

CMSC 132: Object-Oriented Programming II

Sets and Maps



Department of Computer Science
University of Maryland, College Park

How Do Collections Work in Java?

- Elements are NOT copied when inserted
- Collection contains references, not objects
- Finding matching element is based on equals()
- To build a collection for a class
 - Need to define your own equals(Object) method
 - Default equals() uses reference comparison
 - Just like a == b
 - a and b are only equal if they refer to the same object

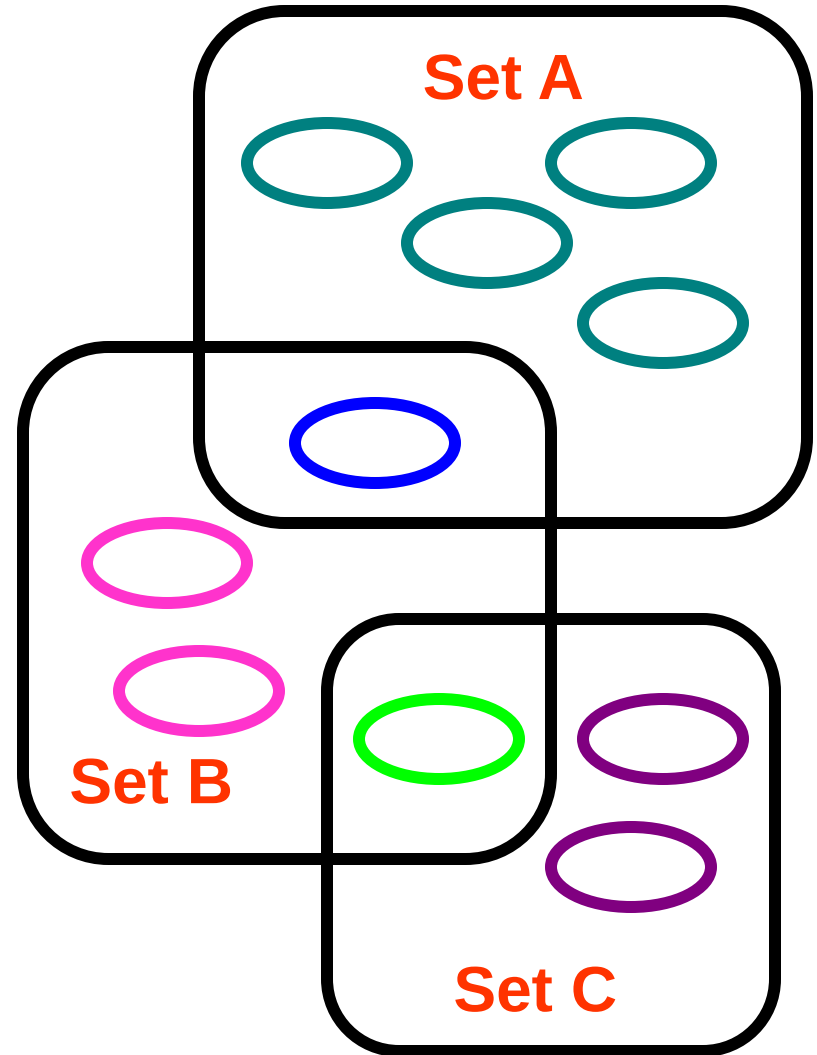
Sets

■ Properties

- Collection of elements without duplicates
- No ordering (i.e., no front or back)
- Order in which elements added doesn't matter

■ Implementation goal

- Offer the ability to find / remove element quickly
- Without searching through all elements



Set Concrete Classes

■ HashSet

- Uses Hash Table
- Elements must implement hashCode() method

■ LinkedHashSet

- Uses Hash Table AND Doubly Linked List
- Elements can be retrieved in order of insertion
- Elements must implement hashCode() method

■ TreeSet

- Elements must be comparable
 - Implement Comparable or provide Comparator
- Guarantees elements in set are sorted

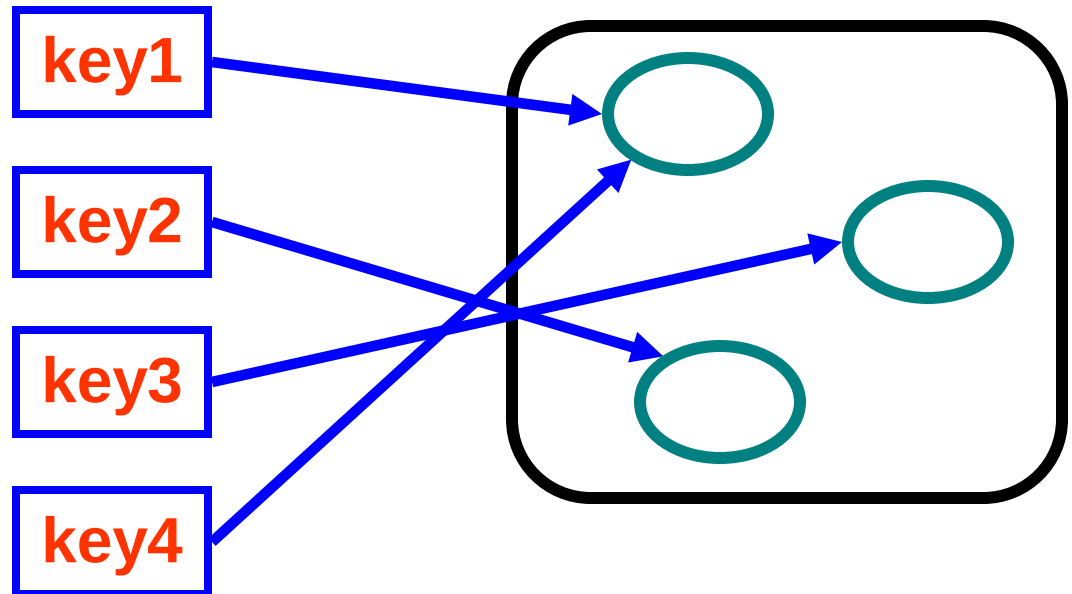
Sets Example

Coding Example about Sets...

Map Definition

■ Map

- Unordered collection of **keys**
- For each key, an associated **value**
- Can use key to retrieve value



Map Properties

■ Map “keys” & map “values”

■ Aliasing

- Each key is associated with ONE value
- But same value may be referred to by multiple keys

■ Can also treat list of “keys” & list of “values” as collections

- Access using `keySet( )`, `values( )`

■ Keys & values may be of complex type

- `Map<Object Type1, Any Object Type2>`
- Including other collections, maps, etc...

Map Concrete Classes

■ **HashMap**

- Keys must implement hashCode() method

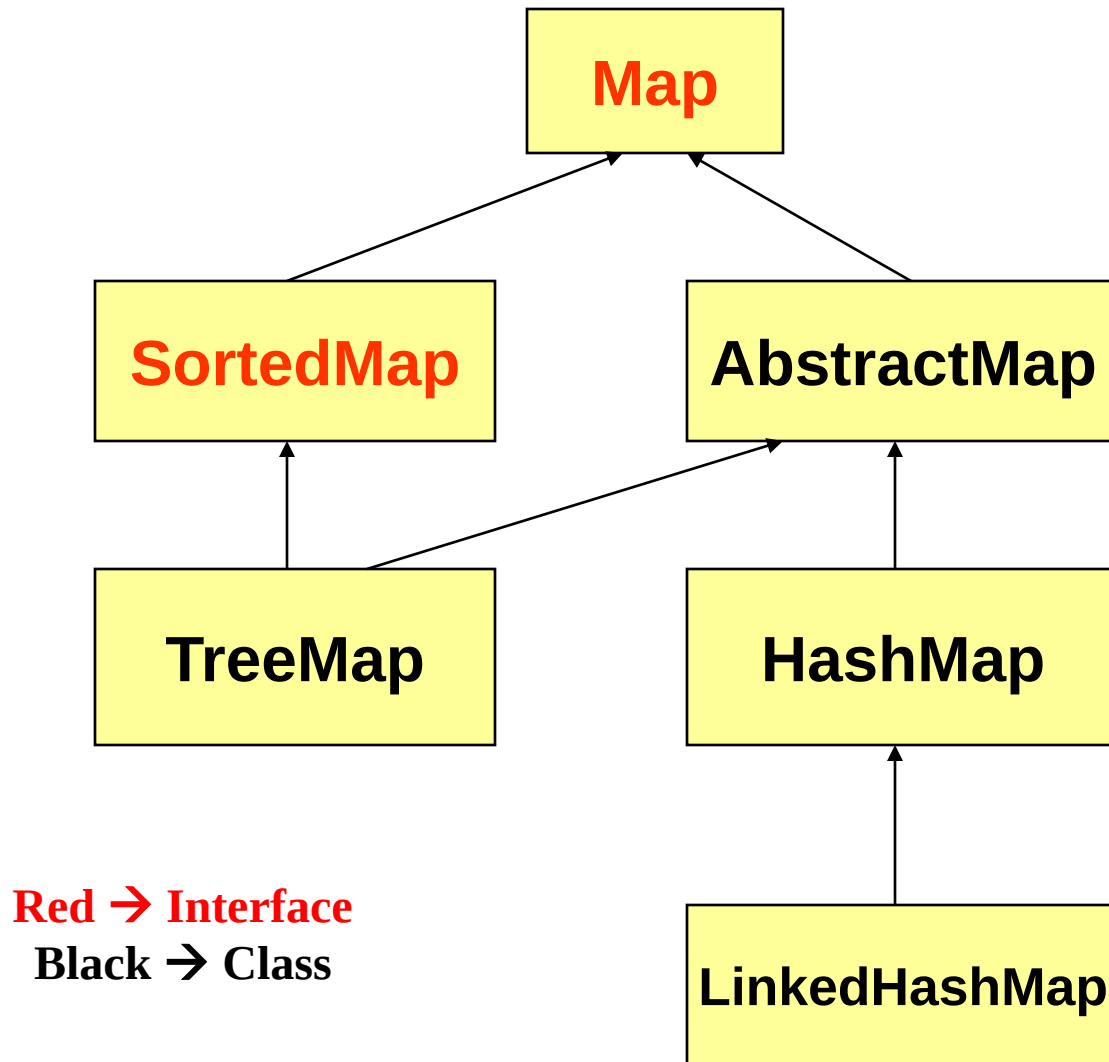
■ **LinkedHashMap**

- HashMap supporting ordering of elements
- Keys/Values can be retrieved in order of insertion
- Keys must implement hashCode() method

■ **TreeMap**

- Keys must be comparable
 - Implement Comparable or provide Comparator
- Keys/Values can be retrieved in sorted order of Keys

Map Hierarchy



Map Interface Methods

■ Methods

- `void put(K key, V value)` // inserts element
- `V get(Object key)` // returns element
- `V remove(Object key)` // removes element
- `int size()` // key-value mappings
- `void clear()` // clears the map
- `boolean containsKey(Object key)` // looks for key
- `boolean containsValue(Object value)` // looks for value
- `boolean isEmpty()` // empty map?
- `Set<K> keySet( )` // entire set of keys
- `Collection<V> values()` // values in the map

Coding Examples

- See the package called “maps” on your CVS repository.