Обработка на грешки чрез изключения

Изключение

- **обект, който дефинира необичайна или предизвикваща грешки ситуация;**
- **обектно-ориентирано решение за обработка на грешки;**
- **хвърля се от програмата или от изпълнителната система;**
- **механизъм, даващ информация за грешки по време на изпълнение;**
- **може да бъде хванато и обработено;**
- **концепция за хващане и обработка на грешки в коректния контекст.**

Грешка

- **подобна на изключение, но обикновено представя непоправима ситуация;**
- **не може да се хване.**

1. Хвърляне на изключение

При възникване на изключителна ситуация методът съобщава на извикващия го метод чрез:

throw <израз>;

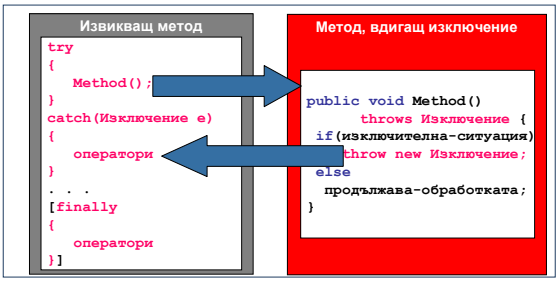
<израз> – **от клас** Throwable

2. Хващане на изключение

```
try
  <блок try>
catch (<декларация на обект на изключение>)
  <блок catch>
...
finally
  <блок finally>
```

<декларация на обект на изключение> – **от клас** Throwable

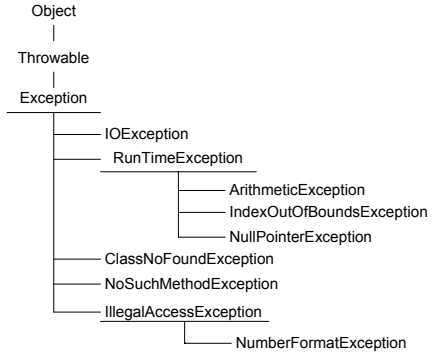
Изпълнява се <блок try>**. Ако той хвърли изключение, контролът се предава на подходящата** catch **клауза и изключението се обработва в** <блок catch>**. Клаузата** finally **не е задължителна и винаги се изпълнява** <блок finally>**.**



3. Опростяване и стандартизиране на обработката на грешки с изключения

- **кодът е по-чист – показва реалната работа, която трябва да се извърши и обработените грешки;**
- **кодът не изисква проверка за върнатите стойности от всеки метод извикан метод и транслиране на кодовете за грешки – използва външен блок, който хваща изключенията от всички извикани методи;**
- **блокът** finally **позволява освобождаване на ресурсите дори при възникване на изключение – не се използват оператори** goto **и други трикове.**

4. Йерархия на класовете за изключения



Клас Exception **(**java.lang**) – сигнализира, че може да се предизвика изключение.**

public Exception(String s)

Конструира Exception **със съобщение** s**.**

public String getMessage()

Връща съобщение за грешка на обекта от тип Throwable**.**

public final Class getClass()

Връща класа на обекта по време на изпълнение.

Клас IOException **(**java.io**) – сигнализира, че може да възникне В/И изключение; то трябва да се обработи от оператор** try **или трябва да се включи в списъка на клаузата** throws **на метод, който може да го разпространи.**

Обработка на изключения

1. **Изключението не се обработва**
2. **Изключението се обработва в мястото на възникване**
3. **Изключението се обработва на друго мястото в програмата**

Пример: Преобразуване на скорост от мили/час в километри/час.

```
import java.io.*;
public class Velocity {
  private static final float MILES_INTO_KILOMETERS = 1.60934f;

  public static void main(String[] args) throws Exception {
  BufferedReader stdin;
  stdin = new BufferedReader(new InputStreamReader(System.in));
  System.out.print ("Въведете скоростта на самолета в мили/час: ");

  float velocity_mph = Float.parseFloat(stdin.readLine());

  float velocity_kmph = MILES_INTO_KILOMETERS * velocity_mph;
  System.out.println("Скоростта на самолета = " + velocity_kmph + "[км/ч]");
  }
}
```

Изключението не се обработва

Хвърля NumberFormatException, ако низът не може да се преобразува във float

Хвърля IOException, ако възникне В/И грешка

Пример: Преобразуване на скорост от мили/час в километри/час.

```
import java.io.*;
public class Velocity {
  private static final float MILES_INTO_KILOMETERS = 1.60934f;
  public static void main(String[] args) {
    BufferedReader stdin;
    stdin = new BufferedReader(new InputStreamReader(System.in));
    System.out.print ("Въведете скоростта на самолета в мили/час: ");
    try {
      float velocity_mph = Float.parseFloat(stdin.readLine());
      float velocity_kmph = MILES_INTO_KILOMETERS * velocity_mph;
      System.out.println("Скоростта на самолета = " +velocity_kmph+"[км/ч]");
    }
    catch (NumberFormatException e) {
      System.out.println("Грешка при въвеждане на скоростта!");
    }
    catch (IOException ioe) {
      System.out.println("В/И грешка при въвеждане на скоростта!");
    }
  }
}
```

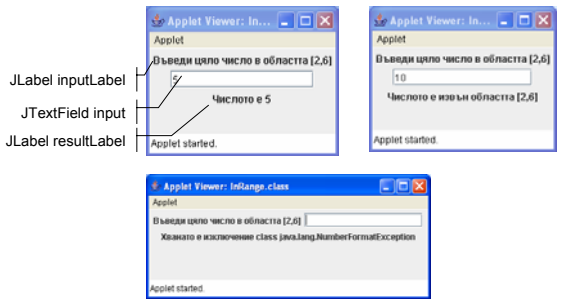
Изключенията се обработват

Резултати:

Въведете скоростта на самолета в мили/час: 800
Скоростта на самолета = 1287.4719[км/ч]

Въведете скоростта на самолета в мили/час: abc
Грешка при въвеждане на скоростта!

Пример: Дефинира собствено изключение OutOfRangeException **(дадена стойност е извън зададена област) – чрез наследяване на класа** Exception



```
// Представя условие за изключение, при което дадена стойност е извън
// зададена област.
public class OutOfRangeException extends Exception {
  public OutOfRangeException(String msg) {
    // msg е съобщението, което описва изключението
    super(msg);
  }
}
```

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class InRange extends JApplet implements ActionListener {
  private int number;
  private static final int MIN = 2;
  private static final int MAX = 6;
  private JLabel inputLabel;
  private JTextField input;
  private JLabel resultLabel;
  public void init() {
    setSize (220,100);
    Container container = getContentPane();
    container.setLayout(new FlowLayout());
    inputLabel = new JLabel("Въведи цяло число в областта [" + MIN +
                            "," + MAX + "]");
    container.add(inputLabel);
    input = new JTextField(15);
    container.add(input);
```

```
    resultLabel = new JLabel();
    container.add(resultLabel);
    input.addActionListener(this);
  }
  public int getNumber() throws Exception {
    int n = 0;
    try {
      n = Integer.parseInt(input.getText() );
    }
    catch(Exception e) {
      throw e;
    }
    if (n < MIN || n > MAX)
      throw new OutOfRangeException("Числото е извън областта [" + MIN +
                                    "," + MAX + "]");
    return n;
  }
```

Хвърля NumberFormatException, ако низът не може да се преобразува в int

Хвърля NullPointerException, ако съдържанието на текстовото поле е празно

Повторно генерира вече хванато изключение

Хвърля изключението OutOfRangeException, ако въведената стойност е извън областта

```
  public void actionPerformed(ActionEvent e) {
    try {
      number = getNumber();
      resultLabel.setText("Числото е " + number);
    }
    catch (OutOfRangeException ore) {
      resultLabel.setText(ore.getMessage());
    }
    catch ( Exception ex ) {
      resultLabel.setText("Хванато е изключение " + ex.getClass());
    }
  }
}
```

Супер класът Exception се обработва последен, за да са достъпни другите блокове catch