ZIEN CODE - COPYRIGHT OF AKIB SHAHJAHAN

Assets.java

```
import com.akibshahjahan.framework.Image;
import com.akibshahjahan.framework.Music;
import com.akibshahjahan.framework.Sound;

public class Assets
{
	public static Image splash, image, backgroundImg, gameOverImg, zienImg,
tapeImg,cutTapeImg, barImg, restartImg, playAgainImg, pauseImg, resumeImg,
tapToPlayImg, centuryClubImg;
    public static Sound beepSound, endSound;
    public static Music music;

    public static void load(SampleGame sampleGame)
    {
        music = sampleGame.getAudio().createMusic("music.wav");
        music.setLooping(true);
        music.setVolume(0.45f);
        music.play();
    }
}
```

Bar.java

```
public class Bar
{
	private int barX = 550;
	private int barY, width, height;

	public Bar(int y, int w, int h)
	{
		barY = y;
		width = w;
		height = h;
	}

	public int getBarX()
	{
		return barX;
	}

	public int getBarY()
	{
		return barY;
	}

	public int getWidth()
	{
```

```
		return width;
	}

	public int getHeight()
	{
		return height;
	}

	public void setBarX(int barX)
	{
		this.barX = barX;
	}

	public void setBarY(int barY)
	{
		this.barY = barY;
	}

	public void setWidth(int width)
	{
		this.width = width;
	}

	public void setHeight(int height)
	{
		this.height = height;
	}
}
```

CutTape.java

```
import android.graphics.Rect;

public class CutTape
{
	private int tapeY;
	private int tapeX = 550;
	private int speedY = -5;
	private int height;
	private int width;
	private boolean scoreOn = true;
	public Rect rect = new Rect(0,0,0,0);
	private Zien z = GameScreen.getZien();

	public CutTape(int y, int h, int w)
	{
		tapeY = y;
		height = h;
		width = w;
		rect = new Rect();
	}
```

```
	public void update()
	{
		tapeY += speedY;

		if (tapeY <= -height)
		{
			tapeY = height*5 ;
					;
		}

		rect.set((int)(tapeX+width/4)-18, tapeY + 80, (int)(tapeX+width/4)-18+(int)
(width/2), (tapeY+80)+76);
		if(Rect.intersects(rect, z.rect)&&scoreOn)
		{
			GameScreen.setScore();
			GameScreen.setBar(true);
			GameScreen.makeSound(1);
		}
	}

	public int getTapeX()
	{
		return tapeX;
	}

	public int getTapeY()
	{
		return tapeY;
	}

	public int getSpeedY()
	{
		return speedY;
	}

	public void setTapeX(int tapeX)
	{
		this.tapeX = tapeX;
	}

	public void setTapeY(int tapeY)
	{
		this.tapeY = tapeY;
	}

	public void setSpeedY(int speedY)
	{
		this.speedY = speedY;
	}

	public void setScoreOn(boolean b)
	{
```

```
                scoreOn = b;
        }
}
```

GameScreen.java

```
import java.util.List;

import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Typeface;

import com.akibshahjahan.framework.Game;
import com.akibshahjahan.framework.Graphics;
import com.akibshahjahan.framework.Image;
import com.akibshahjahan.framework.Input.TouchEvent;
import com.akibshahjahan.framework.Screen;
import com.akibshahjahan.framework.Sound;

public class GameScreen extends Screen
{
        enum GameState
        {
                Ready, Play, Paused, Over
        }

        GameState state = GameState.Ready;
        private static Zien zien;
        private static boolean barOn = false;
        private static int score = 0;
        private static int presses = 0;
        private Image zienImg, backgroundImg, tapeImg, cutTapeImg, barImg,
gameOverImg, restartImg, playAgainImg, pauseImg, resumeImg, tapToPlayImg,
centuryClubImg;
        private static Sound beepSound, endSound;
        private static Tape tape1, tape3, tape5;
        private static CutTape tape2, tape4, tape6;
        private static Bar bar;
        Paint paint, paint2;
        static int getWidth = 800;
        static int getHeight = 480;
        static int tileHeight = getHeight/2;
        static int tileWidth = 100;


        public GameScreen(Game game)
        {
                super(game);

                // Initialize game objects here
                tape1 = new Tape(0, tileHeight, tileWidth);
                tape2 = new CutTape(tileHeight, tileHeight, tileWidth);
```

```
                tape3 = new Tape(tileHeight*2, tileHeight, tileWidth);
                tape4 = new CutTape(tileHeight*3, tileHeight, tileWidth);
                tape5 = new Tape(tileHeight*4, tileHeight, tileWidth);
                tape6 = new CutTape(tileHeight*5, tileHeight, tileWidth);
                zien = new Zien((int)(getWidth/3.6), (int)(getHeight/2.6));
                bar = new Bar(0, tileWidth, getHeight);

                // backgroundImg not set
                zienImg = Assets.zienImg;
                tapeImg = Assets.tapeImg;
                cutTapeImg = Assets.cutTapeImg;
                barImg = Assets.barImg;
                gameOverImg = Assets.gameOverImg;
                restartImg = Assets.restartImg;
                playAgainImg = Assets.playAgainImg;
                pauseImg = Assets.pauseImg;
                resumeImg = Assets.resumeImg;
                beepSound = Assets.beepSound;
                endSound = Assets.endSound;
                tapToPlayImg = Assets.tapToPlayImg;
                centuryClubImg = Assets.centuryClubImg;

                // paint
                Typeface tf = Typeface.create("Abadi MT Condensed Extra Bold",
Typeface.BOLD);
                paint = new Paint();
                paint.setTypeface(tf);
                paint.setTextSize(60);
                paint.setTextAlign(Paint.Align.CENTER);
                paint.setAntiAlias(true);
                paint.setColor(Color.WHITE);

                paint2 = new Paint();
                paint.setTypeface(tf);
                paint2.setTextSize(100);
                paint2.setTextAlign(Paint.Align.CENTER);
                paint2.setAntiAlias(true);
                paint2.setColor(Color.WHITE);
        }
        public void update(float deltaTime)
        {
                List touchEvents = game.getInput().getTouchEvents();
                if(state == GameState.Ready){
                        updateReady(touchEvents);
                }
                if(state == GameState.Play){
                        updatePlay(touchEvents, deltaTime);
                }
                if(state == GameState.Paused){
                        updatePaused(touchEvents);
                }
                if(state == GameState.Over){
```

```
                updateGameOver(touchEvents);
            }
        }

        private void updateReady(List touchEvents)
        {
            if(touchEvents.size()>0){
                state = GameState.Play;
                touchEvents.clear();
                presses = 0;
                score = 0;
            }
        }

        // similar to keypressed and run methods
        private void updatePlay(List touchEvents, float deltaTime)
        {
            // 1. all touch input is handled here:
            int len = touchEvents.size();
            for(int i = 0; i< len; i++)
            {
                TouchEvent event = (TouchEvent)touchEvents.get(i);
                if(event.type == TouchEvent.TOUCH_DOWN)
                {
                    if (inBounds(event, 0, 0, 70, 70))
                    {
                        pause();
                    }
                    else if(inBounds(event, 100, 50, 700, 430) && zien.getSpeedX()
==0)
                    {
                        zien.move();
                    }
                }
            }


            // 2. Call individual update() methods here.
            // This is where all the game updates happen
            // aka all the run() methods stuff
            zien.update();
            tape1.update();
            tape2.update();
            tape3.update();
            tape4.update();
            tape5.update();
            tape6.update();

            // 3. check if game is over
            if(presses>score&&zien.getSpeedX()<0&&zien.getCenterX()<(319-
zien.getWidth()))
            {
```

```
                makeSound(2);
                state = GameState.Over;
            }
            if(score==999)
            {
                makeSound(2);
                state = GameState.Over;
            }
        }

        private boolean inBounds(TouchEvent event, int x, int y, int width, int height)
        {
            if (event.x > x && event.x < x + width - 1 && event.y > y && event.y < y +
height - 1)
            {
                return true;
            }
            else
            {
                return false;
            }
        }

        private void updatePaused(List touchEvents)
        {
            int len = touchEvents.size();
            zien.update();
            tape1.update();
            tape2.update();
            tape3.update();
            tape4.update();
            tape5.update();
            tape6.update();

            for (int i = 0; i < len; i++) {
                TouchEvent event = (TouchEvent)touchEvents.get(i);
                if (event.type == TouchEvent.TOUCH_UP) {
                    if(inBounds(event, 150, 60, 500, 150)) { // tap resume
                        resume();
                    }
                    if(inBounds(event, 150, 260, 500, 150)) // tap restart
                    {
                        score = 0;
                        presses = 0;
                        state = GameState.Ready;
                    }
                }
            }
        }

        private void updateGameOver(List touchEvents) {
            int len = touchEvents.size();
```

```
                zien.update();
                tape1.update();
                tape2.update();
                tape3.update();
                tape4.update();
                tape5.update();
                tape6.update();
                for (int i = 0; i < len; i++)
                {
                        TouchEvent event = (TouchEvent)touchEvents.get(i);
                        if (event.type == TouchEvent.TOUCH_UP)
                        {
                                if(inBounds(event, 170, 250, 460, 130)) // tap "Play Again"
                                {
                                        score = 0;
                                        presses = 0;
                                        state = GameState.Ready;
                                }
                        }
                }


        }

        // similar to paint method
        @Override
        public void paint(float deltaTime)
        {
                Graphics g = game.getGraphics();
                g.drawImage(Assets.backgroundImg, 0, 0, 0, 0, getWidth, getHeight );
                g.drawImage(zienImg, zien.getCenterX(),zien.getCenterY(),zien.getCenterX
(),zien.getCenterY(), zien.getWidth(), zien.getHeight());
                g.drawImage(tapeImg, tape1.getTapeX(), tape1.getTapeY(),tape1.getTapeX
(), tape1.getTapeY(), tileWidth, tileHeight);
                g.drawImage(tapeImg, tape3.getTapeX(), tape3.getTapeY(),tape3.getTapeX
(), tape3.getTapeY(), tileWidth, tileHeight);
                g.drawImage(tapeImg, tape5.getTapeX(), tape5.getTapeY(),tape5.getTapeX
(), tape5.getTapeY(), tileWidth, tileHeight);

                g.drawImage(cutTapeImg, tape2.getTapeX(), tape2.getTapeY
(),tape2.getTapeX(), tape2.getTapeY(),tileWidth, tileHeight);
                g.drawImage(cutTapeImg, tape4.getTapeX(), tape4.getTapeY
(),tape4.getTapeX(), tape4.getTapeY(), tileWidth, tileHeight);
                g.drawImage(cutTapeImg, tape6.getTapeX(), tape6.getTapeY
(),tape6.getTapeX(), tape6.getTapeY(), tileWidth, tileHeight);

                if(state == GameState.Play)
                {
                        g.drawString(rightJustified(score),  740, 50, paint);
                        g.drawImage(pauseImg, 0, 0, 0, 0, 70, 70 );
                }
```

```
                if(barOn)
                {
                        g.drawImage(barImg, bar.getBarX(), bar.getBarY(),bar.getBarX(),
bar.getBarY(), bar.getWidth(), bar.getHeight());
                }
                if (state == GameState.Ready){
                        drawReadyUI();
                }
                if (state == GameState.Play){
                        drawPlayUI();
                }
                if (state == GameState.Paused){
                        drawPausedUI();
                }
                if (state == GameState.Over){
                        drawOverUI();
                }

                // for collision detection
                /**
                g.drawRect((int)Zien.rect.left, (int)Zien.rect.top, (int)Zien.rect.width(), (int)
Zien.rect.height(), Color.RED);
                g.drawRect((int)tape2.rect.left, (int)tape2.rect.top, (int)tape2.rect.width(), (int)
tape2.rect.height(), Color.RED);
                g.drawRect((int)tape4.rect.left, (int)tape4.rect.top, (int)tape4.rect.width(), (int)
tape4.rect.height(), Color.RED);
                g.drawRect((int)tape6.rect.left, (int)tape6.rect.top, (int)tape6.rect.width(), (int)
tape6.rect.height(), Color.RED);
                g.drawRect((int)tape1.rect.left, (int)tape1.rect.top, (int)tape1.rect.width(), (int)
tape1.rect.height(), Color.RED);
                g.drawRect((int)tape3.rect.left, (int)tape3.rect.top, (int)tape3.rect.width(), (int)
tape3.rect.height(), Color.RED);
                g.drawRect((int)tape5.rect.left, (int)tape5.rect.top, (int)tape5.rect.width(), (int)
tape5.rect.height(), Color.RED);
                //**/

        }
        private void nullify()
        {

                // Set all variables to null. You will be recreating them in the
                // constructor.
                paint = null;
                zienImg = null;
                backgroundImg = null;
                tapeImg = null;
                cutTapeImg = null;
                barImg = null;
                gameOverImg = null;

                // Call garbage collector to clean up memory.
                System.gc();
```

```
}
private void drawReadyUI()
{
	Graphics g = game.getGraphics();
	zien.update();
	tape1.update();
	tape2.update();
	tape3.update();
	tape4.update();
	tape5.update();
	tape6.update();
	g.drawARGB(155, 0, 0, 0);
	//g.drawString("Tap to Start", 400, 240, paint);
	g.drawImage(tapToPlayImg, 225, 190,225,190, 350, 100);
	if(SampleGame.settings.getInt("highscore",0)>=100)
	{
		g.drawImage(centuryClubImg, 10, 425, 10, 425, 260, 50);
	}
}

private void drawPlayUI()
{
	Graphics g = game.getGraphics();
}

private void drawPausedUI()
{
	Graphics g = game.getGraphics();
	// Darken the entire screen to display the Paused screen.
	g.drawARGB(155, 0, 0, 0);
	//g.drawString("Resume", 400, 165, paint2);
	g.drawString(rightJustified(score),  740, 50, paint);
	g.drawImage(resumeImg, 150, 60, 150, 60, 500, 150);
	g.drawImage(restartImg, 150, 260, 130, 260, 500, 150);
	//g.drawString("Menu", 400, 360, paint2);
}

private void drawOverUI()
{
	Graphics g = game.getGraphics();
	g.drawARGB(155, 0, 0, 0);
	g.drawImage(gameOverImg, 70, 40, 70, 40, 660, 200);
	g.drawString(rightJustified(score),  740, 50, paint);
	g.drawImage(playAgainImg, 170, 250, 170, 250, 460, 140);
	if(score>SampleGame.settings.getInt("highscore",0))
	{
		g.drawString("High Score: "+score, 400, 448, paint);
		SampleGame.editor.putInt("highscore", score);
		SampleGame.editor.commit();
	}
	else
	{
```

```
                        g.drawString("High Score: "+SampleGame.settings.getInt("highscore",
0), 400, 448, paint);
                }
        }

        @Override
        public void pause()
        {
                if (state == GameState.Play)
                {
                        state = GameState.Paused;
                }
        }

        @Override
        public void resume()
        {
                if (state == GameState.Paused)
                {
                        state = GameState.Play;
                }
        }

        @Override
        public void dispose()
        {

        }

        @Override
        public void backButton()
        {
                pause();
        }

        private void goToMenu()
        {
                game.setScreen(new MainMenuScreen(game));
        }

        public static String rightJustified(int num)
        {
                String s = ""+num;
                if(s.length()==1)
                {
                        s = "    "+num;
                }
                else if(s.length()==2)
                {
                        s = "  "+num;
                }
                return s;
```

```
}

public static int getTheHeight()
{
	return getHeight;
}

public static void reverseZien()
{
	zien.moveLeft();
}

public static Zien getZien ()
{
	return zien;
}

public static void setBar(boolean b)
{
	barOn = b;
}
public static int getScore()
{
	return score;
}

public static void setScore()
{
	score++;
	tape2.setScoreOn(false);
	tape4.setScoreOn(false);
	tape6.setScoreOn(false);
}

public static void setScoreOn()
{
	tape2.setScoreOn(true);
	tape4.setScoreOn(true);
	tape6.setScoreOn(true);
}

public static void setPresses()
{
	presses++;
}

// 1 as argument is for scoring
// 2 argument is for when game is over
public static void makeSound(int i)
{
	if(i==1)
	{
```

```
                beepSound.play(1.00f);
            }
            if(i==2)
            {
                endSound.play(4.00f);
            }
        }
}
```

LoadingScreen.java

```
import com.akibshahjahan.framework.Game;
import com.akibshahjahan.framework.Graphics;
import com.akibshahjahan.framework.Graphics.ImageFormat;
import com.akibshahjahan.framework.Screen;

public class LoadingScreen extends Screen
{
    public LoadingScreen(Game game)
    {

        super(game);
    }

    @Override
    public void update(float deltaTime)
    {
        Graphics g = game.getGraphics();
        //Assets.menu = g.newImage("menu.png", ImageFormat.RGB565);
        Assets.backgroundImg = g.newImage("background.png", ImageFormat.RGB565);
        Assets.zienImg = g.newImage("zien.png", ImageFormat.ARGB4444);

        Assets.tapeImg = g.newImage("tape.png", ImageFormat.ARGB4444);
        Assets.cutTapeImg = g.newImage("cutTape.png", ImageFormat.ARGB4444);

        Assets.barImg = g.newImage("yellowBar.png", ImageFormat.RGB565);

        Assets.gameOverImg = g.newImage("gameOver.png", ImageFormat.RGB565);
        Assets.restartImg = g.newImage("restart.png", ImageFormat.RGB565);
        Assets.playAgainImg = g.newImage("playAgain.png", ImageFormat.RGB565);
        Assets.pauseImg = g.newImage("pause.png", ImageFormat.RGB565);
        Assets.resumeImg = g.newImage("resume.png", ImageFormat.RGB565);
        Assets.tapToPlayImg = g.newImage("tapToPlay.png", ImageFormat.RGB565);
        Assets.centuryClubImg = g.newImage("centuryClub.png", ImageFormat.RGB565);

        Assets.beepSound = game.getAudio().createSound("beep.mp3");
        Assets.endSound = game.getAudio().createSound("end.mp3");

        game.setScreen(new MainMenuScreen(game));
    }

    @Override
```

```
    public void paint(float deltaTime)
    {
        Graphics g = game.getGraphics();
        g.drawImage(Assets.splash, 0, 0);
    }

    @Override
    public void pause()
    {

    }

    @Override
    public void resume()
    {

    }

    @Override
    public void dispose()
    {

    }

    @Override
    public void backButton()
    {

    }
}
```

MainMenuScreen.java

```
import java.util.List;

import com.akibshahjahan.framework.Game;
import com.akibshahjahan.framework.Graphics;
import com.akibshahjahan.framework.Screen;
import com.akibshahjahan.framework.Input.TouchEvent;

public class MainMenuScreen extends Screen
{
    public MainMenuScreen(Game game)
    {
        super(game);
    }

    @Override
    public void update(float deltaTime)
    {
          game.setScreen(new GameScreen(game));
    }
```

```
    private boolean inBounds(TouchEvent event, int x, int y, int width, int height)
    {
        if (event.x > x && event.x < x + width - 1 && event.y > y && event.y < y + height - 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    @Override
    public void paint(float deltaTime)
    {
        Graphics g = game.getGraphics();
    }

    @Override
    public void pause()
    {

    }

    @Override
    public void resume()
    {

    }

    @Override
    public void dispose()
    {

    }

    @Override
    public void backButton()
    {
        android.os.Process.killProcess(android.os.Process.myPid());
    }
}
```

SampleGame.java

```
import android.content.SharedPreferences;

import com.akibshahjahan.framework.Screen;
import com.akibshahjahan.framework.implementation.AndroidGame;

public class SampleGame extends AndroidGame
```

```
{
    public static String map;
    boolean firstTimeCreate = true;
    public static SharedPreferences settings;
    public static SharedPreferences.Editor editor;
    @Override
    public Screen getInitScreen()
    {
        if (firstTimeCreate)
        {
            Assets.load(this);
            firstTimeCreate = false;
        }

         settings = getSharedPreferences("ZIEN_PREFS", 0);
                editor = settings.edit();

                return new SplashLoadingScreen(this);
    }

    @Override
    public void onBackPressed()
    {
        getCurrentScreen().backButton();
    }


    @Override
    public void onResume()
    {
        super.onResume();
        Assets.music.play();
    }

    @Override
    public void onPause()
    {
        super.onPause();
        Assets.music.pause();
    }
}
```

SplashLoadingScreen.java

```
import com.akibshahjahan.framework.Game;
import com.akibshahjahan.framework.Graphics;
import com.akibshahjahan.framework.Screen;
import com.akibshahjahan.framework.Graphics.ImageFormat;

public class SplashLoadingScreen extends Screen
{
    public SplashLoadingScreen(Game game)
```

```
    {
        super(game);
    }

    @Override
    public void update(float deltaTime)
    {
        Graphics g = game.getGraphics();
        Assets.splash= g.newImage("splash.png", ImageFormat.RGB565);

        game.setScreen(new LoadingScreen(game));
    }

    @Override
    public void paint(float deltaTime)
    {

    }

    @Override
    public void pause()
    {

    }

    @Override
    public void resume()
    {

    }

    @Override
    public void dispose()
    {

    }

    @Override
    public void backButton()
    {
```

Tape.java

```
import android.graphics.Rect;

public class Tape
{
        private int tapeY;
        private int tapeX = 550;
        private int speedY = -5;
```

```
private int cycles = 0;
private int height;
private int width;
public Rect rect = new Rect(0,0,0,0);
private Zien z = GameScreen.getZien();

public Tape(int y, int h, int w)
{
	tapeY = y;
	height = h;
	width = w;
	rect = new Rect();
}

public void update()
{
	tapeY += speedY;

	if (tapeY <= -height){
		tapeY = height*5;
	}
	if (tapeY <= 0)
	{
		cycles++;
	}

	rect.set(tapeX, tapeY-45, tapeX+10, tapeY+height+45);
	if(Rect.intersects(rect, z.rect))
	{
		GameScreen.reverseZien();
	}
}

public int getCycles()
{
	return cycles;
}

public int getTapeX()
{
	return tapeX;
}

public int getTapeY()
{
	return tapeY;
}

public int getSpeedY()
{
	return speedY;
}
```

```
	public void setTapeX(int tapeX)
	{
		this.tapeX = tapeX;
	}


	public void setTapeY(int tapeY)
	{
		this.tapeY = tapeY;
	}

	public void setSpeedY(int speedY)
	{
		this.speedY = speedY;
	}
}
```

Zien.java

```
import android.graphics.Rect;
import java.util.Random;

public class Zien
{
	final int MOVESPEED = 20;
	private int centerX = 140;
	private int centerY = 140;
	private int speedX = 0;
	private int limit1 = 0;
	private int limit2 = 390;
	private int limit3 = limit2+60;
	private int width, height;
	private boolean animation = true;
	private double horAnim = 0;
	private double verAnim = 0;
	private double horLeftAnim = 0;
	private double verLeftAnim = 0;
	double animationSpeed;
	double maxAnimationSpeed = 20; //lower means faster
	double minAnimationSpeed = 45; //higher means slower
	public static Rect rect = new Rect(0,0,0,0);

	public Zien(int w, int h)
	{
		setWidth(w);
		setHeight(h);
	}

	public void update()
	{
		//animation = false; // for testing purpose
```

```
		if(animation)
		{
			setAnimationSpeed();
			if(horLeftAnim<=2)
			{
				Random gen = new Random();
				horLeftAnim = gen.nextInt(31)+50;
				horAnim = (horLeftAnim/animationSpeed);
				int direction = gen.nextInt(2);
				if(direction==0)
				{
					horAnim = (horLeftAnim/animationSpeed)*-1;
				}
			}
			if(verLeftAnim<=2)
			{
				Random gen = new Random();
				verLeftAnim = gen.nextInt(31)+50;
				verAnim = (verLeftAnim/animationSpeed);
				int direction = gen.nextInt(2);
				if(direction==0)
				{
					verAnim = (verLeftAnim/animationSpeed)*-1;
				}
			}
			if(centerX+horAnim+getWidth()<limit3&&centerX+horAnim>0)
			{
				centerX+=horAnim;
				horLeftAnim -= Math.abs(horAnim);
			}
			else
			{
				horLeftAnim = 0;
			}
			if(centerY+verAnim+getHeight()<GameScreen.getTheHeight()
&&centerY+verAnim>0)
			{
				centerY+=verAnim;
				verLeftAnim -= Math.abs(verAnim);
			}
			else
			{
				verLeftAnim = 0;
			}
		}

		if(speedX!=0 && centerX+speedX<limit2 && centerX+speedX>limit1 && !
animation)
		{
			centerX +=speedX;
		}
		else
```

```
		{
			stop();
			GameScreen.setScoreOn();
			animation = true;
		}

		if(centerX+speedX>=limit2)
		{
			moveLeft();
		}

		if(centerX<=limit1)
		{
			stop();
			animation = true;

		}
		if(centerX<100)
		{
			GameScreen.setBar(false);
		}

		rect.set((int)(centerX+(getWidth()/1.24)), (int)(centerY+getHeight()/2.3), (int)
(centerX+(getWidth()/1.24))+(int)(getWidth()/8.5), (int)(centerY+getHeight()/2.3)+(int)
(getHeight()/11));
	}

	public void move()
	{
		GameScreen.setPresses();
		horLeftAnim = 0;
		verLeftAnim = 0;
		animation = false;
		moveRight();
	}

	public void moveRight()
	{
		speedX = MOVESPEED;
	}

	public void moveLeft()
	{
		speedX = -MOVESPEED;
	}

	public void stop()
	{
		speedX = 0;
	}

	public void setAnimationSpeed()
```

```
{
        animationSpeed = minAnimationSpeed - GameScreen.getScore()/3;
        if(animationSpeed<maxAnimationSpeed)
        {
                animationSpeed = maxAnimationSpeed;
        }
}

public int getCenterX()
{
        return centerX;
}

public void setCenterX(int centerX)
{
        this.centerX = centerX;
}

public int getCenterY()
{
        return centerY;
}

public void setCenterY(int centerY)
{
        this.centerY = centerY;
}

public int getSpeedX()
{
        return speedX;
}

public void setSpeedX(int speedX)
{
        this.speedX = speedX;
}

public int getWidth()
{
        return width;
}

public void setWidth(int width)
{
        this.width = width;
}

public int getHeight()
{
        return height;
}
```

```
	public void setHeight(int height)
	{
		this.height = height;
	}
}
```