# PAINNASSAU GAME STUDIO

Integrantes:

-Camile Marcele Araújo-01592555,

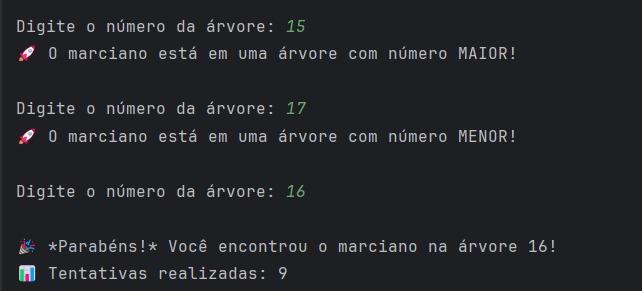
-Márcio Maycom da Silva oliveira-01607574,

-Pedro Miranda Marinho- 01607408,

-Rafaella Guedes Barbosa - 01588710

-Ricardo César Ottoni-01596659

Jogo do Marciano

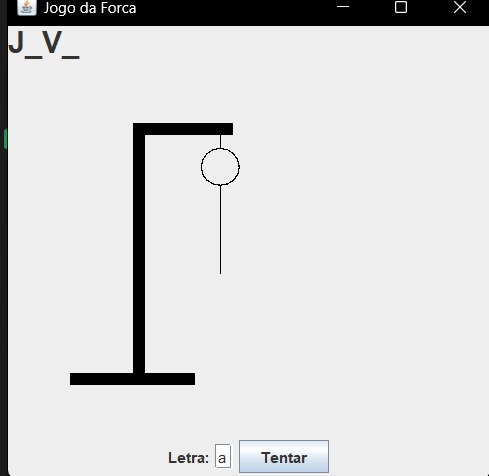


**- Código fonte -**

**JogoMarciano\Marciano.java**

|  |  |
| --- | --- |
| 1 | **package** JogoMarciano; |
| 2 |  |
| 3 | **import** java.util.Random; |
| 4 | **import** java.util.Scanner; |
| 5 |  |
| 6 | **public** **class** **Marciano** { |
| 7 | **public** **static** **void** **main**(String[] args) { |
| 8 | // Configuração inicial |
| 9 | Random random = **new** **Random**(); |
| 10 | Scanner scanner = **new** **Scanner**(System.in); |
| 11 |  |
| 12 | int numeroMarciano = random.nextInt(100) + 1; // Número aleatório entre 1 e 100 |
| 13 | int tentativas = 0; |
| 14 | int palpite; |
| 15 |  |
| 16 | System.out.println("🌌 \*Jogo do JogoMarciano.Marciano\* 🌌"); |
| 17 | System.out.println("Um marciano está escondido em uma árvore numerada de 1 a 100. Encontre-o!"); |
| 18 |  |
| 19 | // Loop principal do jogo |
| 20 | **do** { |
| 21 | System.out.print("\nDigite o número da árvore: "); |
| 22 | palpite = scanner.nextInt(); |
| 23 | tentativas++; |
| 24 |  |
| 25 | **if** (palpite < numeroMarciano) { |
| 26 | System.out.println("🚀 O marciano está em uma árvore com número MAIOR!"); |
| 27 | } **else** **if** (palpite > numeroMarciano) { |
| 28 | System.out.println("🚀 O marciano está em uma árvore com número MENOR!"); |
| 29 | } |
| 30 | } **while** (palpite != numeroMarciano); |
| 31 |  |
| 32 | // Mensagem de vitória |
| 33 | System.out.println("\n🎉 \*Parabéns!\* Você encontrou o marciano na árvore " + numeroMarciano + "!"); |
| 34 | System.out.println("📊 Tentativas realizadas: " + tentativas); |
| 35 |  |
| 36 | scanner.close(); |
| 37 | } |
| 38 | } |

Jogo da Forca



Interface gráfica do usuário

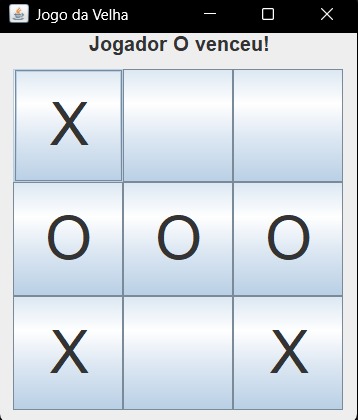
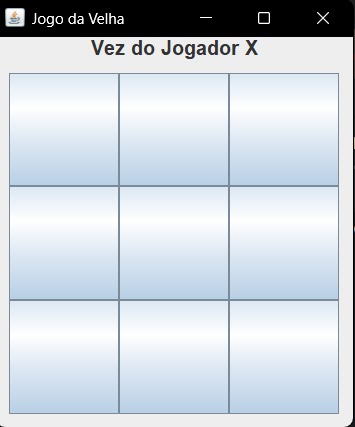
O conteúdo gerado por IA pode estar incorreto.

**- Código fonte -**

**JogoForca\ForcaGUI.java**

|  |  |
| --- | --- |
| 1 | **package** JogoForca; |
| 2 |  |
| 3 | **import** javax.swing.\*; |
| 4 | **import** java.awt.\*; |
| 5 | **import** java.util.Arrays; |
| 6 |  |
| 7 | **public** **class** **ForcaGUI** **extends** **JFrame** { |
| 8 | **private** String palavraSecreta = "JAVA"; |
| 9 | **private** char[] letrasDescobertas; |
| 10 | **private** int tentativasRestantes = 6; |
| 11 | **private** StringBuilder letrasErradas = **new** **StringBuilder**(); |
| 12 |  |
| 13 | **private** JPanel enforcadoPanel; |
| 14 | **private** JLabel palavraLabel; |
| 15 | **private** JTextField letraField; |
| 16 | **private** JButton tentarButton; |
| 17 |  |
| 18 | **public** **ForcaGUI**() { |
| 19 | setTitle("Jogo da Forca"); |
| 20 | setSize(400, 400); |
| 21 | setDefaultCloseOpera­tion(JFrame.EXIT\_ON\_CLOSE); |
| 22 | setLayout(**new** **BorderLayout**()); |
| 23 |  |
| 24 | letrasDescobertas = **new** **char**[palavraSecreta.length()]; |
| 25 | Arrays.fill(letrasDescobertas, '\_'); |
| 26 |  |
| 27 | // Painel do desenho do enforcado |
| 28 | enforcadoPanel = **new** **JPanel**() { |
| 29 | @Override |
| 30 | **protected** **void** **paintComponent**(Graphics g) { |
| 31 | super.paintComponent(g); |
| 32 | desenharForca(g); |
| 33 | } |
| 34 | }; |
| 35 | enforcadoPanel.setPreferredSize(**new** **Dimension**(200, 200)); |
| 36 |  |
| 37 | // Painel de controle |
| 38 | JPanel controlePanel = **new** **JPanel**(); |
| 39 | palavraLabel = **new** **JLabel**(String.valueOf(letrasDescobertas)); |
| 40 | palavraLabel.setFont(**new** **Font**("Arial", Font.BOLD, 24)); |
| 41 |  |
| 42 | letraField = **new** **JTextField**(1); |
| 43 | tentarButton = **new** **JButton**("Tentar"); |
| 44 |  |
| 45 | tentarButton.addActionListener(e -> processarTentativa()); |
| 46 |  |
| 47 | controlePanel.add(**new** **JLabel**("Letra:")); |
| 48 | controlePanel.add(letraField); |
| 49 | controlePanel.add(tentarButton); |
| 50 |  |
| 51 | add(enforcadoPanel, BorderLayout.CENTER); |
| 52 | add(palavraLabel, BorderLayout.NORTH); |
| 53 | add(controlePanel, BorderLayout.SOUTH); |
| 54 |  |
| 55 | setVisible(true); |
| 56 | } |
| 57 |  |
| 58 | **private** **void** **desenharForca**(Graphics g) { |
| 59 | g.setColor(Color.BLACK); |
| 60 | // Base |
| 61 | g.fillRect(50, 250, 100, 10); |
| 62 | // Poste vertical |
| 63 | g.fillRect(100, 50, 10, 200); |
| 64 | // Braço horizontal |
| 65 | g.fillRect(100, 50, 80, 10); |
| 66 | // Corda |
| 67 | g.drawLine(170, 50, 170, 70); |
| 68 |  |
| 69 | **if** (tentativasRestantes < 6) { // Cabeça |
| 70 | g.drawOval(155, 70, 30, 30); |
| 71 | } |
| 72 | **if** (tentativasRestantes < 5) { // Corpo |
| 73 | g.drawLine(170, 100, 170, 170); |
| 74 | } |
| 75 | **if** (tentativasRestantes < 4) { // Braço esquerdo |
| 76 | g.drawLine(170, 120, 140, 140); |
| 77 | } |
| 78 | **if** (tentativasRestantes < 3) { // Braço direito |
| 79 | g.drawLine(170, 120, 200, 140); |
| 80 | } |
| 81 | **if** (tentativasRestantes < 2) { // Perna esquerda |
| 82 | g.drawLine(170, 170, 140, 200); |
| 83 | } |
| 84 | **if** (tentativasRestantes < 1) { // Perna direita |
| 85 | g.drawLine(170, 170, 200, 200); |
| 86 | } |
| 87 | } |
| 88 |  |
| 89 | **private** **void** **processarTentativa**() { |
| 90 | String input = letraField.getText().toUpperCase(); |
| 91 | letraField.setText(""); |
| 92 |  |
| 93 | **if** (input.length() != 1 || !Character.isLetter(input.charAt(0))) { |
| 94 | JOptionPane.showMessageDialog(this, "Digite uma única letra!"); |
| 95 | **return**; |
| 96 | } |
| 97 |  |
| 98 | char letra = input.charAt(0); |
| 99 | boolean acertou = false; |
| 100 |  |
| 101 | **for** (int i = 0; i < palavraSecreta.length(); i++) { |
| 102 | **if** (palavraSecreta.charAt(i) == letra) { |
| 103 | letrasDescobertas[i] = letra; |
| 104 | acertou = true; |
| 105 | } |
| 106 | } |
| 107 |  |
| 108 | **if** (!acertou) { |
| 109 | tentativasRestantes--; |
| 110 | letrasErradas.append(letra).append(" "); |
| 111 | } |
| 112 |  |
| 113 | palavraLabel.setText(String.valueOf(letrasDescobertas)); |
| 114 | enforcadoPanel.repaint(); |
| 115 |  |
| 116 | verificarFimJogo(); |
| 117 | } |
| 118 |  |
| 119 | **private** **void** **verificarFimJogo**() { |
| 120 | **if** (String.valueOf(letrasDescobertas).equals(palavraSecreta)) { |
| 121 | JOptionPane.showMessageDialog(this, "Parabéns! Você ganhou!"); |
| 122 | System.exit(0); |
| 123 | } **else** **if** (tentativasRestantes == 0) { |
| 124 | JOptionPane.showMessageDialog(this, "Game Over! A palavra era: " + palavraSecreta); |
| 125 | System.exit(0); |
| 126 | } |
| 127 | } |
| 128 |  |
| 129 | **public** **static** **void** **main**(String[] args) { |
| 130 | **new** **ForcaGUI**(); |
| 131 | } |
| 132 | } |

Jogo da Velha



**- Código fonte -**

**JogoDaVelha\JogoDaVelhaGUI.java**

|  |  |
| --- | --- |
| 1 | **package** JogoDaVelha; |
| 2 |  |
| 3 | **import** javax.swing.\*; |
| 4 | **import** java.awt.\*; |
| 5 | **import** java.awt.event.ActionEvent; |
| 6 | **import** java.awt.event.ActionListener; |
| 7 |  |
| 8 | **public** **class** **JogoDaVelhaGUI** **extends** **JFrame** { |
| 9 | **private** JButton[][] botoes = **new** **JButton**[3][3]; |
| 10 | **private** char jogadorAtual = 'X'; |
| 11 | **private** boolean jogoAtivo = true; |
| 12 | **private** JLabel statusLabel; |
| 13 |  |
| 14 | **public** **JogoDaVelhaGUI**() { |
| 15 | setTitle("Jogo da Velha"); |
| 16 | setDefaultCloseOpera­tion(JFrame.EXIT\_ON\_CLOSE); |
| 17 | setSize(300, 350); |
| 18 | setLayout(**new** **BorderLayout**()); |
| 19 |  |
| 20 | // Painel de status |
| 21 | statusLabel = **new** **JLabel**("Vez do Jogador " + jogadorAtual, SwingConstants.CENTER); |
| 22 | statusLabel.setFont(**new** **Font**("Arial", Font.BOLD, 16)); |
| 23 | add(statusLabel, BorderLayout.NORTH); |
| 24 |  |
| 25 | // Painel do tabuleiro |
| 26 | JPanel tabuleiroPanel = **new** **JPanel**(**new** **GridLayout**(3, 3)); |
| 27 | tabuleiroPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10)); |
| 28 |  |
| 29 | // Cria os botões do tabuleiro |
| 30 | **for** (int linha = 0; linha < 3; linha++) { |
| 31 | **for** (int coluna = 0; coluna < 3; coluna++) { |
| 32 | JButton botao = **new** **JButton**(); |
| 33 | botao.setFont(**new** **Font**("Arial", Font.PLAIN, 48)); |
| 34 | botao.setFocusPainted(false); |
| 35 | botao.addActionListener(**new** **BotaoClickListener**(linha, coluna)); |
| 36 | botoes[linha][coluna] = botao; |
| 37 | tabuleiroPanel.add(botao); |
| 38 | } |
| 39 | } |
| 40 |  |
| 41 | add(tabuleiroPanel, BorderLayout.CENTER); |
| 42 | setVisible(true); |
| 43 | } |
| 44 |  |
| 45 | **private** **class** **BotaoClickListener** **implements** **ActionListener** { |
| 46 | **private** int linha; |
| 47 | **private** int coluna; |
| 48 |  |
| 49 | **public** **BotaoClickListener**(int linha, int coluna) { |
| 50 | this.linha = linha; |
| 51 | this.coluna = coluna; |
| 52 | } |
| 53 |  |
| 54 | @Override |
| 55 | **public** **void** **actionPerformed**(ActionEvent e) { |
| 56 | JButton botaoClicado = (JButton) e.getSource(); |
| 57 |  |
| 58 | **if** (botaoClicado.getText().isEmpty() && jogoAtivo) { |
| 59 | botaoClicado.setText(String.valueOf(jogadorAtual)); |
| 60 |  |
| 61 | **if** (verificarVencedor()) { |
| 62 | statusLabel.setText("Jogador " + jogadorAtual + " venceu!"); |
| 63 | jogoAtivo = false; |
| 64 | } **else** **if** (verificarEmpate()) { |
| 65 | statusLabel.setText("Empate!"); |
| 66 | jogoAtivo = false; |
| 67 | } **else** { |
| 68 | trocarJogador(); |
| 69 | statusLabel.setText("Vez do Jogador " + jogadorAtual); |
| 70 | } |
| 71 | } |
| 72 | } |
| 73 | } |
| 74 |  |
| 75 | **private** **void** **trocarJogador**() { |
| 76 | jogadorAtual = (jogadorAtual == 'X') ? 'O' : 'X'; |
| 77 | } |
| 78 |  |
| 79 | **private** boolean **verificarVencedor**() { |
| 80 | // Verifica linhas e colunas |
| 81 | **for** (int i = 0; i < 3; i++) { |
| 82 | **if** (checarCombinacao(botoes[i][0], botoes[i][1], botoes[i][2]) || |
| 83 | checarCombinacao(botoes[0][i], botoes[1][i], botoes[2][i])) { |
| 84 | **return** true; |
| 85 | } |
| 86 | } |
| 87 |  |
| 88 | // Verifica diagonais |
| 89 | **return** checarCombinacao(botoes[0][0], botoes[1][1], botoes[2][2]) || |
| 90 | checarCombinacao(botoes[0][2], botoes[1][1], botoes[2][0]); |
| 91 | } |
| 92 |  |
| 93 | **private** boolean **checarCombinacao**(JButton b1, JButton b2, JButton b3) { |
| 94 | **return** !b1.getText().isEmpty() && |
| 95 | b1.getText().equals(b2.getText()) && |
| 96 | b2.getText().equals(b3.getText()); |
| 97 | } |
| 98 |  |
| 99 | **private** boolean **verificarEmpate**() { |
| 100 | **for** (int i = 0; i < 3; i++) { |
| 101 | **for** (int j = 0; j < 3; j++) { |
| 102 | **if** (botoes[i][j].getText().isEmpty()) { |
| 103 | **return** false; |
| 104 | } |
| 105 | } |
| 106 | } |
| 107 | **return** true; |
| 108 | } |
| 109 |  |
| 110 | **public** **static** **void** **main**(String[] args) { |
| 111 | SwingUtilities.invokeLater(() -> **new** **JogoDaVelhaGUI**()); |
| 112 | } |
| 113 | } |