CopyOnWriteArraySet in Java

Last Updated : 24 Jan, 2022

CopyOnWriteArraySet is a member of the Java Collections Framework. It is a Set that uses an internal CopyOnWriteArrayList for all of its operations. It was introduced in JDK 1.5, we can say that it is a thread-safe version of Set. To use this class, we need to import it from java.util.concurrent package.

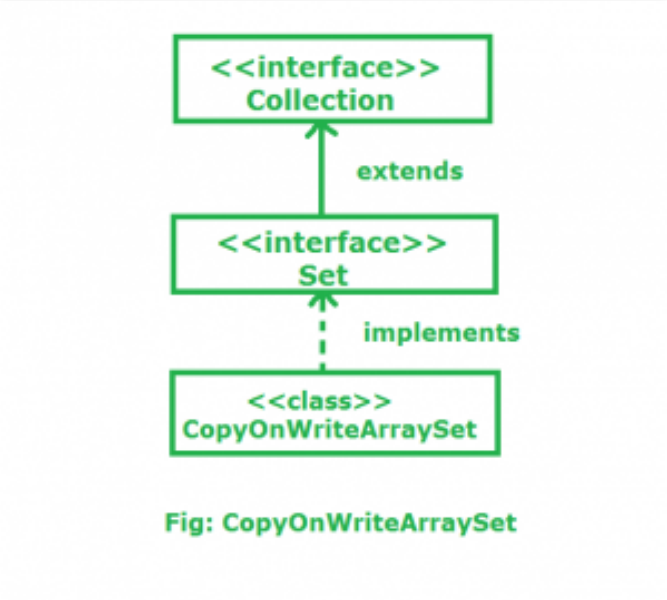

Fig: CopyOnWriteArraySet

It shares some properties of Set and also has its own properties as listed:

- The internal implementation of CopyOnWriteArraySet is CopyOnWriteArrayList only.
- Multiple Threads are able to perform update operations simultaneously but for every update operation, a separate cloned copy is created. As for every update a new cloned copy will be created which is costly. Hence if multiple update operations are required then it is not recommended to use CopyOnWriteArraySet.
- While one thread iterating the Set, other threads can perform updation, here we won't get any runtime exception like ConcurrentModificationException.
- An Iterator of CopyOnWriteArraySet class can perform only read-only and should not perform the deletion, otherwise, we will get Run-time exception UnsupportedOperationException.
- Use CopyOnWriteArraySet in applications in which set sizes generally stay small, read-only operations vastly outnumber mutative operations, and you need to prevent interference among threads during traversal.
- CopyOnWriteArraySet helps in minimizing programmer-controlled synchronization steps and moving the control to inbuilt, well-tested APIs.

Class Hierarchy is as follows:

```
java.lang.Object
   ↳ java.util.AbstractCollection<E>
        ↳ java.util.AbstractSet<E>
             ↳ java.util.concurrent.CopyOnWriteArraySet<E>
```

Syntax: Declaration

```
public class CopyOnWriteArraySet<E> extends AbstractSet<E> implements Serializ
```

Here, **E** is the type of elements stored in this Collection. It implements **Serializable**, **Iterable<E>**, **Collection<E>**, Set<E> interfaces.

Constructors of CopyOnWriteArraySet

1. CopyOnWriteArraySet(): Creates an empty set.

```
CopyOnWriteArraySet<E> c = new CopyOnWriteArraySet<E>();
```

2. CopyOnWriteArraySet(Collection c): Creates a set containing all of the elements of the specified collection.

```
CopyOnWriteArraySet<E> c = new CopyOnWriteArraySet<E>(Collection c);
```

Example:

```
// Java Program to Illustrate CopyOnWriteArraySet Class

// Importing required classes
import java.util.*;
import java.util.concurrent.*;

// Main class
class ConcurrentDemo extends Thread {

    static CopyOnWriteArraySet l
        = new CopyOnWriteArraySet();
```

```
        // Method
        public void run()
        {
            // Child thread trying to add
            // new element in the Set object
            l.add("D");
        }

        // Method 2
        // Main driver method
        public static void main(String[] args)
        {

            // Adding elements
            // using add() method
            l.add("A");
            l.add("B");
            l.add("C");

            // We create a child thread
            // that is going to modify
            // CopyOnWriteArraySet l.
            ConcurrentDemo t = new ConcurrentDemo();

            // Running the child thread
            // using start() method
            t.start();

            // Waiting for the thread to
            // add the element

            // Try block to check for exceptions
            try {

                Thread.sleep(2000);
            }

            // Catch block to handle exceptions
            catch (InterruptedException e) {

                // Print statement
                System.out.println(
                    "child going to add element");
            }

            System.out.println(l);

            // Now we iterate through the
            // CopyOnWriteArraySet and we
            // wont get exception.
            Iterator itr = l.iterator();

            while (itr.hasNext()) {

                String s = (String)itr.next();
                System.out.println(s);
```

```
            if (s.equals("C")) {

                // Here we will get
                // RuntimeException
                itr.remove();
            }
        }
    }
}
```

Output:

```
mayanksolanki@MacBook-Air Desktop % javac GFG.java
Note: GFG.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
mayanksolanki@MacBook-Air Desktop % java ConcurrentDemo
[A, B, C, D]
A
B
C
Exception in thread "main" java.lang.UnsupportedOperationException
        at java.base/java.util.concurrent.CopyOnWriteArrayList$COWIterator.remov
e(CopyOnWriteArrayList.java:1124)
        at ConcurrentDemo.main(GFG.java:51)
mayanksolanki@MacBook-Air Desktop %
```

Iterating over CopyOnWriteArraySet

We can iterate over the elements contained in this set in the order in which these elements were added using the iterator() method. The returned iterator provides an immutable snapshot of the state of the set when the iterator was constructed. Because of this property, **GeeksforGeeks** is not printed at the first iteration. Synchronization is not required while iterating. The iterator does not support the remove method.

Example:

```
// Java program to Illustrate Iterating Over
```

```
// CopyOnWriteArraySet class

// Importing required classes
import java.io.*;
import java.util.*;
import java.util.concurrent.*;

// Main class
// IteratingCopyOnWriteArraySet
class GFG {

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

        // Creating an instance of CopyOnWriteArraySet
        CopyOnWriteArraySet<String> set
            = new CopyOnWriteArraySet<>();

        // Initial an iterator
        Iterator itr = set.iterator();

        // Adding elements
        // using add() method
        set.add("GeeksforGeeks");

        // Display message only
        System.out.println("Set contains: ");

        // Printing the contents
        // of set to the console
        while (itr.hasNext())
            System.out.println(itr.next());

        // Iterator after adding an element
        itr = set.iterator();

        // Display message only
        System.out.println("Set contains:");

        // Printing the elements to the console
        while (itr.hasNext())
            System.out.println(itr.next());
    }
}
```

Output:

```
Set contains:
Set contains:
```

GeeksforGeeks

Methods in CopyOnWriteArraySet

Method	Action Performed
add(E e)	Adds the specified element to this set if it is not already present.
addAll(Collection<? extends E> c)	Adds all of the elements in the specified collection to this set if they're not already present.
clear()	Removes all of the elements from this set.
contains(Object o)	Returns true if this set contains the specified element.
containsAll(Collection<? > c)	Returns true if this set contains all of the elements of the specified collection.
equals(Object o)	Compares the specified object with this set for equality.
forEach(Consumer<? super E> action)	Performs the given action for each element of the Iterable until all elements have been processed or the action throws an exception.
isEmpty()	Returns true if this set contains no elements.
iterator()	Returns an iterator over the elements contained in this set in the order in which these elements were added.
remove(Object o)	Removes the specified element from this set if it is present.
removeAll(Collection<? > c)	Removes from this set all of its elements that are contained in the specified collection.
removeIf(Predicate<? super E> filter)	Removes all of the elements of this collection that satisfy the given predicate.
retainAll(Collection<?> c)	Retains only the elements in this set that are contained in the specified collection.
size()	Returns the number of elements in this set.

Method	Action Performed
spliterator()	Returns a Spliterator over the elements in this set in the order in which these elements were added.
toArray()	Returns an array containing all of the elements in this set.
toArray(T[] a)	Returns an array containing all of the elements in this set; the runtime type of the returned array is that of the specified array.

Methods inherited from class java.util.AbstractSet

METHOD	DESCRIPTION
hashCode()	Returns the hash code value for this set.

Methods inherited from class java.util.AbstractCollection

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.
stream()	Returns a sequential Stream with this collection as its source.

HashSet vs CopyOnWriteArraySet

PROPERTY	HashSet	CopyOnW

PROPERTY	HashSet	CopyOnV
Package	It belongs to **java.util** package	It belongs to **java.util.c**
Synchronization	Synchronization means only one thread can access or modify it. HashSet is not synchronized.	It is synchronized.
Iterators	Iterators returned my methods iterator() and listiterator() are fail-fast.	Iterators returned are f
Added In Version	It was added in JDK 1.2	It was added in JDK 1
Performance	It is fast since it is not synchronized.	It is slower compared t synchronized.
Exception	It may throw **ConcurrentModificationException** since many threads can access it simultaneously.	It does not throws **ConcurrentMo** it is synchronized.