



Výnimky – dokončenie

Pojmy zavedené v 8. prednáške₍₁₎

- behové chyby
- komunikácia klient-server
- trojvrstvový model aplikácie

Pojmy zavedené v 8. prednáške₍₂₎

- defenzívna programovanie
- server - informovanie o chybách
- používateľa
- klienta
 - návratová hodnota
 - výnimka

Pojmy zavedené v 8. prednáške₍₃₎

- výnimky - hierarchia
- druhy
 - Error
 - Exception
 - RuntimeException
- kontrolované výnimky
- nekontrolované výnimky

Pojmy zavedené v 8. prednáške₍₄₎

- vyhadzovanie výnimiek
- príkaz throw
- klauzula throws

Pojmy zavedené v 8. prednáške₍₅₎

- zachytávanie výnimiek
- prikaz try
- try-catch
- try-catch-finally
- try-finally

Ciel' prednášky

- výnimky
 - vlastné výnimky
 - zotavenie sa po chybách/predchádzanie chybám
- vstupy a výstupy
 - štandardný vstup a výstup
 - textové a binárne súbory
 - objektové prúdy
- príklad: KCalB

Vlastné triedy výnimiek

- nová kontrolovaná výnimka – potomok triedy Exception (alebo od nej odvodenej triedy)
- nová nekontrolovaná výnimka – potomok triedy RuntimeException
- konvencia – názov končí slovom Exception

Dôvody použitia vlastných výnimiek

- neexistuje štandardná výnimka, ktorá dostatočne popisuje chybu
- vyčlenenie samostatnej výnimky
- existuje doplňujúca informácia, ktorá môže mať vplyv na riešenie chyby – kontrolované výnimky

Príklad triedy vlastnej výnimky

```
public class NeznameDieloException
```

```
    extends Exception
```

```
{
```

```
    public NeznameDieloException(String paNazov)
```

```
{
```

```
        super("Dielo s nazvom " + paNazov +  
              " nebolo najdene.");
```

```
}
```

```
}
```

Vlastná výnimka

- NeznameDieloException
- definícia triedy – odčlenenie výnimky od štandardných
- kontrolovaná výnimka
- zostavenie textu chybového hlásenia v konštruktore
- využitie konštruktora predka
- ostatné metódy získava od predka

Využitie vlastnej výnimky – Katalog

```
public void vymaz(String paTitul)
                    throws NeznameDieloException
{
    ...
    AVIDielo polozka = this.vyhľadaj(paTitul);
    if (polozka == null) {
        throw new NeznameDieloException(paTitul);
    }
    aZoznamPoloziek.remove(položka);
}
```

Metóda vymaz triedy Katalog

- klauzula throws – vyhadzuje kontrolovanú výnimku
- test výsledku vyhľadania
- vyhodenie výnimky

Zachytenie vlastnej výnimky₍₁₎

```
try {  
    paKatalog.vymaz(paParameter);  
} catch (NeznameDieloException ex) {  
    aTerminal.vypisRiadok(ex.getMessage());  
    ex.printStackTrace();  
}
```

Zachytenie vlastnej výnimky₍₂₎

- príkaz try pri poslaní správy
- chránený príkaz vymazania z katalógu
- zachytenie výnimky = zotavenie sa z chyby
 - informácia pre používateľa
- využitie zdedených metód
 - getMessage() – text správy o chybe
 - printStackTrace() – výpis textu na štandardný chybový výstup – objekt System.err

Kontrola stavu v konštruktore

- nesprávne parametre
- dve možnosti reakcie:
 - zmena počiatočného stavu na správny
 - vyhodenie výnimky

Zmena počiatočného stavu na správny

- riešenie používané doteraz
- napr. AutomatMHD
 - nekladná hodnota ceny lístka => použitie prednastavenej hodnoty
- klient nevie o probléme
 - možnosť informovať používateľa cez terminál
 - možnosť informovať klienta cez špeciálnu správu `dajChybu()`

Vyhodenie výnimky v konštruktore

- oznámenie chyby klientovi
- inštancia sa vôbec nevytvorí

Príklad – server

```
public CD(String paTitul, String paAutor, int paPocetSkladieb,  
          int paCelkovyCas)  
{  
    super(paTitul, paCelkovyCas);  
  
    if (paAutor == null || paAutor.isEmpty()) {  
        throw new IllegalArgumentException("paAutor musi byt  
                                         nastaveny");  
    }  
    aAutor = paAutor;  
  
    if (paPocetSkladieb <= 0 ) {  
        throw new IllegalArgumentException("paPocetSkladieb musi  
                                         byt vacsi ako nula");  
    }  
    aPocetSkladieb = paPocetSkladieb;  
}
```

Príklad – klient – správne???

```
try {  
    CD cd = new CD(titul, autor, pocetSkladieb, cas);  
    paDatabaza.pridaj(cd);  
} catch (IllegalArgumentException ex) {  
    System.out.println("Nesprávne parametre!");  
}
```

Príklad – klient – nesprávne

```
CD cd;  
try {  
    cd = new CD(titul, autor, pocetSkladieb, cas);  
} catch (IllegalArgumentException ex) {  
    System.out.println("Nesprávne parametre!");  
}  
paDatabaza.pridaj(cd);
```

Chyba pri preklade

The screenshot shows a Java code editor window titled "VykonavacPridaj". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar contains "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu labeled "Source Code" is open. The code editor displays the following Java code:

```
51 }  
52  
53 CD cd;  
54 try {  
55     cd = new CD(titul, autor,  
56 } catch (IllegalArgumentExceptionExcep  
57     System.out.println("Nesprávny titul  
58 }  
59     paDatabaza.pridaj(cd);  
60 } else if (paParameter.equals("dvojice")) {  
61     String titul;
```

A tooltip at the bottom of the editor window reads "variable cd might not have been initialized". There are two buttons at the bottom right: a question mark icon and the word "saved".

Príklad – klient – tiež nesprávne

```
CD cd = null;  
try {  
    cd = new CD(titul, autor, pocetSkladieb, cas);  
} catch (IllegalArgumentException ex) {  
    System.out.println("Nesprávne parametre!");  
}  
paDatabaza.pridaj(cd);
```

Chyba za behu

```
BlueJ: Terminal Window - 12-binarnySubor
Options

> pridaj cd
Pridanie noveho CD:
zadaj titul:
zadaj autora:
zadaj pocet skladieb:5
zadaj celkovy cas:10
Nespravne parametre CDcka!

java.lang.IllegalArgumentException: nova polozka nesmie byt null
    at fri.dcaib.databaza.Databaza.pridaj (Databaza.java:30)
    at fri.dcaib.prikazy.VykonavacPridaj.vykonaj (VykonavacPrida
    at fri.dcaib.prikazy.Prikaz.vykonajPrikaz (Prikaz.java:41)
    at fri.dcaib.Aplikacia.spusti (Aplikacia.java:27)
```

Príklad – klient – správne riešenie

```
CD cd = null;  
try {  
    cd = new CD(titul, autor, pocetSkladieb, cas);  
} catch (IllegalArgumentException ex) {  
    System.out.println("Nesprávne parametre!");  
}  
if (cd != null) {  
    paDatabaza.pridaj(cd);  
}
```

Možnosti práce s výnimkami v klientovi

- zotavenie sa z chyby
- predchádzanie chybám

Zotavenie sa z chyby

- príkazy v bloku catch
- riešenie situácie
- niekedy nový pokus s celým príkazom try
 - napr. vstupy
 - nemusí sa podaríť
 - pozor na nekonečný cyklus

Príklad – zápis do súboru₍₁₎

...

```
boolean uspech = false;  
int pocetPokusov = 0;  
while (uspech == false) {  
    // pokus o zapis do suboru – prikaz try  
}
```

...

Príklad – zápis do súboru₍₂₎

```
try {  
    // zapis do suboru  
    uspech = true; // posledny prikaz bloku  
} catch (IOException e) {  
    if (pocetPokusov < MAX_POCET) {  
        pocetPokusov++;  
        // vyber inu moznost, kam zapisat subor  
    } else {  
        throw e; // vynimka sa posuva vyssie  
    }  
}
```

Predchádzanie chybám₍₁₎

- vyžaduje spoluprácu klienta so serverom
- príklad – duplicita v katalógu CD a DVD
 - vyhľadaj – vráti null, ak titul v katalógu nie je
 - ak sa skontroluje, nevyhodí sa výnimka
- server musí poskytnúť možnosť
- klient nemusí riešiť príslušnú výnimku

Predchádzanie chybám₍₂₎

- nevýhody
 - zvyšuje sa implementačná závislosť
 - server sa poistuje – dvojité testovanie

Výnimky a testovanie

- neprípustné parametre spôsobia výnimku?
- ! výnimka ukončí test chybou
 - nie je to, čo chceme
- dve možnosti riešenia
 - try-fail-catch
 - expected (JUnit 4.x)

try-fail-catch

```
@Test  
public void newCDZlyTitul()  
{  
    try {  
        new CD(null, "Niekto", 5, 30);  
        fail("Neošetrená hodnota null ako titul");  
    } catch (IllegalArgumentException ex) {  
        // výnimka nastala = ok  
    }  
}
```

expected (JUnit 4.x)

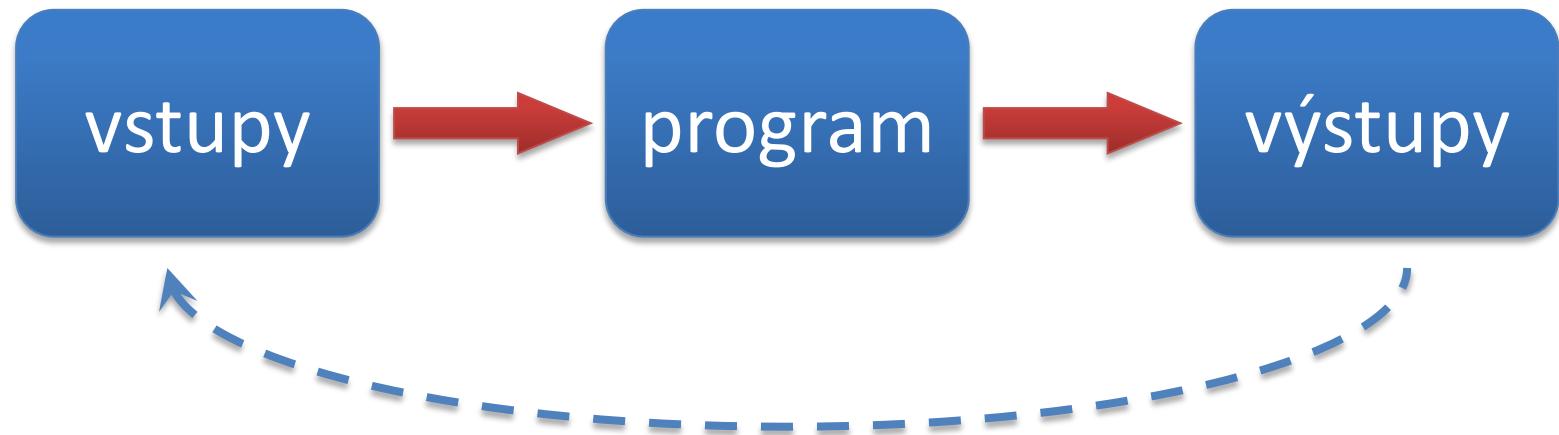
```
@Test(expected = IllegalArgumentException.class)
public void newCDZlyAutor()
{
    new CD("Nieco", null, 5, 30);
}
```

Vstupy a výstupy

Vstupy a výstupy programov

- programy – algoritmy
 - hromadnosť => nie na jediné použitie
 - jeden druh úlohy
 - rôzne začiatočné hodnoty
 - rôzne výsledky

Vstupy a výstupy programov



Štandardný vstup a výstup



Štandardný vstup a výstup

- Trieda System
- atribúty objektového typu:
 - in – vstup
 - out, err – výstup

Štandardný výstup

- výpis textu na obrazovku
- priame použitie v programe
- out – statický typ PrintStream
 - najčastejšie správy
 - `print(String paText)`
 - `println()` – nový riadok
 - `println(String paText)`

Štandardný vstup

- čítanie textu (postupnosti znakov) z klávesnice
- čaká na ukončenie písania na klávesnici – enter
- kontrolný opis na obrazovku – JVM (+OS)
- in – statický typ `InputStream`
 - `System.in` sa väčšinou nevyužíva v programe priamo
 - využitie v programe pomocou triedy `Scanner`
 - `Scanner` – obaľuje `System.in`

Scanner next a hasNext

- oddelovače – skupina znakov (Whitespace): medzera, tabulátor, novy riadok
- rozdelenie vstupného riadku pomocou oddelovačov na slová (tokeny)
- boolean hasNext()
 - existuje ešte slovo?
- String next()
 - prečíta nasledujúce slovo, všetky oddelovače pred slovom vynechá

Scanner

- ďalšie správy inštancii triedy Scanner
- boolean hasNextPrimitivnyTyp()
 - nasledujúce slovo
- PrimitivnyTyp nextPrimitivnyTyp()
 - InputMismatchException
- boolean hasNextLine() -
- String nextLine()

Scanner – vytvorenie

- čítanie štandardného vstupu:

```
Scanner klavesnica = new Scanner(System.in);
```

- čítanie ľubovoľného reťazca

```
String riadok = ...;
```

```
Scanner parser = new Scanner(riadok);
```

Projekt KCalB

- katalóg CD a DVD
- objekty treba vždy nanovo vytvárať
- (alebo nechať navždy zapnutý program)
- úloha: ukladanie katalógu do súborov

Súbory



Trieda File₍₁₎

- poskytuje základné informácie o súbore
 - názov súboru
 - čas zmeny
 - práva na súbor
 - existencia súboru na disku
 - absolútna cesta
 - ...

Trieda File₍₂₎

- operácie so súborom
 - vytvorenie
 - vymazanie
 - premenovanie
 - ...
- spolupráca programu v jazyku Java s OS
- nepracuje s obsahom súboru

Možnosti práce s obsahom

- textové súbory
- binárne súbory
- serializácia objektov

Práca so súbormi

- čítanie
 - otvorenie súboru na čítanie
 - postupné čítanie obsahu
 - zatvorenie súboru
- zápis
 - otvorenie súboru na zápis
 - postupný zápis nového obsahu
 - zatvorenie súboru

Textové súbory

- čitateľné človekom
- najčažšie na strojové spracovanie
- príklady
 - textové súbory .txt
 - zdrojové kódy .java
 - ...

Textové súbory v jazyku Java

- čítanie
 - trieda Scanner
 - Scanner(File paSubor)
- zápis
 - trieda PrintWriter
 - PrintWriter(File paSubor)
 - rozhranie podobné ako System.out
- nutnosť uzavretia súboru správou close()

Metóda zapis v triede Katalog

```
public void zapis() throws IOException  
{  
    File subor = new File("Katalog.txt");  
    PrintWriter zapisovac = new PrintWriter(subor);  
    for (AudiovizualneDielo dielo : aZoznam) {  
        dielo.zapis(zapisovac);  
    }  
    zapisovac.close();  
}
```

Metóda zapis v triede AudiovizualneDielo

```
public abstract void zapis(PrintWriter paZapisovac);
```

Metóda zapis v triede CD

```
public void zapis(PrintWriter paZapisovac)
```

```
{
```

```
    paZapisovac.println("CD");
```

```
    paZapisovac.println(this.dajTitul());
```

```
    paZapisovac.println(this.dajCelkovyCas());
```

```
    paZapisovac.println(aAutor);
```

```
...
```

```
}
```

Metóda zapis v triede DVD

```
public void zapis(PrintWriter paZapisovac)
```

```
{
```

```
    paZapisovac.println("DVD");
```

```
    paZapisovac.println(this.dajTitul());
```

```
    paZapisovac.println(this.dajCelkovyCas());
```

```
    paZapisovac.println(aReziser);
```

```
...
```

```
}
```

Výsledok v súbore

CD

The Best of

30

The Beatles

12

DVD

Tenkrat na zapade

90

Neznamy autor

Metóda nacitaj v triede Katalog

```
public void nacitaj() throws IOException  
{  
    aZoznam.clear();  
    File subor = new File("Katalog.txt");  
    Scanner citac = new Scanner(subor);  
    while (citac.hasNextLine()) {  
        ...  
    }  
    citac.close();  
}
```

Metóda nacitaj v triede Katalog

```
String typ = citac.nextLine();
if (typ.equals("CD")) {
    aZoznam.add(new CD(citac));
} else {
    aZoznam.add(new DVD(citac));
}
```

Nový konštruktor v triede CD

```
public CD(Scanner paCitac)
{
    super(paCitac.nextLine(), paCitac.nextInt());
    aAutor = paCitac.nextLine();
    ...
}
```

Nový konštruktor v triede DVD

```
public DVD(Scanner paCitac)
{
    super(paCitac.nextLine(), paCitac.nextInt());
    aReziser = paCitac.nextLine();
    ...
}
```

Problémy s textovými súbormi₍₁₎

- príklad nebude fungovať
- problém s koncom riadku za číslom (30, 12)
- nextLine po nextInt prečíta prázdný reťazec
- možné riešenia
 - zapisovať čísla bez konca riadku – viac problémov ako úžitku
 - prečítať najsôkôr riadok a ten postupne čítať pomocou triedy Scanner
 - po čísle prečítať zvyšok riadku a zabudnúť

Problémy s textovými súbormi₍₂₎

- ďalší problém – viacriadkové komentáre
- riešenie = DÚ
- problémy s formátovaním
 - zápis – bez problémov ľubovoľný formát
 - čítanie – komplikované

Binárne súbory

- nečitateľné človekom, resp. čitateľné komplikovane
- formát rovnaký ako v pamäti – jednoducho spracovateľný počítačom
- pri dodržaní poradia zápisu a čítania dát (musia byť zhodné) nehrozí problém s formátovaním

Binárne súbory v jazyku Java

- čítanie
 - trieda FileInputStream
 - FileInputStream(File paSubor)
- zápis
 - trieda FileOutputStream
 - FileOutputStream(File paSubor)
- komplikované – možnosť zapisovať iba byte[]
- nutnosť uzavretia súboru správou close()

Binárne súbory v jazyku Java

- zjednodušenie – práca s primitívnymi typmi jazyka Java
- čítanie
 - trieda DataInputStream
 - DataInputStream(InputStream paStream)
- zápis
 - trieda DataOutputStream
 - DataOutputStream(OutputStream paStream)
- nutnosť uzavretia súboru správou close()

Trieda DataOutputStream

- `writePrimitivnyTyp(PrimitivnyTyp paHodnota)`
 - zapíše hodnotu primitívneho typu do súboru
- `writeUTF(String paRetazec)`
 - zapíše reťazec do súboru

Trieda DataInputStream

- *PrimitivnyTyp* [readPrimitivnyTyp\(\)](#)
 - prečíta hodnotu primitívneho typu zo súboru
- String [readUTF\(\)](#)
 - prečíta reťazec zo súboru

Metóda zapis v triede Katalog

```
public void zapis() throws IOException
{
    File subor = new File("Katalog.bin");
    FileOutputStream stream
        = new FileOutputStream(subor);
    DataOutputStream zapisovac
        = new DataOutputStream(stream);
    for (AudiovizualneDielo dielo : aZoznam) {
        dielo.zapis(zapisovac);
    }
    zapisovac.close();
}
```

Metóda zapis v triede AudiovizualneDielo

public abstract void zapis

(DataOutputStream paZapisovac)

throws IOException;

Metóda zapis v triede CD

```
public void zapis(DataOutputStream paZapisovac)
                    throws IOException
{
    paZapisovac.writeUTF("CD");
    paZapisovac.writeUTF(this.dajTitul());
    paZapisovac.writeInt(this.dajCelkovyCas());
    paZapisovac.writeUTF(aAutor);
    ...
}
```

Metóda zapis v triede DVD

```
public void zapis(DataOutputStream paZapisovac)
                    throws IOException
{
    paZapisovac.writeUTF("DVD");
    paZapisovac.writeUTF(this.dajTitul());
    paZapisovac.writeInt(this.dajCelkovyCas());
    paZapisovac.writeUTF(aReziser);
    ...
}
```

Výsledok v súbore

The screenshot shows a hex editor window titled "Lister - [D:\Working\informatika\dcaib\12-binarnySubor\Katalog.txt]". The menu bar includes File, Edit, Options, and Help. The status bar shows 100 % completion. The main area displays binary data in two columns and ASCII text on the right. The first five lines of data are:

Hex Address	Binary Data	ASCII Data
00000000:	00 02 43 44 00 0B 54 68 65 20 42 65 73 74 20 6F	CD The Best o
00000010:	66 00 00 00 1E 00 0B 54 68 65 20 42 65 61 74 6C	f . The Beatl
00000020:	65 73 00 00 00 0C 00 03 44 56 44 00 11 54 65 6E	es 『DVD Ten
00000030:	68 72 61 74 20 6E 61 20 7A 61 70 61 64 65 00 00	krat na zapade
00000040:	00 5A 00 0D 4E 65 7A 6E 61 6D 79 20 61 75 74 6F	Z Neznamy auto
00000050:	72	r

Metóda nacitaj v triede Katalog

```
public void nacitaj()
{
    boolean koniecSuboru = false;
    aZoznam.clear();
    File subor = new File("Katalog.bin");
    FileInputStream strm = new FileInputStream(subor);
    DataInputStream citac = new DataInputStream(strm);
    while (!koniecSuboru) {
        ...
    }
    citac.close();
}
```

Metóda nacitaj v triede Katalog

```
try {  
    String typ = citac.readUTF();  
    if (typ.equals("CD")) {  
        aZoznam.add(new CD(citac));  
    } else {  
        aZoznam.add(new DVD(citac));  
    }  
} catch (EOFException ex) {  
    koniecSuboru = true;  
}
```

Nový konštruktor v triede CD

```
public CD(DataInputStream paCitac)  
{  
    super(paCitac.readUTF(), paCitac.readInt());  
    aAutor = paCitac.readUTF();  
    ...  
}
```

Nový konštruktor v triede DVD

```
public DVD(DataInputStream paCitac)  
{  
    super(paCitac.readUTF(), paCitac.readInt());  
    aReziser = paCitac.readUTF();  
    ...  
}
```

Objektové prúdy

- aplikácia
 - štruktúra spolupracujúcich objektov
 - v operačnej pamäti
 - ľubovoľne zložitá
 - prirodzene nelineárna
- ukladanie na vonkajšiu pamäť
 - postupnosť informácií – lineárna forma
- objektové streamy – serializácia
 - ukladanie stavu objektov na vonkajšiu pamäť
 - obnovenie stavu objektov z vonkajšej pamäte

ObjectOutputStream

- uloží celý objekt = serializácia
- hodnoty všetkých atribútov
 - hodnoty primitívnych typov
 - objektové atribúty ako objekty - rekurzia
- rieši problém viacnásobných odkazov na objekt
- writeObject

ObjectInputStream

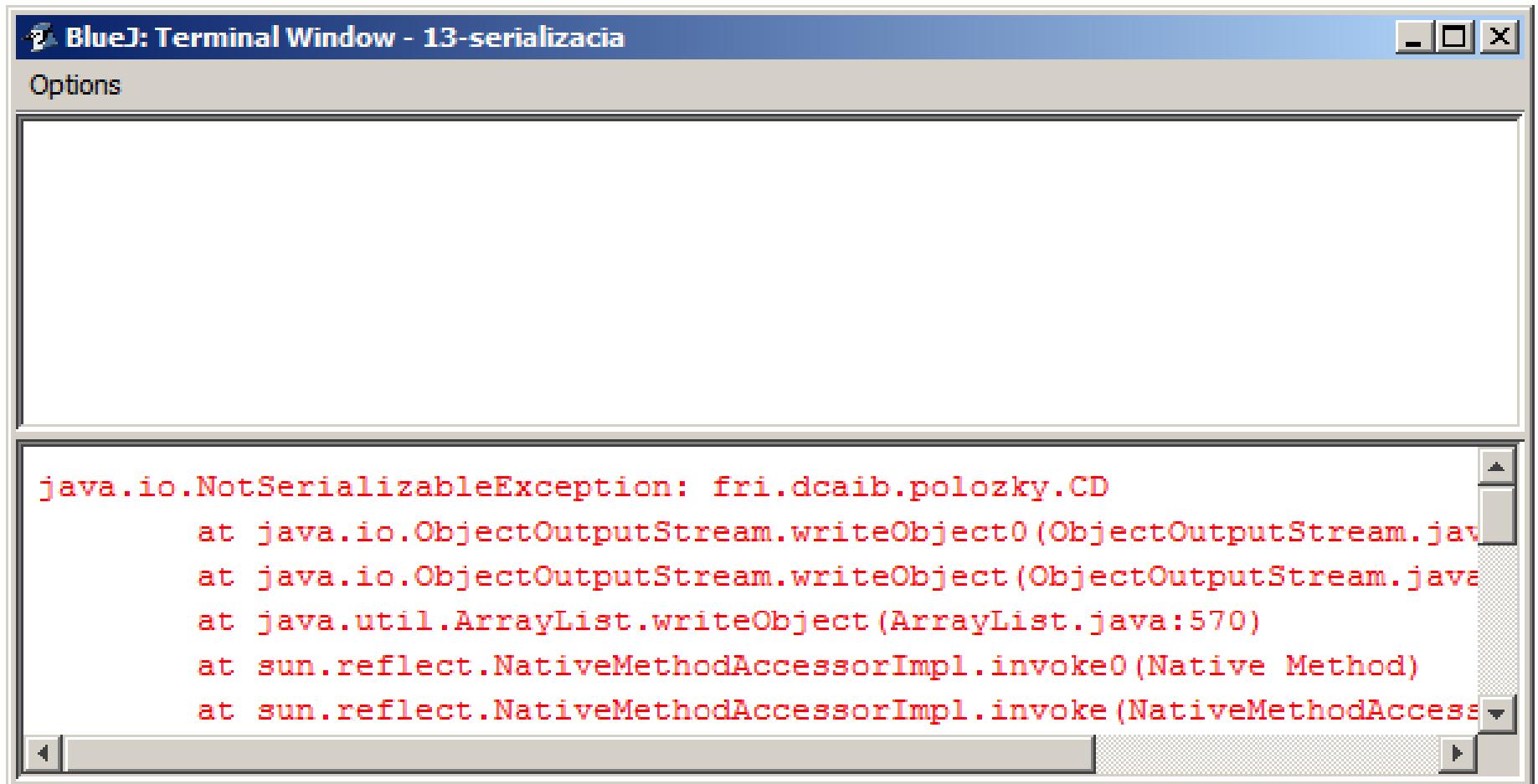
- obnoví celý objekt = deserializácia
- hodnoty všetkých atribútov
- rieši problém viacnásobných odkazov na objekt
- readObject



Podmienka

- trieda ukladaného/obnovovaného objektu implementuje interface Serializable
 - môže dedit' od predka
- značkovací interface
 - žiadne metódy navyše – vnútorné riešenie Java
- triedy z balíčkov knižnice Java
 - bežne implementujú interface Serializable
 - nie je to 100% pravidlo

Prvok kontajnera, chýba Serializable



The screenshot shows a terminal window titled "BlueJ: Terminal Window - 13-serializacia". The window contains a single line of red text representing a Java stack trace:

```
java.io.NotSerializableException: fri.dcaib.polozky.CD
    at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1187)
    at java.io.ObjectOutputStream.writeObject(ObjectOutputStream.java:348)
    at java.util.ArrayList.writeObject(ArrayList.java:570)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccesso
```

Príklad KCalB – podmienka

```
import java.io.Serializable;
```

```
public class CD extends AutorskeDielo  
    implements Serializable
```

```
import java.io.Serializable;
```

```
public class DVD extends AutorskeDielo  
    implements Serializable
```

Metóda zapis v triede Katalog

```
public void zapis() throws IOException  
{  
    File subor = new File("Katalog.bin");  
    FileOutputStream stream =  
        new FileOutputStream(subor);  
    ObjectOutputStream zapisovac =  
        new ObjectOutputStream(stream);  
    zapisovac.writeObject(aZoznam);  
    zapisovac.close();  
}
```

Metóda nacitaj v triede Katalog

```
public void nacitaj()  
    throws IOException, ClassNotFoundException  
{  
    File subor = new File("Katalog.bin");  
    FileInputStream stream =  
        new FileInputStream(subor);  
    ObjectInputStream citac =  
        new ObjectInputStream(stream);  
    aZoznam = (ArrayList<AVIDielo>)citac.readObject();  
    citac.close();  
}
```

Vďaka za pozornosť