JGenea - Guide Utilisateur

Traitement des dépouillements avec JGenea

par Templ

JGenea - Guide Utilisateur: Traitement des dépouillements avec JGenea

par Templ

Publié $Date$

Cette documentation détaille la mise en oeuvre de JGenea en vue de traiter des dépouillements de tables de communes. En effet, JGenea permet de calculer les relations entre les personnes à partir de tables.

Elle montrera le principe de l'application pour reconstituer ces relations à partir des naissances, baptèmes, mariages, décès et inhumations, ainsi que les limites de calcul dues aux manques d'informations et les évolutions à venir de cette fonctionnalités.

Enfin, nous prendrons un exemple pratique à partir du dépouillement de la commune de Beignon entre les années 1700 et 1746 réalisé par Olivier Point que nous remercions pour son aide.

Table des matières

Liste des tableaux

Liste des exemples

Partie I. Principe

JGenea est une application de généalogie écrite en java, opensource. Elle est en fait composée de sous-parties: un outil graphique pour constituer votre arbre généalogique, importer / exporter vos donées, les visualiser de manière interactive, rechercher , générer des documentations sur vos recherches dans différents formats (html, pdf) et gérer vos recherches généalogiques (courriers, recherches sur le web...) et un outil de publication web.

Une des fonctionnalités présentes en version beta dans la version 1.0 est celle de calcul des relations en les différentes personnes.

Le but est une meilleure connaissance des familles d'une commune et de faciliter (du fait du gain de temps) la recherche des enfants d'un couple. JGenea permet également de générer des documentations à partir d'une personne (ascendance ou descendance) ainsi que de visualiser ses arborescences ascendante et descendante.

Table des matières

Chapitre 1. Principe de calcul des relations

Le but est de regrouper tous les événements d'une personne (naissance, baptème, mariages, décès et inhumation) ainsi que trouver ses parents.

JGenea, au niveau de son modèle de données, fait la différence entre les informations concernant les personnes et les données des tables pour une commune. Quand les données sont saisies, modifiées ou supprimées au niveau d'une personne, ces données sont immédiatement synchronisées avec les tables de la commune correspondante.

Le principe du calcul des relations est de faire le travail inverse. Dés que l'on trouve un acte concerne une personne (qui a les mêmes nom et prénoms ainsi que les mêmes nom et prénoms pour les parents), on recherche si, précédemment, cette personne a des actes et si celle-ci a déjà été créée dans la table des personnes.

Ensuite il suffit de rechercher les parents en trouvant leur mariage. S'il n'est pas trouvé, deux personnes "vides" sont insérées dans la table des personnes.

Ce principe correspond aux révisions des classes suivantes:

- *org.genealogie.fusion.table.FusionTable*, révision 1.7
- *org.genealogie.fusion.table.ListeIdentifiantsParents*, révision 1.1
- *org.genealogie.fusion.table.ListeIdentifiantsPersonnes*, révision 1.1
- *org.genealogie.fusion.table.ListeRegistres*, révision 1.2
- *org.genealogie.fusion.table.ListeRegistresParents*, révision 1.1
- *org.genealogie.fusion.table.Parents*, révision 1.1

Chapitre 2. Améliorations envisagées

L'amélioration possible vient du fait que, dans l'algorithme actuel, deux passages dans les tables sont nécessaires: le premier pour regrouper les différents actes d'une personne et le second pour créer les relations de parenté entre les personnes.

Comme le parcours des tables se fait de manière chronologique, les informations concernant le(s) mariage(s) des parents ont déjà été trouvées. Il faudrait les mémoriser dans un vecteur ou dans une table de hachage pour créer les relations au moment de l'insertion de la personne.

Note

Ces informations sur les mariages seront gardées en mémoire pour une durée correspondant à l'écart maximal en le mariage et la naissance d'un enfant (configuré par défaut à 25 ans).

Il est également envisagé d'utiliser une base de données telle que Postgresql pour ces tests au lieu de Hypersonic qui semble peiner lors du calcul des relations entre les personnes.

Partie II. Cas pratique: dépouillement de la commune de Beignon (Morbihan), 1700 - 1746

Cette partie décrit la mise en oeuvre complète du traitement d'un dépouillement. Nous remercions encore Olivier Point pour nous avoir permis de réaliser cette mise en oeuvre sur son dépouillement de Beignon pour la période 1700 - 1746.

Table des matières

Chapitre 3. Importation des dépouillements dans JGenea

Pour pouvoir traiter les dépouillements dans JGenea, il faut d'abord les importer. Le format d'importation est le suivant:

Exemple 3.1. Format d'importation dans les tables

```
Nom;Prenom;Date;Type;Nom pere;Prenom pere;Age pere;Nom mere;Prenom mere;Age mere;
Sexe;Age;Origine
```

Deux possibilités sont alors envisageables:

- retravailler le dépouillement ou faire le celui-ci avec le format défini.
- coder un convertissement pour passer d'un format à l'autre.

Pour importer des données dans les tables, utiliser la commande suivante:

```
> html -import -villeid 2970 -fichier mon_fichier.txt
```

Note

Le fichier contient les lignes des tables et villeid correspond à l'identifiant de la ville dans JGenea. Pour l'obtenir, utiliser la commande suivante:

```
> ville -nom beignon
```

Attention

Pour chaque élément du dépouillement, il faut spécifier s'il s'agit d'un homme ou d'une femme. Ceci est indispensable pour le bon fonctionnement du traitement.

A l'occasion du traitement du dépouillement de Beignon, deux classes Java ont été développées pour le convertir au bon format.

Attention

Pour un mariage, il faut insérer deux lignes correspondants l'une au mari et l'autre à la femme.

Attention

Toutes les lignes dont le type (naissance, bapteme, mariage civil, mariage religieux, décès ou ihnumation) n'est pas reconnu ou la date est non correctement renseignée (elle doit être exacte) ne seront pas importées. Tous les rejets sont logués dans le fichier d'erreur (jgenea.aaaammjj.log).

Chapitre 4. Traitement des dépouillements dans JGenea

Pour lancer, le traitement du dépouillement, utiliser la commande suivante:

```
> fusion -table -villeid 2970
```

Note

Villeid correspond à l'identifiant de la ville dans JGenea. Pour l'obtenir, utiliser la commande suivante:

```
> ville -nom beignon
```

Chapitre 5. Génération des documentations Html

La génération html permet de voir rapidement le résultat de la génération et le publier soit sur internet (si l'auteur décide de rendre son dépouillement public) ou soit en local ou sur cd.

Pour générer, une documentation en Html décrivant les relations de parenté entre les personnes, utiliser la commande suivante:

```
> html -personne -repertoire ../html/
```

Pour générer, une documentation en Html décrivant les tables telles qu'elles sont été dépouillées (classement par année), utiliser la commande suivante:

```
> html -table -annee -repertoire ../html/
```

Chapitre 6. Synthèse

Les traitements ont été réalisés sur un PIII 500MHz avec 512Mo de RAM. Le système d'exploitation est un Linux Debian Woody.

```
bash$ uname -a
Linux kerion 2.4.18-686 #2 Wed Mar 20 20:21:31 EST 2002 i686 unknown
```

La version de java utilisé est 1.3.1 (voir le guide utilisateur [/docs/userguide/]) et de Hypersonic est 1.61.

Voici les temps de traitement constatés:

Note

Au départ la base ne contenait qu'un pays (la France), qu'un seul département (le Morbihan) et qu'une seule ville (Beignon). Les tables des "personnes" et des "tables" étaient vides.

Tableau 6.1. Résultats

Tâche	Temps
Importation des naissances	1m 51s
Importation des mariages	1m 48s
Fusion (état civil)	52m 16s
Fusion (parenté)	1h 50m 22s
Génération html (personnes)	10m 55s
Génération html (tables)	38s
Total	2h 55s

En conclusion, il est nécessaire d'optimiser le code afin de ne faire plus qu'un seul passage (de manière chronologique) dans les tables pour diminuer le temps de traitement ainsi que de tester cette fonctionnalité sur Postgresql.

Annexe A. Converion du format du dépouillement Beignon au format JGenea

Exemple A.1. Conversion des fichiers des baptèmes, mariages et inhumation au bon format (ConvertFiles.java)

```
import java.util.*;
import java.io.*;

/**
 * Classe de conversion de fichiers sql
 *
 * Format des tables des baptèmes
 *   Année;Date de naissance;Date de baptème;Nom;Prenom;Prenom père;Métier père;
 *     Prenom mère;Nom mère;Lieu
 *
 * Format des tables des mariages
 *   Année;Date fiancailles;Date publication;Date mariage;Prenom epoux;Nom epoux;
 *     Paroisse de l'époux;Prenom pere;Prenom Mere;Nom mere;Prenom epouse;Nom epouse;
 *     Paroisse épouse;Prenom pere;Prenom mere;Nom mere;Commentaires
 *
 * Format tables des sépultures
 *   Année;Date décès;Date sép.;Prénom;Nom;Prenom père;Nom mère;Prenom mere;
 *     Nom conjoint;Prénom conjoint;Age;Lieu;Commentaires
 *
 * @author  Templth
 * @version $Revision: $
 */

public class ConvertFiles {
    private String[] fichiers=new String[3];
    private Hashtable prenomsFeminins=new Hashtable();

    public ConvertFiles() {
        fichiers[0]="beignon-naissances.csv";
        fichiers[1]="beignon-mariages.csv";
        fichiers[2]="beignon-deces.csv";

        //Chargement des prenoms
        Vector feminins=chargerFichier("prenoms_feminins_beignon.txt");
        for(int cpt=0;cpt<feminins.size();cpt++) {
            String elt=(String)feminins.elementAt(cpt);
            if( prenomsFeminins.get(elt)==null )
                prenomsFeminins.put(elt,"");
        }
    }

    /**
     * Decoupage d'une ligne avec comme séparateur '\n'
     *
     * @param     ligne  ligne entrée
     * @return    liste des élémenets contenue dans un vecteur
     */
    private Vector getTokens(String ligne) {
        Vector tokens=new Vector();
        if( !ligne.equals("") ) {
            int indice=0;
            while( (indice=ligne.indexOf(";"))!=-1 ) {
                String token=ligne.substring(0,indice);
                tokens.addElement(token);
                ligne=ligne.substring(indice+1);
            }
```

```
            tokens.addElement(ligne);
        }
        return tokens;
    }

    /**
     * Charge les lignes d'un fichier dans un vecteur
     *
     * @param     fichier  le nom du fichier à charger
     * @return    le vecteur contenant les lignes du fichier
     */
    private Vector chargerFichier(String fichier) {
        Vector lignes=new Vector();
        try {
            BufferedReader br= new BufferedReader(new FileReader(fichier));
            String ligne="";

            while( (ligne=br.readLine())!=null ) {
                lignes.addElement(ligne);
            }
        } catch(Exception ex) {
            System.out.println(ex);
        }
        return lignes;
    }

    /**
     * Ecrit les lignes contenu dans un vecteur dans un fichier
     *
     * @param     fichier  le nom du fichier à charger
     * @param     lignes  le vecteur contenant les lignes à écrire
     */
    private void ecrireFichier(String fichier,Vector lignes) {
        try {
            FileWriter out=new FileWriter(fichier);
            for(int cpt=0;cpt<lignes.size();cpt++) {
                out.write((String)lignes.elementAt(cpt)+"\n");
            }
            out.close();
        } catch(Exception ex) {
            System.out.println(ex);
        }
    }

    /** Format des tables des baptèmes
     *   Année;Date de naissance;Date de baptème;Prenom;Nom;Prenom père;Métier père;
     *      Nom mère;Prenom mère;Lieu
     */
    private void convertNaissances() {
        Vector lignes=chargerFichier(fichiers[0]);
        Vector lignesGenerees=new Vector();
        for(int cpt=0;cpt<lignes.size();cpt++) {
            String ligne=(String)lignes.elementAt(cpt);
            ligne=ligne.trim();
            if( ligne.equals("") )
                continue;

            String annee="";
            String dateNaissance="";
            String dateBapteme="";
            String nom="";
            String prenom="";
            String nomPere="";
            String prenomPere="";
            String nomMere="";
            String prenomMere="";

            // Chargement de la ligne
            Vector tokens=getTokens(ligne);
            int i=0;
            for(int cpt1=0;cpt1<tokens.size();cpt1++) {
                String token=(String)tokens.elementAt(cpt1);
```

```
    token=token.trim();
    if( i==0 )
        annee=token;
    else if( i==1 )
        dateNaissance=token.replace('O','0')+"/"+annee;
    else if( i==2 )
        dateBapteme=token.replace('O','0')+"/"+annee;
    else if( i==3 )
        prenom=token;
    else if( i==4 )
        nom=token;
    else if( i==5 ) {
        nomPere=nom;
        prenomPere=token;
    } else if( i==6 ) {
        //metier
    } else if( i==7 )
        prenomMere=token;
    else if( i==8 )
        nomMere=token;
    i++;
}

//Ecriture de la nouvelle ligne
// Nom;Prenom;Date;Type;Nom pere;Prenom pere;Age pere;Nom mere;Prenom mere;
//   Age mere;Sexe;Age;Origine
StringBuffer ligneNaissance=new StringBuffer();
StringBuffer ligneBapteme=new StringBuffer();
if( !dateNaissance.startsWith("-") && !dateNaissance.equals("/"+annee) ) {
    ligneNaissance.append(nom);
    ligneNaissance.append(";");
    ligneNaissance.append(prenom);
    ligneNaissance.append(";");
    ligneNaissance.append(dateNaissance);
    ligneNaissance.append(";");
    ligneNaissance.append("naissance");
    ligneNaissance.append(";");
    ligneNaissance.append(nomPere);
    ligneNaissance.append(";");
    ligneNaissance.append(prenomPere);
    ligneNaissance.append(";");
    ligneNaissance.append("");
    ligneNaissance.append(";");
    ligneNaissance.append(nomMere);
    ligneNaissance.append(";");
    ligneNaissance.append(prenomMere);
    ligneNaissance.append(";");
    //Age mere
    ligneNaissance.append(";");
    //Sexe
    if( prenomsFeminins.get(prenom.toLowerCase())!=null )
        ligneNaissance.append("femme");
    else
        ligneNaissance.append("homme");
    ligneNaissance.append(";");
    //Age
    ligneNaissance.append(";");
    //Origine
    lignesGenerees.addElement(ligneNaissance.toString());
}

if( !dateBapteme.startsWith("-") && !dateBapteme.equals("/"+annee) ) {
    ligneBapteme.append(nom);
    ligneBapteme.append(";");
    ligneBapteme.append(prenom);
    ligneBapteme.append(";");
    ligneBapteme.append(dateBapteme);
    ligneBapteme.append(";");
    ligneBapteme.append("bapteme");
    ligneBapteme.append(";");
    ligneBapteme.append(nomPere);
    ligneBapteme.append(";");
```

```
                ligneBapteme.append(prenomPere);
                ligneBapteme.append(";");
                ligneBapteme.append("");
                ligneBapteme.append(";");
                ligneBapteme.append(nomMere);
                ligneBapteme.append(";");
                ligneBapteme.append(prenomMere);
                ligneBapteme.append(";");
                //Age mere
                ligneBapteme.append(";");
                //Sexe
                if( prenomsFeminins.get(prenom.toLowerCase())!=null )
                    ligneBapteme.append("femme");
                else
                    ligneBapteme.append("homme");
                ligneBapteme.append(";");
                //Age
                ligneBapteme.append(";");
                //Origine
                lignesGenerees.addElement(ligneBapteme.toString());
            }
        }
        ecrireFichier(fichiers[0]+".new",lignesGenerees);
    }

    /** Format des tables des mariages
     *   Année;Date fiancailles;Date publication;Date mariage;Prenom epoux;
     *     Nom epoux;Paroisse de l'époux;
     *     Prenom pere;Prenom Mere;Nom mere;Prenom epouse;Nom epouse;
     *     Paroisse épouse;Prenom pere;
     *     Prenom mere;Nom mere;Commentaires
     */
    private void convertMariages() {
        Vector lignes=chargerFichier(fichiers[1]);
        Vector lignesGenerees=new Vector();
        for(int cpt=0;cpt<lignes.size();cpt++) {
            String ligne=(String)lignes.elementAt(cpt);
            ligne=ligne.trim();
            if( ligne.equals("") )
                continue;

            String annee="";
            String dateMariage="";

            String nomMari="";
            String prenomMari="";
            String nomPereMari="";
            String prenomPereMari="";
            String nomMereMari="";
            String prenomMereMari="";
            String origineMari="";

            String nomFemme="";
            String prenomFemme="";
            String nomPereFemme="";
            String prenomPereFemme="";
            String nomMereFemme="";
            String prenomMereFemme="";
            String origineFemme="";

            // Chargement de la ligne
            Vector tokens=getTokens(ligne);
            int i=0;
            for(int cpt1=0;cpt1<tokens.size();cpt1++) {
                String token=(String)tokens.elementAt(cpt1);
                token=token.trim();
                if( i==0 ) {
                    annee=token;
                } else if( i==1 ) {
                    //Fiancailles
                } else if( i==2 ) {
                    //Publications
```

```
        } else if( i==3 ) {
            dateMariage=token.replace('O','0')+"/"+annee;
        } else if( i==4 ) {
            prenomMari=token;
        } else if( i==5 ) {
            nomMari=token;
        } else if( i==6 ) {
            if( !token.equals("") )
                origineMari="France,56,"+token;
        } else if( i==7 ) {
            nomPereMari=nomMari;
            prenomPereMari=token;
        } else if( i==8 ) {
            prenomMereMari=token;
        } else if( i==9 ) {
            nomMereMari=token;
        } else if( i==10 ) {
            prenomFemme=token;
        } else if( i==11 ) {
            nomFemme=token;
        } else if( i==12 ) {
            if( !token.equals("") )
                origineFemme="France,56,"+token;
        } else if( i==13 ) {
            nomPereFemme=nomFemme;
            prenomPereFemme=token;
        } else if( i==14 ) {
            prenomMereFemme=token;
        } else if( i==15 ) {
            nomMereFemme=token;
        }
        i++;
    }

    //Ecriture de la nouvelle ligne
    // Nom;Prenom;Date;Type;Nom pere;Prenom pere;Age pere;Nom mere;Prenom mere;
    //    Age mere;Sexe;Age;Origine
    StringBuffer ligneMariageMari=new StringBuffer();
    StringBuffer ligneMariageFemme=new StringBuffer();
    if( !dateMariage.startsWith("-") && !dateMariage.equals("/"+annee) ) {
        ligneMariageMari.append(nomMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append(prenomMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append(dateMariage);
        ligneMariageMari.append(";");
        ligneMariageMari.append("mariage religieux");
        ligneMariageMari.append(";");
        ligneMariageMari.append(nomPereMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append(prenomPereMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append("");
        ligneMariageMari.append(";");
        ligneMariageMari.append(nomMereMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append(prenomMereMari);
        ligneMariageMari.append(";");
        ligneMariageMari.append(";");
        ligneMariageMari.append("homme");
        ligneMariageMari.append(";");
        ligneMariageMari.append(";");
        ligneMariageMari.append(origineMari);
        lignesGenerees.addElement(ligneMariageMari.toString());

        ligneMariageFemme.append(nomFemme);
        ligneMariageFemme.append(";");
        ligneMariageFemme.append(prenomFemme);
        ligneMariageFemme.append(";");
        ligneMariageFemme.append(dateMariage);
        ligneMariageFemme.append(";");
        ligneMariageFemme.append("mariage religieux");
```

```
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(nomPereFemme);
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(prenomPereFemme);
                ligneMariageFemme.append(";");
                ligneMariageFemme.append("");
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(nomMereFemme);
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(prenomMereFemme);
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(";");
                ligneMariageFemme.append("femme");
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(";");
                ligneMariageFemme.append(origineFemme);
                lignesGenerees.addElement(ligneMariageFemme.toString());
            }
        }
        ecrireFichier(fichiers[1]+".new",lignesGenerees);
    }

    /** Format tables des sépultures
     *   Année;Date décès;Date sép.;Prénom;Nom;Prenom père;Nom mère;Prenom mere;
     *     Nom conjoint;Prénom conjoint;Age;Lieu;Commentaires
     */
    private void convertDeces() {
        Vector lignes=chargerFichier(fichiers[2]);
        Vector lignesGenerees=new Vector();
        for(int cpt=0;cpt<lignes.size();cpt++) {
            String ligne=(String)lignes.elementAt(cpt);
            ligne=ligne.trim();
            if( ligne.equals("") )
                continue;

            String annee="";
            String dateDeces="";
            String dateInhumation="";
            String nom="";
            String prenom="";
            String nomPere="";
            String prenomPere="";
            String nomMere="";
            String prenomMere="";
            int age=0;

            // Chargement de la ligne
            Vector tokens=getTokens(ligne);
            int i=0;
            for(int cpt1=0;cpt1<tokens.size();cpt1++) {
                String token=(String)tokens.elementAt(cpt1);
                token=token.trim();
                if( i==0 )
                    annee=token;
                else if( i==1 )
                    dateDeces=token.replace('O','0')+"/"+annee;
                else if( i==2 )
                    dateInhumation=token.replace('O','0')+"/"+annee;
                else if( i==3 )
                    prenom=token;
                else if( i==4 )
                    nom=token;
                else if( i==5 ) {
                    nomPere=nom;
                    prenomPere=token;
                } else if( i==6 ) {
                    prenomMere=token;
                } else if( i==7 ) {
                    nomMere=token;
                } else if( i==8 ) {
                    //prenom conjoint
                } else if( i==9 ) {
```

```
            //nom conjoint
        } else if( i==10 ) {
            int indice=0;
            if( (indice=token.indexOf(" "))!=-1 )
            try {
                age=Integer.parseInt((token.substring(0,indice)).trim());
            } catch(Exception ex) {}
            if( token.indexOf("mois")!=-1 )
                age=0;
        }
        i++;
    }

    //Ecriture de la nouvelle ligne
    // Nom;Prenom;Date;Type;Nom pere;Prenom pere;Age pere;Nom mere;Prenom mere;
    //   Age mere;Sexe;Age;Origine
    StringBuffer ligneDeces=new StringBuffer();
    StringBuffer ligneInhumation=new StringBuffer();
    if( !dateDeces.startsWith("-") && !dateDeces.equals("/"+annee) ) {
        ligneDeces.append(nom);
        ligneDeces.append(";");
        ligneDeces.append(prenom);
        ligneDeces.append(";");
        ligneDeces.append(dateDeces);
        ligneDeces.append(";");
        ligneDeces.append("deces");
        ligneDeces.append(";");
        ligneDeces.append(nomPere);
        ligneDeces.append(";");
        ligneDeces.append(prenomPere);
        ligneDeces.append(";");
        ligneDeces.append("");
        ligneDeces.append(";");
        ligneDeces.append(nomMere);
        ligneDeces.append(";");
        ligneDeces.append(prenomMere);
        ligneDeces.append(";");
        ligneDeces.append(";");
        //Sexe
        if( prenomsFeminins.get(prenom.toLowerCase())!=null )
            ligneDeces.append("femme");
        else
            ligneDeces.append("homme");
        ligneDeces.append(";");
        if( age!=0 )
            ligneDeces.append(String.valueOf(age));
        ligneDeces.append(";");
        lignesGenerees.addElement(ligneDeces.toString());
    }

    if( !dateInhumation.startsWith("-") && !dateInhumation.equals("/"+annee) ) {
        ligneInhumation.append(nom);
        ligneInhumation.append(";");
        ligneInhumation.append(prenom);
        ligneInhumation.append(";");
        ligneInhumation.append(dateInhumation);
        ligneInhumation.append(";");
        ligneInhumation.append("inhumation");
        ligneInhumation.append(";");
        ligneInhumation.append(nomPere);
        ligneInhumation.append(";");
        ligneInhumation.append(prenomPere);
        ligneInhumation.append(";");
        ligneInhumation.append("");
        ligneInhumation.append(";");
        ligneInhumation.append(nomMere);
        ligneInhumation.append(";");
        ligneInhumation.append(prenomMere);
        ligneInhumation.append(";");
        ligneInhumation.append(";");
        //Sexe
        if( prenomsFeminins.get(prenom.toLowerCase())!=null )
```

```
                    ligneInhumation.append("femme");
                else
                    ligneInhumation.append("homme");
                ligneInhumation.append(";");
                if( age!=0 )
                    ligneInhumation.append(String.valueOf(age));
                ligneInhumation.append(";");
                lignesGenerees.addElement(ligneInhumation.toString());
            }
        }
        ecrireFichier(fichiers[2]+".new",lignesGenerees);
    }

    public void convert() {
        convertNaissances();
        convertMariages();
        convertDeces();
    }

    public static void main(String[] args) {
        ConvertFiles cf=new ConvertFiles();
        cf.convert();
    }
}
```

Annexe B. Récupération des prénoms pour déterminer le sexe des personnes

Exemple B.1. Récupération des prénoms (GetDistinctsPrenoms.java)

```
import java.util.*;
import java.io.*;

/**
 * Classe de conversion de fichiers sql
 *
 * Format des tables des baptèmes
 *   Année;Date de naissance;Date de baptème;Nom;Prenom;Prenom père;Métier père;
 *     Prenom mère;Nom mère;Lieu
 *
 * Format des tables des mariages
 *   Année;Date fiancailles;Date publication;Date mariage;Prenom epoux;Nom epoux;
 *     Paroisse de l'époux;Prenom pere;Prenom Mere;Nom mere;Prenom epouse;Nom epouse;
 *       Paroisse épouse;Prenom pere;Prenom mere;Nom mere;Commentaires
 *
 * Format tables des sépultures
 *   Année;Date décès;Date sép.;Prénom;Nom;Prenom père;Nom mère;Prenom mere;Nom conjoint;
 *     Prénom conjoint;Age;Lieu;Commentaires
 *
 * @author  Templier Thierry
 * @version $Revision: $
 */

public class GetDistinctsPrenoms {
    private String[] fichiers=new String[3];

    public GetDistinctsPrenoms() {
        fichiers[0]="beignon-naissances.csv";
        fichiers[1]="beignon-mariages.csv";
        fichiers[2]="beignon-deces.csv";
    }

        /**
         * Decoupage d'une ligne avec comme séparateur '\n'
         *
         * @param     ligne  ligne entrée
         * @return    liste des élémenets contenue dans un vecteur
         */
    private Vector getTokens(String ligne) {
        Vector tokens=new Vector();
        if( !ligne.equals("") ) {
            int indice=0;
            while( (indice=ligne.indexOf(";"))!=-1 ) {
                String token=ligne.substring(0,indice);
                tokens.addElement(token);
                ligne=ligne.substring(indice+1);
            }
            tokens.addElement(ligne);
        }
        return tokens;
    }

        /**
         * Charge les lignes d'un fichier dans un vecteur
         *
         * @param     fichier  le nom du fichier à charger
         * @return    le vecteur contenant les lignes du fichier
```

```
     */
    private Vector chargerFichier(String fichier) {
        Vector lignes=new Vector();
        try {
            BufferedReader br= new BufferedReader(new FileReader(fichier));
            String ligne="";

            while( (ligne=br.readLine())!=null ) {
                lignes.addElement(ligne);
            }
        } catch(Exception ex) {
            System.out.println(ex);
        }
        return lignes;
    }

        /**
         * Ecrit les lignes contenu dans un vecteur dans un fichier
         *
         * @param     fichier  le nom du fichier à charger
         * @param     lignes  le vecteur contenant les lignes à écrire
         */
    private void ecrireFichier(String fichier,Vector lignes) {
        try {
            FileWriter out=new FileWriter(fichier);
            for(int cpt=0;cpt<lignes.size();cpt++) {
                out.write((String)lignes.elementAt(cpt)+"\n");
            }
            out.close();
        } catch(Exception ex) {
            System.out.println(ex);
        }
    }

    /** Format des tables des baptèmes
     *   Année;Date de naissance;Date de baptème;Prenom;Nom;Prenom père;Métier père;
     *     Nom mère;Prenom mère;Lieu
     */
    private Hashtable convertNaissances(Hashtable prenoms) {
        Vector lignes=chargerFichier(fichiers[0]);
        Vector lignesGenerees=new Vector();
        for(int cpt=0;cpt<lignes.size();cpt++) {
            String ligne=(String)lignes.elementAt(cpt);
            ligne=ligne.trim();
            if( ligne.equals("") )
                continue;

            String annee="";
            String dateNaissance="";
            String dateBapteme="";
            String nom="";
            String prenom="";
            String nomPere="";
            String prenomPere="";
            String nomMere="";
            String prenomMere="";

            // Chargement de la ligne
            Vector tokens=getTokens(ligne);
            int i=0;
            for(int cpt1=0;cpt1<tokens.size();cpt1++) {
                String token=(String)tokens.elementAt(cpt1);
                token=token.trim();
                if( i==0 )
                    annee=token;
                else if( i==1 )
                    dateNaissance=token+"/"+annee;
                else if( i==2 )
                    dateBapteme=token+"/"+annee;
                else if( i==3 )
                    prenom=token.toLowerCase();
                else if( i==4 )
```

```
                nom=token.toLowerCase();
            else if( i==5 ) {
                nomPere=nom;
                prenomPere=token.toLowerCase();
            } else if( i==6 ) {
                //metier
            } else if( i==7 )
                prenomMere=token.toLowerCase();
            else if( i==8 )
                nomMere=token.toLowerCase();
            i++;
        }

        //Checker les prenoms
        if( prenoms.get(prenom)==null )
            prenoms.put(prenom,"");
        if( prenoms.get(prenomPere)==null )
            prenoms.put(prenomPere,"");
        if( prenoms.get(prenomMere)==null )
            prenoms.put(prenomMere,"");
    }
    return prenoms;
}

/** Format des tables des mariages
 *   Année;Date fiancailles;Date publication;Date mariage;Prenom epoux;Nom epoux;
 *     Paroisse de l'époux;Prenom pere;Prenom Mere;Nom mere;Prenom epouse;Nom epouse;
 *     Paroisse épouse;Prenom pere;Prenom mere;Nom mere;Commentaires
 */
private Hashtable convertMariages(Hashtable prenoms) {
    Vector lignes=chargerFichier(fichiers[1]);
    Vector lignesGenerees=new Vector();
    for(int cpt=0;cpt<lignes.size();cpt++) {
        String ligne=(String)lignes.elementAt(cpt);
        ligne=ligne.trim();
        if( ligne.equals("") )
            continue;

        String annee="";
        String dateMariage="";

        String nomMari="";
        String prenomMari="";
        String nomPereMari="";
        String prenomPereMari="";
        String nomMereMari="";
        String prenomMereMari="";
        String origineMari="";

        String nomFemme="";
        String prenomFemme="";
        String nomPereFemme="";
        String prenomPereFemme="";
        String nomMereFemme="";
        String prenomMereFemme="";
        String origineFemme="";

        // Chargement de la ligne
        Vector tokens=getTokens(ligne);
        int i=0;
        for(int cpt1=0;cpt1<tokens.size();cpt1++) {
            String token=(String)tokens.elementAt(cpt1);
            token=token.trim();
            if( i==0 ) {
                annee=token;
            } else if( i==1 ) {
                //Fiancailles
            } else if( i==2 ) {
                //Publications
            } else if( i==3 ) {
                dateMariage=token+"/"+annee;
            } else if( i==4 ) {
```

```
                prenomMari=token.toLowerCase();
            } else if( i==5 ) {
                nomMari=token.toLowerCase();
            } else if( i==6 ) {
                origineMari="France,56,"+token;
            } else if( i==7 ) {
                nomPereMari=nomMari;
                prenomPereMari=token.toLowerCase();
            } else if( i==8 ) {
                prenomMereMari=token.toLowerCase();
            } else if( i==9 ) {
                nomMereMari=token.toLowerCase();
            } else if( i==10 ) {
                prenomFemme=token.toLowerCase();
            } else if( i==11 ) {
                nomFemme=token.toLowerCase();
            } else if( i==12 ) {
                origineFemme="France,56,"+token;
            } else if( i==13 ) {
                nomPereFemme=nomFemme;
                prenomPereFemme=token.toLowerCase();
            } else if( i==14 ) {
                prenomMereFemme=token.toLowerCase();
            } else if( i==15 ) {
                nomMereFemme=token.toLowerCase();
            }
            i++;
        }

        //Checker les prenoms
        if( prenoms.get(prenomMari)==null )
            prenoms.put(prenomMari,"");
        if( prenoms.get(prenomPereMari)==null )
            prenoms.put(prenomPereMari,"");
        if( prenoms.get(prenomMereMari)==null )
            prenoms.put(prenomMereMari,"");
        if( prenoms.get(prenomFemme)==null )
            prenoms.put(prenomFemme,"");
        if( prenoms.get(prenomPereFemme)==null )
            prenoms.put(prenomPereFemme,"");
        if( prenoms.get(prenomMereFemme)==null )
            prenoms.put(prenomMereFemme,"");
    }
    return prenoms;
}

/** Format tables des sépultures
 *   Année;Date décès;Date sép.;Prénom;Nom;Prenom père;Nom mère;Prenom mere;
 *     Nom conjoint;Prénom conjoint;Age;Lieu;Commentaires
 */
private Hashtable convertDeces(Hashtable prenoms) {
    Vector lignes=chargerFichier(fichiers[2]);
    Vector lignesGenerees=new Vector();
    for(int cpt=0;cpt<lignes.size();cpt++) {
        String ligne=(String)lignes.elementAt(cpt);
        ligne=ligne.trim();
        if( ligne.equals("") )
            continue;

        String annee="";
        String dateDeces="";
        String dateInhumation="";
        String nom="";
        String prenom="";
        String nomPere="";
        String prenomPere="";
        String nomMere="";
        String prenomMere="";
        int age=0;

        // Chargement de la ligne
        Vector tokens=getTokens(ligne);
```

```
            int i=0;
            for(int cpt1=0;cpt1<tokens.size();cpt1++) {
                String token=(String)tokens.elementAt(cpt1);
                token=token.trim();
                if( i==0 )
                    annee=token;
                else if( i==1 )
                    dateDeces=token+"/"+annee;
                else if( i==2 )
                    dateInhumation=token+"/"+annee;
                else if( i==3 )
                    prenom=token.toLowerCase();
                else if( i==4 )
                    nom=token.toLowerCase();
                else if( i==5 ) {
                    nomPere=nom;
                    prenomPere=token.toLowerCase();
                } else if( i==6 ) {
                    prenomMere=token.toLowerCase();
                } else if( i==7 ) {
                    nomMere=token.toLowerCase();
                } else if( i==8 ) {
                    //prenom conjoint
                } else if( i==9 ) {
                    //nom conjoint
                } else if( i==10 ) {
                    int indice=0;
                    if( (indice=token.indexOf(" "))!=-1 )
                    try {
                        age=Integer.parseInt((token.substring(0,indice)).trim());
                    } catch(Exception ex) {}
                    if( token.indexOf("mois")!=-1 )
                        age=0;
                }
                i++;
            }
            //Checker les prenoms
            if( prenoms.get(prenom)==null )
                prenoms.put(prenom,"");
            if( prenoms.get(prenomPere)==null )
                prenoms.put(prenomPere,"");
            if( prenoms.get(prenomMere)==null )
                prenoms.put(prenomMere,"");
        }
        return prenoms;
    }

    public void convert() {
        Hashtable prenoms=new Hashtable();
        prenoms=convertNaissances(prenoms);
        prenoms=convertMariages(prenoms);
        prenoms=convertDeces(prenoms);

        Vector prenomsDistincts=new Vector();
        for(Enumeration e=prenoms.keys();e.hasMoreElements();) {
            prenomsDistincts.addElement(e.nextElement());
        }

        ecrireFichier("prenoms_beignon.txt",prenomsDistincts);
    }

    public static void main(String[] args) {
        GetDistinctsPrenoms cf=new GetDistinctsPrenoms();
        cf.convert();
    }
}
```

Note

Le but de récupérer les prénoms du dépouillements est de constituer le fichier des prénoms féminins. Ceci ne peut être fait que manuellement à partir du fichier généré par cette classe.

Annexe C. Fichier de commandes

Il est possible grâce à la console JGenea d'exécuter à la chaîne une liste de commande définie dans un fichier. Ce fichier s'exécute de la manière suivante:

```
bash$ console -b test1 -f beignon.jg
```

Exemple C.1. Fichier de commandes pour la console (beignon.jg)

```
viderbase
Oui
table -import -villeid 2970 -fichier ..\beignon\beignon-naissances.csv.new
Oui
table -import -villeid 2970 -fichier ..\beignon\beignon-mariages.csv.new
Oui
table -import -villeid 2970 -fichier ..\beignon\beignon-deces.csv.new
Oui
fusion -table -villeid 2970
html -table -annee -villeid 2970 -repertoire ../html
html -personne 2970 -repertoire ../html
```