

```
package gui;

import generator.ObjectCreated;
import randomizer.Randomizer;
import generator.TestInfo;

import java.awt.Component;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.util.ArrayList;

import javax.swing.Box;
import javax.swing.BoxLayout;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

/**
 * Class that shows and manage the GUI of creating a new Object
 * @author Sergio Alcocer
 */
public class CreateObjectUI extends JFrame{

    private static final long serialVersionUID = 1L;

    private Icon icon;
    private ArrayList<String> params;
    private JLabel lblConstructorPart1, lblConstructorPart2, lblConstructorPart3;
    private JTextField txtParamValue, txtObjectName;
    private JButton jbMore, jbLess, jbWand;
    private ast.Method method;
    private int currentParam;

    /**
```

```
* Constructor of the class
* @param _method Constructor that is being used (ast.Method)
*/
public CreateObjectUI(ast.Method _method) {
    lblConstructorPart1 = new JLabel();
    lblConstructorPart2 = new JLabel();
    lblConstructorPart3 = new JLabel();
    txtParamValue = new JTextField();
    txtObjectName = new JTextField();
    jbMore = new JButton();
    jbLess = new JButton();
    jbWand = new JButton();
    method = _method;
    params = new ArrayList<String>(method.getParamNumber());
    icon = new ImageIcon("wand.gif");
    init();
}

private void init() {
    setDefaultCloseOperation(DISPOSE_ON_CLOSE);

    for (int i = 0; i < method.getParamNumber(); i++) {
        params.add("");
    }

    this.addWindowListener(new WindowListener() {
        public void windowActivated(WindowEvent arg0) {}
        public void windowClosed(WindowEvent arg0) {
            MainFrame.getInstance().unlock();
            MainFrame.getInstance().ToFront();
        }
        public void windowClosing(WindowEvent arg0) {}
        public void windowDeactivated(WindowEvent arg0) {}
        public void windowDeiconified(WindowEvent arg0) {}
        public void windowIconified(WindowEvent arg0) {}
        public void windowOpened(WindowEvent arg0) {}
    });

    Box mainBox = new Box(BoxLayout.Y_AXIS);
    Box constructorBox = new Box(BoxLayout.X_AXIS);
    lblConstructorPart2.setForeground(new java.awt.Color(200,0,0));
```

```
lblConstructorPart1.setFont(new java.awt.Font("Courier", java.awt.Font.PLAIN, 20));
lblConstructorPart2.setFont(new java.awt.Font("Courier", java.awt.Font.ITALIC, 20));
lblConstructorPart3.setFont(new java.awt.Font("Courier", java.awt.Font.PLAIN, 20));

constructorBox.add(lblConstructorPart1);
constructorBox.add(lblConstructorPart2);
constructorBox.add(lblConstructorPart3);
mainBox.add(constructorBox);
mainBox.add(Box.createGlue());

Box constructorInfoBox = new Box(BoxLayout.X_AXIS);
jblLess.addActionListener(new ActionListener () {
    public void actionPerformed(ActionEvent arg0) {
        params.set(currentParam, txtParamValue.getText());
        currentParam--;
        updateConstructorLabel();
    }
});
jblMore.addActionListener(new ActionListener () {
    public void actionPerformed(ActionEvent arg0) {
        params.set(currentParam, txtParamValue.getText());
        if(jblMore.getText().equals("OK")){
            saveObject();
        }else{
            currentParam++;
            updateConstructorLabel();
        }
    }
});
jblWand.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0){
        autoGenerateInputs();
    }
});
jblWand.setIcon(icon);
jblWand.setMinimumSize(new java.awt.Dimension(26,26));
jblWand.setMaximumSize(new java.awt.Dimension(26,26));
jblWand.setPreferredSize(new java.awt.Dimension(26,26));
```

```
txtParamValue.setMinimumSize(new java.awt.Dimension(80,27));
txtParamValue.setMaximumSize(new java.awt.Dimension(2000,27));
txtParamValue.setPreferredSize(new java.awt.Dimension(200,27));
constructorInfoBox.add(jbLess);
constructorInfoBox.add(txtParamValue);
constructorInfoBox.add(jbMore);
constructorInfoBox.add(jbWand);
mainBox.add(constructorInfoBox);
mainBox.add(Box.createGlue());

Box objectNameBox = new Box(BoxLayout.X_AXIS);
JLabel lblObjectName = new JLabel("Object Name: ");
objectNameBox.add(lblObjectName);
txtObjectName.setMinimumSize(new java.awt.Dimension(80,27));
txtObjectName.setMaximumSize(new java.awt.Dimension(2000,27));
txtObjectName.setPreferredSize(new java.awt.Dimension(200,27));
objectNameBox.add(txtObjectName);
mainBox.add(objectNameBox);

setContentPane(mainBox);
setVisible(true);
currentParam = 0;
updateConstructorLabel();
pack();

}

private void updateConstructorLabel() {
    int paramNumber = method.getParamNumber();
    String lbl1, lbl2, lbl3;
    lbl1 = new String(method.getReturnedType() + "( ");
    lbl2 = new String();
    lbl3 = new String();

    jbLess.setEnabled(currentParam != 0);
    jbLess.setText("<<");
    jbMore.setText(((currentParam == method.getParamNumber()-1)?"OK":">>"));
```

```
txtParamValue.setText(params.get(currentParam));

for (int i = 0; i < paramNumber; i++){
    if (currentParam > i)
        lbl1 += method.getParam(i) + " ";
    else if(currentParam == i)
        lbl2 += method.getParam(i) + " ";
    else
        lbl3 += method.getParam(i) + " ";
}

lbl3 += ")";

lblConstructorPart1.setText(lbl1);
lblConstructorPart2.setText(lbl2);
lblConstructorPart3.setText(lbl3);

}

private void saveObject(){
    if (!txtObjectName.getText().equals("")){
        if (!txtObjectName.getText().contains(" ")){
            if(!TestInfo.getInstance().contains(txtObjectName.getText())){
                boolean allFilled = true;
                for (int i = 0; i < params.size() && allFilled; i++){ // check that all the parameters are filled
                    String param = params.get(i);
                    if(param.equals("") || param.charAt(0) == ' ' || param.charAt(param.length()-1) == ' '){
                        allFilled = false;
                    }
                }
                if (allFilled){
                    ObjectCreated theNew = new ObjectCreated(txtObjectName.getText(), params);
                    TestInfo.getInstance().addObject(theNew);
                    dispose();
                }else{
                    msgBox("All the parameters must be filled, and cannot start or end with <space>");
                }
            }else{

```

```
        msgBox("There is another object with that name");
    }
    }else{
        msgBox("Not space characters are allowed");
    }
    }else{
        msgBox("Not empty name is allowed");
    }
}

private void msgBox(String msg){
    JOptionPane.showMessageDialog(((Component) this), msg);
}

private void autoGenerateInputs(){
    boolean error = false;
    Object read;
    for (int i = 0; i < method.getParamNumber(); i++){
        String paramType = method.getParam(i);
        read = Randomizer.getInstance().getRandom(paramType);
        if (read == null){
            error = true;
        }else{
            params.set(i, read.toString());
        }
    }
    if (error){
        msgBox ("One or more types are not in the Randomizer. \r\nFor that reason they haven't been set");
    }
    txtParamValue.setText(params.get(currentParam));
}
}
```