Vector in Java

BY CHAITANYA SINGH | FILED UNDER: JAVA COLLECTIONS

`Vector` implements List Interface. Like `ArrayList` it also maintains insertion order but it is rarely used in non-thread environment as it is synchronized and due to which it gives poor performance in searching, adding, delete and update of its elements.

Three ways to create vector class object:

Method 1:

```
Vector vec = new Vector();
```

It creates an empty Vector with the default initial capacity of 10. It means the Vector will be re-sized when the 11th elements needs to be inserted into the Vector. Note: By default vector doubles its size. i.e. In this case the Vector size would remain 10 till 10 insertions and once we try to insert the 11th element It would become 20 (double of default capacity 10).

Method 2:
Syntax: `Vector object= new Vector(int initialCapacity)`

```
Vector vec = new Vector(3);
```

It will create a Vector of initial capacity of 3.

Method 3:
Syntax:

```
Vector object= new vector(int initialcapacity, capacityIncrement)
```

Example:

```
Vector vec= new Vector(4, 6)
```

Here we have provided two arguments. The initial capacity is 4 and capacityIncrement is 6. It means upon insertion of 5th element the size would be 10 (4+6) and on 11th insertion it would be 16(10+6).

Complete Example of Vector in Java:

```
import java.util.*;

public class VectorExample {

   public static void main(String args[]) {
      /* Vector of initial capacity(size) of 2 */
      Vector<String> vec = new Vector<String>(2);

      /* Adding elements to a vector*/
      vec.addElement("Apple");
      vec.addElement("Orange");
      vec.addElement("Mango");
      vec.addElement("Fig");

      /* check size and capacityIncrement*/
      System.out.println("Size is: "+vec.size());
      System.out.println("Default capacity increment is: "+vec.capacity());

      vec.addElement("fruit1");
      vec.addElement("fruit2");
      vec.addElement("fruit3");

      /*size and capacityIncrement after two insertions*/
      System.out.println("Size after addition: "+vec.size());
      System.out.println("Capacity after increment is: "+vec.capacity());

      /*Display Vector elements*/
      Enumeration en = vec.elements();
      System.out.println("\nElements are:");
      while(en.hasMoreElements())
         System.out.print(en.nextElement() + " ");
   }
}
```

Output:

```
Size is: 4
Default capacity increment is: 4
Size after addition: 7
Capacity after increment is: 8

Elements are:
Apple Orange Mango Fig fruit1 fruit2 fruit3
```

Commonly used methods of Vector Class:

1. **void addElement(Object element):** It inserts the element at the end of the Vector.
2. **int capacity():** This method returns the current capacity of the vector.
3. **int size():** It returns the current size of the vector.
4. **void setSize(int size):** It changes the existing size with the specified size.
5. **boolean contains(Object element):** This method checks whether the specified element is present in the Vector. If the element is been found it returns true else false.
6. **boolean containsAll(Collection c):** It returns true if all the elements of collection c are present in the Vector.
7. **Object elementAt(int index):** It returns the element present at the specified location in Vector.
8. **Object firstElement():** It is used for getting the first element of the vector.

9. **Object lastElement():** Returns the last element of the array.
10. **Object get(int index):** Returns the element at the specified index.
11. **boolean isEmpty():** This method returns true if Vector doesn’t have any element.
12. **boolean removeElement(Object element):** Removes the specifed element from vector.
13. **boolean removeAll(Collection c):** It Removes all those elements from vector which are present in the Collection c.
14. **void setElementAt(Object element, int index):** It updates the element of specifed index with the given element.