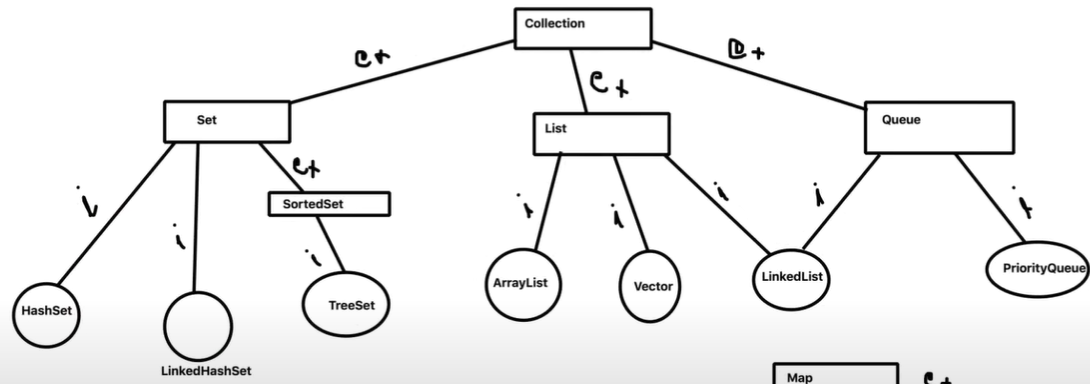
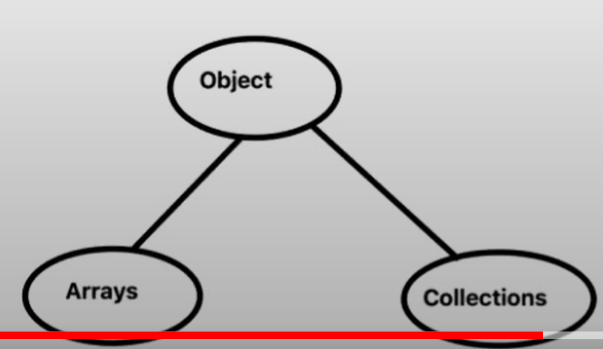
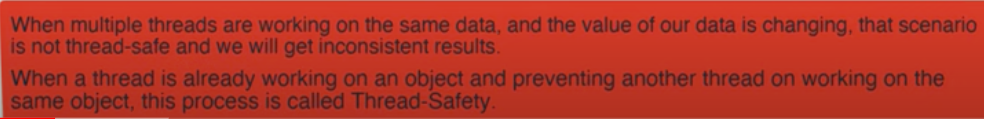
**#2 - Difference between List/Set/Map Based Classes (Ordering/Access/Key-Value/Duplicate/Thread-Safe)**





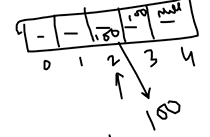
I have explained the difference between different collection classes in Java on the basis of : Ordering, Random Access, Key-Value, Duplicate Elements, Null Key and Thread Safety.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Collection** | **Ordering** | **Random Access** | **Key-Value** | **Duplicate Elements** | **Null Key** | **Thread Safety** |  |
| **ArrayList** | **yes** | **yes** | **No** | **yes** | **yes** | **No** |  |
| **LinkedList** | **yes** | **No** | **NO** | **yes** | **yes** | **No** |  |
| **HashSet** | **No** | **No** | **No** | **No** | **yes** | **No** |  |
| **TreeSet** | **yes** | **No** | **No** | **No** | **No** | **No** |  |
| **HashMap** | **No** | **yes** | **yes** | **No** | **yes** | **No** |  |
| **TreeMap** | **Yes** | **Yes** | **Yes** | **No** | **No** | **No** |  |
| **Vector** | **Yes** | **yes** | **No** | **Yes** | **Yes** | **Yes** |  |
| **HashTable** | **No** | **yes** | **yes** | **No** | **No** | **yes** |  |
| **Properties** | **No** | **Yes** | **Yes** | **No** | **No** | **yes** |  |
| **Stack** | **Yes** | **No** | **No** | **Yes** | **Yes** | **yes** |  |
| **CopyOnWriteArrayList** | **Yes** | **Yes** | **No** | **Yes** | **Yes** | **yes** |  |
| **ConCurrentHashMap** | **No** | **Yes** | **yes** | **No** | **No** | **yes** |  |
| **CopyOnWriteArraySet** | **No** | **No** | **No** | **No** | **Yes** | **yes** |  |



**ArrayList -**It maintenance the data in the of order and in the form of indexing. It means Whenever you are going to store the data in this **particular ArrayList objects. Random access is possible because it store the value in the form of indexing. [Random access, Ordering, duplicate values, null key] but it is not a thread safe and not key value format.**

Example – Let’s see this is a Dynamic ArrayList that we are going to create and it will store a data on the basic of indexing over here. Here Indexing like this 0,1,2,3, and then 4 and the data will be stored over here. So that’s why random access is possible.



So What is the value available on the Second position. So the value is available on the second position is 100.it’s will give you the hundred over here like that. That’s why we say that yaa random access is possible. I can access the 2nd position is 100.So Random access is possible.it store the value directly , there is no key.

Duplicate elements are allowed, It will store 100 n number of times duplicate values are allowed. Null element are also allowed. You can store null values also.

It is **not thread safety** for the threading point of view. It means we are really want to use this particular collection for multi-threading environment.then it will be a problem.

Example – Let’s see One data or object is available over here. And there are numbers of thread are available , like 3 threads are there and they are trying to data and try to access this particular object/ variable /or whatever. Let’s see this is not thread safe. It means the value is available and i=10; and

T1 20

Total= i+100

Then we are using this particular i to calculate something. Let’s see I am calculating this i to further calculation. Let’s see Assumed that total equal to /Total=i+100. Something like that I am doing. So This Mr Thread T1 is coming and then trying to access to this particular value and then then change value from i=10 to i=20.

So the latest value of the total will be changed in that case but by the time this thread is also coming over here, It means there is no restrictions that okay any other thread is coming over here and then trying to change value over here. It means the moment this T1 is trying to change the value from i=10 to i=20. By the time instance to getting the latest value this thread is also getting the previous value.

So the problem is that whatever the upcoming thread are coming to access this particular object, they always might get the old value. Because they are not sink to each other. That’s why this concept is called thread safety. This is not available in ArrayLIst.

So Any Multiple threads they can attack this particular object and then they can get the value. There is no guarantee that okay they will get the latest value or the updated value. that’s why this is problem with the arrayList.

Thread safety means if it is yes .

Let’s see Thread T1 is already working on this particular object and T2 is also there . So Java will not allowed to give access to T2 and T3 and other threads . Java will block this particular thread. Once the thread T1 ‘s work is done then only T2 come over here and will get the latest value. That is called thread safety.

==============================

**Linked List –** The same implementation is coming from the same family the list interface **and it[Both Arraylist and LinkedList] maintain the order to store the value** **. Random Access is not available because they store the value in the form of nodes. [ordering , duplicate element , null key] but not thread safe**

**The only Difference between Array List and LinkedList is that Random access is not possible. Other than that everything is exactly same. Duplicate value is allowed. They are not thread safety. There is no key value and it is also allowed null value.**

**In LinkedList** Random Access is not available because they store the value in the form of nodes.

In ArrayList Random access is possible because it store the value in the form of indexing.

**HashSet**

HashSet is coming from the set family.[ HasSet, LinkedHashSet and Tree set.]

Hashset does not maintain any order.it does not maintain any indexing, that’s why random access is not possible. It’s not like I am passing the index and getting the value just like an ArrayList or LinkedList.

Key value pair is not possible . Duplicate element also not possible .So In hashset I can not store duplicate element –

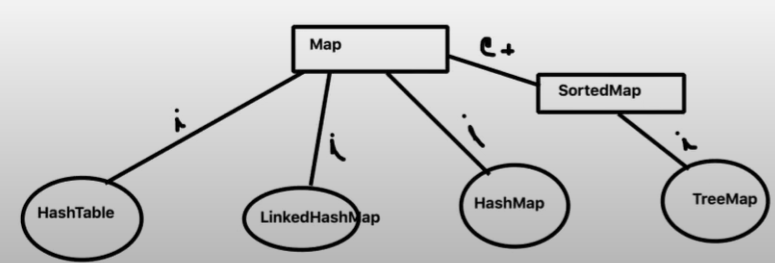
If you have seen that in selenium whenever we are using Driver.getWindoHandles, it’s returning set object over there. It means I can not store duplicate values over there. **We use HashSet to maintain the Unique values .**In Hashset null elements are allowed. And it is also not thread safe.

**TreeSet**

**TreeSet**- it maintains the order.and it maintains sorting order also.No random access, no key value format, duplicate elememts also not allowed, null element is not allowed and it is not a thread safe.

**HashMap**

**In Hashset Random access, Key value and null value.**



**Whenever we are talking about Map. Map means key and value pair format.**

**HashMap and TreeMap they are both key and value pair format.**

**HashMap does not maintain the order.it maintains on the basis of key value pair format.it pass the key and get the value. Ie this is the key and this is the value . it maintains the data In the key value format.**

**Duplicate element are not allowed in Hashmap but null elements are allowed. And it is not a thread safe and this is not synchronize .**

Tree map

It is coming from the sorted map and it maintains key value format in the form of sorting order.It maintans order, random access is also available, it maintains data in the basis of key value pair format.

**Vector**

It is very old coming from JDK 1.2. After JDK 1.2 then they introduced collection like LinledList and all those things.

Early we have vector, properties, Dictionary those things over there.

Vector is coming from the list family. Like a LinkedList and ArrayList.

It maintain the order, it does not maintain the key value pair format.

It allows duplicate value, null value and it is also a thread safe.

**Difference between the ArrayList and Vector---the only difference between the ArrayList and Vector-**

**ArrayList is not thread safe but Vector is thread safe.**

**Hash table**

It is coming from the Map family.

**Difference between the HashTable and HashMap?**

**HashTable- it does not maintain the order, Random access is possible. It pass the key and get the value and maintain key value pair format and it does not allowed duplicate elements and null element.**

**Very Important between HashMap and HashTable is that HashMap allow null element but Hash table does not allowed null element and the Hash table is thread safety that means it is synchronized. It provides thread safety. But HashMap is not synchronized.**

**Example –** this is my Hash table object and multiple threads they are using this particular object at the same time they can not do that because they provide synchronization between the thread. Once the thread T1 is done then only T2 will come over here and this is done and then only T3 will come over here. To access the object properties over here.

**T1 T2 T3**

Hash Table object

Whenever you want to use an application where synchronized is needed, you have to prefer Hashtable . and once the synchronization is not needed , in that case you have to prefer HashMap. They both are coming from Map family and they both are stored the value on the basis of Key value pair format.

**Properties**

We have seen properties that in Selenium and any development framework to maintain the configuration data in the form of key and value pair format which is also a kind of collection. So properties file is also a kind of collection.

* It does no maintain the order,
* Stores the value on the basis of key and value pair format.

Example – You can create one properties file. let’s see qa.properties or config.properties. it maintains the order key is equal to value [k=v].like browser= chrome.

* You pass the key and get the value. I simple say that prop.properties and get the value over there.
* Duplicate elements and null values are not allowed. Exp- you can not maintain the same key over there with a different duplicate values
* Properties files are always thread safety. That’s why they are really good for configuration data. Or Environment variable data that you are really want to store because they are always thread safe.

Example- Let’s see I want to store my Username and password in this properties file and they provide synchronization. It means if some one is changing my username and password or password also someone is changing it. So the next thread will be getting the latest password all the time. Tha is the important of properties file.

**Stack**

* It maintains the order.
* No random access in stack
* It does not maintain key and value pair format.
* Duplicate Element , null values are allowed
* It is Thread safety. It provides sysnchronization.
* Stack is LIFO

Example-

This is my stack. It stores the value. First value will be stored over here. And 2nd value , 3rd value and 4th value will be stored over here. Stack is LIFO . this is the 1st, 2nd, 3rd and 4th guys we have entered. The Last guys will be come out first. That’s why This is Last in First Out, Fifo order it maintains.

5

4 3rd val

2nd value

1st value

Guyes

**Stack**

**Queue—is FIFO**

**Queue would be like this----**and this is first guys , 2nd guys, 3 guys, and4 guys First in First in the Movie ticket Queue . So first value will store over here and then First out in the movie ticket queue.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **5** | **4 guys** | **3** | **2** | **1** |

**--🡪 Queue**

**In the movie ticked queue.**

**Queue is FIFO and 1st values is stored over here and then First out in the movie ticket queue. Queue maintain First in First out. Like Waiting list queue, Movie ticket queue.**

**CopyOnRightArrayList**

Difference between CopyOnRightArrayList and the ArrayList?

**ArrayList-** Order maintain , Random Access is allowed, It does not maintain key value, Duplicate elements and null values are allowed and But ArrayList is not thread safe.

**CopyOnRightArrayList--** Order maintain , Random Access is allowed, It does not maintain key value, Duplicate elements and null values are allowed .Every thing is same as ArrayList except thread safe. CopyOnRightArrayList is synchronized. It provides thread safe.

Whenever you want to use, this is the Enhanced version of ArrayList and then if you really want to use an ArrayList with the Synchronized features or with the thread safety features , you can use **CopyOnRightArrayList.**

**So this is the Synchronized version of ArrayList.**

**ConcurrentHashMap**

* **Random, Key value and Thread safe.**

**Question --Difference between Hash Map and ConcurrentHash Map ?**

In Hash Map, It does not maintain order. Random Access is possible , Key value pair format is also allowed. Duplicate values is not allowed. **But Null values are allowed** . Hash Map is not a thread safe.

In **CopyOnRightArrayList --** In **CopyOnRightArrayList**, It does not maintain order. **Random Access is possible** , Key value pair format is also allowed. Duplicate values **null values are not allowed. But It is a thread safe.**

**Example – If I really want to use an Hash Map with the thread safety , What is the option?**

**The option is that we can use ConcurrentHashMap.**

**CopyOnWriteArraySet**

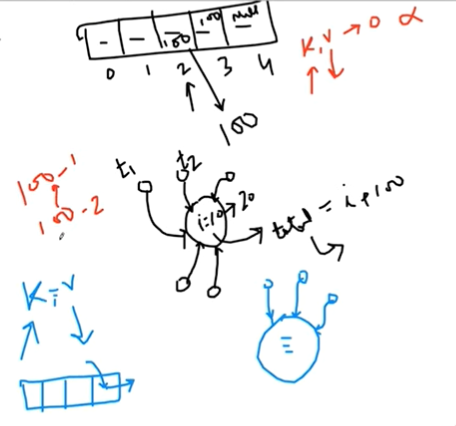
It is a part of HashSet. HashSet is not thread safe but **CopyOnWriteArraySet** is thread safe .

The different between HashSet and CopyOnWriteArraySet is that HashSet is not thread safe but CopyOnWriteArraySet is thread safe.

The other things Both are allowed null values and They are not maintained order, random access is not possible, duplicate elements are not allowed , No key and value pair format in both cases.







=====-

Private int myInt =0;

Public int AddOne()

{

Int tmp = myInt;

tmp = tmp+1;

myInt = tmp;

return tmp;

}

I am writing basic code over here. This private myInt=0. In public One method addOne(). I am Creating a temporary variable🡪 tmp variable. Whatever interger zero I am storing over here. And temporary = 1. I am increasing the value and then tmp wil be 1 in that case. And then storing tmp inside this myInt , it means

I am changing the value of myInt overhere and then but returning the temporary value.

So in this particular method, what I have done ? I have Changed the value of myInt overhere.

What is the problem ?

If I making this particular piece of code , this is not a thread safe.

Assumed that this piece of code is not thread safe. Then what will happen?

The problem is there are multiple threads are running. Let’s se this is my thread number 1,2,3, and thread number 4. The four threads are there. And then this particular thread 1 are coming over here and trying to access this particular method and this method is not thread safe. Then this guys will get the temporary

synchronized

1 2

4 3

Private int myInt =0;

Public int AddOne()

{

Int tmp = myInt;

tmp = tmp+1;

myInt = tmp;

return tmp;

}

Latest Value or temporary value let’s see 1 .So we are returning 1 over here.

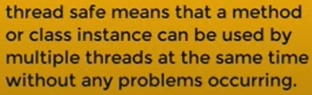
But suddenly this guy thread 4 is also coming over here. This is not thread safe. Any thread can come and call this particular method . So Second thread is coming over here. And then this thread is also checking this okay, hey myInt. But thread 1 is not completed as work .So this thread 4[ 2nd thread ] will get the previous value /old value of myInt. So the calculation will happen on the basis of myInt is zero[myInt= 0;] , not myInt is 1[ myInt= 1]. This is the problem of non synchronized piece of code.

Assumed that this code is not synchronized. Why ? because I have not return any synchronized keyword over here. This method is not declared with synchronized keyword. I have not return any synchronized keyword over here.

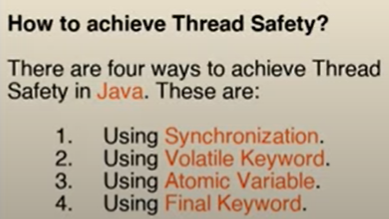
To provide the synchronization the moment I will write one keyword that is called synchronized keyword over here. It means the method is synchronized. It means all the threads which are coming [java will not allow to access this particular threads until the first thread work is done. Once the 1st thread is done then only it will come [i.e the next thread will come ]

Now Let’s see the 1st thread is completed / is done then the 2nd thread will come over here [so what is the latest value of myInt. So the latest value of myInt is 1.]. then it will get the latest value myInt =1. Now.

So the calculation will happen on the basis of latest value . This is the advantage of the synchronized keyword.



How will you get the know this is synchronized or Not synchronized.



There are four ways of creating particular method

Array List is not thread safe. Hash Map is not thread safe. These are the two popular collection which are not thread safe .

But Hash Table is thread safe