Table of Contents

[Introduction 2](#_Toc79173121)

[Main classes 2](#_Toc79173122)

[NewEnvironment 2](#_Toc79173123)

[Panel 2](#_Toc79173124)

[workingGUI 2](#_Toc79173125)

[Panels classes 3](#_Toc79173126)

[Frontpanel 3](#_Toc79173127)

[HolidayPlanPanel 4](#_Toc79173128)

[Policypanel 5](#_Toc79173129)

[SelectionPanel 6](#_Toc79173130)

[RoWpanel 7](#_Toc79173131)

[HazarDpanel & WinteRpanel 8](#_Toc79173132)

[TravellerSpanel 8](#_Toc79173133)

[Displaypanel 10](#_Toc79173134)

[Extra Files 11](#_Toc79173135)

[DataContainer 11](#_Toc79173136)

[PictureJPGforpanel 12](#_Toc79173137)

[RedirectingOutputStrmforWorkingGUI 12](#_Toc79173138)

[Tests 12](#_Toc79173139)

[Test 1 12](#_Toc79173140)

[Test 2 15](#_Toc79173141)

[Test 3 19](#_Toc79173142)

[Test 4 – crash test 20](#_Toc79173143)

[Conclusion 25](#_Toc79173144)

# Introduction

WorkingGUI was created for the portfolio to improve my own skills and to demonstrate some of my abilities. Working GUI is a development of the OOPAssigm2MHiszpanski project, which was the final exam for Object Oriented Programming. The program itself is started using the NewEnvironment file. The keynote was to create a fully functioning user interface based on the assumptions of the Holiday Plan Examination. I started to develop the project from the beginning of July to the beginning of August. Due to the fact that I still work as a HGV driver and I spend 5 or even 6 days in England, I could not spend more time and correct all the errors. I will continue to develop this project as there are unanswered questions.

# Main classes

## NewEnvironment

The class that starts the program.

## Panel

|  |
| --- |
| abstract - Panel extends JPanel implements ActionListener |
| #frame:JFrame  #dataContainer:DataContainer |
| +Panel(JFrame,DataContainer)  +Panel  +actionPerormed(ActionEvent)  +setVisible:void |

An abstract class which is a JPanel extension and will implement an Action Listener.

## workingGUI

The working GUI class is a frame and an environment where the designed activities follow one another. It has been designed in such a way as to meet the minimum expectations. By using the actionPerformed method, subsequent panels appear in the correct order.

|  |
| --- |
| workingGUI extends Jframe implements ActionListener |
| -frame: JFrame  -HPpanel: HolidayPlanPanel  -Fpanel: Frontpanel  -Spanel: Selectionpanel  -dataContainer: DataContainer  -RoWpanel, RowForHazardpanel, RoWfor WInterpanel : RoWpanel  -TSpanelRow, TSpanelHAZ, TSpanelWIN: TravellerSpanel  -clickCount,number,cyrentnumber:int |
| +workingGUI  +getFrame:void  #Override  +actionPerformed(ActionEvent)  +DisplaypanelOptions:void  +settingsForButtons(TravellerSpanel)  +returnLocation(int):string  +Display:void  // getters setters |

# Panels classes

## Frontpanel

Front panel. The first panel that appears. It also contains an image that was uploaded using the JLabel function.

|  |
| --- |
| Frontpanel extends Panel |
| -button1:JButton  -label,label2:JLabel |
| +Frontpanel(Jframe,DataContainer)  +getButtons:void  +getLabels:void  ++ (extends)  // getters setters |

Graphical user interface

Description automatically generated

## HolidayPlanPanel

HolidayPlanPanel a panel with which data are entered, taking into account the original version of the program. In this panel the button will change to enabled (true) when new data is entered in the selected JTextField.

|  |
| --- |
| HolidayPlanPanel extends Panel |
| -button1:JButton  -label1,label2,label3,label4,label5,label6,label7:JLabel  -field1,field2,field3,field4,field5,field6:JTextField |
| + HolidayPlanPanel (Jframe,DataContainer)  +getFields:void  +getButtons:void  +getLabels:void  + returnDateDoB:string  ++ (extends)  // getters setters |

Graphical user interface

Description automatically generated with medium confidence

## Policypanel

Policypanel a panel with which data are entered, taking into account the original version of the program. In this panel the button will change to enabled (true) when new data is entered in the selected JTextField.

|  |
| --- |
| Policypanel extends Panel |
| -button1,button2:JButton  -label1, label2, label3, label4, label5, labelSmonth, labelSyear, labelEmonth, labelEyear, labelSday, labelEday:JLabel  -field1, field2, field3, field4, field5, fieldSmonth, fieldSyear, fieldEmonth, fieldEyear, fieldSday, fieldEday:JTextField  -panel2:JPanel  -name,date,id,Sdate,Edate:string |
| + Policypanel (Jframe,DataContainer)  +getExtraPanels:void  +addToPanel2:void  +getFields:void  +getButtons:void  +getLabels:void  + returnPolicyDateDoB:string  + returnPolicyStartDate:string  + returnPolicyEndDate:string  ++ (extends)  // getters setters |

Graphical user interface, treemap chart

Description automatically generated

## SelectionPanel

SelectionPanel is a panel through which the user has the possibility to choose what type of policy he would like for himself. When creating this panel, it was assumed that the user will know the differences between individual policies. The point was that with one click the user could make a quick choice.

|  |
| --- |
| SelectionPanel extends Panel |
| -button1,button2,button3:JButton  -label1:JLabel |
| + SelectionPanel (Jframe,DataContainer)  +getButtons:void  +getLabels:void  ++ (extends)  // getters setters |

Chart

Description automatically generated

## RoWpanel

RoWpanel class, the panel in which the JComboBox help the user can choose 'Destination' and choose the number of travellers to be saved as Integer value. For technical reasons, I limited this number from 1 to 6. This value is then used during the constructor of the TravellerSpanel class.

|  |
| --- |
| RoWpanel extends Panel |
| #panel1,panel2,panel3: JPanel  #label1,label4: JLabel  #box1,box4: JComboBox  # worldDestination: String[]  #numberOfTravellers: String[]  #userDestination: String  #number: int  #TSpanel: TravellerSpanel  #button1,button2:JButton |
| +RoWpanel(JFrame,DataContainer)  +RoWpanel(JFrame,DataContainer,String)  +getPanels: void  +getComboBox: void  +getButtons: void  +getLabels: void  ++ (extends)  // getters setters |

Graphical user interface

Description automatically generated

## HazarDpanel & WinteRpanel

HazarDpanel & WinteRpanel classes that have not been used. These classes were an extension of the RoWpanel class. There are incomplete complications with ActionListener for me. I spent a lot of time trying to understand the problem. Adding an ActionListener to buttons from this class in the workingGUI class did not get the expected effect. The funny thing is that these classes are hardly different from each other. Therefore, the solution to the problem was to use a new constructor in the RoWpanel class.

## TravellerSpanel

TravellerS panel class, a panel through which we can enter and save the data of travelers. The lower button in the panel allows you to accept the entered data. The dispaly button is activated when an appropriately pre-selected number is entered. This class uses the for loop and the number of travellers is the limit of this loop. This class contains 3 constructors. The only difference is adding the amount of JComboBox. Each constructor is an equivalent of the policy for which it was issued.

|  |
| --- |
| TravellerSpanel extends Panel |
| -box2,box3:JComboBox  -button1,button2:JButton  - label2,label3,labelTravmonth,labelTravyear:JLabel  - field1,field2,field3,fieldTravmonth,fieldTravyear :JTextField  -Activities:String[]  -userSkills:String[]  -policyname,ID:String  -number,currnetnumber:int |
| + TravellerSpanel (Jframe,DataContainer,RoWpanel,int) - for Rest of World Policy  + TravellerSpanel (Jframe,DataContainer,RoWpanel,int,String) - for Hazard Policy  + TravellerSpanel (Jframe,DataContainer,RoWpanel,int,String,String) - for Winter Policy  +getThisPanel: void  +getComboBox: void  +getButtons: void  +getLabels: void  +getFields: void  +returnPolicyOwnerDoB: String  +addToContainer: void  +setAnYDate: Calendar  +getActivityString: String  +getTravLevelString: String  ++ (extends)  // getters setters |

Panel look using constructor for Rest of World Policy

Graphical user interface

Description automatically generated

Panel look using constructor for Hazard Policy

Graphical user interface

Description automatically generated

Panel look using constructor for Winter Policy

Graphical user interface

Description automatically generated

## Displaypanel

Displaypanel is a panel through which we can see the results. They are displayed in the JTextArea. Output from the console is redirected to the text area. A PrintStream adds functionality to another output stream. Unlike other output streams, a PrintStream never throws an IOException. The User has the choice to accept the Policy, subject to the costs, or to opt out. What will display the data again including cancellations.

|  |
| --- |
| Displaypanel extends Panel |
| -textArea: JTestArea  -panel1,panel2,panel3: JPanel  -label1: JLabel  -button1,button2,button3:JButton |
| + Displaypanel (Jframe,DataContainer)  +getThisPanel: void  +getTextArea: void  +getButtons: void  +getLabels: void  ++ (extends)  // getters setters |

//Standard Output Stream to any user defined value

System.setOut(new PrintStream(new RedirectingOutputStrmForWorkingGUI(this), true));

A picture containing graphical user interface

Description automatically generatedText

Description automatically generated with low confidence

# Extra Files

## DataContainer

DataContainer class whose main task is data mangerization. It is a data container. Some methods have been added to calculate the premium value, to set the appropriate value for the DailyRate, and a method to display the String data in an orderly and scheduled manner.

|  |
| --- |
| DataContainer |
| - ArrayList<String> TravellersNames  - ArrayList<Calendar> TravelelrsDoB  - ArrayList<String> TravellerLevel  - ArrayList<String> TravellerActivity  - String HPid, HPname;  - Calendar HPdateCreated;  - Policy HPpolicy;  - String HPinsuranceStatus;  - String HPdestination;  - double HPcost;  - String policyNum, policyOwnername,status,policyType;  - Calendar PownerDoB, PsDate, PeDate,HPownerDoB;  - String HPownerDoBstring,SDatestring,EDatestring,PownerDoBstring;  - double PdailyRate, premium;  - String ForPolicyholidayPlanID;  - String userDestination;  - int NumberofTravellers;  - final double EquipmnetCover=40;  - double highestRate; |
| + DataContainer  +toString:String  +DisplayTravellersRoW: String  +DisplayTravellerHazard: String  +DisplayTravellersWinter: String  +setAnYDate: Calendar  +calPremiumRoW(double)  +getDailyRateRoW: double  +calPremiumHazard(double)  +calPremiumWinter(double)  +setproperDailyRateWinter: double  // getters setters |

## PictureJPGforpanel

PictureJPGforpanel a class with which we can add any picture to our program, by placing it, for example, in a selected panel. This solution was not finally used, but I wanted to show you a different way of adding photos.

|  |
| --- |
| PictureJPGforpanel extends JPanel |
| -image: BufferedImage |
| + PictureJPGforpanel  @Override  paintComponent(Graphics): void |

## RedirectingOutputStrmforWorkingGUI

RedirectingOutputStrmforWorkingGUI class with which the data stream is redirected from the console to the place indicated in the constructor argument. In this case it is Displaypanel.

|  |
| --- |
| RedirectingOutputStrmforWorkingGUI extends OutputStream |
| -displaypanel: Displaypanel |
| + RedirectingOutputStrmforWorkingGUI (Displaypanel)  @Override  +write(int) throws IOException : void |

# Tests

## Test 1

The test was carried out on the assumption that the data will be entered correctly and logically. Policy 'Winter' was selected during the test. 4 travelers have also been added.

Holiday Plan panel Input

Chart

Description automatically generated with low confidence

Policy panel Input

Graphical user interface, chart, treemap chart

Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generated

A picture containing graphical user interface

Description automatically generated

## Test 2

The test was carried out on the assumption that the data will be entered correctly and logically. Policy 'Hazard' was selected during the test. Two travellers have also been added.

A picture containing text

Description automatically generated

Graphical user interface, treemap chart

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

A picture containing text

Description automatically generated

## Test 3

This test, carried out on the assumption and on the date, will be entered randomly but in order to get a Premium result negative. On this example you can see there are no restrictions for the user when entering dates. This is a big problem as it often leads to bad, incomprehensible results. This is one of the design elements I still need to improve. The project is still open and I hope that by increasing my knowledge it will eliminate errors. By the way, there was a problem with the almost unlimited ability to enter data in JTextField. Please check, for example, the dates of the birthdays Trav 6.

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface, chart

Description automatically generated

## Graphical user interface Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

A picture containing table

Description automatically generated

## Test 4 – crash test

Text

Description automatically generated with low confidenceGraphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

As you can see in the screenshot below, the 'Click to Accept' button is enabled. After entering 'data' for the Trav 1 and pressing the button, the program is suspended.

Graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

At this point, the program crashes. You need to 'manually' close the program.

# Conclusion

WorkingGUI is the first fully (but not completely) user interface that I have built. After overcoming many difficulties and carefully planning the operation of the program, I enclose the current version. The class files are located in :

Assign2MHiszpanski \ OOPAssigm2MHiszpanski \ src \ WorkingGUI

Entering the dates turned out to be the biggest challenge. I set up and the date will be entered as String and then converted to Calendar. The experience I have gained will definitely help me in future projects. I emphasize once again, due to my current work, I could not spend more time than I would like to refine the project. This work is fully prepared by me, and I often look for inspiration at [www.stackoverflow.com](http://www.stackoverflow.com) . I will try to solve the remaining problems in the future.