Programmation Orienté Objets en JAVA

Daniele Varacca
Departement d'Informatique
Université Paris Est

2014

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les exceptions

En certains langages, les erreurs à l'exécution peuvent être sérieuses!

Les exceptions

En Java, grâce à la JVM, on peut mieux gérer les erreurs et diagnostiquer les problèmes

Quand un problème se produit à l'exécution, le programme *lance une exception*.

Une exception

- est un objet d'une certaine classe
- contient plusieurs informations sur les raisons qui ont amené la programme à la lancer

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les exceptions

Exemple `Cours8.java`

- ArrayIndexOfBoundsException
- NullPointerException

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les exceptions

D'autres classes d'exceptions

- NegativeArraySizeException
- StringIndexOutOfBoundsExcpetion
- IllegalArgumentException
- ArithmeticException
- NumberFormatException
- IOException
- InterruptedExcpetion

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les RuntimeException

Les erreurs qui sont affichées pendant l'exécution sont du type `java.lang.RuntimeException`

Si une de ces exceptions est lancée, le programme s'arrête et l'information contenue dans l'exception est affichée en console

Lancer une exception

On peut décider de programmer le lancement d'une exception.

Cela peut se faire partout dans un programme

- dans le main
- dans un constructeur
- dans une méthode

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Lancer une exception

Pour lancer une exception il faut

- avoir un objet de type RuntimeException
- le lancer avec le mot clé throw

```
RuntimeException rte = new RuntimeException();
if (...//quelque choses de méchant
  throw rte;
```

Lancer une exception

Souvent on crée l'exception à la volée, sans la stocker dans une variable

```
throw new RuntimeException();
```

Meilleure pratique: créer une sousclasse

```
class UpecException extends RuntimeException {}
...
  throw  new UpecException();
```

Lancer une exception

Les constructeurs des exception prennent souvent un String pour un message

```
throw new RuntimeException("My Taylor is Rich");
```

Attention, si on crée une sous-classe, il faut définir le constructeur!

Ceci ne compile pas:

```
class UpecException extends RuntimeException {}
...
  throw  new UpecException("My Taylor is Rich");
```

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Créer une classe exception

```
class UpecException extends RuntimeException {
    public UpecException(String message) {
     super(message);
    }
}
...
  throw  new UpecException("My Taylor is Rich");
```

Créer ses propres exceptions permet de mieux comprendre ce qui s'est passé quand l'exception a été lancée

Attraper une exception

Les exceptions peuvent être *attrapées*!

Quand une exception qui a été lancée est attrapée

- le programme ne s'arrête pas
- du code *exceptionnel* peut être executé

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Attraper une exception

Si on pense qu'un morceau de code puisse lance une exception on peut l'attraper en l'entourant par une clause Try-Catch

```
try {
    //code qui peut lancer une NullPointerException
   }
catch (NullPointerException e) {
   //ici on écrit le code exceptionnel
  }
```

Exemple

Attraper une exception

On peut attraper plusieurs exceptions à la fois

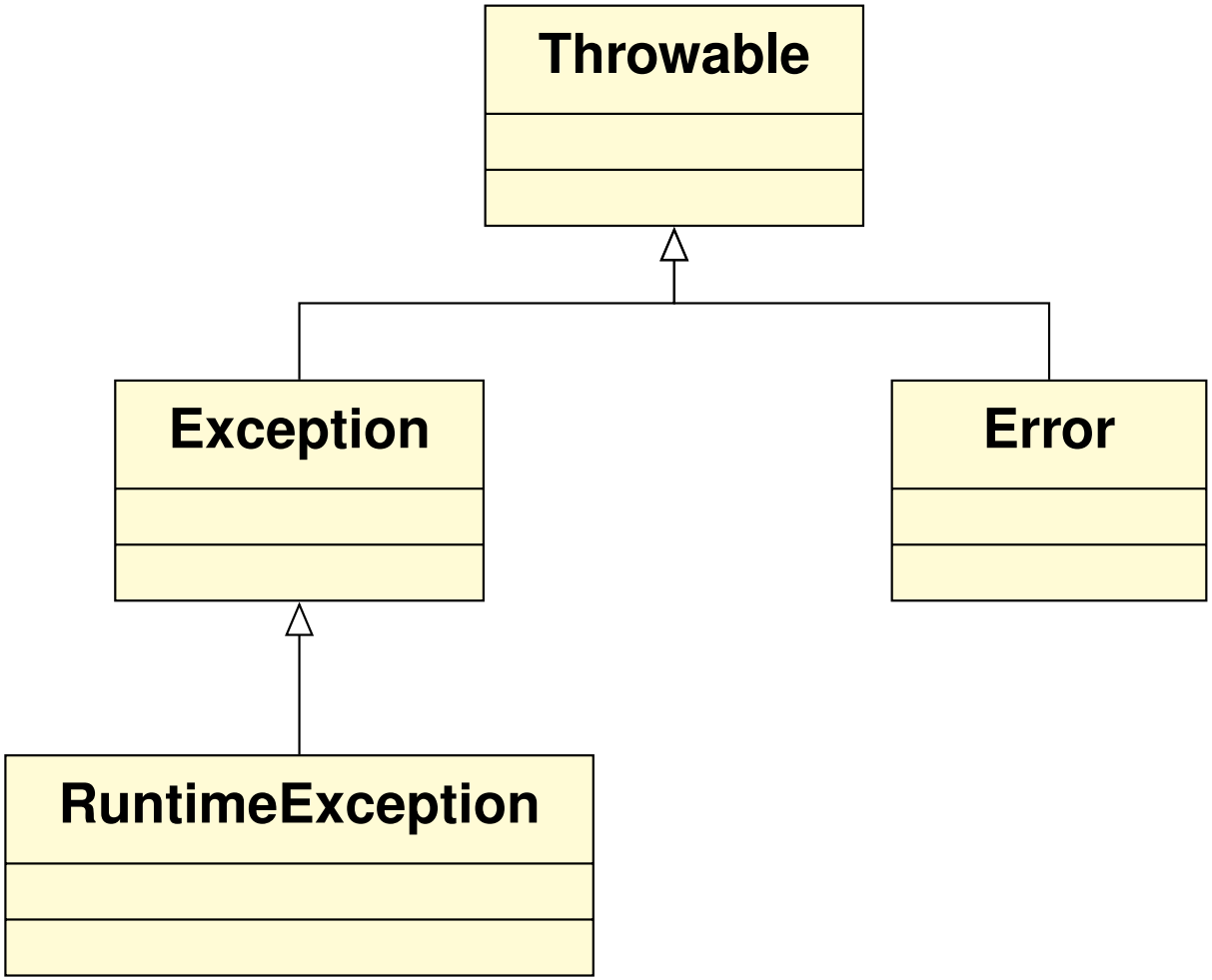
```
try {
    //code qui peut lancer l'exception
  }
catch ( NullPointerException
      | ArrayIndexOutOfBoundsException
      | UpecException e)
  {
   //ici on écrit le code exceptionnel
  }
```

Attraper une exception

On peut attraper plusieurs exceptions, et exécuter du code exceptionnel différent

```
try {
     //code qui peut lancer l'exception
    }
catch (NullPointerException e)
    {
    //ici on écrit pour cette exception
   }
catch (ArrayIndexOutOfBoundsException e)
    {
    //ici on écrit d'autre code
   }
```

La hiérarchie des exceptions



Les exceptions contrôlées

Les Exceptions qui ne sont pas du type RuntimeException, s'appellent *contrôlées* et doivent être *déclarées*.

Toute méthode qui contient du code qui peux lancer une exception contrôlées doit déclarer qu'elle peut la lancer avec le mot clé throws (avec s attention)

```
class Paris12Exception extends Exception{}

public void f () throws Paris12Exception {
...
if (machin-truc) throw new Paris12Exception();

...
}
```

Les exceptions contrôlées

```
class Paris12Exception extends Exception{}

public void f () throws Paris12Exception {
...
if (machin-truc) throw new Paris12Exception();

...
}
```

Toute méthode qui utilise f, doit

- soit attraper l'exception Paris12Exception avec un try catch.
- soit elle aussi déclarer throws Paris12Exception
- (même le main!)

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Compléments sur les exceptions

- une sous-exception est attrapée par la superclasse la plus proche
- on peut faire quelque chose après dans tous les cas avec finally

Les entrées/sorties

packages

`java.nio.file`

`java.io`

Le problème du rapport avec le système:

- les fichiers, l'écran, le clavier existent en dehors de la JVM
- ils sont gérés par le système d'exploitation
- la JVM doit communiquer avec celui ci

Les Flux

Les classes

- OutputStream
- InputStream

La classe PrintStream est une sous-classe de OutputStream

- System.in est un objet de type InputStream qui représente le clavier
- System.out est un objet de type PrintStream qui représente la console

Les Flux

Pour pouvoir communiquer avec le clavier, il faut construire un gestionnaire de Flux, typiquement un objet de la classe java.util.Scanner

```
Scanner sc = new Scanner(System.in);
sc.nextInt()
sc.nextLine()
...
```

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les fichiers

Pour gerer les fichier on utilise les classes utilitaires

- java.nio.file.Files
- java.nio.file.Paths

Elles contiennent des méthodes statiques pour la gestion des fichiers (création, déplacement)

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les fichiers

Plusieurs façon de traiter un fichier

- une suite de bits
- une suite d'octets
- une suite de caractères
- ... encore plus de structure...

Pour gérer les fichier texte on utilise les classes BufferedReader et BufferedWriter

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

Les fichiers

```
Path file = Paths.get("toto.txt");
BufferedReader reader =
    Files.newBufferedReader(file,StandardCharsets.UTF_8);
String line;
while (true) {
    line = reader.readLine();
    if (line=null) break;
    System.out.println(line);
}
```

Les fichiers

```
Path file = Paths.get("toto.txt");
BufferedWriter writer =
    Files.newBufferedWriter(file,StandardCharsets.UTF_8);
for (int i=0; i<10; i++)
        writer.write("This␣is␣line␣number␣" + i + "\n");
writer.flush();
```

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL VAL DE MARNE
Connaissance - Action

On peut aussi utiliser PrintWriter

```
BufferedWriter w =
    Files.newBufferedWriter(file,StandardCharsets.UTF_8);
PrintWriter writer = new PrintWriter(w);
for (int i=0; i<10; i++)
    writer.println("This is line number " + i);
    writer.flush();
    }
```