



## XÂY DỰNG ỨNG DỤNG CRUS VỚI JPA

---

### LẬP TRÌNH JAVA #4 (P2.2)

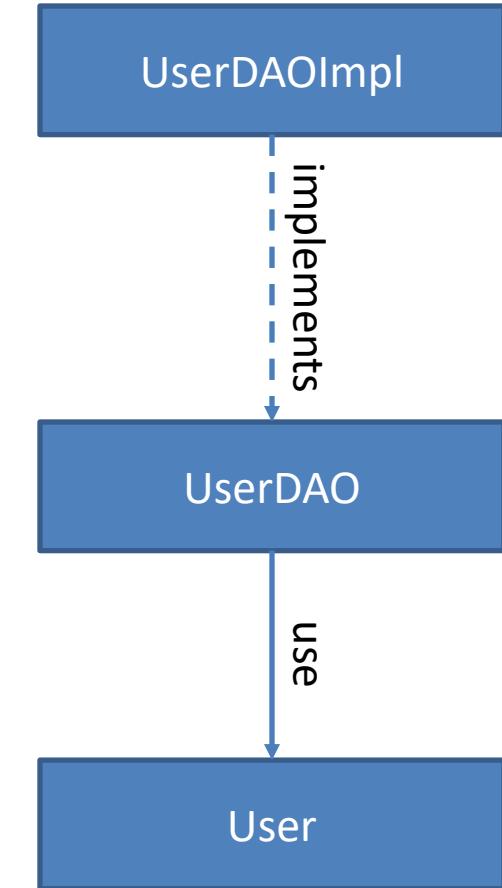
- Xây dựng lớp tiện ích XJPA
- Xây dựng DAO
- Tích hợp DAO vào Servlet (Controller)



```
public class XJPA {  
    static EntityManagerFactory factory;  
    static {  
        factory = Persistence.createEntityManagerFactory("PolyOE");  
    }  
    public static EntityManager getEntityManager(){  
        return factory.createEntityManager();  
    }  
}
```

*PolyOE là PersistentUnit được khai báo trong persistence.xml*

```
public interface UserDAO {  
    /**Truy vấn tất cả*/  
    List<User> findAll();  
    /**Truy vấn theo mã*/  
    User findById(String id);  
    /**Thêm mới*/  
    void create(User item);  
    /**Cập nhật*/  
    void update(User item);  
    /**Xóa theo mã*/  
    void deleteById(String id);  
}
```



- ❑ Khai báo các phương thức cần thiết làm việc với thực thể User

```
public class UserDAOImpl implements UserDAO {  
    EntityManager em = XJPA.getEntityManager();  
    @Override  
    protected void finalize() throws Throwable{em.close();}  
    @Override  
    public List<User> findAll() {...}  
    @Override  
    public User findById(String id) {...}  
    @Override  
    public void create(User entity) {...}  
    @Override  
    public void update(User entity) {...}  
    @Override  
    public void deleteById(String id) {...}  
}
```

*Chú ý: finalize() sẽ được gọi khi đối tượng bị giải phóng khỏi bộ nhớ (dọn rác)*

```
@Override
public List<User> findAll() {
    String jpql = "SELECT o FROM User o";
    TypedQuery<User> query = em.createQuery(jpql, User.class);
    return query.getResultList();
}

@Override
public User findById(String id) {
    return em.find(User.class, id);
}
```

```
@Override  
public void create(User entity) {  
    try {  
        em.getTransaction().begin();  
        em.persist(entity);  
        em.getTransaction().commit();  
    } catch (Exception e) {  
        em.getTransaction().rollback();  
    }  
}
```

```
@Override  
public void update(User entity) {  
    try {  
        em.getTransaction().begin();  
        em.merge(entity);  
        em.getTransaction().commit();  
    } catch (Exception e) {  
        em.getTransaction().rollback();  
    }  
}
```

```
@Override  
public void deleteById(String id) {  
    User entity = em.find(User.class, id);  
    try {  
        em.getTransaction().begin();  
        em.remove(entity);  
        em.getTransaction().commit();  
    } catch (Exception e) {  
        em.getTransaction().rollback();  
    }  
}
```

- ❑ Bổ sung dòng code tạo đối tượng DAO sau đây vào đầu phương thức service()
  - ❖ `UserDAO dao = new UserDAOImpl();`
- ❑ Gọi các phương thức truy xuất dữ liệu của **dao** tại các vị trí phù hợp của service() theo hướng dẫn sau:

Path	Code
<code>path.contains("create")</code>	<code>dao.create(form);</code> <code>form = new User();</code>
<code>path.contains("update")</code>	<code>dao.update(form);</code>
<code>path.contains("delete")</code>	<code>dao.deleteById(form.getId());</code> <code>form = new User();</code>
<code>path.contains("edit")</code>	<code>form = dao.findById(id);</code>
Thay <code>new ArrayList&lt;&gt;()</code> bằng <code>dao.findAll()</code> để truy vấn tất cả users	

```
UserDAO dao = new UserDAOImpl();
String path = req.getServletPath();
if (path.contains("edit")) {
    String id = req.getPathInfo().substring(1);
    form = dao.findById(id);
} else if (path.contains("create")) {
    dao.create(form);
    form = new User();
} else if (path.contains("update")) {
    dao.update(form);
} else if (path.contains("delete")) {
    dao.deleteById(form.getId());
    form = new User();
} else {
    form = new User();
}
```



# DEMOSTATION

---

- ✓ Xây dựng lớp tiện ích XJPA
- ✓ Xây dựng DAO
- ✓ Tích hợp DAO vào Servlet (Controller)





Cảm ơn