cada letra.

Tódalas consultas á base de datos á base de datos local do dicionario como a maioría (sobretodo as pesadas) á *Realtime Database* de *Firebase* faranse mediante *AsyncTask*s.

Bibliografía

[1] "Android developers," https://developer.android.com/.

[2] "How to switch night mode/ dark mode on and off programmatically in your android app? – programmerworld," https://programmerworld.co/android/how-to-switch-night-mode-dark-mode-on-and-off-programmatically-in-your-android-app/.

[3] "How can i disable landscape mode in android? - stack overflow," https://stackoverflow.com/questions/582185/how-can-i-disable-landscape-mode-in-android.

[4] "Start new activity and finish current one in android? - stack overflow," https://stackoverflow.com/questions/11308198/start-new-activity-and-finish-current-one-in-android.

[5] "Login con mail firebase y java en android studio parte 1 2021 - youtube," https://www.youtube.com/watch?v=aVyKywyFOzM.

[6] "thyrlian/awesomevalidation: Android validation library which helps developer boil down the tedious work to three easy steps." https://github.com/thyrlian/AwesomeValidation.

[7] "Login con mail firebase y java en android studio parte 2 2021 - youtube," https://www.youtube.com/watch?v=yQXZmC9ytjU.

[8] "android - firebase: how to check if user is logged in? - stack overflow," https://stackoverflow.com/questions/44583834/firebase-how-to-check-if-user-is-logged-in.

[9] "codigofacilito/hangman," https://github.com/codigofacilito/hangman.

[10] "Android - loading spinner," https://www.tutorialspoint.com/android/android_loading_spinner.htm.

[11] "android - firebase getvalue() not retrieving boolean correctly? - stack overflow," https://stackoverflow.com/questions/38671701/firebase-getvalue-not-retrieving-boolean-correctly.

[12] "android - error when getting an object from datasnapshot - stack overflow," https://stackoverflow.com/questions/47186438/error-when-getting-an-object-from-datasnapshot.

[13] "Firebase realtime database (android-java)- "class does not define a no-argument constructor" - stack overflow," https://stackoverflow.com/questions/68340491/firebase-realtime-database-android-java-class-does-not-define-a-no-argument.

[14] "android - binderproxy cannot be cast to clienttransaction exception when changing defaultnightmode - stack overflow," https://stackoverflow.com/questions/61483708/binderproxy-cannot-be-cast-to-clienttransaction-exception-when-changing-defaultn.