**Instituto Politécnico de Coimbra**

***Instituto Superior de Engenharia de Coimbra***

**Arquiteturas Móveis**

Trabalho Prático 1 – Android/Kotlin

**Docente: Álvaro Santos**

Marco António de Jesus Coelho – 2018012765 – LEI

Pedro Jorge Fernandes Morais – 2018020733 – LEI

segunda-feira, 9 de janeiro de 2023

**Índice**

[1 Variações de níveis 1](#_Toc124143140)

[2 GameBoard 2](#_Toc124143141)

[3 Game 4](#_Toc124143142)

[4 Singleplayer 5](#_Toc124143143)

[4.1 Seleção da equação 5](#_Toc124143144)

[4.2 Tempo restante 6](#_Toc124143145)

[4.3 Sair do jogo 6](#_Toc124143146)

[5 Login 7](#_Toc124143147)

[5.1 Email/Password 7](#_Toc124143148)

[5.2 Google 8](#_Toc124143149)

[6 Firebase 9](#_Toc124143150)

[6.1 Autenticação 9](#_Toc124143151)

[6.2 Storage 9](#_Toc124143152)

[6.3 Firestore Database 10](#_Toc124143153)

[7 Idiomas 12](#_Toc124143154)

[8 Orientações ecrã 13](#_Toc124143155)

**Índice de Figuras**

[Figura 1 - Constants 1](#_Toc124143156)

[Figura 2 - operatorFromString() 2](#_Toc124143157)

[Figura 3 - calculateValueOperation() 3](#_Toc124143158)

[Figura 4 - Game 4](#_Toc124143159)

[Figura 5 - Verificação seleção, Primeira linha e coluna 5](#_Toc124143160)

[Figura 6 - Timer tempo restante de jogo 6](#_Toc124143161)

[Figura 7 - Confirmação sair jogo atual 6](#_Toc124143162)

[Figura 8 - Autenticação Email/Password 7](#_Toc124143163)

[Figura 9 - Autenticação Google 8](#_Toc124143164)

[Figura 10 - Firebase Storage 9](#_Toc124143165)

[Figura 11 - Atualização da imagem de perfil na Firebase 10](#_Toc124143166)

[Figura 12 - Firestore Database 11](#_Toc124143167)

[Figura 13 - Criação ou atualização da pontuação na Firestore Database 11](#_Toc124143168)

[Figura 14 - Traduções 12](#_Toc124143169)

[Figura 15 - Orientação ecrã, Bloqueio orientação do Login 13](#_Toc124143170)

[Figura 16 - Orientação de ecrã, Modo Portrait 14](#_Toc124143171)

[Figura 17 - Orientação de ecrã, Modo Landscape 14](#_Toc124143172)

# Variações de níveis

No decorrer da evolução do jogo os parâmetros dos níveis vão evoluindo, sendo que até ao nível 3 só usamos números de 1 a 9, depois até ao nível 7 utilizamos números de 1 a 99 e depois usamos de 1 a 999.

Nas equações os operadores também se vão alterando, sendo que no primeiro nível é só equações de soma, no segundo soma e subtração, no terceiro soma, subtração e multiplicação e no quarto e em diante soma, subtração, multiplicação e divisão.

Cada nível tem um total de cinco tabuleiros de jogo.

Os tempos que o jogador tem para acabar cada nível também vão diminuindo, sendo que começa em 60 segundos e diminui 10 segundos a cada nível que passa, até um mínimo de 10 segundos por nível.

O jogador por cada resposta certa que der recebe 5 segundos extra, mas se errar uma resposta perde também 5 segundos.

Todas estas configurações estão no ficheiro de constantes (Constants.kt).

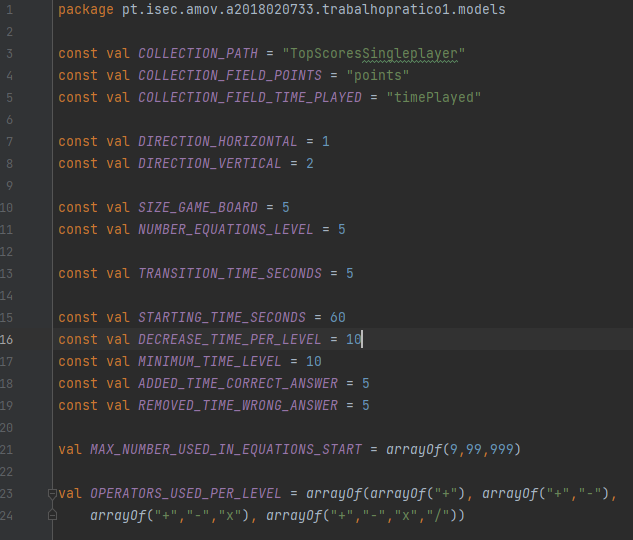


Figura - Constants

# GameBoard

Cada nível é constituído por vários tabuleiros de jogo, sendo que cada um tem duas equações que dão pontos ao utilizador. Os números e operadores de cada tabuleiro são aleatórios tendo em conta o nível atual. Cada tabuleiro guarda ainda o valor e posição da sua maior e segunda maior equação.

Está ainda definido na classe GameBoard duas funções estáticas, através do *companion* *object*, que permitem calcular o valor de uma dada equação. Para isto recebe a equação num Array de Strings, em que nas posições pares tem os valores numéricos e nas ímpares os operadores. O cálculo da equação é feito da esquerda para a direita, sendo que tem em conta as prioridades dos operadores, ou seja, a multiplicação e divisão é realizada antes da soma e subtração.

A função *operatorFromString()* permite realizar uma operação matemática enviando como argumento um operador.

*Uma imagem com texto

Descrição gerada automaticamente*

Figura - operatorFromString()

A função *calculateValueOperation()* recebe um *array* de *Strings* e realiza o cálculo da equação que é enviado no mesmo. Devolve o resultado da equação.



Figura - calculateValueOperation()

# Game

Para guardarmos os vários tabuleiros de jogo que vamos tendo, utilizamos a classe *Game*, sendo que esta estende da classe *ViewModel*, de forma a que quando a atividade seja recarregada, por exemplo a alterar a orientação do ecrã do dispositivo, não sejam perdidos os dados do jogo.

A classe *Game* guarda para além de um *array* com todos os tabuleiros de jogo para todos os níveis até agora decorridos, a pontuação do jogador, o tempo de jogo do jogador, quantos segundos faltam até o jogador perder se não completar o nível atual e qual o nível e equação em que o jogador se encontra atualmente.

Na inicialização da classe são criados os tabuleiros de jogo para o primeiro nível.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Game

# Singleplayer

## Seleção da equação

Para o utilizador selecionar uma equação é utilizado o gesto Fling, sendo que para isto a atividade *SingleplayerActivity* estende da classe *GestureDetector*.*OnGestureListener*. Quando um gesto *onFling* é detetado, são verificadas as coordenadas, sendo que o utilizador apenas precisa de realizar o gesto na linha ou coluna pretendida, sem ter a necessidade de começar num ponto específico e terminar num ponto específico.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Verificação seleção, Primeira linha e coluna

## Tempo restante

Para efetuar a contagem decrescente do tempo restante, e respetiva contagem crescente do tempo a que o utilizador está a jogar é utilizado a classe *Timer*.

Para isto utilizamos um *fixedRateTimer* que de um em um segundo decrementa o tempo restante e aumenta o tempo de jogo. Verifica ainda o término do jogo quando o tempo restante chega a zero.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Timer tempo restante de jogo

## Sair do jogo

Para o utilizador poder sair do jogo terá de carregar no botão *back* duas vezes seguidas num espaço de tempo de cinco segundos. O tempo e o segundo clique servem de verificação para confirmar que o utilizador pretende mesmo deixar o jogo atual.

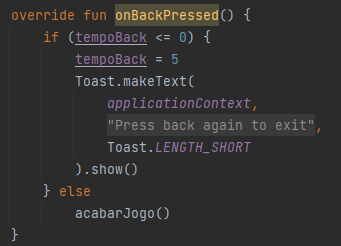


Figura - Confirmação sair jogo atual

# Login

Ao realizarmos o login na aplicação podemos fazê-lo de duas formas distintas:

* Através de um email e password;
* Através da conta Google.

Para realizarmos o login com o email e password, temos de antes utilizar o registo da aplicação e criarmos uma conta de utilizador. Já com a conta Google é um processo muito mais simples.

## Email/Password

Para realizarmos o login com email e password utilizamos a autenticação da Firebase, tendo de enviar estes dados para a Firebase confirmar a sua autenticidade.



Figura - Autenticação Email/Password

## Google

Para realizar a autenticação com a conta Google, do ponto de vista do utilizador basta selecionar a conta Google associada ao seu dispositivo, ou associar uma nova. Já no programa temos de utilizar o GoogleSignInClient para verificar a autenticação do mesmo, existindo ainda uma integração posterior com a Firebase para efetuar a autenticação na aplicação.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Autenticação Google

# Firebase

Para o desenvolvimento de algumas funcionalidades da nossa aplicação foi necessário a utilização da Firebase, nomeadamente para efetuar a autenticação de utilizadores, guardar as imagens de perfil dos utilizadores e armazenar as pontuações dos utilizadores.

## Autenticação

Como referido no capítulo do Login utilizamos a Firebase para efetuar a autenticação de utilizadores na nossa aplicação.

## Storage

De forma a guardarmos as imagens de perfil de cada utilizador utilizamos a Storage, em que temos uma pasta “users” que contém as imagens de perfil, sendo que estas são identificadas pelo seu nome que é igual ao email do utilizador a quem elas pertencem.

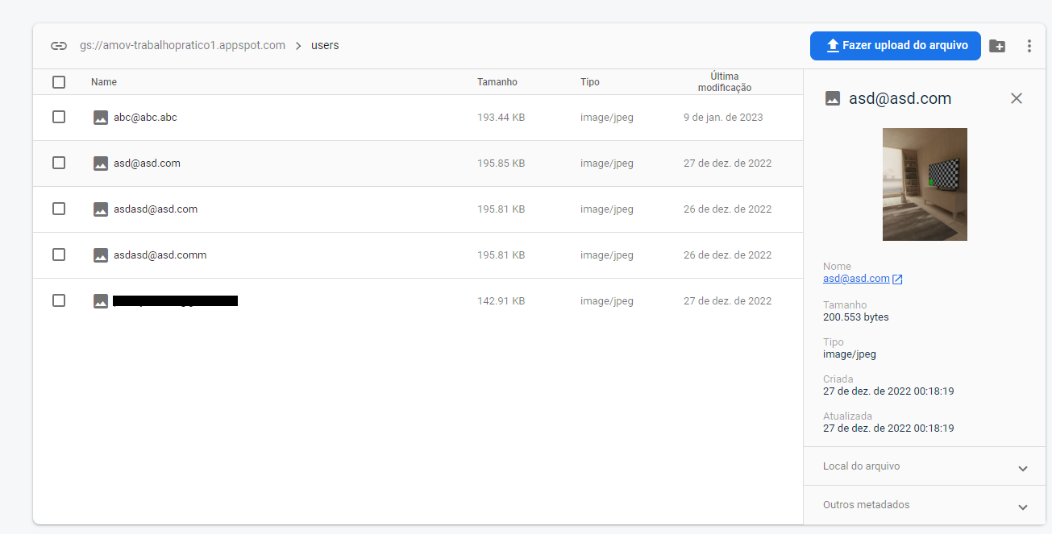


Figura - Firebase Storage

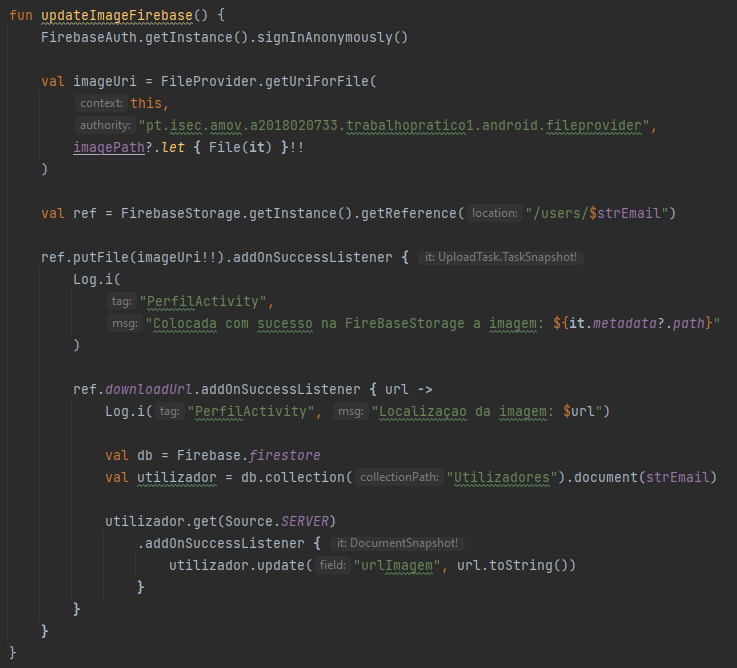


Figura - Atualização da imagem de perfil na Firebase

## Firestore Database

Para guardarmos as pontuações dos utilizadores utilizamos a *database* da *firestore*, sendo que esta permite ter coleções em que cada coleção tem vários documentos e cada documento tem vários campos. Utilizamos uma coleção chamada “TopScoresSingleplayer” onde temos vários documentos, sendo eles identificados pelo email do utilizador que representam. Cada um destes documentos tem dois campos, a maior pontuação do utilizador em questão e o tempo que este demorou a atingi-la.

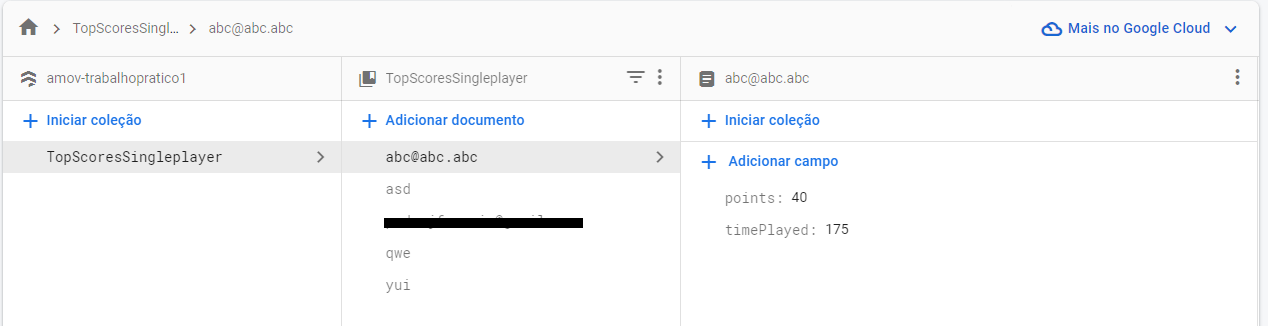


Figura - Firestore Database

Caso este utilizador ainda não tenha pontuação guardada, é criado um novo documento para ele.

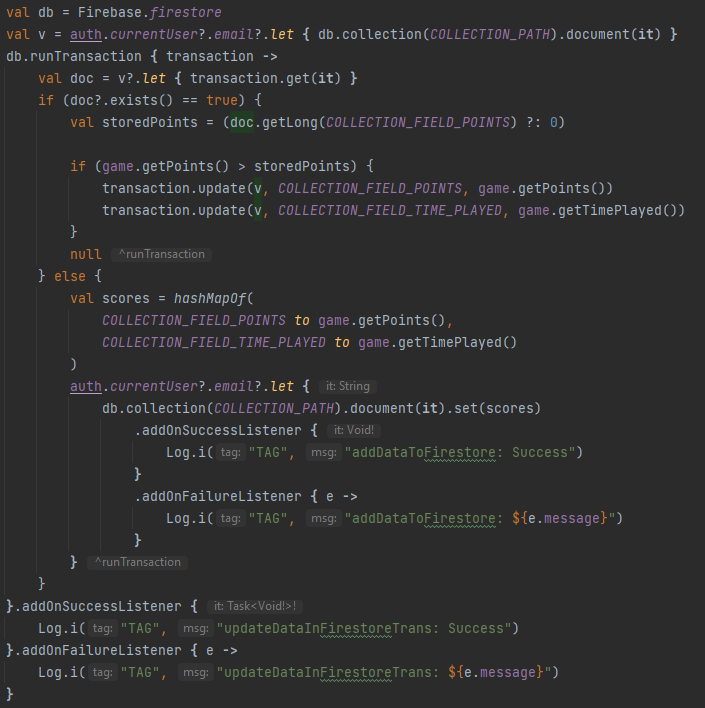


Figura - Criação ou atualização da pontuação na Firestore Database

# Idiomas

No decorrer do desenvolvimento da aplicação foi sempre tido em atenção a utilização de dois idiomas, nomeadamente o Português e o Inglês. Para isto foram criados dois ficheiros *xml* de *Strings*, que contêm a tradução para cada um destes idiomas das várias frases utilizadas na interface do programa.



Figura - Traduções

# Orientações ecrã

De forma a melhorar a apresentação visual da nossa aplicação em certas atividades foi pensada a possibilidade do utilizador poder ter o dispositivo em diferentes orientações. Como tal, foi criada uma interface especifica para o jogo para quando o utilizador tem o dispositivo em modo *Portrait* ou em modo *Landscape*.

Em certas atividades em que isto não é permitido foi bloqueada a orientação a uma especifica, neste caso à orientação Portrait. Estas atividades serão a de Login, Registo, Edição da conta de utilizador, Top 5 utilizadores com mais pontos e os Créditos da aplicação.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Orientação ecrã, Bloqueio orientação do Login

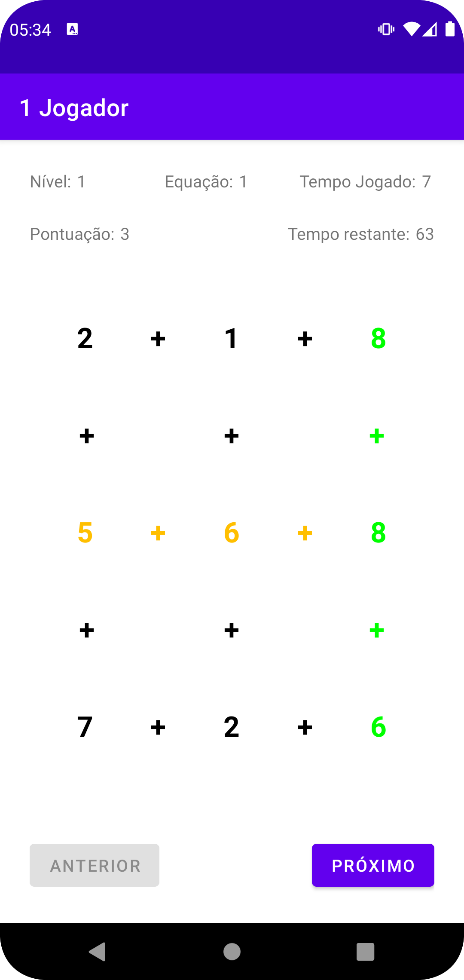


Figura - Orientação de ecrã, Modo Portrait



Figura - Orientação de ecrã, Modo Landscape