Space Pong Term Project by Tuğra Demirel

Github: Deantu

Classes:

🡪TopSide:

* TopSide extends JPanel
* It requires GamePlay to get created.

Contains:

* private GamePlay play;
* Used to get health, score, time and level information from GamePlay class.
* private JTextField: time, level, score, health
* Used to show health, score, time and level information on user interface.
* public constructor
* public update method;
* Uses GamePlay getters to update all JTextFields.
* This method used in GamePlay.

🡪BotSide:

* BotSide extends JPanel implements ActionListener
* It requires GamePlay to get created.

Contains:

* private GamePlay play;
* Used to change play, theme, concept in GamePlay class.
* private JButton: playBut, reset, theme, concept

🡪When an action performed on those buttons;

* playBut uses play.setPlay(Boolean) to start/pause and resume game.
* reset uses play.restart() to start new game
* theme uses play.changeColorTheme() to change theme of game from “Dark” to “Bright or reverse, and changes name of button.
* Concept uses play.changeGameConcept() to change concept of game and changes name of button.
* public constructor
* public update method:
* Uses gameplay isPlay getter to change playBut name if game ends or game starts without using playBut

🡪Main:

* Creates a JFrame that called as obj with 1024x768 resolution and not resizable format.
* Creates GamePlay play
* Creates BotSide bot, TopSide top with using play
* Sets Bot and Top as not focusable and uses play.setBot and play.setTop to assign they to game
* Selected default close method to exit on close, added top,play and bot with using BorderLayout(North,Center,South)
* Setted visible the game.

-GamePlay:

* GamePlay extends JPanel and implements KeyListener & ActionListener
* GamePlay is the place that all the game created, it can work without TopSide and BotSide.
* While creating I used the [video of Brick Breaker Game](https://www.youtube.com/watch?v=K9qMm3JbOH0) to get main idea of game.

Contains Private Variables:

🡪 components of player;

* int playerX = 452;
* That is middle of usable X axis
* Used to place paddle.
* int playerY = 654;
* That is bottom of usable Y axis for paddle
* Used to place paddle.
* int health = 3;
* boolean play;
* Used to decide game should run or pause.
* int limit = 0;
* Used to limit level up mechanism for each minutes.
* int level = 1;
* Used to count and store players level.
* int score = 0;
* Used to count and store players score.
* Timer timer;

int delay = 20;

* Used to repeat operations in every 20 ms.
* int time = 0;
* Used to count time in seconds
* int count;
* It is helper of time to count 1 seconds with using the formula that is 1000ms/delay(20ms) = 50 (that shows how many count equals a second)

🡪 components of ball;

* int ballX = 10;
* int ballY = 10;
* double ballXSpeed = 4.0;
* double ballYSpeed = 1.0;
* int ballRadius = 10;

🡪 components of stellars;

For each stellar code contains;

* Image i;
* int positionX;
* int positionY;
* int RadiusX;
* int RadiusY;
* Boolean stellarLives;
* this variable is false for all stellars at start.
* double gravity = 9.8;
* Used to speed up & down ball

🡪 Color/Theme/Concept part

* Used to change theme and concept
* Color background;
* As default white
* Color paddle;
* As default black
* String concept;
* As default UFO concept
* String theme;
* As default Bright theme

🡪 Other Parts of Game:

* Used to update JTextFields and change theme.
* TopSide top;
* BotSide bot;

Contains Methods:

* GamePlay constructor
* createObjects(String) helper method
* Used to read file and assign images from folder to stellars, including RadiusX and RadiusY components.
* Uses createImg helper method.
* As standart it uses UFO concept.
* createImg(String) helper method
* Returns Image
* Uses ImageIO to read file and get BufferedImage
* I used stackoverflow while creating this method.
* paint(Graphics) public method override
* Used to paint all components of GamePlay
* actionPerformed(ActionEvent) public method override
* Checks if there is an interaction between objects by using Rectangle methods
  + - If there is an interaction between;
* Star and Ball : Add +100 to scores
* UFO and Ball : Reduce HP with reducehp() helper method.
* Meteor and Ball : Increase speed of ball %10 and change the direction. (Used \*= -1.1)
* Health and Ball : Increase health +1
  + - With create part, stellars be created, when the position of ball is suitable to create.
    - Health can be created if player has less than 3 lives.
* calculateTime() private helper method
* It uses count variable to count each 20ms delayed repetition.
* When count reaches a multiple of 50 it makes a second then method increases time by 1.
* When time reaches a multiple of 60 it makes a minute then method uses levelUp method to level up and it increases the limit to prevent overleveling.
* levelUp() private helper method
* It increases level by 1.
* It increases initial speed by %50 and the starting speed %50 for remaining lives.
* keyPressed(KeyEvent) public method override
* It used to move paddle right and left with using moveRight and moveLeft helper methods if there is enough space to move.
* moveLeft() and moveRight() private helper methods
* They move the paddle 30 pixels left/right.
* reduceHP() private helper method
* It reduces health by 1 and resets game if there is remaining lives.
* If there is any remaining lives it stops game.
* changeColorTheme() protected method
* It is used in BotSide in theme button to change “Bright”, and “Dark” theme
* changeGameConcept() protected method
* It is used in BotSide in concept button to change concept in “UFO”, “tooth”, and “covid”.
* It uses restart() and createObjects(String) method.
* restart() protected method
* It clears everything on the game, resets level, ball, and player position with helper methods.
* It also resets time, count and score.
* clearStellars() private helper
* Chages all stellarLives to false to clear stellars.
* levelReset() private helper
* Changes limit to 0 and level to 1.
* ballReset() private helper
* Resets ball position to (10,10) uses Math.pow to calculate starting speed of ball.
* playerReset() private helper
* Resets player position to middle and changes remaining lives to 3.

🡪 For Each Stellar:

* createStellar() private method
* That method gives a random position to stellar and checks if other objects intersect with it.
* If there is any interaction between objects it gives a new random position to stellar in do-while loop.
* public setters & getters