**STEPS TO BE DONE IN** **OptionsFrame**

In the **OptionsFrame** we need to do following steps:

1. Detecting which **RadioButton** user has clicked

2. Opening the respective Frame

3. Disposing the current Frame

**DETECTING WHICH RADIOBUTTON IS SELECTED**

1. The **JRadioButton**  object has a method called **isSelected( )** which returns **true** if the **JRadioButton** is selected otherwise it returns **false**.

2. So to detect whether **jrbAddEmp** is selected or not we would write:

***if(jrbAddEmp.isSelected( ))***

***{***

***}***

**OPENING ANOTHER FRAME FROM CURRENT FRAME**

1. Create an object of the frame to be opened. For ex if you want to open **AddEmployeeFrame** then:

***AddEmployeeFrame addEmpFrame=new AddEmployeeFrame();***

2. Call the method **setVisible( )** of the Frame so that it becomes visible:

***addEmpFrame.setVisible(true);***

3. Call the method **dispose()** of current frame so that it becomes invisible:

***this.dispose();***

**How To Handle Do Task Button**

The next task is to allow the user to select an options amongst **Add Emp, Search Emp, View All Emp** and **Quit**

To do this we need to handle the "**Do Task**" Button in the following way:

**a. We will detect which radio button has been selected and according to that we will open the next frame and dispose the current frame**

**b. If no radio button has been selected then we will generate an error message and return.**

**c. If user has selected to quit then we will terminate the app.**

**WRITING THE CODE FOR BUTTON Button.png IN OptionsFrame**

***private void btnDoTaskActionPerformed(java.awt.event.ActionEvent evt) {***

***if(jrbAddEmp.isSelected())***

***{***

***AddEmployeeFrame addEmpFrame=new AddEmployeeFrame();***

***addEmpFrame.setVisible(true);***

***this.dispose();***

***}***

***else if(jrbSearchEmp.isSelected())***

***{***

***SearchEmployeeFrame searchEmpFrame=new SearchEmployeeFrame();***

***searchEmpFrame.setVisible(true);***

***this.dispose();***

***}***

***else if(jrbShowAllEmp.isSelected())***

***{***

***ViewAllEmployeeFrame viewAllEmpFrame=new ViewAllEmployeeFrame();***

***viewAllEmpFrame.setVisible(true);***

***this.dispose();***

***}***

***else if(jrbQuit.isSelected())***

***{***

***int answer=JOptionPane.showConfirmDialog(null,"Are u sure ?","Quitting?",JOptionPane.YES\_NO\_OPTION,JOptionPane.QUESTION\_MESSAGE);***

***if(answer==JOptionPane.YES\_OPTION)***

***{***

***JOptionPane.showMessageDialog(null,"Thank you for using the app!");***

***System.exit(0);***

***}***

***}***

***else***

***{***

***JOptionPane.showMessageDialog(null,"Please make a selection first!");***

***}***

}

**WRITING THE CODE FOR BUTTON Button.png IN AddEmployeeFrame**

1. When the user will click the **Add Emp** button then it will add the record of new employee in the **Employee** table.

2. Following are it's important points:

**a. It will first validate whether all the data has been properly filled all or not.**

**b. If not , then it will generate an error message and return.**

**c. Otherwise , it will create an EmpPojo object , fill all the values in it and pass it to the method addEmp( ) of the EmpDAO**

**d. If the method addEmp( ) returned true then it will display the message "Record inserted successfully" , otherwise it will display an error message.**

**f. It will also handle NumberFormatException as well as any SQLException that will be thrown by the method addEmp( )**

3. Based upon the above facts following is it's code:

***private void btnAddEmpActionPerformed(java.awt.event.ActionEvent evt) {***

***String eno=txtEmpNo.getText().trim();***

***String ename=txtEmpName.getText().trim();***

***String sal=txtSal.getText().trim();***

***if(eno.isEmpty()||ename.isEmpty()||sal.isEmpty())***

***{***

***JOptionPane.showMessageDialog(null,"Please input all values!","Error!",JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***try***

***{***

***int empno=Integer.parseInt(eno);***

***double salary=Double.parseDouble(sal);***

***EmpPojo obj=new EmpPojo(empno,ename,salary);***

***boolean result=EmpDAO.addEmp(obj);***

***if(result==true)***

***{***

***JOptionPane.showMessageDialog(null,"Record inserted successfully!","Success!",JOptionPane.INFORMATION\_MESSAGE);***

***}***

***else***

***{***

***JOptionPane.showMessageDialog(null,"Record cannot be inserted!","Problem!",JOptionPane.WARNING\_MESSAGE);***

***}***

***}***

***catch(NumberFormatException ex)***

***{***

***JOptionPane.showMessageDialog(null,"Please input correct values!","Error!",JOptionPane.ERROR\_MESSAGE);***

***ex.printStackTrace();***

***}***

***catch(SQLException ex)***

***{***

***JOptionPane.showMessageDialog(null,"Some error occurred in DB!","Error!",JOptionPane.ERROR\_MESSAGE);***

***ex.printStackTrace();***

***}***

***}***

**WRITING THE CODE FOR BUTTON Button.png IN AddEmployeeFrame**

1. When the user will click the **Back** button then it will **dispose** the **current frame** and open the **OptionsFrame**

2. Based upon the above facts following is it's code:

***private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {***

***OptionsFrame opFrame=new OptionsFrame();***

***opFrame.setVisible(true);***

***this.dispose();***

***}***

**WRITING THE CODE FOR BUTTON Button.png IN SearchEmployeeFrame**

1. When the user will click the **Search** button then it will find and display the record of the employee from the **Employee** table

2. Following are it's important points:

**a. It will first validate whether employee number has been given or not.**

**b. If not , then it will generate an error message and return.**

**c. Otherwise , it will call the method findEmpById( ) method of EmpDAO passing it the Employee Number as argument**

**d. If the method findEmpById( ) returned EmpPojo object then it will display the details of the employee i.e. his/her name and salary , otherwise it will display the message "record not found".**

**e. It will also handle NumberFormatException as well as any SQLException that will be thrown by the method findEmpById( )**

3. Based upon the above facts following is it's code:

***private void btnSearchEmpActionPerformed(java.awt.event.ActionEvent evt) {***

***String eno=txtEmpNo.getText().trim();***

***if(eno.isEmpty())***

***{***

***JOptionPane.showMessageDialog(null,"Please input employe no!","Error!",JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***try***

***{***

***int empno=Integer.parseInt(eno);***

***EmpPojo obj=EmpDAO.findEmpById(empno);***

***if(obj==null)***

***{***

***JOptionPane.showMessageDialog(null,"Record not found!","Not Found!",JOptionPane.INFORMATION\_MESSAGE);***

***return;***

***}***

***txtEmpName.setText(obj.getEname());***

***txtSal.setText(String.valueOf(obj.getSal()));***

***}***

***catch(NumberFormatException ex)***

***{***

***JOptionPane.showMessageDialog(null,"Please input integer value as empno!","Error!",JOptionPane.ERROR\_MESSAGE);***

***ex.printStackTrace();***

***}***

***catch(SQLException ex)***

***{***

***JOptionPane.showMessageDialog(null,"Some error occurred in DB!","Error!",JOptionPane.ERROR\_MESSAGE);***

***ex.printStackTrace();***

***}***

***}***

**WRITING THE CODE FOR BUTTON Button.png IN ViewAllEmployeeFrame**

1. When the user will click the **Show All Records** button then it will pull and display the records of all the Employees from the **Employees** table

2. Following are it's important points:

**a. It will call the method getAllEmp ( ) of the EmployeeDAO class**

**b. It will then check whether the list returned by the method getAllEmp( ) is empty or not. If it is empty then it will display the message "No record found"**

**c. Otherwise , it will iterate over the list of Employee objects in the list and display each object's data in the Text Area**

**d. It will also handle SQLException that will be thrown by the method getAllEmp( )**

3. Based upon the above facts following is it's code:

***private void btnShowAllEmpActionPerformed(java.awt.event.ActionEvent evt) {***

***try***

***{***

***ArrayList<EmpPojo> empList=EmpDAO.getAllEmp();***

***txtAllEmp.setText("");***

***for(EmpPojo e:empList)***

***{***

***int eno=e.getEmpno();***

***String ename=e.getEname();***

***double sal=e.getSal();***

***String rec=eno+"\t"+ename+"\t"+sal+"\n";***

***txtAllEmp.append(rec);***

***}***

***}***

***catch(Exception ex)***

***{***

***ex.printStackTrace();***

***JOptionPane.showMessageDialog(null,"Exception:"+ex);***

***}   
}***

Based upon the above understanding , students have to implement the functionality of remaining 2 Frames i.e **UpdateEmployeeFrame** and **DeleteEmployeeFrame** themselves.