CSC 285 Session 5 Java Packages

// HW was to add the phone number from the database!!!!!

// Employee.java

//package finalproject;

public class Employee {

private int id;

private String name;

private String address;

private double hours;

private double rate;

private char sex;

private int age;

private boolean active;

private String ssn;

private String phone;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public double getHours() {

return hours;

}

public void setHours(double hours) {

this.hours = hours;

}

public double getRate() {

return rate;

}

public void setRate(double rate) {

this.rate = rate;

}

public char getSex() {

return sex;

}

public void setSex(char sex) {

this.sex = sex;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public boolean isActive() {

return active;

}

public void setActive(boolean active) {

this.active = active;

}

public String getSSN() {

return ssn;

}

public void setSSN(String ssn) {

this.ssn = ssn;

}

public String getPhone() {

return phone;

}

public void setPhone(String phone) {

this.phone = phone;

}

double calculateGrossPay() {

return (hours \* rate);

}

double calculateFederalTax() {

double yearlyIncome = calculateGrossPay() \* 52;

double taxRate;

if (yearlyIncome < 30000.00)

taxRate = .28;

else if (yearlyIncome < 50000.00)

taxRate = .32;

else

taxRate = .38;

return (calculateGrossPay() \* taxRate);

}

double calculateStateTax() {

return (calculateGrossPay() \* .0561);

}

double calculateNetPay() {

return (calculateGrossPay() - calculateFederalTax() - calculateStateTax());

}

//Note toString returns name only because it is used by the JList widget to populate

//the users in the pick list.

public String toString() {

return name;

}

}

// EmployeeDAO.java

//package finalproject;

import java.sql.\*;

import java.util.List;

import java.util.ArrayList;

import java.util.logging.Logger;

public class EmployeeDAO {

private static Logger log = Logger.getLogger(EmployeeDAO.class.getName());

public EmployeeDAO() throws Exception {

// Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

//String url = "jdbc:odbc:Driver={Microsoft Access Driver (\*.mdb)};DBQ=C:\\EmployeeManager\\employees.mdb;";

String url = "jdbc:odbc:Driver={Microsoft Access Driver (\*.mdb)};DBQ=C:\\EmployeeManager\\employees.mdb;";

// String url = "jdbc:odbc:Driver={Microsoft Access Driver (\*.mdb)};DBQ=C:\\test1\\employee.mdb;";

String username = "anonymous";

String password = "guest";

// Load the driver to allow connection to the database

try {

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection connection = DriverManager.getConnection(url, username, password);

} catch (ClassNotFoundException cnfex) {

System.err.println("Failed to load JDBC/ODBC driver.");

cnfex.printStackTrace();

System.exit(1); // terminate program

} catch (SQLException sqlex) {

System.err.println("Unable to connect");

sqlex.printStackTrace();

}

}

public List<Employee> getAllEmployees() throws Exception {

log.fine("getAllEmployees called");

Connection connection = null;

Statement statement = null;

ResultSet resultSet = null;

try {

connection = getConnection();

statement = connection.createStatement();

resultSet = statement.executeQuery("SELECT \* FROM EMPLOYEES ORDER BY name");

return resultSetToEmployees(resultSet);

} finally {

close(resultSet, statement, connection);

}

}

public void addEmployee(Employee employee) throws Exception {

log.fine("addEmployee called");

Connection connection = null;

PreparedStatement statement = null;

try {

connection = getConnection();

statement = connection.prepareStatement("INSERT INTO EMPLOYEES (name, address, hours, rate, sex, age, active, ssn,phone) VALUES (?,?,?,?,?,?,?,?,?)");

int i = 1;

statement.setString(i++, employee.getName());

statement.setString(i++, employee.getAddress());

statement.setDouble(i++, employee.getHours());

statement.setDouble(i++, employee.getRate());

statement.setString(i++, String.valueOf(employee.getSex()));

statement.setInt(i++, employee.getAge());

statement.setBoolean(i++, employee.isActive());

statement.setString(i++, employee.getSSN());

statement.setString(i++, employee.getPhone());

statement.executeUpdate();

} finally {

close(null, statement, connection);

}

}

public void updateEmployee(Employee employee) throws Exception {

log.fine("updateEmployee called");

Connection connection = null;

PreparedStatement statement = null;

try {

connection = getConnection();

statement = connection.prepareStatement("UPDATE EMPLOYEES SET name = ?, address = ?, hours = ?, rate = ?, sex = ?, age = ?, active = ?, ssn = ?, phone = ? WHERE id = ?");

int i = 1;

statement.setString(i++, employee.getName());

statement.setString(i++, employee.getAddress());

statement.setDouble(i++, employee.getHours());

statement.setDouble(i++, employee.getRate());

statement.setString(i++, String.valueOf(employee.getSex()));

statement.setInt(i++, employee.getAge());

statement.setBoolean(i++, employee.isActive());

statement.setString(i++, employee.getSSN());

statement.setString(i++, employee.getPhone());

statement.setInt(i++, employee.getId());

statement.executeUpdate();

} finally {

close(null, statement, connection);

}

}

public void deleteEmployee(Employee employee) throws Exception {

log.fine("deleteEmployee called");

Connection connection = null;

PreparedStatement statement = null;

try {

connection = getConnection();

statement = connection.prepareStatement("DELETE FROM EMPLOYEES WHERE id = ?");

statement.setInt(1, employee.getId());

statement.executeUpdate();

} finally {

close(null, statement, connection);

}

}

private List<Employee> resultSetToEmployees(ResultSet resultSet) throws Exception {

log.fine("resultSetToEmployees called");

ArrayList<Employee> employees = new ArrayList<Employee>();

while (resultSet.next()) {

Employee employee = new Employee();

employee.setId(resultSet.getInt("id"));

employee.setName(resultSet.getString("name"));

employee.setAddress(resultSet.getString("address"));

employee.setHours(resultSet.getDouble("hours"));

employee.setRate(resultSet.getDouble("rate"));

String sex = resultSet.getString("sex");

if (sex != null && sex.length() > 0)

employee.setSex(sex.charAt(0));

employee.setAge(resultSet.getInt("age"));

employee.setActive(resultSet.getBoolean("active"));

employee.setSSN(resultSet.getString("ssn"));

employee.setPhone(resultSet.getString("phone"));

employees.add(employee);

}//while

return employees;

}

private void close(ResultSet resultSet, Statement statement, Connection connection) {

if (resultSet != null) {

try {

resultSet.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if (statement != null) {

try {

statement.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

if (connection != null) {

try {

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

private Connection getConnection() throws Exception {

log.fine("getConnection called");

//c:\windows\syswow64\odbcad32

//ideally these connection would come from a connection pool...

// return DriverManager.getConnection("jdbc:odbc:employees");

//return DriverManager.getConnection("jdbc:odbc:MS ACCESS DataBase"+";DBQ=c:\\EmployeeManager\\employees.mdb");

return DriverManager.getConnection("jdbc:odbc:Driver={Microsoft Access Driver (\*.mdb)};DBQ=C:\\EmployeeManager\\employees.mdb");

}

}

// EmployeeManager.java

//package finalproject;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Logger;

import java.util.logging.Level;

import java.util.logging.Handler;

import java.text.NumberFormat;

import java.text.DecimalFormat;

public class EmployeeManager extends JFrame {

private static Logger log = Logger.getLogger(EmployeeManager.class.getName());

// private static Logger log = Logger.getLogger("InfoLogging");

private EmployeeDAO employeeDAO;

private JTextField idField;

private JTextField nameField;

private JTextField addressField;

private JTextField hoursField;

private JTextField rateField;

private JTextField sexField;

private JTextField ageField;

private JTextField activeField;

private JTextField ssnField;

private JTextField phoneField;

private JTextField grossPayField;

private JTextField stateTaxField;

private JTextField federalTaxField;

private JTextField netPayField;

private JList employeeList;

private DefaultListModel employeeListModel;

private JButton saveButton;

private ArrayList<JTextField> editableTextFields;

private ArrayList<JTextField> allTextFields;

public EmployeeManager() {

super("Employee Manager");

System.out.println(EmployeeManager.class.getName());

try {

employeeDAO = new EmployeeDAO();

} catch (Exception e) {

handleFatalException(e);

}

editableTextFields = new ArrayList<JTextField>();

allTextFields = new ArrayList<JTextField>();

//center panel

JPanel centerPanel = new JPanel();

centerPanel.setLayout(new GridLayout(13, 2));

setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

setSize(400, 400);

//the ID field does not get shown, but it gets used by this class

idField = new JTextField("-1"); // YOU DO NOT DISPLAY THE idField

nameField = addLabelAndTextField("Name", 100, true, centerPanel);

addressField = addLabelAndTextField("Address", 100, true, centerPanel);

hoursField = addLabelAndTextField("Hours", 10, true, centerPanel);

rateField = addLabelAndTextField("Rate", 10, true, centerPanel);

sexField = addLabelAndTextField("Sex", 10, true, centerPanel);

ageField = addLabelAndTextField("Age", 10, true, centerPanel);

activeField = addLabelAndTextField("Active", 10, true, centerPanel);

ssnField = addLabelAndTextField("SSN", 20, true, centerPanel);

phoneField = addLabelAndTextField("Phone Number", 100, true, centerPanel);

grossPayField = addLabelAndTextField("Gross Pay", 100, false, centerPanel);

stateTaxField = addLabelAndTextField("State Tax", 100, false, centerPanel);

federalTaxField = addLabelAndTextField("Federal Tax", 100, false, centerPanel);

netPayField = addLabelAndTextField("Net Pay", 100, false, centerPanel);

add(centerPanel, BorderLayout.CENTER);

//east panel

JPanel eastPanel = new JPanel(new GridLayout(2, 1));

employeeListModel = new DefaultListModel();

employeeList = new JList(employeeListModel);

employeeList.setLayoutOrientation(JList.VERTICAL\_WRAP);

employeeList.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

try {

refreshEmployeeList();

} catch (Exception e) {

handleFatalException(e);

}

eastPanel.add(employeeList);

JPanel eastPanelBottom = new JPanel(new GridBagLayout());

GridBagConstraints constraints = new GridBagConstraints();

constraints.gridy = 0;

JButton viewButton = new JButton("View");

viewButton.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent event) {

viewEmployee();

}

}

);

eastPanelBottom.add(viewButton, constraints);

constraints.gridy = 1;

JButton editButton = new JButton("Edit");

editButton.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent event) {

editEmployee();

}

}

);

eastPanelBottom.add(editButton, constraints);

constraints.gridy = 2;

JButton deleteButton = new JButton("Delete");

deleteButton.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent event) {

deleteEmployee();

}

}

);

eastPanelBottom.add(deleteButton, constraints);

eastPanel.add(eastPanelBottom);

add(eastPanel, BorderLayout.EAST);

//south panel

JPanel southPanel = new JPanel();

southPanel.setLayout(new GridLayout(1, 2));

JButton newButton = new JButton("New");

newButton.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent event) {

clearTextFields();

setFieldsEditable(true);

saveButton.setEnabled(true);

}

}

);

southPanel.add(newButton);

saveButton = new JButton("Save");

saveButton.setEnabled(false);

saveButton.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent event) {

try {

saveEmployee();

} catch (Exception e) {

handleFatalException(e);

}

}

}

);

southPanel.add(saveButton);

add(southPanel, BorderLayout.SOUTH);

//show the UI

setVisible(true);

}

private void deleteEmployee() {

Employee employee = getSelectedEmployee();

if (employee != null) {

int result = JOptionPane.showOptionDialog(this, "Are you sure you want to delete " + employee + "?", "Confirm Delete",

JOptionPane.YES\_NO\_OPTION, JOptionPane.QUESTION\_MESSAGE, null, null, null);

if (result == JOptionPane.YES\_OPTION) {

try {

employeeDAO.deleteEmployee(employee);

refreshEmployeeList();

} catch (Exception e) {

handleFatalException(e);

}

if (Integer.parseInt(idField.getText()) == employee.getId()) {

clearTextFields();

setFieldsEditable(false);

saveButton.setEnabled(false);

}

}

}

}

private void editEmployee() {

Employee employee = getSelectedEmployee();

if (employee != null) {

clearTextFields();

setFieldsEditable(true);

populateFieldsFromEmployee(employee);

saveButton.setEnabled(true);

}

}

private void viewEmployee() {

Employee employee = getSelectedEmployee();

if (employee != null) {

clearTextFields();

setFieldsEditable(false);

populateFieldsFromEmployee(employee);

saveButton.setEnabled(false);

}

}

private Employee getSelectedEmployee() {

int selectedIndex = employeeList.getSelectedIndex();

if (selectedIndex == -1) {

JOptionPane.showMessageDialog(null, "Please select an employee from the list.", "ERROR", JOptionPane.PLAIN\_MESSAGE);

return null;

}

return (Employee) employeeListModel.getElementAt(selectedIndex);

}

private void saveEmployee() throws Exception {

Employee employee = populateEmployeeFromFields();

if (employee != null) {

//if the ID is -1, this is a new employee - otherwise it's an update

if (employee.getId() != -1) {

employeeDAO.updateEmployee(employee);

} else {

employeeDAO.addEmployee(employee);

}

clearTextFields();

setFieldsEditable(false);

refreshEmployeeList();

saveButton.setEnabled(false);

}

}

private void refreshEmployeeList() throws Exception {

List<Employee> employees = employeeDAO.getAllEmployees();

employeeListModel.clear();

for (Employee employee : employees) {

log.fine("Adding employee to list: " + employee);

employeeListModel.addElement(employee);

}

}

private void setFieldsEditable(boolean b) {

for (JTextField textField : editableTextFields) {

textField.setEditable(b);

}

}

protected void clearTextFields() {

for (JTextField textField : allTextFields) {

textField.setText("");

}

idField.setText("-1");

}

private void handleFatalException(Exception e) {

JOptionPane.showMessageDialog(null, e.getMessage(), "ERROR", JOptionPane.PLAIN\_MESSAGE);

System.exit(1);

}

private Employee populateEmployeeFromFields() {

try {

Employee employee = new Employee();

employee.setId(Integer.parseInt(idField.getText()));

employee.setName(nameField.getText());

employee.setAddress(addressField.getText());

employee.setHours(getDoubleValue(hoursField.getText(), "Hours"));

employee.setRate(getDoubleValue(rateField.getText(), "Rate"));

String sex = sexField.getText();

if (sex.length() > 0) {

employee.setSex(sex.charAt(0));

} else {

employee.setSex(' ');

}

employee.setAge((int) getDoubleValue(ageField.getText(), "Age"));

employee.setActive(Boolean.valueOf(activeField.getText()));

employee.setSSN(ssnField.getText());

employee.setPhone(phoneField.getText());

return employee;

} catch (NumberFormatException nfe) {

JOptionPane.showMessageDialog(null, nfe.getMessage(), "ERROR", JOptionPane.PLAIN\_MESSAGE);

return null;

}

}

private void populateFieldsFromEmployee(Employee employee) {

NumberFormat dollarsFormat = new DecimalFormat("$0.00");

NumberFormat hoursFormat = new DecimalFormat("0.00");

idField.setText(String.valueOf(employee.getId()));

nameField.setText(employee.getName());

addressField.setText(employee.getAddress());

hoursField.setText(hoursFormat.format(employee.getHours()));

rateField.setText(hoursFormat.format(employee.getRate()));

sexField.setText(String.valueOf(employee.getSex()));

ageField.setText(String.valueOf(employee.getAge()));

activeField.setText(String.valueOf(employee.isActive()));

ssnField.setText(employee.getSSN());

phoneField.setText(employee.getPhone());

grossPayField.setText(dollarsFormat.format(employee.calculateGrossPay()));

federalTaxField.setText(dollarsFormat.format(employee.calculateFederalTax()));

stateTaxField.setText(dollarsFormat.format(employee.calculateStateTax()));

netPayField.setText(dollarsFormat.format(employee.calculateNetPay()));

}

private double getDoubleValue(String input, String fieldName) {

try {

return Double.valueOf(input);

} catch (NumberFormatException e) {

throw new NumberFormatException(fieldName + " must contain a numeric value!");

}

}

//addLabelAndTextField("Name", 100, true, centerPanel);

private JTextField addLabelAndTextField(String label, int fieldLength, boolean textFieldIsEditable, JPanel panel) {

panel.add(new JLabel(label));

JTextField textField = new JTextField(fieldLength);

textField.setEditable(false);

panel.add(textField);

if (textFieldIsEditable)

editableTextFields.add(textField);

allTextFields.add(textField);

return textField;

}

public static void main(String[] args) {

Level enabledLoggingLevel = Level.FINEST;

Logger.getLogger("finalproject").setLevel(enabledLoggingLevel);

Logger rootLogger = Logger.getLogger("");

Handler[] handlers = rootLogger.getHandlers();

for (Handler handler : handlers) {

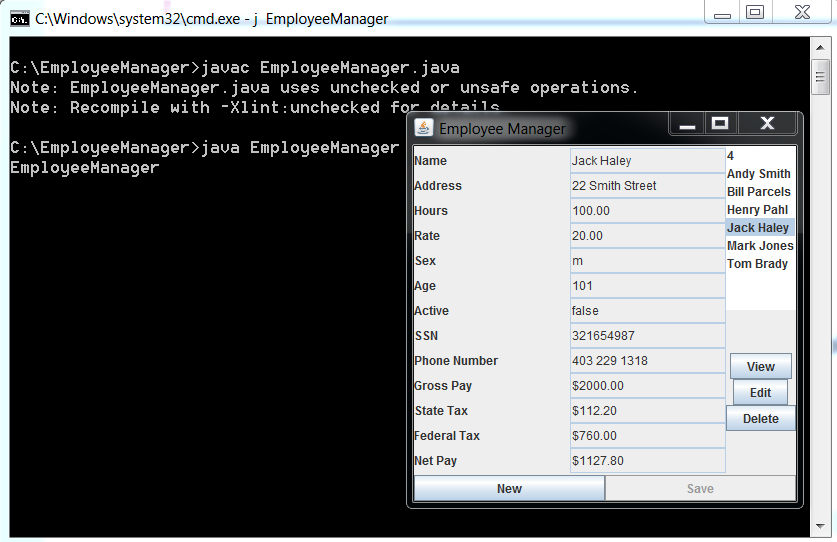
handler.setLevel(enabledLoggingLevel);

}

new EmployeeManager();

}

}



Hi,

I have had a few questions on how to run the packages example.

This will make it simple.

download the 2 attached files (Emprec.java and Emp\_Ex.java) to a directory that you create called

c:\packages

Now change to that directory (cd \packages should do the trick)

Next set that path to your compiler

path = "c:\program files\java\jdk1.6.0\_23\bin" (NOTE:You may have a newer compiler !!!!)

now type

javac -d c:\packages Emprec.java

what just happened is that you created Emprec.class but it was placed in the

c:\packages\com\jhaley\pk1 directory. THIS DIRECTORY WAS MADE BY JAVA NOT BY YOU THE PROGRAMMER.

THE ONLY DIRECTORY THAT YOU MADE WAS THE c:\packages DIRECTORY TO PLACE THE 2 .JAVA FILES IN. JAVA DID THE REST.

TO TEST THE PACKAGE TYPE

javac Emp\_Ex.java

java Emp\_Ex

Now you will see the output.

This is the simple way to do packages. Please note all the times the keywork "public" is used.

OK, here is a question. Why do we need to use "public" in the package code?

I will post this on the Discussion Board.

Good luck,

Jack

339 222 4247 Call with questions.

Emprec.java

//package file

package com.jhaley.pk1;

public class Emprec {

public String name;

public double hours;

public double rate;

public double calc\_gross\_pay() {

int i;

i = 1;

return(500.00);

}

}

Emp\_Ex.java

//import com.jhaley.pk1.\*;

import com.jhaley.pk1.Emprec;

//this class declares and object of type Emprec

class Emp\_Ex {

public static void main(String args[]) {

Emprec employee = new Emprec();

double employee\_gross\_pay;

// lets assign values to the employees instance variables

employee.name = "Mary Smith";

employee.hours = 40;

employee.rate = 15.25;

//lets compute the gross salary for this employee

employee\_gross\_pay = (employee.hours \* employee.rate);

//now lets print the output for this employee

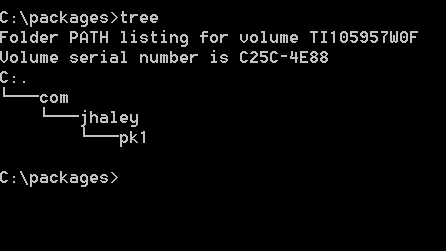
System.out.print(" The gross salary for " + employee.name);

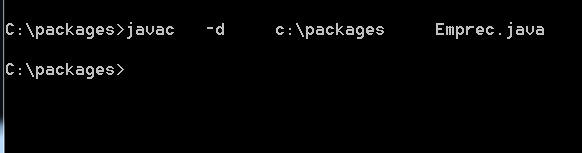
System.out.printf(" is %.2f \n",employee\_gross\_pay);

System.out.printf(" Calling the method the answer is %.2f \n", employee.calc\_gross\_pay());

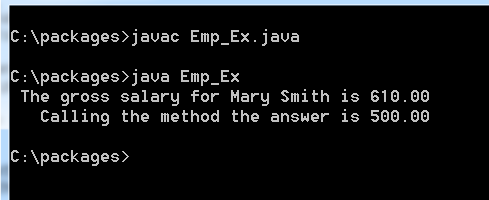
}//main

}//Emp\_Ex





Now to run the program



Here is how to do the Package example from a different directory !

CHECK THIS !!!!!!

javac -d c:\cdf -classpath "c:\cdf" Customer.java

1.) make sure that you make the directory

c:\cdf (or which ever directory that you want to make)

2.)

C:\packages4>javac -d c:\cdf Emprec.java

C:\packages4>javac -classpath ".;c:\cdf" Emp\_Ex.java

//PLEASE NOTE: do not make this package tree on both the root and c:\cdf directories

//or it will always go to the root when you make changes to the c:\cdf This is a common error !!!

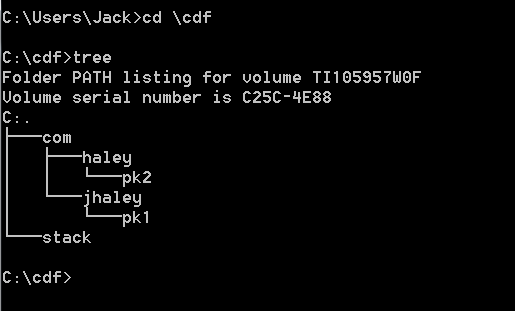
C:\packages4>java -classpath ".;c:\cdf" Emp\_Ex

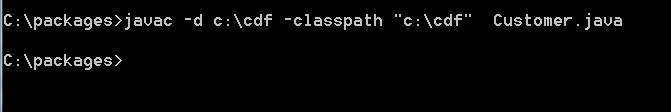
The gross salary for Mary Smith is 610.00

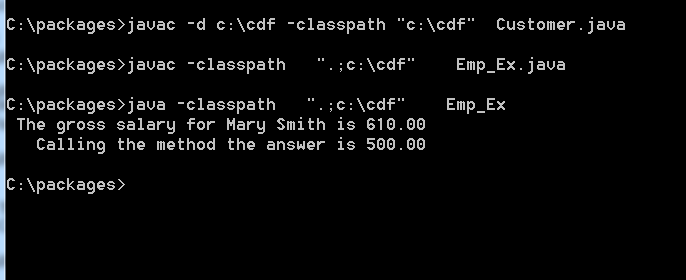
Calling the method the answer is 610.00

C:\packages4>

3.) Make sure that you use the quotes !!!!







//CD.java

package com.haley.pk2;

public class CD {

public double interest\_rate;

public int months;

}//class CD

//CD\_Iface.java

//package com.haley.pk2;

public interface CD\_Iface {

// there may never be any default implementation

// of any method found in an interface !!!

// this is an abstract method and

// must be defined by the implementing class

public double calc\_interest();

// double calc\_interest2();

//double cc() {return 100.0;}

}//class CD\_Iface

//Customer.java

package com.haley.pk2;

public class Customer extends CD implements CD\_Iface{

public String name;

public String address;

public double deposit;

// the implementing class must define the calc\_interest method

// or must itself be declared as abstract !!!!

public double calc\_interest()

{

// we will use the simple formula I = P \* R \* T;

//System.out.println(" the amount of interest earned is " + (deposit \* interest\_rate \* months/12));

return (deposit \* interest\_rate \* months/12);

}//calc\_interest

}// class Customer

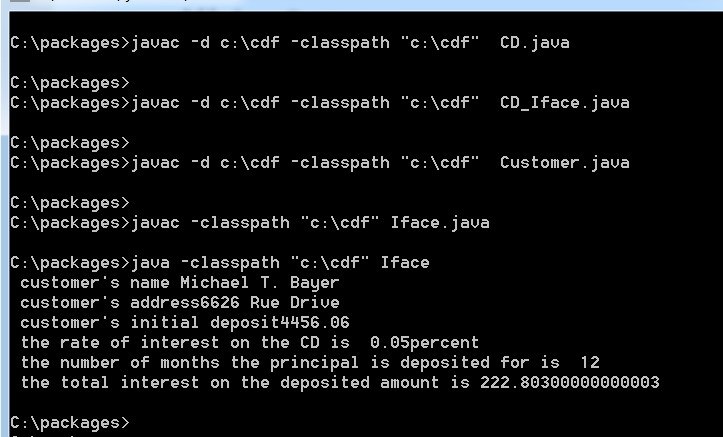
javac -d c:\cdf -classpath "c:\cdf" CD.java

javac -d c:\cdf -classpath "c:\cdf" CD\_Iface.java

javac -d c:\cdf -classpath "c:\cdf" Customer.java

javac -classpath "c:\cdf" Iface.java

java -classpath "c:\cdf" Iface





Happy Holidays!!!

C’est Finis !!!!!!