Latthe Education Society’s Polytechnic,Sangli



**Advanced Java**

Report of Micro-Project

**BrickBreaker Game**

Presented By

|  |  |  |
| --- | --- | --- |
| **Roll No** | **Name** | **Enrolment No** |
| 193319 | Aditya Y Gujar | 1700430075 |

**BrickBreaker Game**

**1.0:Rational:**

Java is a set of computer software and specifications developed by James Gosling at Sun Microsystems, which was later acquired by the Oracle Corporation that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. Java applets, which are less common than standalone Java applications, were commonly run in secure, sandboxed environments to provide many features of native applications through being embedded in HTML pages.

The BrickBreaker game is a computer based application for entertainment .It consist of some bricks , ball and a paddle if all bricks are broken by ball and then you will be winner else you lose the game and score will be displayed on the screen .

1. **Intended course outcomes:**

* Handle events of AWT and GUI component
* Develop programs for handling events in java.
* Develop Java Programs using GUI Framework (Swing).

1. **Literature review:**

* <https://en.wikipedia.org/wiki/Java_(software_platform)>

Java is a set of computer software and specifications developed by James Gosling at Sun Microsystems, which was later acquired by the Oracle Corporation that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. Java applets, which are less common than standalone Java applications, were commonly run in secure, sandboxed environments to provide many features of native applications through being embedded in HTML pages.

* <https://en.wikipedia.org/wiki/Brick_Breaker>

1. Brick Breaker is a [Breakout clone](https://en.wikipedia.org/wiki/Breakout_clone) which the player must smash a wall of bricks by deflecting a [bouncing ball](https://en.wikipedia.org/wiki/Bouncing_ball) with a paddle. The paddle may move horizontally and is controlled with the BlackBerry's track wheel, the computer's mouse or the touch of a finger (in the case of touchscreen). The player gets 3 lives to start with; a life is lost if the ball hits the bottom of the screen. When all the bricks have been destroyed, the player advances to a new, harder level. There are 34 levels. Many levels have unbreakable silver bricks. If all lives are lost, the game is over. There are many versions of brick breaker, some in which you can shoot flaming fireballs or play with more than one ball if the player gets a power up.
2. **4.0 Actual methodology:**

**Code of project:**

**1.Main class:**

package brickbreaker;

import javax.swing.\*;

public class Main {

public static void main(String args[]){

JFrame obj= new JFrame();

obj.setBounds(10,10,700,600);

Gameplay gameplay=new Gameplay();

obj.setTitle("Brick breaker");

obj.setResizable(false);

obj.setVisible(true);

obj.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

obj.add(gameplay);

}

}

**GamePlay class:**

package brickbreaker;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import javax.swing.\*;

public class Gameplay extends JPanel implements KeyListener, ActionListener{

private boolean play =false;

private int score=0;

private int totalBricks=21;

private Timer timer;

private int delay=4;

private int playerX=310;

private int ballposX=120;

private int ballposY=350;

private int ballXdir=-1;

private int ballYdir=-2;

private MapGenerator map;

public Gameplay() {

map =new MapGenerator(3,7);

addKeyListener(this);

setFocusable(true);

setFocusTraversalKeysEnabled(false);

timer =new Timer(delay,this);

timer.start();

}

public void paint(Graphics g) {

//background

g.setColor(Color.BLACK);

g.fillRect(1, 1, 692, 592);

map.draw((Graphics2D)g);

//borders

g.setColor(Color.yellow);

g.fillRect(0, 0, 3, 592);

g.fillRect(0, 0, 692,3);

g.fillRect(691, 0, 3, 592);

//scores

g.setColor(Color.WHITE);

g.setFont(new Font("serif",Font.BOLD,25));

g.drawString(""+score, 590, 30);

//paddle

g.setColor(Color.green);

g.fillRect(playerX, 550, 100, 8);

//ball

g.setColor(Color.yellow);

g.fillOval(ballposX, ballposY, 20, 20);

if(totalBricks<=0) {

if(ballposY >570) {

play=false;

ballXdir=0;

ballYdir=0;

g.setColor(Color.YELLOW);

g.setFont(new Font("serif",Font.BOLD,30));

g.drawString("jinklo Re", 260, 300);

g.setFont(new Font("serif",Font.BOLD,20));

g.drawString("Press enter to restart", 230, 350);

}}

if(ballposY >570) {

play=false;

ballXdir=0;

ballYdir=0;

g.setColor(Color.YELLOW);

g.setFont(new Font("serif",Font.BOLD,30));

g.drawString("Game over, Scores:", 190, 300);

g.setFont(new Font("serif",Font.BOLD,20));

g.drawString("Press enter to restart", 230, 350);

}

g.dispose();

}

@Override

public void actionPerformed(ActionEvent e) {

timer.start();

if(play) {

if(new Rectangle(ballposX, ballposY,20,20).intersects(new Rectangle(playerX,550,100,8))) {

ballYdir= -ballYdir;

}

for(int i =0;i<map.map.length;i++) {

for(int j=0;j<map.map[0].length;j++) {

if(map.map[i][j]>0) {

int brickX =j\*map.brickWidth+80;

int brickY=i\*map.brickHeight+50;

int brickWidth=map.brickWidth;

int brickHeight=map.brickHeight;

Rectangle rect = new Rectangle(brickX,brickY,brickWidth,brickHeight) ;

Rectangle ballRect=new Rectangle(ballposX,ballposY,20,20);

Rectangle brickRect=rect;

if(ballRect.intersects(brickRect)) {

map.setBrickValue(0, i, j);

totalBricks--;

score+=5;

if(ballposX +19<=brickRect.x || ballposX+1>=brickRect.x+brickRect.width) {

ballXdir=-ballXdir;

}else {

ballYdir=-ballYdir;

}

}

}

}

}

ballposX +=ballXdir;

ballposY +=ballYdir;

if(ballposX < 0) {

ballXdir = -ballXdir;

}

if(ballposY < 0) {

ballYdir =-ballYdir;

}

if(ballposX > 670) {

ballXdir = -ballXdir;

}

}

repaint();

}

@Override

public void keyTyped(KeyEvent e) {}

@Override

public void keyReleased(KeyEvent e) {}

@Override

public void keyPressed(KeyEvent e) {

if(e.getKeyCode()== KeyEvent.VK\_RIGHT) {

if(playerX >=600) {

playerX=600;

}else {

moveRight();

}

}

if(e.getKeyCode()== KeyEvent.VK\_LEFT) {

if(playerX <10) {

playerX=10;

}else {

moveLeft();

}

}

if(e.getKeyCode()== KeyEvent.VK\_ENTER) {}

if(!play) {

play=true;

ballposX=120;

ballposY=350;

ballXdir=-1;

ballYdir=-2;

playerX=310;

score=0;

totalBricks=21;

map=new MapGenerator(3,7);

repaint();

}

}

public void moveRight() {

play=true;

playerX+=20;

}

public void moveLeft() {

play=true;

playerX-=20;

}

}

**MapGenerator class:**

package brickbreaker;

import java.awt.BasicStroke;

import java.awt.Color;

import java.awt.Graphics2D;

public class MapGenerator {

public int map[][];

public int brickWidth;

public int brickHeight;

public MapGenerator(int row ,int col ) {

map=new int[row][col];

for(int i=0;i<map.length;i++) {

for(int j=0;j<map[0].length;j++) {

map[i][j]=1;

}

}

brickWidth=540/col;

brickHeight=150/row;

}

public void draw(Graphics2D g) {

for(int i=0;i<map.length;i++) {

for(int j=0;j<map[0].length;j++) {

if(map[i][j]>0) {

g.setColor(Color.white);

g.fillRect(j \* brickWidth +80, i\*brickHeight+ 50, brickWidth, brickHeight);

g.setStroke(new BasicStroke(3));

g.setColor((Color.BLACK));

g.drawRect(j \* brickWidth +80, i\*brickHeight+ 50, brickWidth, brickHeight);

}}}} public void setBrickValue(int value,int row,int col) {

map[row][col]=value;

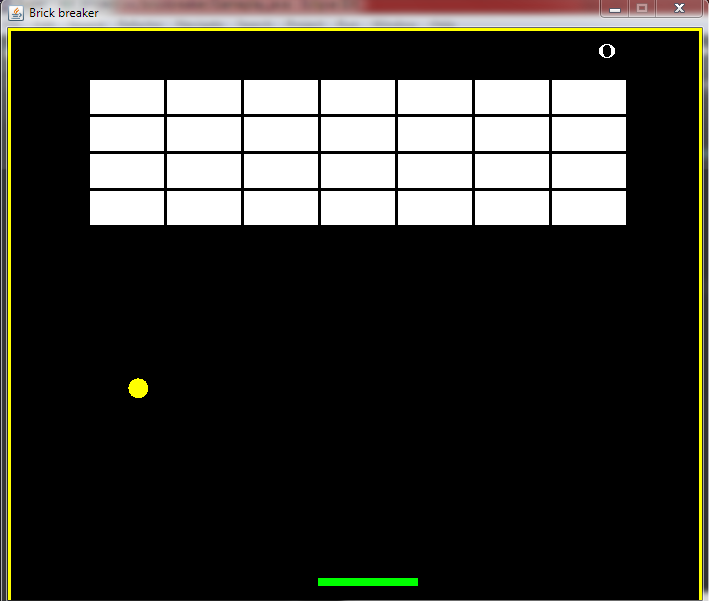
}

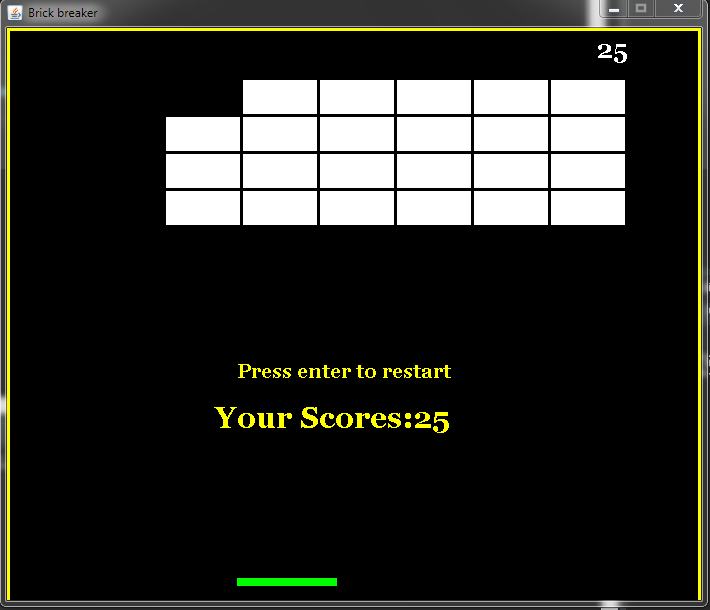
}

**6.0 Resources used**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Name and resources used | Specifications | Oty | Remark |
| 1 | Operating system | Windows | -- | -- |
| 2 | Application Software | Eclipse ide 208-2019 | -- | -- |
| 3 | Hard disc | 1 TB | 1 |  |
| 4 | Ram | 4GB | -- |  |
| 5 | Processor | Intel core 2duo cpu 2.93 GHZ | -- |  |

1. **Output of program:**







1. **Skill development:**

* I have learned the different actionListener from this project
* I understood different graphics methods

1. **future development:**

* Multiple maps can be added in future development
* Different levels can be added in this game start with easy to hard
* we can change the background colour paddle colour brick and ball colour as level increases