JAVA SWING & AWT

RECURSOS BÁSICOS

UM BREVE TUTORIAL

(Versão 1.1)

ALCIDES CALSAVARA

PUCPR

2018

# Criação de um JPanel e correspondente JFrame

Crie um projeto Java.

Crie a classe **Territorio**:

import javax.swing.\*;  
  
public class Territorio extends JPanel {  
  
 private static final int *ALTURA\_BARRA\_TITULO* = 20;  
  
 private JFrame frame;  
  
 public Territorio(String nome) {  
 frame = new JFrame(nome); // cria um frame  
 frame.add(this); // insere o território no frame  
 frame.setSize(400, 300 + *ALTURA\_BARRA\_TITULO* ); // define as dimensões do frame  
 frame.setVisible(true); // torna o frame visível  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*); // define como o frame é fechado  
 }  
}

Crie a classe **Mutantes**:

public class Mutantes {  
 public static void main(String[] args) {  
 Territorio territorio = new Territorio("Mutantes");  
 }  
}

Compile e execute o projeto.

Observe a janela exibida na tela com título "Mutantes".

Feche a janela "Mutantes".

# Sobrecarga do método paint e escrita de textos

Insira o atributo **contador** na classe **Territorio**:

private int contador = 0;

Insira o método **paint** na classe **Territorio**:

public void paint(Graphics g) {  
 super.paint(g);  
 Graphics2D g2d = (Graphics2D) g; // por motivos de compatibilidade com a AWT  
 g2d.setColor(Color.*GRAY*); // define a cor em uso  
 g2d.setFont(new Font("Verdana", Font.*BOLD*, 16)); // define a fonte em uso  
 g2d.drawString("Contador: " + String.*valueOf*(contador), 10, 30); // escreve uma string  
}

É necessário fazer a seguinte importação:

import java.awt.\*;

Compile e execute o projeto.

Observe a janela exibida na tela com título "Mutantes".

Feche a janela "Mutantes".

# Implementação de animação e uso do método repaint

Insira o método **jogar** na classe **Territorio**:

public void jogar() {  
  
 boolean jogando = true;  
  
 while (jogando) {  
 contador++;  
 if (contador == 10) jogando = false;

repaint(); // atualiza a imagem da janela

try {  
 Thread.*sleep*(500); // dorme por meio segundo  
 }  
 catch ( Exception e )  
 {  
 e.printStackTrace();  
 }  
 }  
}

Insira a chamada do método **jogar** na classe **Mutantes**:

territorio.jogar();

Compile e execute o projeto.

Observe a janela exibida na tela com título "Mutantes".

Feche a janela "Mutantes".

Modifique o tempo de espera entre cada passo da repetição (comando *while*) para 1 segundo e observe o efeito na execução do projeto.

Volte o tempo para meio segundo.

Insira uma espera de 5 segundos no início do método **jogar** e observe o efeito na execução do projeto.

# Captura das dimensões da janela

Insira o comando a seguir dentro do comando *while* do método **jogar** para visualizar as dimensões atuais da janela:

System.*out*.println("largura: " + getWidth() + ", altura: " + getHeight());

Aumente o valor no teste da variável **contador** de 10 para 100:

if (contador == 100) jogando = false;

Observe o efeito na execução do projeto aumentando e diminuindo a janela.

Volte o valor no teste da variável **contador** para 10.

# Criação de uma caixa de diálogo

Insira o método **game\_over** na classe **Territorio**:

private void game\_over() {  
 String mensagem = "Parabéns!";  
 JOptionPane.*showMessageDialog*(this, mensagem, "Game Over", JOptionPane.*YES\_NO\_OPTION*);  
}

Insira a chamada do método **game\_over** no final do método **jogar** (depois do comando *while*):

game\_over();

Observe o efeito na execução do projeto.

# Desenho de figuras

Insira o código a seguir no final do método **paint** da classe **Territorio** para desenhar um retângulo verde de 40x20 na posição (30, 60) do território:

g2d.setColor(Color.*GREEN*);  
g2d.fillRect(30, 60, 40, 20);

Observe o efeito na execução do projeto.

O diagrama abaixo mostra a convenção utilizada para as coordenadas de um objeto gráfico. No exemplo, é desenhado um retângulo verde com base 40 e altura 20, no ponto (30, 60).



# Criação de objetos com representação gráfica

Crie a classe **Ser**:

import javax.swing.\*;  
import java.awt.\*;  
  
public class Ser extends JComponent {  
  
 private Territorio territorio;  
 private Color cor;  
 private int x = 50;  
 private int y = 40;

private int altura = 30;

private int largura = 30;  
  
 public Ser(Color cor, Territorio territorio) {  
 this.cor = cor;  
 this.territorio = territorio;  
 }  
  
 public void paint(Graphics g) {  
 Graphics2D g2d = (Graphics2D) g; // compatibilidade com a AWT  
  
 g2d.setColor(cor);  
  
 g2d.fillOval(x, y, largura, altura); // desenha um círculo  
 }  
}

Insira o atributo **ser** na classe **Territorio**:

private Ser ser;

Insira o código para instanciação da classe **Ser** no início do método construtor da classe **Territorio**:

ser = new Ser(Color.*BLUE*, this);

Insira a chamada do método **paint** do **ser** no final do método **paint** da classe **Territorio**:

if (ser != null) ser.paint(g);

Observe o efeito na execução do projeto.

O diagrama abaixo mostra a representação gráfica do **ser** representado como uma elipse (oval) azul de largura 30 e altura 20, no ponto (50, 40).



Insira o código a seguir no início do método **paint** da classe **Territorio** para melhorar a qualidade da imagem:

g2d.setRenderingHint(RenderingHints.*KEY\_ANTIALIASING*,  
 RenderingHints.*VALUE\_ANTIALIAS\_ON*);

Observe o efeito na execução do projeto.

# Tratamento de eventos via teclado

Crie a classe **LeitorSetas**:

import java.awt.event.KeyEvent;  
import java.awt.event.KeyListener;  
  
public class LeitorSetas implements KeyListener {  
  
 private Ser ser;  
  
 public LeitorSetas(Ser ser) {  
 this.ser = ser;  
 }  
  
 public void keyTyped(KeyEvent e) {  
 }  
  
 public void keyPressed(KeyEvent e) {  
 ser.keyPressed(e);  
 }  
  
 public void keyReleased(KeyEvent e) {  
 }  
}

Insira o método **keyPressed** na classe **Ser**:

public void keyPressed(KeyEvent e) {  
 switch(e.getKeyCode()) {  
 case KeyEvent.*VK\_LEFT*: { x = x - 5; break; }  
 case KeyEvent.*VK\_RIGHT*: { x = x + 5; break; }  
 case KeyEvent.*VK\_UP*: { y = y - 5; break; }  
 case KeyEvent.*VK\_DOWN*: { y = y + 5; break; }  
 }  
}

Insira o código para instanciação da classe **LeitorSetas** no início do construtor da classe **Territorio,** logo após a instanciação do **ser**:

KeyListener listener = new LeitorSetas(ser);  
addKeyListener(listener);  
setFocusable(true);

Observe o efeito na execução do projeto pressionando as setas no teclado.

# Implementação de aleatoriedade

Insira o atributo **gerador\_aleatorio** na classe **Ser**:

private static final Random *gerador\_aleatorio* = new Random();

É necessário fazer a seguinte importação:

import java.util.Random;

Insira o código abaixo no final do construtor da classe **Ser** para gerar valores aleatórios para as coordenadas **x** e **y**:

x = *gerador\_aleatorio*.nextInt(200); // gera entre 0 e 199  
y = *gerador\_aleatorio*.nextInt(100); // gera entre 0 e 99  
System.*out*.println("x: " + x + ", y: " + y);

Execute o projeto diversas vezes e observe o efeito.

# Referências

**Game programming for beginners**

http://www.edu4java.com/en/game/game0-en.html