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.locale.converters;  
019  
020import java.sql.Timestamp;  
021import java.text.ParseException;  
022import java.util.Locale;  
023  
024/\*\*  
025 \* <p>Standard {@link org.apache.commons.beanutils.locale.LocaleConverter}  
026 \* implementation that converts an incoming  
027 \* locale-sensitive String into a <code>java.sql.Timestamp</code> object,  
028 \* optionally using a default value or throwing a  
029 \* {@link org.apache.commons.beanutils.ConversionException}  
030 \* if a conversion error occurs.</p>  
031 \*  
032 \* @version $Id$  
033 \*/  
034  
035public class SqlTimestampLocaleConverter extends DateLocaleConverter {  
036  
037  
038 // ----------------------------------------------------------- Constructors  
039  
040 /\*\*  
041 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
042 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
043 \* if a conversion error occurs. The locale is the default locale for  
044 \* this instance of the Java Virtual Machine and an unlocalized pattern is used  
045 \* for the convertion.  
046 \*  
047 \*/  
048 public SqlTimestampLocaleConverter() {  
049  
050 this(false);  
051 }  
052  
053 /\*\*  
054 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
055 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
056 \* if a conversion error occurs. The locale is the default locale for  
057 \* this instance of the Java Virtual Machine.  
058 \*  
059 \* @param locPattern Indicate whether the pattern is localized or not  
060 \*/  
061 public SqlTimestampLocaleConverter(final boolean locPattern) {  
062  
063 this(Locale.getDefault(), locPattern);  
064 }  
065  
066 /\*\*  
067 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
068 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
069 \* if a conversion error occurs. An unlocalized pattern is used for the convertion.  
070 \*  
071 \* @param locale The locale  
072 \*/  
073 public SqlTimestampLocaleConverter(final Locale locale) {  
074  
075 this(locale, (String) null);  
076 }  
077  
078 /\*\*  
079 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
080 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
081 \* if a conversion error occurs.  
082 \*  
083 \* @param locale The locale  
084 \* @param locPattern Indicate whether the pattern is localized or not  
085 \*/  
086 public SqlTimestampLocaleConverter(final Locale locale, final boolean locPattern) {  
087  
088 this(locale, (String) null);  
089 }  
090  
091 /\*\*  
092 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
093 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
094 \* if a conversion error occurs. An unlocalized pattern is used for the convertion.  
095 \*  
096 \* @param locale The locale  
097 \* @param pattern The convertion pattern  
098 \*/  
099 public SqlTimestampLocaleConverter(final Locale locale, final String pattern) {  
100  
101 this(locale, pattern, false);  
102 }  
103  
104 /\*\*  
105 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
106 \* that will throw a {@link org.apache.commons.beanutils.ConversionException}  
107 \* if a conversion error occurs.  
108 \*  
109 \* @param locale The locale  
110 \* @param pattern The convertion pattern  
111 \* @param locPattern Indicate whether the pattern is localized or not  
112 \*/  
113 public SqlTimestampLocaleConverter(final Locale locale, final String pattern, final boolean locPattern) {  
114  
115 super(locale, pattern, locPattern);  
116 }  
117  
118 /\*\*  
119 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
120 \* that will return the specified default value  
121 \* if a conversion error occurs. The locale is the default locale for  
122 \* this instance of the Java Virtual Machine and an unlocalized pattern is used  
123 \* for the convertion.  
124 \*  
125 \* @param defaultValue The default value to be returned  
126 \*/  
127 public SqlTimestampLocaleConverter(final Object defaultValue) {  
128 this(defaultValue, false);  
129 }  
130  
131 /\*\*  
132 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
133 \* that will return the specified default value  
134 \* if a conversion error occurs. The locale is the default locale for  
135 \* this instance of the Java Virtual Machine.  
136 \*  
137 \* @param defaultValue The default value to be returned  
138 \* @param locPattern Indicate whether the pattern is localized or not  
139 \*/  
140 public SqlTimestampLocaleConverter(final Object defaultValue, final boolean locPattern) {  
141  
142 this(defaultValue, Locale.getDefault(), locPattern);  
143 }  
144  
145 /\*\*  
146 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
147 \* that will return the specified default value  
148 \* if a conversion error occurs. An unlocalized pattern is used for the convertion.  
149 \*  
150 \* @param defaultValue The default value to be returned  
151 \* @param locale The locale  
152 \*/  
153 public SqlTimestampLocaleConverter(final Object defaultValue, final Locale locale) {  
154  
155 this(defaultValue, locale, false);  
156 }  
157  
158 /\*\*  
159 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
160 \* that will return the specified default value  
161 \* if a conversion error occurs.  
162 \*  
163 \* @param defaultValue The default value to be returned  
164 \* @param locale The locale  
165 \* @param locPattern Indicate whether the pattern is localized or not  
166 \*/  
167 public SqlTimestampLocaleConverter(final Object defaultValue, final Locale locale, final boolean locPattern) {  
168  
169 this(defaultValue, locale, null, locPattern);  
170 }  
171  
172 /\*\*  
173 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
174 \* that will return the specified default value  
175 \* if a conversion error occurs. An unlocalized pattern is used for the convertion.  
176 \*  
177 \* @param defaultValue The default value to be returned  
178 \* @param locale The locale  
179 \* @param pattern The convertion pattern  
180 \*/  
181 public SqlTimestampLocaleConverter(final Object defaultValue, final Locale locale, final String pattern) {  
182  
183 this(defaultValue, locale, pattern, false);  
184 }  
185  
186 /\*\*  
187 \* Create a {@link org.apache.commons.beanutils.locale.LocaleConverter}  
188 \* that will return the specified default value  
189 \* if a conversion error occurs.  
190 \*  
191 \* @param defaultValue The default value to be returned  
192 \* @param locale The locale  
193 \* @param pattern The convertion pattern  
194 \* @param locPattern Indicate whether the pattern is localized or not  
195 \*/  
196 public SqlTimestampLocaleConverter(final Object defaultValue, final Locale locale, final String pattern, final boolean locPattern) {  
197  
198 super(defaultValue, locale, pattern, locPattern);  
199 }  
200  
201 // --------------------------------------------------------- Methods  
202  
203 /\*\*  
204 \* Convert the specified locale-sensitive input object into an output object of the  
205 \* specified type.  
206 \*  
207 \* @param value The input object to be converted  
208 \* @param pattern The pattern is used for the convertion  
209 \* @return The converted value  
210 \*  
211 \* @throws org.apache.commons.beanutils.ConversionException if conversion  
212 \* cannot be performed successfully  
213 \* @throws ParseException if an error occurs parsing a String to a Number  
214 \*/  
215 @Override  
216 protected Object parse(final Object value, final String pattern) throws ParseException {  
217  
218 return new Timestamp(((java.util.Date) super.parse(value, pattern)).getTime());  
219 }  
220}