

Objektorientierte Programmierung in Java

Vorlesung 6 - Abstrakte Elemente

Emily Lucia Antosch

HAW Hamburg

04.11.2024

Inhaltsverzeichnis

1. Einleitung	3
2. Ausnahmebehandlung	7
3. Ausnahme werfen	16
4. License Notice	22

1. Einleitung

1.1 Wo sind wir gerade?

- In der letzten Vorlesung haben wir uns mit dem Erstellen von graphischen Oberflächen beschäftigt
- Sie können nun
 - ▶ Fenster erzeugen, in dem andere Elemente leben können,
 - ▶ Elemente mittels Layouts und Panels arrangieren
 - ▶ und Grafiken direkt in Java erzeugen.
- Heute geht es weiter mit den **Ausnahmebehandlungen**.

1.1 Wo sind wir gerade?

1. Imperative Konzepte
2. Klassen und Objekte
3. Klassenbibliothek
4. Vererbung
5. Schnittstellen
6. Graphische Oberflächen
7. **Ausnahmebehandlung**
8. Eingaben und Ausgaben
9. Multithreading (Parallel Computing)

1.2 Das Ziel dieses Kapitels


- Sie behandeln bei Programmausführung auftretende Ausnahmen und Fehler, um in aufgetretenen Ausnahmesituationen einen geordneten Programmfluss herzustellen.
- Sie definieren eigene, an die Bedürfnisse Ihrer konkreten Anwendung angepasste, Ausnahmeklassen.

2. Ausnahmebehandlung

? Frage

- Was wird von folgendem Programm ausgegeben?

```
1 public class ProvokeException {
2     public static void main(String[] args) {
3         int a = 3;
4         int b = 2;
5         printRatio(a, b);
6         System.out.println("Exiting main()");
7     }
8
9     public static void printRatio(int a, int b) {
10        int ratio = a / b;
11        System.out.println("Ratio = " + ratio);
12    }
13 }
```

 Java

? Frage

- Und was wird für a = 7 und b = 0 ausgegeben?

? Frage

- Was kann in einem Programm alles „schief gehen“?
- Wann muss der normale Programmfluss unterbrochen werden?
- Wann muss ein Programm beendet werden, wann kann es fortgeführt werden?



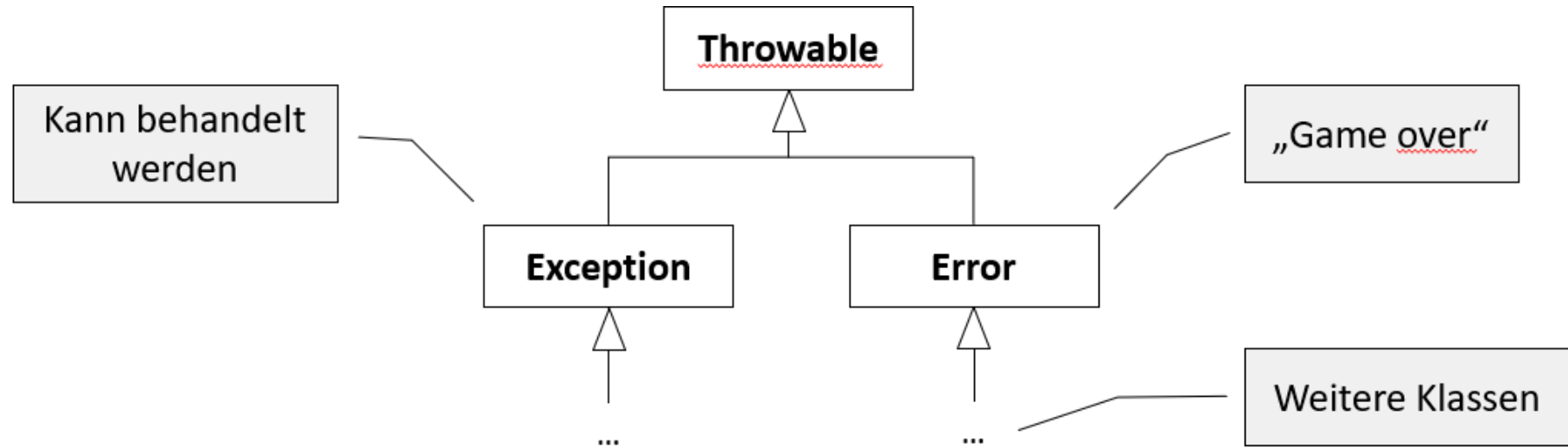
Beispiel

- Division durch Null
- Aufruf `a.method()`, obwohl Variable `a` den Wert `null` hat
- Negativer oder zu hoher Index für Arrays
- Wandeln der Zeichenkette „Dies ist Text“ in eine Ganzzahl vom Typ `int`
- Datei nicht gefunden
- Kein Speicher mehr verfügbar

2.1 Einführendes Beispiel

2. Ausnahmebehandlung

- Ausnahmen und Fehler durch Objekte spezieller Klassen dargestellt
- Basisklasse aller Ausnahmeklassen ist Throwable

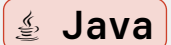


- Man unterscheide:
 - ▶ Exception (Ausnahme): Behandelbar, Programm kann fortgeführt werden
 - ▶ Error bzw. fatal error (Fehler): Nicht behandelbar, Programm beenden

! Merke

- Ausnahme wird auch als Oberbegriff für Ausnahmen und Fehler verwendet.
 - Ausnahmebehandlung wird auch als Exception handling bezeichnet.
-
- Einige Klassen für Ausnahmen:
 - ▶ Division durch Null (ArithmeticException)
 - ▶ Zugriff auf Methode oder Attribut über null-Referenz (NullPointerException)
 - ▶ Unzulässiger Feldindex (ArrayIndexOutOfBoundsException)
 - ▶ Unzulässige Zeichen beim Lesen einer Zahl (NumberFormatException)
 - ▶ Datei nicht gefunden (FileNotFoundException)

```
1 int[] array = {1, 2, 3, 4};  
2 System.out.println(array[4]);  
3  
4 String message;
```



2.1 Einführendes Beispiel

2. Ausnahmebehandlung

```
5 System.out.println(message.length());  
6  
7 int code = Integer.parseInt("12a4");
```

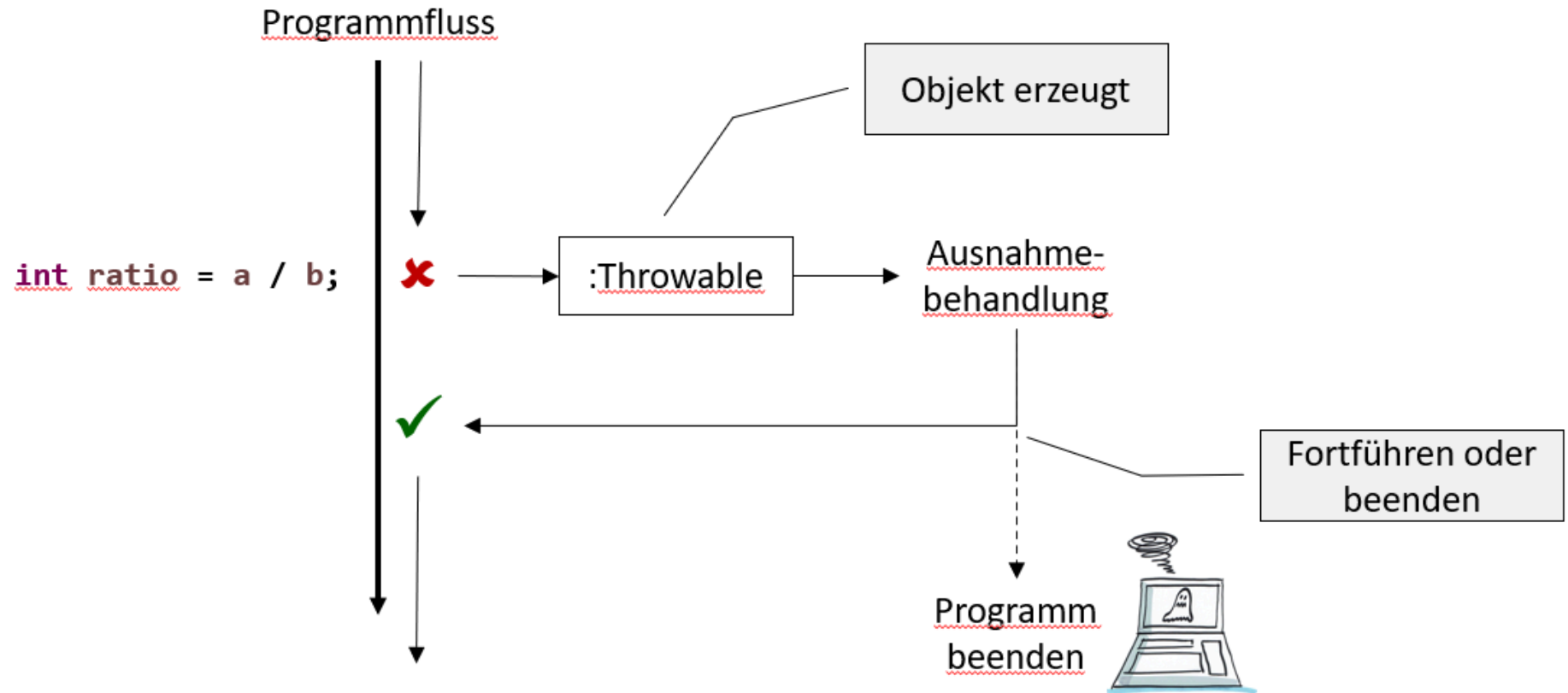
2.2 Ablauf der Ausnahmebehandlung

2. Ausnahmebehandlung

1. Ausnahme werfen:
 - Programmfluss wird unmittelbar unterbrochen
 - Objekt erzeugt, das Ausnahme repräsentiert
2. Ausnahme fangen:
 - Programmierer kann Ausnahme abfangen und behandeln

2.2 Ablauf der Ausnahmebehandlung

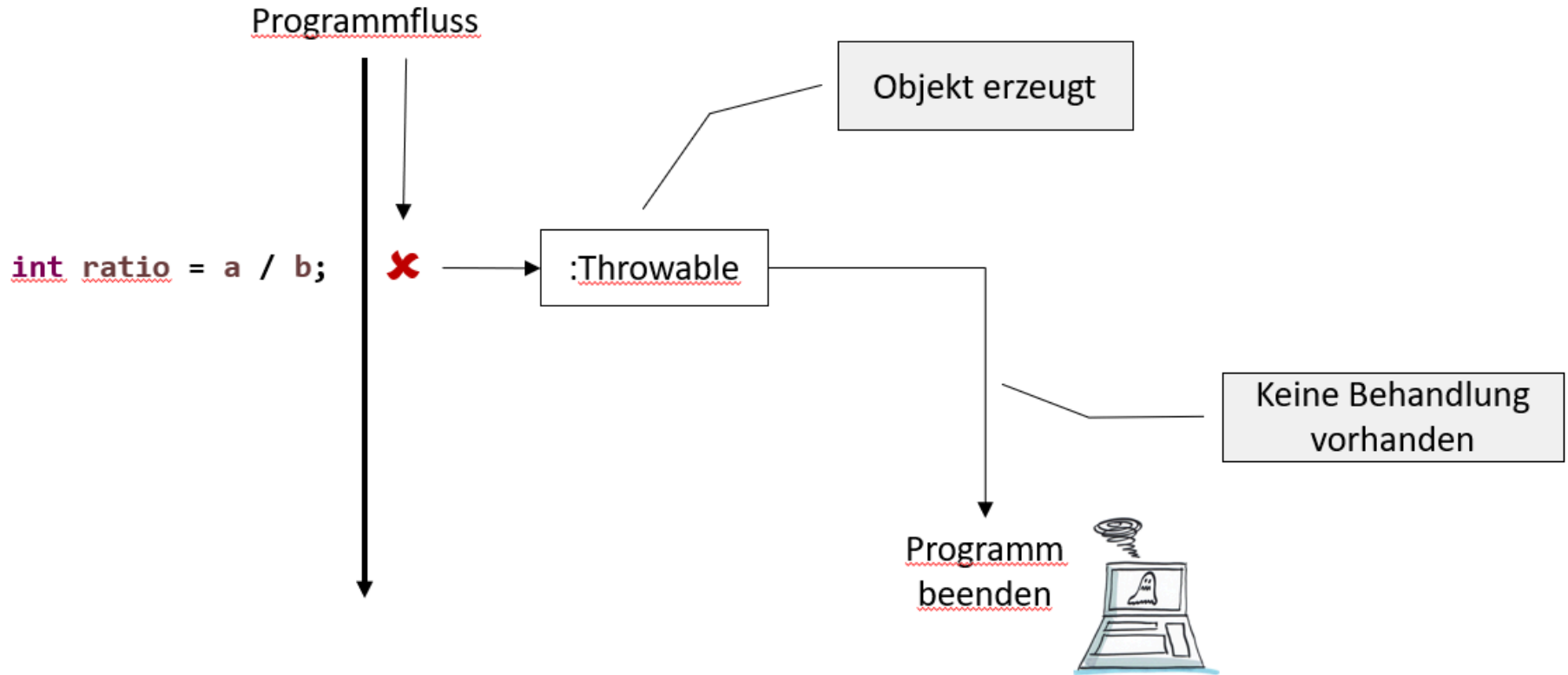
2. Ausnahmebehandlung



2.2 Ablauf der Ausnahmebehandlung

2. Ausnahmebehandlung

- Falls keine Ausnahmebehandlung programmiert: Programm wird beendet



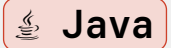
3. Ausnahme werfen

3.1 Ausnahme werfen

3. Ausnahme werfen

- Im Fehlerfall werden Ausnahmen automatisch erzeugt (z.B. Division durch Null).
- Ausnahmen lassen sich aber auch explizit werfen.

```
1 throw ExceptionObject;
```

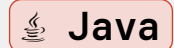


- Konstruktor kann String (z.B. als Fehlerbeschreibung) übergeben werden



Beispiel

```
1 throw new Exception();  
2 throw new Exception("Division by zero");  
3 Exception exception = new Exception(); throw exception;
```

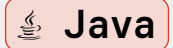


3.1 Ausnahme werfen

3. Ausnahme werfen

- Zur Veranschaulichung:
 - ▶ Werfen Sie eine Ausnahme, bevor versucht wird, durch Null zu teilen.

```
1  public class ThrowException {
2      public static void main(String[] args) {
3          int a = 3;
4          int b = 0;
5          printRatio(a, b);
6          System.out.println("Exiting main()");
7      }
8
9      public static void printRatio(int a, int b) {
10         int ratio = a / b;
11         System.out.println("Ratio = " + ratio);
12     }
13 }
```



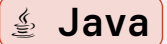
Java

3.1 Ausnahme werfen

3. Ausnahme werfen

- Beispiellösung:

```
1  public class ThrowException {
2      public static void main(String[] args) {
3          int a = 3;
4          int b = 0;
5          printRatio(a, b);
6          System.out.println("Exiting main()");
7      }
8
9      public static void printRatio(int a, int b) {
10         if (b == 0) {
11             throw new ArithmeticException("Division by zero");
12         }
13         System.out.println("Ratio = " + (a / b));
14     }
```



Java

3.1 Ausnahme werfen

3. Ausnahme werfen

```
15 }
```

3.1 Ausnahme werfen

3. Ausnahme werfen

- Ausgabe im Fehlerfall:
 - ▶ Ausnahmetyp (z.B. `ArithmeticException`)
 - ▶ Fehlermeldung (z.B. „Division by zero“)
 - ▶ Stacktrace (d.h. Kette der aufgerufenen Methoden)



Beispiel

```
1 Exception in thread "main" java.lang.ArithmeticException: Division by zero at
2     kapitel8_exceptions.ThrowException.printRatio(E02_ThrowException.java:20)
   at
3     kapitel8_exceptions.ThrowException.main(E02_ThrowException.java:14)
```

- Methode `main()` hat in Zeile 14 `printRatio()` aufgerufen
- Methode `printRatio()` hat in Zeile 20 die Ausnahme geworfen

4. License Notice

4.1 Attribution

- This work is shared under the CC BY-NC-SA 4.0 License and the respective Public License
- `link(„https://creativecommons.org/licenses/by-nc-sa/4.0/“)`
- This work is based off of the work Prof. Dr. Marc Hensel.
- Some of the images and texts, as well as the layout were changed.
- The base material was supplied in private, therefore the link to the source cannot be shared with the audience.