Super Mario

I created a custom version of old famous game Super Mario. It is a 2D platform game where the player's goal is to jump through all the worlds, collecting useful items, gaining score and eliminating enemies.

User guide:

Mario is the protagonist of the game, controllable by the player. The movement is controlled by left arrow, right arrow and space: to jump. The goal of the game is to pass all worlds while collecting as much score as possible. Score can be collected either by eliminating enemies or by collecting special items. When all the worlds are passed, Congratulations message comes up and game starts from the beginning.

To eliminate an enemy, Mario must jump on it, while automatically bouncing a little after doing so and getting 100 score points. Any other contact with the enemy as welll as falling into a hole makes Mario to loose a life. After loosing a life, Mario resets his position on the begininng of the current world. When Mario reaches zero lives, Game Over message comes up and the game starts from the beginning. There are two kinds of enemies: flying birds and ground goombas.

There are two items: flower and coin. Both can be found simply on the ground or in the boxes. To open a box, hit it from the bottom by jumping into it. Coin adds 100 score points. Flower adds 50 score points. Additionally, it makes Mario temporarily invincible, turning him grey. If Mario hits an enemy while being invincible, he looses invincibility intead of dying while eliminating the enemy in the process.

Maps:

It is possible to create custom maps in a simple way by adding new text representation of the level. The guide how to create a world is placed in readme document inside levels directory.

Code documentation:

All classes and methods are properly documented in the attached JavaDocs.

The game is based on deriving from Thread, JPanel and JFrame.

Important classes:

Game class

Inherits from Thread. Runs the application in a loop, until closed. Calls the necessary functions in one tick and then sleeps the thread. It takes care of the following functions:

- checks for Mario being out of bounds, effectively starting a new world
- checks for Mario dying, reinitializing the game
- manages the keyboard controls, issuing new commands for Mario
- moves the enemies
- check for collisions

GameFrame class

Inherits from JFrame. Takes care of the layout of the game, having the panel inside of it.

GamePanel class

Inherits from JPanel. It is the actual place where the game is being drawn using Graphics. It is placed inside GameFrame. The elements of the game, such as label, Map grid or Mario are put in it and being drawn.

Map class

The grid of the game. Whole game is 20*10 grid of 50 pixel blocks. This grid parses the text representation of the world and draws accordingly. Also adds enemies. The collision detection makes use of the grid.

Mario class

The character class, ensures movements and collision detection. Can detect when to stop jumping (hit the terrain), stop falling (felt terrain), when to collect an item, when to bounce from enemy and when to die.

Block class

The grid is made of Blocks. Block is a common class for all elements of the game except for Mario. Classes Enemy, Collectible and Box are derived from it. Other types of Blocks (wall, tunnel, floor) are not derived as they differ only in the image name and serve the same purpose. Enemies are further subdivided into Bird and Goomba.