スタンドアロンで使う

サンプル

ディレクトリ構成

```
[root]
  |   build.xml
  |   s2jdbc-gen-build.xml
  |
  |---conf
  |       app.dicon <- アプリケーションを構成するルートとなる dicon ファイルです.
  |       convention.dicon <- アプリケーションのネーミング規約を定義する dicon ファイルで, SMART deploy
を利用する場合に必要となります.
  |       jdbc.dicon <- JDBC データソースを定義する dicon ファイル
  |       s2jdbc.dicon <- jdbcManager の定義等
  |
  |---lib
  |---src
```

設定ファイル

■ app.dicon

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
"http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="convention.dicon"/>
  <include path="aop.dicon"/>
  <include path="s2jdbc.dicon"/>
  <component name="runtest" class="examples.RunTest" instance="singleton" />
</components>
```

■ convention.dicon

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <component class="org.seasar.framework.convention.impl.NamingConventionImpl">
    <initMethod name="addRootPackageName">
      <arg>""</arg>

    </initMethod>
  </component>
  <component class="org.seasar.framework.convention.impl.PersistenceConventionImpl"/>
</components>
```

■ s2jdbc.dicon

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
"http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="jdbc.dicon"/>

  <include path="s2jdbc-internal.dicon"/>
  <component name="jdbcManager" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
    <property name="maxRows">0</property>

    <property name="fetchSize">0</property>
    <property name="queryTimeout">0</property>
    <!--<property name="dialect">sqliteDialect</property>-->
    <property name="dialect">hsqlDialect</property>
```

```
  </component>
</components>
```

■ jdbc.dicon

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
"http://www.seasar.org/dtd/components24.dtd">
<components namespace="jdbc">
  <include path="jta.dicon"/>

  <component name="xaDataSource"
    class="org.seasar.extension.dbcp.impl.XADataSourceImpl">
    <property name="driverClassName">
      "org.hsqldb.jdbcDriver"
    </property>
    <property name="URL">
      "jdbc:hsqldb:file:hsql.db"
    </property>
    <property name="user">"sa"</property>
    <property name="password">""</property>
  </component>

  <!--<component name="xaDataSource"
  class="org.seasar.extension.dbcp.impl.XADataSourceImpl">
  <property name="driverClassName">
  "org.sqlite.JDBC"
  </property>
  <property name="URL">
  "jdbc:sqlite:c:/data/source/java/s2jdbc/test.db"
  </property>
  <property name="user">""</property>
  <property name="password">""</property>
  </component>-->

  <component name="connectionPool"
    class="org.seasar.extension.dbcp.impl.ConnectionPoolImpl">
    <property name="timeout">600</property>
    <property name="maxPoolSize">10</property>
    <property name="allowLocalTx">true</property>
    <destroyMethod name="close"/>
  </component>


  <component name="DataSource"
  class="org.seasar.extension.dbcp.impl.DataSourceImpl"
  />


</components>
```

エンティティ作成

■ s2jdbc-gen-build.xml

```
<?xml version="1.0"?>
<project name="sample" default="gen-entity" basedir=".">
  <path id="classpath">
    <fileset dir="lib"/>
  </path>
  <taskdef resource="s2jdbc-gen-task.properties" classpathref="classpath"/>

  <target name="gen-entity">
    <gen-entity
    rootpackagename="examples"
    classpathref="classpath"
    classpath="conf"
    overwrite="true"
    />
  </target>

  <target name="gen-ddl">
    <gen-ddl
    classpathdir="build/classes"
```

```
      rootpackagename="examples"
      classpathref="classpath"
      classpath="conf"
      />
    </target>

    <target name="migrate">
      <migrate
      classpathdir="build/classes"
      rootpackagename="examples"
      classpathref="classpath"
      />
    </target>

  </project>
```

■エンティティ作成

```
ant -f s2jdbc-gen-build.xml
```

コンパイル、実行

■テストソース

Run.java

RunTest.java

■ build.xml

```
<?xml version="1.0"?>
<project name="sample" default="compile" basedir=".">

  <property name="app.home" location="."/>
  <property name="app.lib"   location="${app.home}/lib"/>
  <property name="app.conf"   location="${app.home}/conf"/>
  <property name="app.class" location="${app.home}/build/classes"/>
  <property name="app.src"   location="${app.home}/src"/>
  <property name="app.dist"   location="${app.home}/dist"/>

  <path id="classpath">
    <fileset dir="${app.lib}" includes="**/*.jar"/>
    <pathelement location="${app.conf}"/>
    <pathelement location="${app.class}"/>
  </path>


  <target name="compile" >
    <mkdir dir="${app.class}"/>

    <javac includeantruntime="false" srcdir="${app.src}" destdir="${app.class}" encoding="UTF-8">
      <classpath refid="classpath" />
    </javac>
  </target>

  <target name="run" depends="compile">
    <java classname="examples.Run">
      <classpath refid="classpath" />
    </java>
  </target>

  <!-- クリーン処理 -->
    <target name="clean" >
      <delete dir="${app.class}"/>
    </target>

</project>
```

■コンパイルと実行

```
ant run
```

Unit テスト

S2Unit を使う。

■サンプル

ID とか実行日時などの項目でテストの対象から除外したいカラムがある場合は、
xls ファイルの項目から削除しておくと、比較対象にならない。
参考：http://ameblo.jp/kazuya232323/entry-11227861408.html

Struts で使う

設定

■ classes/app.dicon

```
<include path="s2jdbc.dicon"/>
```

を有効にする。

■ classes/s2jdbc.dicon

のダイアレクトを自分の DB に合わせる。
例えば、Oracle の場合は

```
<property name="dialect">oracleDialect</property>
```

■ classes/jdbc.dicon

自分の DB に合わせて

```
xaDataSource
connectionPool
DataSource
```

を有効にする。

エンティティを使わない

```
List<BeanMap> results =
jdbcManager
.selectBySql(
BeanMap.class,
"select * from test")
.getResultList();
```

でエンティティを使わずにデータを取得できる。

エンティティを使う

■準備

```
S2JDBC-Gen
```

を使ってエンティティを生成する。

1.S2JDBC-Gen をダウンロードして解凍
2.s2jdbc-gen/resources/s2jdbc-gen-build.xml を編集
3. ant -f s2jdbc-gen-build.xml gen-entity

■エンティティを使う

```
List<hogehoge.entity.Test> results =
jdbcManager
.from(hogehoge.entity.Test.class)
.where("id = ?", 1)
.getResultList();
```

複数の DB へ接続する

http://ymotoba.blogspot.jp/2008/10/s2jdbcdb.html

■作成するファイル

```
jdbca.dicon
jdbcb.dicon
s2jdbca.dicon
s2jdbcb.dicon
```

■ s2jdbc.dicon

まずは、デフォルトで用意されている s2jdbc.dicon を以下のように編集

```
<components>
<include path="s2jdbca.dicon"/>
<include path="s2jdbcb.dicon"/>
</components>
```

■ s2jdbca.dicon

```
<components namespace="s2jdbcA">
  <include path="jdbca.dicon"/>
  <include path="s2jdbc-internal.dicon"/>
  <component name="jdbcManagerA" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
    <property name="maxRows">0</property>
    <property name="fetchSize">0</property>
    <property name="queryTimeout">0</property>
    <property name="dialect">mysqlDialect</property>
    <property name="dataSource">DataSource</property>
  </component>
</components>
```

■ s2jdbcb.dicon

```
<components namespace="s2jdbcB">
  <include path="jdbcb.dicon"/>
  <include path="s2jdbc-internal.dicon"/>
  <component name="jdbcManagerB" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
    <property name="maxRows">0</property>
    <property name="fetchSize">0</property>
    <property name="queryTimeout">0</property>
    <property name="dialect">mysqlDialect</property>
    <property name="dataSource">DataSource</property>
  </component>
</components>
```

■ jdbca.dicon

```
<components namespace="jdbc">
  <include path="jta.dicon"/>

  <component name="xaDataSource"
    class="org.seasar.extension.dbcp.impl.XADataSourceImpl">
    <property name="driverClassName">"org.h2.Driver"</property>
    <property name="URL">
      "jdbc:h2:tcp://localhost:8082/test"
    </property>
    <property name="user">"sa"</property>
    <property name="password">""</property>
    <destroyMethod>
      @org.seasar.framework.util.DriverManagerUtil@deregisterAllDrivers()
    </destroyMethod>
  </component>

  <component name="connectionPool"
    class="org.seasar.extension.dbcp.impl.ConnectionPoolImpl">
    <property name="timeout">600</property>
    <property name="maxPoolSize">10</property>
    <property name="allowLocalTx">true</property>
    <destroyMethod name="close"/>
  </component>

  <component name="DataSource" class="org.seasar.extension.dbcp.impl.DataSourceImpl"/>

</components>
```

■ jdbcb.dicon

```
<components namespace="jdbc">
  <include path="jta.dicon"/>

  <component name="xaDataSource"
    class="org.seasar.extension.dbcp.impl.XADataSourceImpl">
    <property name="driverClassName">
      "com.mysql.jdbc.Driver"
    </property>
    <property name="URL">
                  "jdbc:mysql://localhost:3306 /clontalk03 ?useUnicode=true&characterEncoding=UTF8
&zeroDateTimeBehavior=convertToNull"
    </property>
    <property name="user">"root"</property>
    <property name="password">""</property>
  </component>


  <component name="connectionPool"
    class="org.seasar.extension.dbcp.impl.ConnectionPoolImpl">
    <property name="timeout">600</property>
    <property name="maxPoolSize">10</property>
    <property name="allowLocalTx">true</property>
    <destroyMethod name="close"/>
  </component>

  <component name="DataSource"
  class="org.seasar.extension.dbcp.impl.DataSourceImpl"
  />

</components>
```

■ JdbcManager

上記設定が完了すれば。

下記のように @Binding でインジェクションしたい JdbcManager を選択すればオッケー！

```
@Binding("jdbcManagerA")
private JdbcManager h2JdbcManager;

@Binding("jdbcManagerB")
private JdbcManager mySqlJdbcManagerB;
```