

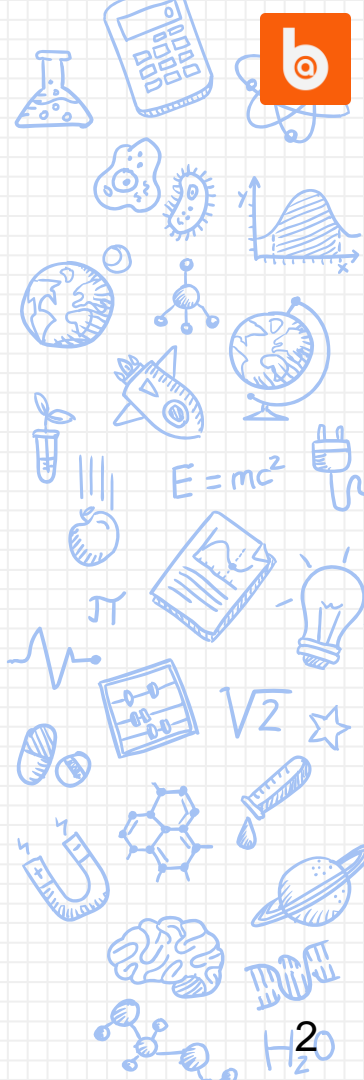
Bài 2.11: Lớp Vector

<https://braniumacademy.net>



Nội dung bài học

- ☐ Tổng quan
- ☐ Các phương thức thường dùng và mô tả
- ☐ Ví dụ minh họa





Tổng quan

- ❑ Vector là một lớp triển khai của mảng các đối tượng có thể co dãn kích thước.
- ❑ Vector hỗ trợ truy cập phần tử sử dụng chỉ số.
- ❑ Kích thước của vector tự động thay đổi cho phù hợp khi thêm mới hoặc xóa phần tử.
- ❑ Vector sử dụng capacity và capacityIncrement để tối ưu quản lý bộ nhớ. Khi cần mở rộng khả năng lưu trữ, vector sẽ tăng kích thước lên 1 lượng bằng capacityIncrement.

Tổng quan

- ❑ Vector là đồng bộ hóa. Nếu việc truy cập luồng an toàn là không cần thiết, thay thế nó bằng ArrayList.



Các phương thức và mô tả

Phương thức	Mô tả
<code>Vector()</code>	Phương thức khởi tạo một vector rỗng với capacity = 10 và capacityIncrement = 0.
<code>Vector(Collection<? Extends E> c)</code>	Tạo vector từ một collection truyền vào theo đúng thứ tự các phần tử hiện có của c.
<code>Vector(int capacity)</code>	Tạo vector rỗng với khả năng lưu trữ cho trước và lượng mở rộng khả năng lưu trữ bằng 0.
<code>Vector(int capacity, int capacityIncrement)</code>	Tạo một vector rỗng với khả năng lưu trữ và lượng mở rộng khả năng lưu trữ cho trước.
<code>boolean add(E e)</code>	Thêm phần tử e vào cuối vector.
<code>void add(int index, E e)</code>	Chèn phần tử e vào vị trí index trong vector.
<code>boolean addAll(Collection<? Extends E> c)</code>	Thêm tất cả các phần tử trong collection c vào cuối vector hiện tại. Theo đúng trật tự trong c.
<code>boolean addAll(int index, Collection<? Extends E> c)</code>	Chèn tất cả các phần tử trong collection c vào vị trí index của vector hiện tại.
<code>void addElement(E e)</code>	Thêm một phần tử vào cuối vector, tăng kích thước vector lên 1.
<code>int capacity()</code>	Trả về khả năng lưu trữ hiện tại của vector.
<code>void clear()</code>	Xóa toàn bộ các phần tử trong vector hiện tại.
<code>Object clone()</code>	Trả về bản sao của vector hiện tại.



Các phương thức và mô tả

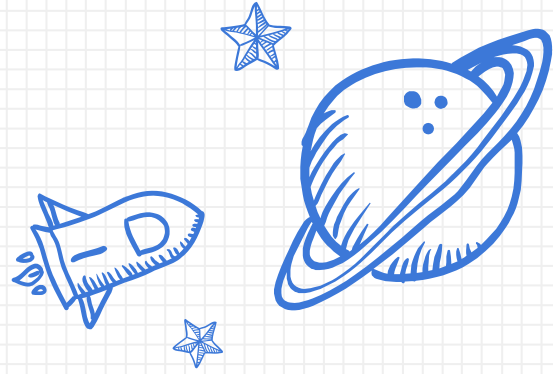
<code>boolean contains(Object o)</code>	Kiểm tra xem phần tử o có trong vector không.
<code>boolean containsAll(Collection<?> c)</code>	Kiểm tra xem vector hiện thời có chứa tất cả các phần tử cho trong collection c của tham số không.
<code>void copyInto(Object[] anArray)</code>	Sao chép các phần tử hiện có của vector vào một mảng cho trước.
<code>E elementAt(int index)</code>	Trả về phần tử tại vị trí index cho trước.
<code>Enumeration<E> elements()</code>	Trả về một enum các thành phần của vector hiện thời.
<code>void ensureCapacity(int minCapacity)</code>	Tăng khả năng lưu trữ của vector hiện thời nếu cần thiết để chắc chắn rằng nó có thể chứa ít nhất minCapacity phần tử.
<code>boolean equals(Object o)</code>	So sánh vector hiện thời với o xem chúng có tương đương không.
<code>E firstElement()</code>	Trả về phần tử đầu tiên của vector tại vị trí chỉ số = 0.
<code>void forEach(Consumer<? super E> action)</code>	Thực hiện hành động action cho trước đối với từng phần tử của vector cho đến khi tất cả các phần tử của vector đã được xử lý hoặc đến khi gặp ngoại lệ.
<code>E get(int index)</code>	Trả về phần tử tại vị trí index trong vector.
<code>int hashCode()</code>	Trả về mã băm của vector hiện thời.
<code>int indexOf(Object o)</code>	Trả về vị trí xuất hiện đầu tiên của o trong vector hiện tại. Nếu không tồn tại o, trả về -1.
<code>int indexOf(Object o, int index)</code>	Trả về vị trí xuất hiện đầu tiên của o trong vector bắt đầu từ vị trí index cho trước. Nếu không tìm thấy, trả về -1.

Các phương thức và mô tả

<code>void insertElementAt(E obj, int index)</code>	Chèn một phần tử vào vector tại vị trí xác định trong index.
<code>int lastIndexOf(Object o)</code>	Trả về vị trí xuất hiện cuối cùng của o trong vector. Trả về -1 nếu không tìm thấy.
<code>int lastIndexOf(Object o, int index)</code>	Trả về vị trí xuất hiện cuối cùng của o trong vector tính từ vị trí index. Trả về -1 nếu không tìm thấy.
<code>boolean isEmpty()</code>	Trả về true nếu vector rỗng và ngược lại.
<code>Iterator<E> iterator()</code>	Trả về iterator dùng để duyệt các phần tử trong vector hiện thời.
<code>ListIterator<E> listIterator()</code>	Trả về một list iterator để duyệt các phần tử trong vector hiện thời.
<code>ListIterator<E> listIterator(int index)</code>	Trả về một list iterator để duyệt các phần tử trong vector hiện thời bắt đầu từ vị trí index.
<code>E remove(int index)</code>	Xóa phần tử ở vị trí index.
<code>boolean remove(Object o)</code>	Xóa phần tử o đầu tiên xuất hiện trong vector.
<code>boolean removeAll(Collection<?> c)</code>	Xóa tất cả các phần tử có mặt trong c khỏi vector hiện tại.
<code>void removeAllElements()</code>	Xóa tất cả các phần tử trong vector hiện tại.
<code>boolean removeElement(Object o)</code>	Xóa phần tử o đầu tiên xuất hiện trong vector.
<code>void removeElementAt(int index)</code>	Xóa phần tử tại vị trí index.

Các phương thức và mô tả

<code>boolean removeIf(Predicate<? super E> filter)</code>	Xóa tất cả các phần tử trong vector hiện thời thỏa mãn điều kiện được chỉ ra trong tham số.
<code>protected void removeRange(int fromIndex, int toIndex)</code>	Xóa các phần tử có chỉ số bắt đầu từ <code>fromIndex</code> đến trước <code>toIndex</code> .
<code>void replaceAll(UnaryOperator<E> op)</code>	Thay thế phần tử trong vector với kết quả sau khi áp dụng toán tử <code>op</code> với phần tử đó.
<code>boolean retainAll(Collection<?> c)</code>	Giữ lại chỉ các phần tử trong vector hiện thời mà có mặt trong <code>c</code> .
<code>E set(int index, E element)</code>	Thay thế phần tử tại vị trí <code>index</code> bởi giá trị <code>element</code> .
<code>void setElementAt(E e, int index)</code>	Gán giá trị <code>e</code> cho phần tử tại vị trí <code>index</code> trong vector.
<code>void sort(Comparator<? Super E> c)</code>	Sắp xếp vector theo trật tự được chỉ ra bởi <code>Comparator c</code> .
<code>void setSize()</code>	Thiết lập kích thước của vector.
<code>int size()</code>	Trả về số lượng các phần tử hiện có trong vector.
<code>List<E> subList(int fromIndex, int toIndex)</code>	Trả về một list của các phần tử bắt đầu từ vị trí <code>fromIndex</code> đến trước vị trí <code>toIndex</code> .
<code>Object[] toArray()</code>	Trả về mảng tất cả các đối tượng trong vector theo thứ tự đầu đến cuối.
<code>T[] toArray(T[] a)</code>	Trả về một mảng tất cả các phần tử trong vector ở kiểu <code>T</code> .
<code>String toString()</code>	Trả về string đại diện cho vector hiện thời, trong đó chứa tất cả các phần tử của vector.
<code>void trimToSize()</code>	Cắt giảm khả năng lưu trữ của vector về kích thước hiện thời của nó.



Tiếp theo

Danh sách liên kết vòng