Bruk av flere databaser i JGrape-generert kode

v. 1.0 - omb - 05/12/2005

Innhold:

1. Innledning

Fram til nå har det bare vært mulig å kjøre mot en database fra JGrape-generert kode. Fra versjon G-810Alpha10 vil det være mulig både å angi hvilken database man ønsker å kjøre mot samt å åpne flere dataforbindelser mot ulike databaser samtidig.

JGrape-generert kode som ikke er endret vil kjøre som før mot default database. Dette innebærer at Hibernate-settinger leses fra hhv. "hibernate.properties" og "hibernate.cfg.xml" i classpath.

2. Begrep

- context: det innføres et nytt begrep "database context". Dette er et logisk navn på en databasekonfigurasjon hvor det for en gitt server og en gitt databasemapping (fra Genova workspace) finnes et sett gjeldende konfigurasjonsparametre.

  Databasekontekst er innført som et nytt streng-attributt i no.genova.support.ClientContext: get/set-DatabaseContext.

- server: dette er et logisk navn på et sett Hibernate-settinger hvor det bl.a. angis connection-URL eller datasource osv. Det skal være mulig å parallellt kjøre mot datakilder av ulik art så lenge de er støttet av Hibernate.

3. Konfigurasjon

Database-settinger for JGrape-kode styres fra "genova.config.properties". Følgende nye settinger er innført:

- Application.<app.name>.osmapping=<path>
  Her angis det path (med utgangspunkt i classpath) til den filen som inneholder mapping fra alle objektseleksjoner til deres aktuelle databasemapping. Filen vil genereres ut sammen med service-koden, men må vedlikeholdes manuelt siden det pr. i dag ikke ligger noen informasjon i workspace eller modell som knytter objektseleksjoner til en gitt databasemapping.

- Database.Context.<context name>.<mapping name>.server=<server name>
  Her angis det hvilken server (logisk navn) som skal brukes for en gitt kombinasjon av database-kontekst og -mapping. Det bør finnes en slik setting for alle aktuelle databasemappinger i en gitt kontekst.

- Database.Server.<server name>.properties=<properties file name>
  Her angis navn på den properties-fil som inneholder Hibernate-settinger for et gitt logisk servernavn.

- Database.Mapping.<mapping name>.configuration=<XML config file name>

Navn på den Hibernate-konfigurasjonsfil som inneholder
mapping-elementer for alle domeneklasser i en gitt
databasemapping.

Det er mulig å lage en felles databasemapping for alle
domeneklasser hvis det er ønskelig for å foreknle oppsettet i
"genova.config.properties" (tilsvarende som for SysdulMap.c).

3.1 Eksempel på konfigurasjon

3.1.1 genova.config.properties

```
Application.Fastsettelse.osmapping=/
Database.Context.testbase.TVINN.server=testserver
Database.Context.testbase.POST.server=testserver
Database.Context.testbase.WORM.server=wormserver
Database.Server.testserver.properties=testserver.properties
Database.Server.wormserver.properties=wormserver.properties
Database.Mapping.TVINN.configuration=tvinn.hibernate.cfg.xml
Database.Mapping.POST.configuration=post.hibernate.cfg.xml
Database.Mapping.WORM.configuration=worm.hibernate.cfg.xml
```

3.1.2 Fastsettelse.osmapping.properties

```
# Fastsettelse properties, database mapping used in object selections.
# Generated with 'Genova' by 'omb on Fri Dec 02 16:39:55 2005
#Fri Dec 02 16:39:55 2005
MaskeVedl_os=TVINN
MaskeOL_os=TVINN
```

3.1.3 testserver.properties

```
hibernate.connection.datasurce=TvinnDataSource
hibernate.transaction.manager_lookup_class=org.hibernate.transaction.WeblogicTra
nsactionManagerLookup
hibernate.connection.pool_size=1
hibernate.dialect=org.hibernate.dialect.SybaseDialect
```

3.1.4 tvinn.hibernate.cfg.xml

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration
  PUBLIC "-//Hibernate/Hibernate Configuration DTD//EN"
  "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
      <session-factory>
            <mapping resource="no/toll/tvinn/domain/kunde/Kunde.hbm.xml"/>
            <mapping resource="no/toll/tvinn/domain/kunde/Kundeadr.hbm.xml"/>
            <mapping
resource="no/toll/tvinn/domain/deklarasjon/Innfdekl.hbm.xml"/>
      </session-factory>
</hibernate-configuration>
```

4. Praktisk bruk

Det er ClientContext-objektet som angir hvilken databasekontekst som
til enhver tid benyttes av generert kode. ClientContext-objektet som
benyttes av generert kode på klientsiden (og som sendes videre til
serverkoden) ligger som et attributt på genrert "Client"-klasse for
applikasjonen:

```
  ClientFastsettelse.clientContext.setDatabaseContext("testbase");
```

Det kan være naturlig at databasekontekst settes som følge av valg brukeren har gjort under innlogging, evt. kan den endres i hooks ved behov.

Dersom man ønsker å operere med flere samtidige databasekoblinger trenger man en "no.genova.dataaccess.Session" pr. kobling. Det er SessionFactory som lager nye sessions med følgende metode:

```
  public Session getSession(final String contextName, final String mappingName)
```

Her angis hvilken databasekontekst man ønsker å benytte samt gjeldende databasemappingnavn.

Følgende static-metode på HibernateSessionFactory kan benyttes for å få tak i en SessionFactory:

```
  public static SessionFactory getSessionfactory()
```

Gjeldende databasemappingnavn for en gitt objektseleksjon kan f.eks. hentes via no.genova.support.Registry:

```
  Registry reg = Registry.getRegistry();
  String mappingName  =
reg.getApplicationProperty(objectSelection.getApplication()+".osmapping",
objectSelection.getObjectSelection());
```