Objectifs

Lire et écrire des fichiers textes

Manipuler des chaînes de caractères

INF1120 – Programmation 1

Fichiers textes et chaînes de caractères

Jacques Berger

Prérequis

Boucles

String

Fichiers

On retrouve 2 types de fichiers

Les fichiers binaires
Les fichiers textes

Fichiers

Les fichiers binaires sont des données encodées dans un format uniquement compréhensible par des logiciels

Fichiers

Les fichiers textes contiennent des chaînes de caractères

Ils peuvent être facilement lus par un humain

Le contenu correspond à un ensemble de chaînes de caractères

Fichiers

Un fichier doit d'abord être ouvert

Puis, il est lu

Et finalement, il est fermé

Il est nécessaire de toujours fermer un fichier après l'avoir ouvert

Fichiers

En Java, il existe plusieurs façons de lire un fichier

Une façon simple est d'utiliser la classe Scanner, en spécifiant un fichier en paramètre

```
Scanner fichier = new Scanner(new File("newfile.txt"));
```

Fichiers

On ferme le fichier avec une fonction close

```
fichier.close();
```

Fichiers

Pour lire une ligne de texte dans le fichier, on utilise la fonction nextLine du Scanner

```
String line = fichier.nextLine();
```

Fichiers

Lorsqu'un fichier a été lu au complet, continuer de le lire provoquera une erreur

Une fonction hasNextLine permet de savoir s'il reste des lignes à lire dans le fichier

On combine cette fonction à une boucle while pour lire un fichier au complet

Fichiers

```
Scanner fichier = new Scanner(new File("newfile.txt"));
while (fichier.hasNextLine()) {
    String line = fichier.nextLine();
    // Traiter la ligne de texte
}
fichier.close();
```

Fichiers

Pour écrire dans un fichier, il existe aussi plusieurs façons de faire

L'approche suggérée est d'utiliser un PrintWriter

```
PrintWriter fichier = new PrintWriter(new FileWriter("test.txt"));
```

Fichiers

Le PrintWriter s'utilise comme le System.out

```
fichier.println("Hello World!");
```

Fichiers

Le PrintWriter doit aussi être fermé après son utilisation avec la fonction close

À chaque fois qu'on crée un PrintWriter, le fichier spécifié en paramètre sera créé ou écrasé (s'il existe déjà)

Chaînes

Avec les fichiers textes, il faut s'attendre à faire plusieurs manipulations de chaînes de caractères

Nous avons déjà étudié :
equals
equalsIgnoreCase
String.format

Voici quelques fonctions très utiles

Chaînes

```
char charAt(int index)
```

Retourne le caractère à un indice précis de la chaîne

Comme les tableaux, le premier indice d'une chaîne est 0

```
String chaine = "Bonjour";
char lettre = chaine.charAt(4);
```

Chaînes

```
String concat(String str)
```

Crée une nouvelle chaîne contenant la chaîne en paramètre juxtaposée (concaténée) à la chaîne originale

```
String chaine = "Bon";
String chaineComplete = chaine.concat("jour");
```

Chaînes

```
int indexOf(char caractere)
```

Retourne l'indice de la première occurence du caractère en paramètre

```
String chaine = "Je suis une patate!";
int position = chaine.indexOf(' ');
```

Chaînes

```
int length()
```

Retourne le nombre de caractères dans la chaîne

```
String chaine = "test";
int taille = chaine.length();
```

Chaînes

```
String replace(char old, char new)
```

Crée une nouvelle chaîne à partir de la chaîne originale où le vieux caractère a été remplacé par le nouveau

```
String chaine = "Je suis étudiant";
String nouvelleChaine = chaine.replace('é', 'e');
```

Chaînes

```
String substring(int indiceDebut)
String substring(int indiceDebut, int indiceFin)
```

Extraire une sous-chaîne de la chaîne originale

```
String chaine = "Je suis une patate";
String apresEspace = chaine.substring(chaine.indexOf(' ') + 1);
```

Chaînes

```
String toUpperCase()
String toLowerCase()
```

Crée une nouvelle chaîne à partir de l'originale, mais toute en majuscule ou minuscule

```
String chaine = "SALUT";
String lower = chaine.toLowerCase();
```

Chaînes

```
String trim()
```

Retirer les caractères d'espacement au début et à la fin de la chaîne originale

```
String chaine = "  Bonjour le groupe    ";
String trimmed = chaine.trim();
```

Chaînes

```
String[] split(String expression)
```

Retourne un tableau de chaînes contenant la chaîne originale séparée en plusieurs morceaux selon l'expression en paramètre

```
String chaine = "2016-11-12";
String[] champsDate = chaine.split("-");
```

Plus loin...

Le livre de Java premier langage, 11ème édition
Anne Tasso
Eyrolles, 2016
Chapitres 7 et 10