Server Side Web Development

Lecture 6 – Data Access Objects (DAO)

Lecturer: Mr Daryl Parker (dparker@ait.ie)

DAO - Recap

- It is good practice to encapsulate access to the database using Data Access Objects (DAO).
- This class will be responsible for saving objects, loading objects etc.
- Only a **single instance** of the DAO class will be needed – it is a singleton class.
- An **enum** is often used to implement singleton classes

DAO - Recap

- Normally an enum has more than one possible value.
- UserDAO.instance refers to the singleton object

```
public enum UserDAO {
    instance;
```

Data Access Objects - Overview

- A layer in your application where all the database access code goes.
- We are going to create a DAO which just creates objects rather than reading from the database.
- Later we will be accessing the database in the DAO

DAO – Model (User.java).

```java
package edu.ait.model;

public class User {
    private String name;
    private String address;

    public User(String name, String address) {
        super();
        this.name = name;
        this.address = address;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }
}
```

UserDAO – DAO for User Model Class

```
package edu.ait.model;

import java.util.ArrayList;
import java.util.List;

public enum UserDAO {
    instance;
    List<User> users;

    private UserDAO() {
        users = new ArrayList<User>( initialCapacity: 3);
        User user1 = new User( name: "Paul", address: "Athlone");
        users.add(user1);
        User user2 = new User( name: "John", address: "Tullamore");
        users.add(user2);
        User user3 = new User( name: "David", address: "Roscommon");
        users.add(user3);
    }

    public List<User> list() {
        return users;
    }
}
```

Belongs in the model package

Imports for Java Collections

Public Enumeration definition with single instance

List to hold our Users (not needed if using DB)

Populate some instances of our User model into our users list. Normally would be read from Database.

List method (equivalent to Read operation in CRUD)
DAO should support full CRUD operations

UserControllerServlet

- In doPost()

```
List<User> userList = UserDAO.instance.list();
```

- Normally the list() methods would read a list of Users from the database.
- Should have other CRUD mapped methods
  - Save (Create)
  - Update (Update)
  - Delete (Delete)