Java Výjimky (Exceptions) - Kompletní přehled

🚨 Co jsou výjimky?

Výjimka (Exception) je událost, která nastane během běhu programu a naruší normální tok vykonávání příkazů. Je to způsob, jak Java zpracovává chyby a neočekávané situace.

📊 Hierarchie výjimek

```
Throwable
├── Error (systemové chyby - neřešíme)
└── Exception
    ├── RuntimeException (unchecked)
    │   ├── NullPointerException
    │   ├── ArrayIndexOutOfBoundsException
    │   ├── IllegalArgumentException
    │   └── ArithmeticException
    └── Checked Exceptions
        ├── IOException
        ├── SQLException
        └── ClassNotFoundException
```

🏷️ Typy výjimek

1. Checked Exceptions (Kontrolované)

- **Musí být** ošetřeny v kódu nebo deklarovány v `throws`
- Kontroluje je **kompilátor**
- Typicky I/O operace, síťová komunikace

2. Unchecked Exceptions (Runtime Exceptions)

- **Nemusí být** ošetřeny
- Kontroluje je **běhové prostředí**
- Typicky programátorské chyby

3. Errors

- **Závažné systemové chyby**
- Neošetřují se (OutOfMemoryError, StackOverflowError)

🔧 Ošetření výjimek - try-catch

Základní syntaxe

java

```java
try {
    // Kód, který může vyhodit výjimku
    rizikovyKod();
} catch (TypVyjimky e) {
    // Ošetření konkrétní výjimky
    System.out.println("Nastala chyba: " + e.getMessage());
} finally {
    // Kód, který se vykoná VŽDY (volitelné)
    uklid();
}
```

Více catch bloků

java

```java
try {
    rizikovyKod();
} catch (NullPointerException e) {
    System.out.println("Null pointer: " + e.getMessage());
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Špatný index: " + e.getMessage());
} catch (Exception e) {
    // Obecná výjimka - musí být POSLEDNÍ
    System.out.println("Obecná chyba: " + e.getMessage());
}
```

Multi-catch (Java 7+)

java

```java
try {
    rizikovyKod();
} catch (NullPointerException | IllegalArgumentException e) {
    System.out.println("Jedna z výjimek: " + e.getMessage());
}
```

🎯 Nejčastější Runtime Exception

1. NullPointerException

Kdy nastává: Volání metody na null objektu

java

```java
String text = null;
int delka = text.length(); // ❌ NullPointerException

// Řešení:
if (text != null) {
    int delka = text.length(); // ✅ OK
}
```

2. ArrayIndexOutOfBoundsException

Kdy nastává: Přístup k neexistujícímu indexu pole

java

```java
int[] cisla = {1, 2, 3};
int hodnota = cisla[5]; // ❌ Index 5 neexistuje

// Řešení:
if (index >= 0 && index < cisla.length) {
    int hodnota = cisla[index]; // ✅ OK
}
```

3. IllegalArgumentException

Kdy nastává: Neplatný argument metody

java

```java
public void nastavVek(int vek) {
    if (vek < 0) {
        throw new IllegalArgumentException("Věk nemůže být záporný");
    }
    this.vek = vek;
}
```

4. ArithmeticException

Kdy nastává: Neplatná matematická operace

java

```java
int vysledek = 10 / 0; // ❌ Dělení nulou

// Řešení:
if (delitel != 0) {
    int vysledek = dividend / delitel; // ✅ OK
}
```

5. ClassCastException

Kdy nastává: Neplatné přetypování

java

```java
Object obj = "Hello";
Integer cislo = (Integer) obj; // ❌ String nelze přetypovat na Integer

// Řešení:
if (obj instanceof Integer) {
    Integer cislo = (Integer) obj; // ✅ OK
}
```

📤 Vyhazování výjimek - throw/throws

Vyhození výjimky - throw

java

```java
public void zkontrolujVek(int vek) {
    if (vek < 0) {
        throw new IllegalArgumentException("Věk nesmí být záporný");
    }
    if (vek > 150) {
        throw new IllegalArgumentException("Věk je příliš vysoký");
    }
}
```

Deklarace výjimek - throws

java

```java
public void cteSoubor(String nazev) throws IOException {
    FileReader reader = new FileReader(nazev); // může vyhodit IOException
    // ... čtení souboru
}

// Při volání musíme ošetřit:
try {
    cteSoubor("data.txt");
} catch (IOException e) {
    System.out.println("Chyba při čtení: " + e.getMessage());
}
```

🛠️ Vlastní výjimky

Vytvoření vlastní výjimky

java

```java
// Checked exception
public class NeznamyUzivatelException extends Exception {
    public NeznamyUzivatelException(String zprava) {
        super(zprava);
    }
}

// Unchecked exception
public class NeplatneDataException extends RuntimeException {
    public NeplatneDataException(String zprava) {
        super(zprava);
    }
}
```

Použití vlastní výjimky

java

```java
public class UzivatelService {
    public Uzivatel najdiUzivatele(String jmeno) throws NeznamyUzivatelException {
        Uzivatel uzivatel = databaze.najdi(jmeno);
        if (uzivatel == null) {
            throw new NeznamyUzivatelException("Uživatel " + jmeno + " nebyl nalezen")
        }
        return uzivatel;
    }
}

// Použití:
try {
    Uzivatel uzivatel = service.najdiUzivatele("Jan");
} catch (NeznamyUzivatelException e) {
    System.out.println("Chyba: " + e.getMessage());
}
```

💡 Try-with-resources (Java 7+)

Automatické uzavření zdrojů - pro třídy implementující `AutoCloseable`

java

```java
// Starý způsob:
FileReader reader = null;
try {
    reader = new FileReader("soubor.txt");
    // čtení souboru
} catch (IOException e) {
    e.printStackTrace();
} finally {
    if (reader != null) {
        try {
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

// Nový způsob (try-with-resources):
try (FileReader reader = new FileReader("soubor.txt")) {
    // čtení souboru
} catch (IOException e) {
    e.printStackTrace();
} // reader se automaticky uzavře
```

📋 Užitečné metody výjimek

java

```java
try {
    rizikovyKod();
} catch (Exception e) {
    // Zpráva výjimky
    String zprava = e.getMessage();

    // Typ výjimky
    String typ = e.getClass().getSimpleName();

    // Stack trace (všechny volání metod)
    e.printStackTrace(); // vypíše do konzole

    // Stack trace jako String
    StringWriter sw = new StringWriter();
    PrintWriter pw = new PrintWriter(sw);
    e.printStackTrace(pw);
    String stackTrace = sw.toString();

    // Původní příčina (pokud je výjimka zabalená)
    Throwable pricina = e.getCause();
}
```

⚡ Best Practices

✅ Dělej

1. **Ošetřuj konkrétní výjimky** před obecnými
2. **Používej informativní zprávy** při vyhazování výjimek
3. **Zaloguj chyby** před jejich přehozením
4. **Používej try-with-resources** pro zdroje
5. **Neřeš Error třídy** (OutOfMemoryError apod.)

java

```java
// ✅ Dobře
try {
    int vysledek = deleni(a, b);
} catch (ArithmeticException e) {
    logger.error("Dělení nulou: a=" + a + ", b=" + b, e);
    throw new IllegalArgumentException("Nelze dělit nulou", e);
} catch (Exception e) {
    logger.error("Neočekávaná chyba při dělení", e);
    throw e;
}
```

❌ Nedělej

1. **Nepřehlušuj výjimky** prázdným catch blokem
2. **Nepoužívej výjimky** pro řízení toku programu
3. **Nevyhazuj příliš obecné** výjimky (Exception)
4. **Nezapomeň na cleanup** v finally bloku

java

```java
// ❌ Špatně
try {
    rizikovyKod();
} catch (Exception e) {
    // Nic - výjimka je "spolknuta"
}

// ❌ Špatně - řízení toku
try {
    return pole[index];
} catch (ArrayIndexOutOfBoundsException e) {
    return null; // Lepší je zkontrolovat index předem
}
```

🔍 Debugging výjimek

Čtení Stack Trace

```
Exception in thread "main" java.lang.NullPointerException: Cannot invoke
"String.length()" because "text" is null
    at com.example.MojeTrida.zpracujText(MojeTrida.java:15)
    at com.example.MojeTrida.main(MojeTrida.java:8)
```

Význam:

- **Typ výjimky**: `NullPointerException`
- **Zpráva**: `Cannot invoke "String.length()" because "text" is null`
- **Místo**: `MojeTrida.java:15` (řádek 15)
- **Call stack**: Sekvence volání metod

📚 Příklady z praxe

Validace vstupů

java

```java
public class Uzivatel {
    private String email;
    private int vek;

    public void setEmail(String email) {
        if (email == null || email.trim().isEmpty()) {
            throw new IllegalArgumentException("Email nesmí být prázdný");
        }
        if (!email.contains("@")) {
            throw new IllegalArgumentException("Neplatný formát emailu");
        }
        this.email = email;
    }

    public void setVek(int vek) {
        if (vek < 0) {
            throw new IllegalArgumentException("Věk nesmí být záporný");
        }
        if (vek > 150) {
            throw new IllegalArgumentException("Věk je nereálný");
        }
        this.vek = vek;
    }
}
```

Práce se soubory

```java
public String nactiSoubor(String cesta) throws IOException {
    try (BufferedReader reader = Files.newBufferedReader(Paths.get(cesta))) {
        return reader.lines()
                     .collect(Collectors.joining("\n"));
    } catch (IOException e) {
        throw new IOException("Nelze načíst soubor: " + cesta, e);
    }
}
```

🎯 Shrnutí pro zkoušku

1. **Znaj rozdíl** mezi checked a unchecked výjimkami
2. **Umíj napsat** try-catch-finally blok
3. **Rozuměj** hierarchii výjimek (Throwable → Exception → RuntimeException)
4. **Znaj nejčastější** RuntimeException (NullPointer, ArrayIndexOutOfBounds...)
5. **Umíj vyhodit** vlastní výjimku pomocí throw
6. **Rozuměj** try-with-resources
7. **Víš kdy použít** throws v hlavičce metody
8. **Dokážeš přečíst** stack trace

Tip: Výjimky nejsou jen pro chyby - jsou to normální část objektového návrhu! 🚀