Tutorial - Week 08 5COSC019W – Object Oriented Programming – Java

Implement GUI App

14 - 15 November

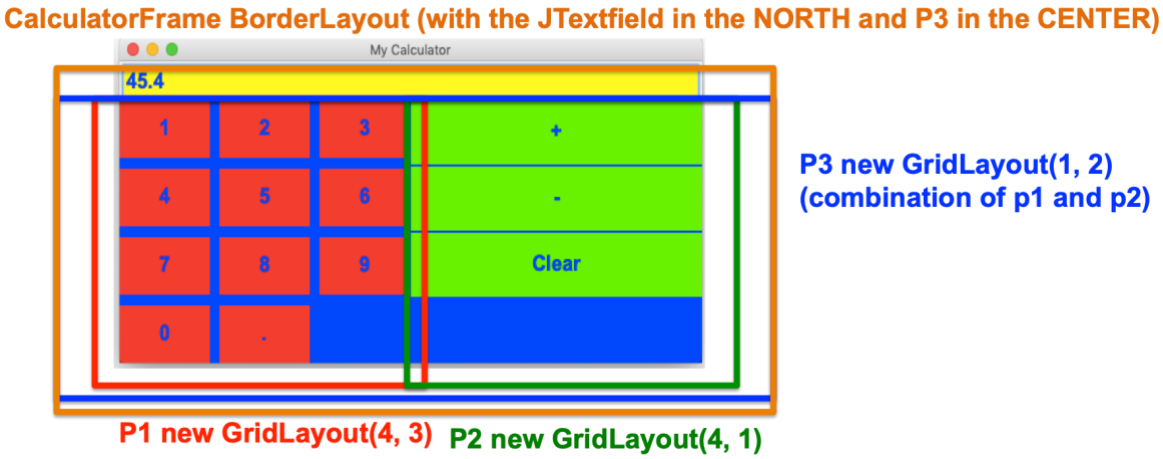
GUI application

1) Write a Swing GUI application called “My Calculator” as shown in the figure below.
   In this exercise you need to use a combination of Layouts in order to arrange the components as shown.

Some Guidance (please refer also to the “Microwave” exercise in the lecture week 07). The user interface has:

- JButtons
- JTextField

They are organised using different panels with different layouts.



Note: Swing Components are kept in package javax.swing. They begin with a prefix "J", e.g., JButton, JLabel, JFrame.

```
import java.awt.*;       // Using AWT's layouts
import javax.swing.*;    // Using Swing components and containers
```

Write a class CalculatorFrame that extends JFrame:

```
public class CalculatorFrame extends JFrame{

    // Constructor to setup UI componentspublic CalculatorFrame () {

      // set here the layout for different panels

      // here you need to implement your code
      //…
}

public static void main(String[] args) {
        // TODO code application logic here
        CalculatorFrame myCalculator = new CalculatorFrame();
        myCalculator.setVisible(true);
        myCalculator.setSize(600, 300);
        myCalculator.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```

Implement the Constructor

2) Set the background color of your frame and your buttons, using `.setBackground(Color c)` method.

3) Change the font of the components using the method `.setFont(Font f);`

   where Font f is an object of type Font, e.g. `new Font("SansSerif ", Font.BOLD, 16);`

4) JTable

Implement a class **Person** and the subclasses **Teacher** and **Student**. The classes should hold information aboutthe *name*, the *date of birth* and include appropriate set and get methods. In particular:

- The **Teacher** class should include appropriate methods and hold information about the *salary* .
- The **Student** class should include methods and information about *ID* and the *Course* they are studying.
- Implement a class **Date** to represent the *date of birth* of each person.
- You can reuse classes that you previously implemented during tutorials.

Display in a JTable the information for each person. Color the table cell in green if it is a Teacher and in blue if it is a student.

Hint 1:

Create a custom TableModel that holds the data and extends the `AbstractTableModel`.

```
public class PersonTableModel extends AbstractTableModel{


    private String[] columnNames = {"Name","Date of Birth","Type"};
    private ArrayList<Person> list;

    public PersonTableModel(ArrayList<Person> personList){
        list = personList;
```

```
    }

    @Override
    public int getRowCount() {
    return list.size();
    }

    @Override
    public int getColumnCount() {
        return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        Object temp = null;
        if (columnIndex == 0) {
         temp = list.get(rowIndex).getName();
      }
      lse if (columnIndex == 1) {
         temp = list.get(rowIndex).getDOB().getDate();
      }
      else if (columnIndex == 2) {
          if(list.get(rowIndex) instanceof Teacher)
            temp = "Teacher";
          else
            temp = "Student";
      }
      return temp;
    }

    // needed to show column names in JTable
   public String getColumnName(int col) {
      return columnNames[col];
   }
}
```

Override the abstract methods of AbstractTableModel

Check the instance type and set the value to be Teacher or Student

Hint 2:

In order to color a specific cell you need to set a cellRender for your table and override the `getTableCellRenderComponent(..)` method.

If `myTable` is an instance of your `JTable` then you will have to implement the following code:

```
// color code the cell indicating the type of person: if it is a teacher is green, if student
blue

myTable.getColumnModel().getColumn(2).setCellRenderer(new DefaultTableCellRenderer(){


   Color originalColor = null;

  @Override
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected,
  boolean hasFocus, int row, int column) {


         Component renderer =
super.getTableCellRendererComponent(table,  value,
isSelected, hasFocus, row, column);

          // check the type and set the render accordingly
            if (value == "Teacher") {
                renderer.setBackground(Color.GREEN);
            } else {
                renderer.setBackground(Color.BLUE);
            }

            return renderer;
        }
});
```

Set a DefaultTableCellRender

Column 2 because it is the column showing the Type

Provide the implementation for this method

If the type is Teacher set the cell background to green, otherwise to blue