Hibernate Interceptors & Events

"Code with Passion!"

Topics

- Interceptors
- Events

Interceptors

Interceptors

- Interceptor interface provides callbacks from the session to the application, allowing the application to
  - inspect and/or
  - manipulate properties of a persistent object before it is saved, updated, deleted or loaded.
- Two kinds of inteceptors:
  - Session-scoped – applied to a particular session

    Session session = sf.openSession(new AuditInterceptor());
  - SessionFactory-scoped – applied to all sessions created from the Session factory

    new Configuration().setInterceptor(new AuditInterceptor());

How to build an Interceptor?

- You can either implement *Interceptor* directly or extend *EmptyInterceptor class*

```
public class AuditInterceptor extends EmptyInterceptor {

    private int updates;
    private int creates;
    private int loads;
.   ...
    public boolean onFlushDirty(Object entity,
                                Serializable id,
                                Object[] currentState,
                                Object[] previousState,
                                String[] propertyNames,
                                Type[] types) {

        if ( entity instanceof Auditable ) {
            updates++;
            for ( int i=0; i < propertyNames.length; i++ ) {
                if ( "lastUpdateTimestamp".equals( propertyNames[i] ) ) {
                    currentState[i] = new Date();
                    return true;
                }
            }
        }
        return false;
    }
```

Lab:

Exercise 1: Hibernate Interceptor
3521_hibernate_events.zip

Events

Event System

- If you have to react to particular events in your persistence layer, you can also use the Hibernate3 event architecture
  - The event system can be used in addition, or as a replacement, for interceptors.
- All the methods of the Session interface correlate to an event.
  - LoadEvent,
  - FlushEvent
  - etc.

Event Listener

- Implements the appropriate interface for the event it wants to process and/or extend one of the convenience base classes

```
public class MyLoadListener implements LoadEventListener {

    // this is the single method defined by the LoadEventListener interface
    public void onLoad(LoadEvent event, LoadEventListener.LoadType loadType)
            throws HibernateException {
        if ( !MySecurity.isAuthorized( event.getEntityClassName(), event.getEntityId() ) ) {
            throw MySecurityException("Unauthorized access");
        }
    }

}
```

Configuration of Event Listener

- Either in configuration file or programmatically

```
<!-- Through configuration file -->
<hibernate-configuration>
   <session-factory>
      ...
      <event type="load">
         <listener class="com.eg.MyLoadListener"/>
         <listener class="org.hibernate.event.def.DefaultLoadEventListener"/>
      </event>
   </session-factory>
</hibernate-configuration>

// Progmatically
 Configuration config = new Configuration();
 config.setListener("save-update", new UseringSaveOrUpdateEventListener());
```

Lab:

Exercise 2: Hibernate Events
3521_hibernate_events.zip

Code with Passion!