ImplementingHibernateLongSessions

Implementing the Hibernate long-session-pattern

Being asked about how to use long hibernate sessions with Hivemind a couple of times on the Tapestry-Users-List - Here is what I did:

Overview

Basically all one needs to accomplish this is

- a persistence service which I called ClientStateStorage. It can be provided with an http session as the underlying storage mechanism for production (via a servlet filter). Otherwise it uses a simple Map for testing.
- a service-model extending AbstractServiceModelImpl which works much like the PooledServiceModel, the differences being firstly, that it comes with a different lifecycle and, secondly that it doesn't store service-implementations in a pool common to all clients but in the aforementioned Client StateStorage.

Implementation

Here is the hivemodule.xml making the new service-model available to other modules. As well, it provides a very simple schema to configure the Hibernate Session Factory.

```
<?xml version="1.0"?>
<module id="scm.hivemind" version="1.0.0"
                package="scm.hivemind.statefulservice">

  <contribution configuration-id="hivemind.ServiceModels">
    <service-model class="StatefulServiceModelFactory"
                              name="stateful"/>
  </contribution>

  <service-point id="ClientStateStorage">
    <invoke-factory model="pooled">
                <construct class="ClientStateStorageImpl" />
    </invoke-factory>
  </service-point>

   <schema id="HibernateSessionFactory">
      <element name="property">
               a hibernate property
        <attribute name="name" required="true">
          the property name
        </attribute>
        <attribute name="value">
          the value ( as in config.hbm.xml without the leading "hibernate.")
        </attribute>
        <conversion class="scm.hivemind.hibernate.HibernateProperty"/>
      </element>
      <element name="config-xml">
               the hibernate config.xml File
        <attribute name="name" required="true">
           the name of the config.xml file
        </attribute>
        <conversion class="scm.hivemind.hibernate.HibernateConfigFile"/>
      </element>
    </schema>

</module>
```

Java-Code

Here's the Java Code:

Client-State-Store

- ClientStateStorage

- ClientStateStorageImpl
- StateStorageClearanceListener
- StatefulHivemindFilter

service-model stuff

- StatefulServiceModel
- StatefulServiceModelFactory
- StatefulServiceLifecycleListener

the hibernate session-factory

- HibernateSessionFactory
- HibernateProperty
- HibernateConfigFile

Usage

The following snippet from an applications hivemodule.xml shows how to configure long hibernate Session which then can be injected into the app-specific services.

```
<service-point id="DWKHibernateSession" interface="net.sf.hibernate.Session">
     The hibernate-session itself.
     <invoke-factory service-id="webkit.awk.DWKSessionFactory"
                                     model="stateful"/>
  </service-point>

  <configuration-point id="DWKSessionFactory"
                         schema-id="scm.hivemind.HibernateSessionFactory"/>

  <contribution configuration-id="DWKSessionFactory" >
          <config-xml name="/scm/extranet/domain/hibernate.cfg.xml"/>
          <property name="dialect" value="net.sf.hibernate.dialect.OracleDialect"/>
          <property name="cache.provider_class" value="net.sf.ehcache.hibernate.Provider"/>
          <property name="cache.use_query_cache" value="true"/>
  </contribution>

  <service-point id="DWKSessionFactory"
                 interface="org.apache.hivemind.ServiceImplementationFactory">
        <invoke-factory model="singleton">
                 <construct
                     class="scm.hivemind.hibernate.HibernateSessionFactory"
            initialize-method="init">
                       <configuration>DWKSessionFactory</configuration>
        </construct>
          </invoke-factory>
          <interceptor service-id="hivemind.LoggingInterceptor"/>
  </service-point>
```

In a real web-application you would have to configure the StatefulHivemindFilter for your controller Servlet.

Precautions

- Take care to avoid concurrent requests to your controller servlet (sync on the session or something)
- While the stuff will work on a cluster, it won't support transparent failover of sessions, because the Hivemind-Proxies don't serialize instance state.