The java.util package

30) Make appropriate selection of collection classes/interfaces to suit specified behavior requirements.

java.util package has utility classes covering six basic areas namely − Collections, i18n, Calendar, Jarfiles, Zipfiles, other
Note : Discussed only collection classes/interfaces.

General area	*Interface/Abstract class*	*concrete class*
Collections	Collection	(basic access and update functions)
Set	HashSet(a set of values built on a hash table)	
TreeSet (a set in sorted order)		
List	ArrayList (use instead of Vector)	
LinkedList		
Map	HashMap (use instead of Hashtable)	
TreeMap (a map in sorted order)		
WeakHashMap (a table whose entries go away. When the thing they refer to is garbage collected)		
Other	Stack, Array, BitSet, Iterator(replaces Enumerator)	

Collection :

- A collection represents a group of objects, known as its elements.
- This is just an interface that defines a dozen or so methods for adding to, removing from and querying a data structure.
- It has couple of methods that will flaten all the elements of data structure into an array. This array is either one it allocates, or one you give it as an argument.
- The JDK does not provide any direct implementations of this interface: There are 2 interfaces that extend collection in different directions − Set and List.There is also an interface, Map, that represents key/value pairs.

interface Set

- A Set is a collection that has no duplicate elements.
- There are some methods in this interface to find if a set is empty, or if it contains a specified object and to do an intersection and set difference between this set and another.
- More formally, sets contain no pair of elements e1 and e2 such that e1.equals(e2), and at most one null element.
- An example of a set is the set of fonts that a system can display on its monitor.

HashSet implements the Set interface and extends AbstractSet

- This class maintains a collection of individual objects and you can do intersection, set difference and iteration over the collection. The hash table makes these operations fast.
- You would use HashSet when you have a collection that is not going to have duplicates and you want fast retrieval.
- This class permits the null element.

TreeSet :

- This class is also a Set
- The addition feature of this is, it stores the elements in ascending order, and not in the order in which you added.

java.util.List :

- A List is a collection that has an order assoicated with its elements.
- The user can access elements by their index position in the list, and search for elements in the list.
- Lists can have duplicate elements.
- The List interface provides two methods to efficiently insert and remove multiple elements at an arbitrary point in the list.
- List is an interface, you don't declare it. you declare instances of one of the class that implements it. ArrayList or LinkedList.

ArrayList implements List extends AbstractList :

- Provides methods to manipulate the size of the array that is used internally to store the list.
- To access the ArrayList elements, you can use the List's methods to get set the elements at particular indices.
- permits all elements, including null.
- Each ArrayList instance has a capacity. When you want to add an element and the array is already full, a new larger array is allocated to make room, and the old array is copied over. To cut down an incremental reallocation, you can tell the runtime system how many elements the array should hold when it is constructed.The default is ten elements.

java.util.Map :

- A map is a way of storing key/value pairs. The way of storing a Map is similar to two−column table.
- Given the key, you can get the value associated with it.
- each key can map to atmost one value.
- This interface provides three collection views, which allow viewing a map's contents as a set of keys, a collection of values, or a set of key/value pairs.
- Map doesn't implement the Collections interface.
- A map cannot contain duplicate keys.

HashMap implements Map extends AbstractMap:

- Instead of storing items with index values (ie, 0,1,2,3), you provide the key to be associated with the object. And to get the right object, you only need to provide that key.
- permits null values and the null key
- This class makes no guarantees as to the order of the map; in particular, it does not guarantee that the order will remain

constant over time.**Others :**

BitSet :

- Maintains a set of bits that are identified by the value of an nonnegative integers.
- Individual indexed bits can be examined, set, or cleared.

- Every bit set has a current size,which is the number of bits of space currently in use by the bit set..
- The bit set will increase dynamically as needed to accomodate extra bits you add to it.

Stack :

- The Stack class represents a Last−In−First−Out stack of Objects.
- You can push(store), pop, peek at the top item on the stack.
- There are methods to test whether the stack is empty, to search the stack for an item and discover how far it is from the top.
- When a stack is first created, it contains no items.

Iterator :

- You can retrieve all the elements in a data structure without have to know all the details about how or where they are stored.
- The class itself knows those details and will implement the iterator methods to supply a way to count through all the objects in its collection.

Hints to choose between Collection classes :

- If you have a collection of data and if there are no duplicates and also want the elements to in sorted order, you could use TreeSet.
- If you have a collection of data and if there are some duplicate entries, you could use some kind of List.
- If you want to store key/value pairs, you could some kind of Map.
- If you want to read and evaluate expressions, you could choose a Stack.

Remember :

- Set : Can't contain duplicate elements − HashSet, TreeSet
- List : Can contain duplicate elements elements are accessed by their integer index − ArrayList, LinkedList
- Map : Can't contain duplicate keys − HashMap, TreeMap
- Set and List implements Collection interface and Map doesn't

---
