**COMP1549 Logbook Upload Template 2018/19**

Complete this document for each of the four logbook exercises to include in your final report.

**Basic Information**

|  |  |
| --- | --- |
| 1.1 Student name | **Nkem Akwari** |
| 1.2 Who did you work with? Name and/or id | **Trevor** |
| 1.3 Which lab topic does this document relate to? | Creating software components |
| 1.4 How well do you feel you have done? | * I have completed the exercise and am totally satisfied with my work |
| 1.5 Briefly explain your answer to question 1.4 | **While I have completed the work, I feel like this was the most challenging logbook exercise to do and highlighted a lot of errors I have made that I need to make sure I do not repeat, as well as new methods of coding. The optional tasks highlighted this due to the fact that we weren’t knowledgeable with what was needed and had to look up references and ended up taking up more time that initially thought due to our attempts at trying to figure out what we were doing incorrectly with things such as events.** |

1. **Implementation**

2.1 Annotated screen shots demonstrating what you have achieved.

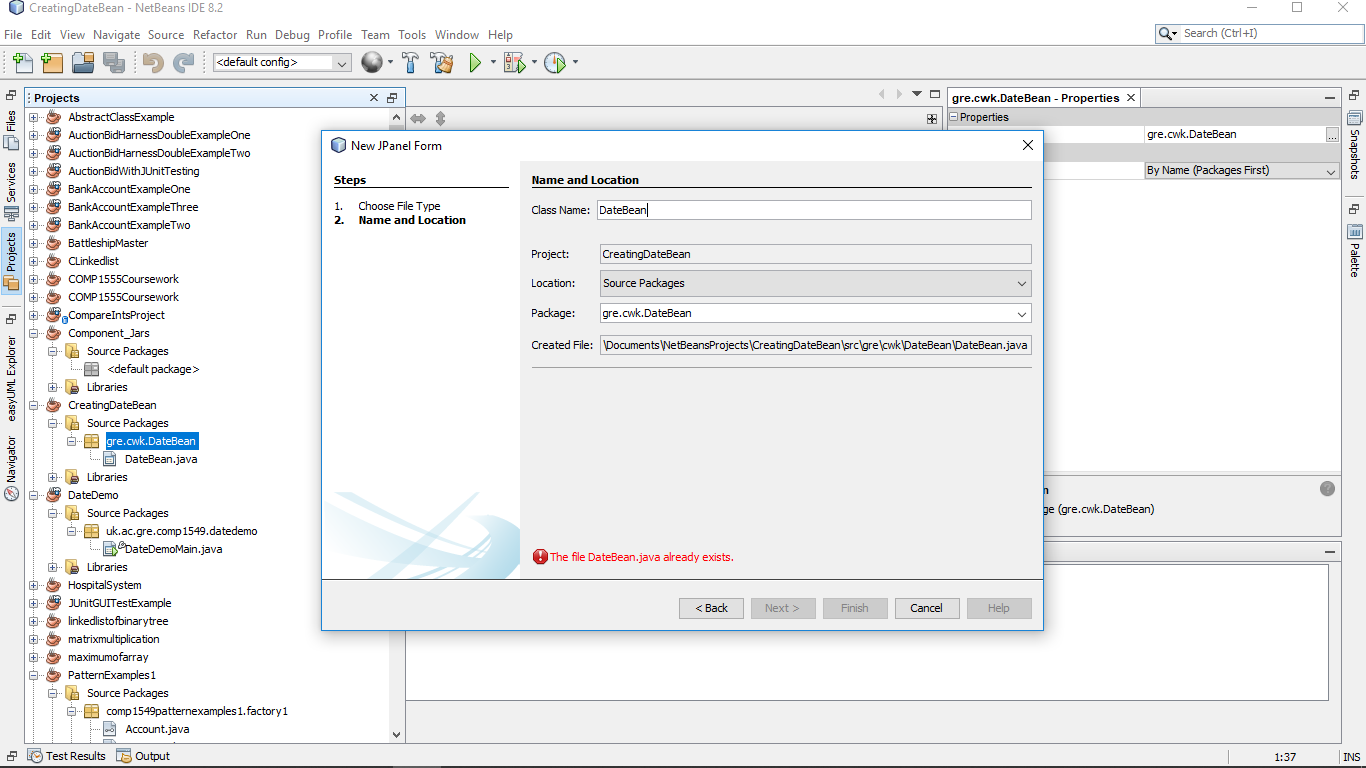


Figure 1Creation of DateBean class for the bean

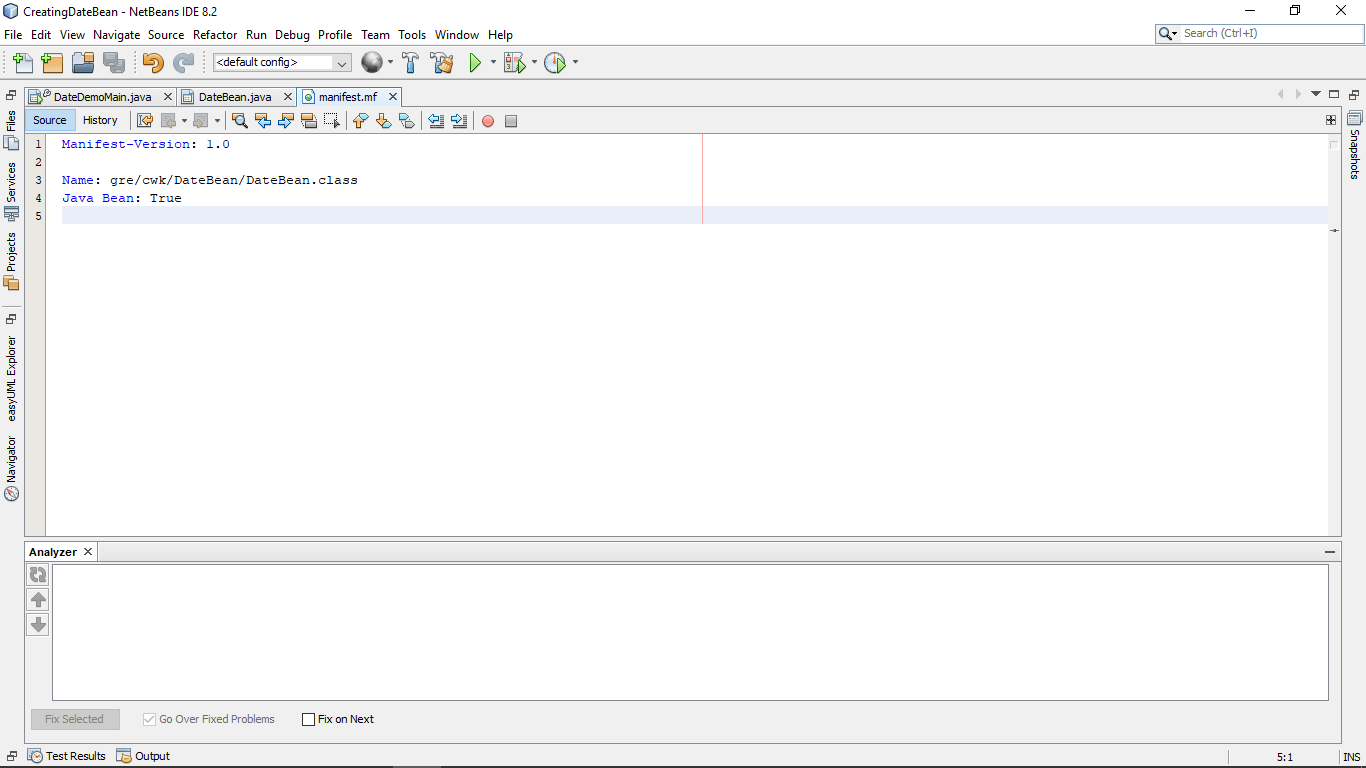


Figure 2Creation of Manifest for DateBean

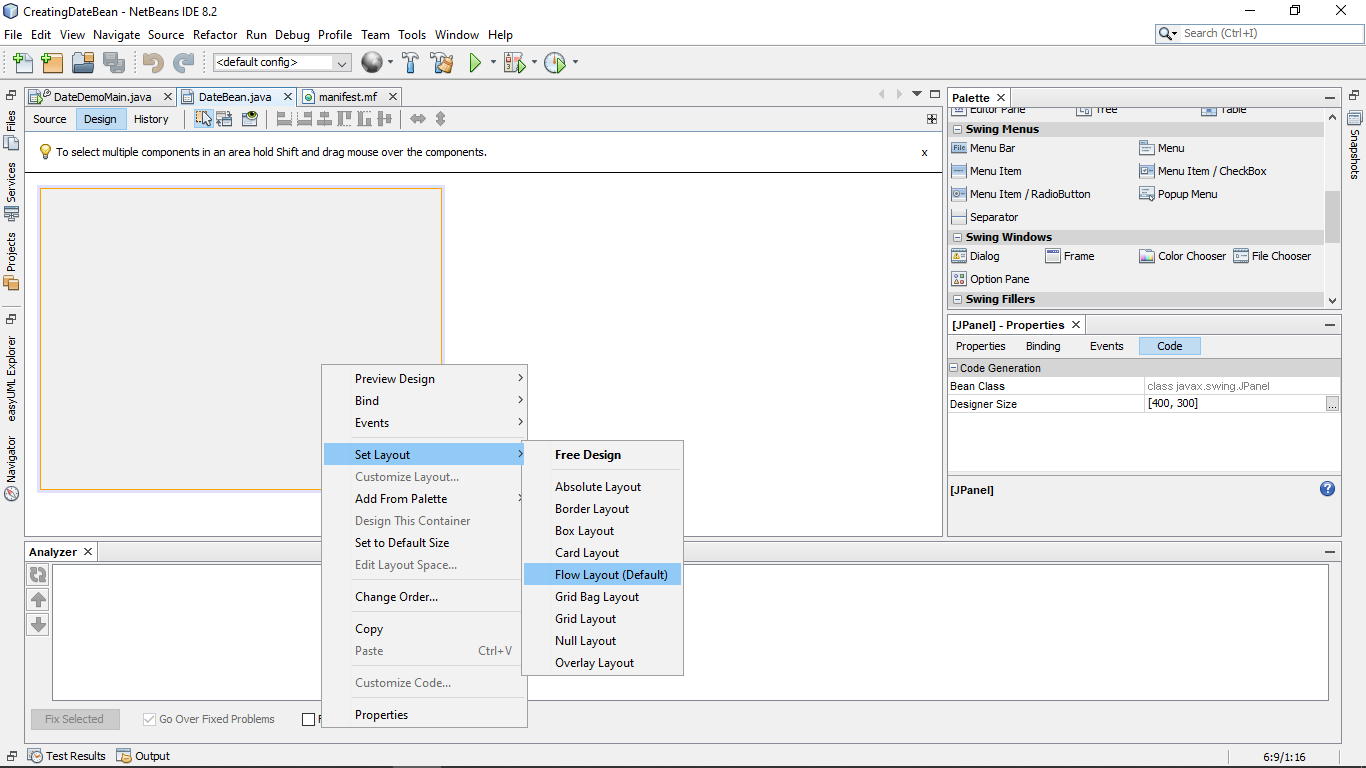


Figure 3Using flowlayout for the layout of the component

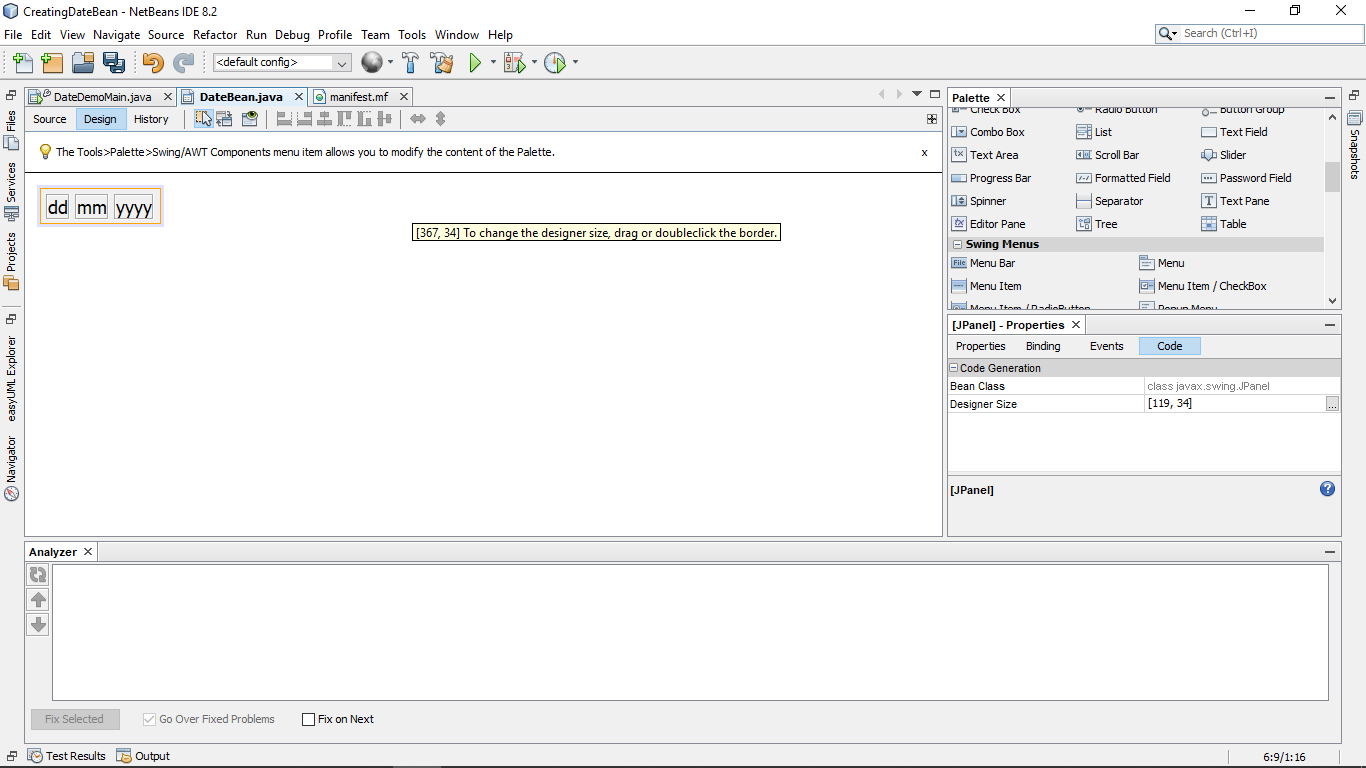
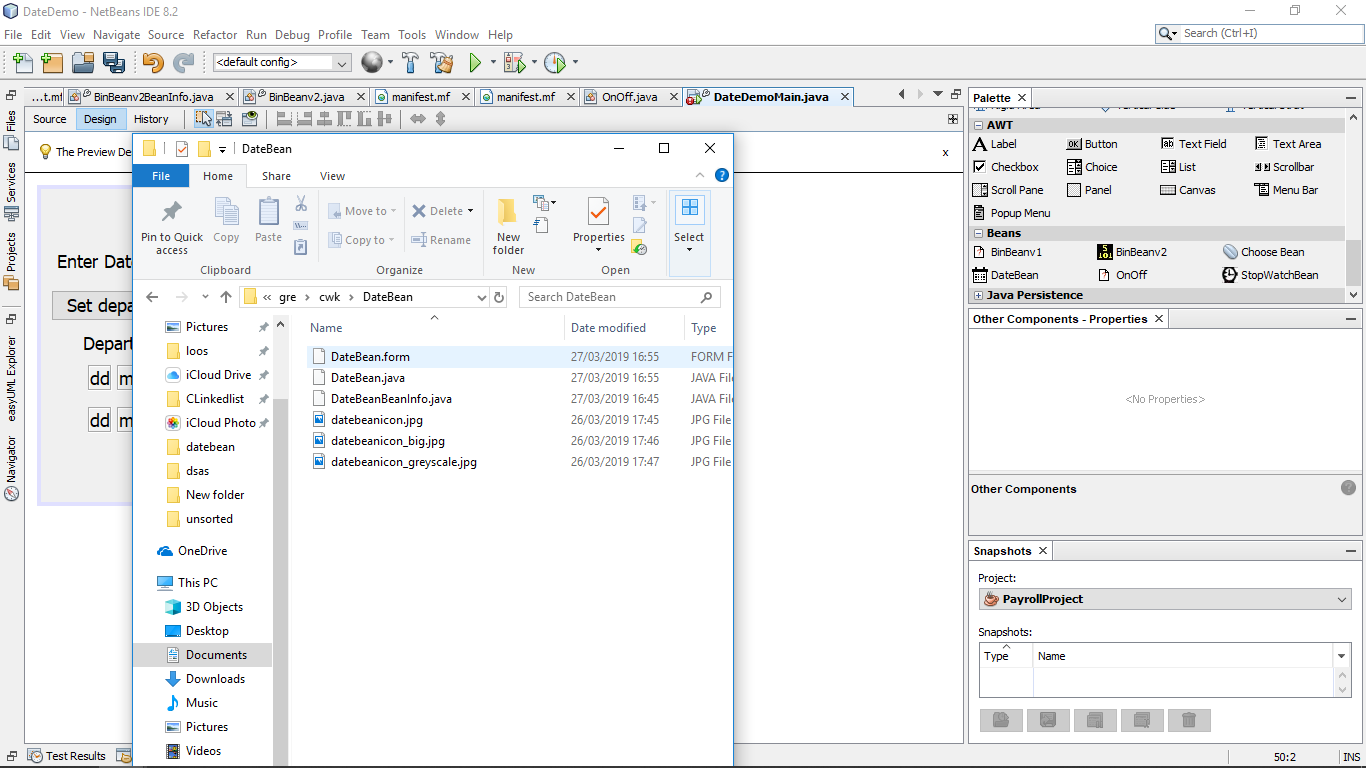


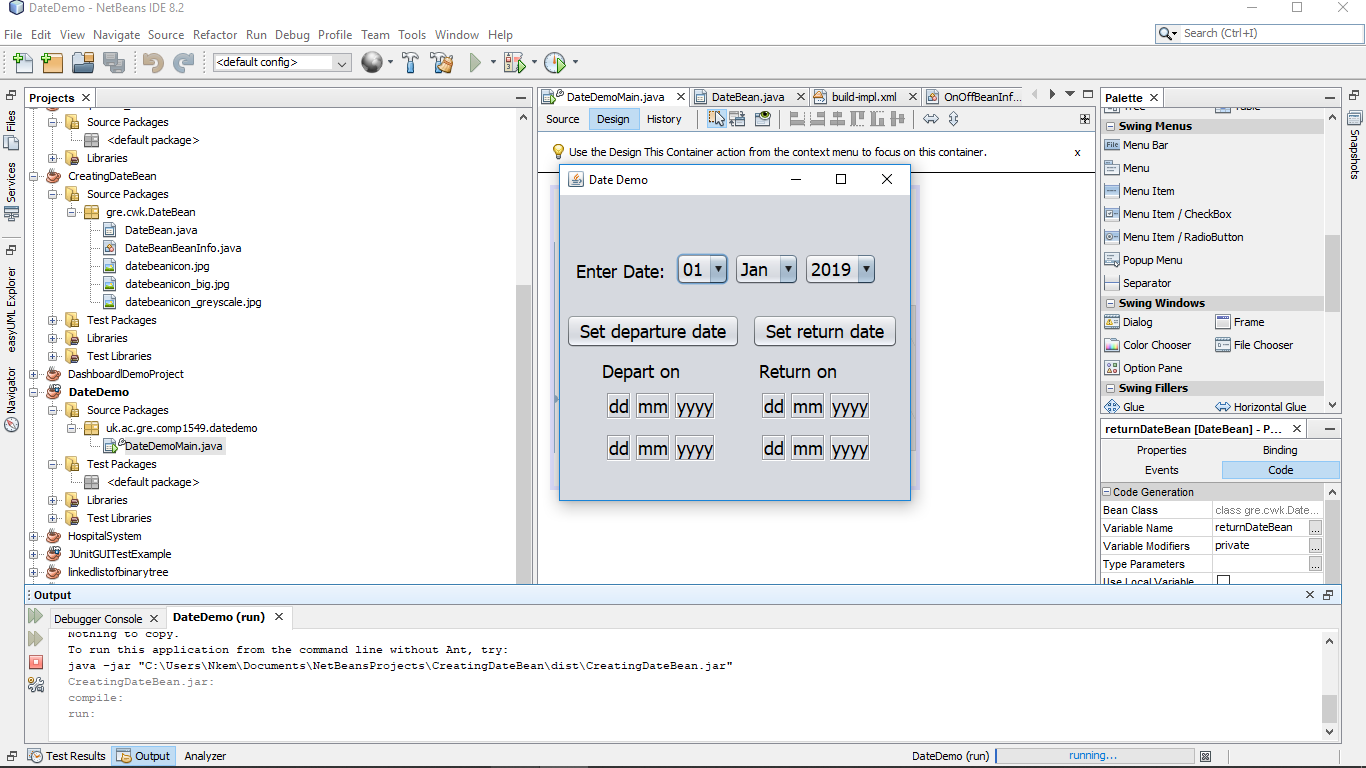
Figure 4pasting the labels in and resizing



Files for the icons need for the bean in its src files.

DateBean Icon in palette

Figure 5Adding icons for the bean



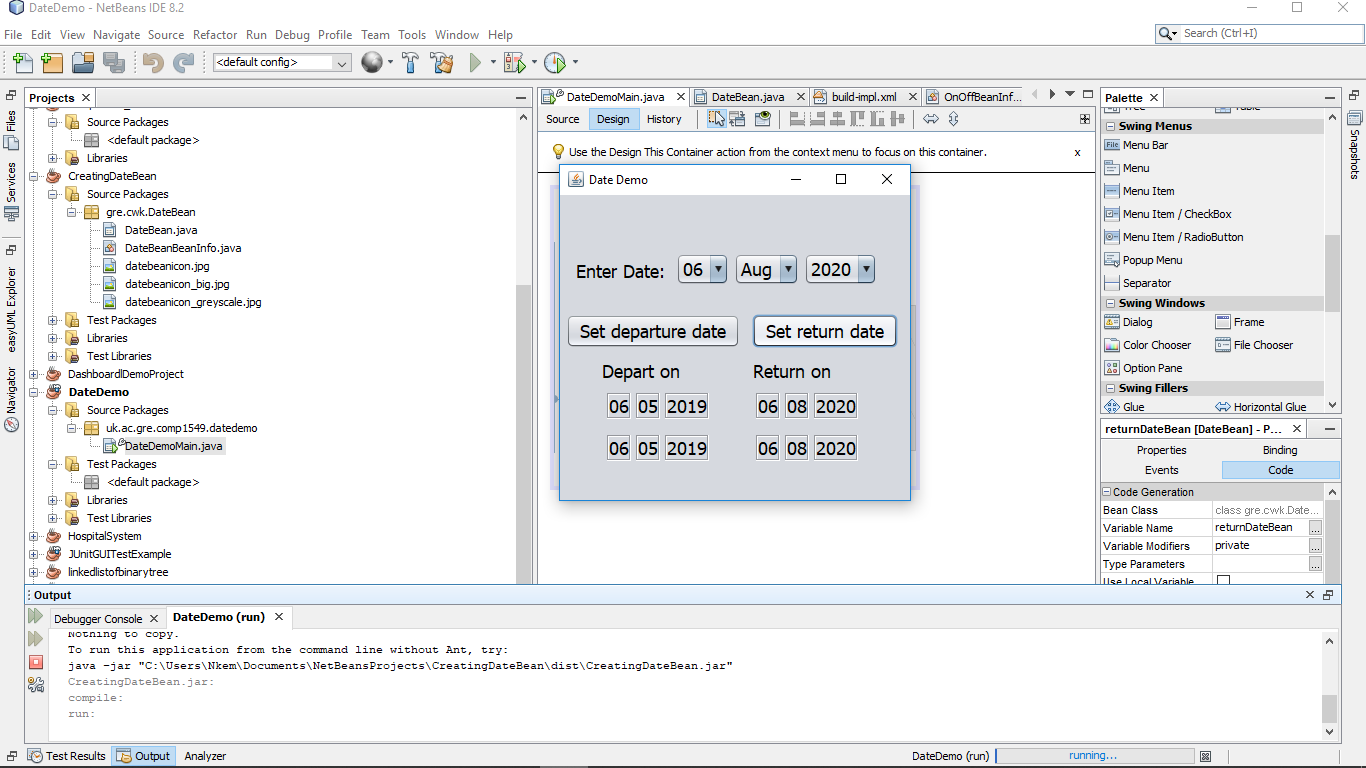
Figure 6Evidence of the code running in full

Figure 7Evidence of the code running 2- look at date/time changing for both the bean(bottom set) and original labels (top)

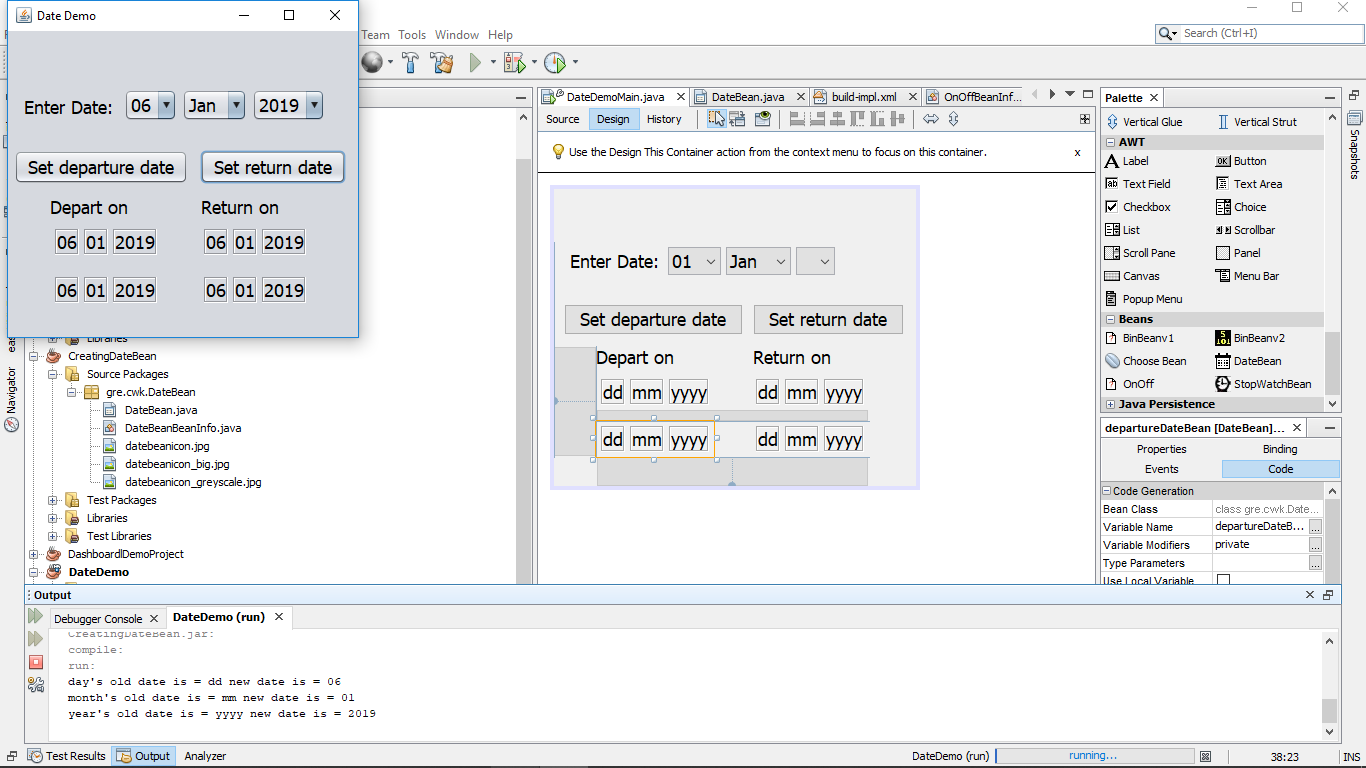
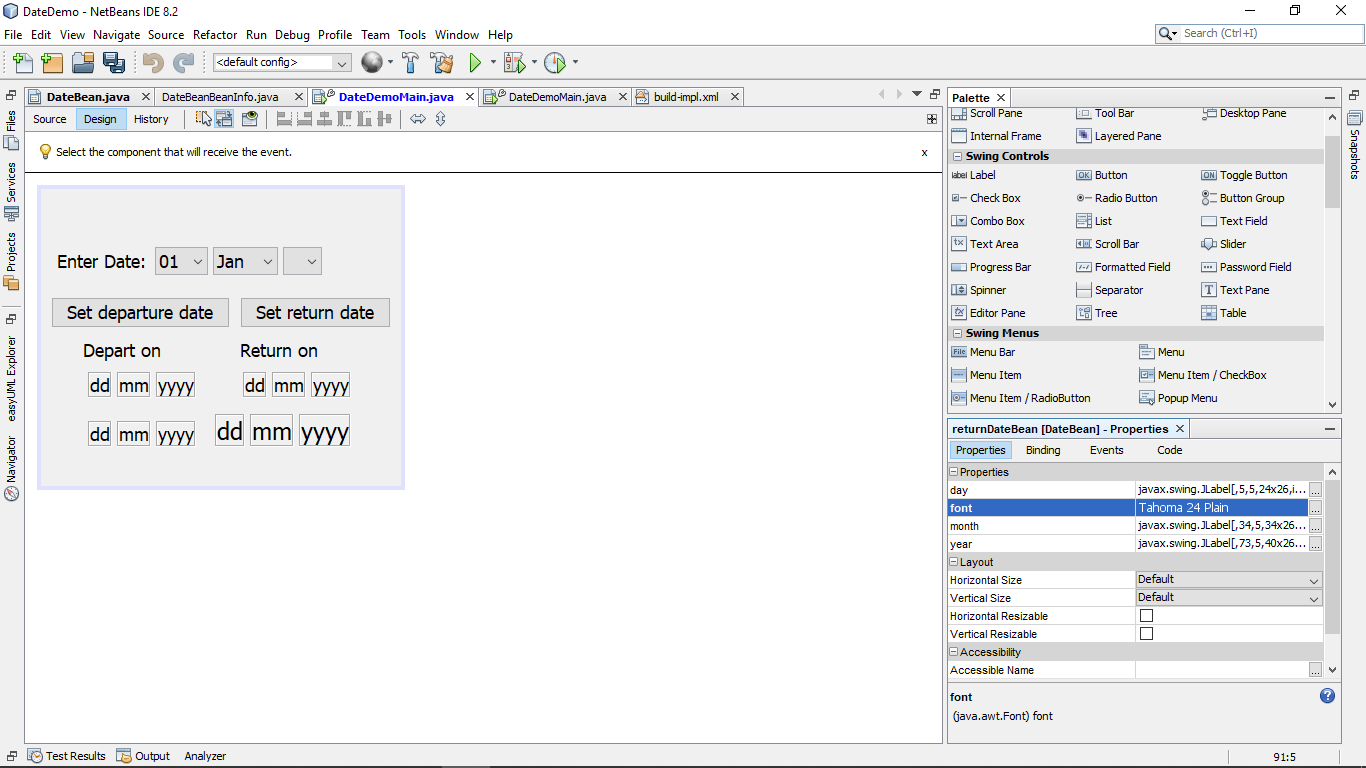


Figure 8Evidence of the code running 2Note how the output shows the old date and the new date for the returnDateBean due to the propertyChangeEvent



Day, month, year and font are reflective of the current state of

Figure Properties reflect the Beaninfo doc and the change in font affects all components for the bean’

2.2 Copy and paste **code that you wrote or amended**. Please **format** it nicely and **make it easy** for the tutor to see and read your code.

**//Manifest code**

Manifest-Version: 1.0

Name: gre/cwk/DateBean/DateBean.class

Java Bean: True

**//BeanInfo Code**

public class DateBeanBeanInfo extends SimpleBeanInfo {

// Get the properties to expose. Here we have decided to expose the "date","month" and year properties defined

// in the DateBean class itself and the "font" property inherited from JLabel

public PropertyDescriptor[] getPropertyDescriptors() {

try {

PropertyDescriptor d = new PropertyDescriptor("day", DateBean.class, "getDay", "setDay");

PropertyDescriptor m = new PropertyDescriptor("month", DateBean.class, "getMonth", "setMonth");

PropertyDescriptor y = new PropertyDescriptor("year", DateBean.class, "getYear", "setYear");

PropertyDescriptor f = new PropertyDescriptor("font", DateBean.class, "getFont", "setFont");

PropertyDescriptor[] pds = new PropertyDescriptor[]{

d, m, y, f};

return pds;

} catch (IntrospectionException ex) {

ex.printStackTrace();

return null;

}

}

// Get the image to use as an icon. Note that the image files need to be included

// in the bean's jar file. One way is to put them in the same folder as the .java files

// Another way is to create a folder (e.g. called icons) and in NetBeans

// right-click the project and choose Properties->Sources to add that folder

// to the Source Packages folders

public Image getIcon(int iconKind) {

switch (iconKind) {

case BeanInfo.ICON\_COLOR\_16x16:

return loadImage("datebeanicon.jpg");

case BeanInfo.ICON\_COLOR\_32x32:

return loadImage("datebeanicon\_big.jpg");

case BeanInfo.ICON\_MONO\_16x16:

return loadImage("datebeanicon\_greyscale.jpg");

case BeanInfo.ICON\_MONO\_32x32:

return loadImage("datebeanicon\_greyscale.jpg");

}

return null;

}

}

**//Modified DateBeanDemo Code**

private void btnSetDepartureDateActionPerformed(java.awt.event.ActionEvent evt) {

// Get the departure date input as ddmmmyyyy

String strDateInput = "" + cmbDay.getSelectedItem() + cmbMonth.getSelectedItem() + cmbYear.getSelectedItem();

// Parse the departure date input to create a LocalDate object

DateTimeFormatter dtf = DateTimeFormatter.ofPattern("ddMMMyyyy");

departureDate = LocalDate.parse(strDateInput, dtf);

if (returnDate != null && returnDate.isBefore(departureDate)) {

JOptionPane.showMessageDialog(this, "can't depart after returning", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Convert departure date to format for display

String strDD = String.format("%02d", departureDate.getDayOfMonth());

String strMM = String.format("%02d", departureDate.getMonthValue());

String strYYYY = "" + departureDate.getYear();

// Display the departure date - this code will change when the display field has

// been converted to a JavaBean component

lblDD.setText(strDD);

lblMM.setText(strMM);

lblYYYY.setText(strYYYY);

departureDateBean.setDay(strDD);

departureDateBean.setMonth(strMM);

departureDateBean.setYear(strYYYY);

}

/\*\*

\* Get the date entered in the set of combo boxes, parse it, check that it

\* is not before the departure date and display it.

\*

\* @param evt event object

\*/

private void btnSetReturnDateActionPerformed(java.awt.event.ActionEvent evt) {

// Get the return date input as ddmmmyyyy

String strDateInput = "" + cmbDay.getSelectedItem() + cmbMonth.getSelectedItem() + cmbYear.getSelectedItem();

// Parse the return date input to create a LocalDate object

DateTimeFormatter dtf = DateTimeFormatter.ofPattern("ddMMMyyyy");

returnDate = LocalDate.parse(strDateInput, dtf);

if (departureDate != null && departureDate.isAfter(returnDate)) {

JOptionPane.showMessageDialog(this, "can't return before departing", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Convert return date to format for display

String strDD = String.format("%02d", returnDate.getDayOfMonth());

String strMM = String.format("%02d", returnDate.getMonthValue());

String strYYYY = "" + returnDate.getYear();

// Display the return date - this code will change when the display field has

// been converted to a JavaBean component

lblDD1.setText(strDD);

lblMM1.setText(strMM);

lblYYYY1.setText(strYYYY);

//add an event listener in order to create the console output

departureDateBean.addPropertyChangeListener((PropertyChangeEvent evt1) -> {

System.out.println("old date is = " + evt1.getOldValue() + " new date is = " + evt1.getNewValue());

});

returnDateBean.setDay(strDD);

returnDateBean.setMonth(strMM);

returnDateBean.setYear(strYYYY);

}

**//DateBean Code**

public class DateBean extends javax.swing.JPanel {

//property change support in order to support the event listener in Date Demo

public PropertyChangeSupport pcs = new PropertyChangeSupport(this);

/\*\*

\* Creates new form DateBean

\*/

public DateBean() {

initComponents();

//creates a new propertychange listener in order to check if an event happens

//once the event happens, it returns both the new and old value for the item which triggered the event

}

//add listner to property changes in Date Demo so pcs causes the elements to properly react

@Override

public void addPropertyChangeListener(PropertyChangeListener listener) {

pcs.addPropertyChangeListener(listener);

}

//setter/getter for day label

public String getDay(){

//returns current value as a string

return lblDD.getText();

}

public void setDay(String dd){

//fire property change for day

//turns a string into the display for the current label

firePropertyChange("day", getDay(),dd );

lblDD.setText(dd);

}

public String getMonth(){

return lblMM.getText();

}

public void setMonth(String mm){

//fire property change for month

firePropertyChange("month", getMonth(),mm );

lblMM.setText(mm);

}

public String getYear(){

return lblYYYY.getText();

}

public void setYear(String yyyy){

//fire property change for year

firePropertyChange("year", getYear(),yyyy );

lblYYYY.setText(yyyy);

}

@Override

public void setFont(Font font){

//if statement to get around the nullpointer exception created when changing

//the font with no text in

if(lblDD != null && lblMM != null && lblYYYY != null ){

//sets all values to the currently inputted font

lblDD.setFont(font);

lblYYYY.setFont(font);

lblMM.setFont(font);

// this.setFont(font);

}

}

@Override

//returns super in order to get overall font setting

public Font getFont(){

return super.getFont();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

lblDD = new javax.swing.JLabel();

lblMM = new javax.swing.JLabel();

lblYYYY = new javax.swing.JLabel();

setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

setName("DateBeanContainer"); // NOI18N

lblDD.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

lblDD.setText("dd");

lblDD.setBorder(javax.swing.BorderFactory.createEtchedBorder());

add(lblDD);

lblMM.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

lblMM.setText("mm");

lblMM.setBorder(javax.swing.BorderFactory.createEtchedBorder());

add(lblMM);

lblYYYY.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

lblYYYY.setText("yyyy");

lblYYYY.setBorder(javax.swing.BorderFactory.createEtchedBorder());

add(lblYYYY);

}// </editor-fold>

// Variables declaration - do not modify

private javax.swing.JLabel lblDD;

private javax.swing.JLabel lblMM;

private javax.swing.JLabel lblYYYY;

// End of variables declaration

}