Lecture 02
Creating Graphical User Interface
Part 3

How to use Tables

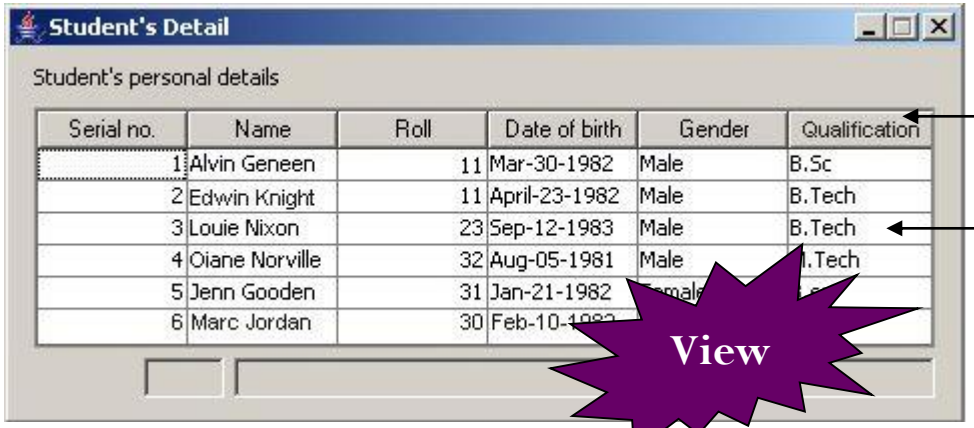
Objectives

- How to present data in a table?
- How to manage events on table?
- How to customize a table?

Contents

- The javax.swing.JTable class
- Customize tables using javax.swing.table.AbstractTableModel
- Replacing the cell editor of a table column

1- The JTable Class



Column Headers

ColumnModel

data

Data Model

```
class javax.swing.JTable
{ tableModel;
  columModel;
  selectionModel;
  cellEditor;
  editorComp;
  cellRenderer;
  tableHeader;
  int editingColumn, edittingRow;
  color ....
  ....
}
```

get/set

Constructors

- `public JTable()`
- `public JTable(int numRows, int numColumns)`
- `public JTable(Object[][] data, Object[] columns)`
- `public JTable(Vector data, Vector columns)`
- `public JTable(TableModel model)`

For more details:
docs-Java8/api/javax/swing/JTable.html

Common Event handling:
MouseEvent
MouseListener
  public void mouseClicked(MouseEvent e)

JTable...

Common Methods

set/get/remove – setModel(..)

addColumn(TableColumn aColumn)

addColumnSelectionInterval(int index0, int index1)

int **columnAtPoint**(Point point) // get column index

createxxxx // create Model, Renderer, Editor, SelectionModel, TableHeader

updateUI()

……

int getRowCount() **int getColumnCount()**

int getSelectedRow() **int getSelectedColumn()**

Object getValueAt(int i, int j) **void setValueAt(Object obj ,int row, int col)**

TableColum getTableModel().getColumn(int col)

JTable…: Related Interfaces and Classes

Interfaces and classes are used in the javax.swing.JTable class.

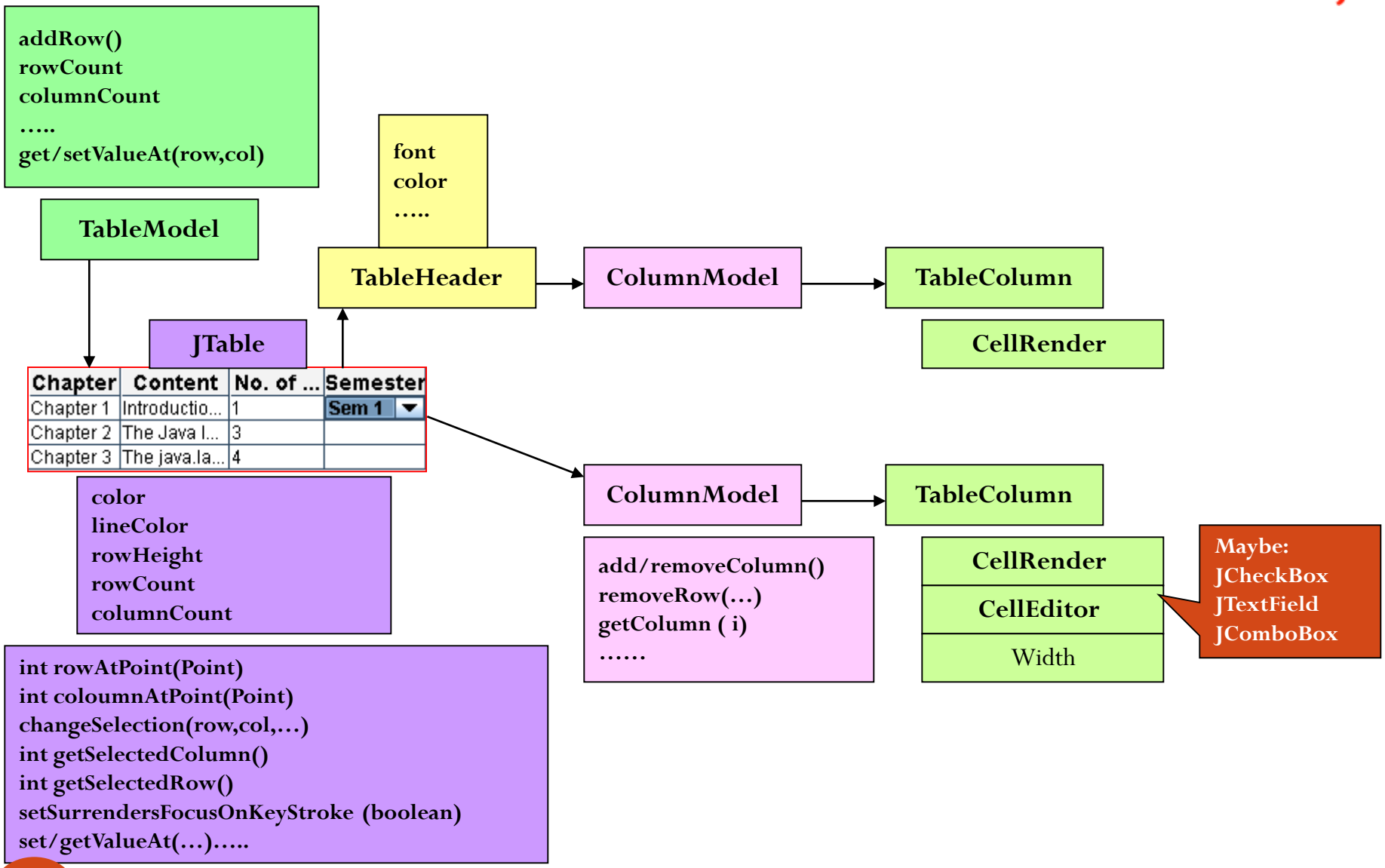
- java.lang.**Object**
  - javax.swing.table.**AbstractTableModel**
    - javax.swing.table.**DefaultTableModel**
  - javax.accessibility.**AccessibleContext**
    - java.awt.**Component.AccessibleAWTComponent**
      - java.awt.**Container.AccessibleAWTContainer**
        - javax.swing.**JComponent.AccessibleJComponent**
          - javax.swing.table.**JTableHeader.AccessibleJTableHeader**
    - javax.swing.table.**JTableHeader.AccessibleJTableHeader.AccessibleJTableHeaderEntry**
  - java.awt.**Component**
    - java.awt.**Container**
      - javax.swing.**JComponent**
        - javax.swing.**JLabel**
          - javax.swing.table.**DefaultTableCellRenderer**
            - javax.swing.table.**DefaultTableCellRenderer.UIResource**
        - javax.swing.table.**JTableHeader**
  - javax.swing.table.**DefaultTableColumnModel**
  - javax.swing.table.**TableColumn**

Interface Hierarchy

- javax.swing.**CellEditor**
  - javax.swing.table.**TableCellEditor**
- javax.swing.table.**TableCellRenderer**
- javax.swing.table.**TableColumnModel**
- javax.swing.table.**TableModel**

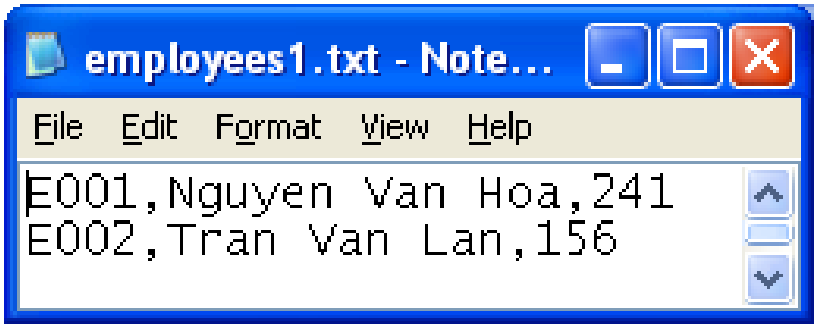
Each Defaultxxxxx class implements the correlative interface.

JTable: Architecture

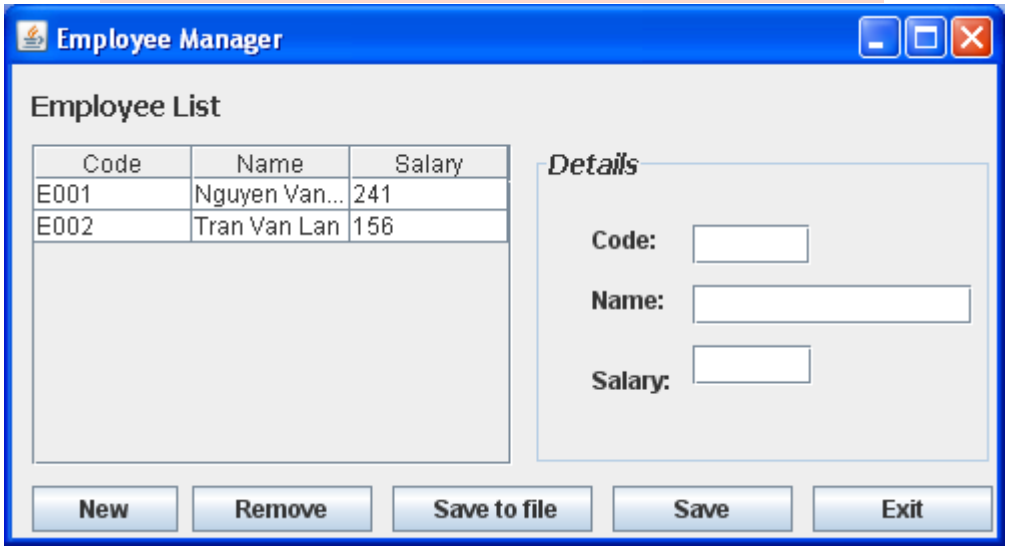


Demo 6: JTable

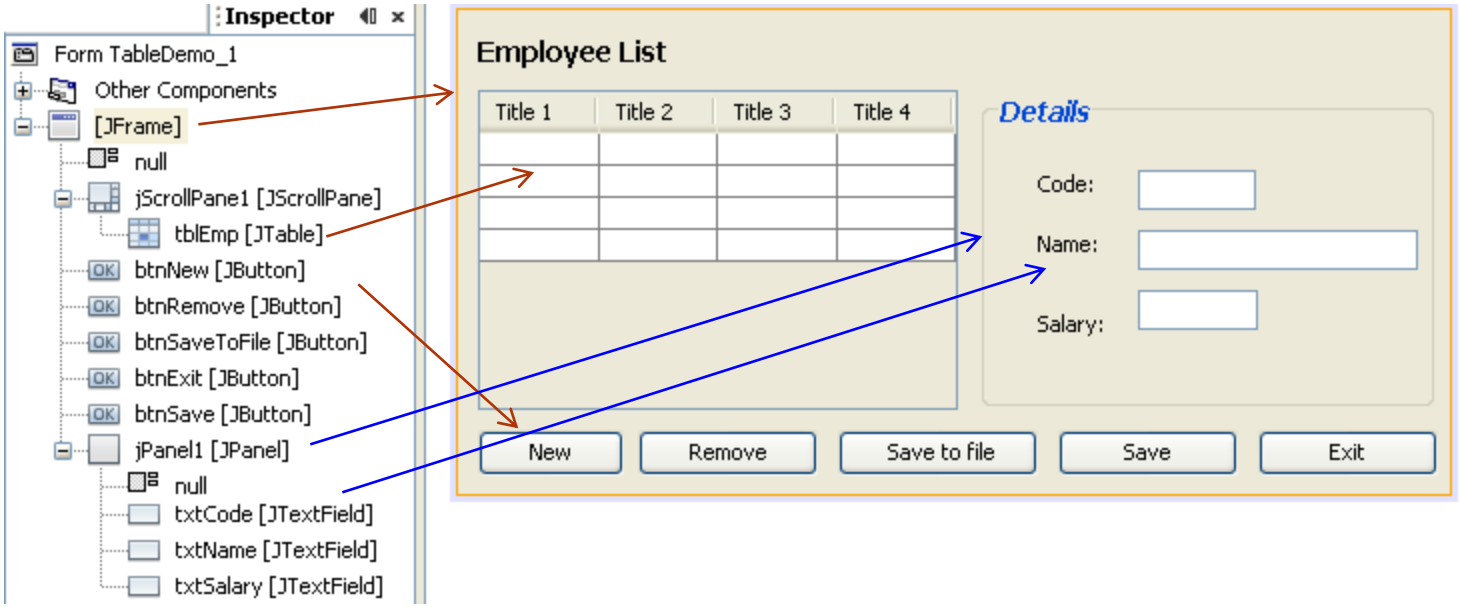
Initial employee details are stored in a file



Managing Program is expected:

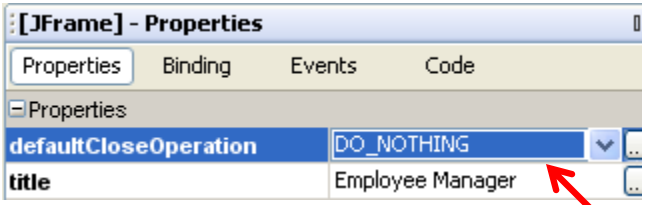


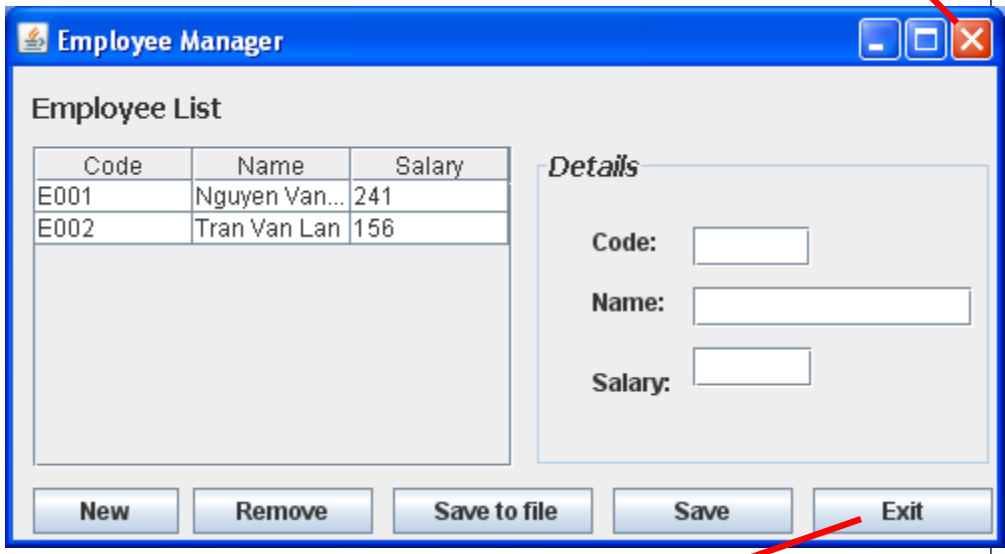
GUI Organization



Demo 6: JTable...

```
import java.util.Vector; // header and data of the table
import java.util.StringTokenizer; // for splitting string
import java.io.File;
import java.io.FileReader;     // reading data from text file
import java.io.BufferedReader;
import java.io.PrintWriter;     // write data to text file
import javax.swing.table.DefaultTableModel; // control table
import javax.swing.JOptionPane;  // for build-in dialog
public class TableDemo_1 extends javax.swing.JFrame {
    String filename="employees1.txt";
    Vector<String> header= new Vector<String>();
    Vector data= new Vector();
    boolean addNew=false;
    boolean changed=false; // Are data changed?
    /** Creates new form TableDemo_1 */
```





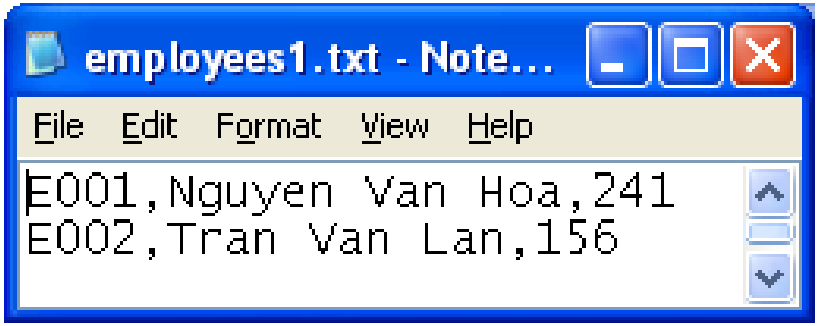

```
private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (changed==true)
    { if (JOptionPane.showConfirmDialog(this,"Data changed. Save Y/N?")==
            JOptionPane.OK_OPTION)
      btnSaveToFileActionPerformed(null);
    }
    System.exit(0);
}
```

Demo 6: JTable…

```
/** Creates new form TableDemo_1 */
public TableDemo_1() {
    initComponents();
    this.setSize(500,270);
    // setup header of the table
    header.add("Code"); header.add("Name");header.add("Salary");
    loadData();
    DefaultTableModel tblModel;
    tblModel= (DefaultTableModel)this.tblEmp.getModel();
    tblModel.setDataVector(data, header);
}
```

Employee List

Code	Name	Salary
E001	Nguyen Van...	241
E002	Tran Van Lan	156




```
private void loadData(){
    try
    {  File f= new File(filename);
       FileReader fr= new FileReader(f);
       BufferedReader bf = new BufferedReader(fr);
       String aDetails;
       while ((aDetails = bf.readLine())!=null){
           StringTokenizer stk= new StringTokenizer(aDetails, ",");
           String code=stk.nextToken();
           String name= stk.nextToken();
           String salaryStr= stk.nextToken();
           Vector<String> v= new Vector<String>();
           v.add(code); v.add(name); v.add(salaryStr);
           data.add(v);
       }
       bf.close(); fr.close();
    }
    catch (Exception e)
    {  JOptionPane.showMessageDialog(this, "The file " +
                filename + " not found." );
    }
}
```

Demo 6: JTable…

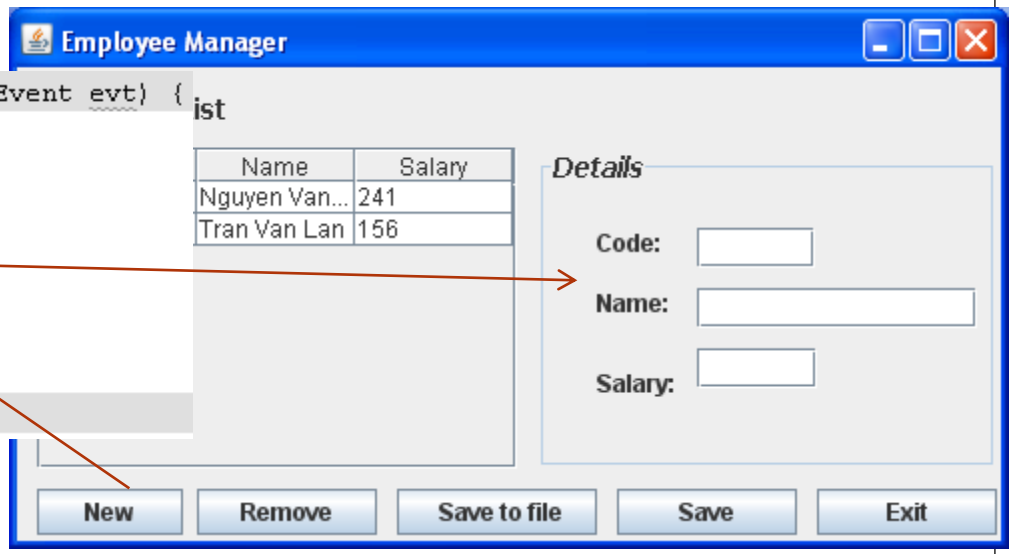
Cách 2

```
public JtableDemo() {
    initComponents();
    this.setSize(1000, 700);
    this.setLocation(300, 150);
    header.add("Code");
    header.add("Name");
    header.add("Salary");
    loaddata();
    DefaultTableModel tblmodel = new DefaultTableModel(data, header);
    tblemp.setModel(tblmodel);

}
```

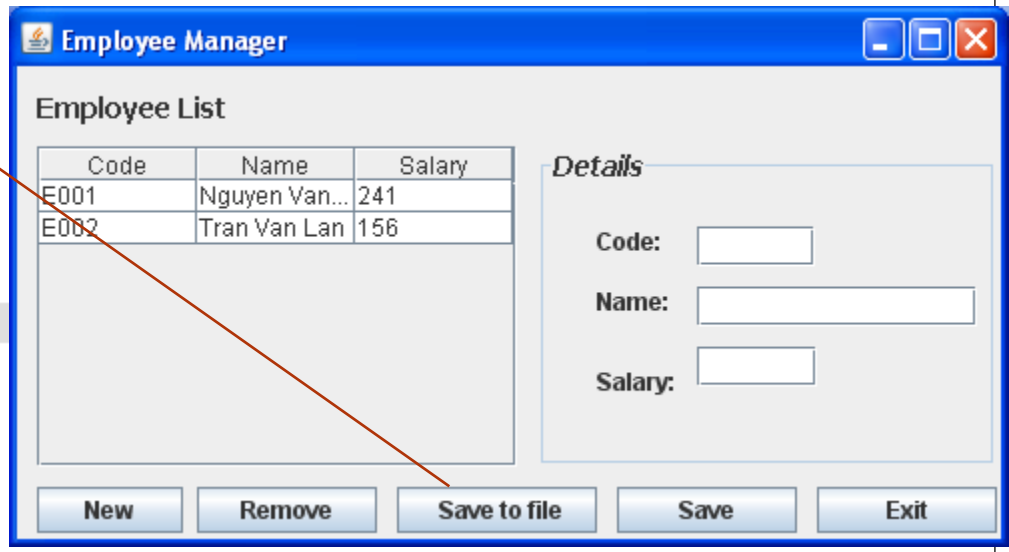
Demo 6: JTable...

```
private void btnNewActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.addNew =true;    // entering the add-new mode
    this.txtCode.setText(""); // preparing input data
    this.txtCode.setEditable(true);
    this.txtName.setText("");
    this.txtSalary.setText("");
    this.txtCode.requestFocus();
}
```



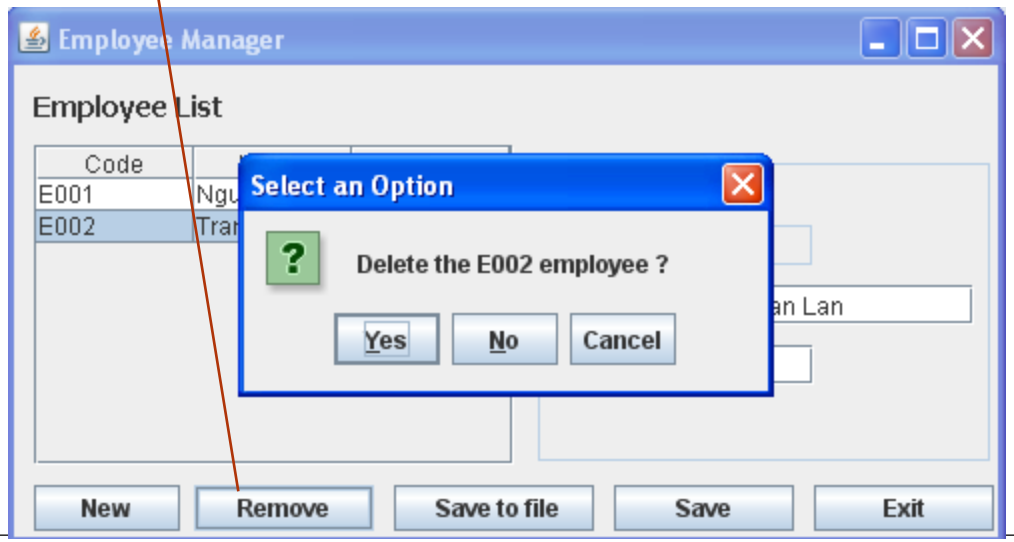
Demo 6: JTable...

```
private void btnSaveToFileActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        File f= new File(filename);
        PrintWriter pf= new PrintWriter(f);
        int n= this.tblEmp.getRowCount();
        for (int i=0;i<n; i++){
          Vector<String> v = (Vector<String>)(data.get(i));
          String S= v.get(0) + "," + v.get(1) + "," + v.get(2);
          pf.println(S);
        }
        pf.close();
        changed=false;
    }
    catch (Exception e){
        JOptionPane.showMessageDialog(this, e);
    }
}
```



Demo 6: JTable...

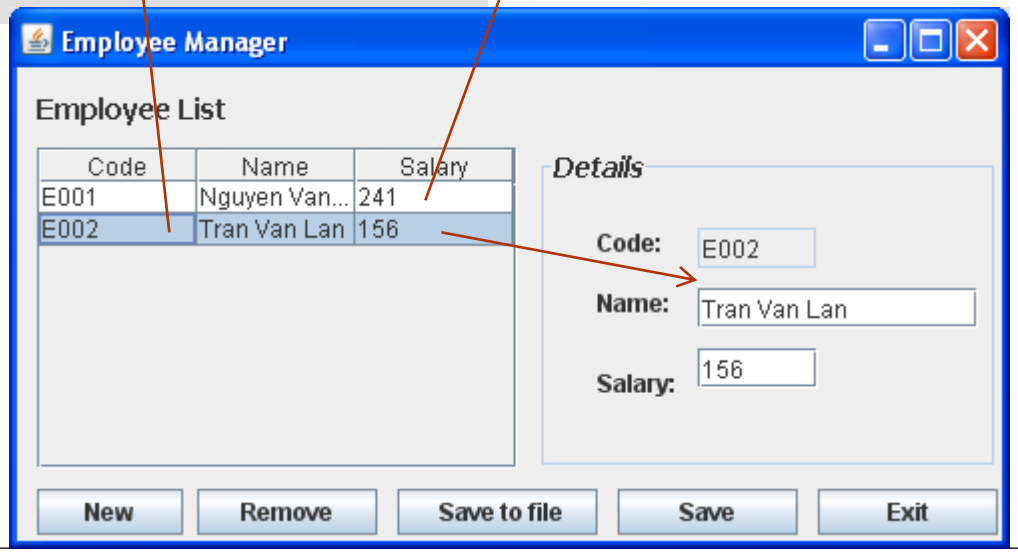
```
private void btnRemoveActionPerformed(java.awt.event.ActionEvent evt) {
    // Get position of seclected employee
    int pos = this.tblEmp.getSelectedRow();
    if (pos>=0)
    {  String code= (String)(tblEmp.getValueAt(pos,0));
       if (JOptionPane.showConfirmDialog(this,"Delete the " +
                code + " employee ?")== JOptionPane.OK_OPTION)
       { data.remove(pos);
         tblEmp.updateUI();
         changed=true;
       }
    }
}
```



Demo 6: JTable...

```
private void tblEmpMouseReleased(java.awt.event.MouseEvent evt) {
    // Turn off the on-table editting mode when user double-clicks on a cell
    if (this.tblEmp.isEditing())
    { int row= tblEmp.getSelectedRow();
      int column= tblEmp.getSelectedColumn();
      tblEmp.getCellEditor(row,column).cancelCellEditing();
    }
}
```

```
private void tblEmpMouseClicked(java.awt.event.MouseEvent evt) {
    // View current employee
    int row= this.tblEmp.getSelectedRow();
    this.txtCode.setText((String)(tblEmp.getValueAt(row,0)));
    this.txtCode.setEditable(false);// user can not update the code
    this.txtName.setText((String)(tblEmp.getValueAt(row,1)));
    this.txtSalary.setText((String)(tblEmp.getValueAt(row,2)));
    addNew = false; // view and update mode
}
```

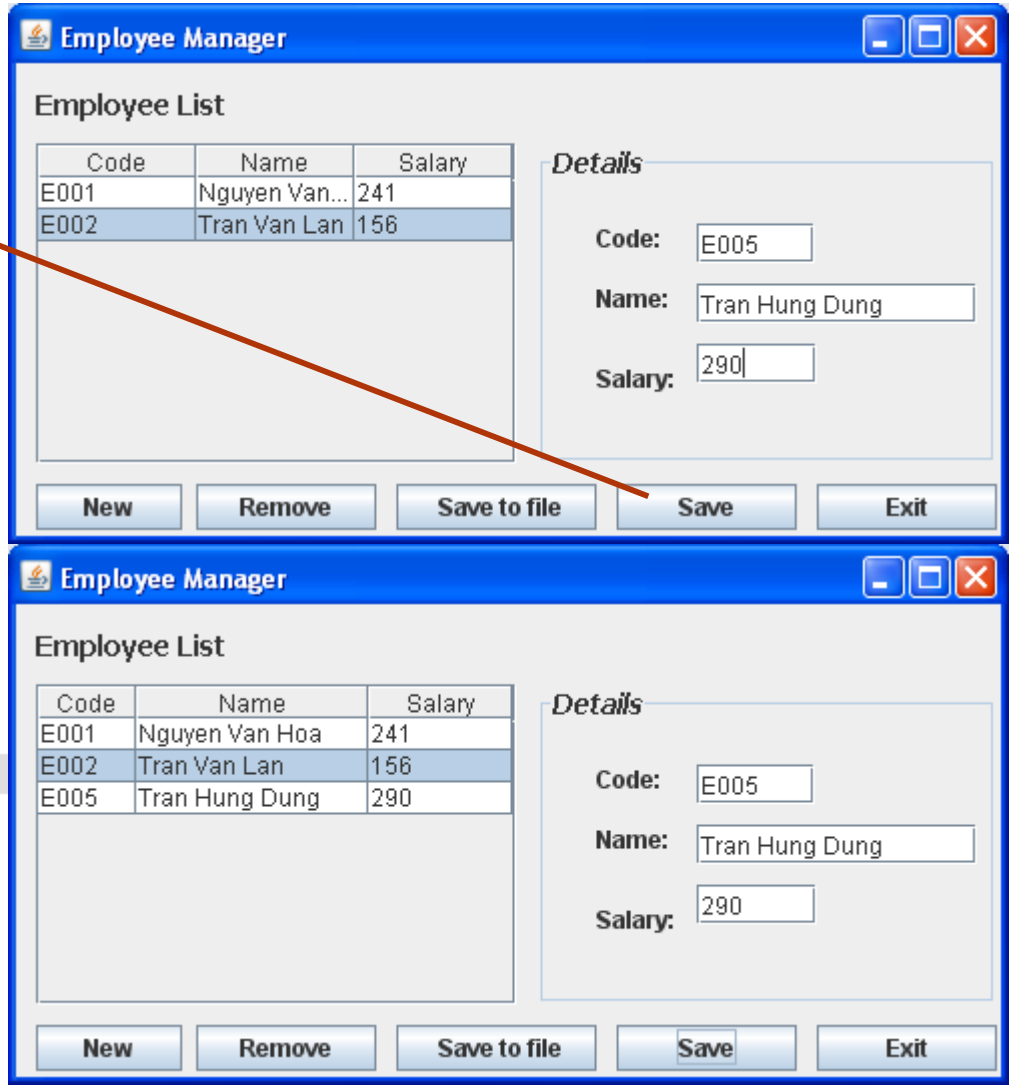


Demo 6: JTable...


```
boolean valid()
{ String s="";
  if (addNew==true) // checking the code
  {  s = this.txtCode.getText().trim().toUpperCase();
     this.txtCode.setText(s);
     if (!s.matches("^E\\d{3}$")) // checking code format
     {  JOptionPane.showMessageDialog(this, "Code format: EOOO");
        return false;
     }
     for (int i=0;i<data.size();i++) // checking duplicated
     { Vector v= (Vector)(data.get(i));
       if (s.equals((String)(v.get(0))))
        {  JOptionPane.showMessageDialog(this, "Code duplicated.");
           txtCode.requestFocus();
           return false;
        }
     }
  }
                          // checking the name
                          s = this.txtName.getText().trim();
                          if (s.length()==0)
                             {  JOptionPane.showMessageDialog(this, "Name is required.");
                                return false;
                             }
                          // checking the salary
                          s = this.txtSalary.getText().trim();
                          if (!s.matches("^\\d+$"))
                             {  JOptionPane.showMessageDialog(this, "Salary is an integer.");
                                return false;
                             }
                          return true;
                        }
```

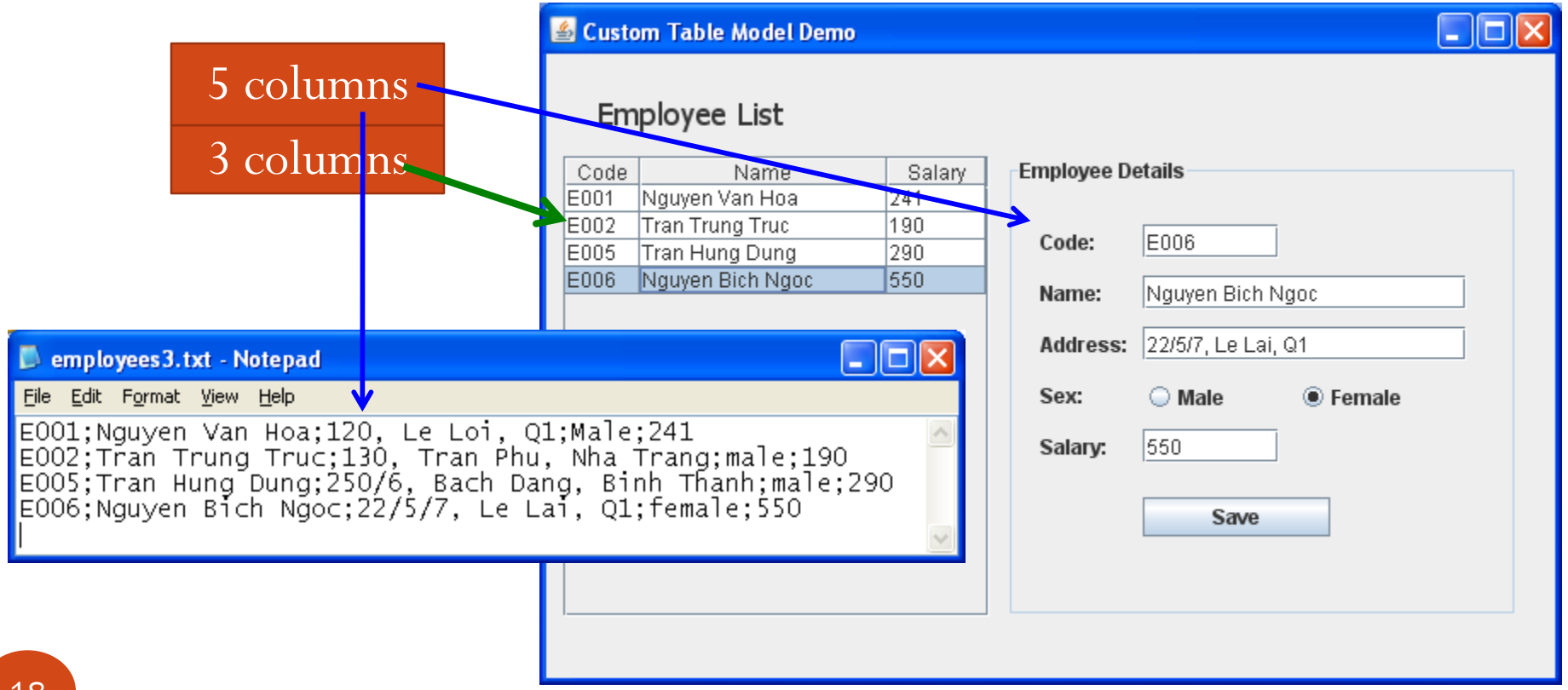
Demo 6: JTable...

```
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!valid()) return;
    String code = txtCode.getText();
    String name = txtName.getText();
    String salaryStr = txtSalary.getText();
    if (addNew)
    { Vector v= new Vector();
      v.add(code); v.add(name); v.add(salaryStr);
      data.add(v);
      addNew=false;
    }
    else
    { int pos= tblEmp.getSelectedRow();
      Vector v= (Vector)data.get(pos);
      v.set(1, name);
      v.set(2, salaryStr);
    }
    tblEmp.updateUI();
    changed=true;
}
```



2- Custom TableModel

- Some times, we want to present some main data columns only.
- The class javax.swing.table.AbstractTableModel will help us.



Demo 7: Custom TableModel

employees3.txt - Notepad

File Edit Format View Help

```
E001;Nguyen Van Hoa;120, Le Loi, Q1;Male;241
E002;Tran Trung Truc;130, Tran Phu, Nha Trang;male;190
E005;Tran Hung Dung;250/6, Bach Dang, Binh Thanh;male;290
E006;Nguyen Bich Ngoc;22/5/7, Le Lai, Q1;female;550
```

5 columns

3 columns

Projects

- Chapter11
  - Source Packages
    - <default package>
    - custom_table_model
      - Employee.java
      - EmployeeManager2.java
      - EmployeeTableModel.java

Custom Table Model Demo

Employee List

Code	Name	Salary
E001	Nguyen Van Hoa	241
E002	Tran Trung Truc	190
E005	Tran Hung Dung	290
E006	Nguyen Bich Ngoc	550

Employee Details

Code: E006

Name: Nguyen Bich Ngoc

Address: 22/5/7, Le Lai, Q1

Sex: Male Female

Salary: 550

Save

Demo 7: Custom TableModel…

Class for an employee

```
package custom_table_model;
public class Employee {
    String code, name, address;
    boolean sex; // true: male, false: female
    int salary;

     public Employee(String code, String name, String address,
                                boolean sex, int salary) {...}
     public String getAddress() {...}
     public void setAddress(String address) {...}
     public String getCode() {...}
     public void setCode(String code) {...}
     public String getName() {...}
     public void setName(String name) {
         this.name = name;
     }
     public int getSalary() {...}
     public void setSalary(int salary) {...}
     public boolean isSex() {
         return sex;
     }
     public void setSex(boolean sex) {
         this.sex = sex;
     }
}
```

Demo 7: Custom TableModel...

Class for a list employee. This list will be a table model of a table

```
/* Custom table model DEmo. */
package custom_table_model;
import javax.swing.table.AbstractTableModel;
import java.util.Vector;
import javax.swing.JOptionPane;
public class EmployeeTableModel <E> extends AbstractTableModel {
    String[] header;
    int[] indexes;
    Vector<Employee> data;

    public EmployeeTableModel(String[] header, int[] indexes) {
        int i=0;
        this.header= new String[header.length];
        for (i=0;i<header.length;i++)this.header[i]=header[i];
        this.indexes= new int[indexes.length];
        for (i=0;i<header.length;i++)this.indexes[i]=indexes[i];
        this.data= new Vector<Employee>();
    }
    public Vector<Employee> getData(){
       return  data;
    }
```

```
 // Overriding from the AbstractTableModel class
    public int getRowCount() {
       return data.size();
    }
    public int getColumnCount(){
        return header.length;
    }
    public String getColumnName(int column){
        return (column>=0 && column<header.length)?
                header[column]:"";
    }
    public Object getValueAt(int row, int column){
      if (row<0 || row>=data.size() ||
          column<0 || column>=header.length)
          return null;
      Employee emp= data.get(row);
      switch (indexes[column]){
          case 0: return emp.getCode();
          case 1: return emp.getName();
          case 2: return emp.getAddress();
          case 3: return emp.isSex();
          case 4: return emp.getSalary();
      }
      return null;
    }
}
```

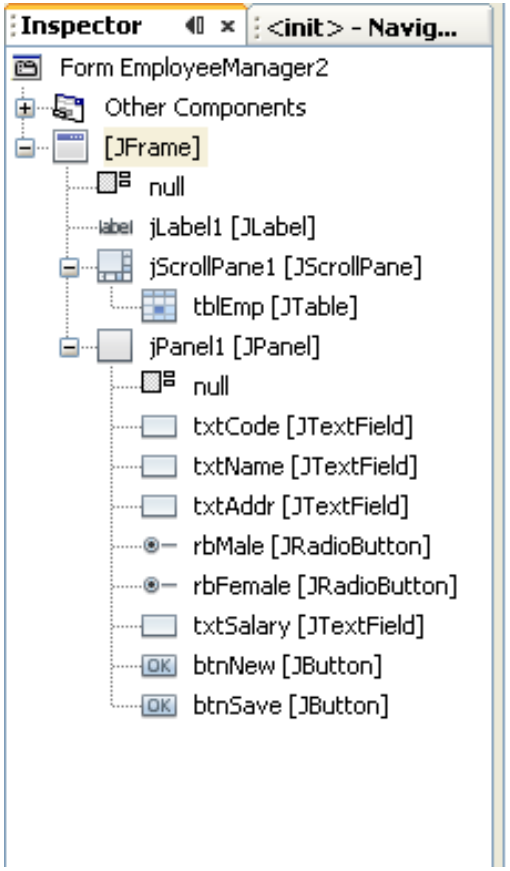
These methods indicate that how to get data from the model and present them on the table

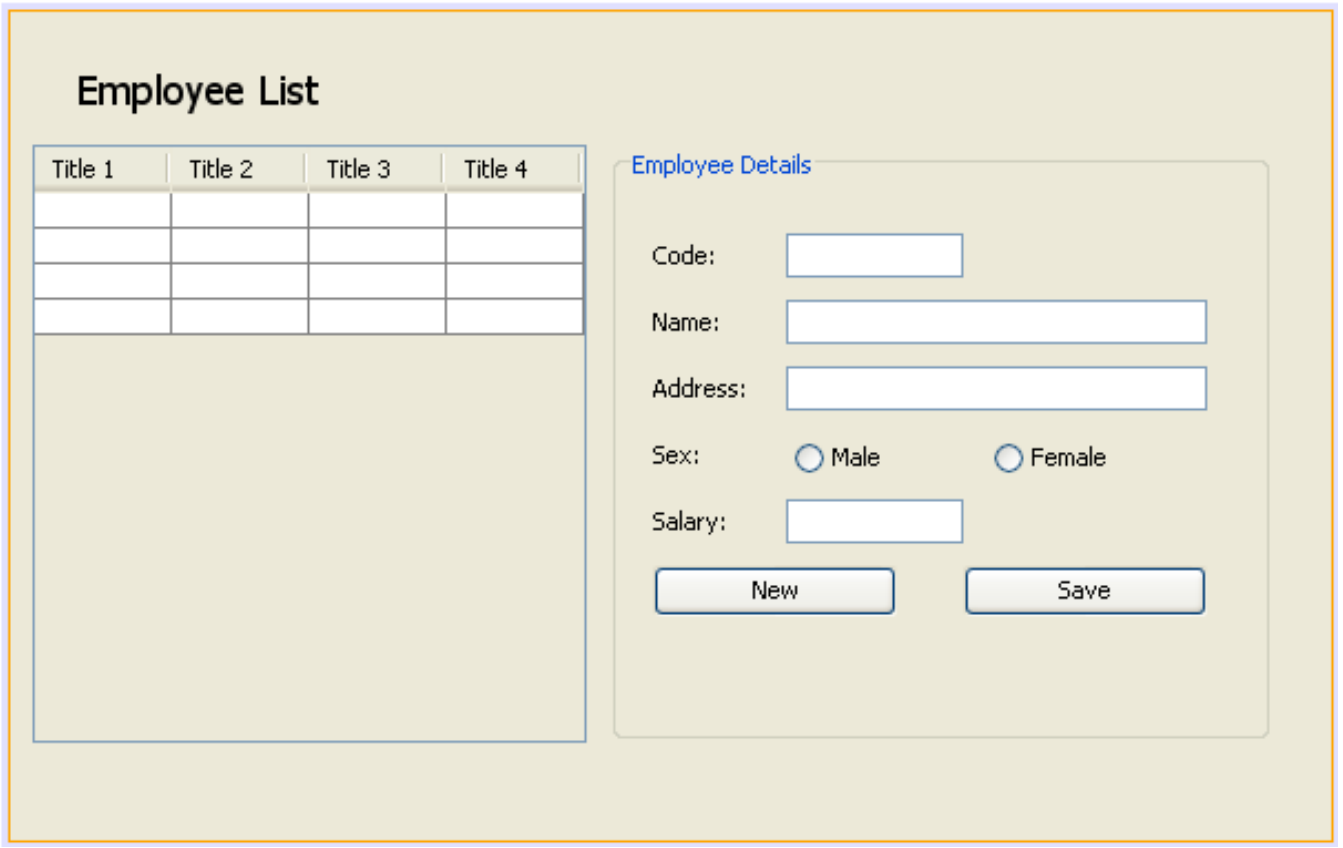
indexes

0	1	2	3	4
code	name	address	sex	salary
0	1	4		

Demo 7: Custom TableModel...

GUI Design





Demo 7: Custom TableModel…

```
/* Using the custom table model */
package custom_table_model;
import java.io.*;
import javax.swing.JOptionPane;
import java.util.Vector;
import java.util.StringTokenizer;

public class EmployeeManager2 extends javax.swing.JFrame {
    String filename="employees3.txt";
    EmployeeTableModel<Employee> model; // list of emp.
    boolean addNew=false; // add new emp. or update?

    /** Creates new form EmployeeManager2 */
    public EmployeeManager2() {
        initComponents();
        this.setSize(600,400);
        // Set up model with 3 column
        int[] indexes={0, 1, 4 };
        String[] header={"Code", "Name", "Salary"};
        model= new EmployeeTableModel<Employee>(header,indexes);
        // Set the model to the table
        this.tblEmp.setModel(model);
        loadData();
    }
```

Demo 7: Custom TableModel…

```
// Load initial data from the file
private void loadData(){
    try{
        FileReader f= new FileReader(filename);
        BufferedReader bf= new BufferedReader(f);
        String S;
        while ((S=bf.readLine())!=null){
            S=S.trim();
            if (S.length()>0){
              StringTokenizer stk= new StringTokenizer(S,";");
              String code=stk.nextToken();
              String name=stk.nextToken();
              String addr=stk.nextToken();
              String sexStr=stk.nextToken();
              boolean sex= (sexStr.equalsIgnoreCase("MALE")==true);
              int salary =Integer.parseInt(stk.nextToken());
              model.getData().add(new Employee(code,name,addr,sex,salary));
            }
        }
        bf.close();f.close();
    }
    catch (Exception e){
       JOptionPane.showMessageDialog(this,e);
    }
}
```

Demo 7: Custom TableModel…

```
private void tblEmpMouseClicked(java.awt.event.MouseEvent evt) {
    // Show the current employee details
    int pos = tblEmp.getSelectedRow();
    Employee curEmp = model.getData().get(pos);
    this.txtCode.setText(curEmp.getCode());
    this.txtName.setText(curEmp.getName());
    this.txtAddr.setText(curEmp.getAddress());
    this.rbMale.setSelected(curEmp.isSex());
    this.rbFemale.setSelected(!curEmp.isSex());
    this.txtSalary.setText(curEmp.getSalary()+"");
    this.txtCode.setEditable(false);
    this.addNew=false;
}

private void btnNewActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    addNew=true;
    this.txtCode.setText("");
    this.txtName.setText("");
    this.txtAddr.setText("");
    this.rbMale.setSelected(true);
    this.rbFemale.setSelected(false);
    this.txtSalary.setText("");
    this.txtCode.setEditable(true);
    this.txtCode.requestFocus();
}
private boolean validData() {
    // Your code here for validation
    return true;
}
```

Demo 7: Custom TableModel…

```
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!validData()) return;
    String code= txtCode.getText();
    String name= txtName.getText();
    String addr= txtAddr.getText();
    boolean sex= this.rbMale.isSelected();
    int salary = Integer.parseInt(txtSalary.getText());
    Employee emp= new Employee(code,name,addr,sex,salary);
    if (addNew) model.getData().add(emp);
    else // replace an employee details
    {   int pos= tblEmp.getSelectedRow();
        model.getData().set(pos, emp);
    }
    tblEmp.updateUI();
}
```

addNew=false;

Do yourself

- Add extra buttons to the GUI as below. Refer to the previous demontrations and complete this program

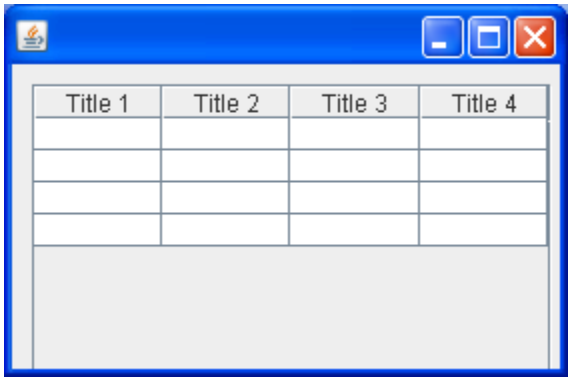
Algorithm for Saving the list to file:

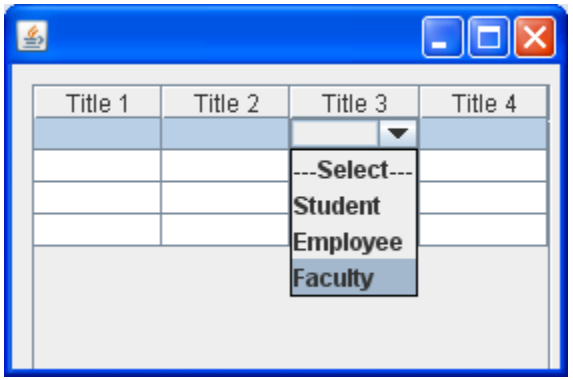
Open file → PrintWriter pw

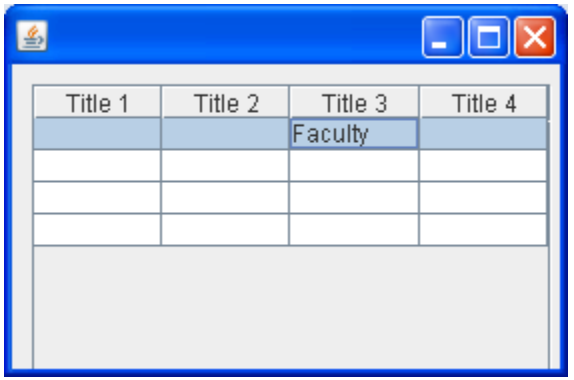
```
int  n= model.getData().size();
for (int i=0; i<n; i++)
{  Employee emp =  model.getData().get(i);
   String S=  emp.getCode() + ";" +  emp.getName() + ";" +. . .
   pw.println(S);
}
```

Close file

Demo 8: Replace Cell Editor








```
import javax.swing.JComboBox;
import javax.swing.DefaultCellEditor;
public class Demo_2 extends javax.swing.JFrame {
    /** Creates new form Demo_2 */
    public Demo_2() {
        initComponents();
        // Demo về việc thay editor cột thứ 2 bằng combobox
        JComboBox cb= new JComboBox();
        cb.addItem("---Select---");
        cb.addItem("Student");
        cb.addItem("Employee");
        cb.addItem("Faculty");
        DefaultCellEditor cbEditor= new DefaultCellEditor(cb);
        this.jTable1.getColumnModel().getColumn(2).setCellEditor(cbEditor);

    }
```

Summary

- The javax.swing.JTable class
- Customize tables using javax.swing.table.AbstractTabelModel
- Replacing the cell editor of a table column

Thank You