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  
019import java.util.Collection;  
020import java.util.Set;  
021  
022/\*\*  
023 \* The "read" subset of the {@link java.util.Map} interface.  
024 \*  
025 \* @param <K> the type of the keys in this map  
026 \* @param <V> the type of the values in this map  
027 \*  
028 \* @since 4.0  
029 \* @see Put  
030 \*/  
031public interface Get<K, V> {  
032  
033 /\*\*  
034 \* @param key key whose presence in this map is to be tested  
035 \* @return <code>true</code> if this map contains a mapping for the specified  
036 \* key  
037 \* @see java.util.Map#containsKey(Object)  
038 \*/  
039 boolean containsKey(Object key);  
040  
041 /\*\*  
042 \* @param value value whose presence in this map is to be tested  
043 \* @return <code>true</code> if this map maps one or more keys to the  
044 \* specified value  
045 \* @see java.util.Map#containsValue(Object)  
046 \*/  
047 boolean containsValue(Object value);  
048  
049 /\*\*  
050 \* @return a set view of the mappings contained in this map  
051 \* @see java.util.Map#entrySet()  
052 \*/  
053 Set<java.util.Map.Entry<K, V>> entrySet();  
054  
055 /\*\*  
056 \* @param key the key whose associated value is to be returned  
057 \* @return the value to which the specified key is mapped, or  
058 \* {@code null} if this map contains no mapping for the key  
059 \* @see java.util.Map#get(Object)  
060 \*/  
061 V get(Object key);  
062  
063 /\*\*  
064 \* @param key key whose mapping is to be removed from the map  
065 \* @return the previous value associated with <code>key</code>, or  
066 \* <code>null</code> if there was no mapping for <code>key</code>.  
067 \* @see java.util.Map#remove(Object)  
068 \*/  
069 V remove(Object key);  
070  
071 /\*\*  
072 \* @return <code>true</code> if this map contains no key-value mappings  
073 \* @see java.util.Map#isEmpty()  
074 \*/  
075 boolean isEmpty();  
076  
077 /\*\*  
078 \* @return a set view of the keys contained in this map  
079 \* @see java.util.Map#keySet()  
080 \*/  
081 Set<K> keySet();  
082  
083 /\*\*  
084 \* @return the number of key-value mappings in this map  
085 \* @see java.util.Map#size()  
086 \*/  
087 int size();  
088  
089 /\*\*  
090 \* @return a collection view of the values contained in this map  
091 \* @see java.util.Map#values()  
092 \*/  
093 Collection<V> values();  
094  
095}