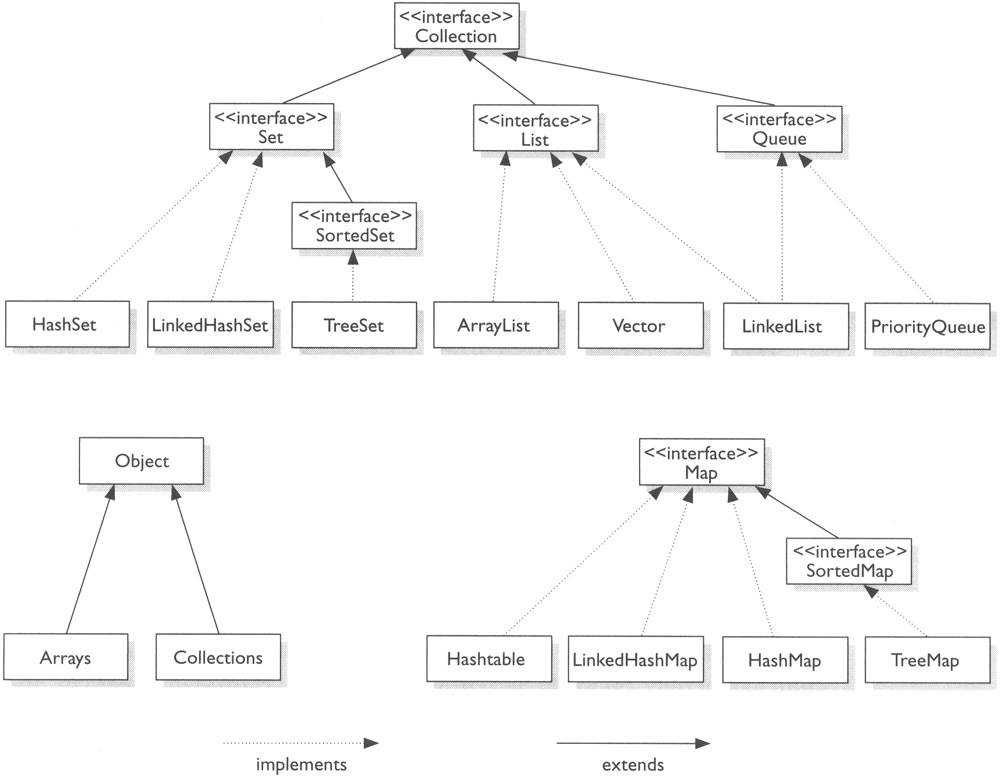
**List và những thư viện đi kèm theo List**

List Interface được kế thừa từ Collection nên List có những đặc tính của Collection, ngoài ra List còn có thêm những thuộc tính của riêng nó như:

- Thao tác với các yếu tố dựa trên vị trí của chúng trong danh sách thông qua các phương thức của nó như: get, set, add, remove, addAll.

- Tìm kiếm trong danh sách bằng lệnh indexOf hoặc lastIndexOf.

- Trong List còn có 2 thứ viện là ArrayList và LinkedList.



Những phương thức riêng của List

|  |  |
| --- | --- |
| **STT** | **Phương thức và Miêu tả** |
| 1 | **void add(int index, Object obj)**  Chèn obj vào trong List đang gọi tại index đã cho. Bất kỳ phần tử nào đã tồn tại trước tại hoặc trên điểm chèn bị bỏ qua. Vì thế, không có phần tử nào bị ghi đè. Nếu add() sẽ chèn vào cuối. |
| 2 | **boolean addAll(int index, Collection c)**  Chèn tất cả phần tử của c vào trong List đang gọi tại chỉ mục đã cho. Bất kỳ phần tử nào đã tồn tại trước tại hoặc trên điểm chèn bị bỏ qua. Vì thế, không có phần tử nào bị ghi đè. Trả về true nếu List đang gọi thay đổi và nếu không trả về false |
| 3 | **Object get(int index)**  Trả về đối tượng được lưu giữ tại index đã cho bên trong Collection đang gọi |
| 4 | **int indexOf(Object obj)**  Trả về index của sự xuất hiện đầu tiên của obj trong List đang gọi. Nếu obj không là một phần tử trong list, -1 được trả về |
| 5 | **int lastIndexOf(Object obj)**  Trả về index của sự xuất hiện cuối cùng của obj trong List đang gọi. Nếu obj không là một phần tử trong list, -1 được trả về |
| 6 | **ListIterator listIterator( )**  Trả về một iterator tới phần đầu của List đang gọi |
| 7 | **ListIterator listIterator(int index)**  Trả về một iterator tới List đang gọi tại index đã cho |
| 8 | **Object remove(int index)**  Gỡ bỏ phần tử tại index từ List đang gọi và trả về phần tử bị xóa đó. List kết quả được compact lại. Đó là, các chỉ mục của dãy phần tử phụ bị lượng giảm đi 1. Nếu remove() sẽ xóa đầu và không trả về. |
| 9 | **Object set(int index, Object obj)**  Gán obj tới vị trí được xác định bởi index bên trong List đang gọi |
| 10 | **List subList(int start, int end)**  Trả về một list mà bao gồm các phần tử từ start tới end-1 trong List đang gọi. Các phần tử trong list trả về cũng được tham chiếu bởi đối tượng đang gọi |

\* Sự khác nhau giữa ArrayList và LinkedList

-ArrayList là cấu trúc dữ lieu có chỉ mục nền khi tìm kiềm Array nhanh hơn LinkedList vì LinkedList là cấu trúc dữ liệu không chỉ mục nên mỗi khi muốn tìm phần tử nào ta phải duyệt qua toàn bộ mảng.

- ArrayList ít tốn bộ nhớ hơn vì Array mỗi phần tử chỉ lưu một đối tượng còn LinkedList còn phải chứa còn trỏ trỏ đến đối tượng trước đó và đối tượng sau đó.

- Chèn, Xóa với LinkedList lại là một lời thế vì chúng ta không sợ rủi ro về sự thay đổi kích thước của mảng, và chỉ mục không bj thay đổi vì LinkedList không sử dụng chỉ mục. Nên nếu ứng dụng dùng mảng mà ít tìm kiếm thì ta nên dùng LinkedList còn lại thì ta nên dùng ArrayList.

\* Array và những phương thức của nó:

|  |  |
| --- | --- |
| **STT** | **Phương thức và Miêu tả** |
| 1 | **void add(int index, Object element)**  Chèn phần tử đã cho tại index đã xác định trong list này. Ném IndexOutOfBoundsException nếu index này ở bên ngoài dãy (index < 0 || index > size()) |
| 2 | **boolean add(Object o)**  Phụ thêm phần tử đã xác định tới cuối list này |
| 3 | **boolean addAll(Collection c)**  Phụ thêm tất cả phần tử trong collection đã xác định tới cuối list này, trong thứ tự mà chúng được trả về bởi Iterator của collection đã cho. Ném NullPointerException nếu collection đã cho là null |
| 4 | **boolean addAll(int index, Collection c)**  Chèn tất cả phần tử trong collection đã cho vào trong list này, bắt đầu tại vị trí đã cho. Ném NullPointerException nếu collection đã cho là null |
| 5 | **void clear()**  Gỡ bỏ tất cả phần tử từ list này |
| 6 | **Object clone()**  Trả về một shallow copy của ArrayList này |
| 7 | **boolean contains(Object o)**  Trả về true nếu list này chứa phần tử đã cho. Chính thức hơn, trả về true nếu và chỉ nếu list này chứa ít nhất một phần tử e như (o==null ? e==null : o.equals(e)) |
| 8 | **void ensureCapacity(int minCapacity)**  Tăng dung lượng của ArrayList này, nếu cần thiết, để bảo đảm rằng nó có thể giữ tối thiểu số phần tử đã được xác định bởi tham số minCapacity |
| 9 | **Object get(int index)**  Trả về phần tử tại index đã cho. Ném IndexOutOfBoundsException nếu index đã cho là ở bên ngoài dãy (index < 0 || index >= size()) |
| 10 | **int indexOf(Object o)**  Trả về index trong list này của sự xuất hiện đầu tiên của phần tử đã cho, hoặc -1 nếu List không chứa phần tử này |
| 11 | **int lastIndexOf(Object o)**  Trả về index trong list này của sự xuất hiện cuối của phần tử đã cho, hoặc -1 nếu List không chứa phần tử này |
| 12 | **Object remove(int index)**  Gỡ bỏ phần tử tại index đã cho. Ném IndexOutOfBoundsException nếu index ở ngoài dãy (index < 0 || index >= size()) |
| 13 | **protected void removeRange(int fromIndex, int toIndex)**  Gỡ bỏ từ list này tất cả phần tử mà có index ở giữa fromIndex và toIndex |
| 14 | **Object set(int index, Object element)**  Thay thế phần tử tại vị trí đã cho trong list này với phần tử đã xác định. Ném IndexOutOfBoundsException nếu index ở ngoài dãy (index < 0 || index >= size()) |
| 15 | **int size()**  Trả về số phần tử trong list này |
| 16 | **Object[] toArray()**  Trả về mảng chứa tất cả phần tử trong list này trong thứ tự chính xác. Ném NullPointerException nếu mảng này là null |
| 17 | **Object[] toArray(Object[] a)**  Trả về một mảng chứa tất cả phần tử trong list này trong thứ tự chính xác; kiểu runtime là của mảng trả về là như của mảng đã xác định |
| 18 | **void trimToSize()**  Trim dung lượng của ArrayList này về kích cỡ hiện tại của list đó |

Ngoài những phương thức căn bản thừa kế sẵn từ List Interface ArrayList còn bổ xung thêm những phương thức riêng cho nó như: clone, clear, size,..

\* LinkedList cũng kế thừa từ List: cũng như ArrayList, LinkedList cũng kế thừa từ List và bổ xung thêm những phương thức mới dành riêng cho nó. Nhưng khác với ArrayList, LinkedList chỉ bổ xung 2 constructor là LinkedList() và LinkedList(Collection c). Những phương thức mới của LinkedList như: addFirst, addLast, clear, clone, getFirst, getLast, removeFirst, removeLast….

|  |  |
| --- | --- |
| **STT** | **Phương thức và Miêu tả** |
| 1 | **void add(int index, Object element)**  Chèn element đã xác định tại index đã cho. Ném một IndexOutOfBoundsException nếu index đã cho là ở bên ngoài dãy (index < 0 || index > size()) |
| 2 | **boolean add(Object o)**  Phụ thêm phần tử đã cho tới cuối của List này |
| 3 | **boolean addAll(Collection c)**  Phụ thêm tất cả phần tử trong collection đã cho tới cuối của list này, theo thứ tự mà chúng được trả về bởi Iterator của collection đã cho. Ném một NullPointerException nếu collection đã cho là null |
| 4 | **boolean addAll(int index, Collection c)**  Chèn tất cả phần tử trong collection đã cho vào trong List này, bắt đầu từ vị trí đã cho. Ném NullPointerException nếu collection đã cho là null |
| 5 | **void addFirst(Object o)**  Chèn phần tử đã cho vào phần đầu của list này |
| 6 | **void addLast(Object o)**  Phụ thêm phần tử đã cho vào phần cuối của list này |
| 7 | **void clear()**  Gỡ bỏ tất cả phần tử từ list này |
| 8 | **Object clone()**  Trả về một shallow copy của LinkedList này |
| 9 | **boolean contains(Object o)**  Trả về true nếu list này chứa phần tử đã cho. Chính thức hơn, trả về true nếu và chỉ nếu list này chứa ít nhất một phần tử e để mà (o==null ? e==null : o.equals(e)) |
| 10 | **Object get(int index)**  Trả về phần tử tại vị trí đã cho. Ném IndexOutOfBoundsException nếu index ở bên ngoài dãy (index < 0 || index >= size()) |
| 11 | **Object getFirst()**  Trả về phần tử đầu tiên trong list này. Ném NoSuchElementException nếu list này là trống |
| 12 | **Object getLast()**  Trả về phần tử cuối trong list này. Ném NoSuchElementException nếu list này là trống |
| 13 | **int indexOf(Object o)**  Trả về index trong list này cho sự xuất hiện đầu tiên của phần tử đã cho, hoặc -1 nếu List này không chứa phần tử này |
| 14 | **int lastIndexOf(Object o)**  Trả về index trong list này cho sự xuất hiện cuối của phần tử đã cho, hoặc -1 nếu List này không chứa phần tử này |
| 15 | **ListIterator listIterator(int index)**  Trả về một list-iterator của phần tử trong list này (trong dãy chính xác), bắt đầu từ vị trí đã cho trong list. Ném IndexOutOfBoundsException nếu index đã cho ở bên ngoài dãy (index < 0 || index >= size()) |
| 16 | **Object remove(int index)**  Gỡ bỏ phần tử tại vị trí đã cho. Ném NoSuchElementException nếu list này là trống |
| 17 | **boolean remove(Object o)**  Gỡ bỏ sự xuất hiện đầu tiên của phần tử đã cho. Ném NoSuchElementException nếu list này trống. Ném IndexOutOfBoundsException nếu index ở bên ngoài dãy (index < 0 || index >= size()) |
| 18 | **Object removeFirst()**  Gỡ bỏ và trả về phần tử đầu tiên từ list này. Ném NoSuchElementException nếu list là trống |
| 19 | **Object removeLast()**  Gỡ bỏ và trả về phần tử cuối từ list này. Ném NoSuchElementException nếu list là trống |
| 20 | **Object set(int index, Object element)**  Thay thế phần tử tại vị trí đã cho trong list này với phần tử đã cho. Ném IndexOutOfBoundsException nếu index đã cho ở ngoài dãy (index < 0 || index >= size()) |
| 21 | **int size()**  Trả về số phần tử trong list này |
| 22 | **Object[] toArray()**  Trả về một mảng chứa tất cả phần tử trong list này trong đúng thứ tự. Ném NullPointerException nếu mảng đã xác định là null |
| 23 | **Object[] toArray(Object[] a)**  Trả về một mảng chứa tất cả phần tử trong list này trong đúng thứ tự; kiểu runtime của mảng trả về là như của mảng đã xác định |