Working With JPA

Getting Started:

1. When starting a JPA project you create your project as normal in eclipse.
2. You then need to make a number of imports into this project
    2.1. Accessing the build path in eclipse of the project you need to import the following
        - Every JAR file in the hibernate "required" folder (Hibernate must be downloaded)
        - The "ConnectorJ" jar file from your MySQL folder(should be in your c://program files folder)
3. Then you must create 3 packages
        - Entites – This will hold your classes
        - DAO- This will hold the DAO version of your classes that will allow for persist merge and remove funtionaility
        - Default – Where your project will run
4. Create a folder in the src folder called META-INF
    4.1. In this folder create a file called peristence.xml
    4.2. Code should be taken from Brightspace
    4.3. The only things you should change are the following
        - **Persistence-unit – VALUE SHOULD BE SET TO WHAT YOU WANT TO CALL YOUR PERSISTENCE UNIT**
        - **Javax.persistence.jdbc.url - This tells the project what database is being used so after the 127.0.0.1:3306/ change the database name to what you called it in MYSQL**

Subscriber, Profile, Comment Example:

The premise of this example is the following:

- **Each subscriber has an id, username and password and a profile attached to it**
- **Each profile has an id and description attached to it**
- **Each profile can have many comments**
    - **Each Comment has an id and content associated with it**

Things to note

Persisting Comments

- Profile takes multiple comments so instead of taking comments objects it takes a List of comments , **Arraylist does not work directly so use list and let it equal an arraylist**
- This list of comments has a relationship of "OneToMany" so we must use this annotation
    - @OneToMany
    - private List<Comment> comments = new ArrayList<Comment>();
- The profile class will get another constructor as the profile that is persisted to the database does not need a list of comments it's optional

- The profile class will get two void methods “addComments” and “removeComments” both will call the add and remove methods of the arraylist.
- In the test class the Comment DAO must be created first along with the comments
- The comments must be persisted first as they need id’s using the comment Dao persist method
- Then once the profile is created we can then call the profile.addComments() method passing in any of the created comments , the profile is then persisted and then the subscriber is

Getting Data from the database

- For example pulling the subscribers from the database
    - In the Subscriber class we can either use @NamedQuery for one query or @NamedQueries for multiple queries
        - In this case we will use NamedQueries and NamedQuery nested inside of it
        - When using NamedQuery it has two parameters name and query , name being the name of the query and query being the actual query
    - In the SubcriberDAO class we will create a public method based off the list object being cast to a subscriber
        - In this method we will declare an entity manager
        - Then we will create a list which will be set to entity manager which calls the “createNamedQuery” method where we pass in the name of the query we listed in the subscriber class and then we call the “getResultList” on it which should return the list of subscribers
        - We then close the entity manager
        - And then return the list of subscribers
        - **We can also use parameters for example searching by name**
            - **The only difference is**
                - The method will take in the variable in this case String name
                - When the list is created before calling the “getResultList” method we will use “setParamter” and pass in a name and the value that the method takes in being name then call “getResultList” which returns the entity with that name
                - Then we close the entity manager
                - We then create a new subscriber
                - Create a for loop looping through the list of subscribers and then setting the found subscriber to the new one
                - Then it is returned
    - Then in the Test class we can then call the method appropriately
        - So for returning all subscribers
            - We create a list of subscribers
            - Then set that list to equal the subscriberDAO.getAllSubscribers method
            - We can then do a for loop which iterates through the list returning each profile
        - When returning a subscriber by name we just create a subscriber then cal the getSubscriberbyUsername method that will get the user name and then return this user

- **Same applies for profile and comments**