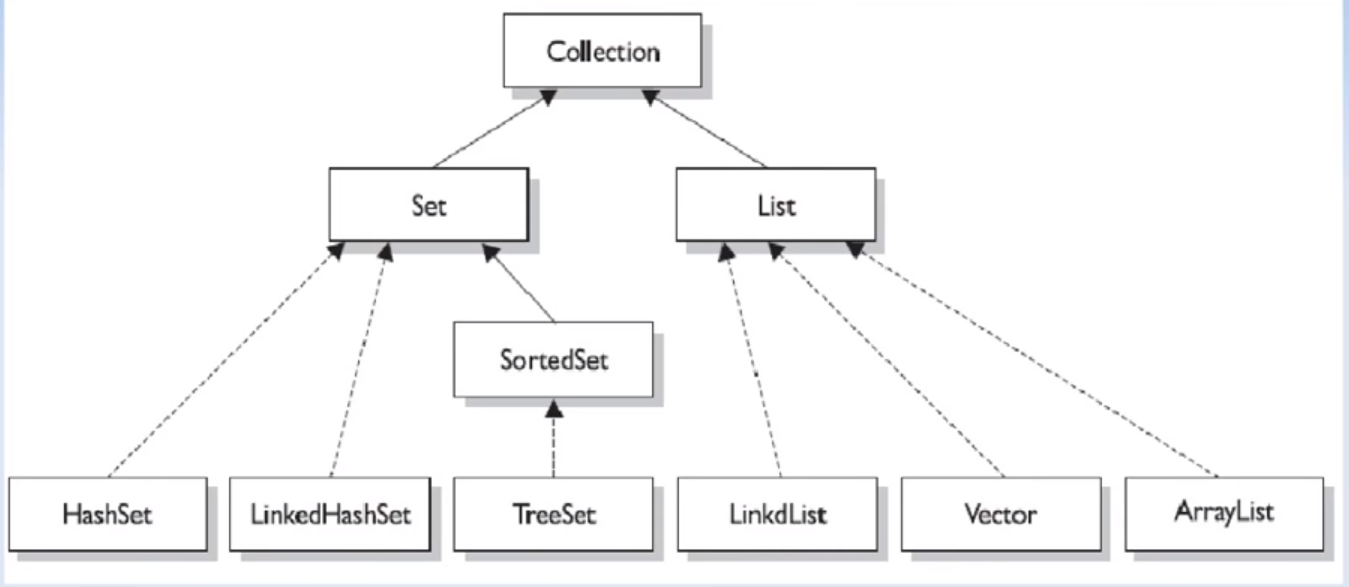
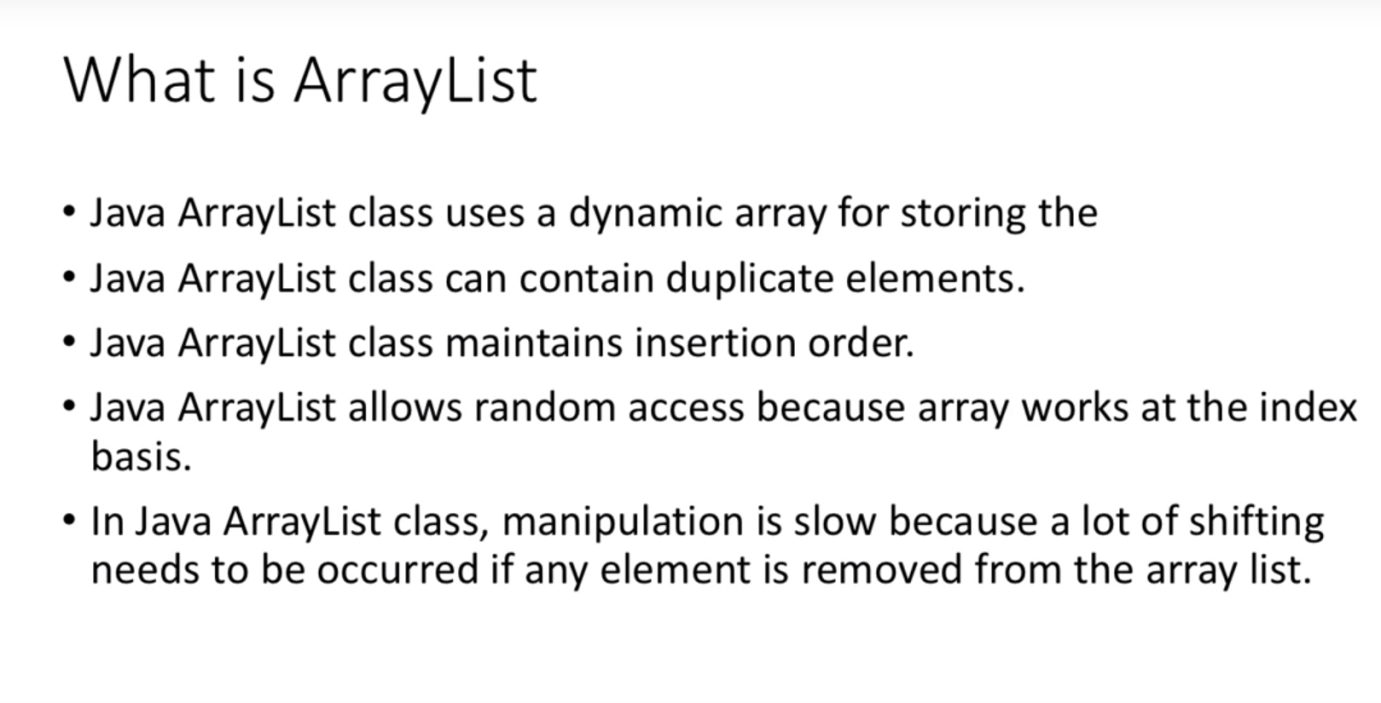
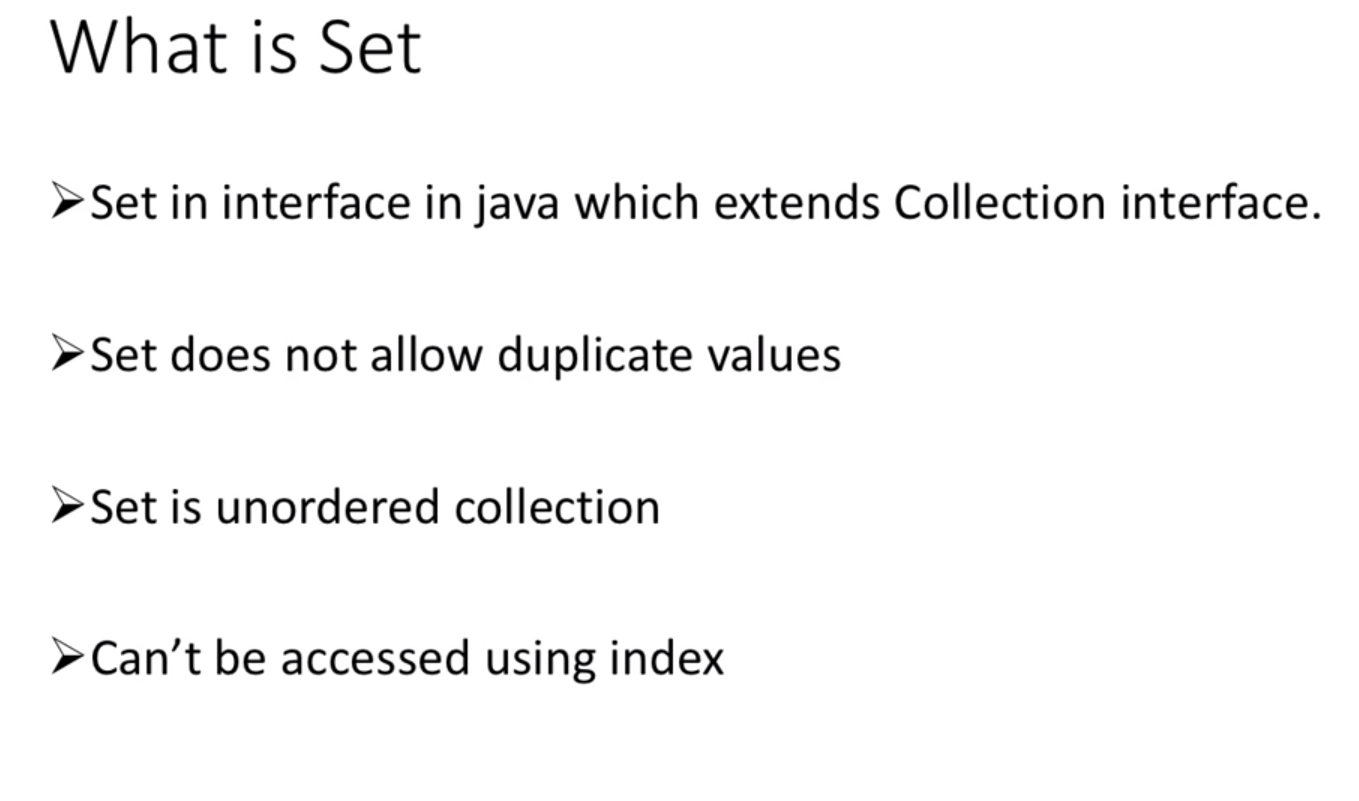
**Java Collections**

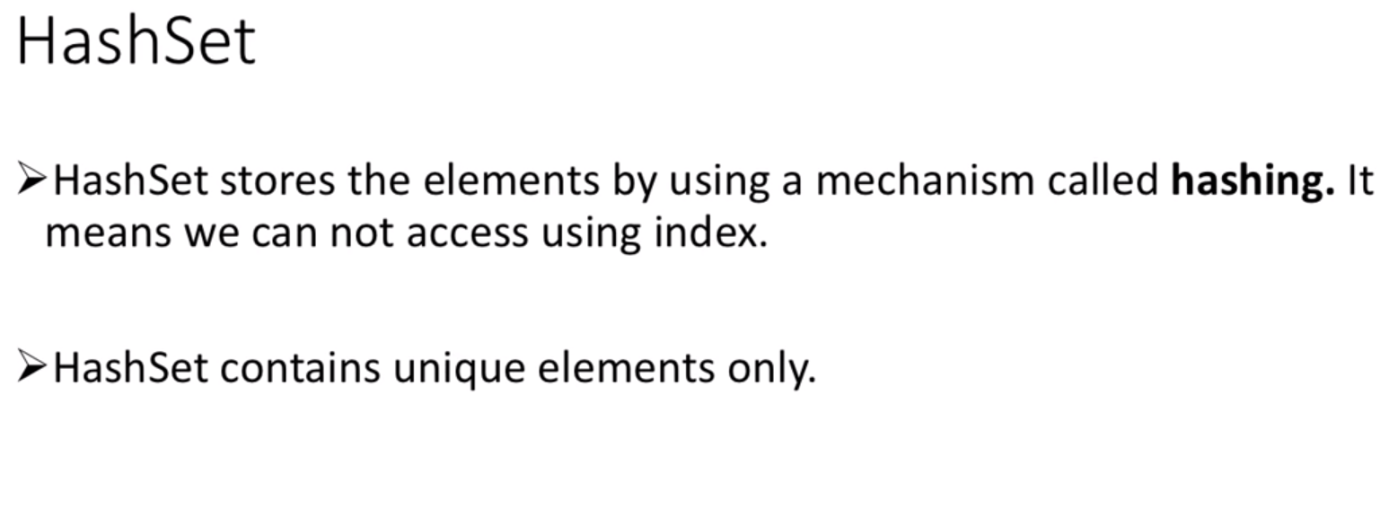
It is a Dynamic Data Structure / framework which provide architecture to store and manipulating group of objects.

Searching, insert, delete, manipulating etc.





****

* 

**Difference between Array List and Linked List:**

**Array List:** It uses dynamic array to store elements internally.

**Linked List:** It uses doubly linked list to store data internally.

**Queue:**

**Priority Queue:**

It is used to retrieve data in sorted order.

Peek: retrieve head value without removing it.

Poll: retrieve head value without removing it.

Example: Input Data: 1 2 10 9 8

Peek: 1 1 1 1 1

Poll: 1 2 8 9 10

**Deque:** Double Ended Queue

**Vector:**

It is synchronised as it is used in multithreaded operations.

**Java Map Interface**

* A map contains values on the basis of key i.e. key and value pair. Each key and value pair is known as an entry. Map is useful if you have to search, update or delete elements **on the basis of key**.
* **Map contains only unique keys.**

Example: Input Data: a=100 a=101 b=100

Output Data: a=101 b=100

**Java Map. Entry interface:**

Entry is the sub interface of Map.It provides methods to get key and value.

**for** ( Map.Entry m: map.entrySet() )

**HashMap:** Maintains no insertion order and can have one null key.

**LinkedHashMap:** Maintains insertion order and have null elements.

**TreeMap** : Maintains ascending order and cannot have null keys.

**Hash Table:**

It is synchronised and cannot have null keys and values.

**Enum:**

Enumerations serve the purpose of **representing a group of named constants** in a programming language.

enum Color {RED, BLACK, BLUE;}

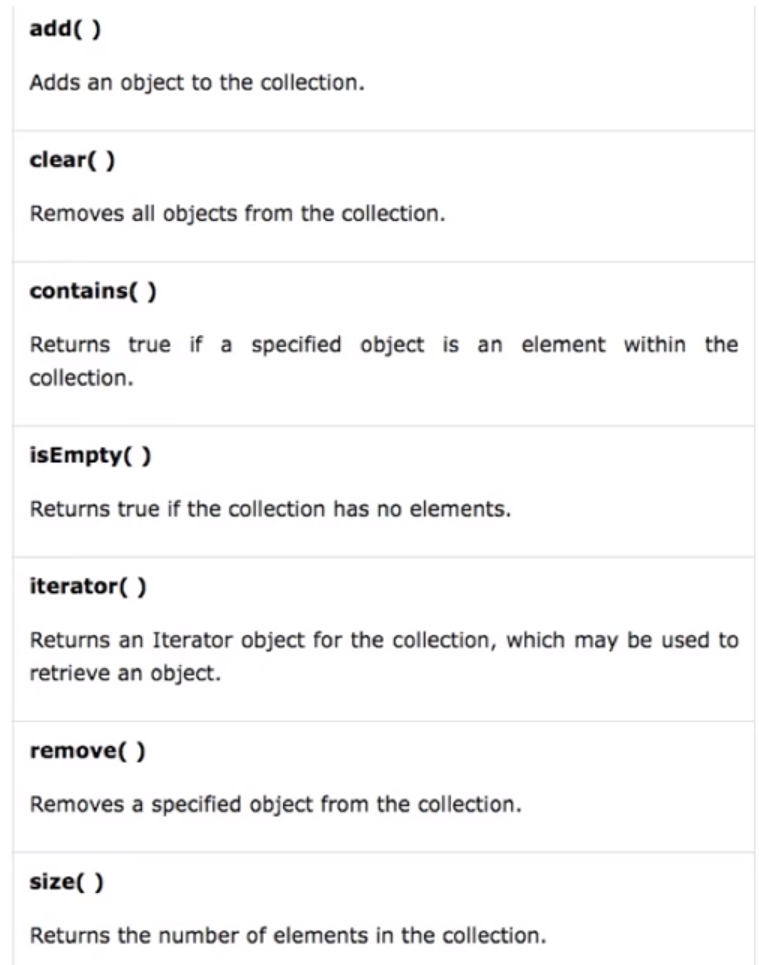
**EnumMap and EnumSet**

**Java Comparable Interface**

We can sort the elements on based on single data member only. For example it may be roll no, name, age or anything else.

**Java Comparator Interface**

We can sort the elements on based on any data member.

****