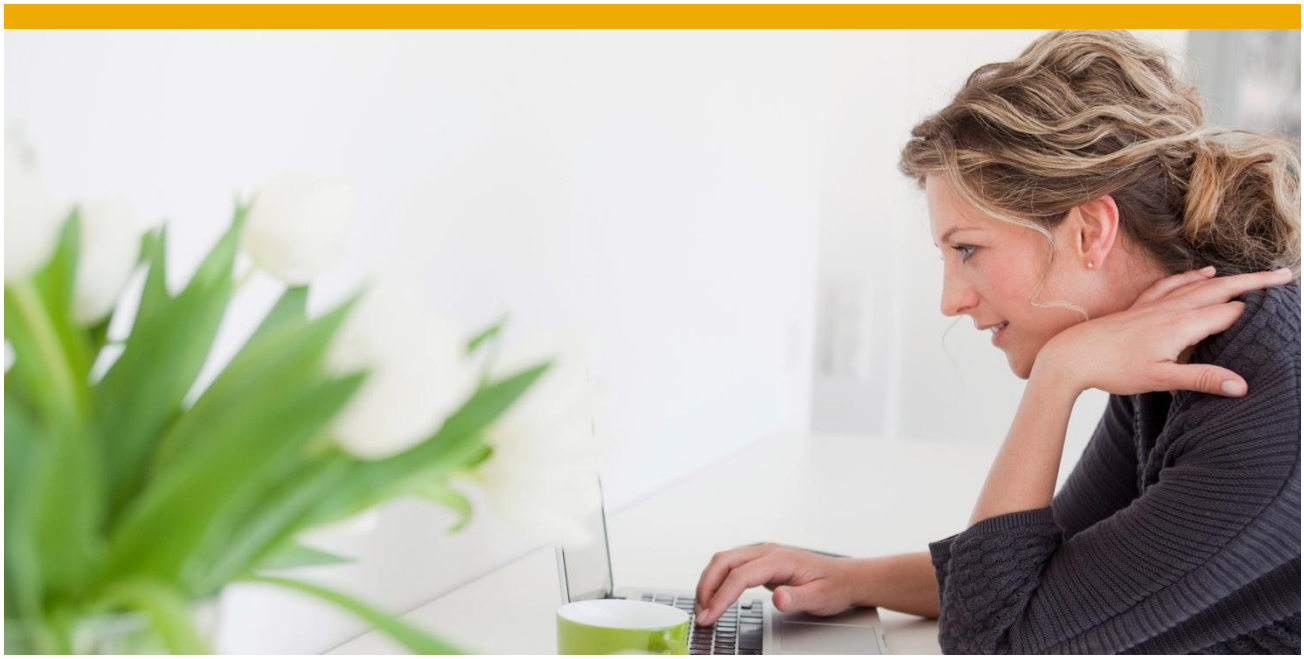

LAB2a: Customizing Database Schema

Create a New Table & Entity Bean

TABLE OF CONTENTS

AGENDA **3**
BEFORE YOU START… **3**
CREATE A NEW TABLE **3**
CREATE JPA BINDING FOR THE NEW TABLE **4**
UPDATE THE SPRING CONFIGURATION **8**

AGENDA

1. Create a new table:
    a. Write the create table statement
    b. Run the distribution build
    c. Run dbmaintain to create the new schema
    d. Verify the table creation

2. Create the JPA binding for the new table:
    a. Create the JPA Entity Bean that represents the table
    b. Create the DAO interface
    c. Create the Hibernate DAO implementation
    d. Update the DAO Factory Interface
    e. Update the Hibernate DAO Factory Implementation
    f. Add the new DAO to the Hibernate beans
    g. Update the Spring Configuration

BEFORE YOU START…

Please start the Mobiliser 5.1 Lab Virtual machine.

The Login with user is "mobiliser" and password "sybase".

CREATE A NEW TABLE

The customized table is a simple lookup table for the ACME departments. The table CUS_DEPARTMENT only contains two columns for the department ID (auto generated unique id) and the department name. This table contains updateable columns and therefore we must add the 4 standard mobiliser columns. So first we need to add a new schema file:

1. Start eclipse
2. Navigate to *com.sybase365.mobiliser.custom.project.persistence* bundle and to /src/main/schema/scripts/ASE/001_SETUP
3. Create a new script file **003_CUSTOM_PROJECT_TRAINING.**sql and add the table CUS_DEPARTMENT using below script:

    (for more information please see Mobiliser_Framework_5.1_Development_Guide.pdf section **4.1 Schema**)

```
CREATE TABLE CUS_DEPARTMENT
(
   ID_DEPARTMENT           NUMERIC(18) NOT NULL,
   STR_DEPARTMENT          NVARCHAR(80) NOT NULL,
   ID_ERROR_CODE           NUMERIC(5) NULL,
   DAT_CREATION            DATETIME NULL,
   ID_CUSTOMER_CREATION    NUMERIC(18) NULL,
   DAT_LAST_UPDATE         DATETIME NULL,
   ID_CUSTOMER_LAST_UPDATE NUMERIC(18) NULL,
  CONSTRAINT PK_DEPARTMENT PRIMARY KEY (ID_DEPARTMENT)
)
go
```

4. Save the file **003_CUSTOM_PROJECT_TRAINING.**sql.
5. Open a terminal (Applications/Accessories/Terminal)

6. Please shutdown the Mobiliser and Web containers before moving on to the next steps:
    a. Shutdown Web Container -- Change directory to ~/workspace/custom/dist/target/com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT/web
    b. Run the following command to shut down the web container:
    c. $ bin/shutdown.sh
    d. Shutdown Money Container -- Change directory to ~/workspace/custom/dist/target/com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT/money
    e. Run the following command to shut down the web container:
    f. $ bin/shutdown.sh
7. Change directory to ~/workspace/custom
8. Do a full build: following command would clean the /dist directory and recreate the deployable artifact (.zip file) and the dbmaintain scripts (.jar file).
    a. mvn clean install
9. Change directory to ~/workspace/custom/dist/target
10. Run dbmaintain: java -jar com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT-scriptarchive-ase.jar
11. Start Squirrel SQL and see if the table has been created.
    a. Chose the “local-ase (MOBR5)” connection
    b. Expend “Tables” under MOBR5 schema
    c. Look for your table “CUS_DEPARTMENT”

CREATE JPA BINDING FOR THE NEW TABLE

Adding the JPA Bindings for the new table affects all five packages in the persistence bundle:

1. *com.sybase365.mobiliser.custom.project.persistence.model*
    a. Create a new JPA Entity class named **Department.java** that extends the abstract class **GeneratedIdEntry** from the Mobiliser framework. (for more information please see Mobiliser_Framework_5.1_Development_Guide.pdf section **4.2 Data Model Beans**)
    b. You must use annotations for:
        i. @Entity – to mark this class as a JPA entity bean
        ii. @Table – to reference the physical database table name
        iii. @AttributeOverride – to specify that the id column is ID_DEPARTMENT
    c. Add the column STR_DEPARTMENT and implement the setter and getter for it. Also implement the isSet method for this column

NOTE: Please pay close attention during copying and pasting the code from the PDF into eclipse workspace because sometimes Word → PDF conversion messes up the “DOUBLE QUOTE” and add extra CR/LF. This is more common when copying XML files.

```
package com.sybase365.mobiliser.custom.project.persistence.model;

import com.sybase365.mobiliser.framework.persistence.model.GeneratedIdEntry;
import javax.persistence.AttributeOverride;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Table;

@Entity
@Table(name = "CUS_DEPARTMENT")
@AttributeOverride(name = "id", column = @Column(name = "ID_DEPARTMENT"))
public class Department extends GeneratedIdEntry {
    private static final long serialVersionUID = 1L;
    @Basic(optional = false)
    @Column(name="STR_DEPARTMENT", nullable=false)
    String name;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    boolean isSetDepartment(){
       return this.name != null;
    }
}
```

2. Please apply java code formatter for greater readability – press “Ctrl+A” to select all the code in Department.java and then press “Ctrl+Shift+F” for formatting the contents of the file.
3. To organize all the imports, please press “Ctrl-Shift-O” – please make sure that imports are correct.
4. Save the file Department.java
5. Go to *com.sybase365.mobiliser.custom.project.persistence.dao.api* project in eclipse
    a. Create the **DepartmentDAO.java** interface that extends the **GeneratedIdDAO** interface and introduce a custom service such as `findDepartmentByName();`

    (for more information please see Mobiliser_Framework_5.1_Development_Guide.pdf section **4.3 Data Access Object API**)

```
package com.sybase365.mobiliser.custom.project.persistence.dao.api;

import com.sybase365.mobiliser.custom.project.persistence.model.Department;
import com.sybase365.mobiliser.framework.persistence.dao.api.GeneratedIdDAO;

public interface DepartmentDAO extends GeneratedIdDAO<Department>{

       Department findDepartmentByName(String name);
}
```

6. Please format the code and organize the imports.
7. Save the file DepartmentDAO.java

8. Go to *com.sybase365.mobiliser.custom.project.persistence.dao.hibernate* in eclipse
    a. Implement the newly added Department DAO interface as **DepartmentDaoHbnImpl.java** (for more information please see Mobiliser_Framework_5.1_Development_Guide.pdf section **4.4 Implementing a DAO**)

```
package com.sybase365.mobiliser.custom.project.persistence.dao.hibernate;

import java.util.List;
import org.hibernate.Query;
import javax.persistence.EntityNotFoundException;
import com.sybase365.mobiliser.custom.project.persistence.dao.api.DepartmentDAO;
import com.sybase365.mobiliser.custom.project.persistence.model.Department;
import com.sybase365.mobiliser.framework.persistence.hibernate.dao.GeneratedIdDAOHbnImpl;

public class DepartmentDaoHbnImpl extends GeneratedIdDAOHbnImpl<Department> implements
DepartmentDAO {

    @Override
    public Class<Department> getEntityClass() {
       return Department.class;
    }

    @Override
    public Department findDepartmentByName(String name) {
      Query query = getSession().createQuery("FROM Department WHERE name = :name");
      query.setString("name", name);

      @SuppressWarnings("unchecked")
      List<Department> deptList = (List<Department>) query.setMaxResults(1).list();
      if(deptList.isEmpty()){
         throw new EntityNotFoundException("No departments with this name");
      }
      return deptList.get(0);
    }
}
```

9. Please format the code and organize the imports.
10. Save the file DepartmentDaoHbnImpl.java
11. Go to *com.sybase365.mobiliser.custom.project.persistence.dao.factory.api* in eclipse
    a. Add the method getDepartmentDao() to the **DaoFactory.java** interface class

```
(...)

DepartmentDAO getDepartmentDao();

(...)
```

12. Please format the code and organize the imports.
13. Save the file DaoFactory.java
14. Go to *com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.factory* in eclipse
    a. Override the missing DAO Factory method in the **HibernateDaoFactoryImpl.java** class (the Department DAO implementation will be injected into this class in the last chapter)

```
(...)

// ////////////////////////////////////////////////
// CUSTOM DAOs
// ////////////////////////////////////////////////

private BlacklistDAO blacklistDao;
private BlacklistTypeDAO blacklistTypeDao;
private WuCustomerDAO wuCustomerDao;
private DepartmentDAO departmentDao;

// ////////////////////////////////////////////////
// GETTER AND SETTER
// ////////////////////////////////////////////////

@Override
public void afterPropertiesSet() {
   Assert.notNull(blacklistDao, "blacklistDao is required");
   Assert.notNull(blacklistTypeDao, "blacklistTypeDao is required");
   Assert.notNull(wuCustomerDao, "wuCustomerDao is required");
   Assert.notNull(departmentDao, "departmentDao is required");
}

(...)

@Override
public DepartmentDAO getDepartmentDao() {
   return departmentDao;
}
public void setDepartmentDao(DepartmentDAO departmentDao) {
   this.departmentDao = departmentDao;
}

(...)
```

15. Please format the code and organize the imports.
16. Save the file HibernateDaoFactoryImpl.java
17. Add the newly created bean to the MobiliserHibernateBeans.java class using cache concurrency strategy as READ_WRITE

```
(...)

// First add all bean classes that needs cache concurrency mode read/write
lpc.addAll(PersistenceClass.createBeans(new Class[] { Blacklist.class,
BlacklistType.class, WuCustomer.class, Department.class },
CacheConcurrencyStrategy.READ_WRITE));

(...)
```

18. Please format the code and organize the imports
19. Save the file MobiliserHibernateBeans.java

UPDATE THE SPRING CONFIGURATION

20. Go to *com.sybase365.mobiliser.custom.project.persistence* in eclipse
21. Navigate to src/main/resources/META-INF/spring
22. Update the following spring configuration.
23. Make sure the configuration files don't get any additional special characters like CR/LF etc. because of Copy/Paste issue.
24. "(…)" is not a valid XML content – this symbol is used for brevity in presenting the XML file content. Having "(…)" in your actual xml configuration file would cause XML parsing errors.

bundle-context.xml

```
(...)

<bean id="daoFactory"
class="com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.factory.HibernateDaoF
actoryImpl">
   <property name="blacklistDao" ref="blacklistDaoImpl" />
   <property name="blacklistTypeDao" ref="blacklistTypeDaoImpl" />
   <property name="wuCustomerDao" ref="wuCustomerDaoImpl"/>
   <property name="departmentDao" ref="departmentDaoImpl"/>
</bean>

<bean id="blacklistDaoImpl" parent="abstractDao"
class="com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.BlacklistDaoHbnImpl"
/>

<bean id="blacklistTypeDaoImpl" parent="abstractDao"
class="com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.BlacklistTypeDaoHbnIm
pl" />

<bean id="wuCustomerDaoImpl" parent="abstractDao"
class="com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.WuCustomerDaoHbnImpl"
/>

<bean id="departmentDaoImpl" parent="abstractDao"
class="com.sybase365.mobiliser.custom.project.persistence.dao.hibernate.DepartmentDaoHbnImpl"
/>

(...)
```

25. Please indent the XML contents by pressing "Ctrl+A" and then "Ctrl+I".
26. Please save the file bundle-context.xml

27. Open a terminal (Applications/Accessories/Terminal)
28. Change to the mobiliser money directory
    $ cd ~/workspace/custom
29. Run following command to build the Mobiliser R5 Vanilla reference application:
    $ mvn clean install
30. Once the maven shows "**BUILD SUCCESS**", go to the following directory:
    $ cd ~/workspace/custom/dist/target
31. You should see following two files:
    $ ls -ll
    com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT-dist-ase.zip
    com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT-scriptarchive-ase.jar

    The **.zip** file is a complete Mobiliser R5 runtime environment including Apache Felix (OSGI Server) and mobiliser deployable services.
    The **.jar** file contains all database scripts used by the dbmaintain utility.

32. Unzip the runtime environment in the same folder ~/workspace/custom/dist/target as follows:
    $ unzip com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT-dist-ase.zip
33. $ cd /home/mobiliser/workspace/custom/dist/target/com.sybase365.mobiliser.custom.project.dist-1.2.0-SNAPSHOT/money
34. Start the money container via bin/startup.sh
35. Examine the mobiliser.log with less (press Shift F for auto update and Crtl C and Q to exit)
36. Check the blue load indicator in the top and **wait until the load drops** this might take 5+ minutes
37. Open Firefox and access the bundle console at http://localhost:8080/system/console
38. Log on with sysmgr/secret (pre-set values)
39. See the bundle status. There should be no bundle in "resolved" or "installed" state

www.sap.com