CSE443 - Object-Orianted Analysis & Desing - HW 1

Fatih Kaan Salgır - 171044009

Design Explanation

Java Swing Platform Game

`Main` class creates a `JFrame` and sets its content pane to new `GamePanel`. `Gamepanel` extends `Jpanel` and implements;

- `Runnable`: for creting game loop.
- `KeyListener`: for listening keys press and relases. `HashMap<Integer, Boolean> keyPressed` with `registerKeyEvent` is used to keep track of keys.

Whenever the keys are pressed the value of the corresponding key in the hash map set to `true`.

- `MouseListener`: for listening clicks on the pane to check whether the user clicked the pause button.

Game loop targets the 60 FPS. Checks the collisions, earned points, power-up acqusitions, key presses, and update the state accordingly. Updates the positions of the each view and HUD and draws into `BufferedImage`. If still there is time before drawing the next frame, the thread sleeps for a number of calculated miliseconds to target 60 FPS.

Lower part of the window is left for debug messages. `JList` with `DefaultListModel<String>` is used on top of `ScorllPane`, so whenever a log is added to the `DefaultListModel`, it is displayed in the `JList`.

`common/Util.java` stores the constants used different classes like the width & height of the window, position & sizes of objects, file paths etc.

Decorator Pattern

Decorater pattern demonstration in game loop;

In `GamePanel.java`:

```
private Multiplier multiplier;
// ...
this.multiplier = new BaseMultiplier();
// ...

// on successfull jump
score += multiplier.unitPoints();

// when new power-up acquired
switch (p.type) {
    case A:
        multiplier = new PowerUpA(multiplier);
        // ...
    break;
    case B:
        multiplier = new PowerUpB(multiplier);
        // ...
    break;
    case C:
        multiplier = new PowerUpC(multiplier);
        // ...
    break;
    // ...

}
```

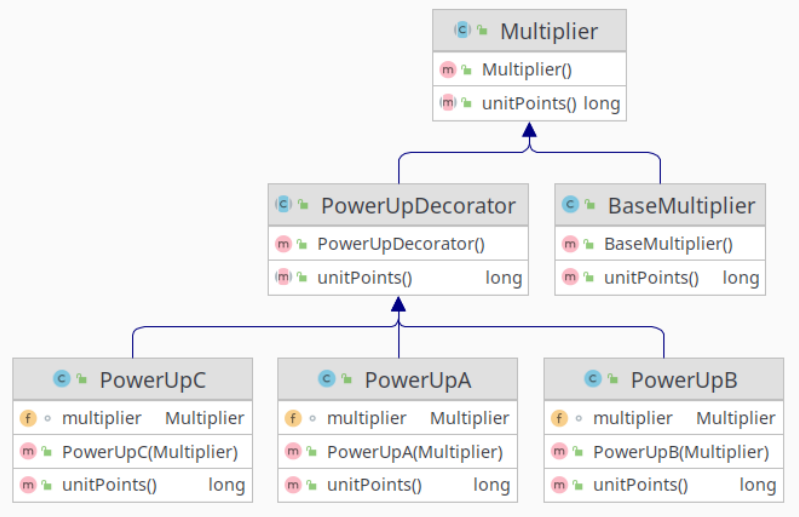


Figure 1: Desing of the Power-up Modal Classes

Strategy Pattern

In game loop there are three methods need to be used to manage any jump behaviour. These are `jump()`, `updatePosition()` and `isJumping()` which is defined as abstract methods in `JumpBehaviour`. Therefore any jump, must implement these functions. `Actor` is an abstract class which only consists of methdos that calls *behaviour* methods and getter/setter for the jump behaviour. Whenever the `Actor` methods are called, methods of the `JumpBehaviour` will be called.

`JumpPhysic` is only a helper class to calculate character's next position according to elapsed time simulating gravitational acceleration. Whenever the jump mode changes, also the initial velocity changes. Depending on the inital velocity, time in the air also changes. Since the power-up D is acquired when jumping, the behaviour doesn't change immediately but prempted till the jump is over.

Dynamically changing the jump type:

In `GameCharacter.java`;

```
// ...
public void toggleJumpMode() {
  if (getJumpBehaviour() instanceof JumpLow) {
    nextJumpBehavior = new JumpHigh(this.characterView);
    characterView.setImage(Util.CHARACTER_WITH_WINGS_FILENAME);
  } else {
    nextJumpBehavior = new JumpLow(this.characterView);
    characterView.setImage(Util.CHARACTER_FILENAME);
  }
}

@Override
public void updatePosition(long start) {
  jumpBehaviour.updatePosition(start);
  if (nextJumpBehavior != null && !isJumping()) {
    setJumpBehaviour(nextJumpBehavior);
  }
}

// ...
```

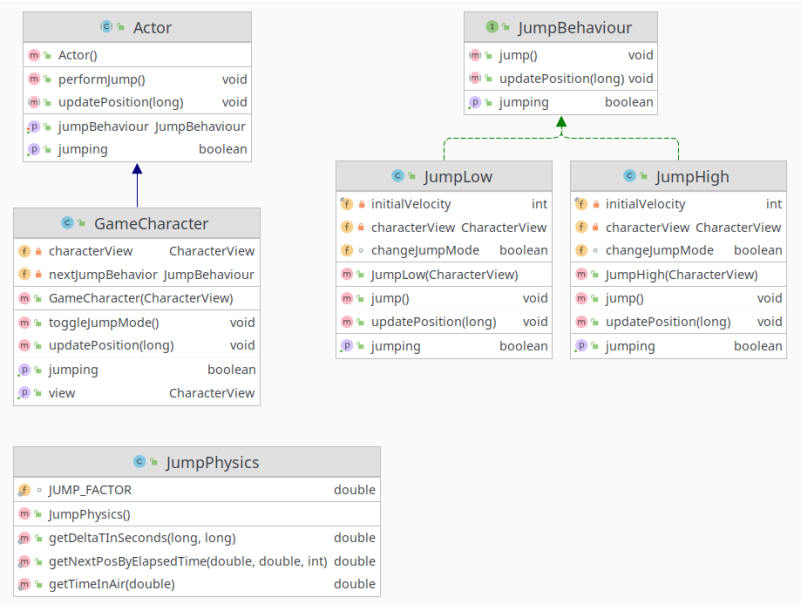
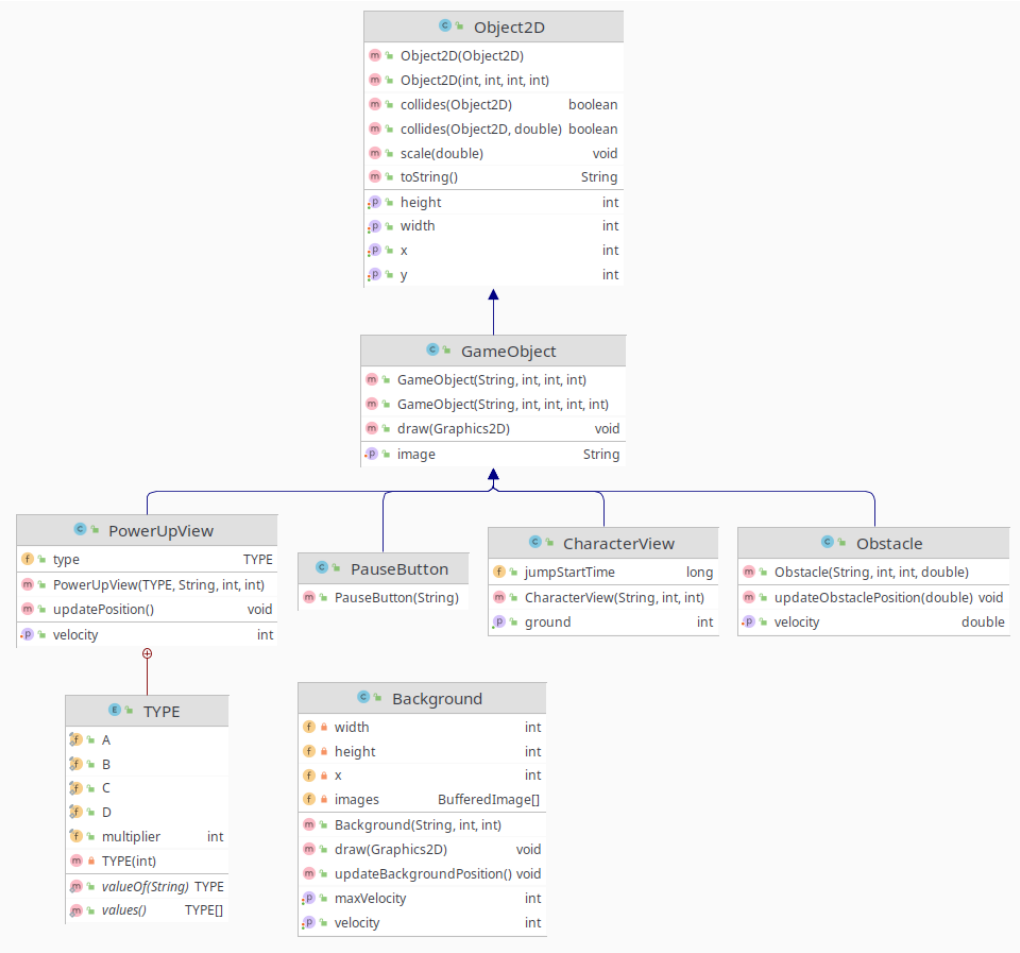


Figure 2: Desing of the JumpBaheviour & Actor Classes

Views

Everything that is drawed in the pane is collected into `View` package.



Play / Pause

When the game is paused the only state should be saved is the position of the main character. The only parameter which sets the character position is the current time and start time of the jump. So start time of the jump is saved when the game paused and restored as **current time − time in the air when paused**.

Notes

- In order to get better game experince; collision detection thershold with obstacles is decreased, and thershold with power-ups is increased.

General Class Diagram