INF1256 Informatique pour les sciences de la gestion

– *Contrôle d'erreurs, String et fichiers textes* –

Johnny TSHEKE, Ing. Jr.

Université du Québec à Montréal
Département d'Informatique
tsheke_shele.johnny@uqam.ca

Séance 09

Exception en général

Événement Exceptionnel qui perturbe le déroulement du programe. 3 types :

- Checked exception, controlé dans l'application avec `try ...catch`
- Runtime exception, interne au programme mais n'a pas été anticipée (pas de try . . . catch prévu)
- Error, externe au programme et peut-être contrôlée mais généralement on n'y pense pas à le faire

Classes Java associées :

Exception : `java.lang.Exception`

Error : `java.lang.Error`

⇒ Les deux ont pour super-classe `java.lang.Throwable`

`https://docs.oracle.com/javase/tutorial/essential/exceptions/`

`https://docs.oracle.com/javase/8/docs/api/java/lang/Exception.html`

Contrôle des exceptions : try . . . catch

```
try{
//code susceptible de produire une exception
}catch(Exception e){
// code exécuter si exception
}finally{
//finally est facultatif et contient le code à exécuter
// à la fin de l'instruction try, qu'il y ait exception ou pas
}
```

Contrôle des exceptions : try . . . catch avec ressources

Un try avec déclaration d'une ou plusieurs ressources qui doivent-être fermées à la fin (`try-with-resources`)

```
try(Scanner clavier = new Scanner(System.in);){
//variable clavier déclarée à titre d'exemple
//code susceptible de produire exception
}catch(Exception e){
// code en cas exception
}finally{
//finally est facultatif et contient le code à exécuter
// à la fin de l'instruction try, qu'il y ait exception ou pas
}
```

Renvoyer volontairement une erreur ou une exception

On utilise l'instruction `throw` pour renvoyer volontairement un `throwable` (Error ou Exception ou encore une sous-classe de ces classes)

```
throw objetThrowable;
```

Ex :

```
throw new Exception("Message");
```

Exemple Exception

Exemple contrôle erreur ExempleException.java

```
package inf1256s09;

import java.util.Locale;
import java.util.Scanner;

public class ExempleException1 {
    public static void main (String[] args){
        Scanner clavier = new Scanner(System.in);
        clavier.useLocale(Locale.US);
        int nombreEntier=0;
        System.out.println("Saisir une chaine pour provoquer une exception");
        try{ //code a surveiller
            nombreEntier = clavier.nextInt();
            System.out.println("Inverse de "+ nombreEntier +" = "+ 1.0/nombreEntier);
            // ATT: en java la division par 0 retourne Infinity et pas une Exception!
        }catch (Exception e){ // si une exception se produit
            //e.printStackTrace();//peut servir à debogguer
            System.out.println("Veuillez saisir un nombre entier positif svp!");
        }
    }
}
```

Exemple renvoi Exception

Exemple renvoyer volontairement une Exception ExempleException.java

```
package inf1256s09;

import java.util.Locale;
import java.util.Scanner;

public class ExempleException2 {
    public static void main (String[] args){
        Scanner clavier = new Scanner(System.in);
        clavier.useLocale(Locale.US);
        int nombreEntier=0;
        System.out.println("Saisir une chaine ou 0 pour provoquer exception");
        try{ // ATT: en java la division par 0 retourne Infinity et pas une Exception!
            nombreEntier = clavier.nextInt();
            if(nombreEntier == 0){//renvoyer exception
                throw new Exception("On ne peut pas diviser un nombre par 0");
            }
            System.out.println("Inverse de "+ nombreEntier +" = "+ 1.0/nombreEntier);
        }catch (Exception e){
            e.printStackTrace();//peut servir pour debogguer
            System.out.println("Veuillez saisir un nombre entier positif svp!");
        }
    }
}
```

Exemple try . . . catch . . . finally

Exemple try .. catch .. finally ExempleException.java

```
package inf1256s09;

import java.util.Locale;
import java.util.Scanner;

public class ExempleException3 {
    public static void main (String[] args){
        Scanner clavier = new Scanner(System.in);
        clavier.useLocale(Locale.US);
        int nombreEntier=0;
        System.out.println("Saisir une chaine ou 0 pour provoquer exception");
        try{ // ATT: en Java la division par 0 retourne Infinity et pas une Exception!
            nombreEntier = clavier.nextInt();
            if(nombreEntier == 0){//renvoyer exception
                throw new Exception("On ne peut pas diviser un nombre par 0");
            }
            System.out.println("Inverse de "+ nombreEntier +" = "+ 1.0/nombreEntier);
        }catch (Exception e){
            e.printStackTrace();//peut servir pour debogguer
            System.out.println("Veuillez saisir un nombre entier positif svp!");
        } finally {
            System.out.println("\n Erreurs sous controle!");
        }
    }
}
```

Classe String

`import java.lang.String` avec comme Constructeur `String();`
`String chaine="petite italie"` et `pattern` un expression régulière

- chaine.split(pattern) → découpe la chaine selon pattern
  chaine.split(" ") → `['petite','italie']`
- chaine.charAt(index) → retourne caractère à l'indice donné
  chaine.charAt(2) → `t`
- chaine.startsWith(prefix) → true si commence avec prefix
- chaine.isEmpty() → true si chaine vide
- chaine.length() → nombre de caractères
- chaine.trim() → suppression espaces au début et fin
- String.valueOf(Object obj) → chaine représentant obj
- ...

⇒ plus d'info : `https://docs.oracle.com/javase/8/docs/api/java/lang/String.html`

classe String (2)

- chaine.toLowerCase(Locale locale) → minuscule selon locale
- chaine.toLowerCase() → mettre em minuscule
- chaine.toUpperCase() → mettre en majuscule
- chaine.substring(int beginIndex, int endIndex)() → sous chaine
- chaine.toCharArray() → tableau des caractères
- chaine.replace(char oldChar, char newChar) → remplace
- chaine.replaceAll(String regex, String replacement) → remplace tout
- chaine.matches(String regex) → true si match regex (mettre préfixe (?i) pour insensibilité à la casse)
- chaine.contains(CharSequence s) → true si contient s
- chaine.concat(String str) → concatene str à la fin
- chaine.compareTo(String anotherString) → comparaison sensible à la casse.(0 si égales, négatif si argument lexicographicament plus grand, sinon positif)
- chaine.compareToIgnoreCase(String str) → comparaison sans tenir compte de la casse.

Exemple manipulation chaines

Manipulation chaines Chaines.java

```
        public static void main(String[] args) {
                String chaine = "petite italie";
                System.out.println(chaine.charAt(2));
                //index dernière lettre t
                System.out.println(chaine.lastIndexOf("t"));
                //index première lettre i
            System.out.println(chaine.indexOf("i"));
            //index 1er i à partir de l'index 5
            System.out.println(chaine.indexOf("i", 5));
          //index 1er i à partir de l'index 5
            System.out.println(chaine.indexOf("i", 16));//-1 -> existe pas
                //majuscule
                System.out.println(chaine.toUpperCase());
                //capitalisation 1ere lettre
                System.out.println(chaine.substring(0,1).toUpperCase().concat(chaine.↙
    ↳ substring(1)));
            }
}
```

Fichier texte

- Un fichier peut contenir des données de tout type
- Ici on s'intéresse au fichier texte
- Dans un fichier , on peut avoir plusieurs lignes de textes
- Les lignes sont séparées par un caractère spécial de retour à la ligne (*newline*) → `System.lineSeparator()` donne le défaut du système
- `\n` (ou `\r`)pour retour à la ligne (*newline* ou *retour*)
- `\t` pour la tabulation
- Pour manipuler (Lecture, Écriture) un fichier, il faut l'associer à une variable
- Après manipulation, c'est mieux de fermer le fichier pour éviter des pertes de données éventuelles. À partir de Java 8, en utilisant `try-with-resources Statement`, la fermeture se fera automatiquement à la fin de l'instruction `try` mêm si on le ferme pas explicitement

⇒ En Java, un fichier texte peut-être traité comme une séquence de lignes

Manipulation des fichiers textes

Dans un `try-with-resources Statement` pour fermeture automatique de fichier.

import java.io.* ; java.nio.file.* ; Lecture
ouverture

```
//Path path = Paths.get("src/fichier1.txt");
try(BufferedReader reader = Files.newBufferedReader(path)){
//mettre le code de lecture ici
}catch (IOException x){
    //code Exception ici
}
```

Quelques méthodes lecture

```
reader.readLine();// ligne de texte lue ou null fin fichier
reader.read();//un seul caractere
```

Ecriture
ouverture

```
//path est le chemin du fichier texte
try(BufferedWriter writer = Files.newBufferedWriter(path)){
//mettre le code de écriture ici
}catch (IOException x){
    //code Exception ici
}
```

Quelques méthodes écriture

```
writer.newLine();// retour a ligne
writer.write(chaine);// écrire chaine dans le fichier
write(String s, int off, int len);//une chaine, debut et longeur
```

⇒https://docs.oracle.com/javase/8/docs/api/java/nio/file/Files.html
⇒https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html

Exemple de lecture d'un fichier

Fichier texte à lire fichier1.txt

```
ligne1 du fichier
ligne2 du contenu
ligne3 blabla
ligne4 haha
ligne5 termine
```

Lecture d'un fichier texte LireFichier.java

```
public static void main(String[] args) {
        Path fichier = Paths.get("src/fichier1.txt");//dossier projet racine
        try(BufferedReader reader = Files.newBufferedReader(fichier);){
            String ligne=reader.readLine();//lecture 1ere ligne
            while(ligne!=null){
                System.out.println(ligne);
                ligne=reader.readLine();//ligne suivante
            }
              }catch (IOException x){
                 System.out.println("Erreur lecture fichier");
              }
}
```

Exemple d'écriture dans un fichier texte

Fichier texte écrit fichier2.txt

```
Bonjour
Merci
```

Écriture dans un fichier texte EcrireFichier.java

```
    public static void main(String[] args) {
            String separateur = System.lineSeparator();
            Scanner clavier = new Scanner(System.in).useDelimiter(separateur);
            Path fichier = Paths.get("src/fichier2.txt");//racine: dossier projet
            try(BufferedWriter writer = Files.newBufferedWriter(fichier);){
                while(true){
                    System.out.println("entrez une chaine ou appuyez sur espace ↙
↳ puis entree pour terminer");
                    String ligne=clavier.next().trim();//lecture 1ere ligne
                    if(ligne.isEmpty()){
                            break;
                    }
                    writer.write(ligne);
                    writer.newLine();
                }
                }catch (IOException x){
                   System.out.println("Erreur lecture fichier");
                }
    }
```