# LAMPIRAN

1. ***Class* Menu**

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  import static marblegun.playMusic.playMusic;  public class Menu extends javax.swing.JFrame {  public Menu() {  initComponents();  // mengambil ukuran layar  // mengambil ukuran layar  Dimension layar = Toolkit.getDefaultToolkit().getScreenSize();  // membuat titik x dan y  int x = layar.width / 2 - this.getSize().width / 2;  int y = layar.height / 2 - this.getSize().height / 2;  this.setLocation(x, y);  }  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jPanel1 = new javax.swing.JPanel();  title = new javax.swing.JLabel();  newgame = new javax.swing.JButton();  score = new javax.swing.JButton();  help = new javax.swing.JButton();  info = new javax.swing.JButton();  Background = new javax.swing.JLabel();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  setBackground(new java.awt.Color(167, 47, 109));  setPreferredSize(new java.awt.Dimension(500, 500));  jPanel1.setBackground(new java.awt.Color(255, 255, 255));  jPanel1.setLayout(null);  title.setFont(new java.awt.Font("Comic Sans MS", 1, 36)); // NOI18N  title.setForeground(new java.awt.Color(255, 255, 255));  title.setText(" MARBLE GUN");  title.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  jPanel1.add(title);  title.setBounds(90, 80, 310, 70);  newgame.setBackground(new java.awt.Color(255, 255, 255));  newgame.setFont(new java.awt.Font("Comic Sans MS", 1, 18)); // NOI18N  newgame.setText("NEW GAME");  newgame.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  newgame.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  newgameActionPerformed(evt);  }  });  jPanel1.add(newgame);  newgame.setBounds(170, 210, 160, 40);  score.setBackground(new java.awt.Color(255, 255, 255));  score.setFont(new java.awt.Font("Comic Sans MS", 1, 18)); // NOI18N  score.setText("HIGH SCORE");  score.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  score.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  scoreActionPerformed(evt);  }  });  jPanel1.add(score);  score.setBounds(170, 270, 160, 40);  help.setBackground(new java.awt.Color(255, 255, 255));  help.setFont(new java.awt.Font("Comic Sans MS", 1, 18)); // NOI18N  help.setText("HELP");  help.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  help.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  helpActionPerformed(evt);  }  });  jPanel1.add(help);  help.setBounds(170, 330, 160, 40);  info.setBackground(new java.awt.Color(255, 255, 255));  info.setFont(new java.awt.Font("Comic Sans MS", 1, 18)); // NOI18N  info.setText("INFO");  info.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  info.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  infoActionPerformed(evt);  }  });  jPanel1.add(info);  info.setBounds(170, 390, 160, 40);  Background.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/wallpaper3-jpg-500x500.jpg"))); // NOI18N  jPanel1.add(Background);  Background.setBounds(0, 0, 500, 500);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 503, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, Short.MAX\_VALUE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 502, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, Short.MAX\_VALUE))  );  pack();  }  private void newgameActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_newgameActionPerformed  // TODO add your handling code here:  EnterName entergame = new EnterName();  setVisible(false);  entergame.run();  }//GEN-LAST:event\_newgameActionPerformed  private void scoreActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_scoreActionPerformed  // TODO add your handling code here:  HighScore wy = new HighScore();  setVisible(false);  wy.gas();  }//GEN-LAST:event\_scoreActionPerformed  private void infoActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_infoActionPerformed  // TODO add your handling code here:  Info in = new Info();  setVisible(false);  in.gas();  }//GEN-LAST:event\_infoActionPerformed  private void helpActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_helpActionPerformed  // TODO add your handling code here:  Help he = new Help();  setVisible(false);  he.gas();  }  public static void main(String args[]) {  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  //</editor-fold>  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new Menu().setVisible(true);  }  });  }  private javax.swing.JLabel Background;  private javax.swing.JButton help;  private javax.swing.JButton info;  private javax.swing.JPanel jPanel1;  private javax.swing.JButton newgame;  private javax.swing.JButton score;  private javax.swing.JLabel title;  // End of variables declaration//GEN-END:variables  } |

1. ***Class Game Panel***

|  |
| --- |
| package marblegun;  import javax.swing.JPanel;  import java.awt.\*;  import java.awt.event.KeyEvent;  import java.awt.event.KeyListener;  import java.awt.image.\*;  import java.util.ArrayList;  import java.awt.event.\*;  import java.util.\*;  import java.awt.Dimension;  import java.awt.Toolkit;  import static marblegun.playMusic.playMusic;  public class GamePanel extends JPanel implements Runnable, KeyListener {  //fields  public static int WIDTH = 650;  public static int HEIGHT = 650;  private Thread thread;  private boolean running;  private BufferedImage image;  private Graphics2D g;  private int FPS = 60;  private double averageFPS;  public static Player player;  public static ArrayList<Bullet> bullets;  public static ArrayList<Enemy> enemies;  public static ArrayList<Text> texts;  private long waveStartTimer;  private long waveStartTimerDiff;  private int waveNumber;  private boolean waveStart;  private int waveDelay = 2000;  //insert db data  static String nama;  int scoreAkhir;  //  //Constructor  public GamePanel() {  super();  setPreferredSize(new Dimension(WIDTH, HEIGHT));  setFocusable(true);  requestFocus();  playMusic("/home/mereska/NetBeansProjects/MarbleGun/src/marblegun/music/musicbg.mp3");  }  //get set nama  public String getNama() {  return nama;  }  public void setNama(String nama) {  this.nama = nama;  }  //get set nama  //functions  public void addNotify() {  super.addNotify();  if (thread == null) {  thread = new Thread(this);  thread.start();  }  addKeyListener(this);  }  public void run() {  running = true;  image = new BufferedImage(WIDTH, HEIGHT, BufferedImage.TYPE\_INT\_RGB);  g = (Graphics2D) image.getGraphics();  g.setRenderingHint(  RenderingHints.KEY\_ANTIALIASING,  RenderingHints.VALUE\_ANTIALIAS\_ON);  g.setRenderingHint(  RenderingHints.KEY\_TEXT\_ANTIALIASING,  RenderingHints.VALUE\_TEXT\_ANTIALIAS\_ON);  player = new Player();  bullets = new ArrayList<Bullet>();  enemies = new ArrayList<Enemy>();  texts = new ArrayList<Text>();  waveStartTimer = 0;  waveStartTimerDiff = 0;  waveStart = true;  waveNumber = 0;  long startTime;  long URDTimeMillis;  long waitTime;  long totalTime = 0;  int frameCount = 0;  int maxFrameCount = 60;  long targetTime = 1000 / FPS;  //game loop  while (running) {  startTime = System.nanoTime();  gameUpdate();  gameRender();  gameDraw();  URDTimeMillis = (System.nanoTime() - startTime) / 1000000;  waitTime = targetTime - URDTimeMillis;  try {  Thread.sleep(waitTime);  } catch (Exception e) {  }  totalTime += System.nanoTime() - startTime;  frameCount++;  if (frameCount == maxFrameCount) {  averageFPS = 1000.0 / ((totalTime / frameCount) / 1000000);  frameCount = 0;  totalTime = 0;  }  }  g.setColor(new Color(0, 100, 255));  g.fillRect(0, 0, WIDTH, HEIGHT);  g.setColor(Color.WHITE);  g.setFont(new Font("Century Gothic", Font.PLAIN, 16));  String s = "G A M E O V E R";  int length = (int) g.getFontMetrics().getStringBounds(s, g).getWidth();  g.drawString(s, (WIDTH - length) / 2, HEIGHT / 2);  g.drawString(nama, (WIDTH - length) / 2, HEIGHT / 2 + 50);  s = "Final Score : " + player.getScore();  length = (int) g.getFontMetrics().getStringBounds(s, g).getWidth();  g.drawString(s, (WIDTH - length + 10) / 2, HEIGHT / 2 + 30);  gameDraw();  }  private void gameUpdate() {  //new wave  if (waveStartTimer == 0 && enemies.size() == 0) {  waveNumber++;  waveStart = false;  waveStartTimer = System.nanoTime();  } else {  waveStartTimerDiff = (System.nanoTime() - waveStartTimer) / 1000000;  if (waveStartTimerDiff > waveDelay) {  waveStart = true;  waveStartTimer = 0;  waveStartTimerDiff = 0;  }  }  //create enemies  if (waveStart && enemies.size() == 0) {  createNewEnemies();  }  //player update  player.update();  //bullet update  for (int i = 0; i < bullets.size(); i++) {  boolean remove = bullets.get(i).update();  if (remove) {  bullets.remove(i);  i--;  }  }  // enemy update  for (int i = 0; i < enemies.size(); i++) {  enemies.get(i).update();  }  //text update  for (int i = 0; i < texts.size(); i++) {  boolean remove = texts.get(i).update();  if (remove) {  texts.remove(i);  i--;  }  }  //bullet-enemy collision  for (int i = 0; i < bullets.size(); i++) {  Bullet b = bullets.get(i);  double bx = b.getx();  double by = b.gety();  double br = b.getr();  for (int j = 0; j < enemies.size(); j++) {  Enemy e = enemies.get(j);  double ex = e.getx();  double ey = e.gety();  double er = e.getr();  double dx = bx - ex;  double dy = by - ey;  double dist = Math.sqrt(dx \* dx + dy \* dy);  if (dist < br + er) {  e.hit();  bullets.remove(i);  i--;  break;  }  }  }  // check dead enemies  for (int i = 0; i < enemies.size(); i++) {  if (enemies.get(i).isDead()) {  Enemy e = enemies.get(i);  player.addScore(e.getType() + e.getRank());  enemies.remove(i);  i--;  }  }  //check dead player  if (player.isDead()) {  //kalo mati  Player pemain = new Player();  scoreAkhir = player.getScore();  String namePlayer = nama;  ///insert data  koneksi yz = new koneksi();  yz.KoneksiDB();  yz.push(namePlayer, scoreAkhir);  ///batas insert  //batas mati  running = false;  }  //player-enemy collision  if (!player.isRecovering()) {  int px = player.getx();  int py = player.gety();  int pr = player.getr();  for (int i = 0; i < enemies.size(); i++) {  Enemy e = enemies.get(i);  double ex = e.getx();  double ey = e.gety();  double er = e.getr();  double dx = px - ex;  double dy = py - ey;  double dist = Math.sqrt(dx \* dx + dy \* dy);  if (dist < pr + er) {  player.loseLife();  }  }  }  }  private void gameRender() {  //draw background  g.setColor(new Color(0, 100, 255));  g.fillRect(0, 0, WIDTH, HEIGHT);  //draw player  player.draw(g);  //draw bullet  for (int i = 0; i < bullets.size(); i++) {  bullets.get(i).draw(g);  }  //draw enemy  for (int i = 0; i < enemies.size(); i++) {  enemies.get(i).draw(g);  }  //draw text  for (int i = 0; i < texts.size(); i++) {  texts.get(i).draw(g);  }  //draw wave number  if (waveStartTimer != 0) {  g.setFont(new Font("Century Gothic", Font.PLAIN, 18));  String s = "- W A V E " + waveNumber + " -";  int length = (int) g.getFontMetrics().getStringBounds(s, g).getWidth();  int alpha = (int) (255 \* Math.sin(3.14 \* waveStartTimerDiff / waveDelay));  if (alpha > 255) {  alpha = 255;  }  g.setColor(new Color(255, 255, 255, alpha));  g.drawString(s, WIDTH / 2 - length / 2, HEIGHT / 2);  }  // draw player lives  for (int i = 0; i < player.getLives(); i++) {  g.setColor(Color.WHITE);  g.fillOval(20 + (20 \* i), 20, player.getr() \* 2, player.getr() \* 2);  g.setStroke(new BasicStroke(3));  g.setColor(Color.WHITE.darker());  g.drawOval(20 + (20 \* i), 20, player.getr() \* 2, player.getr() \* 2);  g.setStroke(new BasicStroke(1));  }  // draw player score  g.setColor(Color.WHITE);  g.setFont(new Font("Century Gothic", Font.PLAIN, 14));  g.drawString("score: " + player.getScore(), WIDTH - 100, 30);  }  private void gameDraw() {  Graphics g2 = this.getGraphics();  g2.drawImage(image, 0, 0, null);  g2.dispose();  }  private void createNewEnemies() {  enemies.clear();  Enemy e;  if (waveNumber == 1) {  for (int i = 0; i < 4; i++) {  enemies.add(new Enemy(1, 1));  }  }  if (waveNumber == 2) {  for (int i = 0; i < 8; i++) {  enemies.add(new Enemy(1, 1));  }  }  if (waveNumber == 3) {  for (int i = 0; i < 12; i++) {  enemies.add(new Enemy(1, 1));  }  enemies.add(new Enemy(2, 1));  }  if (waveNumber == 4) {  for (int i = 0; i < 16; i++) {  enemies.add(new Enemy(2, 1));  }  }  if (waveNumber == 5) {  for (int i = 0; i < 20; i++) {  enemies.add(new Enemy(2, 1));  }  }  if (waveNumber == 6) {  for (int i = 0; i < 24; i++) {  enemies.add(new Enemy(2, 1));  }  enemies.add(new Enemy(3, 1));  }  if (waveNumber == 7) {  for (int i = 0; i < 30; i++) {  enemies.add(new Enemy(3, 1));  }  }  if (waveNumber == 8) {  for (int i = 0; i < 34; i++) {  enemies.add(new Enemy(3, 1));  }  }  if (waveNumber == 9) {  for (int i = 0; i < 38; i++) {  enemies.add(new Enemy(3, 1));  }  }  if (waveNumber == 10) {  for (int i = 0; i < 50; i++) {  enemies.add(new Enemy(3, 1));  }  }  if (waveNumber == 11) {  running = false;  }  public void keyTyped(KeyEvent key) {  }  public void keyPressed(KeyEvent key) {  int keyCode = key.getKeyCode();  if (keyCode == KeyEvent.VK\_LEFT) {  player.setLeft(true);  }  if (keyCode == KeyEvent.VK\_RIGHT) {  player.setRight(true);  }  if (keyCode == KeyEvent.VK\_UP) {  player.setUp(true);  }  if (keyCode == KeyEvent.VK\_DOWN) {  player.setDown(true);  }  if (keyCode == KeyEvent.VK\_Z) {  player.setFiring(true);  }  }  public void keyReleased(KeyEvent key) {  int keyCode = key.getKeyCode();  if (keyCode == KeyEvent.VK\_LEFT) {  player.setLeft(false);  }  if (keyCode == KeyEvent.VK\_RIGHT) {  player.setRight(false);  }  if (keyCode == KeyEvent.VK\_UP) {  player.setUp(false);  }  if (keyCode == KeyEvent.VK\_DOWN) {  player.setDown(false);  }  if (keyCode == KeyEvent.VK\_Z) {  player.setFiring(false);  }  }  } |

1. ***Class Game***

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  import javax.swing.JFrame;  public class Game extends javax.swing.JFrame {  void run() {  JFrame window = new JFrame("MARBLE GUN");  window.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  window.setContentPane(new GamePanel());  window.pack();  window.setLocationRelativeTo(null);// solved by : https://stackoverflow.com/questions/2442599/how-to-set-jframe-to-appear-centered-regardless-of-monitor-resolution  window.setVisible(true);  // membuat titik x dan y  Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();  int x = dim.width / 2 - this.getSize().width / 2;  int y = dim.height / 2 - this.getSize().height / 2;  this.setLocation(x,y);  }  public static void main(String[] args) {  }  } |

1. ***Class Enemy***

|  |
| --- |
| package marblegun;  import java.awt.\*;  import java.awt.image.BufferedImage;  public class Enemy extends Sub implements Aktor{  //fields    private int health ;  private final int type ;  private final int rank ;  private Color color1 ;  private boolean ready ;  private boolean dead ;    private BufferedImage image ;  //constructor  public Enemy (int type , int rank) {  this.type = type ;  this.rank = rank ;  //default enemy  if (type==1){  color1 = Color.BLUE ;  image = Gambar.getResourceImage("musuh.png");  if (rank == 1){  speed = 1 ;  r = 5 ;  health = 1 ;  }  }  else if (type==2){  color1 = Color.YELLOW ;  image = Gambar.getResourceImage("musuh2.png");  if (rank == 1){  speed = 2 ;  r = 5 ;  health = 3 ;  }  }  else if (type==3){  color1 = Color.RED ;  image = Gambar.getResourceImage("musuh1.png");  if (rank == 1){  speed = 10 ;  r = 5 ;  health = 5 ;  }  }  x = Math.random() \* GamePanel.WIDTH / 2 + GamePanel.WIDTH / 4 ;  y = -r ;  double angle = Math.random() \* 140 + 20 ;  rad = Math.toRadians(angle) ;  dx = Math.cos(rad) \* speed ;  dy = Math.sin(rad) \* speed ;  ready = false ;  dead = false ;  }  @Override  public double getx() {  return x;  }  @Override  public double gety() {  return y;  }  @Override  public double getr() {  return r;  }  public int getType(){  return type ;  }  public int getRank(){  return rank ;  }  @Override  public boolean isDead() {  return dead ;  }  public void hit(){  health -- ;  if (health <= 0){  dead = true ;  }  }  @Override  public void update (){  x += dx ;  y += dy ;  if (!ready){  if (x > r && x < GamePanel.WIDTH - r &&  y > r && y < GamePanel.HEIGHT -r ){  ready = true ;  }  }  if (x < r && dx < 0) dx = -dx ;  if (y < r && dy < 0) dy = -dy ;  if (x > GamePanel.WIDTH - r && dx > 0) dx = -dx ;  if (y > GamePanel.HEIGHT - r && dy > 0) dy = -dy ;  }  @Override  public void draw (Graphics2D g){  g.setColor(color1) ;  g.fillOval((int) (x - r), (int) (y - r), 2 \* r , 2 \* r);  g.setStroke(new BasicStroke(3));  g.setColor(color1.darker());  g.drawImage(image, (int)(x - r), (int)(y - r), null) ;  g.setStroke(new BasicStroke(1));  }  } |

1. ***Class Bullet***

|  |
| --- |
| package marblegun;  import java.awt.\*;  import java.awt.image.BufferedImage;  public class Bullet extends Sub{  private Color color1 ;  private BufferedImage image ;  //constructor  public Bullet (double angle, double x, double y) {  this.x = x;  this.y = y;  r = 5 ;  image = Gambar.getResourceImage("peluru.png");  rad = Math.toRadians(angle) ;  speed = 10 ;  dx = Math.cos(rad) \* speed ;  dy = Math.sin(rad) \* speed;  color1 = Color.YELLOW ;  }  @Override  public double getx() {  return x;  }  @Override  public double gety() {  return y;  }  @Override  public double getr() {  return r;  }  public boolean update (){  x += dx ;  y += dy ;  return x < -r || x > GamePanel.WIDTH + r ||  y < -r || y > GamePanel.HEIGHT + r ;  }  @Override  public void draw (Graphics2D g){  g.setColor(color1) ;  g.drawImage(image, (int)(x - r), (int)(y - r), null) ;  }  } |

1. ***Class High Score***

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  import java.sql.Connection;  import java.sql.ResultSet;  import java.sql.Statement;  import javax.swing.table.DefaultTableModel;  public class HighScore extends javax.swing.JFrame {  Connection con;  Statement stat;  static ResultSet rs;  String sql;  public DefaultTableModel model;  public HighScore() {  initComponents();  // mengambil ukuran layar  Dimension layar = Toolkit.getDefaultToolkit().getScreenSize();  // membuat titik x dan y  int x = layar.width / 2 - this.getSize().width / 2;  int y = layar.height / 2 - this.getSize().height / 2;  this.setLocation(x, y);  //database  String [] header = {"Nama","Score"};  model = new DefaultTableModel (header,0);  tabelScore.setModel(model);  koneksi yz = new koneksi();  yz.KoneksiDB();  tampil(); //baru tampilin  }    public void tampil() {  //konek db  con = koneksi.highscore;  stat = koneksi.query;  sql = "SELECT \* FROM player INNER JOIN score on player.Id\_player = score.Id\_player ORDER by SCORE DESC";  try {  rs = stat.executeQuery(sql);  while(rs.next()) {  String [] row = {rs.getString(2), rs.getString(4)};  model.addRow(row);  }  tabelScore.setModel(model);    } catch (Exception e){  e.printStackTrace();  }  }  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jPanel2 = new javax.swing.JPanel();  jScrollPane1 = new javax.swing.JScrollPane();  tabelScore = new javax.swing.JTable();  jLabel1 = new javax.swing.JLabel();  jButton1 = new javax.swing.JButton();  Background = new javax.swing.JLabel();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  setBackground(new java.awt.Color(26, 211, 253));  jPanel2.setLayout(null);  tabelScore.setModel(new javax.swing.table.DefaultTableModel(  new Object [][] {  {null, null},  {null, null},  {null, null},  {null, null},  {null, null},  {null, null},  {null, null}  },  new String [] {  "Nama", "Score"  }  ));  jScrollPane1.setViewportView(tabelScore);  jPanel2.add(jScrollPane1);  jScrollPane1.setBounds(50, 210, 390, 170);  jLabel1.setFont(new java.awt.Font("Comic Sans MS", 1, 36)); // NOI18N  jLabel1.setForeground(new java.awt.Color(255, 255, 255));  jLabel1.setText(" HIGH SCORE");  jLabel1.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));  jPanel2.add(jLabel1);  jLabel1.setBounds(120, 80, 270, 70);  jButton1.setFont(new java.awt.Font("Comic Sans MS", 1, 18)); // NOI18N  jButton1.setForeground(new java.awt.Color(255, 255, 255));  jButton1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/left-curve-arrow (1).png"))); // NOI18N  jButton1.setToolTipText("");  jButton1.setBorderPainted(false);  jButton1.setContentAreaFilled(false);  jButton1.setIconTextGap(10);  jButton1.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  jButton1ActionPerformed(evt);  }  });  jPanel2.add(jButton1);  jButton1.setBounds(0, 430, 90, 76);  Background.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/wallpaper3-jpg-500x500.jpg"))); // NOI18N  jPanel2.add(Background);  Background.setBounds(0, 0, 500, 500);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, 500, Short.MAX\_VALUE)  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, 500, Short.MAX\_VALUE)  );  pack();  }// </editor-fold>//GEN-END:initComponents  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed  // TODO add your handling code here:  new Menu().setVisible(true);  setVisible(false);  }  void gas () {  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new HighScore().setVisible(true);  }  });  }  public static void main(String args[]) {  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(HighScore.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(HighScore.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(HighScore.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(HighScore.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  }  private javax.swing.JLabel Background;  private javax.swing.JButton jButton1;  private javax.swing.JLabel jLabel1;  private javax.swing.JPanel jPanel2;  private javax.swing.JScrollPane jScrollPane1;  private javax.swing.JTable tabelScore;  } |

1. ***Class Insert Name***

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  public class EnterName extends javax.swing.JFrame {  public EnterName() {  initComponents();  // mengambil ukuran layar  Dimension layar = Toolkit.getDefaultToolkit().getScreenSize();  // membuat titik x dan y  int x = layar.width / 2 - this.getSize().width / 2;  int y = layar.height / 2 - this.getSize().height / 2;  this.setLocation(x, y);  }  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jPanel1 = new javax.swing.JPanel();  jLabel1 = new javax.swing.JLabel();  fieldName = new javax.swing.JTextField();  PLAY = new javax.swing.JButton();  BACK = new javax.swing.JButton();  background = new javax.swing.JLabel() setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  jPanel1.setPreferredSize(new java.awt.Dimension(500, 500));  jPanel1.setLayout(null);  jLabel1.setFont(new java.awt.Font("Comic Sans MS", 1, 36)); // NOI18N  jLabel1.setForeground(new java.awt.Color(255, 255, 255));  jLabel1.setText(" INSERT NAME");  jLabel1.setBorder(new javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED));  jPanel1.add(jLabel1);  jLabel1.setBounds(90, 40, 310, 60);  jPanel1.add(fieldName);  fieldName.setBounds(90, 170, 310, 50);  PLAY.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N  PLAY.setText("PLAY");  PLAY.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  PLAYActionPerformed(evt);  }  });  jPanel1.add(PLAY);  PLAY.setBounds(290, 290, 110, 50);  BACK.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N  BACK.setText("BACK ");  BACK.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  BACKActionPerformed(evt);  }  });  jPanel1.add(BACK);  BACK.setBounds(90, 290, 110, 50);  background.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/wallpaper3-jpg-500x500.jpg"))); // NOI18N  jPanel1.add(background);  background.setBounds(0, 0, 500, 500);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()  .addGap(0, 0, Short.MAX\_VALUE)  .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel1, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  );  pack();  }// </editor-fold>//GEN-END:initComponents  private void PLAYActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_PLAYActionPerformed  // TODO add your handling code here:  Game xy = new Game();  GamePanel yz = new GamePanel(); //solved pakai static  yz.setNama(fieldName.getText().toString()); //solved pakai static  setVisible(false);  xy.run();    }//GEN-LAST:event\_PLAYActionPerformed  private void BACKActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_BACKActionPerformed  // TODO add your handling code here:  new Menu().setVisible(true);  setVisible(false);  }  void run () {  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new EnterName().setVisible(true);  }  });  }  public static void main(String args[]) {    try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(EnterName.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(EnterName.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(EnterName.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex){  java.util.logging.Logger.getLogger(EnterName.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  }  private javax.swing.JButton BACK;  private javax.swing.JButton PLAY;  private javax.swing.JLabel background;  private javax.swing.JTextField fieldName;  private javax.swing.JLabel jLabel1;  private javax.swing.JPanel jPanel1;  // End of variables declaration//GEN-END:variables  } |

1. ***Class Player***

|  |
| --- |
| package marblegun;  import java.awt.\*;  import java.awt.image.BufferedImage;  import java.awt.image.ImageObserver;  import java.io.File;  import java.io.IOException;  import javax.swing.ImageIcon;  import javax.imageio.ImageIO;  public class Player implements Aktor {  private int x;  private int y;  private final int r ;  private double dx ;  private double dy ;  private final double speed ;  private boolean left ;  private boolean right ;  private boolean up ;  private boolean down ;  private boolean firing ;  private long firingTimer ;  private final long firingDelay ;  private boolean recovering ;  private long recoveryTimer ;  private int lives ;  private final Color color1 ;  private final Color color2 ;  private int score ;  private BufferedImage image ;  private BufferedImage image1 ;    public Player() {  image = Gambar.getResourceImage("karakter.png");  image1 = Gambar.getResourceImage("karakter1.png");    x = GamePanel.WIDTH / 2 ;  y = GamePanel.HEIGHT / 2 ;  r = 10 ;  dx = 10;  dy = 5 ;  speed = 3;  lives = 3 ;  color1 = Color.WHITE ;  color2 = Color.RED ;  firing = false ;  firingTimer = System.nanoTime() ;  firingDelay = 200 ;  recovering = false ;  recoveryTimer = 0 ;  score = 0 ;  }  public int getx() {  return x ;  }  public int gety() {  return y ;  }  public int getr() {  return r ;  }  public int getScore(){  return score ;  }  public int getLives(){  return lives ;  }  @Override  public boolean isDead(){  return lives <= 0 ;  }  public boolean isRecovering(){  return recovering ;  }  public void setLeft(boolean b) {  left = b;  }  public void setRight(boolean b) {  right = b;  }  public void setUp(boolean b) {  up = b;  }  public void setDown(boolean b) {  down = b;  }  public void setFiring(boolean b){  firing = b ;  }  public void addScore(int i){  score += i ;  }  public void loseLife(){  lives --;  recovering = true ;  recoveryTimer = System.nanoTime() ;  }  @Override  public void update() {  if (left){  dx = -speed ;  }  if (right){  dx = speed ;  }  if (up){  dy = -speed ;  }  if (down){  dy = speed ;  }  x += dx ;  y += dy ;  if (x < r) x = r ;  if (y < r) y = r ;  if (x > GamePanel.WIDTH - r) x = GamePanel.WIDTH - r ;  if (y > GamePanel.HEIGHT - r) y = GamePanel.HEIGHT - r ;  dx = 0 ;  dy = 0 ;  if(firing){  long elapsed = (System.nanoTime()-firingTimer)/1000000 ;  if(elapsed > firingDelay) {  GamePanel.bullets.add(new Bullet (270, x, y)) ;  firingTimer = System.nanoTime() ;  }  }  long elapsed = (System.nanoTime() - recoveryTimer) / 1000000 ;  if(elapsed > 2000){  recovering = false ;  recoveryTimer = 0 ;  }  }  @Override  public void draw (Graphics2D g) {  if (recovering){  g.setColor(color2) ;  //g.fillOval(x - r, y - r, 2 \* r, 2 \* r) ;    g.setStroke(new BasicStroke(3));  g.setColor(color2.darker()) ;  g.drawImage(image1, x - r, y - r,25,25, null) ;    g.setStroke(new BasicStroke(1)) ;  }  else {    g.setColor(color1) ;  //g.fillOval(x - r, y - r, 2 \* r, 2 \* r) ;    g.setStroke(new BasicStroke(3));  g.setColor(color1.darker()) ;      g.drawImage(image, x - r, y - r,25,25, null) ;  g.setStroke(new BasicStroke(1)) ;  }  }  } |

1. ***Class Teks***

|  |
| --- |
| package marblegun;  import java.awt.\*;  public class Text{  private double x ;  private double y ;  private long time ;  private String s ;  private long start ;  //constructor  public Text(double x, double y, long time, String s){  this.x = x ;  this.y = y ;  this.time = time ;  this.s = s ;  start = System.nanoTime() ;  }  public boolean update() {  long elapsed = (System.nanoTime() - start) / 1000000 ;  if (elapsed > time){  return true ;  }  return false ;  }  public void draw (Graphics2D g){  g.setFont(new Font("Century Gothic", Font.PLAIN, 12)) ;  long elapsed = (System.nanoTime() - start) / 1000000 ;  int alpha = (int) (255 \* Math.sin(3.14 \* elapsed / time));  if (alpha > 255) alpha = 255 ;  g.setColor(new Color(255, 255, 255, alpha)) ;  int length = (int) g.getFontMetrics().getStringBounds(s, g).getWidth() ;  g.drawString(s, (int) (x - (length / 2)), (int) y) ;  }  } |

1. ***Class Help***

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  import javax.swing.JFrame;  public class Help extends javax.swing.JFrame {  public Help() {  initComponents();  setResizable(false);  Dimension layar = Toolkit.getDefaultToolkit().getScreenSize();  // membuat titik x dan y  int x = layar.width / 2 - this.getSize().width / 2;  int y = layar.height / 2 - this.getSize().height / 2;  this.setLocation(x, y);  }  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jPanel1 = new javax.swing.JPanel();  jLabel1 = new javax.swing.JLabel();  OK = new javax.swing.JButton();  jScrollPane1 = new javax.swing.JScrollPane();  jTextArea1 = new javax.swing.JTextArea();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  setBackground(new java.awt.Color(0, 0, 0));  jPanel1.setBackground(new java.awt.Color(0, 0, 0));  jPanel1.setLayout(null);  jLabel1.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N  jLabel1.setForeground(new java.awt.Color(255, 255, 255));  jLabel1.setText("HOW TO PLAY");  jPanel1.add(jLabel1);  jLabel1.setBounds(110, 0, 190, 60);  OK.setText("OK");  OK.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  OKActionPerformed(evt);  }  });  jPanel1.add(OK);  OK.setBounds(170, 250, 50, 23);  jScrollPane1.setBackground(new java.awt.Color(0, 0, 0));  jTextArea1.setBackground(new java.awt.Color(0, 0, 0));  jTextArea1.setColumns(20);  jTextArea1.setForeground(javax.swing.UIManager.getDefaults().getColor("Button.background"));  jTextArea1.setRows(5);  jTextArea1.setText("THE FIRST STEP KLIK NEW PLAY, AND INSERT\nNAME, AFTER THAN PLAY TO GAME. \n");  jTextArea1.setAutoscrolls(false);  jTextArea1.setBorder(null);  jTextArea1.setCaretColor(new java.awt.Color(255, 255, 255));  jTextArea1.setDisabledTextColor(new java.awt.Color(255, 255, 255));  jScrollPane1.setViewportView(jTextArea1);  jPanel1.add(jScrollPane1);  jScrollPane1.setBounds(40, 60, 330, 180);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 400, Short.MAX\_VALUE)  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 300, Short.MAX\_VALUE)  );  pack();  }// </editor-fold>//GEN-END:initComponents  private void OKActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_OKActionPerformed  // TODO add your handling code here:  new Menu().setVisible(true);  setVisible(false);  }//GEN-LAST:event\_OKActionPerformed  void gas(){  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new Help().setVisible(true);  }  });  }  public static void main(String args[]) {  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(Help.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(Help.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(Help.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(Help.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  }  private javax.swing.JButton OK;  private javax.swing.JLabel jLabel1;  private javax.swing.JPanel jPanel1;  private javax.swing.JScrollPane jScrollPane1;  private javax.swing.JTextArea jTextArea1;  } |

1. ***Class* Info**

|  |
| --- |
| package marblegun;  import java.awt.Dimension;  import java.awt.Toolkit;  import javax.swing.JFrame;  public class Info extends javax.swing.JFrame {  public Info() {  setResizable(false);  Dimension layar = Toolkit.getDefaultToolkit().getScreenSize();  // membuat titik x dan y  int x = layar.width / 2 - this.getSize().width / 2;  int y = layar.height / 2 - this.getSize().height / 2;  this.setLocation(x, y);  }  @SuppressWarnings("unchecked")    private void initComponents() {  jPanel1 = new javax.swing.JPanel();  jButton1 = new javax.swing.JButton();  jLabel2 = new javax.swing.JLabel();  jLabel3 = new javax.swing.JLabel();  jLabel4 = new javax.swing.JLabel();  jLabel5 = new javax.swing.JLabel();  jLabel6 = new javax.swing.JLabel();  jLabel7 = new javax.swing.JLabel();  jButton2 = new javax.swing.JButton();  OK = new javax.swing.JButton();  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  jPanel1.setBackground(new java.awt.Color(0, 0, 0));  jPanel1.setLayout(null);  jButton1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/WIND\_\_1575272841\_41875.png"))); // NOI18N  jButton1.setContentAreaFilled(false);  jButton1.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  jButton1ActionPerformed(evt);  }  });  jPanel1.add(jButton1);  jButton1.setBounds(300, 220, 80, 60);  jLabel2.setFont(new java.awt.Font("Comic Sans MS", 1, 15)); // NOI18N  jLabel2.setForeground(new java.awt.Color(51, 255, 255));  jLabel2.setText("MELISA N.S");  jPanel1.add(jLabel2);  jLabel2.setBounds(110, 140, 210, 30);  jLabel3.setFont(new java.awt.Font("Comic Sans MS", 1, 15)); // NOI18N  jLabel3.setForeground(new java.awt.Color(51, 255, 255));  jLabel3.setText("I KETUT NADI ANGGARA");  jPanel1.add(jLabel3);  jLabel3.setBounds(110, 110, 210, 30);  jLabel4.setFont(new java.awt.Font("Comic Sans MS", 1, 15)); // NOI18N  jLabel4.setForeground(new java.awt.Color(51, 255, 255));  jLabel4.setText("MUHAMMAD ILHAN J");  jPanel1.add(jLabel4);  jLabel4.setBounds(110, 170, 210, 30);  jLabel5.setFont(new java.awt.Font("Comic Sans MS", 1, 15)); // NOI18N  jLabel5.setForeground(new java.awt.Color(51, 255, 255));  jLabel5.setText("MUHAMMAD NAUFAL R");  jPanel1.add(jLabel5);  jLabel5.setBounds(110, 200, 210, 30);  jLabel6.setFont(new java.awt.Font("Chiller", 1, 34)); // NOI18N  jLabel6.setForeground(new java.awt.Color(255, 255, 255));  jLabel6.setText("THE LEGEND OF APPA");  jPanel1.add(jLabel6);  jLabel6.setBounds(100, 40, 290, 30);  jLabel7.setFont(new java.awt.Font("Comic Sans MS", 1, 15)); // NOI18N  jLabel7.setForeground(new java.awt.Color(51, 255, 255));  jLabel7.setText("DEFRIALDY");  jPanel1.add(jLabel7);  jLabel7.setBounds(110, 80, 210, 30);  jButton2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/Appa\_\_1575272544\_74888\_\_1575272544\_32867.png"))); // NOI18N  jButton2.setContentAreaFilled(false);  jButton2.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  jButton2ActionPerformed(evt);  }  });  jPanel1.add(jButton2);  jButton2.setBounds(0, 10, 100, 110);  OK.setText("OK");  OK.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  OKActionPerformed(evt);  }  });  jPanel1.add(OK);  OK.setBounds(160, 260, 60, 29);  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 400, Short.MAX\_VALUE)  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 300, Short.MAX\_VALUE)  );  pack();  }  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  this.dispose();  }  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  }  private void OKActionPerformed(java.awt.event.ActionEvent evt) {  new Menu().setVisible(true);  setVisible(false);  }  void gas (){  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new Info().setVisible(true);  }  });  }  public static void main(String args[]) {  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(Info.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(Info.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(Info.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) { java.util.logging.Logger.getLogger(Info.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  }  private javax.swing.JButton OK;  private javax.swing.JButton jButton1;  private javax.swing.JButton jButton2;  private javax.swing.JLabel jLabel2;  private javax.swing.JLabel jLabel3;  private javax.swing.JLabel jLabel4;  private javax.swing.JLabel jLabel5;  private javax.swing.JLabel jLabel6;  private javax.swing.JLabel jLabel7;  private javax.swing.JPanel jPanel1;  } |

1. ***Class* Sub**

|  |
| --- |
| package marblegun;  import java.awt.Graphics2D;  public abstract class Sub {  public double x ;  public double y ;  public int r ;  public double dx ;  public double dy ;  public double rad ;  public double speed ;    public abstract double gety();  public abstract double getx();  public abstract double getr();  public abstract void draw(Graphics2D g);  } |

1. ***Class* Aktor**

|  |
| --- |
| package marblegun;  import java.awt.Graphics2D;  public interface Aktor {  public abstract boolean isDead();  public abstract void update ();  public abstract void draw(Graphics2D g);  } |

1. ***Class Play* Musik**

|  |
| --- |
| package marblegun;  import java.io.File;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.InputStream;  import javax.swing.JOptionPane;  import sun.audio.AudioPlayer;  import sun.audio.AudioStream;  public class playMusic {  public static void main(String[] args) {  }  public static void playMusic(String filepath) {  InputStream music;  try {  music = new FileInputStream(new File(filepath));  AudioStream audios = new AudioStream(music);  AudioPlayer.player.start(audios);  }  catch (IOException e) {  JOptionPane.showMessageDialog(null, "Error");  }  }  } |

1. ***Class Image***

|  |
| --- |
| package marblegun;  import java.awt.image.BufferedImage ;  import java.io.File;  import java.io.IOException;  import javax.imageio.ImageIO;  public class Image {  public static BufferedImage getResourceBufferedImage (String patch){  BufferedImage img = null ;  try {  img = ImageIO.read(new File(patch)) ;  }  catch (IOException ex){  ex.printStackTrace() ;  }  return img ;  }  } |

1. ***Class* Gambar**

|  |
| --- |
| package marblegun;  import java.awt.image.BufferedImage;  import java.io.File;  import java.io.IOException;  import javax.imageio.ImageIO;  public class Gambar{  public static BufferedImage getResourceImage(String path){  BufferedImage img=null;  try {  img=ImageIO.read(new File(path));  } catch (IOException ex) {  ex.printStackTrace();  }return img;  }  } |

1. ***Class* Koneksi**

|  |
| --- |
| package marblegun;  import java.sql.\*;  import java.sql.DriverManager;  import java.sql.Connection;  import java.sql.Statement;  import java.sql.ResultSet;  import javax.swing.JOptionPane;  import java.sql.PreparedStatement;  import java.util.logging.Level;  import java.util.logging.Logger;  public class koneksi {  public static Connection highscore;  public static Statement query;  String sql;  String sql2;  String sql3;  static ResultSet rs;  int id;  public void push(String nama, int score) {  sql = "INSERT INTO player (Nama) VALUES ('" + nama + "')";  sql2 = "SELECT Id\_player FROM player WHERE Nama = '" + nama + "'";    try {  PreparedStatement stmt = highscore.prepareStatement(sql);  stmt.execute();  PreparedStatement stmt2 = highscore.prepareStatement(sql2);  stmt2.execute();  //ambil data  rs = query.executeQuery(sql2);  if (rs.next()) {  id = rs.getInt(1);  }  sql3 = "INSERT INTO score (Score,Id\_player) VALUES (" + score + "," + id + ")";  PreparedStatement stmt3 = highscore.prepareStatement(sql3);  stmt3.executeUpdate();      } catch (SQLException ex) {  System.err.format("SQL State: %s\n%s", ex.getSQLState(), ex.getMessage());  }  }  public void KoneksiDB() {  try {  String DB = "jdbc:mysql://localhost/game"; // delta\_db database  String user = "root"; // user database  String pass = ""; // password database  Class.forName("com.mysql.jdbc.Driver");  highscore = DriverManager.getConnection(DB, user, pass);  query = highscore.createStatement();  } catch (Exception e) {  JOptionPane.showMessageDialog(null, ("gagal koneksi" + e.getMessage()));  }  }  } |