Table of Contents

Mit JVx kann die Business Logik sowohl in der Datenbank als auch in der Middle Tier entwickelt werden. Um die Business Logik vollständig in der Middle Tier zu entwickeln, sind sogenannte Server-Side Triggers eine Grundvoraussetzung. Diese funktionieren ähnlich wie die DataBook Ereignisse am Client.

Ein Server-Side Trigger wird z.B aufgerufen wenn am Client ein neuer Datensatz eingefügt/geändert/gelöscht wurde.

Anwendungsbeispiel

Wir verwenden am Client ein RemoteDataBook für die Erstellung von Benutzerkonten. Ein Benutzer wird definiert durch Benutzername, Passwort, Vor- und Nachname. Das Passwort muss jedoch verschlüsselt in der Datenbank abgelegt werden.

Wir definieren die Storage:

```
public DBStorage getUsers() throws Exception
{
   DBStorage users = (DBStorage)get("users");
   if (users == null)
   {
      users = new DBStorage();
      users.setDBAccess(getDBAccess());
      users.setWritebackTable("USERS");
      users.open();

      users.eventBeforeInsert().addListener(this, "doEncryptPwd");
      users.eventBeforeUpdate().addListener(this, "doEncryptPwd");

      put("users", users);
   }
   return users;
}
```

und den Trigger für Insert und Update:

```
public void doEncryptPwd(StorageEvent pEvent) throws Exception
{
   IBean bn = pEvent.getNew();

   String sNew = (String)bn.get("PASSWORD");
   String sOld;

   IBean bnOld = pEvent.getOld();

   if (bnOld != null)
   {
      sOld = (String)bnOld.get("PASSWORD");
   }
   else
   {
```

```
      sOld = null;
   }

   if (!CommonUtil.equals(sOld, sNew))
   {
      bn.put("PASSWORD", AbstractSecurityManager.getEncryptedPassword(
                         SessionContext.getCurrentSessionConfig(), sNew));
   }
}
```

In unserem Beispiel verwendeten wir ein IBean um auf die Properties zuzugreifen. Der Event erlaubt auch die Verwendung von POJOs wie in folgendem Beispiel:

```
public void doEncryptPwd(StorageEvent pEvent)
{
    ...
    ...
    User user = pEvent.getNew(User.class);

    user.setPassword(AbstractSecurityManager.getEncryptedPassword(
                     SessionContext.getCurrentSessionConfig(),
                     user.getPassword()));
}
```

Hinweis

Es kann jedes beliebige POJO verwendet werden. Der implementierte Mechanismus versucht, die Properties über die Spaltenbezeichnungen aus der Datenbank miteinander abzugleichen. In unserem letzten Beispiel könnte auch ein Adress POJO verwendet werden, sofern darin relevante Properties vorhanden sind. Es werden immer nur die möglichen Properties abgeglichen.