**Feature Details**

1. **Doodle Movement:**

**Description:**

The doodle’s movement mimics elasticity and gravity. It is always experiencing the downward acceleration of 10 pixels per timerFired. If the doodle jumps on a platform, it will experience an upward acceleration of 50 per timerFired. Meanwhile, the vertical and horizontal speed is set to zero. The player controls the doodle by pressing the Left key or Right key. Pressing the key will generate a 20 pixel per timerFired acceleration. If the horizontal speed is not zero, the doodle will experience a horizontal resistance opposite to the movement direction. When moving leftward, the doodle will face left. When moving rightward, the doodle will face right. When shooting the bullets, the doodle will face up.

**Related Function and Line Count:**

doodleMove(app) – 56 lines

drawDoodle(app,canvas) – 4 lines

**Total Line Count – 60 lines**

1. **Infinitely Scrolling Up/Down:**

**Description:**

When the doodle goes higher than half of the canvas, the canvas will scroll up at the speed of half of the doodle’s speed. So, sometimes the doodle goes out of the canvas's upper bound. This is a feature to increase the difficulties of the game. Similarly, when the game is over, the canvas will scroll down until the doodle goes out of the lower bound.

**Related Function and Line Count:**

doodleVisible(app) – 15 lines

**Total Line Count – 15 lines**

1. **Normal Platforms Generation and Renew:**

**Description:**

At the beginning of the game, 20 platforms will be generated on canvas. The position of the platforms is random, yet the game is still winnable. That’s to say, there is at least one platform that the doodle can continue jumping on. To be counted as jumping on a platform, at least half of the doodle’s width needs to be on that platform (The nose of the doodle count as a part of the doodle’s width).

With the doodle going up, the canvas can scroll up infinitely. As the old platforms get out of the bound of the canvas, the new platforms will be generated, while the number of the platform will always be the same. Every 1000 pixels the doodle goes up, the number of the platform will decrease by two.

**Related Function and Line Count:**

class Platform - 6 lines

platformNotToClose(app,platforms,newX,newY) – 11 lines

platformNotToFar(app,platforms,newX,newY) – 8 lines

getFirstPlatforms(app) – 5 lines

getPlatforms(app) – 14 lines

renewPlatforms(app) – 31 lines

changePlatformsAndCoinsNum – 9 lines

doodleJumpingOnPlatform(app, platformX, platformY) – 6 lines

drawPlatforms(app,canvas) – 5 lines

**Total Line Count – 95 lines**

1. **Moving platform:**

**Description:**

Moving platforms are the platform that moves left and right. Every 1000 pixels the doodle goes up, the number of the platform will increase by one.

**Related Function and Line Count:**

class movingPlatform(Platform) – 4 lines

getMovingPlatforms(app) – 12 lines

renewMovingPlatforms(app) – 24 lines

moveMovingPlatforms(app) – 7 lines

doodleJumpingOnMovingPlatform(app, platformX, platformY) – 6 lines

drawMovingPlatforms(app,canvas) – 5 lines

**Total Line Count – 58 lines**

1. **Coins:**

**Description:**

3 to 5 coins randomly appear on the platforms. If the doodle jumps on a platform with a coin, the doodle will get that coin, and the coin will be disappeared on the platform. As the old coins are collected by the doodle or getting out of the canvas bound, new coins will be generated. The number of coins will always be the same unless we are unable to generate enough coins. Note that a coin may appear at the same position several times.

**Related Function and Line Count:**

getCoins(app) – 9 lines

renewCoin(app) – 14 lines

doodleTakesCoin(app) – 9 lines

drawCoins(app,canvas) – 6 lines

**Total Line Count – 38 lines**

1. **Monster:**

**Description:**

When the doodle gets more than 20 coins, a monster will appear every 1000 pixels while the doodle gets up. The monster will move horizontally on the screen if it doesn’t detect any platform 60 pixels below. When the monster is moving to the left, the face is facing leftward. When the monster is moving to the right, the face is facing rightward.

If the doodle bumps a monster while jumping up, the game is over. The doodle can kill the monster by shooting a bullet toward it or stepping on it. As long as the doodle touches the monster, it will be counted as bumping on the monster if the doodle is going upward and stepping on the monster if the doodle is going downward. The killed monster will disappear.

**Related Function and Line Count:**

class Monster – 14 lines

class Bullet – 5 lines

monsterIsLegal(app,x,y) – 5 lines

getMonster(app) – 11 lines

renewMonster(app) – 7 lines

monsterShaking(app) – 3 lines

isMonsterMovingHorizontal(app) – 5 lines

monsterMovingHorizontal(app) – 21 lines

bumpOnMonster(app) – 11 lines

shootingBullet(app) – 5 lines

renewBullet(app) – 5 lines

bulletHitMonster(app) – 7 lines

bulletKillMonster(app) – 5 lines

doodleJumpingOnMonster(app) – 11 lines

drawMonster(app,canvas) – 9 lines

drawBullet(app,canvas) – 4 lines

**Total Line Count – 138 lines**

1. **Flying Hat:**

**Description:**

A flying hat will appear when the canvas is not scrolling. If the doodle takes the flying hat, it will get an acceleration of 90 pixels upwards. The doodle can only take the flying hat once in the game. That is to say, after taking a flying hat, the flying hat will not appear again.

**Related Function and Line Count:**

class FlyingHat – 8 lines

needFlyingHat(app) – 4 lines

getFlyingHat(app,platformsList) – 13 lines

renewFlyingHat(app) – 5 lines

takeFlyingHat(app) – 14 lines

drawFlyingHat – 5 lines

**Total Line Count – 49 lines**

1. **Spring**

**Description:**

One or two springs will randomly appear on the platform. If the doodle jumps on a spring, it will get an upward acceleration of 70 pixels.

**Related Function and Line Count:**

getSpring(app) – 8 lines

renewSpring(app) – 14 lines

jumpOnSpring(app) - 6 lines

drawSpring(app,canvas) – 5 lines

**Total Line Count – 33 lines**

1. **User Interface:**

**Description:**

When the players first enter the game, they will see a menu. By clicking on the button on the menu, we can go to a different mode. Instruction mode gives guidelines. how to play the game, and play mode is where the game is being played.

**Related Function and Line Count:**

Class Button – 20 lines

menuMode\_mousePressed(app,event) – 6 lines

instructionMode\_mousePressed(app,event) – 3 lines

gameMode\_mousePressed(app,event) – 6 lines

**Total Line Count – 53 lines**

1. **Others**

**Description:**

Some buildin features and supporting features.

**Related Function and Line Count:**

appStartet(app) – 103 lines

restartGame(app) – 51 lines

checkWin(app) – 3 lines

calculateScore(app) – 3 lines

gameMode\_timerFired(app) – 31 lines

gameMode\_keyPressed(app,event) – 22 lines

drawScoreAndCoins(app,canvas) – 5 lines

drawWinMessage(app,canvas) – 9 lines

drawGameOverMessage(app,canvas) – 6 lines

menuMode\_redrawAll(app,canvas) – 6 lines

gameMode\_redrawAll(app,canvas) – 13 lines

instructionMode\_redrawAll(app,canvas) – 30 lines

**Total Line Count – 282 lines**

**Algorithm**

1. **Scrolling Up and Down**

This feature is achieved by moving everything except the doodle downward.

1. **Getting and Renewing Objects**

I use a list in appStarted to keep track of most of the objects. After a scrolling takes place. We will check if the object is still within the bounds of the canvas. If it is, append it to a new list. Then, if the length of the new list is shorter than the desired number. We will generate a new object and check if the newly generated object is legal. If it is, then append it to the new list. Last, we replace the list in the appStarted with the new list.

1. **Detecting if Jumping on an Object or Not**

This feature is achieved in two steps. First, we check if the doodle’s y position fulfills the requirement of jumping on an object. We do so by checking if the doodle’s y position is within the bound of (object y position – speed, object y position + speed). Second, we check if the doodle’s x position fulfills the requirement of touching the object’s x position. If we detect the doodle as jumping on a platform, we will set the y position at the position where the doodle touches the object.

1. **Making Sure the Game is Winnable**

There are several features to help guarantee that the game is winnable. First, a helper function is going to make sure that the platforms aren’t too far from each other. That function is guarantee that the doodle always has a platform to jump on.

Second, even if we are unable to generate new platforms at a certain position, the spring, the moving platform, and the flyer higher is still going to ensure that the doodle can continue going upward until the new platform is able to be generated.