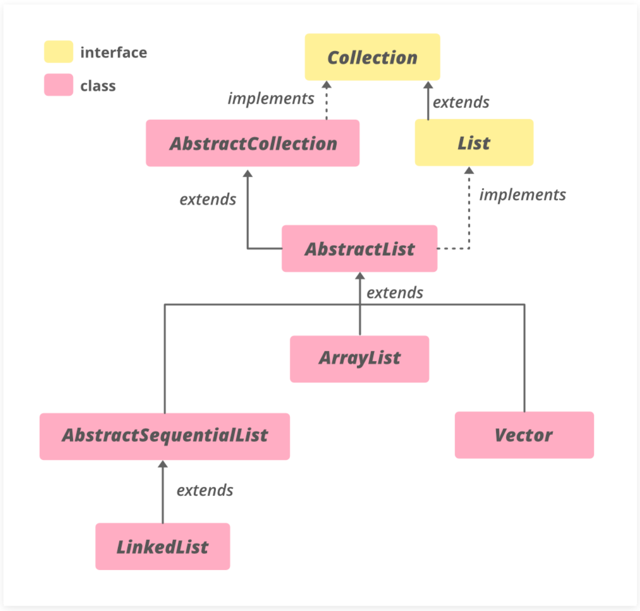
AbstractSequentialList in Java with Examples

Difficulty Level : Medium Last Updated : 11 Nov, 2020

The **AbstractSequentialList** class in Java is a part of the Java Collection Framework and implements the *Collection interface* and the *AbstractCollection class*. It is used to implement an unmodifiable list, for which one needs to only extend this AbstractList Class and implement only the get() and the size() methods.

This class provides a skeletal implementation of the List interface to minimize the effort required to implement this interface backed by a "sequential access" data store (such as a linked list). For random access data (such as an array), AbstractList should be used in preference to this class.

Class Hierarchy:



Declaration:

```
public abstract class AbstractSequentialList<E>
    extends AbstractList<E>

Where E is the type of element maintained by this List.
```

It implements **Iterable<E>**, **Collection<E>**, List<E> interfaces. LinkedList is the only direct subclass of AbstractSequentialList.

Constructor: *protected AbstractSequentialList()* – The default constructor, but being protected, it doesn't allow to create an AbstractSequentialList object.

AbstractSequentialList<E> asl = new LinkedList<E>();

Example 1: AbstractSequentialList is an abstract class, so it should be assigned an instance of its subclass such as LinkedList.

```java
// Java code to illustrate AbstractSequentialList
import java.util.*;

public class GfG {

    public static void main(String[] args)
    {
        // Creating an instance of
        // the AbstractSequentialList
        AbstractSequentialList<Integer> absl
            = new LinkedList<>();

        // adding elements to absl
        absl.add(5);
        absl.add(6);
        absl.add(7);

        // Printing the list
        System.out.println(absl);
    }
}
```

Output:

```
[5, 6, 7]
```

Example 2:

```java
// Java code to illustrate
// methods of AbstractSequentialList

import java.util.*;
import java.util.AbstractSequentialList;

public class AbstractSequentialListDemo {
    public static void main(String args[])
    {

        // Creating an empty AbstractSequentialList
        AbstractSequentialList<String>
            absqlist = new LinkedList<String>();

        // Using add() method to
          // add elements in the list
        absqlist.add("Geeks");
        absqlist.add("for");
        absqlist.add("Geeks");
        absqlist.add("10");
        absqlist.add("20");

        // Output the list
        System.out.println("AbstractSequentialList: "
                           + absqlist);

        // Remove the head using remove()
        absqlist.remove(3);

        // Print the final list
        System.out.println("Final List: "
                           + absqlist);

        // Fetching the specific
          // element from the list
        // using get() method
        System.out.println("The element is: "
                           + absqlist.get(2));
    }
}
```

Output:

```
AbstractSequentialList: [Geeks, for, Geeks, 10, 20]
Final List: [Geeks, for, Geeks, 20]
The element is: Geeks
```

Methods of AbstractSequentialList

METHOD	DESCRIPTION
add(int index, E element)	Inserts the specified element at the specified position in this list (optional operation).
addAll(int index, Collection<? extends E> c)	Inserts all of the elements in the specified collection into this list at the specified position (optional operation).
get(int index)	Returns the element at the specified position in this list.
iterator()	Returns an iterator over the elements in this list (in proper sequence).
listIterator(int index)	Returns a list iterator over the elements in this list (in proper sequence).
remove(int index)	Removes the element at the specified position in this list (optional operation).
set(int index, E element)	Replaces the element at the specified position in this list with the specified element (optional operation).

Methods Inherited From class java.util.AbstractList

METHOD	DESCRIPTION
add(E e)	Appends the specified element to the end of this list (optional operation).
clear()	Removes all of the elements from this list (optional operation).
equals(Object o)	Compares the specified object with this list for equality.
hashCode()	Returns the hash code value for this list.
indexOf(Object o)	Returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element.

METHOD	DESCRIPTION
lastIndexOf(Object o)	Returns the index of the last occurrence of the specified element in this list, or -1 if this list does not contain the element.
listIterator()	Returns a list iterator over the elements in this list (in proper sequence).
removeRange(int fromIndex, int toIndex)	Removes from this list all of the elements whose index is between fromIndex, inclusive, and toIndex, exclusive.
subList(int fromIndex, int toIndex)	Returns a view of the portion of this list between the specified fromIndex, inclusive, and toIndex, exclusive.

Methods Inherited From class java.util.AbstractCollection

METHOD	DESCRIPTION
addAll (Collection<? extends E> c)	Adds all of the elements in the specified collection to this collection (optional operation).
contains(Object o)	Returns true if this collection contains the specified element.
containsAll (Collection<?> c)	Returns true if this collection contains all of the elements in the specified collection.
isEmpty()	Returns true if this collection contains no elements.
remove(Object o)	Removes a single instance of the specified element from this collection, if it is present (optional operation).
removeAll (Collection<?> c)	Removes all of this collection’s elements that are also contained in the specified collection (optional operation).
retainAll (Collection<?> c)	Retains only the elements in this collection that are contained in the specified collection (optional operation).
toArray()	Returns an array containing all of the elements in this collection.
toArray(T[] a)	Returns an array containing all of the elements in this collection; the runtime type of the returned array is that of the specified array.

METHOD	DESCRIPTION
toString()	Returns a string representation of this collection.

Methods Inherited From Interface java.util.Collection

METHOD	DESCRIPTION
parallelStream()	Returns a possibly parallel Stream with this collection as its source.
removeIf (Predicate<? super E> filter)	Removes all of the elements of this collection that satisfy the given predicate.
stream()	Returns a sequential Stream with this collection as its source.
toArray (IntFunction<T[]> generator)	Returns an array containing all of the elements in this collection, using the provided generator function to allocate the returned array.

Methods Inherited From Interface java.lang.Iterable

METHOD	DESCRIPTION
forEach (Consumer<? super T> action)	Performs the given action for each element of the Iterable until all elements have been processed or the action throws an exception.

Methods Inherited From Interface java.util.List

METHOD	DESCRIPTION
addAll(Collection<? extends E> c)	Appends all of the elements in the specified collection to the end of this list, in the order that they are returned by the specified collection's iterator (optional operation).

METHOD	DESCRIPTION
contains(Object o)	Returns true if this list contains the specified element.
containsAll(Collection<?> c)	Returns true if this list contains all of the elements of the specified collection.
isEmpty()	Returns true if this list contains no elements.
remove(Object o)	Removes the first occurrence of the specified element from this list, if it is present (optional operation).
removeAll(Collection<?> c)	Removes from this list all of its elements that are contained in the specified collection (optional operation).
replaceAll (UnaryOperator<E> operator)	Replaces each element of this list with the result of applying the operator to that element.
retainAll(Collection<?> c)	Retains only the elements in this list that are contained in the specified collection (optional operation).
size()	Returns the number of elements in this list.
sort(Comparator<? super E> c)	Sorts this list according to the order induced by the specified Comparator.
spliterator()	Creates a Spliterator over the elements in this list.
toArray()	Returns an array containing all of the elements in this list in proper sequence (from first to last element).
toArray(T[] a)	Returns an array containing all of the elements in this list in proper sequence (from first to last element); the runtime type of the returned array is that of the specified array.

Reference: https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/AbstractSequentialList.html