```java
package main;

/*
 * JCalculator.java
 *
 * Pier Donini, 9 Jan 2004.
 * edited by Minder Valentin and Bron Sacha on Dec 11 2014.
 */

import javax.swing.*;

import operator.*;

import java.awt.*;
import java.awt.event.*;

public class JCalculator extends JFrame {
    // Tableau representant une pile vide
    private final String[] empty = { "< empty stack >" };

    // Zone de texte contenant la valeur introduite ou resultat courant
    private final JTextField jNumber = new JTextField("0");

    // Composant liste representant le contenu de la pile
    private final JList jStack = new JList(empty);

    // Contraintes pour le placement des composants graphiques
    private final GridBagConstraints constraints = new GridBagConstraints();

    /*
     * Mise a jour de l'interface apres une operation (jList et jStack)
     */
    private void update() {
        // Modifier une zone de texte, JTextField.setText(string nom)
        // Modifier un composant liste, JList.setListData(Object[] tableau)
        jNumber.setText(State.getInstance().getValueString());
        Object [] stack = State.getInstance().getStackState();
        if (stack.length == 0) {
            stack = empty;
        }
        jStack.setListData(stack);
    }

    /*
     * Ajout d'un bouton dans l'interface et de l'operation associee, instance
     * de la classe Operation, possedeant une methode execute()
     */
    private void addOperatorButton(String name, int x, int y, Color color,
            final Operator operator) {
        JButton b = new JButton(name);
        b.setForeground(color);
        constraints.gridx = x;
        constraints.gridy = y;
        getContentPane().add(b, constraints);

        b.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                operator.execute();
                update();
            }
        });
    }

    /*
     * Constructeur
     */
    public JCalculator() {
        super("JCalculator");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(new GridBagLayout());

        // Contraintes des composants graphiques
        constraints.insets = new Insets(3, 3, 3, 3);
        constraints.fill = GridBagConstraints.HORIZONTAL;

        // Nombre courant
        jNumber.setEditable(false);
        jNumber.setBackground(Color.WHITE);
        jNumber.setHorizontalAlignment(JTextField.RIGHT);
        constraints.gridx = 0;
        constraints.gridy = 0;
        constraints.gridwidth = 5;
        getContentPane().add(jNumber, constraints);
        constraints.gridwidth = 1; // reset width

        // Rappel de la valeur en memoire
        addOperatorButton("MR", 0, 1, Color.RED, new MROperator());

        // Stockage d'une valeur en memoire
        addOperatorButton("MS", 1, 1, Color.RED, new MSOperator());

        // Backspace
        addOperatorButton("<=", 2, 1, Color.RED, new BackSpaceOperator());

        // Mise a zero de la valeur courante + suppression des erreurs
        addOperatorButton("CE", 3, 1, Color.RED, new CEOperator());

        // Comme CE + vide la pile
        addOperatorButton("C", 4, 1, Color.RED, new COperator());

        // Boutons 1-9
        for (int i = 1; i < 10; i++)
            addOperatorButton(String.valueOf(i), (i - 1) % 3, 4 - (i - 1) / 3,
                    Color.BLUE, new DigitOperator(i));
        // Bouton 0
        addOperatorButton("0", 0, 5, Color.BLUE, new DigitOperator(0));

        // Changement de signe de la valeur courante
        addOperatorButton("+/-", 1, 5, Color.BLUE, new SignOperator());

        // Operateur point (chiffres apres la virgule ensuite)
        addOperatorButton(".", 2, 5, Color.BLUE, new DotOperator());

        // Operateurs arithmetiques a deux operandes: /, *, -, +
        addOperatorButton("/", 3, 2, Color.RED, new DivOperator());
        addOperatorButton("*", 3, 3, Color.RED, new TimesOperator());
        addOperatorButton("-", 3, 4, Color.RED, new MinusOperator());
        addOperatorButton("+", 3, 5, Color.RED, new PlusOperator());

        // Operateurs arithmetiques a un operande: 1/x, x^2, Sqrt
        addOperatorButton("1/x", 4, 2, Color.RED, new OneOverXOperator());
        addOperatorButton("x^2", 4, 3, Color.RED, new SquareOperator());
        addOperatorButton("Sqrt", 4, 4, Color.RED, new SqrtOperator());

        // Entree: met la valeur courante sur le sommet de la pile
        addOperatorButton("Ent", 4, 5, Color.RED, new EnterOperator());

        // Affichage de la pile
        JLabel jLabel = new JLabel("Stack");
        jLabel.setFont(new Font("Dialog", 0, 12));
        jLabel.setHorizontalAlignment(JLabel.CENTER);
        constraints.gridx = 5;
        constraints.gridy = 0;
        getContentPane().add(jLabel, constraints);

        jStack.setFont(new Font("Dialog", 0, 12));
        jStack.setVisibleRowCount(8);
        JScrollPane scrollPane = new JScrollPane(jStack);
        constraints.gridx = 5;
        constraints.gridy = 1;
        constraints.gridheight = 5;
        getContentPane().add(scrollPane, constraints);
        constraints.gridheight = 1; // reset height

        setResizable(false);
        pack();
    }

    /*
     * main()
     */
    public static void main(String args[]) {
        new JCalculator().setVisible(true);
    }
}

```