Programmieren in Java

Vorlesung 11: Generic Methods

Prof. Dr. Peter Thiemann

Albert-Ludwigs-Universität Freiburg, Germany

SS 2017

Inhalt

Lösungsbeispiel: w09-1 list-editor

Stichworte

- Framework Programmierung
- Abstrakte Klassen und Vererbung
- Template Methoden

Demo Time

Lösungsbeispiel: w09-2 sessions

Stichworte

- Vereinigung von Klassen
- Rekursive Klassen
- Interfaces
- "Offizieller Name": **Composite Pattern**

Demo Time

Generische Methoden

Typvariablen in Signaturen

- Nicht nur Klassen und Interfaces können generisch sein
- Auch Methoden können generische Parameter haben

Beispiel: Generische Auswahl

```
1  class Selector {
2    boolean state = false;
3    public boolean flip() {
4      return (state = !state);
5    }
6    /**
7    * Return one of the arguments dependening on the state.
8    */
9    public <T> T choose(T x, T y) {
10     return (state ? x : y);
11   }
12 }
```

Bemerkung

<T> vor dem Rückgabetyp deklariert T als Typvariable für die Signatur und den Rumpf der Methode.

Beispiel: Zählen

Aufgabe (noch nicht generisch)

Zähle die Anzahl der Vorkommen eines Strings in einem Array von Strings.

```
1  public static int countOccurrences(String goal, String[] subjects) {
2    int count = 0;
3    for (String s : subjects) {
4      if (Objects.equals (goal, s)) {
5        count++;
6      }
7    }
8    return count;
9  }
```

Beispiel: Zählen

Aufgabe (noch nicht generisch)

Zähle die Anzahl der Vorkommen eines Strings in einem Array von Strings.

```
1  public static int countOccurrences(String goal, String[] subjects) {
2    int count = 0;
3    for (String s : subjects) {
4      if (Objects.equals (goal, s)) {
5        count++;
6      }
7    }
8    return count;
9  }
```

Bemerkung

Nichts an diesem Code ist String-spezifisch!

Beispiel: Generisches Zählen

Aufgabe

Zähle die Anzahl der Vorkommen eines Objekts in einem Array von Objekten gleichen Typs.

```
1  public static <T> int countOccurrences(T goal, T[] subjects) {
2    int count = 0;
3    for (T s : subjects) {
4      if (Objects.equals (goal, s)) {
5        count++;
6      }
7    }
8    return count;
9  }
```

Beispiel: Minimum von String

Aufgabe (noch nicht generisch)

Bestimme das Minimum einer nicht-leeren Collection von Strings.

```
1  public static String minString(Collection<String> subjects) {
2    if (subjects.size() == 0) { throw new IllegalArgumentException(); }
3    String minVal;
4    boolean first = true;
5    for (String s : subjects) {
6      if (first) {
7        minVal = s;
8        first = false;
9      } else if (minVal.compareTo(s) > 0) {
10       minVal = s;
11     }
12   }
13   return minVal;
14 }
```

Beispiel: Minimum von Comparable

Aufgabe

Bestimme das Minimum einer nicht-leeren Collection von Objekten vom Typ T, die miteinander Comparable sind: T **extends** Comparable<T>.

```
1  public static <T extends Comparable<T>> T min(Collection<T> subjects) {
2    if (subjects.size() == 0) { throw new IllegalArgumentException(); }
3    T minVal;
4    boolean first = true;
5    for (T s : subjects) {
6      if (first) {
7        minVal = s;
8        first = false;
9      } else if (minVal.compareTo(s) > 0) {
10       minVal = s;
11     }
12   }
13   return minVal;
14 }
```

Beispiel: Minimum (alternativ mit Iterator)

```
1  public static <T extends Comparable<T>>
2     T min(Collection<T> subjects) {
3     Iterator<T> iter = subject.iterator();
4     if (!iterator.hasNext()) { throw new IllegalArgumentException(); }
5     T minVal = iterator.next();
6     while(iterator.hasNext()) {
7        T s = iterator.next();
8        if (minVal.compareTo(s) > 0) {
9           minVal = s;
10       }
11    }
12    return minVal;
13 }
```

Fragen