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;  
022  
023/\*\*  
024 \* Predicate implementation that returns true if either of the predicates return true.  
025 \*  
026 \* @since 3.0  
027 \*/  
028public final class OrPredicate<T> implements PredicateDecorator<T>, Serializable {  
029  
030 /\*\* Serial version UID \*/  
031 private static final long serialVersionUID = -8791518325735182855L;  
032  
033 /\*\* The array of predicates to call \*/  
034 private final Predicate<? super T> iPredicate1;  
035 /\*\* The array of predicates to call \*/  
036 private final Predicate<? super T> iPredicate2;  
037  
038 /\*\*  
039 \* Factory to create the predicate.  
040 \*  
041 \* @param <T> the type that the predicate queries  
042 \* @param predicate1 the first predicate to check, not null  
043 \* @param predicate2 the second predicate to check, not null  
044 \* @return the <code>and</code> predicate  
045 \* @throws NullPointerException if either predicate is null  
046 \*/  
047 public static <T> Predicate<T> orPredicate(final Predicate<? super T> predicate1,  
048 final Predicate<? super T> predicate2) {  
049 if (predicate1 == null || predicate2 == null) {  
050 throw new NullPointerException("Predicate must not be null");  
051 }  
052 return new OrPredicate<>(predicate1, predicate2);  
053 }  
054  
055 /\*\*  
056 \* Constructor that performs no validation.  
057 \* Use <code>orPredicate</code> if you want that.  
058 \*  
059 \* @param predicate1 the first predicate to check, not null  
060 \* @param predicate2 the second predicate to check, not null  
061 \*/  
062 public OrPredicate(final Predicate<? super T> predicate1, final Predicate<? super T> predicate2) {  
063 super();  
064 iPredicate1 = predicate1;  
065 iPredicate2 = predicate2;  
066 }  
067  
068 /\*\*  
069 \* Evaluates the predicate returning true if either predicate returns true.  
070 \*  
071 \* @param object the input object  
072 \* @return true if either decorated predicate returns true  
073 \*/  
074 @Override  
075 public boolean evaluate(final T object) {  
076 return iPredicate1.evaluate(object) || iPredicate2.evaluate(object);  
077 }  
078  
079 /\*\*  
080 \* Gets the two predicates being decorated as an array.  
081 \*  
082 \* @return the predicates  
083 \* @since 3.1  
084 \*/  
085 @Override  
086 @SuppressWarnings("unchecked")  
087 public Predicate<? super T>[] getPredicates() {  
088 return new Predicate[] {iPredicate1, iPredicate2};  
089 }  
090  
091}