Moab Viewpoint

[Home](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/index.php) [Up](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/index.php) [Previous](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/installing_viewpoint.php) [Next](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/configuring_the_connection_to_moab.php)

**Configuring the Viewpoint Database (hibernate.cfg.xml)**

Viewpoint uses a database and relies on Hibernate, an Object/Relational Mapping (ORM) tool for Java programs, to communicate with the database. For Viewpoint to recognize and work with your database management system (DBMS), you must configure the hibernate.cfg.xml settings file.

**Setting Up Your DBMS**

Viewpoint uses Hibernate, which relies on the JDBC specification. Viewpoint works with the following database management systems: MySQL, SQL Server, PostgreSQL, and Oracle. Because each DBMS is different, you may have to consult the vendor documentation for your DBMS to learn how to do the following steps:

1. Make sure your DBMS is running and that it accepts remote connections.
2. Create a database in your DBMS for use with Viewpoint.
3. Use the SQL DDL files in the sql folder of your unzipped Viewpoint distribution to set up the tables, foreign key constraints, and so forth that Viewpoint is expecting. If using an unsupported JDBC driver, tailor one of the schema files to fit any DBMS specific syntax you have.

**Configuring Viewpoint to Communicate with Your DBMS**

To configure Viewpoint to point to your DBMS, do the following:

1. Set the hibernate.dialect property to the appropriate class for your DBMS as listed in the Hibernate Dialects table that follows.
2. Set the hibernate.connection.driver\_class to the appropriate class for your DBMS as listed in the Hibernate Dialects table that follows.
3. Change the hibernate.connection.url to point to the location where the database is located. As each DBMS database driver has slightly different connection URL syntax, you may need to consult the documentation for the JDBC driver for your DBMS. However, they often have the following form:
4. jdbc:<DBMS vendor>://<server>[:<port>]/<database>

|  |  |
| --- | --- |
| Note | To help you get started quickly, some sample hibernate.connection.urls are listed in the Hibernate Dialects table that follows. |

1. Specify a username and password in the respective property elements.
2. If using MySQL or Oracle, you will need to download the JDBC drivers separately and place them where Viewpoint can find them.
   * The MySQL driver can be downloaded from <http://www.mysql.com/downloads/connector/j/>. The driver will come in a tar.gz or a zip file. The jar file will be inside this tar.gz or zip file and will likely be named something like mysql-connector-java-x.x.x-bin.jar.
   * The Oracle driver can be downloaded from the Oracle web site at <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>. Get the appropriate driver jar file for your Oracle database version. We recommend the drivers in the "JDBC Thin for All Platforms" section of this page. Also note you will need to get an Oracle driver compatible with JDK 1.4 and 1.5.  
       
     After downloading the appropriate driver for your database version, extract the contents of this file and place the jar file in the tomcat/webapps/moab/WEB-INF/lib directory before you start Viewpoint.

|  |  |
| --- | --- |
| SQL Server Hibernate Dialects | |
| Dialect | org.hibernate.dialect.SQLServerDialect |
| connection.driver\_class | net.sourceforge.jtds.jdbc.Driver |
| Sample Connection URL | jdbc:jtds:sqlserver://localhost:41433/my\_database |
|  |  |
| MySQL Hibernate Dialects | |
| Dialect | org.hibernate.dialect.MySQLInnoDBDialect |
| connection.driver\_class | com.mysql.jdbc.Driver |
| Sample Connection URL | jdbc:mysql://localhost/my\_database |
|  |  |
| PostgreSQL Hibernate Dialects | |
| Dialect | org.hibernate.dialect.PostgreSQLDialect |
| connection.driver\_class | org.postgresql.Driver |
| Sample Connection URL | jdbc:postgresql://localhost/my\_database |
|  |  |
| Oracle Hibernate Dialects | |
| Dialect | com.cri.poller.hibernate.Oracle10gDialect |
| connection.driver\_class | oracle.jdbc.pool.OracleDataSource |
| Sample Connection URL | jdbc:oracle:thin:moab/moab@localhost:1521/XE |

**Configuring Schema Containers**

Some DBMS packages support using schema namespaces. Schema namespaces are a way of logically grouping the tables within a single database into separate containers. Viewpoint supports the use of schema containers with the optional hibernate.default\_schema property.

To set the schema property, do the following:

1. Add the following to the hibernate.cfg.xml file:

<property name="hibernate.default\_schema">myschema</property>

1. Modify the SQL DDL files in the sql folder of your unzipped Viewpoint distribution that set up the tables, foreign key constraints,and so forth that Viewpoint is expecting. For every line in the SQL DDL file that references a table name, prepend "<schema name>." to the table name.

For example, prepend the table names in following lines with "<schema name>.":

drop table MoabNode\_VM;

create table MoabNode\_VM (MoabNode\_nodeID varchar(255) not null, vms\_id varchar(255) not null, ...

alter table MoabNode\_VM add constraint FK\_MoabNodeVM\_VM\_ID foreign key ...

The amended lines should appear as follows:

drop table myschema.MoabNode\_VM;

create table myschema.MoabNode\_VM (MoabNode\_nodeID varchar(255) not null, vms\_id varchar(255) not null, ...

alter table myschema.MoabNode\_VM add constraint FK\_MoabNodeVM\_VM\_ID foreign key ...

**Other Configuration Parameters**

In addition to the JDBC connection parameters you will notice that the sample hibernate.cfg.xml file has many other properties. In addition you may notice some mapping elements, like

<mapping class="com.cri.cart.server.ShoppingCart" />

|  |  |
| --- | --- |
| Caution | Refrain from modifying properties or mappings without talking to support first as modifications may cause problems with the communication between Viewpoint and the database. For more information on what these configuration options do, please refer to the Hibernate documentation. |

Mapping elements enable Viewpoint to manage database tables for certain features. In particular, Viewpoint will be able to manage tables for the the Java class referenced by the class attribute. It is recommended that you not modify these mappings or Viewpoint may not be able communicate properly with the database.

Below is an explanation of the other properties in the hibernate.cfg.xml file.

|  |  |
| --- | --- |
| hibernate.transaction.auto\_close\_session | |
| Suggested value: | true (Strongly recommended) |
| Description: | If enabled, the persistence session is automatically closed after the completion phase of its database transactions. The alternative is to have the application close these sessions manually. In Viewpoint this should be set to true. |
|  |  |
| hibernate.connection.autocommit | |
| Suggested value: | false (Strongly recommended) |
| Description: | When true, this parameter causes autocommit to be enabled for the JDBC connections in the connection pool. It is strongly recommended that this be false. Autocommit mode allows each individual SQL statement to be executed in its own transaction. Starting these extra transactions causes lots of overhead and destroys atomicity and isolation guarantees for operations that should be grouped together in a transaction. |
|  |  |
| hibernate.current\_session\_context\_class | |
| Suggested value: | thread (Strongly recommended) |
| Description: | How the hibernate session should be bound. Although hibernate supports "thread", "jta", and "managed" as legal values for this parameter, in Viewpoint this should be set to "thread". "thread" causes sessions to be bound to threads. "jta" binds sessions to JTA transactions and "managed" causes the responsibility for managing session scope, start, and end to the application. |
|  |  |
| hibernate.show\_sql | |
| Suggested value: | false |
| Description: | Causes all SQL statements to be printed to the console. A better option is to cause SQL to be sent to the log instead of the console. To do this set the log category org.hibernate.SQL to debug in the log4j.xml file. |
|  |  |
| hibernate.format\_sql | |
| Suggested value: | true |
| Description: | Causes the SQL printed to the log and the console to be pretty printed |
|  |  |
| connection.provider\_class | |
| Suggested value: | org.hibernate.connection.C3P0ConnectionProvider |
| Description: | Configures the JDBC connection pool which makes communication with the database more efficient. Viewpoint supports only c3p0, or no connection pool. To disable connection pooling. the connection.provider\_class property can be commented out. |
|  |  |
| hibernate.c3p0.min\_size | |
| Suggested value: | 5 |
| Description: | The minimum number of JDBC connections that c3p0 keeps ready at all times. This property is supported with only the c3p0 connection provider. |
|  |  |
| hibernate.c3p0.max\_size | |
| Suggested value: | 20 |
| Description: | The maximum number of JDBC connections JDBC can use. A runtime exception is thrown if this number is exhausted. This property is supported with only the c3p0 connection provider. |
|  |  |
| hibernate.c3p0.timeout | |
| Suggested value: | 25000 |
| Description: | The connection timeout period in seconds. After this amount of time idle connections are removed from the pool. It is recommended to make this value rather large (around 25000) because at smaller values (like 40) some customers have reported problems with connections timing out too quickly. This property is supported with only the c3p0 connection provider. |
|  |  |
| hibernate.c3p0.max\_statements | |
| Suggested value: | 0 |
| Description: | The number of prepared statements that c3p0 will cache. Although prepared statement caching improves performance it has been linked to sporadic race conditions due to bugs in c3p0. For that reason it is recommended to set this to 0 to disable prepared statement caching. This property is supported with only the c3p0 connection provider. |
|  |  |
| hibernate.c3p0.idle\_test\_period | |
| Suggested value: | 3000 |
| Description: | The idle time in seconds before a connection is automatically validated. This causes the connection to be kept alive even when idle so it doesn't get closed by a firewall. This property is supported with only the c3p0 connection provider. |

**Hibernate JDBC Properties Quick Reference**

The property elements contain the necessary configuration information for the JDBC connection. The following table offers more information on the JDBC properties:

|  |  |
| --- | --- |
| Hibernate JDBC Properties | |
| **Property Name** | **Description** |
| hibernate.connection.driver\_class | The JDBC connection (driver) class for the specific database. |
| hibernate.dialect | The particular SQL variant (dialect) Hibernate generates for the database. |
| hibernate.connection.url | The full JDBC URL to the database. |
| hibernate.connection.username | The database user's username. |
| hibernate.connection.password | The database user's password to authenticate the username. |

The following represents the out-of-the-box hibernate.cfg.xml settings:

<hibernate-configuration>

<session-factory>

<!-- General properties -->

<property name="hibernate.transaction.auto\_close\_session">true</property>

<property name="hibernate.connection.autocommit">false</property>

<property name="hibernate.current\_session\_context\_class">thread</property>

<!-- Setting the schema namespace (Optional) -->

<!--

<property name="hibernate.default\_schema">dbo</property>

-->

<!-- MySQL connections -->

<property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLInnoDBDialect</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost/my\_database</property>

<property name="hibernate.connection.username">moab</property>

<property name="hibernate.connection.password">p@ssw0rd</property>

<!-- Postgres connections -->

<!--

<property name="hibernate.connection.driver\_class">org.postgresql.Driver</property>

<property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>

<property name="hibernate.connection.url">jdbc:postgresql://localhost/my\_database</property>

<property name="hibernate.connection.username">moab</property>

<property name="hibernate.connection.password">p@ssw0rd</property>

-->

<!-- H2 connections -->

<!--

<property name="hibernate.connection.driver\_class">org.h2.Driver</property>

<property name="hibernate.dialect">org.hibernate.dialect.H2Dialect</property>

<property name="hibernate.connection.url">jdbc:h2:tcp://localhost/~/my\_database</property>

<property name="hibernate.connection.username">sa</property>

<property name="hibernate.connection.password">p@ssw0rd</property>

-->

<!-- Sql Server connections -->

<!--

<property name="hibernate.connection.driver\_class">net.sourceforge.jtds.jdbc.Driver</property>

<property name="hibernate.dialect">org.hibernate.dialect.SQLServerDialect</property>

<property name="hibernate.connection.url">jdbc:jtds:sqlserver://localhost:41433/my\_database;tds=8.0</property>

<property name="hibernate.connection.username">moab</property>

<property name="hibernate.connection.password">p@ssw0rd</property>

-->

<!-- Use the C3P0 connection pool provider -->

<property name="connection.provider\_class">org.hibernate.connection.C3P0ConnectionProvider</property>

<property name="hibernate.c3p0.min\_size">5</property>

<property name="hibernate.c3p0.max\_size">20</property>

<property name="hibernate.c3p0.timeout">40</property>

<property name="hibernate.c3p0.max\_statements">0</property>

<property name="hibernate.c3p0.idle\_test\_period">3000</property>

<!-- Show SQL on stdout -->

<property name="hibernate.show\_sql">false</property>

<property name="hibernate.format\_sql">false</property>

<!-- Shopping Cart classes -->

<mapping class="com.cri.cart.server.ShoppingCart"/>

<mapping class="com.cri.cart.server.ShoppingCartLineItem"/>

<mapping class="com.cri.cart.server.ShoppingCartItem"/>

<!-- Archive classes -->

<mapping class="com.cri.archival.server.model.Archive"/>

<mapping class="com.cri.archival.server.model.ArchiveObject"/>

<!-- VPC user classes -->

<mapping class="com.cri.provisioning.environmentmgt.server.model.VpcUser"/>

<mapping class="com.cri.component.server.form.persistmodel.FormStatePersistModel"/>

<mapping class="com.cri.security.server.SecurityUser"/>

</session-factory>

</hibernate-configuration>

[Home](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/index.php) [Up](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/index.php) [Previous](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/installing_viewpoint.php) [Next](http://docs.adaptivecomputing.com/viewpoint/archive/2-0/configuring_the_connection_to_moab.php)