Dagim Belete
Malyaka Imran

Updated Project Specification

PART 1:
Our project is a game called Brick Breaker. It will basically comprise of 3 components:

1. A ball that bounces
2. Bricks that break on collision with the ball
3. Saucer that the ball lands on and bounces off from every time it comes back.

The user will control the saucer's movements. Input would be given by the user through keyboard keys 'a' and 'd'. 'a' will move the saucer leftwards and 'd' will move it rightwards.

The rules of the game are such that the more number of bricks are broken, the more points you get. The user has 3 lives. Everytime the saucer misses the ball and the ball hits the floor, the user loses a life. The bricks will be stationary and will disappear once they are broken. When the user completes level 1, they can either quit or go to level 2. The game is won when level 2 is won. There will be some red bricks that give you bonus points. There are also power-ups that will fall when the ball breaks a bonus brick. The power-ups will increase speed of the saucer, freeze the ball, increase lives, earns you more score, or does nothing, just to troll the user.
PART 2:
There will be a class called Main which will extend JPanel and implement KeyboardListener and MouseListener. It will have width, height and Frames per second (FPS) as its instance variables. Width(int) and height(int) determine the size of the screen of the game and FPS(int) determines the number of times the screen will be repainted. Width and Height are ints because that's what the Graphics methods for creating the screen require. FPS is an int because you have to have a WHOLE number of frames. These 3 variables will be final because they don't have to be changed throughout the game. This class will call World to get details to fill in the

screen with our game's details (described below). We will use the Runner class from 'KeyBoardSpheres' lab in this class in order to update the screen per frame. Main also has a constructor that does not take a parameter. It has methods KeyPressed and KeyReleased that move the saucer and MouseClicked that will detect user's click on the screen to select difficulty level or quitting etc. It also overrides other methods in

MouseListener and KeyBoardListener. Main also consists of paintcomponent that draws the entire setting of our game to the screen. paintcomponent also checks for every frame the number lives, the number of broken bricks, and current level. It draws graphics to the screen according to the status of the above variables. For example, it'll draw the specific orientation for level 1 if that is the level, it'll draw the game lost image if lives are over, and game over image if the user chooses to quit after the first level. It also has play methods that play the background audio, the collision audio, and the power up audio. It also has a Timer Object that allows power ups to be retained for a specific length of time only.

World class is public as it is to be called by Main. Its constructor will take in Height and Width for the screen size, and NumBricks, the number of bricks. The constructor fills up an array of bricks which is then drawn to the screen by , makeBricks. The constructor also creates buffered image objects and loads them so that they don't have to be loaded in the paintcomponent which would have made the game lag a lot. makeBricks method also assigns some bricks to be bonus. Bonus bricks have extra points, are red in color and drop down powerups. World creates a saucer and a ball object by calling their draw methods. Score is a static variable in World which is updated by bounce method in ball every time a brick is broken.World will also have an updateBall method that calls updatePos method in Ball class and updates its position. updateBall is public because it is called by Runner class. All these methods are non-static because they are implemented on an instance of world.

The Saucer class is a public class because it is called by World. Its instance variables will be width, length, velocity, position, acceleration and color. Width and height will be used to draw a rectangle that will represent the saucer. It has a draw method that uses graphics to draw the shape. Velocity is set by setVelocity and is boosted by powerups also. On pressing the keys, the acceleration of the saucer changes using setAcceleration method. It also has update and setPosition method to update the position per frame.

The Ball class is a public class because it is called by World. Its instance variables are radius, color, a Pair of velocity x and velocity y, and a Pair of position X and position Y. Radius is an int and color is a Color object owing to the demands of the drawOval method of graphics. The constructor just assigns values to the instance variables without taking in any parameters. It has a draw method that uses graphics to draw the shape. This method is public because World calls it. setVelocity controls the velocity of the ball and is used to alter the velocity of the ball according to the level of difficulty at the start of the game. It has a Bounce method which will call FlipX and FlipY on a Pair instance if the ball collides with the bricks, the sides, or the saucer. On collision, this method also changes the boolean broken of the brick to true so that the brick stops existing for the ball, and plays the audio of collision in the background, as well increasing the count (number of broken bricks) in world. Ball also takes in a world instance in order to be able to interact with the bricks and saucer. It has an update method that changes the position X and position Y of the ball according to the velocity of the ball. This method will be called by Runner Class per frame so its public.

The Pair class is public as it is used in Ball. It has a constructor that takes in 2 doubles. It has a flipX and flipY method that would multiply that first and second component of the pair respectively with -1. It also has divide, add, and times methods to perform these operations on the Pair.

The Brick Class will be public as it is used in World. It has a draw method that uses graphics, length, width and color values from instance variables to draw the shape. This method is public because World calls it. It has a brickBroke method that turns boolean broken for the brick to true and add the points of the brick to the total score in world. The variable color is public because it is changed to background color in World if the brick breaks. It has an update method that is called by the Runner every frame and checks if the brick is bonus and broken and then drops down the power up object. Powerup is a public class as it is called by other classes. It provides power-ups depending on a String assigned in the parameter in the constructor. For example, if the string says speed, the power up will speed up the saucer for a fixed time span. It has draw and update methods to draw the power up image on the screen and update it every frame. The absorb method here allows the saucer to “absorb”/ catch the power up image and then creates the effect of the power up.

Overall, the overarching class is Main which mainly interacts with Runner, World, and has the main method. World calls other component classes including Ball, Saucer, Brick, Powerup to set up the game. Runner will interact with World to update the screen, the game status and the score every frame. We also added new libraries for audio, images, timer, and mouselistener to add to the details of the game.