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.set;  
018  
019import java.util.Set;  
020  
021import org.apache.commons.collections4.collection.AbstractCollectionDecorator;  
022  
023/\*\*  
024 \* Decorates another <code>Set</code> to provide additional behaviour.  
025 \* <p>  
026 \* Methods are forwarded directly to the decorated set.  
027 \* </p>  
028 \*  
029 \* @param <E> the type of the elements in this set  
030 \* @since 3.0  
031 \*/  
032public abstract class AbstractSetDecorator<E> extends AbstractCollectionDecorator<E> implements  
033 Set<E> {  
034  
035 /\*\* Serialization version \*/  
036 private static final long serialVersionUID = -4678668309576958546L;  
037  
038 /\*\*  
039 \* Constructor only used in deserialization, do not use otherwise.  
040 \* @since 3.1  
041 \*/  
042 protected AbstractSetDecorator() {  
043 super();  
044 }  
045  
046 /\*\*  
047 \* Constructor that wraps (not copies).  
048 \*  
049 \* @param set the set to decorate, must not be null  
050 \* @throws NullPointerException if set is null  
051 \*/  
052 protected AbstractSetDecorator(final Set<E> set) {  
053 super(set);  
054 }  
055  
056 /\*\*  
057 \* Gets the set being decorated.  
058 \*  
059 \* @return the decorated set  
060 \*/  
061 @Override  
062 protected Set<E> decorated() {  
063 return (Set<E>) super.decorated();  
064 }  
065  
066 @Override  
067 public boolean equals(final Object object) {  
068 return object == this || decorated().equals(object);  
069 }  
070  
071 @Override  
072 public int hashCode() {  
073 return decorated().hashCode();  
074 }  
075  
076}