Development Documentation

Project State

The project is ready and functional.

Code Coverage

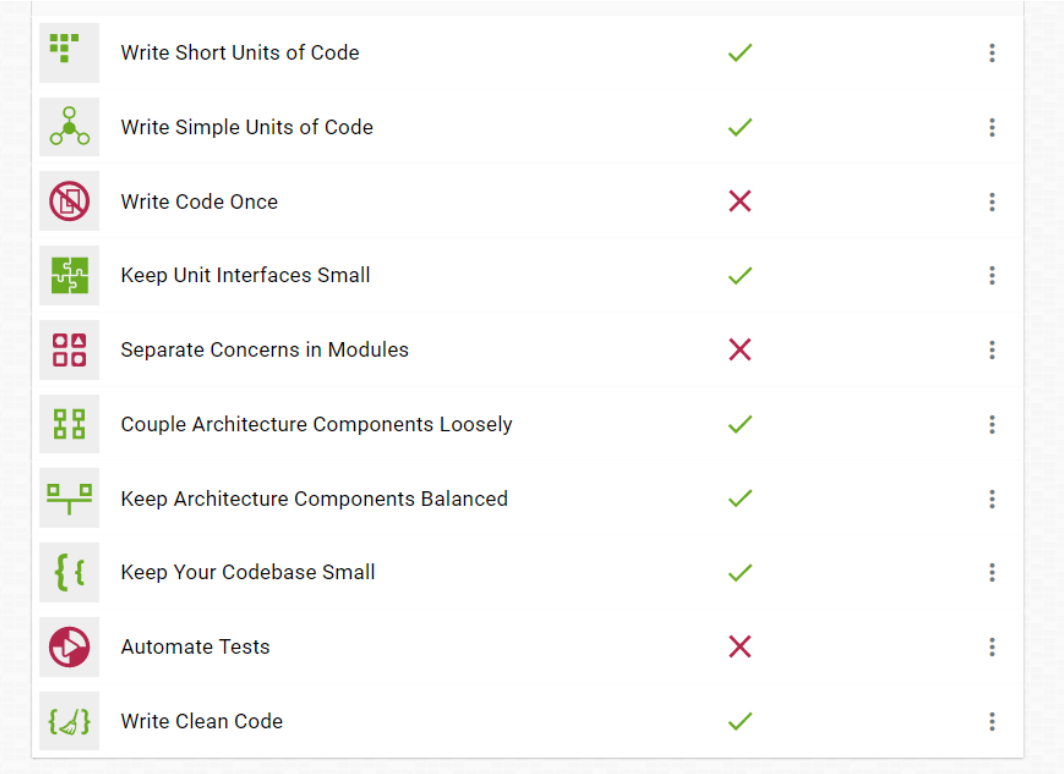
Element	Class, %	Method, %	Line, %
Logic	99% (102/103)	92% (509/551)	91% (2370/2585)
Managers	0% (0/3)	0% (0/11)	0% (0/118)
net	0% (0/16)	0% (0/47)	0% (0/170)
Scenes	0% (0/5)	0% (0/22)	0% (0/169)
Screens	0% (0/18)	0% (0/238)	0% (0/1164)
Tools	33% (4/12)	25% (24/95)	32% (213/655)
Bombic	0% (0/1)	0% (0/4)	0% (0/24)
Bombic	0% (0/1)	0% (0/4)	0% (0/24)

99% classes, 91% lines covered in package 'Logic'

Element	Class, %	Method, %	Line, %
EnemiesStrategy	100% (17/17)	100% (60/60)	96% (393/408)
GameLogic	96% (27/28)	78% (100/127)	72% (307/422)
Sprites	100% (58/58)	95% (349/364)	95% (1670/1755)

BetterCodeHub Code Quality metrics

https://bettercodehub.com/edge/badge/diogotorres97/LPOO1617_T1G2?branch=finalRelease&token=0904237974da39ad53e22b89cdbb075ec70d08f6



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.

AI Types

Type 1 – The most basic AI, moves randomly only avoiding going to places where are objects (walls, barrels, boxes, …).

Type 2 – This AI moves randomly, avoids objects and doesn't go to flames.

Type 3 – This AI moves randomly, avoids objects, doesn't go to flames nor ticking tiles (tiles where will be flames within a few seconds).

Type 4 – The most dangerous AI, besides moving like AI 3, also persecutes players that are within a range of 3 tiles from it.

Other Relevant Design Decisions

We had to decide whether to use or not MVC, and evaluating the pros and cons of its implementation in our project we opted for not using it.

Major difficulties along the way

We need to learn about de 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.

The definition of each AI strategy of movement as well as the variations of them led to a few tries before the final implementation of 4 different types of AI.

Dealing with multiple bugs and solving them.

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

Have been spent in average 190h by both team members.

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.