Objektovo orientované programovanie

Učiteľ:
Ing. Jozef Wagner PhD.

Učebnica:
https://oop.wagjo.com/

OPG
Teória 5

1. Správa chýb pomocou výnimiek

Výnimka - Exception

Výnimka je objekt, ktorý reprezentuje chybový alebo nečakaný stav počas behu programu

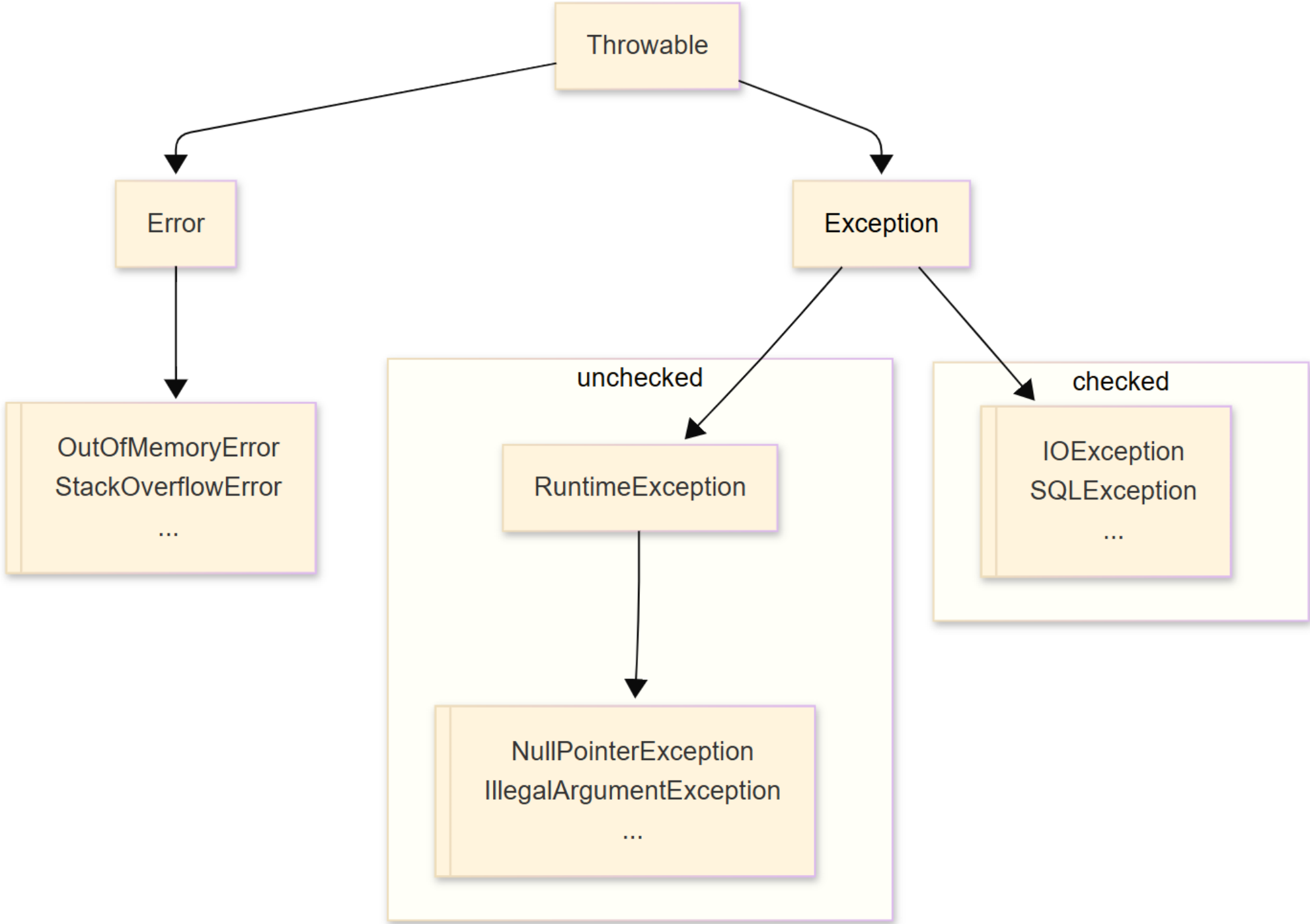
Ak nastane chyba, program 'vyhodí' výnimku

Vyhodená výnimka sa musí ošetriť v metóde, kde nastala, alebo v nadradených metódach

Ak sa neošetrí ani v `main()`, tak program skončí chybou

Kategórie výnimiek

1. **Errors** - Veľmi vážne chyby programu, neošetrujeme, program by sa mal ukončiť
2. **Unchecked Exceptions** - Bežné chyby pri programovaní
3. **Checked Exceptions** - Vážnejšie chyby, ktoré musíme ošetriť, ináč sa program ani neskompiluje a nespustí


Throwable
Error
Exception
OutOfMemoryError
StackOverflowError
...
unchecked
RuntimeException
NullPointerException
IllegalArgumentException
...
checked
IOException
SQLException
...

Zachytenie výnimiek

Pomocou try-catch blokov

```
try {
    // kód, ktorý môže vyhodiť výnimku
} catch (TypVynimky e) {
    // spracovanie výnimky
} finally {
    // vždy sa vykoná, aj keď došlo k výnimke
}
```

Nepovinný blok `finally` sa vykoná vždy, či nastala chyba alebo nie

Zachytenie výnimiek

```
try {
    int a = 10;
    int b = 0;
    int result = a / b; // toto spôsobí ArithmeticException
    System.out.println("Výsledok: " + result);
} catch (ArithmeticException e) {
    System.out.println("Chyba: nemožno deliť nulou!");
    System.out.println("Detail chyby: " + e.getMessage());
}
```

Zachytenie výnimiek

Neošetrené **checked výnimky** musíme uviesť v deklarácii metódy pomocou `throws`

```
public void readFile(String filePath) throws IOException {
    BufferedReader reader =
        new BufferedReader(new FileReader(filePath));
    String line = reader.readLine();
}
```

Je dovolené zadeklarovať aj unchecked výnimky, ale nie je to nutné.

Vyhadzovanie výnimiek

Vyhadzovanie výnimiek robíme pomocou príkazu `throw`

Pri vytváraní výnimky je vhodné uviesť krátku správu, aká chyba nastala

```
public double divide(double numerator, double denominator) {
    if (denominator == 0) {
        throw new IllegalArgumentException("Deliteľ je 0");
    }
    return numerator / denominator;
}
```

Automatické zatvorenie zdrojov

try-with-resources automatické zatvára zdroje

Zdroje, ktoré chcem zatvoriť, dám do zátvoriek hneď za príkazom `try`

Je to lepšie ako použiť blok `finally`

```
try (BufferedReader reader = new BufferedReader(fileReader)) {
    String firstLine = reader.readLine();
    System.out.println("Prvý riadok súboru: " + firstLine);
} catch (IOException e) {
    System.out.println("Chyba pri čítaní: " + e.getMessage());
}
```