Table of Contents

Our DBAccess supports a simple configurable connection pool with ConnectionPoolProvider.

The ConnectionPoolProvider creates a default DataSource and an IConnectionPool.

We currently use Hikari and Tomcat for our connection pool implementation. With Hikari we have an independent solution and Tomcat will only work on Tomcat application servers.

Configuration

The connection pool can be enabled by simply calling `setConnectionPoolEnabled(true)` on DBAccess, or it can be enabled in config.xml.

Example:

```
<application>
   ...
   <connectionpool enabled = "true"/>
   ...
</application>
```

We support different use-cases:

1. Implementing getDataSource() in application life cycle object (Application.java)
2. Configure specific IDataSourceCreator in config.xml, e.g.

   ```
     <connectionpool enabled = "true">
   <dataSourceCreatorClass>com.sibvisions.rad.persist.jdbc.datasource.Tomc
   atDataSourceCreator</dataSourceCreatorClass>
     </connectionpool>
   ```

3. Configure ServiceLoader for IDataSourceCreator
4. Register custom IDataSourceCreator in ConnectionPoolProvider, e.g.
   `ConnectionPoolProvider.registerDataSourceCreator(CustomDataSourceCreator.class);`
   There are 2 IDataSourceCreator registered by default:
   * HikariDataSourceCreator
   * TomcatDataSourceCreator

The default implementations of IDataSourceCreator supports several parameter by default. The following example config.xml shows all supported parameters.

```
<application>
   ...
   <connectionpool enabled = "true">
     <!-- ---- General parameters ---- -->

     <!-- dataSourceCreatorClass is optional, HikaraDataSourceCreator or
TomcatDataSourceCreator will be used by default depending on availability.
<dataSourceCreatorClass>com.sibvisions.rad.persist.jdbc.datasource.TomcatDat
aSourceCreator</dataSourceCreatorClass> -->
     <!-- maxActive is optional, 80 will be used by default.
```

```
     <maxActive>80</maxActive> -->
     <!-- minIdle is optional, 10 will be used by default.
     <minIdle>10</minIdle> -->
     <!-- useAliveQuery is optional, true sets DBAccess.getAliveQuery(),
false is default, therefore either validationQuery or connection.isValid()
will be used.
     <useAliveQuery>true</useAliveQuery> -->
     <!-- validationQuery is optional, if it is not set, Hikari and Tomcat
pool uses connection.isValid().
     <validationQuery>select 1</validationQuery> -->
     <!-- validationQueryTimeout is optional, in seconds, if it is not set,
Hikari and Tomcat default values are used.
     <validationQueryTimeout>3</validationQueryTimeout> -->

     <!-- ---- Hikari specific parameters ---- -->

     <!-- validationTimeout is optional, in milliseconds, if it is not set,
validationQueryTimeout or Hikari default value is used.
     <validationTimeout>1000</validationTimeout> -->
     <!-- keepaliveTime is optional, in milliseconds, if it is not set,
Hikari default value is used.
     <keepaliveTime>1000</keepaliveTime> -->

     <!-- ---- Tomcat specific parameters ---- -->

     <!-- maxIdle is optional, if it is not set, minIdle is used.
     <maxIdle>10</maxIdle> -->
     <!-- initialSize is optional, if it is not set, minIdle is used.
     <initialSize>10</initialSize> -->
     <!-- testWhileIdle is optional, if it is not set, false is used.
     <testWhileIdle>true</testWhileIdle> -->
     <!-- testOnBorrow is optional, if it is not set, true is used.
     <testOnBorrow>true</testOnBorrow> -->
     <!-- testOnReturn is optional, if it is not set, false is used.
     <testOnReturn>true</testOnReturn> -->
     <!-- validationInterval is optional, in milliseconds, if it is not set,
Tomcat default value is used.
     <validationInterval>3000</validationInterval> -->
     <!-- timeBetweenEvictionRunsMillis is optional, in milliseconds, if it
is not set, Tomcat default value is used.
     <timeBetweenEvictionRunsMillis>5000</timeBetweenEvictionRunsMillis> -->
     <!-- minEvictableIdleTimeMillis is optional, in milliseconds, if it is
not set, Tomcat default value is used.
     <minEvictableIdleTimeMillis>3000</minEvictableIdleTimeMillis> -->
     <!-- removeAbandoned is optional, if it is not set, Tomcat default
value is used.
     <removeAbandoned>true</removeAbandoned> -->
     <!-- removeAbandonedTimeout is optional, in seconds, if it is not set,
Tomcat default value is used.
     <removeAbandonedTimeout>3000</removeAbandonedTimeout> -->
```

```
   </connectionpool>
   ...
</application>
```