

Development Documentation

Project State

The project is ready and functional.

Code Coverage

Add prints :D

BetterCodeHub Code Quality metrics

Add prints and link :D

Details of Implementation

Throughout the project, we have tried to implement state machines since it is a way to make code more readable and structured. So the first state machine is part of the screens structure. From this we can control the various transitions between menus and the game. The other state machines are in the sprites states.

The project was developed by layers to create one hierarchy structure and modular.

Input and output of project

The project has an output file that creates a binary file where store de available levels unlocked by player. So every time that player play the game, he can jump to another level that he reached before.

This functionality it's not available in android version.

Tools and Frameworks

The project was developed in android studio(using gradle) and the server in Inttelij(using maven).

The main project it's based on libgdx framework. (<http://libgdx.badlogicgames.com/>).

The server it's based on netty. (<https://netty.io/>).

The box2d Was used in order to implement a complex physics model in the game and to forget about many problems related to the physics.

The AI was developed by us and we do not use it in the framework.

Other Relevant Design Decisions

We need to decide if the MVC was really necessary and easy to adapt because we have already starting the project and we concluded that it's not necessary but a barrier in implementation.

Major difficulties along the way

We need to learn about the libgdx framework by reading the API and seeing some tutorials. And even then we had some problems due to the complexity of some modules.

Creating the server was also a problem that took a lot of time to solve, several frameworks were seen to figure out which "best" to use. And even after that there was a lot of time in the configuration part and in realizing how the part of the game synchronization would be processed.

Another problem we had was adapting the server to create the multiplayer. The main idea was to create all the game logic on the server but due to the use of libgdx this was not possible because it was necessary for the server to know the classes of libgdx and thus not having a large level of abstraction. So, multiplayer is based only on receiving input from the user and sending it to the other player. Keeping only a repetition that everything that is done in one client is performed in the other, through the server.

The implementation of unit tests due to the need to create some libGdx-dependent things.

Lessons Learned

In short, the development of this project allowed the group to apply and improve the knowledge about the skills acquired during the semester as well as acquire new skills in unknown technologies.

We learned to work with a new framework and took the first steps towards networking (which we had never done) and AI.

Overall time spent developing

Working distribution amongst team members

The development of the project throughout the semester was equally distributed by the two elements of the group, so the contribution of each element will be equitable.