**Java Vector 类**

Vector类实现了一个动态数组。和ArrayList和相似，但是两者是不同的：

* Vector是同步访问的。
* Vector包含了许多传统的方法，这些方法不属于集合框架。

Vector主要用在事先不知道数组的大小，或者只是需要一个可以改变大小的数组的情况。

Vector类支持4种构造方法。

第一种构造方法创建一个默认的向量，默认大小为10：

Vector()

第二种构造方法创建指定大小的向量。

Vector(int size)

第三种构造方法创建指定大小的向量，并且增量用incr指定. 增量表示向量每次增加的元素数目。

Vector(int size,int incr)

第四中构造方法创建一个包含集合c元素的向量：

Vector(Collection c)

除了从父类继承的方法外Vector还定义了以下方法：

|  |  |
| --- | --- |
| **序号** | **方法描述** |
| 1 | void add(int index, Object element)   在此向量的指定位置插入指定的元素。 |
| 2 | boolean add(Object o)   将指定元素添加到此向量的末尾。 |
| 3 | boolean addAll(Collection c)  将指定 Collection 中的所有元素添加到此向量的末尾，按照指定 collection 的迭代器所返回的顺序添加这些元素。 |
| 4 | boolean addAll(int index, Collection c)  在指定位置将指定 Collection 中的所有元素插入到此向量中。 |
| 5 | void addElement(Object obj)   将指定的组件添加到此向量的末尾，将其大小增加 1。 |
| 6 | int capacity()  返回此向量的当前容量。 |
| 7 | void clear()  从此向量中移除所有元素。 |
| 8 | Object clone()  返回向量的一个副本。 |
| 9 | boolean contains(Object elem)  如果此向量包含指定的元素，则返回 true。 |
| 10 | boolean containsAll(Collection c)  如果此向量包含指定 Collection 中的所有元素，则返回 true。 |
| 11 | void copyInto(Object[] anArray)   将此向量的组件复制到指定的数组中。 |
| 12 | Object elementAt(int index)  返回指定索引处的组件。 |
| 13 | Enumeration elements()  返回此向量的组件的枚举。 |
| 14 | void ensureCapacity(int minCapacity)  增加此向量的容量（如有必要），以确保其至少能够保存最小容量参数指定的组件数。 |
| 15 | boolean equals(Object o)  比较指定对象与此向量的相等性。 |
| 16 | Object firstElement()  返回此向量的第一个组件（位于索引 0) 处的项）。 |
| 17 | Object get(int index)  返回向量中指定位置的元素。 |
| 18 | int hashCode()  返回此向量的哈希码值。 |
| 19 | int indexOf(Object elem)   返回此向量中第一次出现的指定元素的索引，如果此向量不包含该元素，则返回 -1。 |
| 20 | int indexOf(Object elem, int index)   返回此向量中第一次出现的指定元素的索引，从 index 处正向搜索，如果未找到该元素，则返回 -1。 |
| 21 | void insertElementAt(Object obj, int index)  将指定对象作为此向量中的组件插入到指定的 index 处。 |
| 22 | boolean isEmpty()  测试此向量是否不包含组件。 |
| 23 | Object lastElement()  返回此向量的最后一个组件。 |
| 24 | int lastIndexOf(Object elem)   返回此向量中最后一次出现的指定元素的索引；如果此向量不包含该元素，则返回 -1。 |
| 25 | int lastIndexOf(Object elem, int index)  返回此向量中最后一次出现的指定元素的索引，从 index 处逆向搜索，如果未找到该元素，则返回 -1。 |
| 26 | Object remove(int index)   移除此向量中指定位置的元素。 |
| 27 | boolean remove(Object o)  移除此向量中指定元素的第一个匹配项，如果向量不包含该元素，则元素保持不变。 |
| 28 | boolean removeAll(Collection c)  从此向量中移除包含在指定 Collection 中的所有元素。 |
| 29 | void removeAllElements()  从此向量中移除全部组件，并将其大小设置为零。 |
| 30 | boolean removeElement(Object obj)  从此向量中移除变量的第一个（索引最小的）匹配项。 |
| 31 | void removeElementAt(int index)  删除指定索引处的组件。 |
| 32 | protected void removeRange(int fromIndex, int toIndex) 从此 List 中移除其索引位于 fromIndex（包括）与 toIndex（不包括）之间的所有元素。 |
| 33 | boolean retainAll(Collection c)  在此向量中仅保留包含在指定 Collection 中的元素。 |
| 34 | Object set(int index, Object element)  用指定的元素替换此向量中指定位置处的元素。 |
| 35 | void setElementAt(Object obj, int index)  将此向量指定 index 处的组件设置为指定的对象。 |
| 36 | void setSize(int newSize)   设置此向量的大小。 |
| 37 | int size()   返回此向量中的组件数。 |
| 38 | List subList(int fromIndex, int toIndex)  返回此 List 的部分视图，元素范围为从 fromIndex（包括）到 toIndex（不包括）。 |
| 39 | Object[] toArray()  返回一个数组，包含此向量中以恰当顺序存放的所有元素。 |
| 40 | Object[] toArray(Object[] a)  返回一个数组，包含此向量中以恰当顺序存放的所有元素；返回数组的运行时类型为指定数组的类型。 |
| 41 | String toString()  返回此向量的字符串表示形式，其中包含每个元素的 String 表示形式。 |
| 42 | void trimToSize()    对此向量的容量进行微调，使其等于向量的当前大小。 |

**实例**

下面的程序说明这个集合所支持的几种方法：

import java.util.\*;

public class VectorDemo {

public static void main(String args[]) {

// initial size is 3, increment is 2

Vector v = new Vector(3, 2);

System.out.println("Initial size: " + v.size());

System.out.println("Initial capacity: " +

v.capacity());

v.addElement(new Integer(1));

v.addElement(new Integer(2));

v.addElement(new Integer(3));

v.addElement(new Integer(4));

System.out.println("Capacity after four additions: " +

v.capacity());

v.addElement(new Double(5.45));

System.out.println("Current capacity: " +

v.capacity());

v.addElement(new Double(6.08));

v.addElement(new Integer(7));

System.out.println("Current capacity: " +

v.capacity());

v.addElement(new Float(9.4));

v.addElement(new Integer(10));

System.out.println("Current capacity: " +

v.capacity());

v.addElement(new Integer(11));

v.addElement(new Integer(12));

System.out.println("First element: " +

(Integer)v.firstElement());

System.out.println("Last element: " +

(Integer)v.lastElement());

if(v.contains(new Integer(3)))

System.out.println("Vector contains 3.");

// enumerate the elements in the vector.

Enumeration vEnum = v.elements();

System.out.println("\nElements in vector:");

while(vEnum.hasMoreElements())

System.out.print(vEnum.nextElement() + " ");

System.out.println();

}

}

以上实例编译运行结果如下：

Initial size: 0

Initial capacity: 3

Capacity after four additions: 5

Current capacity: 5

Current capacity: 7

Current capacity: 9

First element: 1

Last element: 12

Vector contains 3.

Elements in vector:

1 2 3 4 5.45 6.08 7 9.4 10 11 12