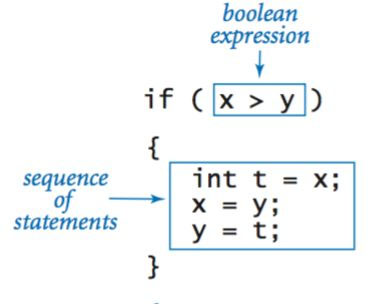
****

**BÁO CÁO ASSIGNMENT--- MÔN MOB1023**

****

**XÂY DỰNG CHƯƠNG TRÌNH GIAO DIỆN VỚI NGÔN NGỮ JAVA**

**GVHD:** Trần Quang Bình

**SVTH:** Nguyễn Thành Châu

**MSSV - Lớp:** PS27269 - IT18204

**TPHCM - 2023**

MỤC LỤC

Contents

[**I. MÔ TẢ BÀI TOÁN..................................................................................................**](#_Toc87558645)

[**II. XÂY DỰNG CHƯƠNG TRÌNH.................................................................................**](#_Toc87558647)

[**III. THỰC HIỆN...........................................................................................................**](#_Toc87558655)

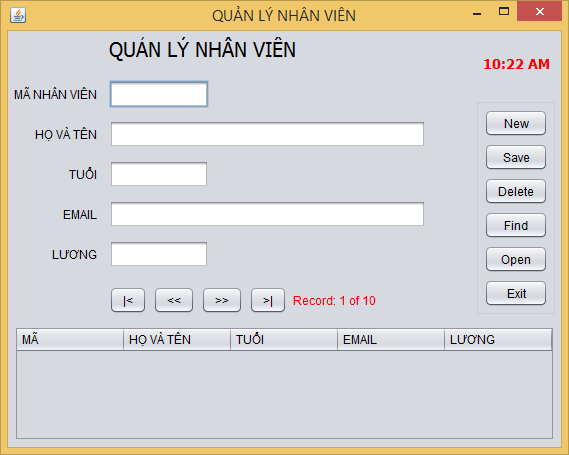
**IV. VALIDATEFORM....................................................................................................**

# MÔ TẢ BÀI TOÁN

Một công ty muốn quản lý nhân viên của mình bằng một chương trình máy tính và bạn là

người được công ty đó thuê để viết chương trình quản lý này. Người sử dụng chương trình sẽ

thực hiện các thao tác quản lý trên giao diện đồ họa như minh họa sau.



1. **XÂY DỰNG CHƯƠNG TRÌNH**

* Nút “Open” để đọc file dữ liệu và đưa vào List<Employee>
* Thông tin nhân viên đầu tiên trong List<Employee> được hiện lên form
* Nếu không có nhân viên nào thì các textbox sẽ để trống
* Dòng trạng thái cho biết số lượng nhân viên và thứ tự của nhân viên hiện tại
* Bảng hiển thị danh sách nhân viên trong List<Employee>
* Nút “Exit” dùng để ghi List<Employee> nhân viên vào file và thoát khỏi chương trình.
* Nút “New” dùng để xóa trắng các ô nhập trên form
* Nút “Find” dùng để tìm nhân viên theo mã, nếu có thì sẽ hiển thị chi tiết nhân viên đó trên

các textbox, nếu không có thì hiển thị cửa sổ JOptionPane thông báo không có nhân viên

này.

* Khi nhấp chuột vào một hàng trên bảng thì thông tin chi tiết được hiển thị lên form
* Nút “Save” sẽ thực hiện một trong 2 việc sau
* Bổ sung vào List<Employee> nếu trước đó nhấn nút “New”
* Cập nhật nhân viên đang xem nếu sửa đổi thông tin nhân viên đang xem
* Nút “Delete” dùng để xóa nhân viên theo id, nếu xóa thành công thì xuất hiện cửa sổ

JOptionPane thông báo đã xóa thành công. Cần kiểm lỗi không để trống mã.

* Kiểm các lỗi sau khi nhấp nút “Save”MOB1023 – *Lập trình Java 2 Assignment*
* Không để trống mã, không cho phép trùng mã nếu bổ sung mới.
* Tuổi phải từ 16 đến 55
* Lương phải trên 5 triệu
* Email phải đúng định dạng
* Các nút [ |< ],[ << ],[ >> ],[ >| ] để di chuyển tới các nhân viên ở vị trí tương ứng.

*Lưu ý*:Các thuộc tính của nhân viên trên đây là tối thiểu, bạn có thể bổ sung thêm

1. **THỰC HIỆN**

* Thiết kế giao diện như hình trên
* Xây dựng lớp NhanVien chú ý có implements Serializable.

public class NhanVien implements Serializable{

String manv, hoten, email;

double luong;

int tuoi;

public NhanVien() {

}

public NhanVien(String manv, String hoten, String email, double luong, int tuoi) {

this.manv = manv;

this.hoten = hoten;

this.email = email;

this.luong = luong;

this.tuoi = tuoi;

}

public String getManv() {

return manv;

}

public void setManv(String manv) {

this.manv = manv;

}

public String getHoten() {

return hoten;

}

public void setHoten(String hoten) {

this.hoten = hoten;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public double getLuong() {

return luong;

}

public void setLuong(double luong) {

this.luong = luong;

}

public int getTuoi() {

return tuoi;

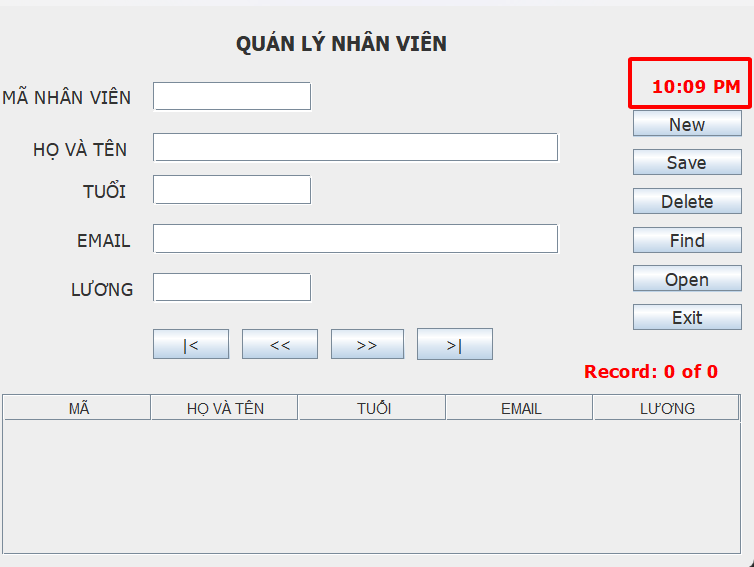
}

public void setTuoi(int tuoi) {

this.tuoi = tuoi;

}

Sau đó tạo Thread hàm tạo và start() thread để hiển thị đồng hồ hệ thống



* @Override
* public void run() {
* while(true) {
* try {
* Date now = new Date();
* SimpleDateFormat formater = new SimpleDateFormat();
* formater.applyPattern("hh:mm:ss aa");
* String time = formater.format(now);
* lblDongHo.setText(time);
* Thread.sleep(1000);
* } catch (Exception e) {
* break;
* }
* }
* }
* }
* Thread t1 = new Thread(this);
* t1.start();

Viết bổ sung vào lớp cửa sổ các trường và phương thức sau:

* **Mở file và nhập thông tin từ file vào List<Employee>**

public class XFile {

public static Object readObject(String path) {

try {

ObjectInputStream ois = new ObjectInputStream(new FileInputStream(path));

Object object = ois.readObject();

ois.close();

return object;

} catch (Exception e) {

throw new RuntimeException(e);

}

}

public static void writteObject(String path, Object object) {

try {

ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(path));

oos.writeObject(object);

oos.close();

} catch (Exception e) {

throw new RuntimeException(e);

}

}

}

* **Đọc File**

public void readFile() {

try {

em = (List<NhanVien>) XFile.readObject("ListNV.date");

fillToTable();

} catch (Exception ex) {

System.out.println(ex);

}

}

* **Lưu File**

public void saveFile() {

try {

Object em = null;

XFile.writteObject("ListNV.date",em);

JOptionPane.showMessageDialog(null, "Lưu thành công");

} catch (Exception ex) {

System.out.println(ex);

}

}

* **Ghi List<Employee> vào file và kết thúc chương trình**

Ghi sự kiện cho nút Exit câu lệnh:

saveFile(); và

System.exit(0); để kết thúc chương trình

* **Hiển thị List<Employee> lên bảng**
* **fillToTable(đổ lên bản)**

public void fillTable() {

tblModel.setRowCount(0);

for (NhanVien nv : list) {

tblModel.addRow(new Object[]{nv.getManv(), nv.getHoten(), nv.getTuoi(), nv.getEmail(), nv.getLuong()});

}

}

* **Tìm nhân viên và hiển thị lên form**
* **Viết trong Sự kiến nút Find**

**Duyệt ID cần tìm**

public NhanVien timTheoID(String ID) {

for (NhanVien nv : list) {

if(nv.getManv().equalsIgnoreCase(ID)) {

return nv;

}

}

return null;

}

public void showDetail(NhanVien nv) {

txtMaNV.setText(nv.getManv());

txtTen.setText(nv.getHoten());

txtTuoi.setText(String.valueOf(nv.getTuoi()));

txtEmail.setText(nv.getEmail());

txtLuong.setText(String.valueOf(nv.getLuong()));

}

**Viết sự kiện cho nút Find**

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

if(timTheoID(txtMaNV.getText())==null) {

JOptionPane.showMessageDialog(this, "Không có nhân viên bạn tìm!");

}else {

showDetail(timTheoID(txtMaNV.getText()));

}

}

* **Hiển thị nhân viên lên form khi click vào một hàng trên bảng**
* **showDetail(hiển thị lên bản)**

public void showDetail(int index) {

txtMaNV.setText(list.get(index).getManv());

txtTen.setText(list.get(index).getHoten());

txtTuoi.setText(String.valueOf(list.get(index).getTuoi()));

txtEmail.setText(list.get(index).getEmail());

txtLuong.setText(String.valueOf(list.get(index).getLuong()));

}

* **Sau đó tạo sự kiện mouseClick để đưa thông tin một hàng lên bảng**

private void tblNVMouseClicked(java.awt.event.MouseEvent evt) {

index = tblNV.getSelectedRow();

showDetail(index);

}

* **Hiển thị thông tin trạng thái** 

public String Record() {

if(tblNV.getSelectedRow() == 0) {

return "Record: 1" + " of " + list.size();

}

return "Record: " + (index + 1) + " of " + list.size();

}

* **Sau đó đưa vào sự kiện windowOpened để khi chạy sẽ hiện thị lên**

private void formWindowOpened(java.awt.event.WindowEvent evt) {

lblSotrang.setText(Record());

}

* **Thực hiện các nút điều hướng**
* **MLeft(Cuối Trái)**

public void MLeft() {

index = 0;

tblNV.setRowSelectionInterval(index, index);

showDetail(index);

lblSotrang.setText(Record());

}

* **MRight(Cuối Phải)**

public void MRight() {

index = list.size() - 1;

tblNV.setRowSelectionInterval(index, index);

showDetail(index);

lblSotrang.setText(Record());

}

* **Left(Qua trái)**

public void Left() {

if(index == 0) {

MRight();

}else {

index--;

}

tblNV.setRowSelectionInterval(index, index);

showDetail(index);

lblSotrang.setText(Record());

}

* **Right(Qua Phải)**

public void Right() {

if(index == list.size() - 1) {

MLeft();

} else{

index++;

}

tblNV.setRowSelectionInterval(index, index);

showDetail(index);

lblSotrang.setText(Record());

}

**Sau đó tạo Sự kiện cho các nút |< , << và >>, >|**

* **MLeft();**

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

MLeft();

}

* **Left();**

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

Left();

}

* **MRight();**

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

MRight();

}

* **Right();**

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

Right();

}

* **Thêm nhân viên và có kiểm tra dữ liệu phải nhập trước khi lưu**

public void addNV() {

if(index == -1) {

list.add(readForm());

fillTable();

JOptionPane.showMessageDialog(this, "Thêm nhân viên thành công!");

}

else {

updateNV(readForm());

fillTable();

JOptionPane.showMessageDialog(this, "Cập nhật nhân viên thành công!");

}

}

* **readFrom(đọc form)**

public NhanVien readForm() {

return new NhanVien(txtMaNV.getText(), txtTen.getText(), txtEmail.getText(), Double.parseDouble(txtLuong.getText()), Integer.valueOf(txtTuoi.getText()));

}

* **Cập nhật và có kiểm tra dữ liệu phải nhập trước khi lưu**

public void updateNV(NhanVien NNv) {

NhanVien nv1 = timTheoID(NNv.getManv());

if(nv1 != null) {

nv1.setHoten(NNv.getHoten());

nv1.setTuoi(NNv.getTuoi());

nv1.setEmail(NNv.getEmail());

nv1.setLuong(NNv.getLuong());

}

}

* **Xóa nhân viên**

public void deleteNV() {

list.remove(index);

fillTable();

clearForm();

JOptionPane.showMessageDialog(this, "Xóa thành công!");

}

1. **VALIDATEFORM**

* **Bắt lỗi nếu như chưa nhập vào thông tin và sai định dạng (Email)**

public boolean validateForm() {

if(txtMaNV.getText().equals("")) {

JOptionPane.showMessageDialog(this, "Vui long dien manv!");

return false;

}

if(txtTen.getText().equals("")) {

JOptionPane.showMessageDialog(this, "Vui long dien ten!");

return false;

}

if(txtTuoi.getText().equals("")) {

JOptionPane.showMessageDialog(this, "Vui long dien tuoi!");

return false;

}

try {

int tuoi = Integer.parseInt(txtTuoi.getText());

if(tuoi < 16 || tuoi > 65) {

JOptionPane.showMessageDialog(this, "Tuoi phai lon hon 16 nho hon 65!");

return false;

}

} catch (Exception e) {

JOptionPane.showMessageDialog(this, "Tuoi phai la chu so!");

return false;

}

if(txtEmail.getText().equals("")) {

JOptionPane.showMessageDialog(this, "Vui long dien email!");

return false;

}

Matcher matcher = Pattern.compile(M\_Email).matcher(txtEmail.getText());

if(!matcher.matches()) {

JOptionPane.showMessageDialog(this, "Email sai dinh dang!");

return false;

}

if(txtLuong.getText().equals("")) {

JOptionPane.showMessageDialog(this, "Vui long dien luong!");

return false;

}

try {

Double luong = Double.parseDouble(txtLuong.getText());

if(luong < 0) {

JOptionPane.showMessageDialog(this, "Luong phai lon hon 0!");

return false;

}

} catch (Exception e) {

JOptionPane.showMessageDialog(this, "Luong phai la so!");

return false;

}

return true;

}

**Khai báo định dạng cho Gmail :**

private String M\_Email = "^(?=.{1,64}@)[A-Za-z0-9\_-]+(\\.[A-Za-z0-9\_-]+)\*@[^-][A-Za-z0-9-]+(\\.[A-Za-z0-9-]+)\*(\\.[A-Za-z]{2,})$";