package calculator.com.math;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class CalculatorGUI extends JFrame {

private JTextField inputField;

private JButton[] buttons;

private JButton equalsButton;

private JButton clearButton;

private double firstNumber;

private String operator;

public CalculatorGUI() {

setTitle("Calculator");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(300, 400);

setLayout(new BorderLayout());

inputField = new JTextField();

inputField.setPreferredSize(new Dimension(300, 50));

inputField.setFont(new Font("Arial", Font.PLAIN, 24));

add(inputField, BorderLayout.NORTH);

JPanel buttonPanel = new JPanel();

buttonPanel.setLayout(new GridLayout(5, 4));

buttons = new JButton[16];

buttons[0] = new JButton("7");

buttons[1] = new JButton("8");

buttons[2] = new JButton("9");

buttons[3] = new JButton("/");

buttons[4] = new JButton("4");

buttons[5] = new JButton("5");

buttons[6] = new JButton("6");

buttons[7] = new JButton("\*");

buttons[8] = new JButton("1");

buttons[9] = new JButton("2");

buttons[10] = new JButton("3");

buttons[11] = new JButton("-");

buttons[12] = new JButton("0");

buttons[13] = new JButton(".");

buttons[14] = new JButton("=");

buttons[15] = new JButton("+");

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

buttonPanel.add(buttons[i]);

int finalI = i;

buttons[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String text = buttons[finalI].getText();

if (!text.equals("=")) {

inputField.setText(inputField.getText() + text);

} else {

evaluateExpression();

}

}

});

}

add(buttonPanel, BorderLayout.CENTER);

setVisible(true);

}

private void evaluateExpression() {

String expression = inputField.getText();

try {

double result = Calculator.evaluate(expression);

inputField.setText(String.valueOf(result));

} catch (ArithmeticException e) {

inputField.setText("Error");

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

public void run() {

new CalculatorGUI();

}

});

}

}

class Calculator {

public static double evaluate(String expression) {

try {

return new Object() {

int pos = -1, ch;

void nextChar() {

ch = (++pos < expression.length()) ? expression.charAt(pos) : -1;

}

boolean eat(int charToEat) {

while (ch == ' ') nextChar();

if (ch == charToEat) {

nextChar();

return true;

}

return false;

}

double parse() {

nextChar();

double x = parseExpression();

if (pos < expression.length()) throw new RuntimeException("Unexpected: " + (char) ch);

return x;

}

double parseExpression() {

double x = parseTerm();

for (; ; ) {

if (eat('+')) x += parseTerm();

else if (eat('-')) x -= parseTerm();

else return x;

}

}

double parseTerm() {

double x = parseFactor();

for (; ; ) {

if (eat('\*')) x \*= parseFactor();

else if (eat('/')) x /= parseFactor();

else return x;

}

}

double parseFactor() {

if (eat('+')) return parseFactor();

if (eat('-')) return -parseFactor();

double x;

int startPos = this.pos;

if (eat('(')) {

x = parseExpression();

eat(')');

} else if ((ch >= '0' && ch <= '9') || ch == '.') {

while ((ch >= '0' && ch <= '9') || ch == '.') nextChar();

x = Double.parseDouble(expression.substring(startPos, this.pos));

} else {

throw new RuntimeException("Unexpected: " + (char) ch);

}

return x;

}

}.parse();

} catch (Exception e) {

throw new ArithmeticException("Invalid Expression");

}

}

}

