Cours 6 JAVA

Mlle AMEUR .K
E-Mail: ameur.khadidja@univ-ouargla.dz

Niveau 1

- **Sessions 1:** Les bases du langage Java
  - Introduction
  - Données
  - Structures de contrôle
  - Tableaux
  - **Exceptions**

C'est quoi une exception?

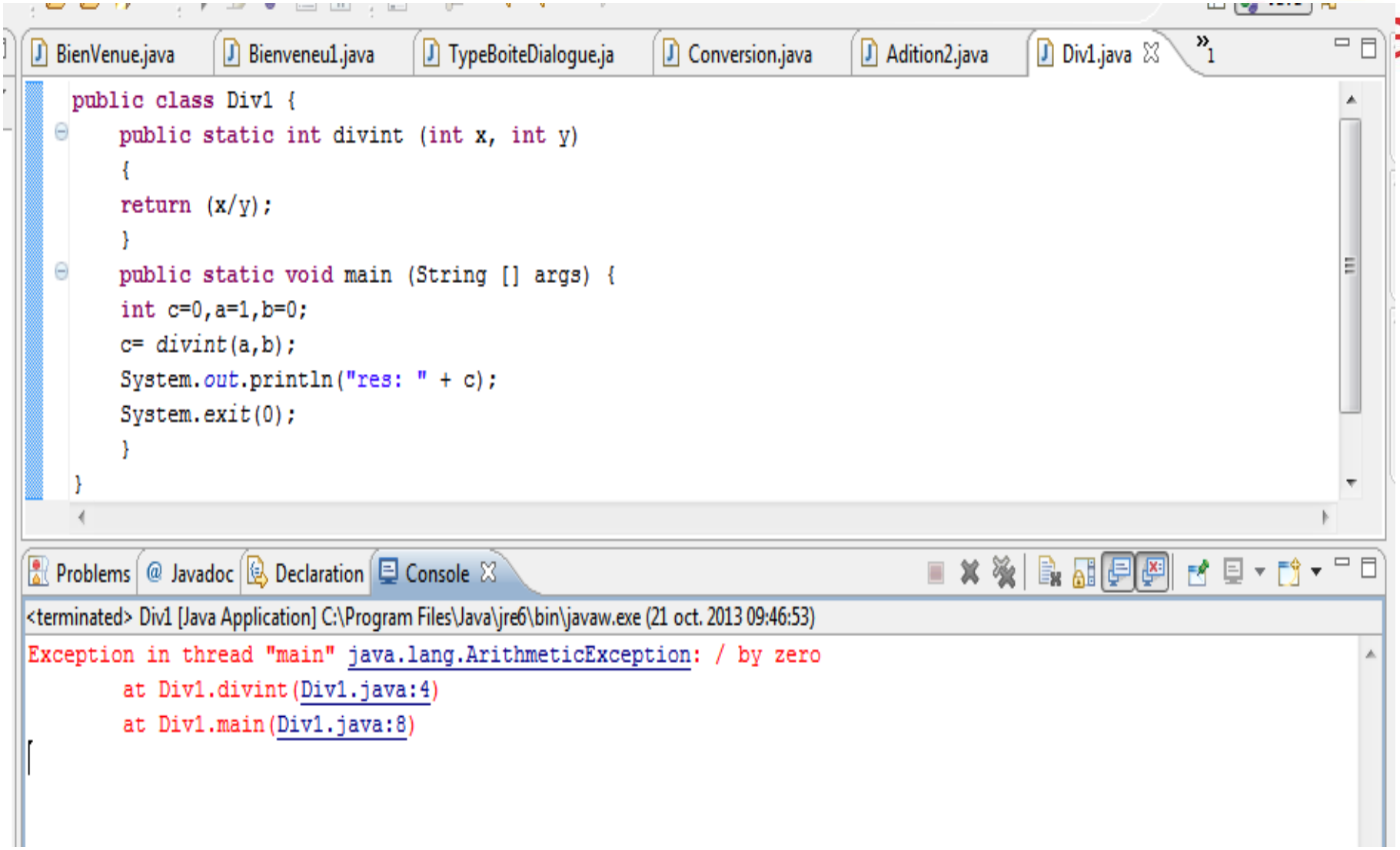
- un événement (une erreur) qui se produit lors de l'exécution d'un programme, et qui va provoquer un fonctionnement anormal de ce dernier.

Exemple

- La simple ouverture d'un fichier peut provoquer beaucoup d'erreurs:
- l'inexistence du fichier,
- un mauvais format,
- une interdiction d'accès,
- une erreur de connexion au périphérique,
- …

➔ Pour que notre programme soit **robuste**, il faut que toutes les erreurs possibles soient détectées et traitées.

Exemple


BienVenue.java
Bienveneu1.java
TypeBoiteDialogue.ja
Conversion.java
Adition2.java
Div1.java
1
public class Div1 {
public static int divint (int x, int y)
{
return (x/y);
}
public static void main (String [] args) {
int c=0,a=1,b=0;
c= divint(a,b);
System.out.println("res: " + c);
System.exit(0);
}
}
Problems
Javadoc
Declaration
Console
<terminated> Div1 [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (21 oct. 2013 09:46:53)
Exception in thread "main" java.lang.ArithmeticException: / by zero
at Div1.divint(Div1.java:4)
at Div1.main(Div1.java:8)

Gestion des exceptions

- La gestion des exceptions se substitue en quelque sorte à l'algorithmique permettant la gestion des erreurs.
- Une exception est un objet qui peut être émis par une méthode si un événement d’ordre “exceptionnel” (les erreurs rentrent dans cette catégorie) se produit.

Gestion des exceptions
Exemple


```
if (y!=0)
return(x/y);
else
// traitement de l'erreur.
```

Gestion des exceptions
Exemple


```
public class Div1
{
public static int divint (int x, int y)
{
    return (x/y);
}
public static void main (String [] args)
{
    int c=0,a=1,b=0;
    c= divint(a,b);
    System.out.println("res: " + c);                System.exit(0);
}
}
```

Gestion des exceptions

La propagation d'une émission se déroule

selon les étapes suivantes :

1. Une exception est générée à l'intérieur d'une méthode ;
2. Si la méthode prévoit un traitement de cette exception, on va au point 4, sinon au point 3 ;
3. L'exception est renvoyée à la méthode ayant appelé la méthode courante, on retourne au point 2 ;
4. L'exception est traitée et le programme reprend son cours après le traitement de l'exception.

Exemple

La gestion d'erreurs par propagation d'exception présente deux atouts majeurs :

- Une facilité de programmation et de lisibilité : il est possible de regrouper la gestion d'erreurs

- Une gestion des erreurs propre et explicite : certains langages de programmation utilisent

Déclaration

- **public static int parseInt(String s) throws NumberFormatException { …}**

Interception et traitement

```
public class ExempleTraitementException {
public static void main(String[] args) {
System.out.print("Entrez le numero d'un mois : ");
try {
BufferedReader input = new BufferedReader(
new InputStreamReader(System.in));
String choix = input.readLine();
int numero = Integer.parseInt(choix);
System.out.println(ExempleException.month(numero));
} catch (IndexOutOfBoundsException e) {
System.err.println("Numero incorrect : "
+ e.getMessage());
} catch (NumberFormatException e) { System.err.println("Entrée incorrecte : " +
    e.getMessage());
} catch (IOException e) { System.err.println("Erreur d'accès : " + e.getMessage()); }
}
}
```

Classes d’exception

- **AWTException**
- **ClassCastException**
- **FileNotFoundException**
- **IndexOutOfBoundsException**
- **IOException**
- **NullPointerException**

Classification des erreurs

- **Erreurs de compilation**
- **Erreurs d'exécution.**
- **Exception non vérifiée**
- **Exception vérifiée.**

Exemple

- **Remarque : Il n'est pas indispensable d'intercepter les exceptions héritant de la** classe RuntimeException (dans le tableau ci-dessus, les classes qui en héritent sont
- ClassCastException, IndexOutOfBoundsException et NullPointerException). Celles-ci
- peuvent être propagées jusqu'à la machine virtuelle et n'apparaître que pendant l'exécution. Ces
- exceptions sont appelées exception non vérifiées ou unchecked exceptions.
- Si aucune des classes d'exception ne correspond à un type d'erreur que vous souhaitez exprimer,
- vous pouvez également écrire vos propres classes d'exception. Pour cela, il suffit de faire hériter votre classe de la classe java.lang.Exception.

Références

- Mohamed N. Lokbani, Aspects avancés en Java: Les exceptions en java