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.functors;  
018  
019import java.io.Serializable;  
020  
021import org.apache.commons.collections4.Transformer;  
022  
023/\*\*  
024 \* Transformer implementation that returns the result of calling  
025 \* <code>String.valueOf</code> on the input object.  
026 \*  
027 \* @since 3.0  
028 \*/  
029public final class StringValueTransformer<T> implements Transformer<T, String>, Serializable {  
030  
031 /\*\* Serial version UID \*/  
032 private static final long serialVersionUID = 7511110693171758606L;  
033  
034 /\*\* Singleton predicate instance \*/  
035 private static final Transformer<Object, String> INSTANCE = new StringValueTransformer<>();  
036  
037 /\*\*  
038 \* Factory returning the singleton instance.  
039 \*  
040 \* @param <T> the input type  
041 \* @return the singleton instance  
042 \* @since 3.1  
043 \*/  
044 @SuppressWarnings("unchecked")  
045 public static <T> Transformer<T, String> stringValueTransformer() {  
046 return (Transformer<T, String>) INSTANCE;  
047 }  
048  
049 /\*\*  
050 \* Restricted constructor.  
051 \*/  
052 private StringValueTransformer() {  
053 super();  
054 }  
055  
056 /\*\*  
057 \* Transforms the input to result by calling <code>String.valueOf</code>.  
058 \*  
059 \* @param input the input object to transform  
060 \* @return the transformed result  
061 \*/  
062 @Override  
063 public String transform(final T input) {  
064 return String.valueOf(input);  
065 }  
066  
067 private Object readResolve() {  
068 return INSTANCE;  
069 }  
070  
071}