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 \*/  
017  
018package org.apache.commons.beanutils;  
019  
020import org.apache.commons.collections.Predicate;  
021import org.apache.commons.logging.Log;  
022import org.apache.commons.logging.LogFactory;  
023  
024import java.lang.reflect.InvocationTargetException;  
025  
026/\*\*  
027 \* <p>Predicate implementation that applies the given <code>Predicate</code>  
028 \* to the result of calling the given property getter.  
029 \* </p>  
030 \*  
031 \* @version $Id$  
032 \*/  
033public class BeanPredicate implements Predicate {  
034  
035 private final Log log = LogFactory.getLog(this.getClass());  
036  
037 /\*\* Name of the property whose value will be predicated \*/  
038 private String propertyName;  
039 /\*\* <code>Predicate</code> to be applied to the property value \*/  
040 private Predicate predicate;  
041  
042 /\*\*  
043 \* Constructs a <code>BeanPredicate</code> that applies the given  
044 \* <code>Predicate</code> to the named property value.  
045 \* @param propertyName the name of the property whose value is to be predicated,  
046 \* not null  
047 \* @param predicate the <code>Predicate</code> to be applied,  
048 \* not null  
049 \*/  
050 public BeanPredicate(final String propertyName, final Predicate predicate) {  
051 this.propertyName = propertyName;  
052 this.predicate = predicate;  
053 }  
054  
055 /\*\*  
056 \* Evaluates the given object by applying the {@link #getPredicate()}  
057 \* to a property value named by {@link #getPropertyName()}.  
058 \*  
059 \* @param object The object being evaluated  
060 \* @return the result of the predicate evaluation  
061 \* @throws IllegalArgumentException when the property cannot be evaluated  
062 \*/  
063 public boolean evaluate(final Object object) {  
064  
065 boolean evaluation = false;  
066  
067 try {  
068 final Object propValue = PropertyUtils.getProperty( object, propertyName );  
069 evaluation = predicate.evaluate(propValue);  
070 } catch (final IllegalArgumentException e) {  
071 final String errorMsg = "Problem during evaluation.";  
072 log.error("ERROR: " + errorMsg, e);  
073 throw e;  
074 } catch (final IllegalAccessException e) {  
075 final String errorMsg = "Unable to access the property provided.";  
076 log.error(errorMsg, e);  
077 throw new IllegalArgumentException(errorMsg);  
078 } catch (final InvocationTargetException e) {  
079 final String errorMsg = "Exception occurred in property's getter";  
080 log.error(errorMsg, e);  
081 throw new IllegalArgumentException(errorMsg);  
082 } catch (final NoSuchMethodException e) {  
083 final String errorMsg = "Property not found.";  
084 log.error(errorMsg, e);  
085 throw new IllegalArgumentException(errorMsg);  
086 }  
087  
088 return evaluation;  
089 }  
090  
091 /\*\*  
092 \* Gets the name of the property whose value is to be predicated.  
093 \* in the evaluation.  
094 \* @return the property name, not null  
095 \*/  
096 public String getPropertyName() {  
097 return propertyName;  
098 }  
099  
100 /\*\*  
101 \* Sets the name of the property whose value is to be predicated.  
102 \* @param propertyName the name of the property whose value is to be predicated,  
103 \* not null  
104 \*/  
105 public void setPropertyName(final String propertyName) {  
106 this.propertyName = propertyName;  
107 }  
108  
109 /\*\*  
110 \* Gets the <code>Predicate</code> to be applied to the value of the named property  
111 \* during {@link #evaluate}.  
112 \* @return <code>Predicate</code>, not null  
113 \*/  
114 public Predicate getPredicate() {  
115 return predicate;  
116 }  
117  
118 /\*\*  
119 \* Sets the <code>Predicate</code> to be applied to the value of the named property  
120 \* during {@link #evaluate(Object)}.  
121 \* @param predicate <code>Predicate</code>, not null  
122 \*/  
123 public void setPredicate(final Predicate predicate) {  
124 this.predicate = predicate;  
125 }  
126  
127}