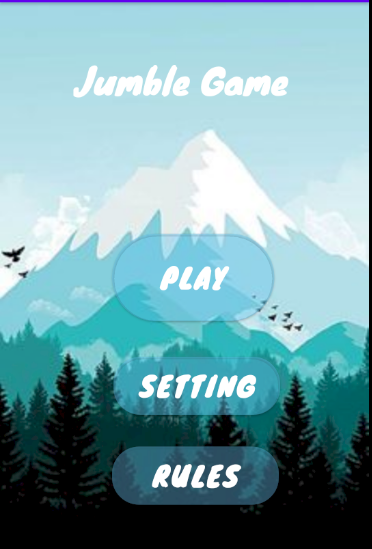
Manon JACQUET – 3064101

**User Interface**

Explanations:

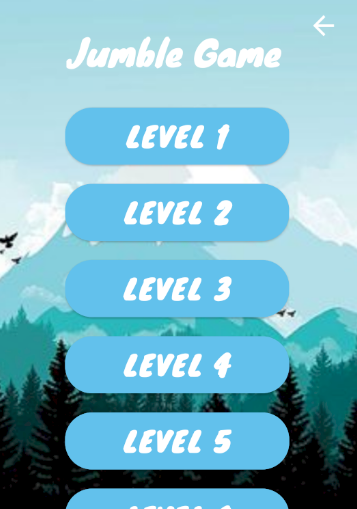
This app is a jumble game. The aim of the game is to guess a word from a string of mixed letters. There are six levels. To reach the next level ten words must be found. Each word of the level one contains three letters and there are ten tries per word. When the player reaches the next level, each word contains one more letter and there are 2 fewer attempts to guess the word.

Home Page/ Menu :

Here we can see the menu of the game : there are 3 buttons :

* The “Play” button starts the game ( and allows you to choose the level you want to play) ~~gives access to the different levels~~ (1)
* The “Setting” button gives access, through a new window that pops up, to the sound settings and the game credits thanks to a popup(2)
* The “Rules” button gives access to the game’s instructions, thanks to a popup (3)

1. The Play button :



On this page, you can choose the level you want to play. You can click on any level you want to have access to it and start the game. the higher the level is, the harder it is to find the words !

The arrow at the top right corner returns the player to the main menu

When one of these buttons is pushed, here is what occurs on the screen :



Number of the remaining attempts to find the word

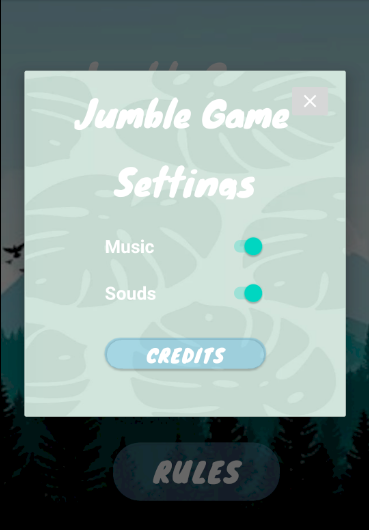
Mixed letters that composed the word to find

returns the player to the level page

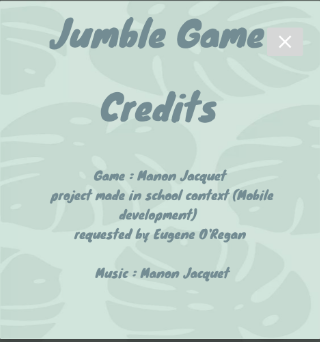
Place to write the guessed word

Button to validate the written word and check whether the answer is correct, or not

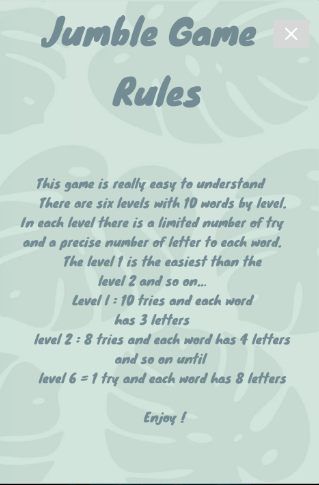
1. The “Setting” Button :



On this popup, you can switch on/off the music and the sounds of the game. you can access to this setting page only from the main menu.

Credits is just a popup to explain who made this app and the music.

1. The “Rules” Button :



Just a popup to explain the rules of the game.

**Documentation**

To begin, I thought about the content and the aspect the game should have. To be appreciated, a game must be attractive and easy to understand. Thus, to create this game I worked first on the programming and then, on the design.

Content of the app :

This app is composed by three activities :

* The main activity ( e.g. the central menu)
* The chapters (= the levels)
* Find

I created each of these activities and then I worked on “chapters “ and “find”

First of all, I created the different levels ( so the different *Strings[]* with the different words). At the beginning, they were located on the “find” activity. But doing that meant that I had to create six different “find” activities for each level array. To prevent this, I decided to put the six tabs in *private Strings[]*  in chapter activity and use the “*getters”* command to pick up them with an “*HasExtra “* command. Thus, after creating a scrolling view with my six buttons (each button has access to one level), I linked each button to the right array. Then, in my “find” activity, thanks to “*putExtra”,* I could pick up the array that I wanted.

The “Find” activity was the biggest part of my project. First, I had to create the function that generates the anagram. I created an array with each letter of the word and switched them at a random place. Then, I made the XML file with an *editText* and a *textView*. The text view is set with the anagram word in a function called *next*. This function gives the word to jumble and set the anagram to a string that is displayed by the *textView*. The “*OnCreate”* activity allows to set and check whether the player found, or not, the word.

*OnCreate* :

After I got the right tab, I made a list with the words in our arrays. To take the words in a different order each time a level is launched, I have to make a “random int”. Nevertheless with this method, a same word could appear many times. To avoid the same word to be selected several times for a same level, I wanted to remove them from the list once they are guessed by the player. Nevertheless, removing a word in an array is more complicated than removing a word in a list. That is why I first filled a list with the word array and then use all my function in this list. Then, clicking on the “*Validate” Button allows to*  check whether the user found the word, or not. If he does not find the word, the number of tries is decremented by one , the next word appears while the previous word is deleted from the list. If the number of tries is equal to 0 a popup appears and tells the user to play again.

The next step was to create the settings and the rules. The rules were indicated by a text in a popup. It was not complicated to made. Finally, the last thing that I had to manage was the music. During the game, there is some music in the background. This music can be stopped or played from the “settings” popup. One of my problem was to find a way to play the music continuously even if the activity is changed. To achieve that, I decide to never stop the mainActivity. Thanks to this method, the music is played in the background and it can only be stopped from the settings. For the sound button, I imported a sound find on YouTube and I have created an “int” that is set to 1 if the switch button is set to “true” otherwise, the int is set to 0 when it is false. Tab is int, thanks to *HasExtra* and *putExtra,* is connected with every activity and allows to turn the sound on or off. .

Sometimes, it can happen that the music sputters but it is not always the case, and I do not really understand why. This music was composed by me in France and I have register it. I thought that this sound fits very well with this puzzle game.

Finally, I work on the design of my app. I wanted a game with a relaxing look, something in shades of blue. I took a background from Pinterest. Then, I selected the font of my text and the design of the button. At the beginning, I created a special XML file, but, for a reason that I do not understand, the background was not displayed. Hence, I changed all the design in my XML activity file. The final appearance is close to the one of an application that could be very well be found on Appstore for instance.

**References**

**Tutorial Video:**

# Develop simple Anagram game in Android Studio : <https://www.youtube.com/watch?v=qo6cdULtWik>

# How to play sound with button click Android Studio : <https://www.youtube.com/watch?v=9oj4f8721LM>

# How to change Button Background on Clicked or Pressed | Android Custom Button Tutorial 2020 : <https://www.youtube.com/watch?v=BmeqkM4nMbE>

# Android Studio - Pop Up Window : <https://www.youtube.com/watch?v=4GYKOzgQDWI&t=273s>

# Creating Scroll View - Android Studio latest version : <https://www.youtube.com/watch?v=U0da4JgJwHI&t=342s>

# Android Switch Button Example : <https://www.youtube.com/watch?v=ZcWN-d3tTT4>

# Other :

# List : <https://www.journaldev.com/31869/java-list-remove-methods-arraylist-remove>