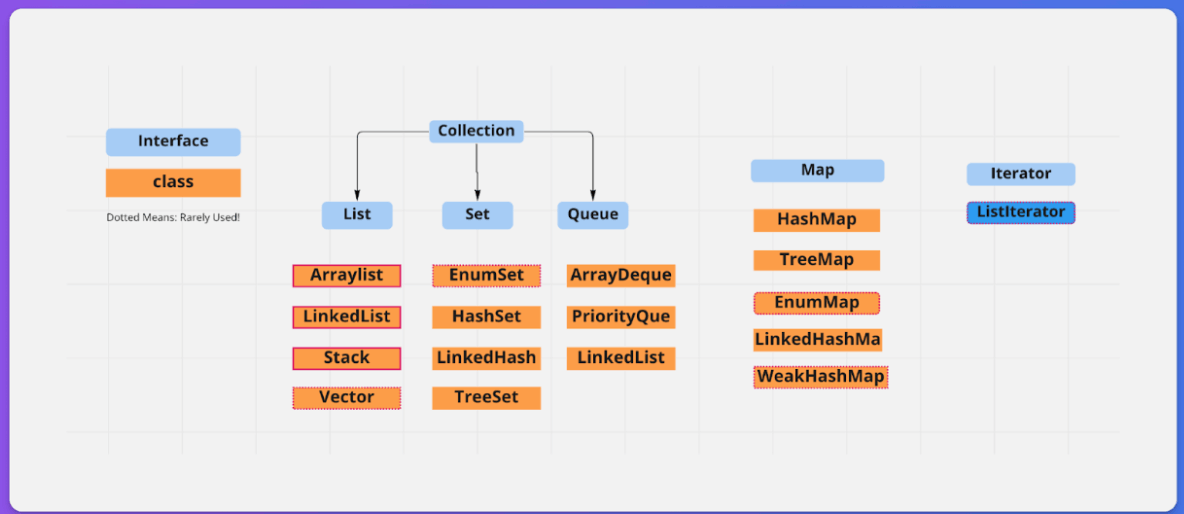
What the problem with the Arrays? and Why Collection Framework (CF) is introduced.

Problem with Arrays

1. // Size -> Fixed Size Problem in array.
2. // There are not in built functions available in the arrays.
3. // sorting, updating, delete, increasing size, decrease size.
4. array can store one type of data type.

**Collection Framework**



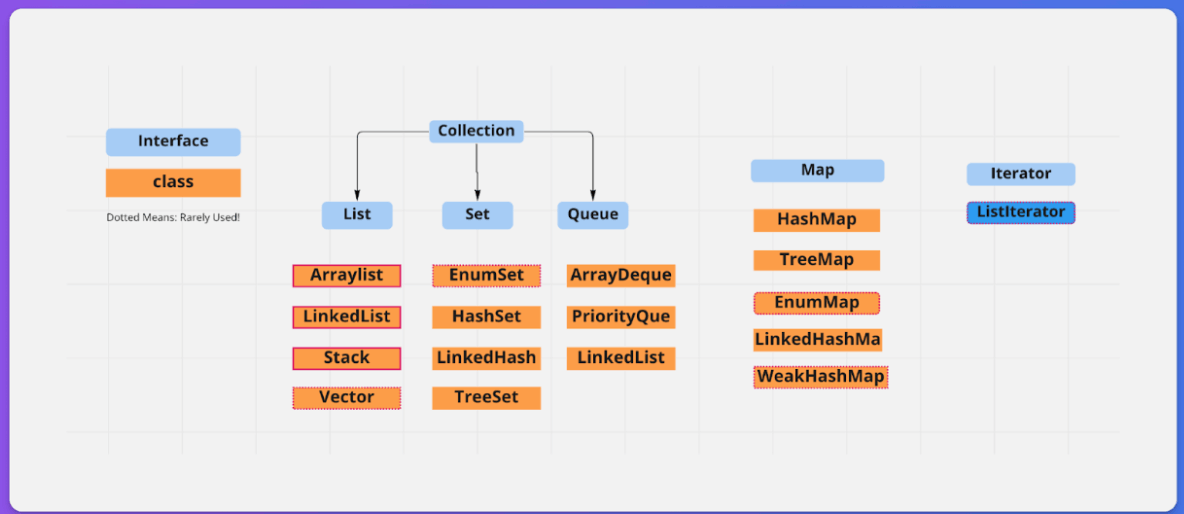
**List – Keep Duplicates, one or more Null values allowed, Order – Insertion Order**

**Set – Don’t Keep Duplicates**

**Queue – FIFO (First in First Out)**

**Collection Framework(+File Handling)**

* Data Structure algorithm.
* Focus on Main **Business Logic** rather than Low Level Logics.
* import java.util.Collection;
* List, Set, Queue, Map -> Add, read, delete, updating -> (CRUD) operation ->



JDK 1.2 - 1997

**Legacy Classes**

1. // Vector - functions
2. // Stack
3. **// Properties**
4. **// Hash table**
5. // Dictionary

JDK 23 - Advance Collection Class that we can use.

Collection(I) (GGF) -> List, Set, Queue (I) -> Respective class

1. List -> Balaji, Ramal, Snehal, Ramal.... I want to keep the duplicates
2. Set -> Remove the Duplicates.
3. Queue -> FIFO

**List (I)**

1. List of Items (They can be duplicate).
2. lists typically allow duplicate elements.

* Shopping List -> Morning -> milk, bread, butter and paneer
* 1,2,3,4,5,5,6,7 - Duplicate is allowed (lists typically allow duplicate elements)
* 2 Kachori, 4 paranths, 4 Teas.
* 1, Pramod, true, hello, Gaurav, Apple, car, Tesla
* Fruits - Oragne, Apple, Gauva....

List

List -> ArrayList( Vector), LinkedList, Stack.

* Ordered Collection
* Control over the insertion
* List allows Duplicate.

Way to Initialise List (I)

* List fruits = List.of("orange","apple"); // Static Of - hard functions.
  + **ArrayList** - Underline Array - Get element is easy, **inserting, Delete is Costly.**
  + **LinkedList** - LinkedList ( Doubly LinkedList) - How elements stored) - Insert, Deletion is easy, finding element is difficult.
  + Vector -> Array - Old Legacy
  + Stack -> LIFO (book of stack)

Vector

* Old legacy
* Simirlar to AL, only this is it is **Thread Safe.**
* All the operations in the Vector are **synchronized**, they are thread safe. ( All the operation are done by one by one, by each thread).

**ArraysList**

Behind the scene it is **growable array**(dynamic array).

1. Underline Storage Array - How they store the elements
2. ArrayList, Vector -> They are stored like similar to Arrays -> Continuous memory - 0, 1, 2 - [][][][][].
3. Dynamic Size (the moment you read to the n-1, it will automatically increase the sizs, 10 - AL, 9th -> 2 -> 20 Size, 19th -> 40

// Selenium - -> Store and **Insertion and Deletion** - DS - LL

// API Testing - STORE, **access it fast**. -> AL

API Testing -> access FAST -> ArrayList -> , Add element, delete element operation - more - LL

**ArrayList ->**

* Insertion - **Heavy**
* **Searching and Updation - Light**
* Delete -> **Heavy**

**LinkedList**

* Insertion - **Light**
* **Searching and Updation - Heavy**
* Delete -> **Light**

**LinkedList**

* ​​LinkedList allows duplicates and internally implements a **doubly-linked list data structure.**
* LinkedList elements will be stored internally using **node representation.**
* **It is fast to insert or delete the elements**, but s**low for accessing the elements.**

**Vector**

* Vector allows duplicates, and it is implemented as a resizable array.
* Vector is a **legacy class** and elements will be stored internally using **indexing notation.**
* Vector methods are **synchronized**, so vector objects can’t be accessed by multiple threads at a time.(Slow).

**Disclaimer**

* Vector, Stack - Legacy Classes -> 95% of time we are not going to use it in automation.
* **ArrayList** - 95% we are going to use.

**Set**

* Collection Framework ( Collection I)
* Doesn't allow Duplicate

Set hs = new HashSet();// Hashing mechanism to store the element, no order

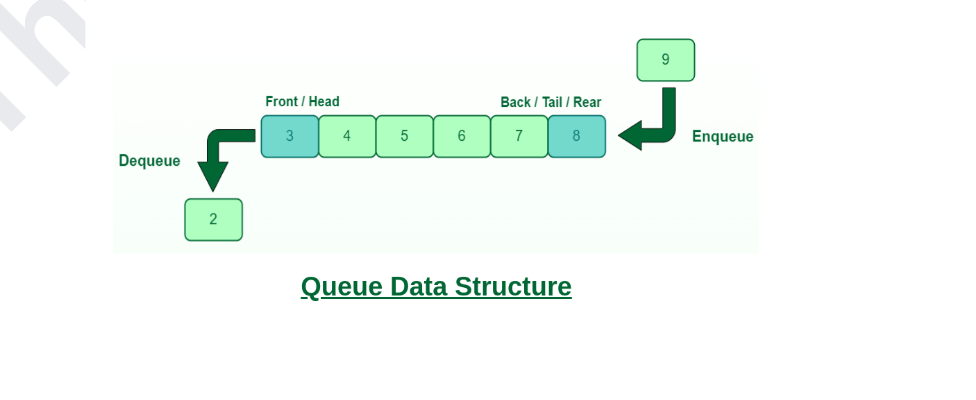
**Is Set an interface or a class in Java?**

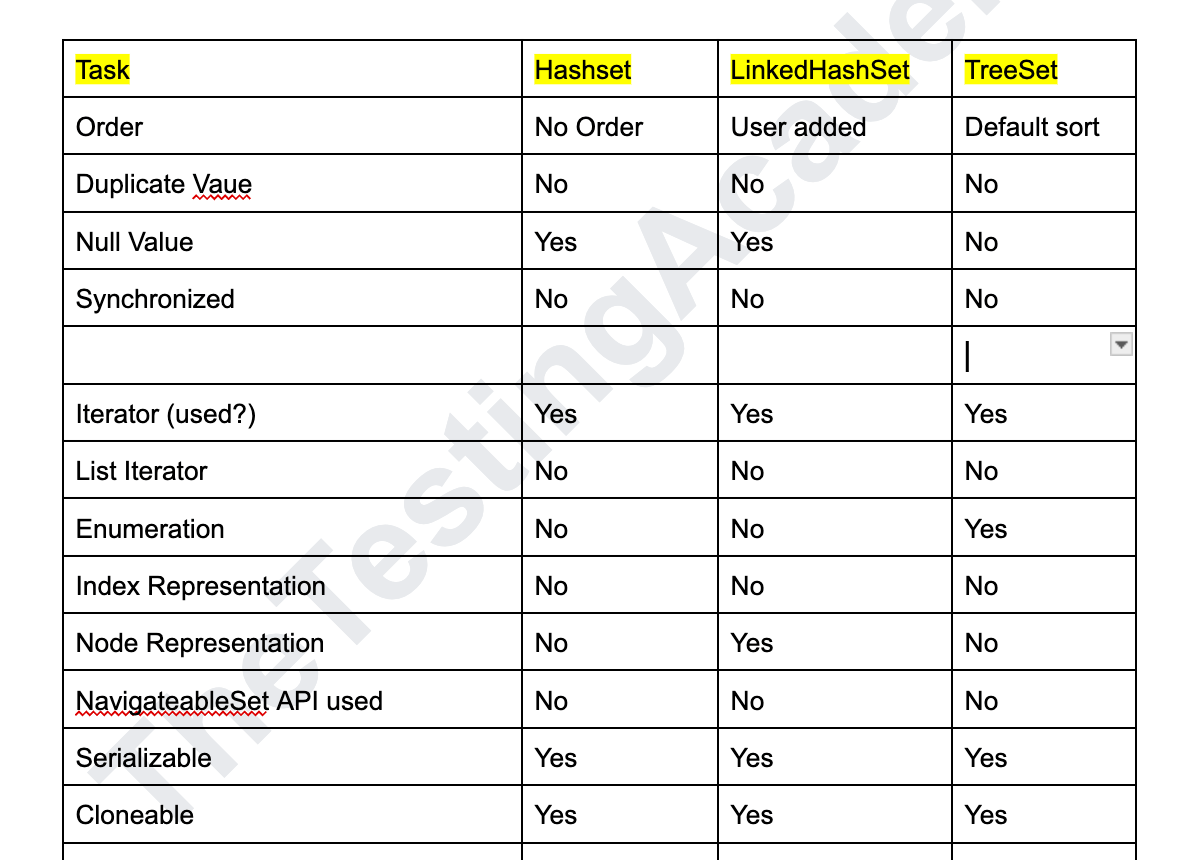
interface

1. **HashSet**
2. LinkedHashSet
3. TreeSet
4. EnumSet( Rarely)

**Queue**

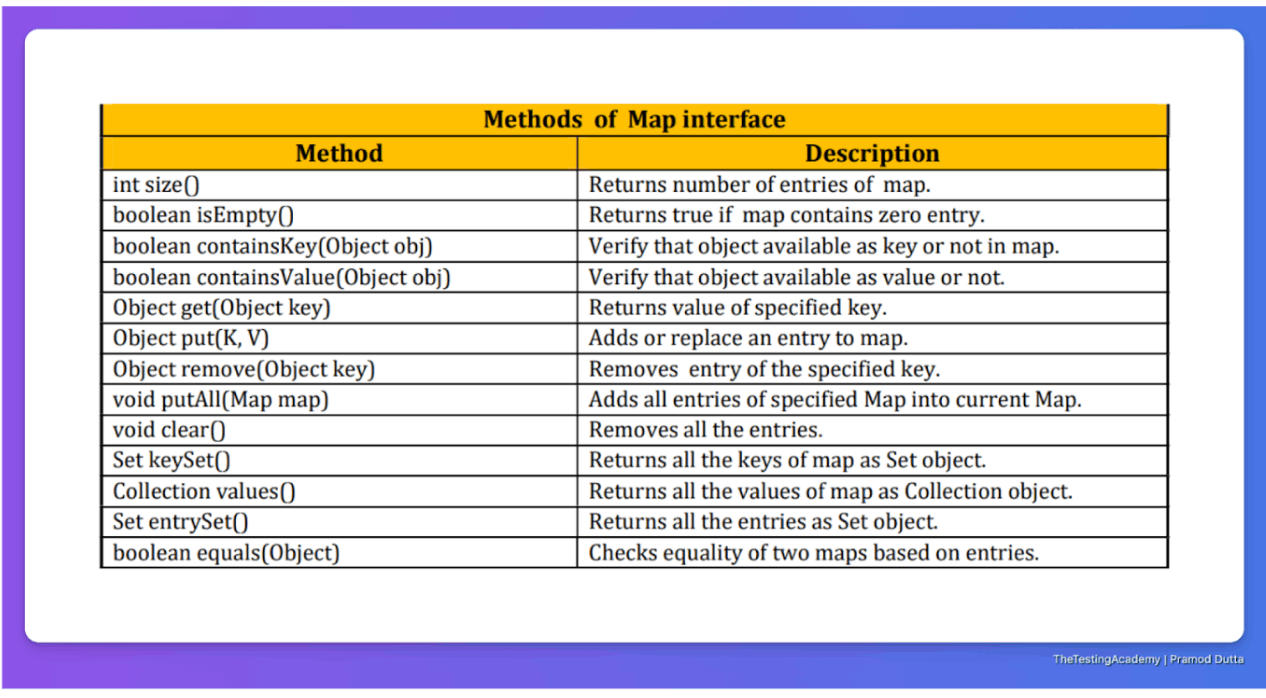
* Added interface in Java 5.
* **PriorityQueue** is one of the commonly used subclasses of Queue interface.
* PriorityQueue doesn’t allow null values.
* You can add only comparable objects to PriorityQueue.





**Map**

* Map is an interface available in **java.util** package and introduced in the Collection Framework API.
* a key and a value.
* The **key is always unique** in the Map.
* A Map **cannot contain duplicate keys** and each key can map to at most one value.
* Some implementations allow null key and null values like the HashMap and LinkedHashMap, but some do not like the TreeMap.
* Map interface has following concrete sub classes:
  + o **HashMap**
  + o LinkedHashMap
  + o TreeMap
  + o Hashtable (LEGACY)

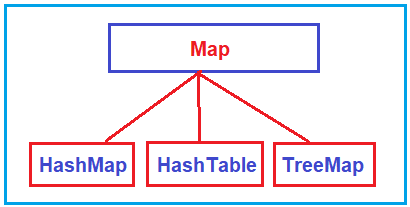


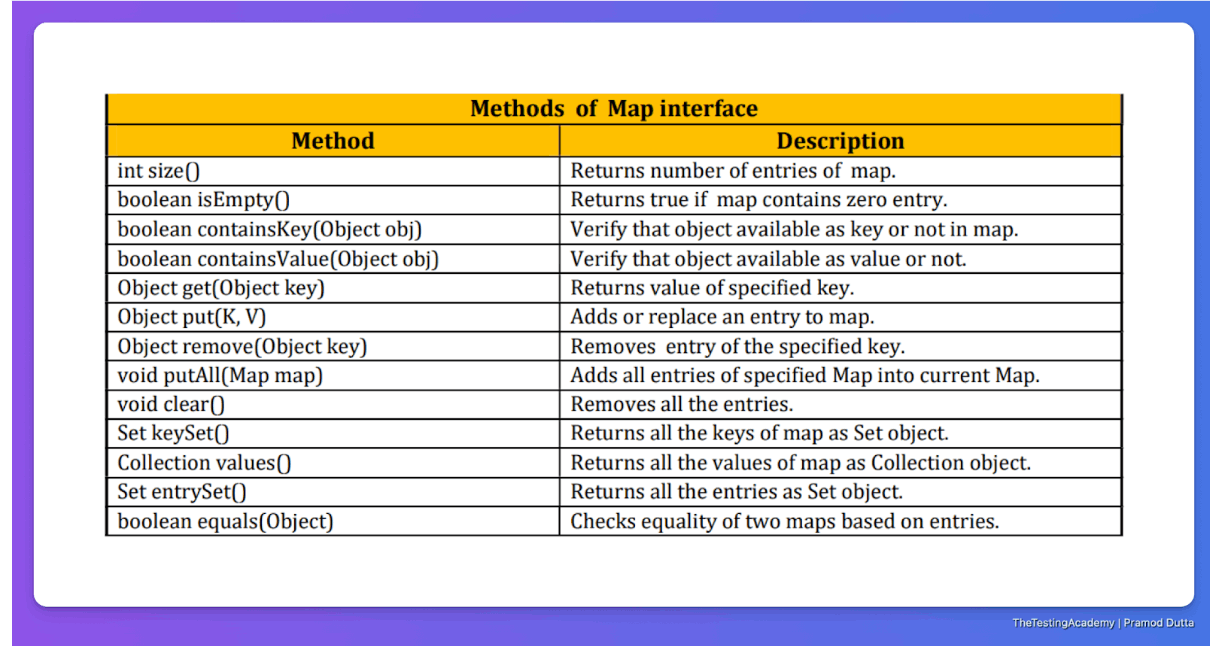
**HashMap** is a part of Java’s collection since Java 1.2. It provides the basic implementation of the Map interface of Java. It stores the data in (Key, Value) pairs.

No Duplicate Key on HashMap

**LinkedHashMap** is just like HashMap with an additional feature of **maintaining an order** of elements inserted into it.

**The TreeMap** in Java is used to implement the Map interface and NavigableMap along with the Abstract Class. The map is sorted according to the **natural ordering of its keys.**





Very Important.

