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.Equator;  
022import org.apache.commons.collections4.Predicate;  
023  
024/\*\*  
025 \* Predicate implementation that returns true if the input is the same object  
026 \* as the one stored in this predicate by equals.  
027 \*  
028 \* @since 3.0  
029 \*/  
030public final class EqualPredicate<T> implements Predicate<T>, Serializable {  
031  
032 /\*\* Serial version UID \*/  
033 private static final long serialVersionUID = 5633766978029907089L;  
034  
035 /\*\* The value to compare to \*/  
036 private final T iValue;  
037  
038 /\*\* The equator to use for comparison \*/  
039 private final Equator<T> equator;  
040  
041 /\*\*  
042 \* Factory to create the predicate.  
043 \*  
044 \* @param <T> the type that the predicate queries  
045 \* @param object the object to compare to  
046 \* @return the predicate  
047 \*/  
048 public static <T> Predicate<T> equalPredicate(final T object) {  
049 if (object == null) {  
050 return NullPredicate.nullPredicate();  
051 }  
052 return new EqualPredicate<>(object);  
053 }  
054  
055 /\*\*  
056 \* Factory to create the identity predicate.  
057 \*  
058 \* @param <T> the type that the predicate queries  
059 \* @param object the object to compare to  
060 \* @param equator the equator to use for comparison  
061 \* @return the predicate  
062 \* @since 4.0  
063 \*/  
064 public static <T> Predicate<T> equalPredicate(final T object, final Equator<T> equator) {  
065 if (object == null) {  
066 return NullPredicate.nullPredicate();  
067 }  
068 return new EqualPredicate<>(object, equator);  
069 }  
070  
071 /\*\*  
072 \* Constructor that performs no validation.  
073 \* Use <code>equalPredicate</code> if you want that.  
074 \*  
075 \* @param object the object to compare to  
076 \*/  
077 public EqualPredicate(final T object) {  
078 // do not use the DefaultEquator to keep backwards compatibility  
079 // the DefaultEquator returns also true if the two object references are equal  
080 this(object, null);  
081 }  
082  
083 /\*\*  
084 \* Constructor that performs no validation.  
085 \* Use <code>equalPredicate</code> if you want that.  
086 \*  
087 \* @param object the object to compare to  
088 \* @param equator the equator to use for comparison  
089 \* @since 4.0  
090 \*/  
091 public EqualPredicate(final T object, final Equator<T> equator) {  
092 super();  
093 iValue = object;  
094 this.equator = equator;  
095 }  
096  
097 /\*\*  
098 \* Evaluates the predicate returning true if the input equals the stored value.  
099 \*  
100 \* @param object the input object  
101 \* @return true if input object equals stored value  
102 \*/  
103 @Override  
104 public boolean evaluate(final T object) {  
105 if (equator != null) {  
106 return equator.equate(iValue, object);  
107 }  
108 return iValue.equals(object);  
109 }  
110  
111 /\*\*  
112 \* Gets the value.  
113 \*  
114 \* @return the value  
115 \* @since 3.1  
116 \*/  
117 public Object getValue() {  
118 return iValue;  
119 }  
120  
121}