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.Set;  
020  
021/\*\*  
022 \* Defines a map that holds a set of values against each key.  
023 \* <p>  
024 \* A {@code SetValuedMap} is a Map with slightly different semantics:  
025 \* </p>  
026 \* <ul>  
027 \* <li>Putting a value into the map will add the value to a {@link Set} at that key.</li>  
028 \* <li>Getting a value will return a {@link Set}, holding all the values put to that key.</li>  
029 \* </ul>  
030 \*  
031 \* @param <K> the type of the keys in this map  
032 \* @param <V> the type of the values in this map  
033 \* @since 4.1  
034 \*/  
035public interface SetValuedMap<K, V> extends MultiValuedMap<K, V> {  
036  
037 /\*\*  
038 \* Gets the set of values associated with the specified key.  
039 \* <p>  
040 \* Implementations typically return an empty {@code Set} if no values  
041 \* have been mapped to the key.  
042 \* <p>  
043 \*  
044 \* @param key the key to retrieve  
045 \* @return the {@code Set} of values, implementations should return an  
046 \* empty {@code Set} for no mapping  
047 \* @throws NullPointerException if the key is null and null keys are invalid  
048 \*/  
049 @Override  
050 Set<V> get(K key);  
051  
052 /\*\*  
053 \* Removes all values associated with the specified key.  
054 \* <p>  
055 \* The returned set <i>may</i> be modifiable, but updates will not be  
056 \* propagated to this set-valued map. In case no mapping was stored for the  
057 \* specified key, an empty, unmodifiable set will be returned.  
058 \*  
059 \* @param key the key to remove values from  
060 \* @return the {@code Set} of values removed, implementations should  
061 \* return null for no mapping found, but may return an empty collection  
062 \* @throws UnsupportedOperationException if the map is unmodifiable  
063 \* @throws NullPointerException if the key is null and null keys are invalid  
064 \*/  
065 @Override  
066 Set<V> remove(Object key);  
067}