```
 1 /*
 2  *
 3  * MariaDB Client for Java
 4  *
 5  * Copyright (c) 2012-2014 Monty Program Ab.
 6  * Copyright (c) 2015-2019 MariaDB Ab.
 7  *
 8  * This library is free software; you can redistribute it
   and/or modify it under
 9  * the terms of the GNU Lesser General Public License as
   published by the Free
10  * Software Foundation; either version 2.1 of the License,
   or (at your option)
11  * any later version.
12  *
13  * This library is distributed in the hope that it will be
   useful, but
14  * WITHOUT ANY WARRANTY; without even the implied warranty
   of MERCHANTABILITY or
15  * FITNESS FOR A PARTICULAR PURPOSE.  See the GNU Lesser
   General Public License
16  * for more details.
17  *
18  * You should have received a copy of the GNU Lesser
   General Public License along
19  * with this library; if not, write to Monty Program Ab
   info@montyprogram.com.
20  *
21  * This particular MariaDB Client for Java file is work
22  * derived from a Drizzle-JDBC. Drizzle-JDBC file which is
   covered by subject to
23  * the following copyright and notice provisions:
24  *
25  * Copyright (c) 2009-2011, Marcus Eriksson
26  *
27  * Redistribution and use in source and binary forms, with
   or without modification,
28  * are permitted provided that the following conditions are
    met:
29  * Redistributions of source code must retain the above
   copyright notice, this list
30  * of conditions and the following disclaimer.
31  *
32  * Redistributions in binary form must reproduce the above
   copyright notice, this
33  * list of conditions and the following disclaimer in the
```

```
33 documentation and/or
34  * other materials provided with the distribution.
35  *
36  * Neither the name of the driver nor the names of its
   contributors may not be
37  * used to endorse or promote products derived from this
   software without specific
38  * prior written permission.
39  *
40  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS  AND
   CONTRIBUTORS "AS IS"
41  * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT
   NOT LIMITED TO, THE IMPLIED
42  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
   PARTICULAR PURPOSE ARE DISCLAIMED.
43  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS
   BE LIABLE FOR ANY DIRECT,
44  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
   CONSEQUENTIAL DAMAGES (INCLUDING, BUT
45  * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
   SERVICES; LOSS OF USE, DATA, OR
46  * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
    ANY THEORY OF LIABILITY,
47  * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (
   INCLUDING NEGLIGENCE OR OTHERWISE)
48  * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
    IF ADVISED OF THE POSSIBILITY
49  * OF SUCH DAMAGE.
50  *
51  */
52
53 package org.mariadb.jdbc;
54
55 import org.mariadb.jdbc.internal.util.*;
56 import org.mariadb.jdbc.internal.util.constant.*;
57 import org.mariadb.jdbc.util.*;
58
59 import java.lang.reflect.*;
60 import java.sql.*;
61 import java.util.*;
62
63 public final class Driver implements java.sql.Driver {
64
65   static {
66     try {
67       DriverManager.registerDriver(new Driver(), new
```

```
 67 DeRegister());
 68     } catch (SQLException e) {
 69       throw new RuntimeException("Could not register
    driver", e);
 70     }
 71   }
 72
 73   /**
 74    * Connect to the given connection string.
 75    *
 76    * @param url the url to connect to
 77    * @return a connection
 78    * @throws SQLException if it is not possible to connect
 79    */
 80   public Connection connect(final String url, final
    Properties props) throws SQLException {
 81
 82     UrlParser urlParser = UrlParser.parse(url, props);
 83     if (urlParser == null || urlParser.getHostAddresses
    () == null) {
 84       return null;
 85     } else {
 86       return MariaDbConnection.newConnection(urlParser,
    null);
 87     }
 88   }
 89
 90   /**
 91    * returns true if the driver can accept the url.
 92    *
 93    * @param url the url to test
 94    * @return true if the url is valid for this driver
 95    */
 96   @Override
 97   public boolean acceptsURL(String url) {
 98     return UrlParser.acceptsUrl(url);
 99   }
100
101   /**
102    * Get the property info.
103    *
104    * @param url the url to get properties for
105    * @param info the info props
106    * @return all possible connector options
107    * @throws SQLException if there is a problem getting
    the property info
```

```
108     */
109   public DriverPropertyInfo[] getPropertyInfo(String url,
    Properties info) throws SQLException {
110     Options options;
111     if (url != null && !url.isEmpty()) {
112       UrlParser urlParser = UrlParser.parse(url, info);
113       if (urlParser == null || urlParser.getOptions() ==
    null) {
114         return new DriverPropertyInfo[0];
115       }
116       options = urlParser.getOptions();
117     } else {
118       options = DefaultOptions.parse(HaMode.NONE, "", info
    , null);
119     }
120
121     List<DriverPropertyInfo> props = new ArrayList<>();
122     for (DefaultOptions o : DefaultOptions.values()) {
123       try {
124         Field field = Options.class.getField(o.
    getOptionName());
125         Object value = field.get(options);
126         DriverPropertyInfo propertyInfo =
127             new DriverPropertyInfo(field.getName(), value
     == null ? null : value.toString());
128         propertyInfo.description = o.getDescription();
129         propertyInfo.required = o.isRequired();
130         props.add(propertyInfo);
131       } catch (NoSuchFieldException |
    IllegalAccessException e) {
132         // eat error
133       }
134     }
135     return props.toArray(new DriverPropertyInfo[props.size
    ()]);
136   }
137
138   /**
139    * gets the major version of the driver.
140    *
141    * @return the major versions
142    */
143   public int getMajorVersion() {
144     return Version.majorVersion;
145   }
146
```

```
147   /**
148    * gets the minor version of the driver.
149    *
150    * @return the minor version
151    */
152   public int getMinorVersion() {
153     return Version.minorVersion;
154   }
155
156   /**
157    * checks if the driver is jdbc compliant.
158    *
159    * @return true since the driver is not compliant
160    */
161   public boolean jdbcCompliant() {
162     return true;
163   }
164
165   public java.util.logging.Logger getParentLogger() throws
     SQLFeatureNotSupportedException {
166     throw new SQLFeatureNotSupportedException("Use logging
     parameters for enabling logging.");
167   }
168 }
169
```