TD – Besoin de modéliser

À partir du code donné en annexe du td, répondez aux questions suivantes.

Question 1. *En une phrase, quels sont les rôles de chacune des classes ?*
Voici la liste des classes et leur rôle :
• Repertoire représente le répertoire (dans l'application il n'y en a qu'un).
• Personne représente une fiche d'une personne dans le répertoire.
• Adress représente une adresse postale.
• UIRepertoire correspond à la représentation graphique d'un répertoire à partir de laquelle les actions sur le répertoire pourront être appelées.
• UIPersonne correspond à la représentation graphique d'une fiche d'une personne à partir de laquelle les actions sur la fiche d'une personne pourront être appelées.
• UIMenuActionListener représente la classe chargée de capter tous les clics destinés à interagir avec un répertoire et d'appeler les traitements associés.
• MyAssistant est la classe qui contient la fonction principale (le main) de l'application.
Elle crée l'interface associée au répertoire géré par l'application

Question 2. *Peut-on dire qu'il existe des classes représentant des données et des classes représentant des interfaces graphiques ? Si oui, pourquoi et quelles sont ces classes ?*
Toutes les classes commençant par UI sont graphiques. Les classes UIRepertoire et UIPersonne représentent les interfaces graphiques permettant la gestion des répertoires et des personnes, tandis que la classe UIMenuActionListener est en charge de la gestion des actions sur les interfaces et de la réalisation des fonctionnalités associées à ces actions.
Les classes Repertoire, Personne et Adresse représentent les données manipulées par l'application.
La classe MyAssistant ne représente ni des données ni des interfaces graphiques ; elle sert à « lancer » l'application.

Les classes métiers (client) : personne, adress, repertoire
Les classes techniques (votre métier) : ui*

Question 3 *Est-il possible que le numéro de téléphone d'une personne soit +33 1 44 27 00 00 ?*
Oui, parce que les propriétés telephoneMaison, telephonePortable et telephoneBureau de la classe Personne sont de type string et qu'aucune vérification n'est faite sur sa valeur.

Question 4 *Est-il possible que l'adresse e-mail d'une personne soit « je_ne_veux_pas_donner_mon_email » ?*
Oui, parce que la propriété mail de la classe Personne (qui correspond à l'adresse email) est de type string et qu'aucune vérification n'est faite sur sa valeur.

Question 5 *Quelles sont les fonctionnalités proposées par les menus graphiques de cette application ?*

Il faut regarder les classes gérant les interfaces graphiques. Elles créent, dans notre cas, soit des menus déroulants, soit des boutons de soumission.

La classe UIRepertoire crée trois menus déroulants : Fichier, Organisation et Aide. Les fonctionnalités sont les éléments de chacun de ces trois menus. Mis à part le nom de ces fonctionnalités, le code de cette classe ne permet pas de connaître le service réalisé par chacune de ces fonctionnalités.
Voici chacun des menus avec les éléments qu'il contient :

• Menu Fichier :
  – Nouveau
  – Ouvrir
  – Enregistrer
  – Enregistrer Sous
• Menu Organisation :
  – Ajouter Nouvelle Personne
  – Rechercher Personne(s)
• Menu Aide :
  – A Propos

La classe UIPersonne est l'interface graphique associée à une personne. Elle crée deux boutons de soumission : Save et Cancel. Comme pour les menus déroulants, seul le nom des boutons peut nous renseigner
.

Question 6 *Quelles sont les fonctionnalités réellement réalisées par cette application ?*
Pour avoir la réponse, il faut regarder dans le code associé aux classes de gestion des interfaces.
Pour un répertoire, il s'agit de la classe UIMenuActionListener, dans le code de laquelle nous constatons que seules les fonctionnalités Ajouter Nouvelle Personne et Nouveau sont réalisées. Les autres fonctionnalités ne font qu'afficher un message.
Pour une personne, il n'y a pas de classe nommée associée à la gestion de l'interface.
Il faut regarder, dans le code de la classe UIPersonne, le code associé à la classe anonyme assurant la gestion de l'interface. Nous constatons alors que les fonctionnalités Save et Cancel sont réalisées.

Question 7 *Est-il possible de sauvegarder un répertoire dans un fichier ?*
C'est a priori possible puisque l'option Enregistrer Sous du menu Fichier le laisse supposer. Cependant, le code nous informe que cette fonctionnalité n'est pas encore réalisée.

Question 8 *Si vous aviez à rédiger un document décrivant tout ce que vous savez sur cette application afin qu'il puisse être lu par un développeur qui veut réutiliser cette application et un chef de projet qui souhaite savoir s'il peut intégrer cette application, quelles devraient être les caractéristiques de votre document ?*

Le document doit contenir toutes les informations relatives aux différentes vues (diversité), aux différents niveaux d'abstraction sans oublier les relations de cohérence reliant tous ces éléments.
Plus précisément, à partir du code de l'application, il est possible (mais pas simple) de rédiger une documentation :
• des services offerts par l'application (utile au développeur et au chef de projet) ;
• de conception de l'application (utile au développeur) ;
• d'architecture de l'application (utile au développeur) ;
• des logiciels nécessaires à l'utilisation de l'application (utile au développeur et au chef de projet).

Question 9 *Rédigez un document présentant l'application MyAssistant.*
L'application MyAssistant permet de gérer des répertoires. Un répertoire contient des informations relatives à des personnes. Pour chaque personne, il est possible de stocker un nom, un prénom, un numéro de téléphone du domicile, un numéro de téléphone du travail, un numéro de téléphone de portable, un numéro de fax, un titre, une société, une adresse et une adresse e-mail.

Question 10 *Rédigez un document décrivant les fonctionnalités de l'application MyAssistant.*

se fait en fonction du nom du menu associé et par similitude à ce qui se fait dans la majorité des applications.
Les fonctionnalités offertes par l'application sont de deux sortes. Les fonctionnalités sur un répertoire et les fonctionnalités sur une personne.
Les fonctionnalités de gestion d'un répertoire sont les suivantes :
• Créer un nouveau répertoire (menu Fichier/Nouveau).
• Ouvrir un répertoire déjà existant (menu Fichier/Ouvrir).
• Enregistrer un répertoire, c'est-à-dire enregistrer toutes les informations sur les personnes identifiées dans le répertoire (menu Fichier/Enregistrer). L'enregistrement se fait dans le fichier d'origine du répertoire s'il s'agit du répertoire déjà existant. Sinon, l'enregistrement se fait dans un nouveau fichier que l'utilisateur aura à identifier.

• Enregistrer un répertoire dans un autre fichier que le fichier d'origine, s'il existe (menu Fichier/Enregistrer Sous).
• Ajouter une nouvelle personne dans le répertoire ouvert (menu Organisation/
Ajouter Nouvelle Personne). Cette fonctionnalité propose à l'utilisateur de saisir
les informations correspondant à une nouvelle personne.
• Rechercher une personne (menu Organisation/Rechercher Personne). Cette fonctionnalité permet de rechercher la totalité des informations sur une personne en n'en saisissant qu'une partie (nom, numéro de téléphone, une partie du nom, etc.).
Il faudrait avoir le cahier des charges ou un code complet pour savoir exactement à partir de quoi la recherche peut être faite.
• Afficher l'aide sur l'application (menu Aide/A Propos). Cette fonctionnalité permet à l'utilisateur d'accéder à l'aide disponible sur l'application. Là aussi, il n'y a aucune information sur la forme de cette aide.
Les fonctionnalités de gestion d'une personne sont les suivantes
• Sauvegarder les informations saisies pour une personne (bouton Save de l'interface graphique associée à une personne). Attention : pour ajouter effectivement une personne à un répertoire, il faut enregistrer le répertoire. Se contenter de sauvegarder les informations relatives à la personne n'est pas suffisant.
• Annuler les modifications faites sur les informations d'une personne (bouton Cancel de l'interface graphique associée à une personne). Cette fonctionnalité annule toutes les modifications faites depuis la dernière sauvegarde des informations.

Question 11 *Rédigez un document décrivant l'architecture générale de l'application MyAssistant.*

Une solution (qui n'est pas unique) consiste à considérer un composant pour :
• l'interface graphique ;
• la base de données stockant les informations sur les personnes et les répertoires ;
• l'interface avec la base de données ;
• la réalisation de la « logique » des fonctionnalités de l'application.

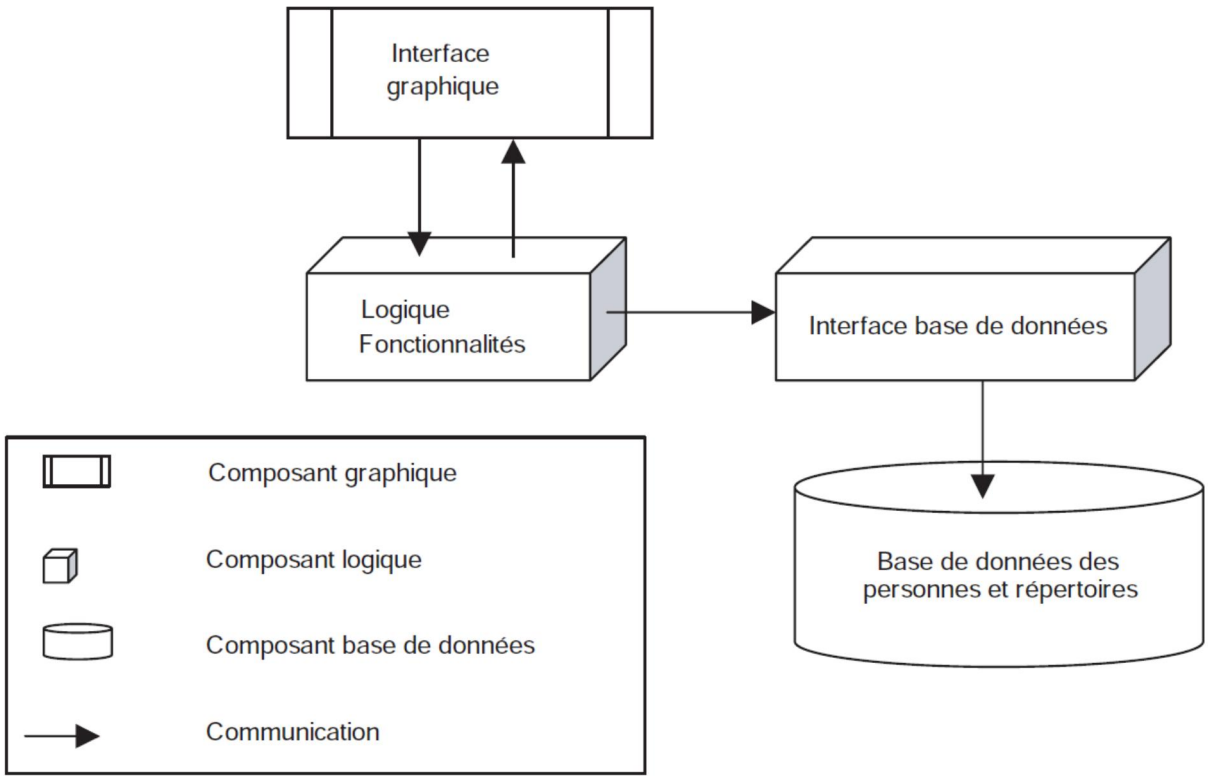
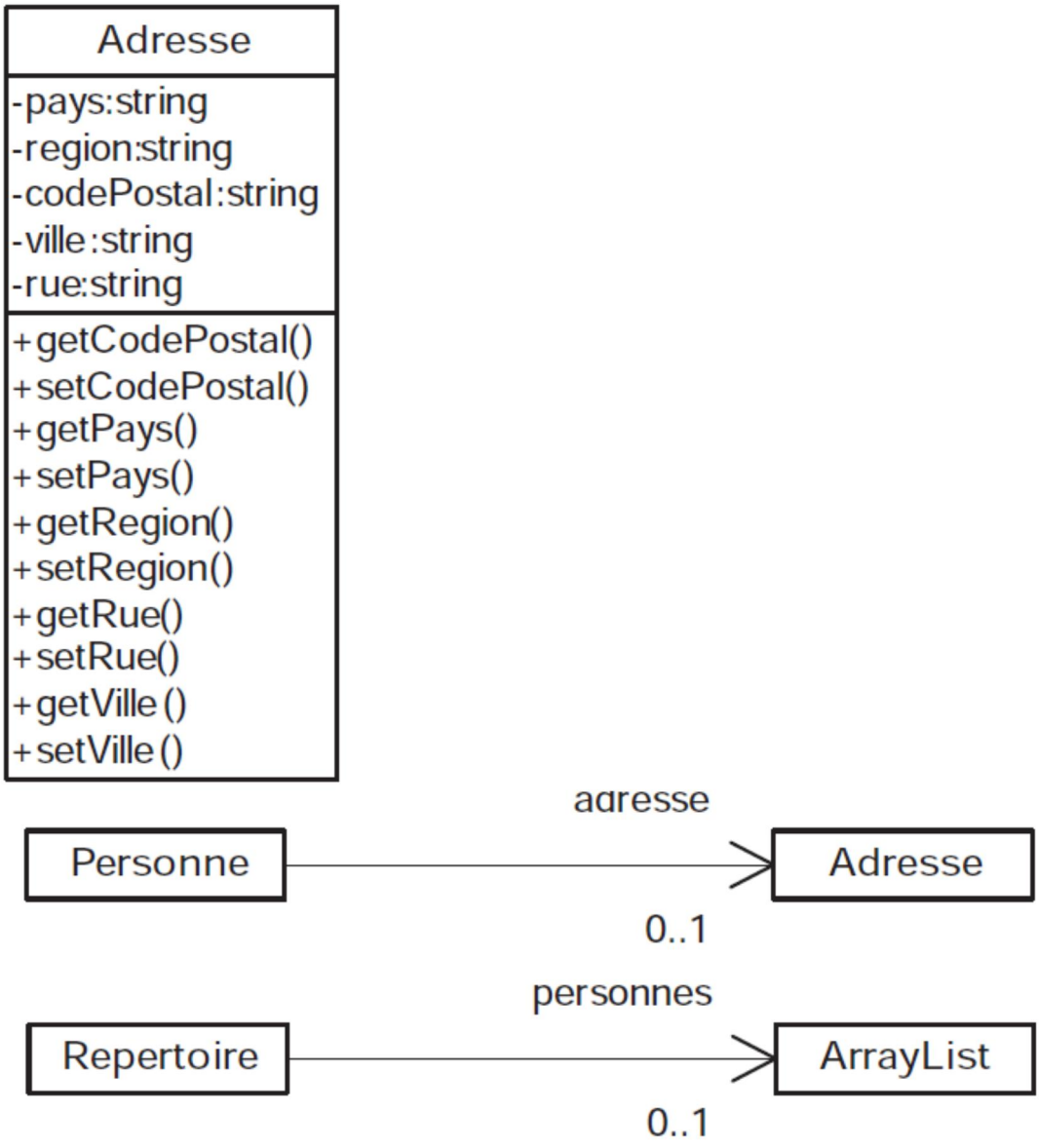


Figure 1
Architecture générale de MyAssistant


Adresse
-pays:string
-region:string
-codePostal:string
-ville:string
-rue:string
+getCodePostal()
+setCodePostal()
+getPays()
+setPays()
+getRegion()
+setRegion()
+getRue()
+setRue()
+getVille()
+setVille()
adresse
Personne
Adresse
0..1
personnes
Repertoire
ArrayList
0..1

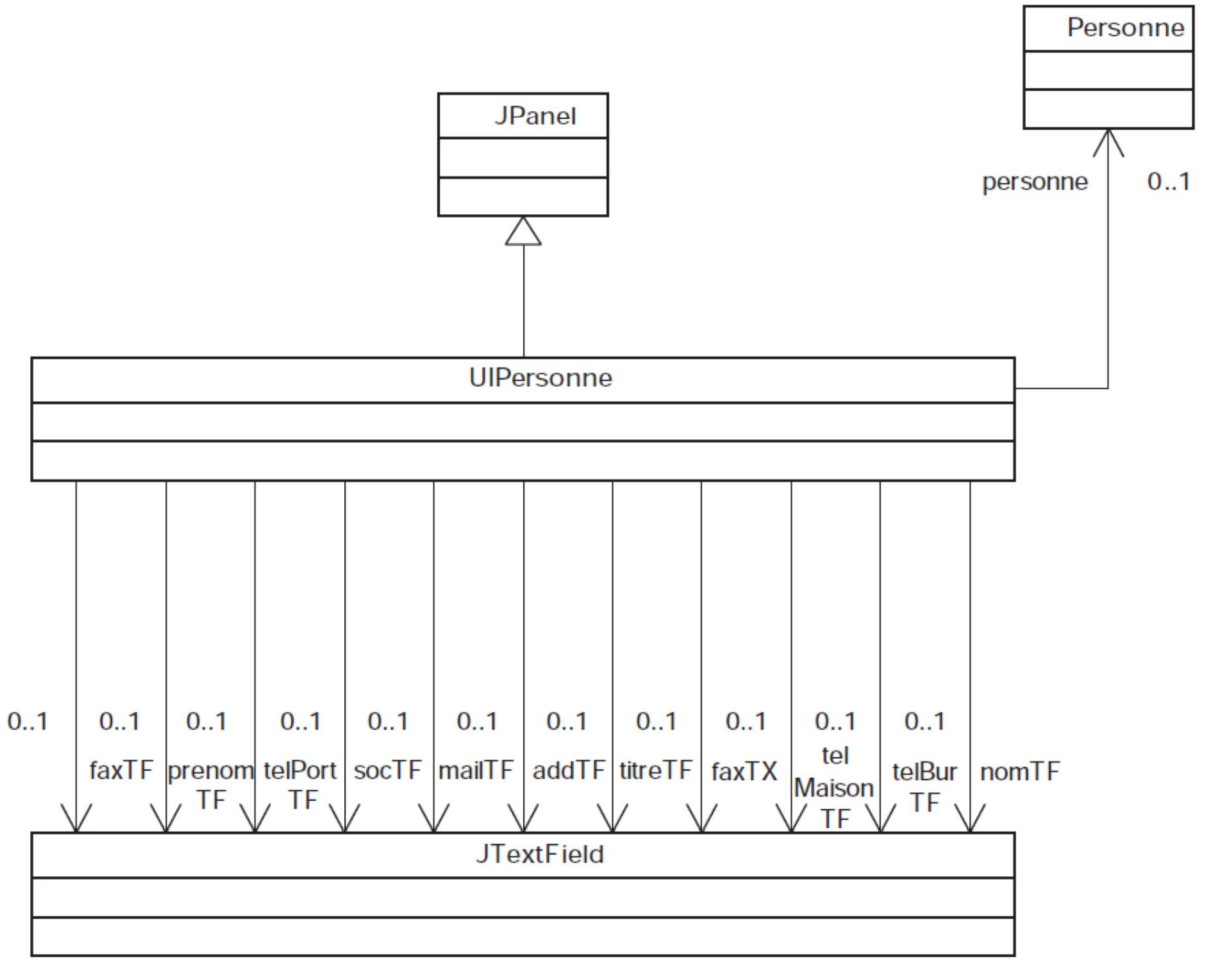


Figure 15

Associations entre UIPersonne *et* JTextField

*Continuer les questions de la partie reverse engineering**

```
/-------------------------------------------------------------------------------/
package repertoire;

public class MyAssistant {
	public static void main(String[] args) {
	UIRepertoire ihm = new UIRepertoire();
}
}
/-------------------------------------------------------------------------------/
package repertoire;

import java.awt.event.*;
```

```
import javax.swing.*;
import javax.swing.event.*;

public class UIRepertoire extends JFrame {
	Repertoire theRepertoire;
	UIMenuActionListener menuListener;
	JMenuBar menu_barre;
	JMenu repertoire_menu, fonction_menu, aide_menu;
	JMenuItem repertoire_menu_ouvrir,
		repertoire_menu_enregistrer,
		repertoire_menu_enregistrersous,
		repertoire_menu_nouveau,
		fonction_menu_ajouterPersonne,
		fonction_menu_rechercherPersonne,
		aide_menu_item;

JSplitPane splitPane;
JList repertoireView;

UIPersonne uipersonne;

public Repertoire getTheRepertoire() {
	return theRepertoire;
}

public void setTheRepertoire(Repertoire theRepertoire) {
	this.theRepertoire = theRepertoire;
refreshUIRepertoire();
}

public UIRepertoire() {
	super("Mon Repertoire");
	menuListener = new UIMenuActionListener(this);
	WindowListener l = new WindowAdapter() {
		public void windowClosing(WindowEvent e) {
			System.exit(0);
		}
		public void windowClosed(WindowEvent e) {
			System.exit(0);
		}
	};
	addWindowListener(l);
	init();
}


public UIRepertoire(Repertoire rep) {
	super("Mon Repertoire");
	theRepertoire = rep;
	menuListener = new UIMenuActionListener(this);
	WindowListener l = new WindowAdapter() {
		public void windowClosing(WindowEvent e) {
			System.exit(0);
		}
		public void windowClosed(WindowEvent e) {
			System.exit(0);
		}
	};
		addWindowListener(l);
		init();
		refreshUIRepertoire();
}

void init() {
```

```
		//Barre de Menu
		menu_barre = new JMenuBar();
		setJMenuBar(menu_barre);
		// Menu FICHIER
		repertoire_menu = new JMenu("Fichier");
		menu_barre.add(repertoire_menu);
		repertoire_menu_nouveau = new JMenuItem("Nouveau");
		repertoire_menu.add(repertoire_menu_nouveau);
		repertoire_menu_nouveau.addActionListener(menuListener);
		repertoire_menu_ouvrir = new JMenuItem("Ouvrir");
		repertoire_menu.add(repertoire_menu_ouvrir);
		repertoire_menu_ouvrir.addActionListener(menuListener);
		repertoire_menu_enregistrer = new JMenuItem("Enregistrer");
		repertoire_menu.add(repertoire_menu_enregistrer);
		repertoire_menu_enregistrer.addActionListener(menuListener);
		//fichier_menu_enregistrer.setMnemonic(KeyEvent.VK_S);
		repertoire_menu_enregistrersous = new JMenuItem("Enregistrer Sous");
		repertoire_menu.add(repertoire_menu_enregistrersous);
		repertoire_menu_enregistrersous.addActionListener(menuListener);

		// Menu FONCTION
		fonction_menu = new JMenu("Organisation");
		menu_barre.add(fonction_menu);
		fonction_menu_ajouterPersonne = new JMenuItem("Ajouter Nouvelle Personne");
		fonction_menu.add(fonction_menu_ajouterPersonne);
		fonction_menu_ajouterPersonne.addActionListener(menuListener);
		fonction_menu_rechercherPersonne = new JMenuItem("Rechercher Personne(s)");
		fonction_menu.add(fonction_menu_rechercherPersonne);
		fonction_menu_rechercherPersonne.addActionListener(menuListener);

		// Menu AIDE
		aide_menu = new JMenu("Aide");
		menu_barre.add(aide_menu);
		aide_menu_item = new JMenuItem("A Propos");
		aide_menu_item.addActionListener(menuListener);
		aide_menu.add(aide_menu_item);

		//Mettre un SplitPane
		splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT);
		getContentPane().add(splitPane);
		setVisible(true);
		pack();
	}

	public void refreshUIRepertoire() {
		// Mettre la JList à gauche
		repertoireView = new JList(theRepertoire.listerPersonnes());
		repertoireView.addListSelectionListener(new ListSelectionListener() {
			public void valueChanged(ListSelectionEvent e) {
				System.out.println("Ok");
				Personne p = (Personne) repertoireView.getSelectedValue();
				uipersonne.setPersonne(p);
			}
		});
		splitPane.setLeftComponent(new JScrollPane(repertoireView));

		//Test à droite
		if (theRepertoire.listerPersonnes().length!=0) {
			uipersonne = new UIPersonne(theRepertoire.listerPersonnes()[0]);
			splitPane.setRightComponent(uipersonne);
		}
	}
	}
```

```
/-------------------------------------------------------------------------------/
package repertoire;
import java.awt.GridLayout;
import java.awt.event.*;
import javax.swing.*;

public class UIPersonne extends JPanel {
	Personne personne;
	JTextField nomTF,
		prenomTF,
		telMaisonTF,
		telPortTF,
		telBurTF,
		faxTF,
		titreTF,
		socTF,
		addTF,
		mailTF;

public UIPersonne() {
	super();
	init();
}

public UIPersonne(Personne p) {
	super();
	personne = p;
	init();
}

public Personne getPersonne() {
	return personne;
}

public void setPersonne(Personne personne) {
	this.personne = personne;
	prenomTF.setText(personne.getPrenom());
	nomTF.setText(personne.getNom());
	telBurTF.setText(personne.getTelephoneBureau());
	telMaisonTF.setText(personne.getTelephoneMaison());
	telPortTF.setText(personne.getTelephonePortable());
	faxTF.setText(personne.getFax());
	titreTF.setText(personne.getTitre());
	socTF.setText(personne.getSociete());
	//Adresse
	mailTF.setText(personne.getMail());
}

public void init() {
	this.setLayout(new GridLayout(0, 2));
	add(new JLabel("nom"));
	nomTF = new JTextField("");
	add(nomTF);
	add(new JLabel("prenom"));
	prenomTF = new JTextField("");
	add(prenomTF);
	add(new JLabel("telephone maison"));
	telMaisonTF = new JTextField("");
	add(telMaisonTF);
	add(new JLabel("telephone portable"));
	telPortTF = new JTextField("");
	add(telPortTF);
	add(new JLabel("telephone bureau"));
	telBurTF = new JTextField("");
```

```
		add(telBurTF);
		add(new JLabel("fax"));
		faxTF = new JTextField("");
		add(faxTF);
		add(new JLabel("titre"));
		titreTF = new JTextField("");
		add(titreTF);
		add(new JLabel("société"));
		socTF = new JTextField("");
		add(socTF);
		add(new JLabel("adresse"));
		addTF = new JTextField("");
		add(addTF);
		add(new JLabel("mail"));
		mailTF = new JTextField("");
		add(mailTF);
		JButton save = new JButton("Save");
		save.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				personne.setPrenom(prenomTF.getText());
				personne.setNom(nomTF.getText());
				personne.setTelephoneBureau(telBurTF.getText());
				personne.setTelephoneMaison(telMaisonTF.getText());
				personne.setTelephonePortable(telPortTF.getText());
				personne.setFax(faxTF.getText());
				personne.setTitre(titreTF.getText());
				personne.setSociete(socTF.getText());
				//personne.setAdresse(addTF.getText());
				personne.setMail(mailTF.getText());
			}
		});

		add(save);
		JButton cancel = new JButton("Cancel");
		cancel.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				prenomTF.setText(personne.getPrenom());
				nomTF.setText(personne.getNom());
				telBurTF.setText(personne.getTelephoneBureau());
				telMaisonTF.setText(personne.getTelephoneMaison());
				telPortTF.setText(personne.getTelephonePortable());
				faxTF.setText(personne.getFax());
				titreTF.setText(personne.getTitre());
				socTF.setText(personne.getSociete());
				//Adresse
				mailTF.setText(personne.getMail());
			}
		});
		add(cancel);
		}
}

/---------------------------------------------------------------------------/
package repertoire;

import java.awt.event.*;
import javax.swing.JMenuItem;

public class UIMenuActionListener implements ActionListener {
		UIRepertoire uirep;

		public UIMenuActionListener(UIRepertoire uirep) {
			super();
			this.uirep = uirep;
```

```
	}

	public void actionPerformed(ActionEvent ev) {
		JMenuItem test = (JMenuItem) ev.getSource();
		if (test.getText() == "A Propos")
			System.out.println("Aide");
		else if (test.getText() == "Rechercher Personne(s)") {
			System.out.println("LOAD ");
		}
		else if (test.getText() == "Ajouter Nouvelle Personne") {
			System.out.println("Ajouter Nouvelle Personne ");
			Personne p = new Personne();
			uirep.getTheRepertoire().ajouterPersonne(p);
			uirep.refreshUIRepertoire();
		}
		else if (test.getText() == "Rechercher Personne(s)") {
			System.out.println("LOAD ");
		}
		else if (test.getText() == "Nouveau") {
			System.out.println("Nouveau ");
			uirep.setTheRepertoire(new Repertoire());
		}
		else if (test.getText() == "Enregistrer Sous") {
			System.out.println("LOAD ");
		}
		else if (test.getText() == "Enregistrer") {
			System.out.println("LOAD ");
		}
		else if (test.getText() == "Ouvrir") {
			System.out.println("LOAD ");
		}
	}
}
/--------------------------------------------------------------------------------/
package repertoire;

public class Adresse {
	String pays;
	String region;
	String codePostal;
	String ville;
	String rue;

	public String getCodePostal() {
		return codePostal;
	}

	public void setCodePostal(String codePostal) {
		this.codePostal = codePostal;
	}

	public String getPays() {
		return pays;
	}

	public void setPays(String pays) {
		this.pays = pays;
	}

	public String getRegion() {
		return region;
	}

	public void setRegion(String region) {
```

```
		this.region = region;
	}

	public String getRue() {
		return rue;
	}

	public void setRue(String rue) {
		this.rue = rue;
	}

	public String getVille() {
		return ville;
	}

	public void setVille(String ville) {
		this.ville = ville;
	}
}
/-------------------------------------------------------------------------------/
package repertoire;
public class Adresse {
	String pays;
	String region;
	String codePostal;
	String ville;
	String rue;

	public String getCodePostal() {
		return codePostal;
	}

	public void setCodePostal(String codePostal) {
		this.codePostal = codePostal;
	}

	public String getPays() {
		return pays;
	}

	public void setPays(String pays) {
		this.pays = pays;
	}

	public String getRegion() {
		return region;
	}

	public void setRegion(String region) {
		this.region = region;
	}

	public String getRue() {
		return rue;
	}

	public void setRue(String rue) {
		this.rue = rue;
	}

	public String getVille() {
		return ville;
	}
```

```
	public void setVille(String ville) {
		this.ville = ville;
	}
}
/-------------------------------------------------------------------------------/
package repertoire;

public class Personne {
	String nom;
	String prenom;
	String telephoneMaison;
	String telephonePortable;
	String telephoneBureau;
	String fax;
	String titre;
	String societe;
	Adresse adresse;
	String mail;

	public Adresse getAdresse() {
		return adresse;
	}

	public void setAdresse(Adresse adresse) {
		this.adresse = adresse;
	}

	public String getFax() {
		return fax;
	}
	public void setFax(String fax) {
		this.fax = fax;
	}
	public String getMail() {
		return mail;
	}
	public void setMail(String mail) {
		this.mail = mail;
	}

	public String getNom() {
		return nom;
	}

	public void setNom(String nom) {
		this.nom = nom;
	}

	public String getPrenom() {
		return prenom;
	}
	public void setPrenom(String prenom) {
		this.prenom = prenom;
	}
	public String getSociete() {
		return societe;
	}
	public void setSociete(String societe) {
		this.societe = societe;
	}
	public String getTelephoneBureau() {
		return telephoneBureau;
	}
	public void setTelephoneBureau(String telephoneBureau) {
```

```
		this.telephoneBureau = telephoneBureau;
	}
	public String getTelephoneMaison() {
		return telephoneMaison;
	}
	public void setTelephoneMaison(String telephoneMaison) {
		this.telephoneMaison = telephoneMaison;
	}
	public String getTelephonePortable() {
		return telephonePortable;
	}
	public void setTelephonePortable(String telephonePortable) {
		this.telephonePortable = telephonePortable;
	}
	public String getTitre() {
		return titre;
	}
	public void setTitre(String titre) {
		this.titre = titre;
	}
	public String toString() {
		return nom+" "+prenom;
	}
}

/-------------------------------------------------------------------------------/
package repertoire;

import java.util.Iterator;
import java.util.ArrayList;

public class Repertoire {
	ArrayList personnes;
	public void ajouterPersonne(Personne p) {
		personnes.add(p);
	}

public void supprimerPersonne(Personne p) {
	personnes.remove(p);
	}


public Personne[] rechercherPersonnesParNom(String nom) {
	ArrayList success = new ArrayList();
	for (Iterator it = personnes.iterator() ; it.hasNext() ;) {
		Personne current = (Personne) it.next();
		if (current.getNom().compareTo(nom)==0) success.add(current);
	}
	Personne[] res = new Personne[0];
	return (Personne[]) success.toArray(res);
}

public Personne[] listerPersonnes() {
	Personne[] res = new Personne[0];
	return (Personne[]) personnes.toArray(res);
}

public Repertoire() {
	personnes = new ArrayList();
}
}
```