001/\*  
002 \* Licensed to the Apache Software Foundation (ASF) under one or more  
003 \* contributor license agreements. See the NOTICE file distributed with  
004 \* this work for additional information regarding copyright ownership.  
005 \* The ASF licenses this file to You under the Apache License, Version 2.0  
006 \* (the "License"); you may not use this file except in compliance with  
007 \* the License. You may obtain a copy of the License at  
008 \*  
009 \* http://www.apache.org/licenses/LICENSE-2.0  
010 \*  
011 \* Unless required by applicable law or agreed to in writing, software  
012 \* distributed under the License is distributed on an "AS IS" BASIS,  
013 \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
014 \* See the License for the specific language governing permissions and  
015 \* limitations under the License.  
016 \*/  
017package org.apache.commons.collections4.iterators;  
018  
019import java.util.Iterator;  
020  
021import org.apache.commons.collections4.Unmodifiable;  
022  
023/\*\*  
024 \* Decorates an iterator such that it cannot be modified.  
025 \* <p>  
026 \* Attempts to modify it will result in an UnsupportedOperationException.  
027 \* </p>  
028 \*  
029 \* @since 3.0  
030 \*/  
031public final class UnmodifiableIterator<E> implements Iterator<E>, Unmodifiable {  
032  
033 /\*\* The iterator being decorated \*/  
034 private final Iterator<? extends E> iterator;  
035  
036 //-----------------------------------------------------------------------  
037 /\*\*  
038 \* Decorates the specified iterator such that it cannot be modified.  
039 \* <p>  
040 \* If the iterator is already unmodifiable it is returned directly.  
041 \*  
042 \* @param <E> the element type  
043 \* @param iterator the iterator to decorate  
044 \* @return a new unmodifiable iterator  
045 \* @throws NullPointerException if the iterator is null  
046 \*/  
047 public static <E> Iterator<E> unmodifiableIterator(final Iterator<? extends E> iterator) {  
048 if (iterator == null) {  
049 throw new NullPointerException("Iterator must not be null");  
050 }  
051 if (iterator instanceof Unmodifiable) {  
052 @SuppressWarnings("unchecked") // safe to upcast  
053 final Iterator<E> tmpIterator = (Iterator<E>) iterator;  
054 return tmpIterator;  
055 }  
056 return new UnmodifiableIterator<>(iterator);  
057 }  
058  
059 //-----------------------------------------------------------------------  
060 /\*\*  
061 \* Constructor.  
062 \*  
063 \* @param iterator the iterator to decorate  
064 \*/  
065 private UnmodifiableIterator(final Iterator<? extends E> iterator) {  
066 super();  
067 this.iterator = iterator;  
068 }  
069  
070 //-----------------------------------------------------------------------  
071 @Override  
072 public boolean hasNext() {  
073 return iterator.hasNext();  
074 }  
075  
076 @Override  
077 public E next() {  
078 return iterator.next();  
079 }  
080  
081 @Override  
082 public void remove() {  
083 throw new UnsupportedOperationException("remove() is not supported");  
084 }  
085  
086}