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 static org.apache.commons.collections4.functors.FunctorUtils.coerce;  
020import static org.apache.commons.collections4.functors.FunctorUtils.validate;  
021import static org.apache.commons.collections4.functors.TruePredicate.truePredicate;  
022  
023import java.util.Collection;  
024  
025import org.apache.commons.collections4.Predicate;  
026  
027/\*\*  
028 \* Predicate implementation that returns true if all the  
029 \* predicates return true.  
030 \* If the array of predicates is empty, then this predicate returns true.  
031 \* <p>  
032 \* NOTE: In versions prior to 3.2 an array size of zero or one  
033 \* threw an exception.  
034 \* </p>  
035 \*  
036 \* @since 3.0  
037 \*/  
038public final class AllPredicate<T> extends AbstractQuantifierPredicate<T> {  
039  
040 /\*\* Serial version UID \*/  
041 private static final long serialVersionUID = -3094696765038308799L;  
042  
043 /\*\*  
044 \* Factory to create the predicate.  
045 \* <p>  
046 \* If the array is size zero, the predicate always returns true.  
047 \* If the array is size one, then that predicate is returned.  
048 \*  
049 \* @param <T> the type that the predicate queries  
050 \* @param predicates the predicates to check, cloned, not null  
051 \* @return the <code>all</code> predicate  
052 \* @throws NullPointerException if the predicates array is null  
053 \* @throws NullPointerException if any predicate in the array is null  
054 \*/  
055 public static <T> Predicate<T> allPredicate(final Predicate<? super T>... predicates) {  
056 validate(predicates);  
057 if (predicates.length == 0) {  
058 return truePredicate();  
059 }  
060 if (predicates.length == 1) {  
061 return coerce(predicates[0]);  
062 }  
063  
064 return new AllPredicate<>(FunctorUtils.copy(predicates));  
065 }  
066  
067 /\*\*  
068 \* Factory to create the predicate.  
069 \* <p>  
070 \* If the collection is size zero, the predicate always returns true.  
071 \* If the collection is size one, then that predicate is returned.  
072 \*  
073 \* @param <T> the type that the predicate queries  
074 \* @param predicates the predicates to check, cloned, not null  
075 \* @return the <code>all</code> predicate  
076 \* @throws NullPointerException if the predicates array is null  
077 \* @throws NullPointerException if any predicate in the array is null  
078 \*/  
079 public static <T> Predicate<T> allPredicate(final Collection<? extends Predicate<? super T>> predicates) {  
080 final Predicate<? super T>[] preds = validate(predicates);  
081 if (preds.length == 0) {  
082 return truePredicate();  
083 }  
084 if (preds.length == 1) {  
085 return coerce(preds[0]);  
086 }  
087 return new AllPredicate<>(preds);  
088 }  
089  
090 /\*\*  
091 \* Constructor that performs no validation.  
092 \* Use <code>allPredicate</code> if you want that.  
093 \*  
094 \* @param predicates the predicates to check, not cloned, not null  
095 \*/  
096 public AllPredicate(final Predicate<? super T>... predicates) {  
097 super(predicates);  
098 }  
099  
100 /\*\*  
101 \* Evaluates the predicate returning true if all predicates return true.  
102 \*  
103 \* @param object the input object  
104 \* @return true if all decorated predicates return true  
105 \*/  
106 @Override  
107 public boolean evaluate(final T object) {  
108 for (final Predicate<? super T> iPredicate : iPredicates) {  
109 if (!iPredicate.evaluate(object)) {  
110 return false;  
111 }  
112 }  
113 return true;  
114 }  
115  
116}