Informatika 2

Výnimky

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Pojmy zavedené v 7. prednáške(1)

- abstraktná trieda
  - v reálnom svete
  - v jazyku Java
  - v BlueJ/UML
- konkrétna trieda
- abstraktná metóda
  - v reálnom svete
  - v jazyku Java

Pojmy zavedené v 7. prednáške(2)

- vzťah is-a
- dedičnosť a extenzia triedy
- instanceof v hierarchii dedičnosti

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Pojmy zavedené v 7. prednáške(3)

- Liskovej substitučný princíp
- riešenie problému štvorec-obdĺžnik pomocou abstraktnej triedy
- riziká porušenia LSP
  - znižovanie rizika
- trieda Object a LSP

- polymorfizmus vs. dedičnosť

Pojmy zavedené v 7. prednáške(4)

- vytváranie dedičnosti
  - princíp gen-spec
- implementačná závislosť v dedičnosti

Cieľ prednášky

- komunikácia klient-server
- defenzívne programovanie
- ošetrenie chýb
  - na strane servera
  - na strane klienta
- výnimky

- príklad: KCaIB

Behové chyby(1)

- chyby v kóde, ktoré spôsobia pád aplikácie
  - nekorektné ukončenie
  - aplikácia sa vymkne spod kontroly
  - nečakané ukončenie v lepšom prípade

Behové chyby(2)

- má padnúť internetový prehliadač, ak žiadame otvoriť neexistujúcu web stránku?
  (napr. sme pri písaní adresy urobili chybu)

- má sa zavrieť (ukončiť prácu) súborový manažér, ak chceme uložiť obrázok na USB kľúč, na ktorom už nie je dostatok miesta?

- má skončiť hra, ak hráč napíše nesprávny príkaz?

Behové chyby(3)

- problémy tohto typu chceme riešiť

Komunikácia klient-server(1)

- komunikácia dvojice objektov – odosielateľ správou žiada adresáta o službu
- odosielateľ – klient žiada o službu
  - klient je aktívny objekt
- adresát – server poskytuje službu
  - server nevykonáva žiadnu akciu z vlastnej vôle, len reakcia na žiadosť klienta

Komunikácia klient-server(2)

- správa – parametre
- správnosť parametrov – prípustné hodnoty

- kto je zodpovedný?

- klient – definícia hodnôt parametrov
- server – použitie hodnôt parametrov

Klient

- definícia hodnôt parametrov
  - doplňujúce informácie správy
  - podrobnejšia charakteristika žiadosti
  - klient vychádza zo svojho stavu

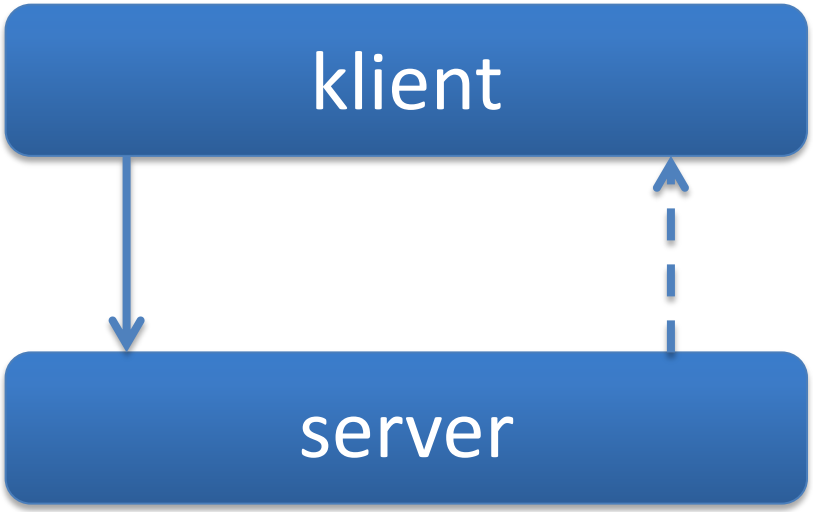
- spracovanie návratovej hodnoty

Server

- server – použitie hodnôt parametrov
  - parametre sú charakterizované typom
  - nie každá hodnota musí byť prípustná
  - určitá hodnota parametra môže vyžadovať určitý stav servera
  - parametre môžu vyžadovať zmenu stavu servera

- server nesmie dopustiť, aby sa dostal do nekorektného stavu

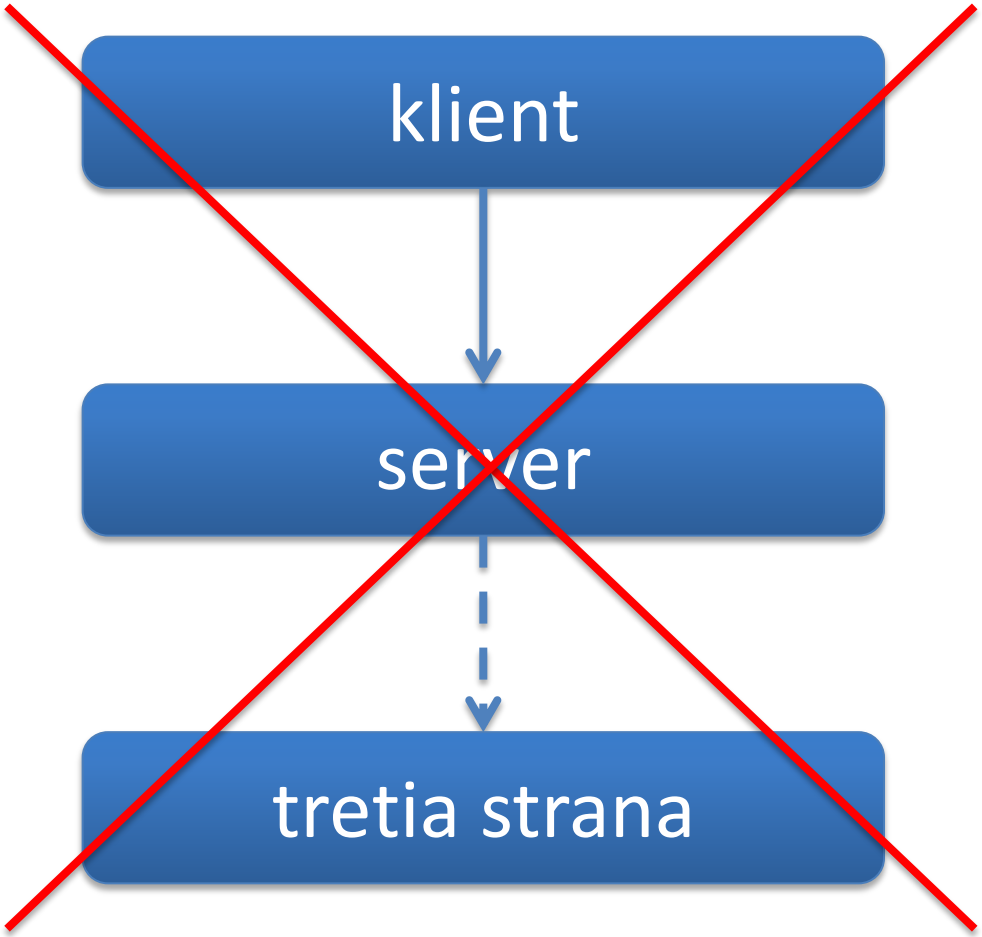
Korektný vzťah klient-server



správa klientovi

oznámenie výsledku

Porušenie vzťahu klient-server



správa klientovi

oznámenie výsledku

Príklad porušenia vzťahu klient-server

- výpis na terminál
  - výpis oznámenia o korektnom ukončení operácie
  - oznámenie chyby používateľovi
- klient sa nedozvie o výsledku operácie

Komunikácia klient-server

- kto je zodpovedný?

- extrémne (nie nesprávne) spôsoby riešenia
  - klient vie presne, čo chce a ako to má od servera korektne žiadať
  - server predpokladá, že sa môže stať čokoľvek a musí sám zabezpečiť, aby pracoval len korektne

Východiská

- klient nekladie nekorektné požiadavky zámerne

- existujúce nekorektné požiadavky – chyby
- existujú problémy mimo aplikácie – klient nemôže poznať

Riešenie – odpovede na otázky

- Aká veľká kontrola správ od klienta má byť v metódach servera?
- Ako má server oznamovať chyby klientovi?
- Ako má klient predchádzať nekorektným požiadavkám na server?
- Ako má klient reagovať na oznámenie o chybe?

Defenzívne programovanie

- kontrola parametrov – server
- kontrola návratovej hodnoty – klient

Riešenie na strane servera

- kontrola parametrov
- reakcia na zistenú chybu
  - nevykoná akciu
  - nezmení svoj stav
  - informovanie o chybe

Príklad – Katalog.vymaz(1)

```
public void vymaz(String paTitul)
{
   AVIDielo polozka = this.vyhladaj(paTitul);
   aZoznamPoloziek.remove(polozka);
}
```

- paTitul
  - nesmie byť null
  - nesmie byť prázdny reťazec
  - musí existovať AVIDielo s takým titulom

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Príklad – Katalog.vymaz(2)

```
public void vymaz(String paTitul)
{
    if (paTitul == null || paTitul.isEmpty()) {
        // osetrenie chyby 1,2
    } else {
        ...
    }
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Príklad – Katalog.vymaz(3)

```
...
AVIDielo polozka = this.vyhladaj(paTitul);
if (polozka == null) {
    // Osetrenie chyby 3
} else {
    aZoznamPoloziek.remove(polozka);
}
...
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Informovanie o chybe

- žiadne – ???
- informovanie používateľa – výpis
- informovanie klienta

Informovanie používateľa

- informácia o chybe sa zobrazí používateľovi

- textová forma – terminál, okno
- zvukový výstup – „pípanie“ pri stlačení nesprávneho klávesu
- ...

- doteraz v projektoch – výpis na terminál

Oznámenie chyby používateľovi(1)

```
public void vymaz(String paTitul)
{
    if (paTitul == null || paTitul.isEmpty()) {
        System.out.println("Titul musi byt zadany!");
    } else {
        ...
    }
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Oznámenie chyby používateľovi(2)

```
...
AVIDielo polozka = this.vyhladaj(paTitul);
if (polozka == null) {
    System.out.println("Titul nie je v katalogu!");
} else {
    aZoznamPoloziek.remove(polozka);
}
...
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Informovanie používateľa – použitie

- prečo/kedy áno
  - jednoduché na implementáciu
  - ak používateľ potrebuje vedieť výsledok operácie
- prečo/kedy nie
  - klient sa nedozvie výsledok operácie
    - porušenie princípu klient/server
    - nie vždy je klientom používateľ
  - nie o všetkých výsledkoch má byť informovaný používateľ

Informovanie klienta – návratová hodnota

- info o chybe

- boolean – false (nastala chyba), true (nenastala)
- int – číselný kód (1 – taká chyba; 2 – iná chyba; 0 – bez chyby)
- enum – položky – rôzne typy chýb
- objekt – null v prípade chyby (napr. pri vyhľadávaní)

Návratová hodnota objekt

```
public AudiovizualneDielo vyhladaj(String paTitul)
{
   for (AVIDielo polozka : aZoznamPoloziek) {
      if (polozka.dajTitul().equals(paTitul)) {
         return polozka;
      }
   }
   return null;
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota objekt – použitie

- štandardné riešenie pri vyhľadávaní
- hľadaný objekt neexistuje => null

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota boolean(1)

```
public boolean vymaz(String paTitul)
{
    if (paTitul != null && !paTitul.isEmpty()) {
        ...
    }
    return false;
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota boolean(2)

```
...
AVIDielo polozka = this.vyhladaj(paTitul);
if (polozka != null) {
    aZoznamPoloziek.remove(polozka);
    return true;
}
...
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota boolean – použitie

- jediná informácia – nastala chyba
- vhodné pre jedinú možnosť
- nevhodné pre viac rôznych chýb v metóde – ak existuje viac možností ich ošetrenia

Návratová hodnota enum(1)

```
public ChybaVymazania vymaz(String paTitul)
{
    if (paTitul == null || paTitul.isEmpty()) {
        return ChybaVymazania.titulNezadany;
    } else {
        ...
    }
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota enum(2)

```
...
AVIDielo polozka = this.vyhladaj(paTitul);
if (polozka == null) {
    return ChybaVymazania.nenasielSa;
} else {
    aZoznamPoloziek.remove(polozka);
    // poprípade: return null;
    return ChybaVymazania.ziadna;
}
...
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Návratová hodnota enum – použitie

- možnosť označenia konkrétnych chýb
- komplikované a prácne
- takmer nepoužívané

Informovanie klienta – výnimky(1)

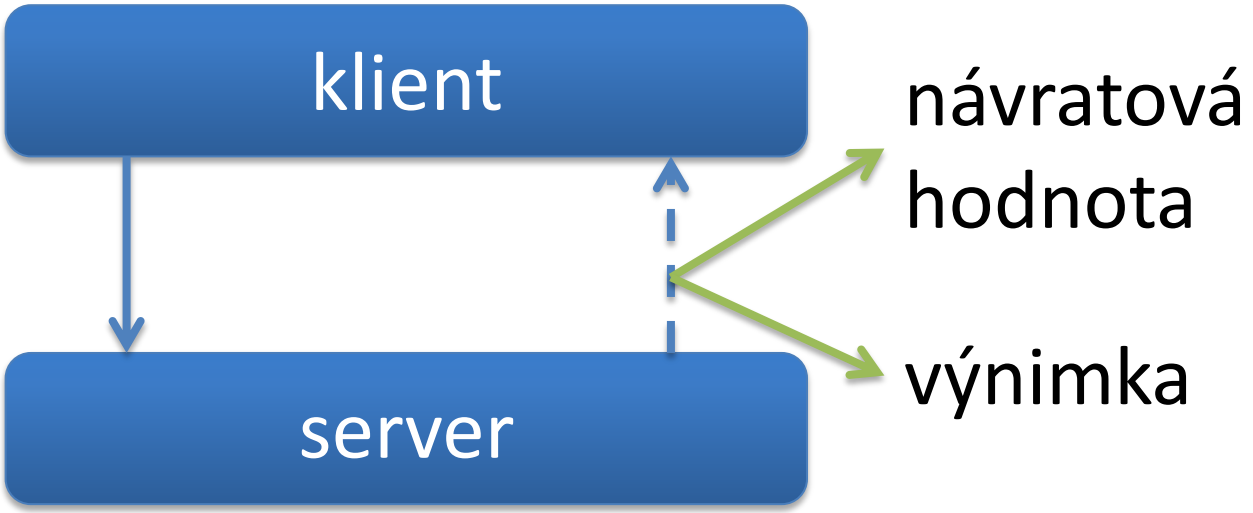
- kontrola výnimočných stavov
- oznamovanie chyby klientovi
- vo väčšine prípadov preferované riešenie

- doposiaľ známe ako behové chyby

Informovanie klienta – výnimky(2)

- výnimka – objekt
- informácia – názov triedy výnimky
- informácia – popis situácie

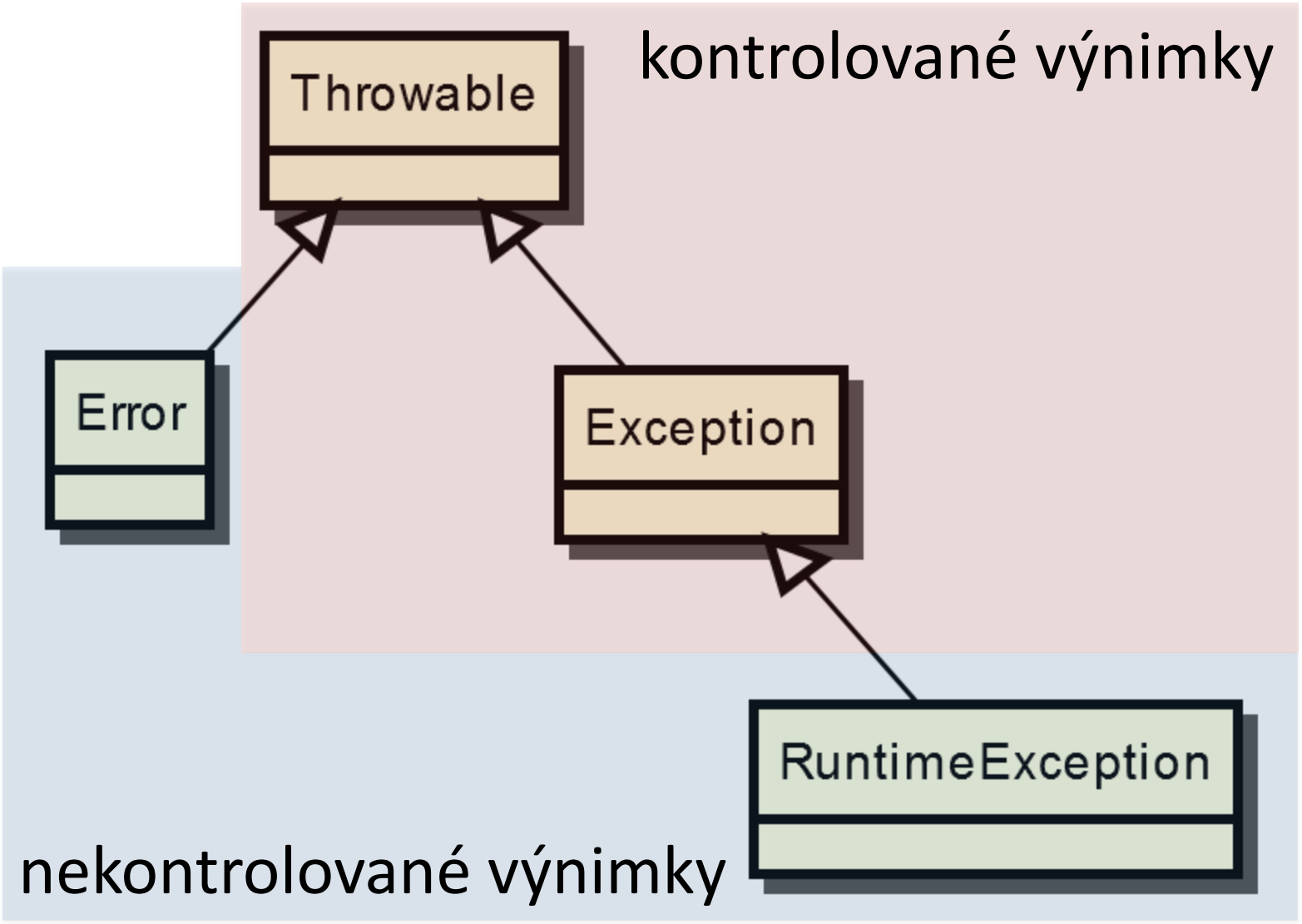
Klient-server – výnimky



Druhy výnimiek

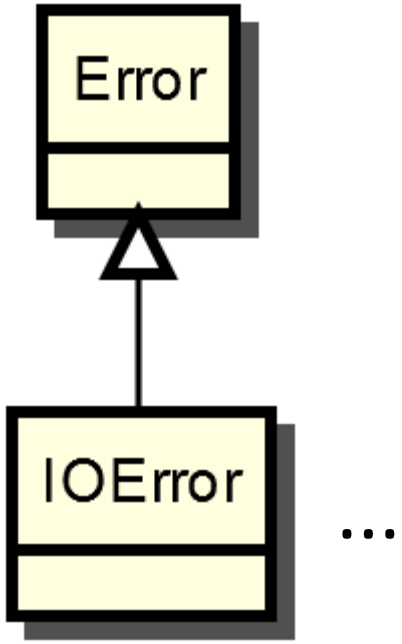
- kontrolované
  - klient nemôže ignorovať
  - kontroluje prekladač
- nekontrolované
  - klient môže ignorovať
  - spôsobí predčasné ukončenie aplikácie – pád
  - prekladač nerieši

Hierarchia výnimiek – koreň Throwable

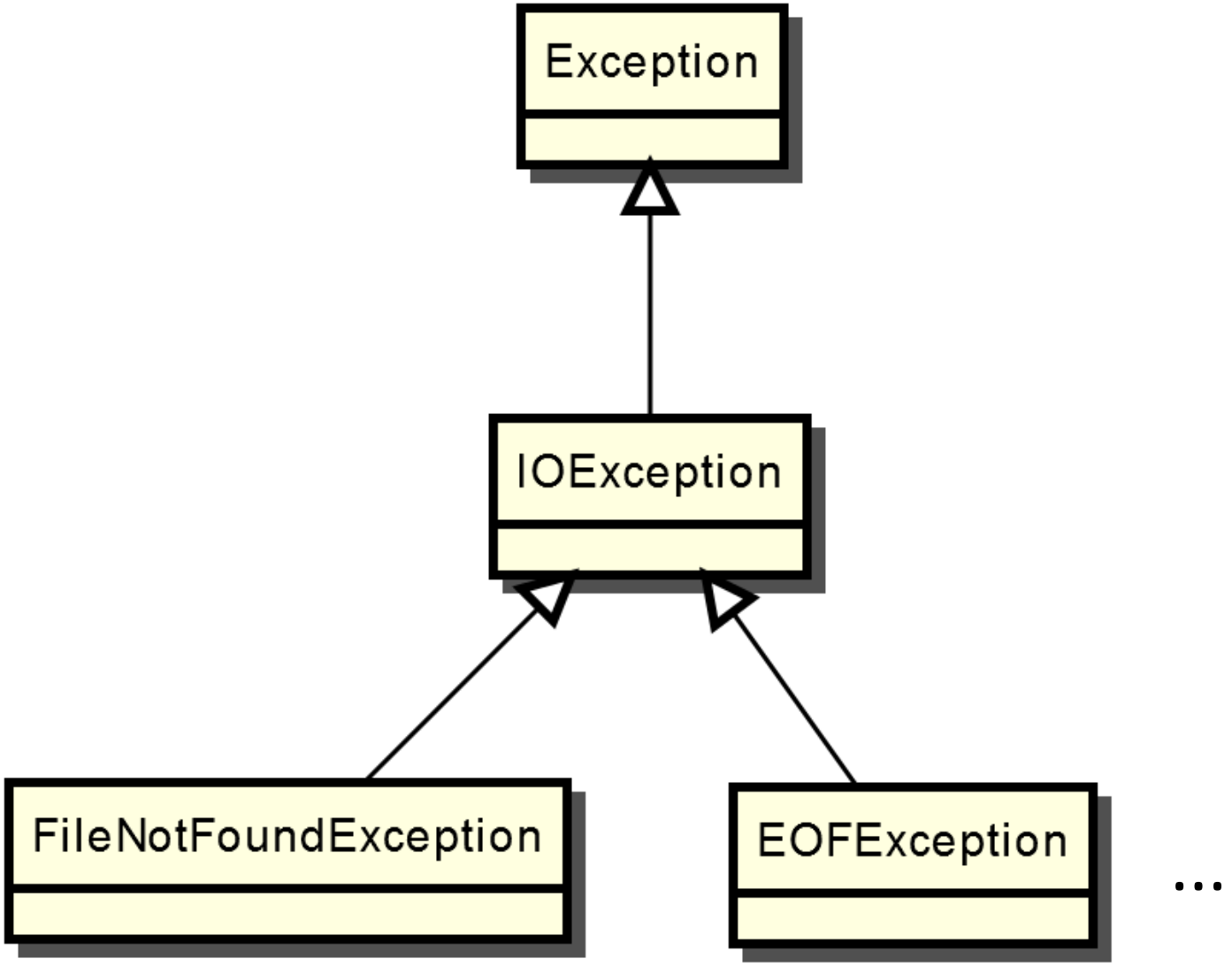


Chyby – Error

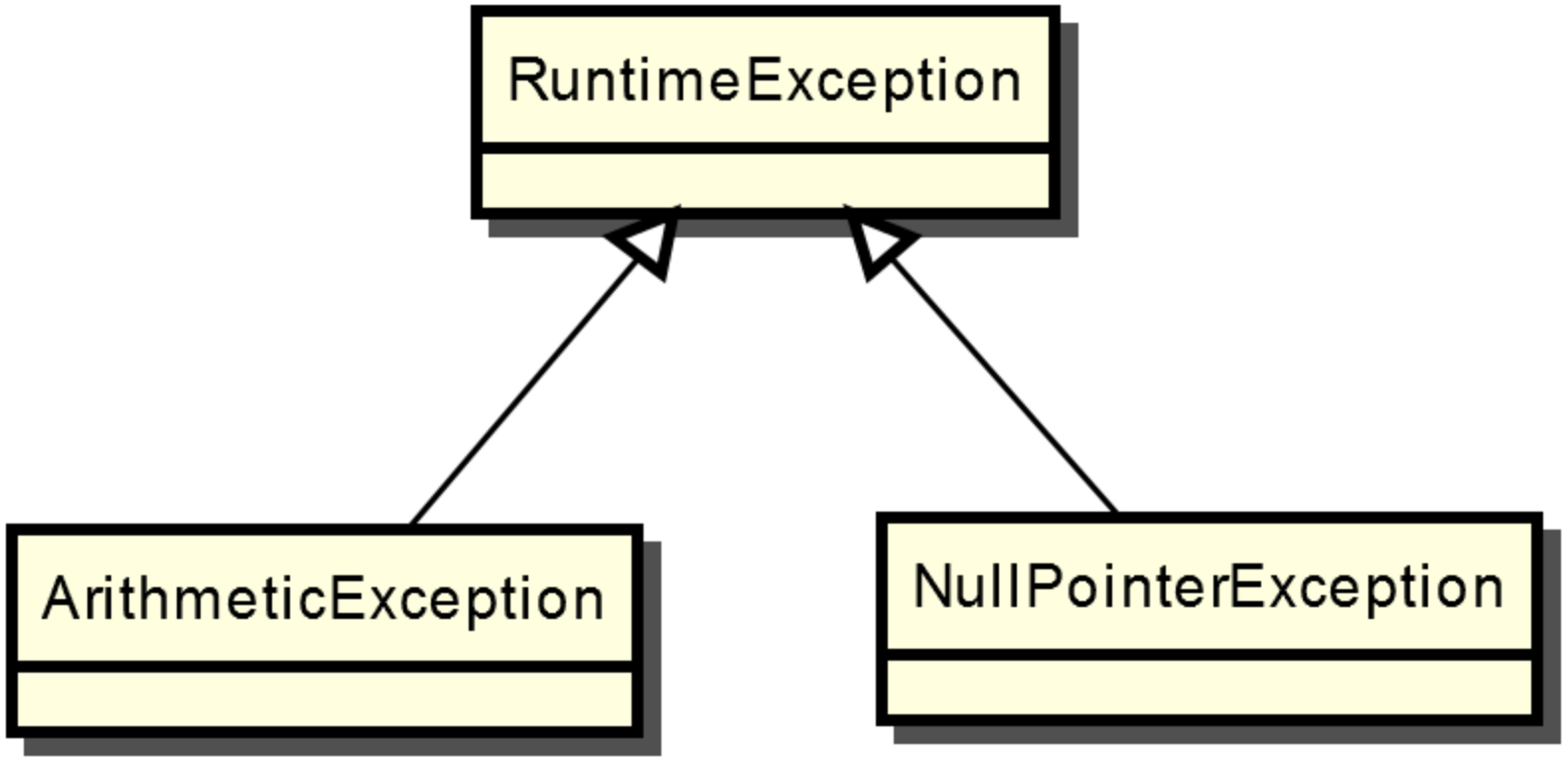
- systémové chyby
- neodchytávajú sa



Kontrolované výnimky



Nekontrolované výnimky



Výnimky a klient-server

- server výnimky vytvára – „vyhadzuje“
- klient výnimky zachytáva

Vyhodenie výnimky(1)

- príkaz throw

```
TypVynimky vynimka = new TypVynimky("popis");
throw vynimka;
```

- skrátený – bežný zápis

```
throw new TypVynimky("popis");
```

Vyhodenie výnimky(2)

- príkaz throw ukončuje vykonávanie metódy
- server vracia riadenie klientovi
- informuje klienta o výnimke

Klauzula throws

- throws v hlavičke metódy
  - výnimky vyhadzované v metóde
  - čiarkou oddelený zoznam
  - kontrolované povinne
  - nekontrolované nepovinne

```
public void uloz(String paNazovSuboru)
                                    throws IOException
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Priklad – vyhodenie výnimky(1)

```
public void vymaz(String paTitul)
{
    if (paTitul == null || paTitul.isEmpty()) {
        throw new IllegalArgumentException (
                                        "paTitul je prazdny");
    } else {
        ...
    }
}
```

Priklad – vyhodenie výnimky(2)

```
...
AVIDielo polozka = this.vyhladaj(paTitul);
if (polozka == null) {
    throw new IllegalArgumentException (
                                        "Titul sa nenasiel");
} else {
    aZoznamPoloziek.remove(polozka);
}
...
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Zachytávanie výnimky

- na strane klienta
- definujeme akcie, ktoré sa vykonajú v prípade výnimky
- príkaz try-catch

Príkaz try-catch(1)

- bloky: try, catch
- samostatne catch pre každú triedu výnimiek

```
try {
   // chranene prikazy
} catch (TriedaVynimky vynimka) {
   // prikazy vykonane v pripade vynimky
}
```

Príkaz try-catch(2)

```
try {
    paKatalog.vymaz(paParameter);
} catch (IllegalArgumentException ex) {
    aTerminal.vypisRiadok("Nenasiel som");
}
```


KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Príkaz try-catch(2)

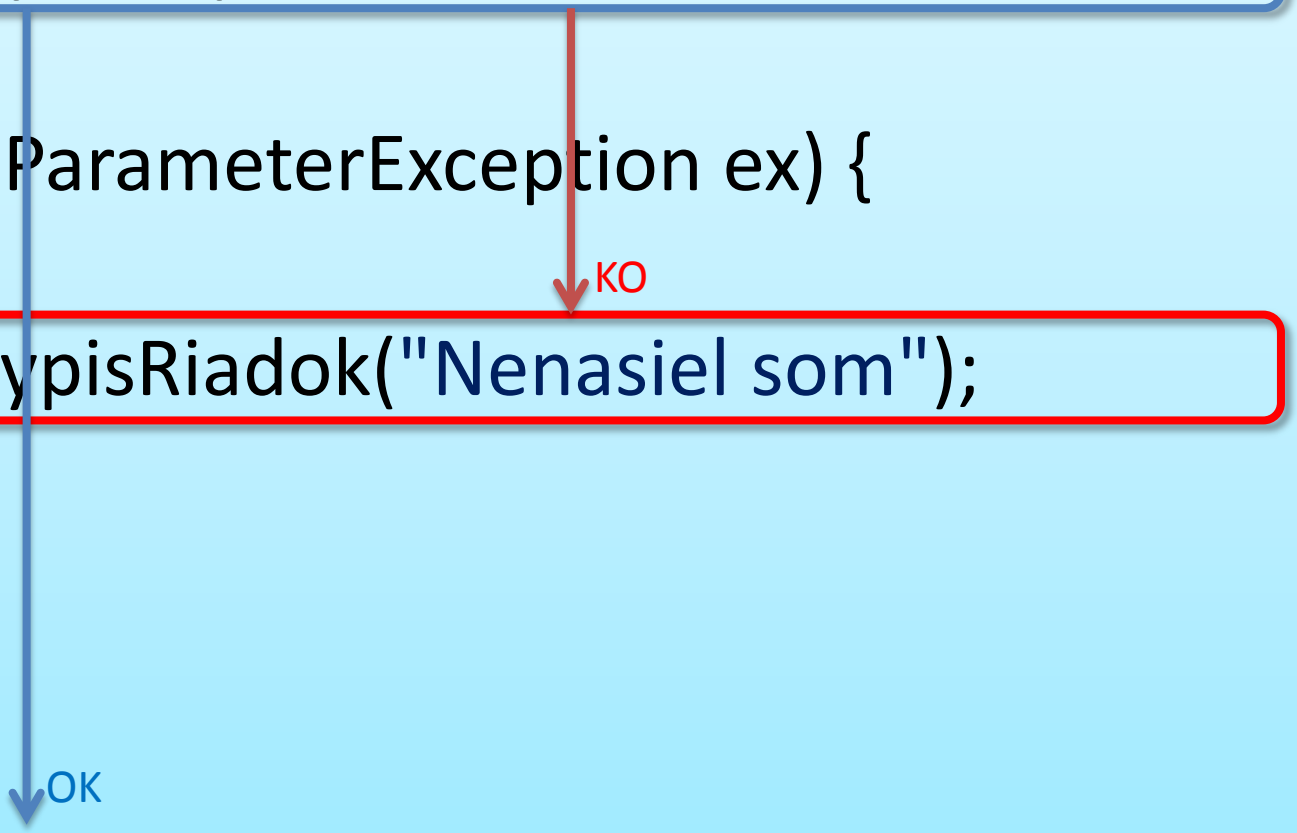
```
try {

    paKatalog.vymaz(paParameter);

} catch (IllegalParameterException ex) {

    aTerminal.vypisRiadok("Nenasiel som");

}
```



KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Viac blokov catch

- postupné vyhľadávanie typu výnimky
- POZOR! len prvý vhodný blok catch

- bloky catch musia byť usporiadané
- najskôr potomok – potom predok

- výnimka sa nenájde – behová chyba

Ošetrenie výnimky v bloku catch

- oznámenie používateľovi
- postúpenie výnimky ďalej
- anglicky (termínus-technikus) re-throw

```
throw ex;
```

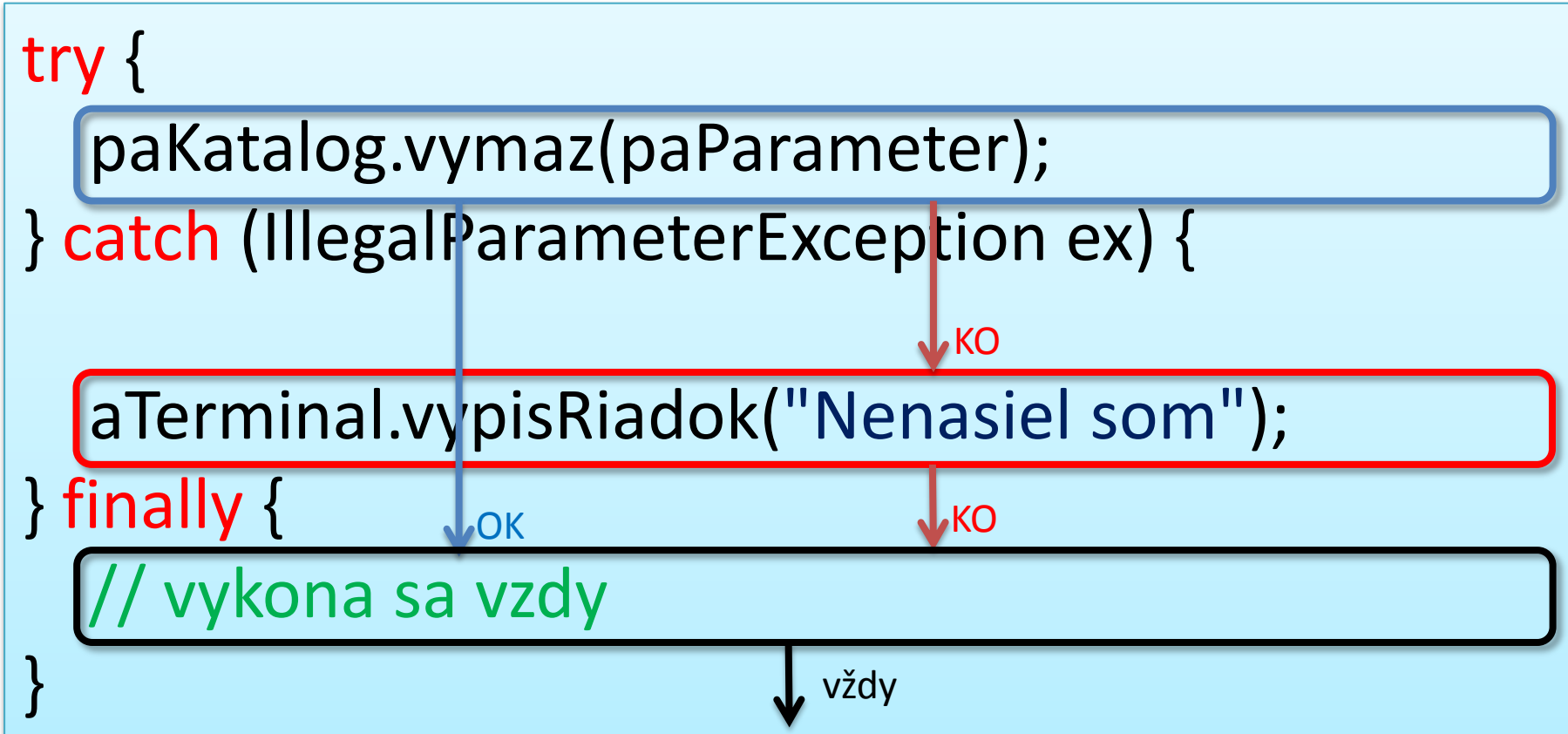

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

Príkaz try-catch-finally$_{(1)}$

- blok finally – príkazy vykonávané vždy
  - ok – všetky príkazy bloku try + blok finally
  - ko – blok catch (ak taký je) + blok finally
- uzatvorenie systémových zdrojov – súbory

- verzia try-finally
- príkazy try môžu byť vnorené – catch, finally

Príkaz try-catch-finally(2)



Vďaka za pozornosť