Thanet Sirichanyaphong
5730251121

Project: Flappy Pipe

This is a game with a goal to get the most score by finish the enemies. Figure 1 shows the menu screen which you can choose start option, high score option, help option or quitting the game.

Figure 2 shows high score screen, this screen will show who gets the highest score. Figure 3 shows help option screen, this screen will show how to play this game, when this screen show you can press any key to continue the screen. Figure 4 shows start game screen. In this game, you have to protect your life as long as you can do. The game will end when you is dead (life point is empty). If you get the high score, the input box will show up and you can input your name (Figure 5). If you don't get the high score, the box will show the game is over (Figure 6) and it will go back to the menu screen.

There are 3 keys that you can use when the game is running. The first is "**spacebar**" for pipe jumping. The second is "**R key**" for using bomb skill. The last is "**enter**" for pause the game (Figure 7). If you eliminate the enemies you will get 1 score point. There are 2 kinds of enemies (normal bird and bomber bird). The normal birds could be finished by hitting the pipe. If normal birds fly through the pipe, you will lose a life point. The bomber birds could be finished by letting them through the pipe. If bomber birds hit the pipe, you will lose a life point.

Every time you lose a life point, the pipe will be flinching for 2 seconds and it can't take any damage when it is flinching.

Thanet Sirichanyaphong
5730251121

Figure 1

Figure 2

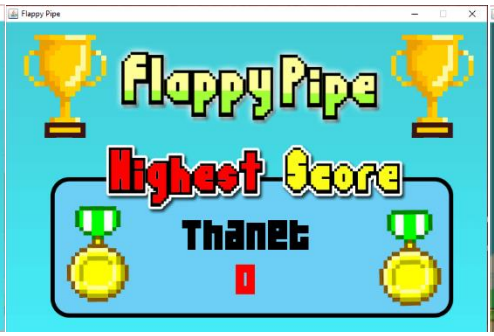


Figure 3

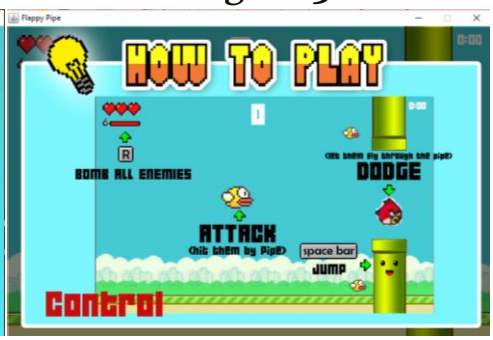


Figure 4

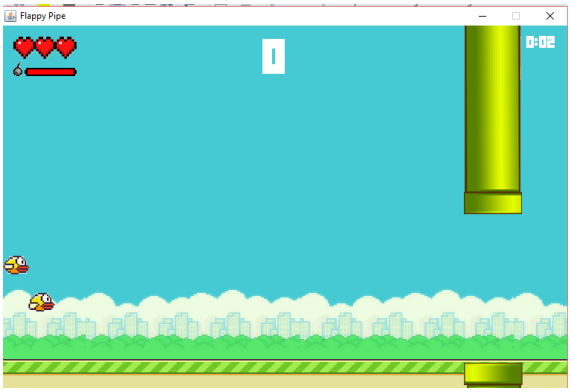


Figure 5

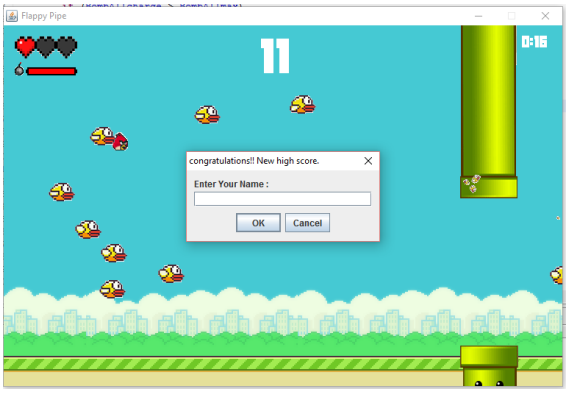


Figure 6

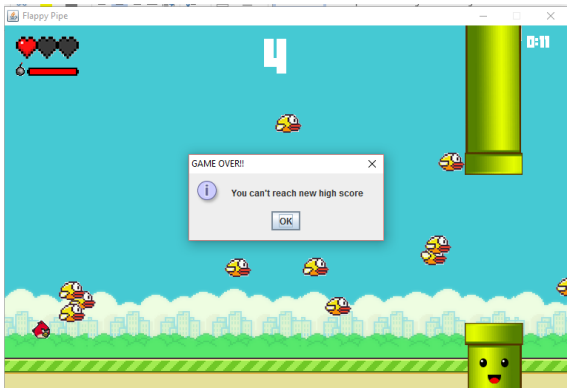
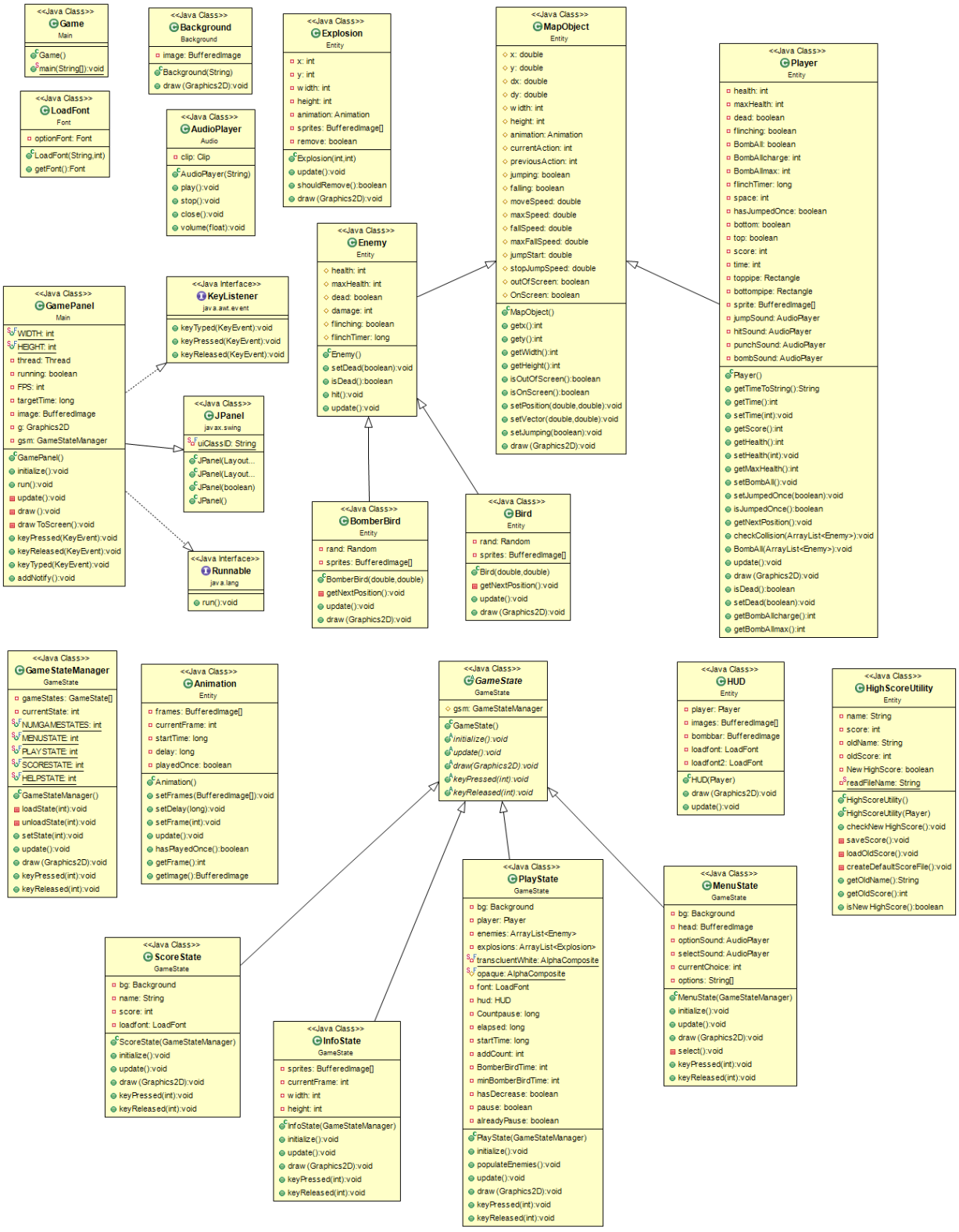


Figure 7

Thanet Sirichanyaphong
5730251121

UML Diagram



Thanet Sirichanyaphong
5730251121

Player:

- There is only one player.
- Player will gain a score point when finishing the enemies.
- Player will lose a life point when enemies get their object.
- Player always move down and cannot go out of bound.
- Player can move only by jumping when the spacebar is triggered.
- Player can use special skill to destroy all enemies on the screen when you pressed the R key.
- The special skill can be stock only 1 time and can be recharged in 15 seconds.
- Player has 3 life point. If Player lose all life point, the game is over.

Enemies:

- There are 2 kinds of enemies (normal bird and bomber bird).
- Normal bird can be destroyed by hitting the pipe.
- Normal bird can attack the player by get through the pipe.
- Bomber bird can be destroyed by getting through the pipe
- Bomber bird can attack the player by hitting the pipe.
- When enemies die it will explode.

High score:

- If there is not high score before, it will create default high score.
- If player get new high score, it will ask the player to input his name.

Thanet Sirichanyaphong
5730251121

❶ Class “Game”

The main class of this game

1.1 Methods

- Static void main (); to create JFrame and set property of JFrame.

❷ Class “GamePanel”

The JPanel class that contains the game loop.

1.1 Fields

- static final int WIDTH = 760; JPanel’s width
- static final int HEIGHT = 490; JPanel’s height
- Thread thread; thread of this game
- boolean running; check the game is running
- int FPS = 60; FPS of this game
- long targetTime = 1000 / FPS; times per FPS
- BufferedImage image; use to create graphic
- Graphics2D g; graphic of this game
- GameStateManager gsm; control current game state

1.2 Constructor

- GamePanel();call super(); set size and requestFocus();

1.3 Methods

- void initialize(); initialize BufferedImage image,Graphics2D g, GameStateManager gsm and set running = true
- void run(); call initialize(); and loop the game
- void update(); update GameStateManager gsm
- void draw(); draw GameStateManager gsm
- void drawToScreen(); draw to screem
- void keyPressed(KeyEvent e); listen the key
- void keyReleased(KeyEvent e); listen the key
- void addNotify(); notify the super class and start the thread

Thanet Sirichanyaphong
5730251121

❸ Class “Background”

The class that control the background screen.

1.1 Fields

- BufferedImage image; contain the background image

1.2 Constructor

- Background(String s); load image by class loader

1.3 Methods

- void draw(Graphics2D g); draw image

❹ Class “LoadFont”

The class that use to load custom font.

1.1 Fields

- Font optionFont; contain custom font

1.2 Constructor

- LoadFont(String s, int size); load custom font from path s and set font size

1.3 Methods

- Font getFont(); return custom font

❺ Class “AudioPlayer”

The class that use to load audio.

1.1 Fields

- Clip clip; contain the audio clip

1.2 Constructor

- AudioPlayer(String s); load audio from path s

1.3 Methods

- void play(); play the audio clip
- void stop(); stop the audio clip
- void close(); close the audio clip
- void volume(float f); set volume of the audio clip

Thanet Sirichanyaphong
5730251121

❻ Abstract Class "GameState"

The class is the template of game state.

1.1 Fields

- GameStateManager gsm; contain gamestatemanager

1.2 Methods

- abstract void initialize();
- abstract void update();
- abstract void draw(Graphics2D g);
- abstract void keyPressed(int k);
- abstract void keyReleased(int k);

❼ Class "GameStateManager"

The class that manager the game state

1.1 Fields

- GameState[] gameStates; contain all game states
- int currentState; current state
- static final int NUMGAMESTATES = 4; gameStates's size
- static final int MENUSTATE = 0; set menu state point
- static final int PLAYSTATE = 1; set play state point
- static final int SCORESTATE = 2; set score state point
- static final int HELPSTATE = 3; set help state point

1.2 Constructor

- GameStateManager(); create gameStates and call loadState(currentState)

1.3 Methods

- void loadState(int state); load gameStates[state]
- void unloadState(int state); set gameStates[state] = null
- void setState(int state); unload current state and load the new state
- void update(); update the current state
- void draw(graphics2D g); draw the current state
- void keyPressed(int k); listen the key
- void keyReleased(int k); listen the key

❽ Class "InfoState"

The game state of help option

1.1 Fields

- BufferedImage[] sprites; images of how to play
- int currentFrame; current image
- int width, height; image's size

1.2 Constructor

- InfoState(GameStateManager gsm);
    - set width,height,currentFrame=0,call initialize();

1.3 Methods

- void initialize(); load image of how to play and split them into sprites
- void update();
- void draw(graphics2D g); draw the current image
- void keyPressed(int k);
    - if press any key go to next image, if it don't have next image go to menu state
- void keyReleased(int k);

❾ Class "ScoreState"

The game state of high score option

1.1 Fields

- Background bg; image of score state's background
- String name ; name of player who get highest score
- int score; highest score
- LoadFont loadfont; load custom font

1.2 Constructor

- ScoreState(GameStateManager gsm); load name,score and background

1.3 Methods

- void initialize();
- void update();
- void draw(graphics2D g); draw the score state
- void keyPressed(int k); if press any key go to menu state
- void keyReleased(int k) ;

⓾ Class “MenuState”

The menu state of game

1.1 Fields

- Background bg; image of background
- BufferedImage head; image of head option
- AudioPlayer optionSound; sound of choosing option
- AudioPlayer selectSound; sound of select option
- int currentChoice = 0; current option
- String[] options = { "Start", "High Score", "HELP", "Quit" }; all options

1.2 Constructor

- MenuState(GameStateManager gsm); load background, all sounds, head image

1.3 Methods

- void initialize();
- void update();
- void draw(graphics2D g); draw background and draw head option
- void select(); select current option
- void keyPressed(int k); press up and down to choose and enter to select option
- void keyReleased(int k) ;

⓫ Class “PlayState”

The state of game playing

1.1 Fields

- Background bg; image of background
- Player player; this player
- ArrayList<Enemy> enemies; all enemies
- ArrayList<Explosion> explosions; all explosion
- static final AlphaComposite transcluentWhite = AlphaComposite.getInstance(AlphaComposite.SRC_OVER, 0.7f);
- static final AlphaComposite opaque = AlphaComposite.getInstance(AlphaComposite.SRC_OVER, 1);
- LoadFont font; custom font
- HUD hud; player’s hud
- long Countpause; starting pause time
- long elapsed; pause time

- long startTime; game start time
- int addCount; count enemies adding
- int BomberBirdTime; time generate bomber bird
- int minBomberBirdTime; min time generate bomber bird
- boolean hasDecrease; check already decrease BomberBirdTime
- boolean pause; the game is pause or not
- Boolean alreadyPause; the game is pausing or not

1.2 Constructor

- PlayState(GameStateManager gsm); call initialize();

1.3 Methods

- void initialize();
  - load background, sound, custom font
  - initialize Countpause, player, hud, enemies, addCount, BomberbirdTime, minBomberbirdTime, explosions, startTime
- void populateEnemies(); generate enemies by random position
  - random yBombBird that in the game area
  - random yBird that in the game area but not in range of yBombBird
  - generate normal bird every time
  - generate bomber bird every BomberBirdTime
- void update(); update everything is running in this state
  - first check game is pause
  - update player, enemies, hud, explosion
  - each time update check the collision
  - call populateEnemies() every 4 seconds
  - check game is over at the end
- void draw(graphics2D g);
  - draw background, player, enemies, explosions, hud, pause
  - when pause draw the pause
  - when unpausing draw the count down time and game will continue
- void keyPressed(int k);
  - press spacebar to jump
  - press enter to pause or unpausing
  - prses R key to use special skill
  - if the game is pausing, player cannot do anything
- void keyReleased(int k) ;
  - check release spacebar and enter

Thanet Sirichanyaphong
5730251121

⓬ Class “Animation”

The class that control animation of this game

1.1 Fields

- BufferedImage[] frames; all images of animation frames
- int currentFrame; current frame’s image
- long startTime; starting time
- long delay; delay of playing animation
- boolean playedOnce; has play this animation before

1.2 Constructor

- Animation(); set playedOnce=false

1.3 Methods

- void setFrames(BufferedImage[] frames);
  - load frames
  - set currentFrame =0 and playOnced=false
  - initialize startTime
- void setDelay(long d); setter delay
- void setFrame(int i); setter current frame
- void update();
  - if delay=-1 return;
  - check elapsed> delay go to next frame
  - if play all frame, set playOnced =true and currentFrame=0
- boolean hasPlayedOnce(); getter playOnced
- int getFrame() ; getter currentFrame
- BufferedImage getImage(); getter frames

Thanet Sirichanyaphong
5730251121

⓭ Class "Enemy"

The template of all enemies

1.1 Fields

- int health; enemy's health
- int maxHealth; enemy's max health
- boolean dead; enemy is dead or not
- int damage; enemy's damage
- boolean flinching; enemy is flinching or not
- long flinchTimer; flinching time

1.2 Constructor

- Enemy();

1.3 Methods

- void setDead(boolean b); setter dead
- boolean isDead(); getter dead
- void hit(); set dead = true
- void update();

⓮ Class "Bird"

The class of normal bird

1.1 Fields

- Random rand; random speed of bird
- BufferedImage[] sprites; animation frame of bird

1.2 Constructor

- Bird(double x, double y);
    - call setPosition(x,y); and random maxSpeed and set moveSpeed =1
    - set width=36,height=25,damage=1
    - load image of animation frames and split into sprites
    - initialize animation and add sprites to animation and set delay =200

1.3 Methods

- void getNextPosition(); set next position of bird
- void update(); update next position
    - check the position is on screen or not
    - update animation
- void draw(Graphics2D g); call super.draw(g);

⓯ Class "BomberBird"

The class of bomber bird

1.1 Fields

- Random rand; random speed of bird
- BufferedImage[] sprites; animation frame of bird

1.2 Constructor

- Bird(double x, double y);
  - call setPosition(x,y); and random maxSpeed and set moveSpeed =1
  - set width=24,height=25,damage=1
  - load image of animation frames and split into sprites
  - initialize animation and add sprites to animation and set delay =150

1.3 Methods

- void getNextPosition(); set next position of bird
- void update(); update next position
  - check the position is on screen or not
  - update animation
- void draw(Graphics2D g); call super.draw(g);

⓰ Class "Explosion"

The class of enemy's explosion

1.1 Fields

- int x, y ; x and y position of explosion
- int width; width of explosion's frame
- int height; height of explosion's frame
- Animation animation; animation of explosion
- BufferedImage[] sprites; images of animation frames
- boolean remove; explosion is finish or not

1.2 Constructor

- Explosion(int x, int y);
  - set position x, y and width=36,height=25,
  - load image of animation frames and split into sprites
  - initialize animation and add sprites to animation and set delay =50

1.3 Methods

- void update(); update animation, if explosion is finish, set remove=true
- boolean shouldRemove() getter remove
- void draw(Graphics2D g); draw image

⓱ Class “HUD”

The class that show status of player

1.1 Fields

- Player player; player of this game
- BufferedImage[] images; images of life bar
- BufferedImage bombbar; image of special skill bar
- LoadFont loadfont; custom font
- LoadFont loadfont2; custom font2

1.2 Constructor

- HUD(Player player);
    - set player and load font
    - load image of life bar, split into images and load image of bomber bar

1.3 Methods

- void update();
- void draw(Graphics2D g); draw all image
    - draw life bar following the player health
    - set font to draw score of player and set font2 to draw time of game
    - draw bomber bar

⓲ Class “HighScoreUtility”

The class that control loading and setting high score

1.1 Fields

- String name; name of current player
- int score; score of current player
- String oldName; name of highest score player
- int oldScore; score of highest score player
- boolean NewHighScore; current player’s score is more than old score or not
- static String readFileName = "highscore"; file name

1.2 Constructor

- HighScoreUtility(); call loadOldScore();
- HighScoreUtility(Player player)
    - set player score = score and call loadOldScore();

1.3 Methods

- void checkNewHighScore();
  - check current player's score is more than old score or not
  - if yes, set NewHighScore=true and let player input his name
  - call saveScore();
- void saveScore();
  - save current score and name to file
- void loadOldScore();
  - if there are not any file before call createDefaultScoreFile();
  - load the name and score in file to oldName and oldScore
- void createDefaultScoreFile();
  - create a file that contain the string "Thanet\no"
- String getOldName(); getter oldName
- int getOldScore(); getter oldScore
- boolean isNewHighScore(); getter NewHighScore

⓳ Class "Player"

The class that control everything of player

1.1 Fields

- int health; player's health
- int maxHealth; player's max health
- boolean dead; player is dead or not
- boolean flinching; player is flinching or not
- boolean BombAll; player is using special skill or not
- int BombAllcharge; special skill charge
- int BombAllmax; special skill max charge
- long flinchTimer; player's flinch time
- int space; space between pipe
- boolean hasJumpedOnce; player is jump before or not
- boolean bottom; pipe is on ground or not
- boolean top; pipe is at the top or not
- int score; player's score
- int time; player's time playing
- Rectangle toppipe; convert image of toppipe to rectangle
- Rectangle bottompipe; convert image of bottompipe to rectangle

- BufferedImage[] sprite; images of pipe
- AudioPlayer jumpSound; jump sound
- AudioPlayer hitSound; hit sound
- AudioPlayer punchSound; punch sound
- AudioPlayer bombSound; special skill sound

1.2 Constructor

- Player();
  - set x=620, y=200, space=200, width=80, height=490, falling=true
  - set BombAllcharge = BombAllmax = 1000
  - set fallSpeed =1, maxFallSpeed=5, jumpStart=-10, health=maxHealth=3
  - load all sounds and images

1.3 Methods

- String getTimeToString();
  - return the playing time to minutes:seconds
- create getter & setter
- void getNextPosition(); pipe always fall down
- void checkCollision(ArrayList<Enemy> enemies);
  - check all ememies collision with pipe
  - if normal bird collide with pipe, player get 1 score point and bird is dead
  - if normal bird is out of screen, player lose 1 life point and bird is dead
  - if bomber bird is out of screen, player get 1 score point and bird is dead
  - if bomber bird collide with pipe, player lose 1 life point and bird is dead
- void BombAll(ArrayList<Enemy> enemies);
  - destroy all enemies in the list and play bomb sound
  - add score equals the enemies that are on the screen
- void update();
  - call time++, BombAllcharge++ every time update is called
  - BombAllcharge cannot be more than BombAllmax
  - call getNextPosition();
  - if player is flinching, flinch 2 seconds
  - if player is jumping, play jump sound
  - update falling
  - player cannot go out the bound
- void draw(Graphics2D g)
  - draw the player
  - if player is flinching draw the player flinches

Thanet Sirichanyaphong
5730251121

⓴ Class "MapObject"

The class that control loading and setting high score

1.4 Fields

- double x; x position of object
- double y; y position of object
- double dx; next x position of object
- double dy; next y position of object
- int width; width of object
- int height; height of object
- Animation animation; animation of object
- int currentAction; currentAction of animation
- int previousAction; previousAction of animation
- boolean jumping; object is jumping or not
- boolean falling; object is falling or not
- double moveSpeed; moveSpeed of object
- double maxSpeed; maxSpeed of object
- double fallSpeed; fallSpeed of object
- double maxFallSpeed; maxFallSpeed of object
- double jumpStart; jumpSpeed of object
- double stopJumpSpeed; stopJumpSpeed of object
- boolean outOfScreen; object is out of screen or not
- boolean OnScreen; object is on screen or not

1.5 Constructor

- MapObject();

1.6 Methods

- create getter & setter
- void setPosition(double x, double y); set x, y
- void setVector(double dx, double dy); set dx, dy
- void draw(Graphics2D g); draw image from animation.getImage()