**Fatec Zona Sul**

Robô que realiza operações aritméticas

Fabricio Bezerra dos Santos

4º Semestre - Analise e Desenvolvimento de Sistemas

08/2017

# 1. Diagramas Astah

## 1.1 Diagrama de ClassesClass Diagram0.png

## 1.2 Diagrama de Caso de Uso UseCaseRobo.png

## 1.3 Diagrama de Sequência

## Robo.png

## 

# 2. Código:

## 2.1 Robo.java

package io.github.eldemonstro.robo;

/\*

\* @author: Fabricio Bezerra (github.com/eldemontro)

\*/

public class Robo {

private String nome;

private String dtFabr;

private int bateria;

public String getNome() {

return nome;

}

public void setNome(String nome) {

this.nome = nome;

}

public String getDtFabr() {

return dtFabr;

}

public void setDtFabr(String dtFabr) {

this.dtFabr = dtFabr;

}

public int getBateria() {

return bateria;

}

public void setBateria(int bateria) {

this.bateria = bateria;

}

public Robo(String nome, String dtFabr, int bateria) {

this.nome = nome;

this.dtFabr = dtFabr;

this.bateria = bateria;

}

public int tiraBateria() {

this.setBateria(this.getBateria() - 1);

return bateria;

}

public double soma(double numero1, double numero2) {

tiraBateria();

return numero1 + numero2;

}

public double multiplicacao(double numero1, double numero2) {

tiraBateria();

return numero1 \* numero2;

}

public double subtracao(double numero1, double numero2) {

tiraBateria();

return numero1 - numero2;

}

public double divisao(double numero1, double numero2) {

tiraBateria();

return numero1 / numero2;

}

}

## 2.2 Main.java

package io.github.eldemonstro.robo;

import java.util.\*;

import javax.swing.JOptionPane;

/\*\*

\* @author: Fabricio Bezerra (github.com/eldemontro)

\*

\*/

public class Main {

public static List<Robo> robos = new ArrayList<>();

public static String joinString(String[] array, String middleChar) {

String completeJoin = "";

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

completeJoin += array[i];

completeJoin += middleChar;

}

return completeJoin;

}

public static String[] listarRobos() {

String[] listaRobos = new String[robos.size()];

for (int i = 0; i < robos.size(); i++) {

listaRobos[i] = i + " - " + robos.get(i).getNome();

}

return listaRobos;

}

public static void realizarOp(int roboIndex, int op) {

int n1 = 0, n2 = 0;

if (op != 6) {

String numero1 = JOptionPane.showInputDialog("Digite o numero 1");

try {

n1 = Integer.parseInt(numero1);

} catch (NumberFormatException exception) {

JOptionPane.showMessageDialog(null, "Numero invalido");

return;

}

String numero2 = JOptionPane.showInputDialog("Digite o numero 2");

try {

n2 = Integer.parseInt(numero1);

} catch (NumberFormatException exception) {

JOptionPane.showMessageDialog(null, "Numero invalido");

return;

}

}

switch (op) {

case 2:

JOptionPane.showMessageDialog(null, "O resultado é: " + robos.get(roboIndex).soma(n1, n2));

break;

case 3:

JOptionPane.showMessageDialog(null, "O resultado é: " + robos.get(roboIndex).subtracao(n1, n2));

break;

case 4:

JOptionPane.showMessageDialog(null, "O resultado é: " + robos.get(roboIndex).multiplicacao(n1, n2));

break;

case 5:

JOptionPane.showMessageDialog(null, "O resultado é: " + robos.get(roboIndex).divisao(n1, n2));

break;

case 6:

JOptionPane.showMessageDialog(null, "A bateria é: " + robos.get(roboIndex).getBateria());

break;

default:

break;

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

String opcao;

int escolha;

String[] menu = {"1-Criar", "2-Soma", "3-Subtracao", "4-Multiplicacao", "5-Divisao", "6-Bateria", "7-Sair"};

boolean sair = false;

while (!sair) {

String menuMsg = "Robo\n" + joinString(menu, "\n");

opcao = JOptionPane.showInputDialog(menuMsg);

if (opcao == null) {

sair = true;

continue;

}

try {

escolha = Integer.parseInt(opcao);

} catch (NumberFormatException exception) {

JOptionPane.showMessageDialog(null, "Numero invalido");

continue;

}

switch (escolha) {

case 1:

String nome = JOptionPane.showInputDialog("Digite o nome do robo");

String dataDeFabricacao = JOptionPane.showInputDialog("Digite a data de fabricacão do robo: no formato DD/MM/AAAA");

String bateriaString = JOptionPane.showInputDialog("Digite o nivel de bateria do robo, deve ser um numero");

int bateria;

try {

bateria = Integer.parseInt(bateriaString);

} catch (NumberFormatException exception) {

JOptionPane.showMessageDialog(null, "Bateria invalida");

break;

}

Robo robo = new Robo(nome, dataDeFabricacao, bateria);

robos.add(robo);

JOptionPane.showMessageDialog(null, "Criado robo: " + robo.getNome() + " em " + robo.getDtFabr()

+ " contendo: " + robo.getBateria() + " de bateria");

break;

case 2:

case 3:

case 4:

case 5:

case 6:

if (robos.size() < 1) {

JOptionPane.showMessageDialog(null, "Crie um robo primeiro!");

break;

}

String escolhaRoboString = JOptionPane.showInputDialog("Escolha um robo:\n" + joinString(listarRobos(), "\n"));

int escolhaRobo;

try {

escolhaRobo = Integer.parseInt(escolhaRoboString);

} catch (NumberFormatException exception) {

JOptionPane.showMessageDialog(null, "Robo invalido");

break;

}

if (escolhaRobo < 0 || escolhaRobo >= robos.size()) {

JOptionPane.showMessageDialog(null, "Robo invalido");

break;

}

realizarOp(escolhaRobo, escolha);

break;

case 7:

sair = true;

break;

default:

JOptionPane.showMessageDialog(null, "Numero invalido");

break;

}

}

}

}