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;  
018  
019/\*\*  
020 \* Defines a map that maintains order and allows both forward and backward  
021 \* iteration through that order.  
022 \*  
023 \* @param <K> the type of the keys in the map  
024 \* @param <V> the type of the values in the map  
025 \*  
026 \* @since 3.0  
027 \*/  
028public interface OrderedMap<K, V> extends IterableMap<K, V> {  
029  
030 /\*\*  
031 \* Obtains an <code>OrderedMapIterator</code> over the map.  
032 \* <p>  
033 \* A ordered map iterator is an efficient way of iterating over maps  
034 \* in both directions.  
035 \*  
036 \* @return a map iterator  
037 \*/  
038 @Override  
039 OrderedMapIterator<K, V> mapIterator();  
040  
041 /\*\*  
042 \* Gets the first key currently in this map.  
043 \*  
044 \* @return the first key currently in this map  
045 \* @throws java.util.NoSuchElementException if this map is empty  
046 \*/  
047 K firstKey();  
048  
049 /\*\*  
050 \* Gets the last key currently in this map.  
051 \*  
052 \* @return the last key currently in this map  
053 \* @throws java.util.NoSuchElementException if this map is empty  
054 \*/  
055 K lastKey();  
056  
057 /\*\*  
058 \* Gets the next key after the one specified.  
059 \*  
060 \* @param key the key to search for next from  
061 \* @return the next key, null if no match or at end  
062 \*/  
063 K nextKey(K key);  
064  
065 /\*\*  
066 \* Gets the previous key before the one specified.  
067 \*  
068 \* @param key the key to search for previous from  
069 \* @return the previous key, null if no match or at start  
070 \*/  
071 K previousKey(K key);  
072  
073}