When to use Set, List & Map

In previous posts we saw difference between Set, List and Map. We also saw popular implementations of Set, List and Map.

Set, List and Map are very important interfaces in Java Collections Framework.

When to use them also makes a good Java Collections interview question.

1. Let’s say that we want a container or Collection of unique elements then we go with Set. If you want unique elements and insertion order is to be maintained use LinkedHashSet<E>. If you want unique elements and elements in natural order use TreeSet<E>.
2. Now, if you want to retrieve elements using index then use List interface’s implementation ArrayList<E> as it is faster. If insertion order is to be maintained then use ArrayList<E> class. If insertion order is to maintained and there are frequent add remove operation use LinkedList<E> class.
3. If you have a pair of Key and Value mapping:
   1. And insertion order is not important use HashMap<K, V>.
   2. And insertion order is to be maintained then use LinkedHashMap<K, V>.
   3. And insert it in sorted order then use TreeMap<K, V>.
   4. And insertion order is not important but thread safety is important then use ConcurrentHashMap<K, V>.

Quick Reference.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Uniqueness | Insertion Order | Natural Order | 1 to 1 mapping | Interface | Class |
| √ |  |  |  | Set | HashSet |
| √ | √ |  |  | Set | LinkedHashSet |
| √ |  | √ |  | Set | TreeSet |
|  | √ |  |  | List | ArrayList |
| √ |  |  | √ | Map | HashMap |
| √ | √ |  | √ | Map | LinkedHashMap |