

Programmering för språkteknologer II

OH-serie 4:

Fel och exceptions (undantag)



UPPSALA
UNIVERSITET

Mats Dahllöf

Institutionen för lingvistik och filologi
September 2008

1

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Programmera "rätt"

- Man måste veta precis **vad** programmet skall göra.
- Man måste bestämma sig för rätt **algoritm**.
- Man måste **implementera** algoritmen på ett korrekt sätt.
- Sedan bör man givetvis göra detta så att det blir **effektivt** ur tids- och minnesanvändning.
- Och sedan bör det man skrivit vara **begripligt och övertygande**.
- Och man bör **kontrollera** att allt detta är fallet.

2

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Fel som man kan göra

- Algoritmrelaterade:
 - Stora tankefel: fel(aktig) algoritm.
 - Mindre tankefel, t.ex. tappat bort något specialfall.
- Implementationsrelaterade:
 - Felaktig kodning av algoritm.
 - Typnings- och syntaxfel.

3

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Konsekvenser av fel

- **Kompileringsfel:** t.ex. typnings- och syntaxfel fångas av kompilatorn. (Automatiskt och tidigt: bra.)
- **Exekveringsfel:** Felet leder till situationer som inte kan hanteras under exekveringen. T.ex. indexerings- och referensfel. Genererar en uttrycklig "protest".
- **Ej uppenbara konsekvenser:** Programmet gör fel men arbetar ytligt sett som det var tänkt. T.ex. ett program räknar ut fel ränta. Denna typ av fel kan lura människor på allvarliga sätt.

4

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Exekveringsfel i Java: "exceptions"

- Exekveringsfel i Java leder till att "exceptions", en typ av objekt, genereras.
- Exceptions kan sedan fångas upp ("be caught" av anropande metoder) och lämpliga åtgärder kan ha angivits i programmet.
- Om en exception inte fångas upp avbryts exekveringen och exceptionen rapporteras.

5

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

API (Application programming interface)

- Ett API dokumenterar en klass, vilka publika variabler, konstruktörer och metoder som finns, inklusive namn, argumentuppsättningar och returtyper.
- Ett API är alltså det man behöver veta för att använda en klass.
- Vilka exceptions som kan genereras (kastas, be thrown) är därför en annan viktig aspekt av API'n.

6

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Exempel 1: kod

```
public class StupidProgram1 {
    public static void main(String[] argv) {
        int[] myArray = {1,2,3};
        myArray[3] = 4;
    }
}
```

Vi försöker indexera utanför fältet. Borde inte gå.

7

Programmering för språkteknologer II — HT 2008 (Mats Dahllöf)

Exempel 1: exceptionrapport

```
linux> java StupidProgram1
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 3
at StupidProgram1.main(StupidProgram1.java:5)
```

8

Fånga exceptions: try-catch-satser

Utseende:

```
try BLOCK1
catch (ExceptionType1 name) BLOCK2
catch (ExceptionType2 name) BLOCK3
```

BLOCK1 – det block i vilket en exception kan kastas (be thrown).

BLOCK2 och BLOCK3 – det som händer vid uppfångande exceptions av ExceptionType1 och ExceptionType2 (första "träff").

9

Exempel 2: kod

```
public class StupidProgram2 {
    public static void main(String[] argv) {
        try {
            int[] myArray = {1,2,3};
            myArray[3] = 4;
        }
        catch (Exception e) {
            System.out.println("Något blev fel.");
        } } }
```

10

Exempel 2: resultat

```
linux> java StupidProgram2
Något blev fel.
```

11

Exempel 3: Två catch-alternativ

```
public class StupidProgram3 {
    public static void main(String[] argv) {
        try {
            int[] myArray = {1,2,3};
            myArray[3] = 4; }
        catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Indexeringsfel."); }
        catch (Exception e) {
            System.out.println("Annat fel."); } } }
```

12

Exceptions

- Exceptions: många standardsubklasser.
- Många situationer i Java då exceptions "kastas" enligt förutbestämda principer.
- Man kan definiera egna exceptionsubklasser.
- Man kan själv definiera situationer då exceptions "kastas" med throw-satser: `throw someThrowableObject;`

13

Kontrollerade (checked) exceptions

- Vissa exceptions måste fångas/kastas vidare: kontrollerade/"checked exceptions".
- Exceptions kan (som sagt) fångas med hjälp av try-catch-satser.
- En metod kan kasta checked exceptions vidare, men det måste deklarerars:


```
public void writeList() throws IOException {...
```

 (Den kan då fångas i en anropande metod istället.)

14

Exempel: checked exception

- Exempel vi ser i API:et att (konstruktorn):


```
public FileReader(File file)
    throws FileNotFoundException
```
- `FileNotFoundException` är checked. Ja, att en fil inte finns är ju ett problem som vi kan förutse och som vi bör ta om hand. Nu blir vi tvungna till det. Annars godkänner inte kompilatorn vår kod.
- Se filen `material/fsa/LineReader.java`

15