Storing Java Collections

• Suppose a bean property holds a container, such as a list or set?
• How can the bean be stored in a database?
• Answer: Use a separate table to map each bean the values in its container.
• Example: Person bean class with nickNames property holding a set of strings.
• Database has a person table and a Person_Nickname table.

NickNamesApp & NickNamesWebApp Projects
Declaring a String Set Mapping

- Include a `set` element inside the class element defining a Java—Database mapping.
- Indicate a bean class property and database table name.
- Identify the column holding the bean’s primary key.
- Identify the column holding the string value.

Person.hbm.xml file in NickNames Projects
Hibernate Keeps Java and DB Consistent

• Retrieving a Person from the database:
  – Hibernate fills the `Person` bean using one record from the `Person` table.
  – Hibernate fills the bean’s `nickNames` property using records from the `Person_NickName` table.

• Adding a Person bean to the database:
  – Hibernate adds record to `Person` table.
  – Person bean includes a set (`nickNames`) of strings.
  – Hibernate also adds one record to `Person_NickName` table for each string in the `nickName` set.

• Deleting a Person bean from the database:
  – Hibernate deletes a record from `Person` table.
  – Hibernate deletes corresponding rows from `Person_NickName` table.
Check Box Input

<form method="post" action="Controller">
    <input type="checkbox" name="extra" value="sprinkles" checked>
    <input type="checkbox" name="extra" value="fudge">
    <input type="checkbox" name="extra" value="cream" checked>
    <input type="submit" name="confirmButton" value="Confirm">
</form>

User may check zero or more boxes. One or more may be initially “checked”. Parameters are encoded in query string as multiple values:

?extra=sprinkles&extra=cream
HibernatingBeanMultiValued

• HTML inputs with multiple values:
  – Check Box.
  – Multiple Selection Menu.
• Annotations and reflection to provide convenient access to multiple values in JSPs.
• Annotations for storing multiply valued properties in a database.
Multiple Selection Menu

<form method="post" action="Controller">

<select name="team" multiple size="2">
    <option value="heat"> Heat
    <option value="marlins" selected> Marlins
    <option value="dolphins"> Dolphins
    <option value="panthers" selected}> Panthers
</select>

<input type="submit" name="confirmButton" value="Confirm">

</form>

User may check zero or more options. One or more may be initially “selected”. Parameters are encoded in query string as multiple values:

?team=marlins&team=panthers
Access of Multiple Valued Query Parameters

```java
request.getParameterValues("extra")
// Returns array: ["sprinkles","cream"]

request.getParameterValues("team")
// Returns array: ["marlins","panthers"]
```
Filling Bean with Multiple-Valued Properties

// Set multiple-valued properties to null.
resetNullable();

// Copy parameters from request to bean.
fillBeanFromRequest(data);

If user selects zero items, the property won’t appear in the query string at all, so `fillBeanFromRequest` will do nothing with that property. Thus the need to nullify the multiply valued properties first.
Maintain Maps of Checked and Selected Items

• Bean annotations indicate multiply valued bean properties:
  - @SetByAttribute(type=AttributeType.CHECKED)
  - @SetByAttribute(type=AttributeType.SELECTED)

• HelperBase class has methods that maintain maps of maps storing values of these properties.

• JSPs can access these maps using EL via the controller helper object stored in the session.
Annotations for Persisting Multiply-Valued Properties

- `@CollectionOfElements`

- `@IndexColumn(name="column_name", base=0)`
Mapping Other Types of Collections

• Hibernate supports mapping of set, list, map, bag, array and primitive-array types.

• Complications due to Java not supporting its own bag type.
Persistent Collection Challenges

- Set
- Array
- List
- Map
- MultiMap
Hibernate Sessions and Transactions

- Transactions happen between opening and closing of a session.
- Database operations happen between begin and commit of a transaction.
- Transaction is rolled back if an error occurs before the commit operation.
- Transaction happens as a single atomic operation.
HibernateHelper

public void updateDB(Object obj)

• If the object originally came from database (via Hibernate) then the update operation will replace the database record from which the object was constructed.

• If the object was created in Java, then it will be added as a new record in the database, even if it is a duplicate.
HibernateHelper

```java
public List getListData(Class classBean, String... parameters)
```

- Retrieves beans from database table meeting conditions in parameters.
- First constructs a `Criteria` object from `classBean`.
- Then successively adds `Restriction` objects to the `Criteria`.
- Uses the `Restriction` class like method to handle regular expressions.
Unloading Database Drivers

- Database drivers must be unregistered when the web application is stopped.
- Otherwise, JVM will have a memory leak.
- Use a ServletContextListener:
  - Declare listener class in web.xml file.
  - Implement contextDestroyed method to unregister the drivers.