Underboss

Project by João Álvaro Ferreira, [up201605592@fe.up.pt](mailto:up201605592@fe.up.pt) – MIEIC FEUP

**UML Diagram**

**Design Patterns**

* **MVC: The core source code of the project is divided into three essential parts: the Model, which establishes the rules for each object and the interactions between classes, the View, which draws on the screen according to what the project demands, and the Controller, that manages all of the elements of these, runs the logical loop and the user’s inputs.**
* **Singleton**: Singleton was used to ensure a class has only one static instance,and provide a global point to access it. In this case GameManager, MyInputProcessor and Underboss will only have one instance which can be easily accessed by each other and by the screens.
* **Decorator: The Character class was heavily built upon so that the Boss and Minion class could all extend it without having to re-do any of Character’s methods.**

**Difficulties**

At the start of the development of this project, my partner (Álvaro Francisco Miranda) did not meet the minimum requirements for the other evaluations, and thus decided to quit the LPOO chair this year. I (João Álvaro Ferreira) was then left to do the final project entirely on my own, which meant a full week (from last Tuesday to Sunday, June the 3rd) of working at least four/five hours every day, plus the last two days in which this project took all of my time. I was also building on top of a demo I had developed for the intermediate checkpoint delivery about a month ago, which gave me a bit of a head start.

This means that the already heavy workload for this final project doubled, and it constrained both the quality of the project and the extent to which I could fulfil my personal goals for it (which I’ll talk about in the “*Design Goals”* section).

Despite the obvious handicap, I can’t say I really used my time as well as I wished. The project was plagued with issues with Android Studio, Gradle, Libgdx and my JDK installation, which left me without being able to open the project with the Desktop Launcher until the last day. I spent nearly as much time debugging and looking for solutions for random errors that occurred due to the botched installations/imports into IDEs as I did coding, which is a big problem if you’re already short on time.

Thankfully, I managed to fix all of these issues by the last day and produce all that was asked, but the further time loss only hampered my project further.

Another issue I faced was following some Libgdx guides way too close to the books, not letting me learn enough about the libraries to really take advantage of them while building the base of my project. By the time I had known about all of positives of using libraries like Body or Stage, I had already developed too much of the project to go back and do it all over, which took a lot of options away from me.

The most important lessons I learned from this project were to always make sure everything is cleanly installed before starting to work, to read up more about the resources I’ll be using before starting to build with them and to be more flexible with my code, enabling future changes if needed.

**Design Goals**

At first, I had planned a 1v1 shoot ’em up in the spirit of a Binding of Isaac boss fight, an inspiration which can be clearly seen in some of the images I use. I had idealized multiple rooms with multiple bosses, all using similar frameworks but with minor tweaks to their behaviour and look that would make them feel different, possibly with an overworld without fights linking them. These were the secondary objectives I was hoping to meet:

* Networking – for 1v1s between players;
* Social – using Facebook Messenger to challenge people to try the app against whoever was giving the challenge;
* Artificial Intelligence – a dynamic boss with actions dependent on the current state of the fight and the behaviour of the player.
* Android – have this game on mobile and use the gyroscope, accelerometer, touchscreen and other relevant frameworks to control the player character and give feedback to the user.

I ended up only meeting the last two goals – and despite doing a good job at them, in my view, I still wish I could’ve done more with the ideas and resources I had. That said, I’m still pleased with what I managed to do, as I feel like the content that is there is polished and works well, leading to a decent fast-paced game.

**User Manual**

## **How to run the development environment**

* Set up Android Studio with an Android Emulator;
* Import the project into Android Studio;
* Unload the HTML module (you can do this by right clicking on the project > Load/Unload Modules);
* Go to Run>Edit Configurations and make sure that the working directory of the DesktopLauncher is “…\Underboss\android\assets” (the first part of the url being the place where you pull the project to);
* With all of this done, you should be able to run the project with both Android and the DesktopLauncher run options.

## **How to run the testing units**

* Open each class individually, hover your pointer over the name of each class and press “Alt + Enter”, which will prompt you to run a test of that class;
* Alternatively, go to Run>Edit Configurations and Add Configuration (the plus sign in the corner) and make sure the Test Kind is class, “Use classpath of module” has core selected, and the classes are “com.underboss.Test.” and then each class one by one.
* The necessity to run every class on its own instead of all of them at the same time is due to a bug where testing units don’t seem to recognize the Libgdx library when running every class at the same time, but do recognize it when running every class on its own. This means that some of the logic functions I use, which require functions like *Gdx.graphics.deltaTime().* The program will still compile, but these functions won’t output the correct results.

## **How to run the Android (.apk) version of Underboss**

* Open “UnderbossAndroid.apk”, located in “Executables” in an Android Emulator or in an Android phone. This will prompt you to install the game.
* After the game is installed, just tap on its icon to run

## **How to run the Desktop (.jar) version of Underboss**

* Open “UnderbossPC.jar”, located in “Executables” with a double click.
* It should open, but if it does not, the command “javaw.exe -jar UnderbossPC.jar” while in the Executables file should be enough.

**Playing the game**

**Intro screen**

* As soon as you open the game, you’ll be greeted by an intro screen.
* Tap or click to go forward.

**Tutorial screen**

* This screen gives you the instructions on how to play the game.
* Tap or click to go forward.

**In-Game**

* Two indicators to show boss and player health, along with status.
* The player can shoot his laser beam and walk in four directions, as was indicated in the tutorial screen.
* The boss has multiple types of attacks: he can summon minions to chase the player, throw energy blasts that will aim at the player’s position at the time that it was fired, and will attack the player with a lightsaber if the player gets too close (or if he gets too close to the player).
* The minions will follow the player and give 2HP damage on touch, but they can be killed if hit by one of the player’s laser beams.
* After the boss’s health declines to less than 1/3, it will start to shoot faster blasts and its projectiles will poison the player on touch, giving the player damage over time until either his next hit or until the player goes down to below 1/3 health himself.
* Game ends when either the boss or the player dies.

**Ending Screen**

* Depending on who dies first, the player might get a Victory or Defeat screen.
* Either way, the player can tap again to go back to the intro screen and start all over again.