

Fakultät Informatik

Professur Softwaretechnologie

OOSE_04

JAVA-COLLECTION-FRAMEWORK

AM BEISPIEL VON TODO-LISTEN

Dr.-Ing. Birgit Demuth
Sommersemester 2019

Phasen in der Softwareentwicklung und Anwendung der UML

Phase	Modellierung in UML (Klassendiagramme)
Analyse (OOA)	aUML (UML-Analyseklassendiagramm)
Entwurf (OOD)	dUML (UML-Entwurfklassendiagramm)
Implementierung und Test (OOP)	jUML (UML-Implementationsklassendiagramm für Java)

Literatur:

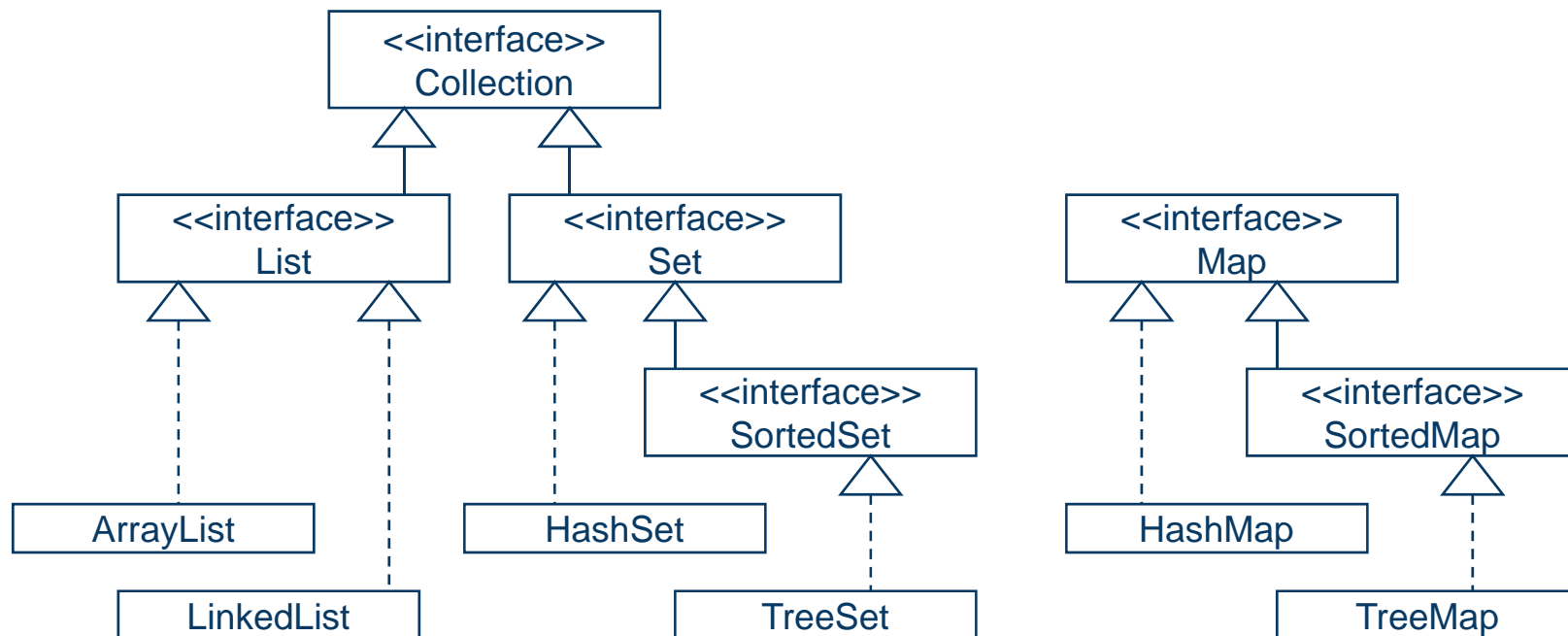
Birgit Demuth (Hrsg.): Softwaretechnologie für Einsteiger.

Pearson Studium, 2. geänderte Auflage, 2014

Abschnitt Klassendiagramme, S. 73 – 112

In Kürze gibt es die 3. Auflage (2019) in der Thalia TU Buchhandlung und in der SLUB!

Überblick zum Collection Framework von Java (Ausschnitt)



ToDo-Listen (aUML) [vgl. Klausur WS 2014/15]

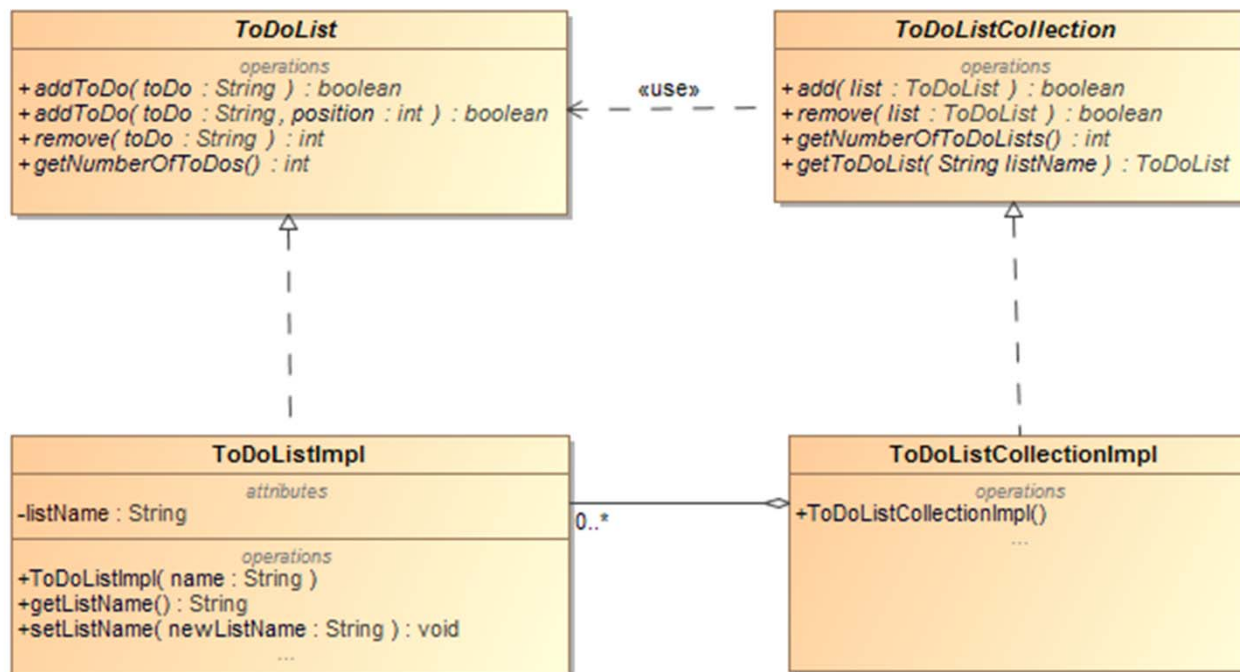


ToDoList: Eine ToDo-Liste beinhaltet in geordneter Reihenfolge „todos“.

ToDoListCollection: Die ToDo-Listen einer Person werden in einer Kollektion verwaltet.

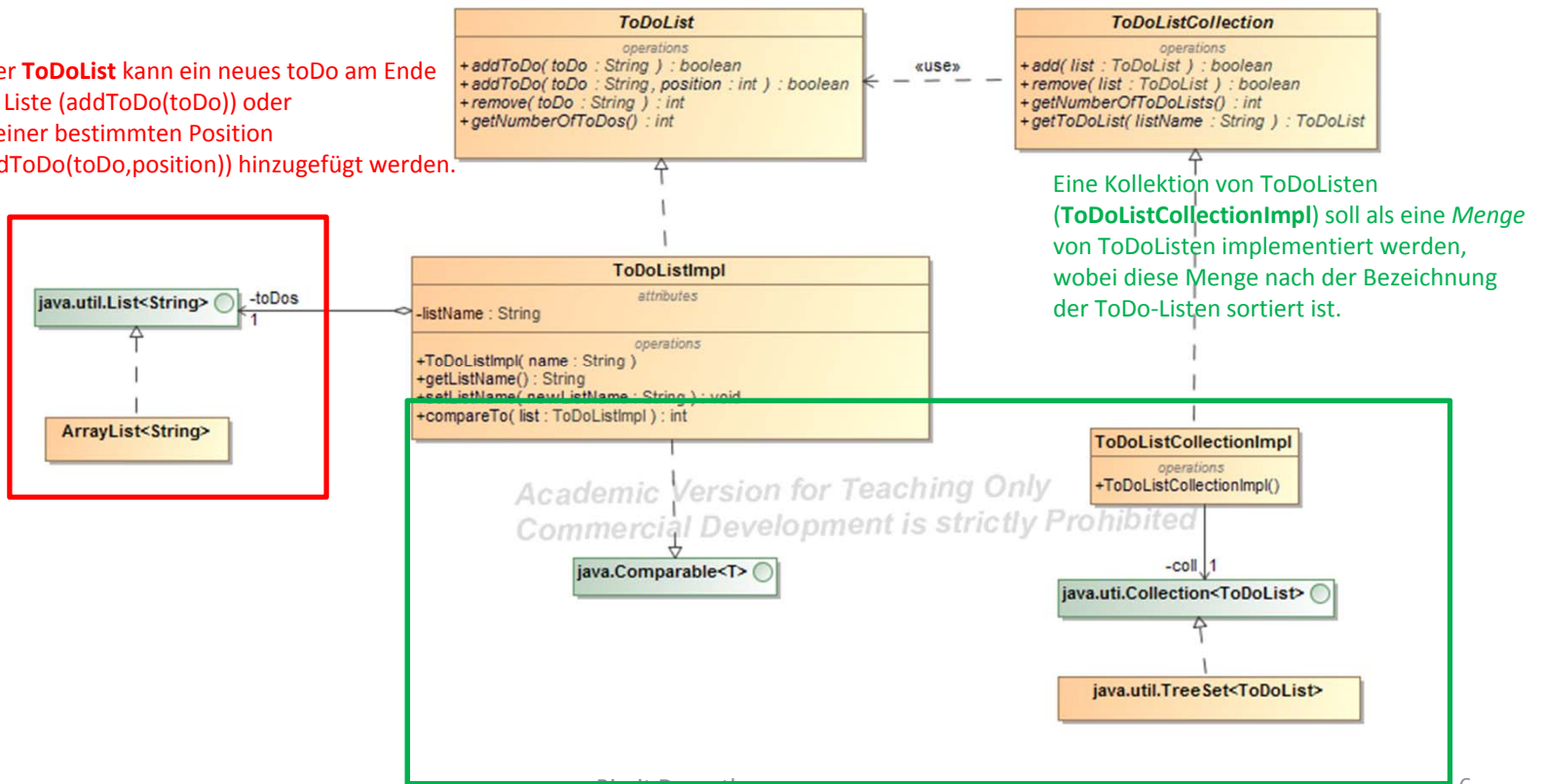
Über die Methode `getToDoList()` soll über den Listennamen (`listname`) auf eine konkrete **ToDoListe** zugegriffen werden.

ToDo-Listen (dUML)

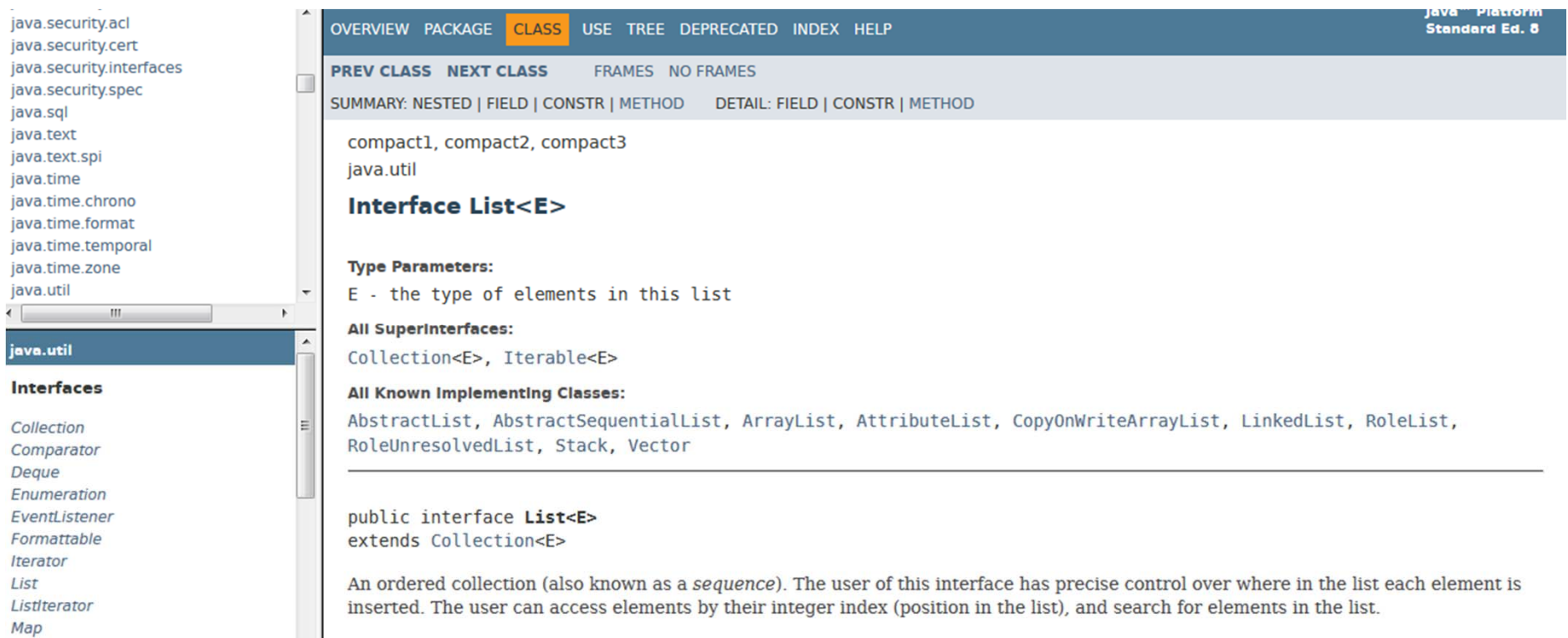


ToDo-Listen (jUML) [weitere Verfeinerung auf Java]

Einer **ToDoList** kann ein neues **todo** am Ende der Liste (`addToDo(todo)`) oder an einer bestimmten Position (`addToDo(todo,position)`) hinzugefügt werden.



Nutze javadoc



The screenshot shows the javadoc page for the `java.util.List` interface. The left sidebar lists various Java packages, with `java.util` selected. The main content area displays the following information:

- Navigation tabs: OVERVIEW, PACKAGE, **CLASS**, USE, TREE, DEPRECATED, INDEX, HELP
- Sub-navigation: **PREV CLASS**, **NEXT CLASS**, FRAMES, NO FRAMES
- Summary: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
- Compact links: compact1, compact2, compact3
- Package: java.util
- Interface List<E>**
- Type Parameters:**
E - the type of elements in this list
- All SuperInterfaces:**
Collection<E>, Iterable<E>
- All Known Implementing Classes:**
AbstractList, AbstractSequentialList, ArrayList, AttributeList, CopyOnWriteArrayList, LinkedList, RoleList, RoleUnresolvedList, Stack, Vector
- Signature:
`public interface List<E>`
`extends Collection<E>`
- Description:
An ordered collection (also known as a *sequence*). The user of this interface has precise control over where in the list each element is inserted. The user can access elements by their integer index (position in the list), and search for elements in the list.

java.util.List<E> (Auszug)

[<https://docs.oracle.com/javase/8/docs/api/java/util/List.html>]

```
public interface List<E> {  
  
    public boolean add (E o)  
    public boolean add (int index, E o)  
    public boolean remove (Object o)  
    public E remove (int index)  
    public void clear()  
    public boolean isEmpty()  
    public boolean contains (Object o)  
    public int size()  
    ...  
    public boolean equals (Object o)  
    public E get(int index)  
    ...  
    public Iterator<E> iterator()  
}
```


java.util.Set<E> (Auszug)

[<https://docs.oracle.com/javase/8/docs/api/java/util/Set.html>]

```
public interface Set<E> {  
  
    public boolean add (E o);  
    public boolean addAll(Collection<? extends E> c)  
    public boolean remove (Object o);  
    public void clear();  
    public boolean isEmpty();  
    public boolean contains (Object o);  
    public int size();  
    public boolean equals (Object o);  
    public int hashCode();  
    ...  
    public Iterator<E> iterator();  
}
```

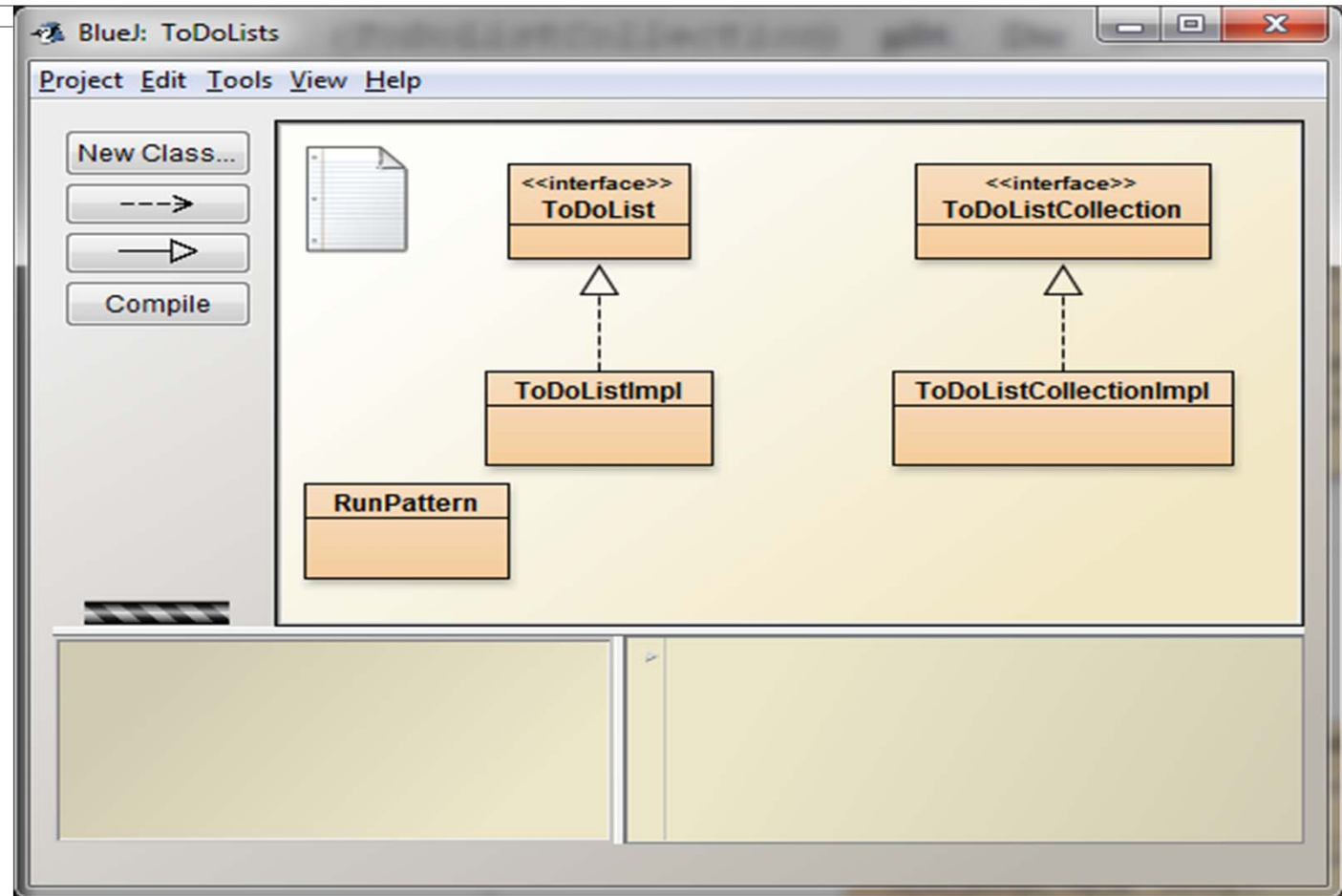
java.lang.Comparable<T>

<https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html>

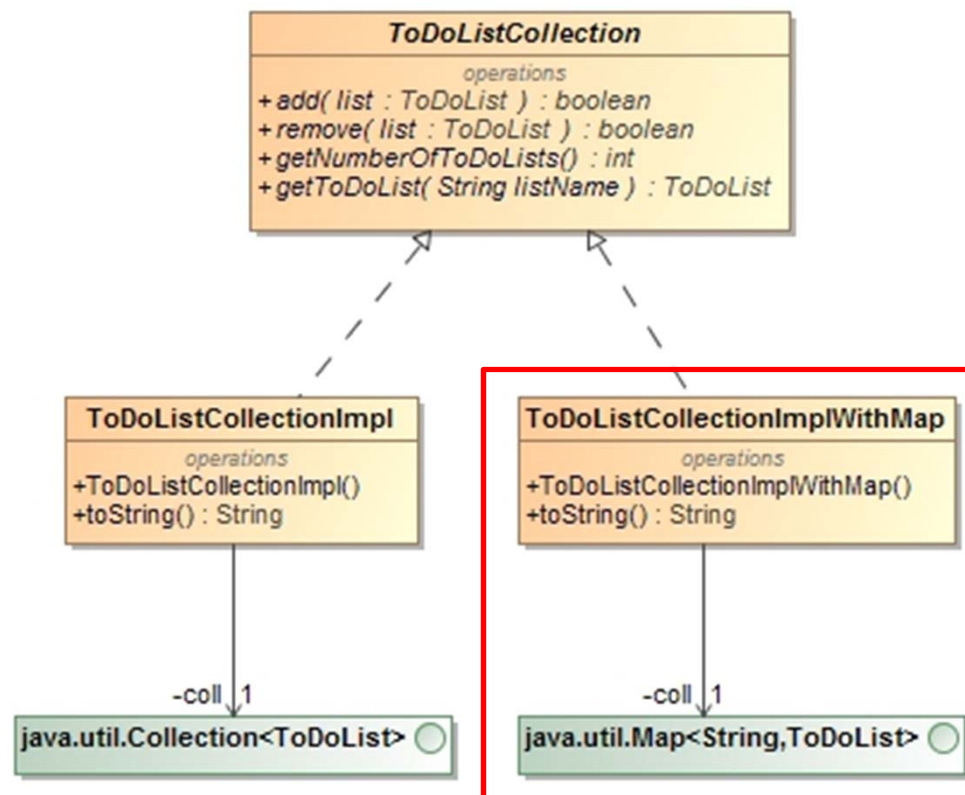
```
public interface Comparable<T> {  
  
    public int compareTo (T o);  
}
```

- Interface für den Test der Ordnung auf Elementen
- T ist der Typ des Objektes, mit dem verglichen wird
- Resultat kleiner/gleich/größer 0:
 - "this" kleiner/gleich/größer als Objekt o
- Standarddatentypen (z.B. String) implementieren Comparable

ToDo-Listen in BlueJ



Erweiterung von ToDo-Listen (jUML) (gegenüber Klausuraufgabe)



Weitere ToDoCollection-
Implementierung:
über den Namen der ToDoList
soll direkt auf die ToDoList
zugegriffen werden.

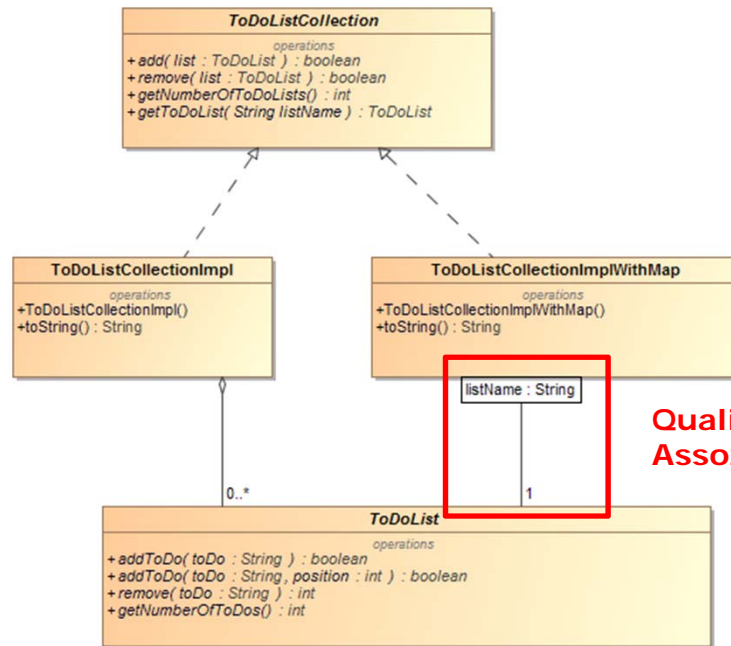
java.util.Map (Auszug)

[<https://docs.oracle.com/javase/8/docs/api/java/util/Map.html>]

```
public interface Map<K,V> {  
    ...  
    public boolean containsKey (Object key);  
    public boolean containsValue (Object value);  
    public V get (Object key);  
    public V put (K key, V value);  
    public V remove (Object key);  
    public int size();  
    public Set<K> keySet();  
    public Collection<V> values();  
    ...  
}
```

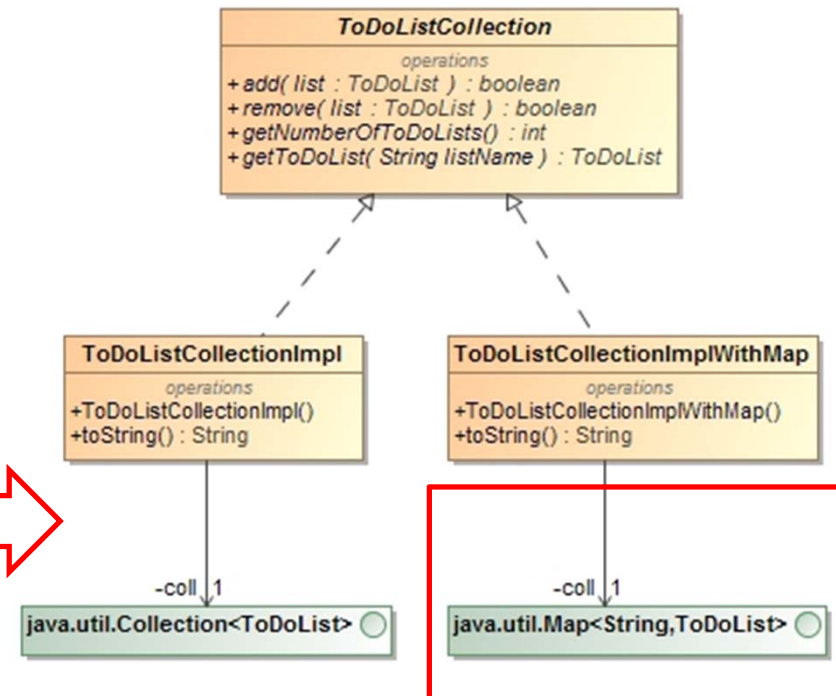
ToDoListCollection Implementierungen

dUML

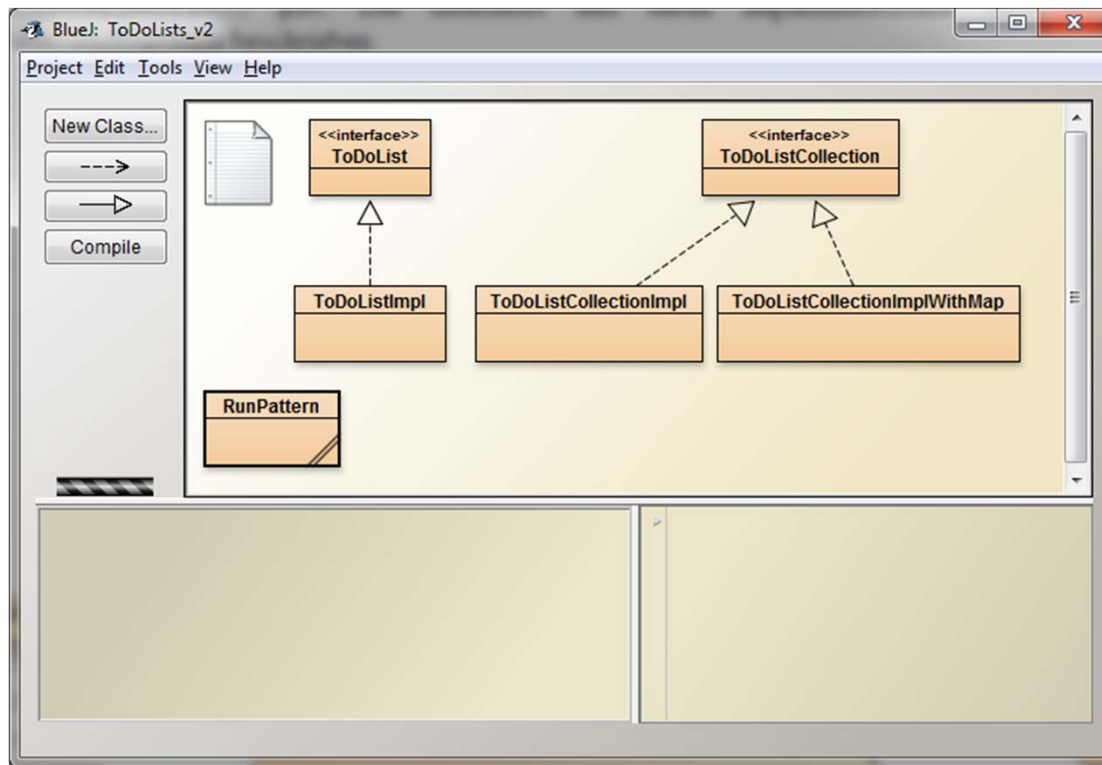


Qualifizierte
Assoziation

jUML



ToDo-Listen erweitert (v2) in BlueJ



Nutzung des AMCS

(Auditorium Mobile Classroom Service)



<https://amcs.website>

Einloggen/Registrieren mit beliebigem
Pseudonym
Passwort

Kurs Softwaretechnologie
PIN: SWT2019

Ende