# Fichiers courants en Java

## Fichiers properties

Il s’agit de fichiers typiquement utilisés pour stocker la configuration d’une application. Il s’agit de fichiers texte et sont de la forme :

key1=val2  
key2=val2  
key3=val3  
…

Pour les gérer on utilise la classe Properties. Il s’agit d’une dérivée de Map.

Pour lire le fichier et récupérer une information, on fait :

Properties p = new Properties() ;  
p.load(InputStream is) ;  
String key1 = p.getProperty(« key1 ») ;  
…  
  
On peut aussi écrire dans ce fichier via la methode store(OutputStream) ;

## Fichiers CSV

Il s’agit de fichiers de données à nombre de colonnes fixes. Typiquement ils sont exportés depuis Excel.

Ils sont donc de la forme :   
val11,val12,val13,val14,…  
val21,val22,val23,val24,…  
….  
  
Le séparateur (ici virgule) peut varier (typiquement point-virgule ou blanc en plus de virgule).  
Ces fichiers peuvent contenir la liste des titres en première ligne.  
  
Ex :  
Titre,Auteur,…  
Les freres Karamazov,Dostoievski,..  
Voyage au bout de la nuit,Celine,…

A noter que les valeurs peuvent parfois être entourées par des guillements (dans les exercices, on laissera cette possibilité de côté).

## Fichiers XML

Ce sont des fichiers présentant des données sous forme arborescente. Ils peuvent être vus comme la représentation texte d’objets.

<?xml version="1.0" encoding="UTF-8"?>

<Contacts>  
 <Person>

<Firstname>John</Firstname>

<Lastname>Smith</Lastname>

<Birthday>1965/03/02</Birthday>

<Company>IBM</Company>

<Position>CEO</Position>

<Email>jsmith@ibm.com</Email>

<Email>jsmith@yahoo.com</Email>

</Person>

<Person>

<Firstname>Tom</Firstname>

<Lastname>Dunne</Lastname>

<Company>Today FM</Company>

<Position/>

<Email>tom.dunne@todayfm.com</Email>

</Person>

</Contacts>

Ici ce fichier XML peut être vu comme la représentation de l’objet Contact qui contient un tableau de 2 Person.

Un fichier XML peut être utilisé comme fichier de configuration (comme un fichier properties), soit comme fichier d’échange de données.

Pour se repérer dans un fichier XML on a la notion de nœud (node). Un nœud est un élément de la forme <xxx/>.

Ex : <Lastname>Smith</Lastname>

On a ici le nœud de **tagname** Lastname.

Ex :

<Person>  
 <Firstname>Tom</Firstname>

<Lastname>Dunne</Lastname>

<Company>Today FM</Company>

<Position/>

<Email>tom.dunne@todayfm.com</Email>

</Person>

Ici, on a le nœud Person qui contient des sous-nœuds (FirstName,LastName,Company,Position,Email) ;

En plus du tagname, on peut aussi ajouter des attributs à un nœud. Ex :

<Person id=»1255»> ….

Ici l’attribut est id.

Etant donné la structure arborescente des fichiers XML, on se doute qu’il est assez complexe de les parcourir et/ou de les analyser. On utilise pour cela ce qu’on appelle des « parser ».

En pratique, Java fournit 2 parseurs :

* Le parseur DOM : Il s’agit d’un parseur qui charge tout le fichier XML en mémoire, le transforme en arbre et on peut le parcourir de manière (relativement) facile. Evidemment, il a comme inconvénient d’avoir une grosse consommation mémoire et parfois, il est inutilisable de ce fait.
* Le parseur SAX : Dans le cas le parseur ne charge (presque) rien en mémoire et appelle le programme utilisateur au fur et à mesure de son parcours du fichier. Il est nettement plus compliqué à utiliser.