**JAC444 - Lecture 9**

Java Collections Segment 2 - Set

**The Set<E> Interface**

**Set** is a collection that **cannot contain duplicate elements**

**Set** interface inherits from **Collection** adds the restrictions to eliminate the duplicate elopements

Implementations: **HashSet** - stores its elements in a hash table and is the best-performing implementation

**TreeSet** - The elements are ordered using their natural ordering

**LinkedHashSet** – Hash table with linked list running through it

**The Set<E> Interface**

|  |
| --- |
| **public interface Set<E> {**  **// Group 1 int size();**  Basic  **boolean isEmpty();**  **boolean contains(E element);** Operations  **boolean add(E element); // Optional boolean remove(E element); // Optional Iterator<E> iterator();**  **2**  **1**  **3**  **// Group 2 boolean containsAll(Collection c);**  Bulk  **boolean addAll(Collection c); // Optional**  **boolean removeAll(Collection c); // Optional** Operations  **boolean retainAll(Collection c); // Optional void clear(); // Optional**  **// Group 3**  **Object[] toArray();**Array  **Object[] toArray(Object a[]);**  **}** Operations |

**Basic Operations**

import java.util.\*; public class **FindDups** {

public static void main(String args[]) { **Set<String> s = new HashSet<>();**  for (int i=0; i < args.length; i++) if (!s.add(args[i])) System.out.println("Duplicate: "+args[i]);

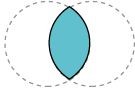
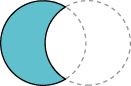
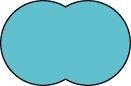
System.out.println(s.size()+ " distinct : " + s);

}

}

**Bulk Operations**

**Set union = new HashSet(s1); union.addAll(s2);**



**Union**

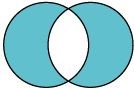
**Intersection**

**Difference**

**Set intersection = new HashSet(s1); intersection.retainAll(s2);**

**Set difference = new HashSet(s1); difference.removeAll(s2);**

**Set symmetricDiff = new HashSet(s1); symmetricDiff.addAll(s2);**



**Symmetric difference**

**Set tmp = new HashSet(s1);**

**tmp.retainAll(s2);**

**symmetricDiff.removeAll(tmp);**

**Duplicate Words Sample**

import java.util.\*;

public class **FindDuplicateWords** { public static void main(String args[]) { Set<String> uniques = new **HashSet<>()**;

Set<String> dups = new **HashSet<>()**;

for (int i=0; i < args.length; i++) if (**!uniques.add(args[i]**)) **dups.add(args[i]);**  **uniques.removeAll(dups);** //Destructive set-difference

System.out.println("Unique: " + **uniques**);

System.out.println("Duplicate: " + **dups**);

}

}

**Set Implementations**

1. HashSet

Does not maintain any order of its elements

1. TreeSet

Sorts elements in ascending order

1. LinkedHashSet

Maintains the insertion order