

# COMMANDO

Timofti Gabriel

**Email:** [gabitim470@gmail.com](mailto:gabitim470@gmail.com)

Faculty of Automatic Control and Computer Engineering, Iasi

**Academic year:** 2018-2022



## Game Description

Commando is a top-down game where your goal is to pass all the levels and gather as much damage points as possible.

You can shoot bullets and use your special ability when near enemies.

Everyone will try to stop you but if you are careful and brave you can reach your goal.

To pass the levels you need to find the portal and stay inside teleportation area for up to 5 second.

When you finish all the levels you will be mentioned in the LeaderBoard.

Are you up to the challenge?

## Game Mechanics

This game is a collision based game. There are 3 types of collision:

1. With water and other solid object. You can't pass.
2. There are places where you will fall and die instantly
3. You need to stay in the teleportation area for up to 5 second
4. Bullets will collide with enemy and produce damage
5. Your ability to produce a lot of damage will collide with the enemy only in front of you.

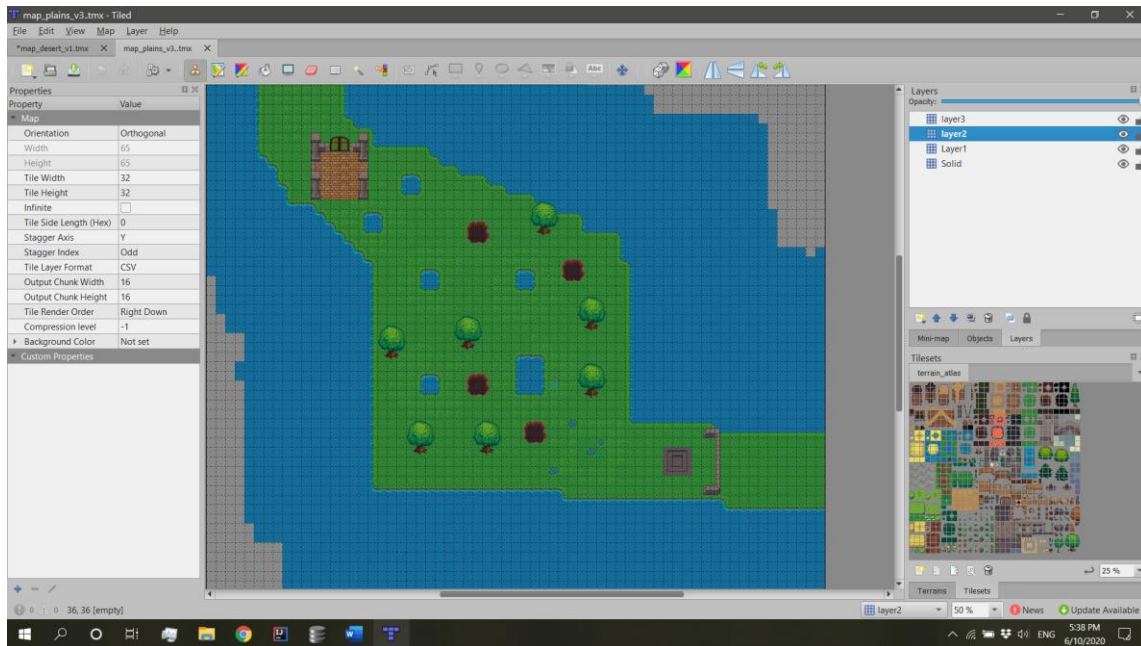
**Other features include:**

- Life system, You start with 4 lifes and lose when have 0 left
- The map is larger than screen size and a camera is implemented to move around
- Enemies will respawn everytime you will kill them
- Enemies will use a simple AI to follow you everywhere
- There are 3 levels of difficulty, with each one the number of mobs increases
- There are 5 types of enemies, each one is different
- Ability to save and load form any point in game from a database
- Ability to save your score and print the highest scores.
- Menu State, Pause State implemented
- You can change between 2 types of maps, and 2 types of character

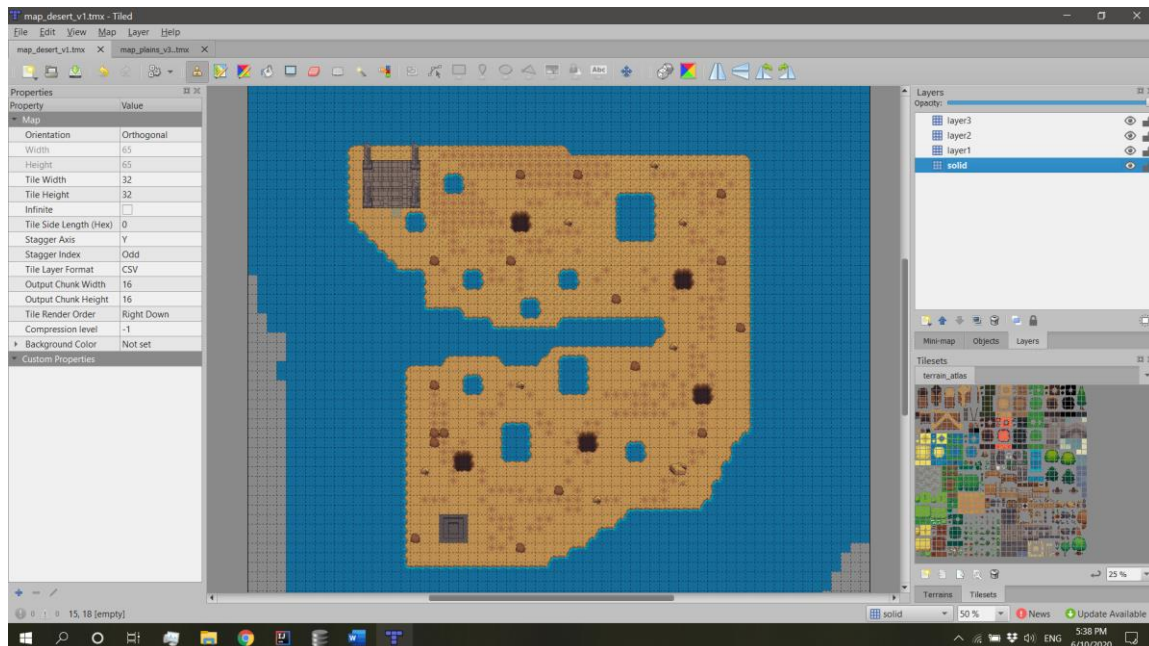
## Map creation

I used Tiled software to create a map based of blocks. Tiled generates a xml file; In game i parse this file and reconstruct the map

Every map has 4 layers. First one is the solid layer where all the collisions happen. The other 3 layers are for decoration and aspect

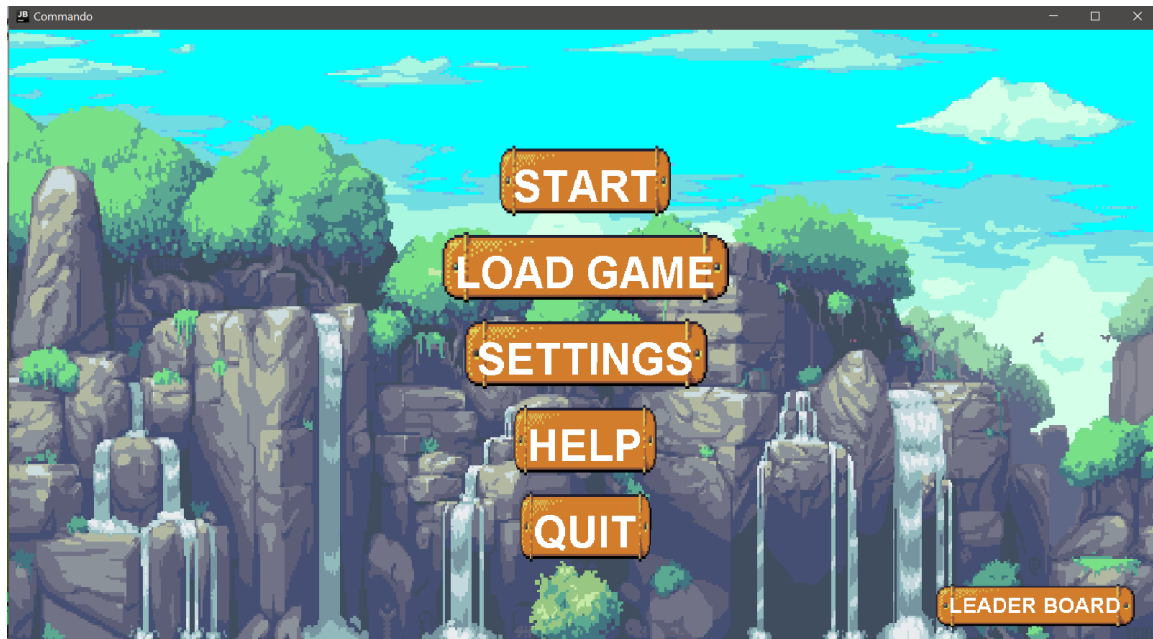


Map plains and Map Desert in **Tiled**.

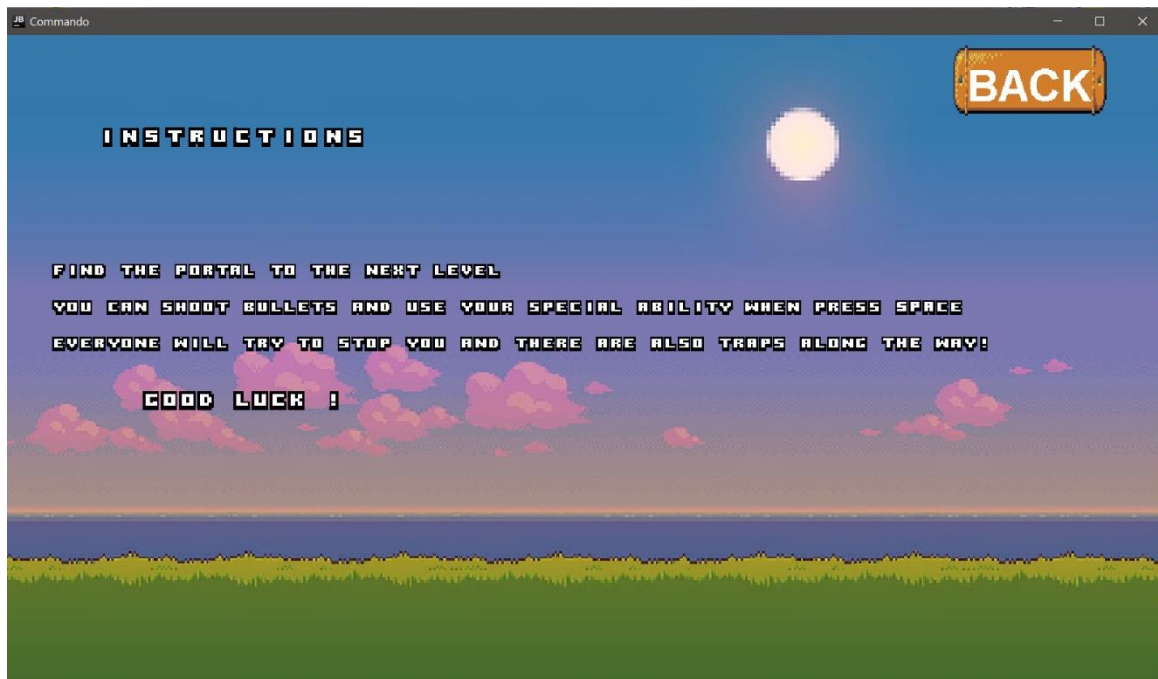


# Screen Shots from the game. UI Components

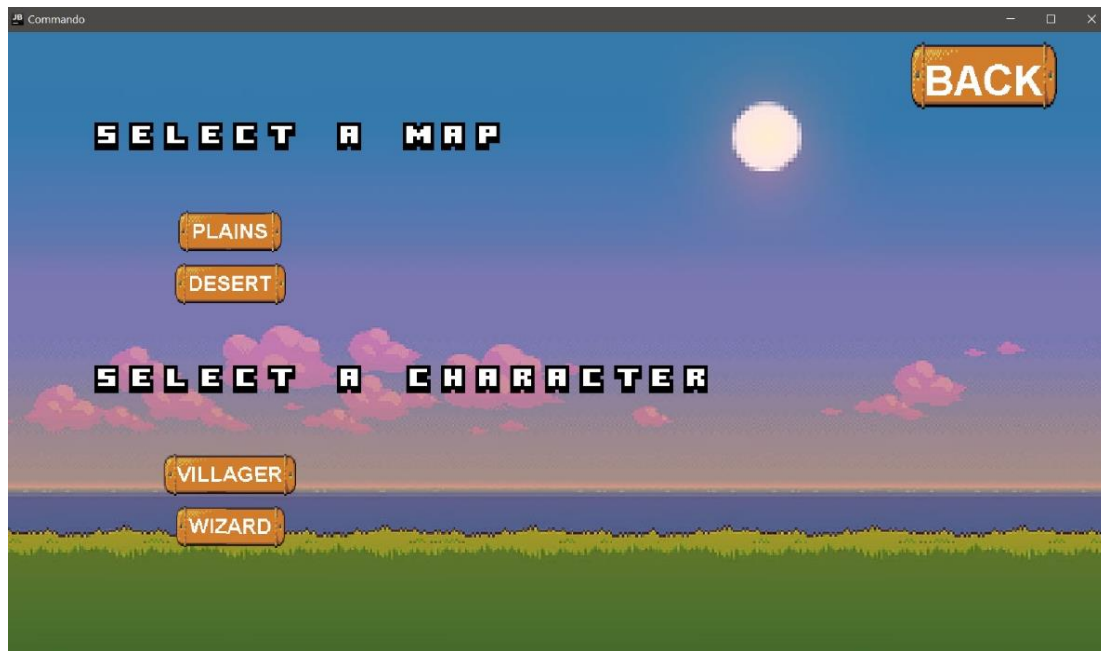
## Menu State



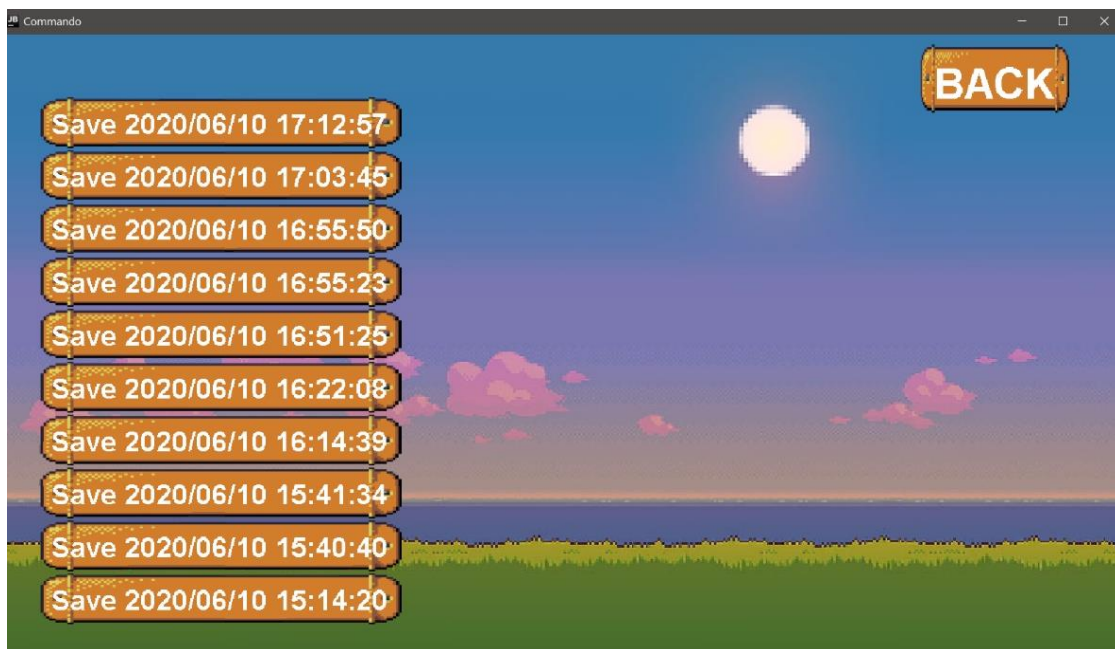
## Help State



## Settings State



## Load Game State



## Pause State



## Play State





## Enemies sprites

- Dwarf



- Goblin



- Orc



- Girl



- **Skeleton**



The enemies change sprite level when moving in the corresponding direction

## Hero Sprites

- **Villager**



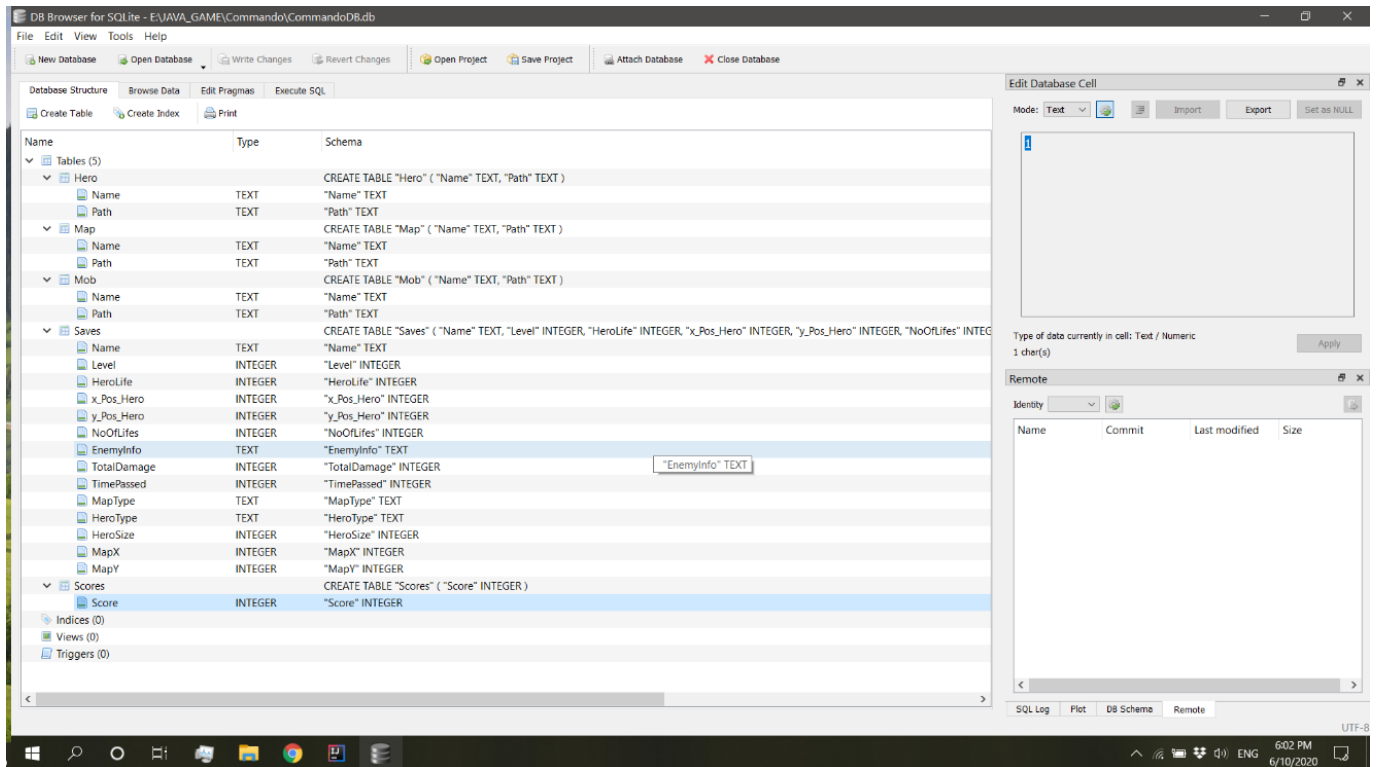
- **Wizard**



Hero also has an animation for death.



# Database

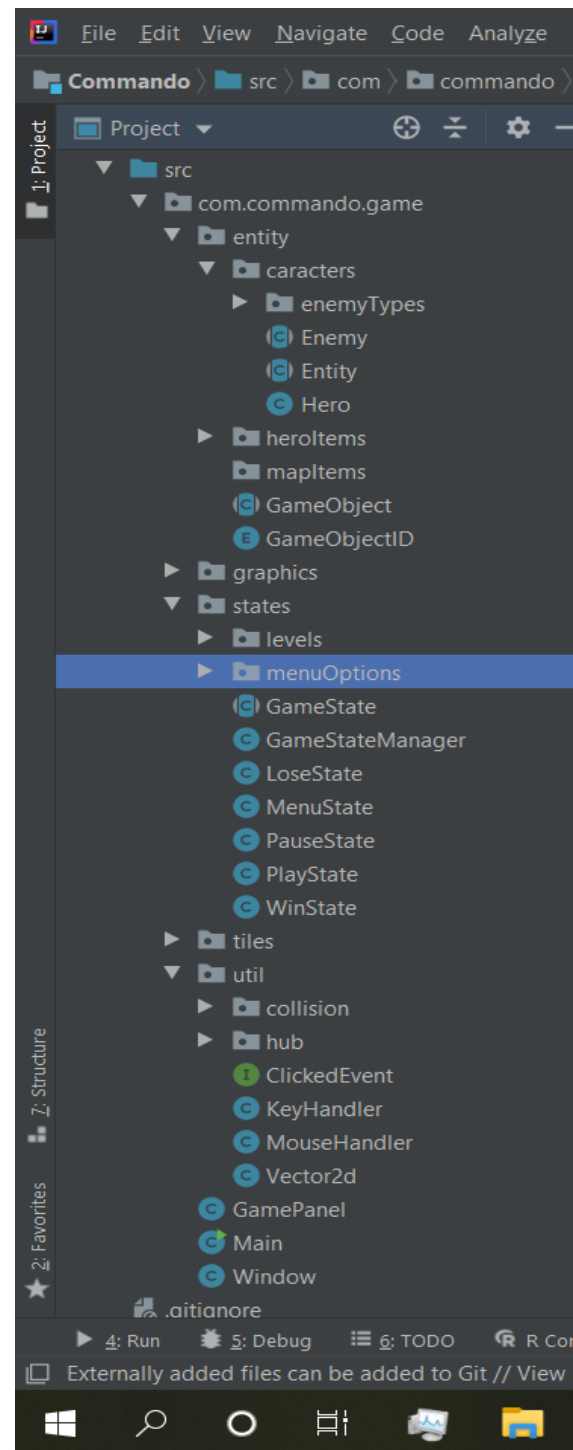


From the database the game loads the paths to the files.

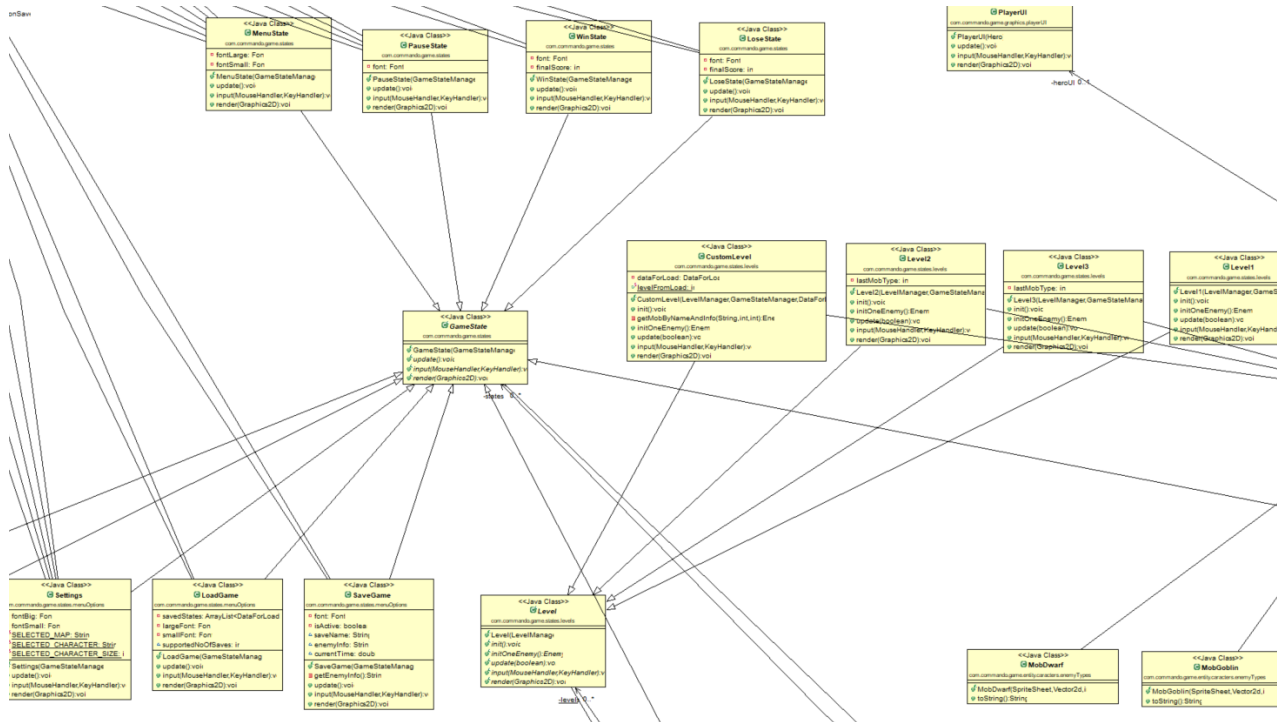
Also the scores and all the saves. In Save state we write in the database and in Load state we read from the database.

# Game Arhitecure

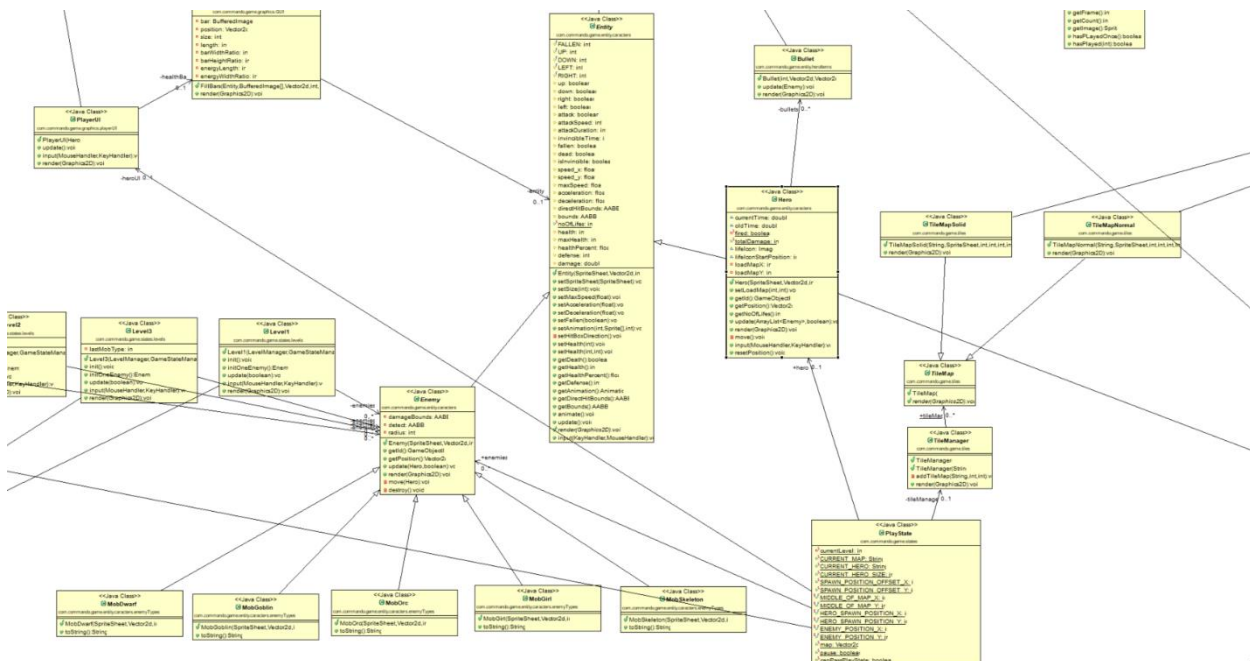
All the packages from the game in  
intellij IDE.



- State and Level System

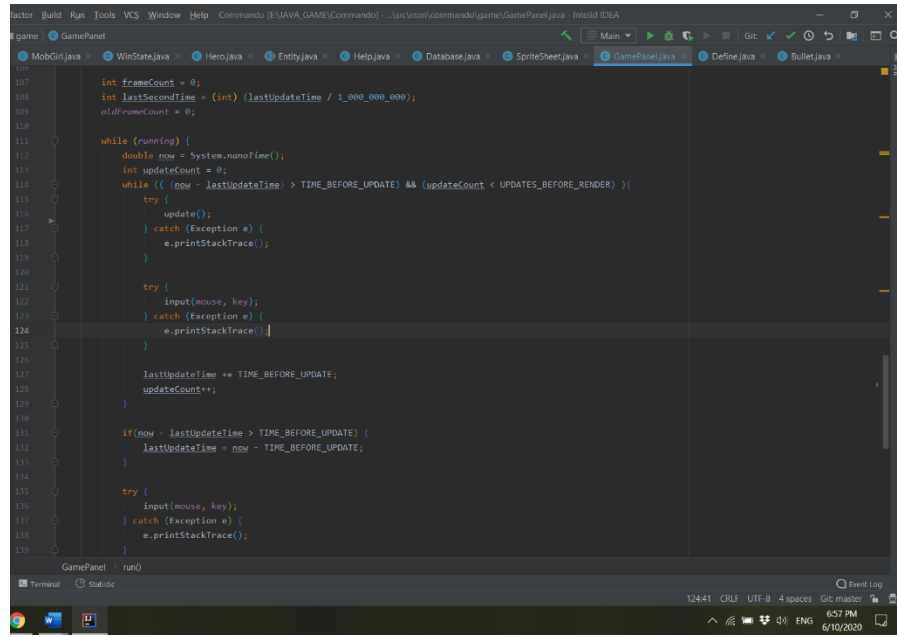


- Entity and Component System, Also PlayState



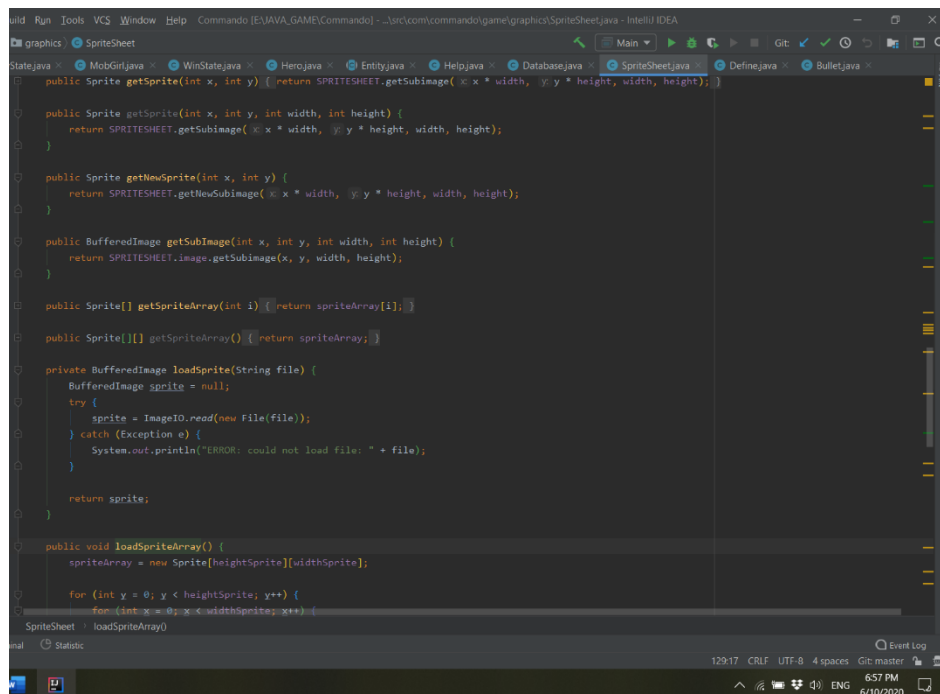
# Error Handling

- Code example from GamePanel



```
107     int frameCount = 0;
108     int lastUpdateTime = (int) (lastUpdateTime / 1_000_000_000);
109     oldFrameCount = 0;
110
111     while (running) {
112         double now = System.nanoTime();
113         int updateCount = 0;
114         while ((now - lastUpdateTime) > TIME_BEFORE_UPDATE && (updateCount < UPDATES_BEFORE_RENDER)) {
115             try {
116                 update();
117             } catch (Exception e) {
118                 e.printStackTrace();
119             }
120
121             try {
122                 input(mouse, key);
123             } catch (Exception e) {
124                 e.printStackTrace();
125             }
126
127             lastUpdateTime += TIME_BEFORE_UPDATE;
128             updateCount++;
129         }
130
131         if (now - lastUpdateTime > TIME_BEFORE_UPDATE) {
132             lastUpdateTime = now - TIME_BEFORE_UPDATE;
133         }
134
135         try {
136             input(mouse, key);
137         } catch (Exception e) {
138             e.printStackTrace();
139         }
140     }
141 }
```

- Code example from Sprite Sheet Class. We track the `NullPointerException` and print a message to console



```
public Sprite getSprite(int x, int y, int width, int height) {
    return SPRITESHEET.getSubImage(x * width, y * height, width, height);
}

public Sprite getNewSprite(int x, int y) {
    return SPRITESHEET.getNewSubImage(x * width, y * height, width, height);
}

public BufferedImage getSubImage(int x, int y, int width, int height) {
    return SPRITESHEET.image.getSubImage(x, y, width, height);
}

public Sprite[] getSpriteArray(int i) { return spriteArray[i]; }
public Sprite[] getSpriteArray() { return spriteArray; }

private BufferedImage loadSprite(String file) {
    BufferedImage sprite = null;
    try {
        sprite = ImageIO.read(new File(file));
    } catch (Exception e) {
        System.out.println("ERROR: could not load file: " + file);
    }

    return sprite;
}

public void loadSpriteArray() {
    spriteArray = new Sprite[heightSprite][widthSprite];
    for (int y = 0; y < heightSprite; y++) {
        for (int x = 0; x < widthSprite; x++) {
            spriteArray[y][x] = loadSprite(file);
        }
    }
}
```

**Source Code:** [https://github.com/gabitim/Commando\\_2D\\_Tile\\_Game](https://github.com/gabitim/Commando_2D_Tile_Game)