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.Predicate;  
022import org.apache.commons.collections4.Transformer;  
023  
024/\*\*  
025 \* Transformer implementation that calls a Predicate using the input object  
026 \* and then returns the result.  
027 \*  
028 \* @since 3.0  
029 \*/  
030public class PredicateTransformer<T> implements Transformer<T, Boolean>, Serializable {  
031  
032 /\*\* Serial version UID \*/  
033 private static final long serialVersionUID = 5278818408044349346L;  
034  
035 /\*\* The closure to wrap \*/  
036 private final Predicate<? super T> iPredicate;  
037  
038 /\*\*  
039 \* Factory method that performs validation.  
040 \*  
041 \* @param <T> the input type  
042 \* @param predicate the predicate to call, not null  
043 \* @return the <code>predicate</code> transformer  
044 \* @throws IllegalArgumentException if the predicate is null  
045 \*/  
046 public static <T> Transformer<T, Boolean> predicateTransformer(final Predicate<? super T> predicate) {  
047 if (predicate == null) {  
048 throw new IllegalArgumentException("Predicate must not be null");  
049 }  
050 return new PredicateTransformer<>(predicate);  
051 }  
052  
053 /\*\*  
054 \* Constructor that performs no validation.  
055 \* Use <code>predicateTransformer</code> if you want that.  
056 \*  
057 \* @param predicate the predicate to call, not null  
058 \*/  
059 public PredicateTransformer(final Predicate<? super T> predicate) {  
060 super();  
061 iPredicate = predicate;  
062 }  
063  
064 /\*\*  
065 \* Transforms the input to result by calling a predicate.  
066 \*  
067 \* @param input the input object to transform  
068 \* @return the transformed result  
069 \*/  
070 @Override  
071 public Boolean transform(final T input) {  
072 return Boolean.valueOf(iPredicate.evaluate(input));  
073 }  
074  
075 /\*\*  
076 \* Gets the predicate.  
077 \*  
078 \* @return the predicate  
079 \* @since 3.1  
080 \*/  
081 public Predicate<? super T> getPredicate() {  
082 return iPredicate;  
083 }  
084  
085}