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 org.apache.commons.collections4.OrderedMapIterator;  
020import org.apache.commons.collections4.Unmodifiable;  
021  
022/\*\*  
023 \* Decorates an ordered map iterator such that it cannot be modified.  
024 \* <p>  
025 \* Attempts to modify it will result in an UnsupportedOperationException.  
026 \* </p>  
027 \*  
028 \* @param <K> the type of keys  
029 \* @param <V> the type of mapped values  
030 \* @since 3.0  
031 \*/  
032public final class UnmodifiableOrderedMapIterator<K, V> implements OrderedMapIterator<K, V>,  
033 Unmodifiable {  
034  
035 /\*\* The iterator being decorated \*/  
036 private final OrderedMapIterator<? extends K, ? extends V> iterator;  
037  
038 //-----------------------------------------------------------------------  
039 /\*\*  
040 \* Decorates the specified iterator such that it cannot be modified.  
041 \*  
042 \* @param <K> the key type  
043 \* @param <V> the value type  
044 \* @param iterator the iterator to decorate  
045 \* @return a new unmodifiable ordered map iterator  
046 \* @throws NullPointerException if the iterator is null  
047 \*/  
048 public static <K, V> OrderedMapIterator<K, V> unmodifiableOrderedMapIterator(  
049 final OrderedMapIterator<K, ? extends V> iterator) {  
050  
051 if (iterator == null) {  
052 throw new NullPointerException("OrderedMapIterator must not be null");  
053 }  
054 if (iterator instanceof Unmodifiable) {  
055 @SuppressWarnings("unchecked") // safe to upcast  
056 final OrderedMapIterator<K, V> tmpIterator = (OrderedMapIterator<K, V>) iterator;  
057 return tmpIterator;  
058 }  
059 return new UnmodifiableOrderedMapIterator<>(iterator);  
060 }  
061  
062 //-----------------------------------------------------------------------  
063 /\*\*  
064 \* Constructor.  
065 \*  
066 \* @param iterator the iterator to decorate  
067 \*/  
068 private UnmodifiableOrderedMapIterator(final OrderedMapIterator<K, ? extends V> iterator) {  
069 super();  
070 this.iterator = iterator;  
071 }  
072  
073 //-----------------------------------------------------------------------  
074 @Override  
075 public boolean hasNext() {  
076 return iterator.hasNext();  
077 }  
078  
079 @Override  
080 public K next() {  
081 return iterator.next();  
082 }  
083  
084 @Override  
085 public boolean hasPrevious() {  
086 return iterator.hasPrevious();  
087 }  
088  
089 @Override  
090 public K previous() {  
091 return iterator.previous();  
092 }  
093  
094 @Override  
095 public K getKey() {  
096 return iterator.getKey();  
097 }  
098  
099 @Override  
100 public V getValue() {  
101 return iterator.getValue();  
102 }  
103  
104 @Override  
105 public V setValue(final V value) {  
106 throw new UnsupportedOperationException("setValue() is not supported");  
107 }  
108  
109 @Override  
110 public void remove() {  
111 throw new UnsupportedOperationException("remove() is not supported");  
112 }  
113  
114}