HDWJ3H

Dana Saker

Assignment 3 – Documentation

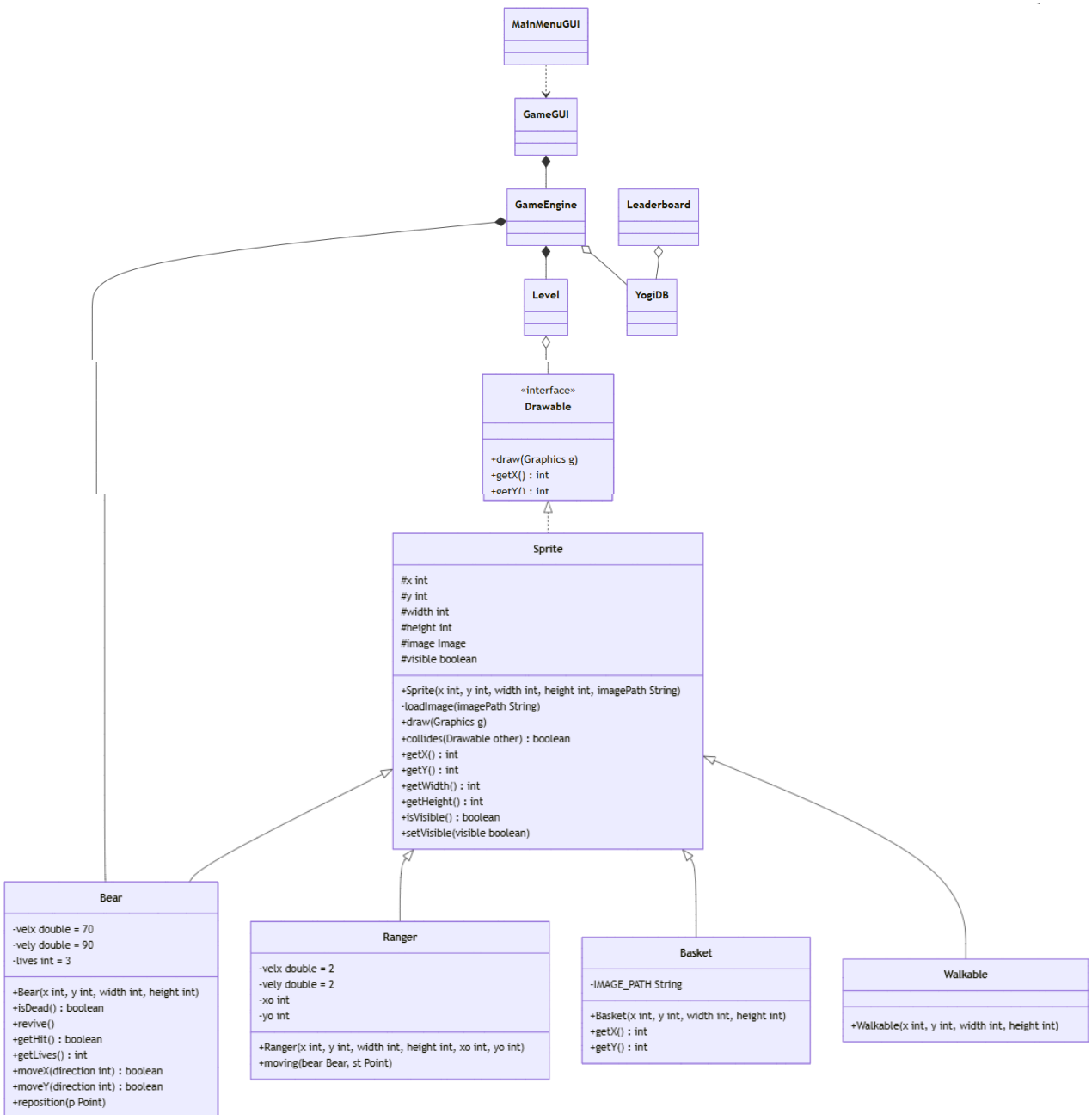
YogiBear

Task 1 :

Yogi Bear wants to collect all the picnic baskets in the forest of the Yellowstone National Park. This park contains mountains and trees, that are obstacles for Yogi. Besides the obstacles, there are rangers, who make it harder for Yogi to collect the baskets. Rangers can move only horizontally or vertically in the park. If a ranger gets too close (one unit distance) to Yogi, then Yogi loses one life. (It is up to you to define the unit, but it should be at least that wide, as the sprite of Yogi.) If Yogi still has at least one life from the original three, then he spawns at the entrance of the park.

During the adventures of Yogi, the game counts the number of picnic baskets, that Yogi collected. If all the baskets are collected, then load a new game level, or generate one. If Yogi loses all his lives, then show a popup messagebox, where the player can type his name and save it to the database. Create a menu item, which displays a highscore table of the players for the 10 best scores. Also, create a menu item which restarts the game.

Class Diagram :



Task Analysis:

The game starts through the MainMenu where the user must input a name to proceed. Each level is define once per game(pre player) and it is updated while thought the level text files that represent multiple levels of the game. Th game engine defines all the keyboard input and assigns tasks to them. Based on the users selection the game is finished once they have agreed not to move to the next level (as long as they don't lose the current level). The game does not try to store the players data unless they win. After the game the leaderboard is shown and gives the player the option to make a new game, play another player.

ActionListener :

In the MainMenu:

The button action is triggered and checks is the user has inputted the name or not and continues accordingly.

In the Game Engine class :

All possible keys and values are defined in a HashMap.

for each key define in the HashMap, I assign the functionality of moving the game object (the bear). Up/down/left/right, with applied restriction on the movement based on the surrounding objects.

In the Leaderboard class:

the new game action listener, identifies a new game and makes sure to remove all components in the screen before doing so.

The most Interesting Part:

In the level class, and based on the willingness of the player to play, the level text file is defined through the constructor and processed line by line to build the map. In the gameEngine, and before the bear moves, using the collides function, if it collides with anything else other than a walkable object it will not walk, basically just block movement.

There ranger is defined based on the text file, the ranger can have two type defined by the parameters using the x and y as variables to describe the movement of the bear. The bear has an original position that it starts from but that position it walkable as long as the bear is not there. In case the bear collides with a ranger it losses on life and reverts it back to the starting position that is stored through the input file as well.

Test Cases :

- Action: Start the game. Result: Verify that Yogi Bear spawns at the entrance of the park.
- Action: Move Yogi Bear around the park. Result: Verifies that Yogi Bear can move freely, as long as he does not collide with obstacles.
- Action: Move Yogi Bear into a tree or mountain. Result: Verify that Yogi Bear cannot move through the obstacle and must find another path.
- Action: Collect a picnic basket. Result: Verify that the basket count is incremented correctly.
- Action: Move Yogi Bear into a ranger. Result: Verifies that Yogi Bear loses one life and respawns at the entrance.
- Action: Lose all three of Yogi Bear's lives. Result: Verify that a popup message appears, allowing the player to enter their name and save the score.
- Action: Restart the game. Result: Verify that the game restarts from the beginning.
- Action: Collect all the picnic baskets. Result: Verifies that a new game level is loaded or generated.
- Action: Access the highscore table. Result: Verifies that the table displays the top 10 scores with player names.

- Action: Lose a life and respawn. Result: Verify that Yogi Bear respawns at the same entrance location.
- Action: Move Yogi Bear into multiple rangers. Result: Verify that Yogi Bear loses a life for each ranger he collides with.
- Action: Play multiple games and achieve different scores. Result: Verify that the highscore table accurately reflects the top 10 scores.
- Action: Enter an invalid player name in the popup. Result: Verify that the game does not accept the invalid name.
- Action: Simulate a database failure when saving the score. Result: Verify that the game handles the error gracefully.
- Action: Attempt to play the game using only keyboard controls. Result: Verify that the game is accessible and can be played without a mouse.