DatabaseConnectionOverview

Purpose of this page

Many wiki pages exist that detail how to set up a specific SQL connection in Cocoon. See SpecificDatabaseConnection for a list of pages detailing specific database platforms. This page is meant to provide a general introduction to setting up a database connection.

Six Steps to a Database Connection in Cocoon

One: What database software are you using? MySQL? MS Access? dBase? xBase? FoxPro? Oracle? Ingres? Postgres? MS SQL Server? Something else? You'll need to know this in order to get the right driver. There is a page on Sun's Java site dedicated to helping you do this. http://servlet.java.sun.com/products/jdbc/drivers This link was current as of May 23, 2003.If you click the "Browse All" button, you will get ALL the availble JDBC drivers. Note that some vendors' drivers support connections to more than one type of database. Also note that there is no direct link from the JDBC driver vendor page to any vendors. You'll need to google the vendor for their website.

Two: Download the appropriate driver. Do whatever you feel is necessary to satisfy yourself that the download is good. (i.e. scan the downloaded driver zip, tarball, or whatever, file for viruses or other trojans or other nasty surprises). A bit paranoid perhaps, but better safe than sorry.

Three: Read the driver documentation and licenses. This is important later, as you will need to know the specific JDBC url syntax necessary for the driver to establish a connection to the database.

Four: If Cocoon is going to use the driver, Cocoon has to be able to find it. That means the driver jar has to be in the classpath. Put a copy of the driver jar in the ***path-to-your-webapp-server*/cocoon/WEB-INF/lib** directory.

Five: Now that the driver class can be found, you have to tell Cocoon to load the driver class. You do so by setting an *init-param* in web.xml, which can be found in ***path-to-your-webapp-server*/cocoon/WEB-INF/**

To your web.xml file, add:

```
<init-param>
  <param-name>load-class</param-name>
  <param-value>fully.qualified.class.name.for.your.jdbcDriver</param-value>
</init-param>
```

a real world example comes from a web.xml for a Cocoon driven site I maintain:

```
<init-param>
  <param-name>load-class</param-name>
  <param-value>org.gjt.mm.mysql.Driver</param-value>
<init-param>
```

Though if you have a recent mysql Connector/J driver you would use `com.mysql.jdbc.Driver`

Six: If you are going to use the ESQL taglib in an extensible server page (XSP), or the SQL transformer, you have to set up a datasource in cocoon.xconf, which can be found in ***path-to-your-webapp-server*/cocoon/WEB-INF**

In cocoon.xconf, find the <datasources> tag. Below is real world example from a cocoon.xconf file for a Cocoon site.

```
<datasources>
  <jdbc name="public-users">
    <pool-controller max="8" min="2"/>
    <driver>org.gjt.mm.mysql.Driver</driver>
    <dburl>jdbc:mysql://dbserver-1.mydomainname.com/databasename</dburl>
    <user>username</user>
    <password>password</password>
  </jdbc>

  <jdbc name="private-users">
    <pool-controller max="2" min="1"/>
    <driver>org.gjt.mm.mysql.Driver</driver>
    <dburl>jdbc:mysql://dbserver-1.mydomainname.com/admindb</dburl>
    <user>admin-username</user>
    <password>admin-password</password>
  </jdbc>
<datasources>
```

And a more recent example:

```
<datasources>
 <jdbc name="authorisatie">
   <pool-controller max="10" min="5"/>
   <dburl>jdbc:mysql://dbserver/auth</dburl>
   <user>username</user>
   <password>secret</password>
  </jdbc>
</datasources>
```

Note that the driver is not required here; it's part of the protocol.

Some explanation of the tags:

- **<jdbc name="*some-named-connection-pool*">** This tag creates a pool of database connections. The name of the pool is specified by the value of the *name* parameter (e.g. *some-named-connection-pool*).
- **<pool-controller min="*minimum*" max="*maximum*">** Determines the minimum and maximum number of allowable database connections. There will always be at least *min* number of database connections in the pool. For those not familiar with pools: If there are less than the maximum allowable database connections in the pool **and** the existing database connections are unavailable (i.e. they are all currently being used by some process(es)) **and** another process requests a database connection, the pool controller will create a new database connection for the requesting process to use.
- **<driver>** The fully qualified class name of the driver Cocoon will use to connect to the database.
- **<dburl>** This is necessary for the driver to connect to the database. The url syntax is not standardized among all vendors, so you must read the driver documentation. For example,
  *Oracle uses *jdbc:oracle:thin:@localhost:1521:mydatabase*
  *MySQL uses *jdbc:mysql://some.database.server/databasename*
- **<user>** the username needed to connect to the database
- **<password>** the password needed to connect to the database

---

Related Wiki Pages

ConnectionPooling

SpecificDatabaseConnection

SpecificDatabaseConnectionNew

SQL

Reader Comments