JAVA - Eccezioni

Metodi Avanzati di Programmazione

Laurea Triennale in Informatica

Università degli Studi di Bari Aldo Moro

Docente: Pierpaolo Basile

Eccezione

- Un’eccezione è un evento che si verifica durante l’esecuzione del programma
- Identifica un’anomalia che impedisce il normale flusso del programma
- Durante l’esecuzione un metodo può generare un’eccezione
  - viene creato un oggetto eccezione con tutte le informazioni su ciò che è accaduto

Gestione delle eccezioni

- Le eccezioni possono essere catturate e gestite
- Quando un'eccezione viene creata JAVA verifica che ci sia un metodo in grado di gestirla
  - scorrendo a ritroso tutte le chiamate al metodo
- Se viene trovato un gestore per l'eccezione questa viene catturata e gestita
  - un gestore di eccezioni può gestire solo un determinato tipo di eccezione

Tipologie di eccezione

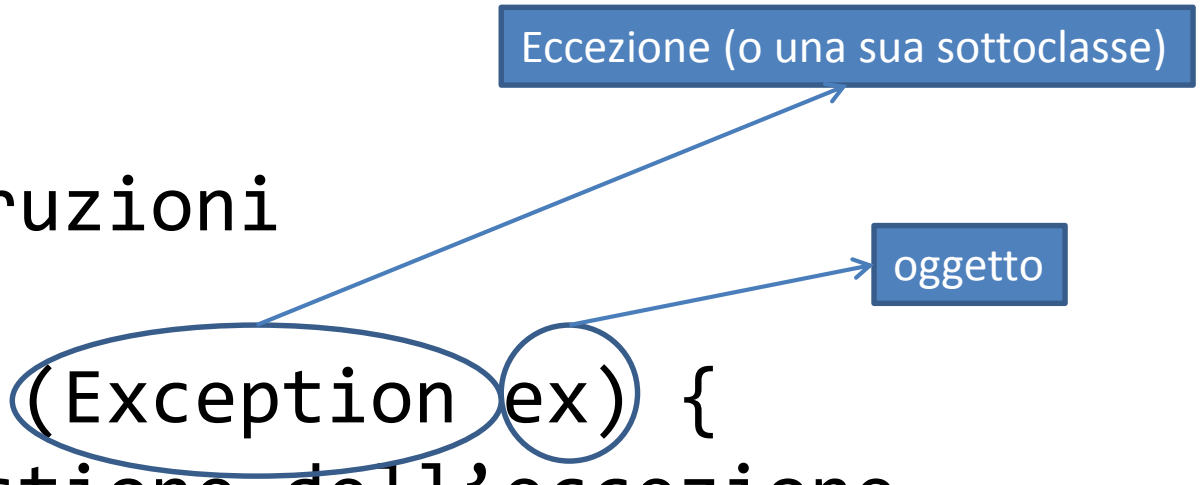
- **Exception** (e sottoclassi): sono eccezioni che possono essere gestite e catturate
  - il programmatore dovrebbe anticipare e gestire questo tipo di eccezioni
- **Error** (e sottoclassi): sono eccezioni che non possono essere gestite e dipendono da qualcosa di esterno al programma
- **RuntimeException** (e sottoclassi): eccezioni interne al programma che non possono essere anticipate o catturate (essenzialmente sono dei bug)

try…catch and finally

```
try {

   \\istruzioni

} catch (Exception ex) {
    \\gestione dell'eccezione
} finally {
    \\istruzioni che devono essere in
ogni caso eseguite
}
```



catch di più eccezioni

```
try {

} catch (IndexOutOfBoundsException e) {
      System.err.println("IndexOutOfBoundsExcept
ion: " + e.getMessage());
} catch (IOException e) {
    System.err.println("Caught IOException: " +
e.getMessage());
}
```

catch di più eccezioni (JAVA SE 7+)

```
try {

} catch (IOException|SQLException ex) {
    System.err.println("Caught Exception: " +
ex.getMessage());
}
```

Blocco finally

- Il blocco **finally** viene eseguito **in ogni caso** dopo il blocco try
  - **sia in caso di eccezione che non**
- Il blocco catch deve essere utilizzato per gestire l'eccezione
- Il blocco **finally** può essere utilizzato per effettuare delle operazioni di ''pulizia''

Lancio di un eccezione…

- I metodi possono dichiarare il tipo di eccezione che si potrebbe verificare al suo interno
  - I metodi **chiamanti** devono gestire l'eccezione, devono includere in un blocco try…catch la chiamata al metodo

```
public File openFile(String filename) throws
IOException {
      //istruzioni
}
```

…lancio di un eccezione

- I metodi possono generare un'eccezione

```
public File openFile(String filename) throws
IOException {
  try {
      //istruzioni
  } catch (IOException ex) {
     throw ex;
  }
}
```

Creazione di una classe eccezione

- Tutte le eccezioni sono sotto classi di Exception
- JAVA mette a disposizione una serie di classi per gestire svariati tipi di eccezione
- E' possibile creare una propria classe Exception
  - estendendo Exception o una delle sue sotto classi

Esempio 1

```
public class NumberException {

    public static void main(String[] args) {
        try {
            double x = Double.parseDouble(args[0]);
            System.out.println("Il quadrato di " + args[0] + " è
" + Math.pow(x, 2));
        } catch (NumberFormatException ex) {
            System.err.println("Il parametro " + args[0] + " non
è un numero.");
        } finally {
            System.out.println("Arrivederci!");
        }
    }

}
```

Esempio 2…

```
public class EmailException extends Exception {
    public String getMessage() {
         return "L'indirizzo email non è
valido";
     }
}
```

…Esempio 2…

```
public class MailParser {

    public static final String MAIL_REGEXP = "....";

    public static void checkMail(String mailAddress)
throws EmailException {

        boolean check = mailAddress.matches(MAIL_REGEXP);
        if (!check) {
            throw new EmailException();
        }
    }

}
```

…Esempio 2

```
public class TestException {
    public static void main(String[] args) {
        try {
            MailParser.checkMail("pippo.mail@mail.com");
             System.out.println("Indirizzo mail valido");
        } catch (EmailException ex) {
            System.err.println("Errore nel controllo
dell'indirizzo: " + ex.getMessage());
        }
    }
}
```

THE END