PersistentManager and Transaction

PersistentManager is used to manage database connection, state of the persistent objects and transaction when the application is running. A database transaction is a unit of the application. It is used to keep integrity of your data in database. Our Persistent Library provides a transaction API for you to create, commit and rollback the transaction for your application.

Start Transaction

When you need to start a transaction, you need to get the session from the PersistentManager and call the beginTransaction function (PersistentManager.instance().getSession().beginTransaction();). Here is a sample:

```
PersistentTransaction t = PersistentManager.instance().getSession().beginTransaction();
```

Commit Transaction

After you have inserted/updated/deleted data, you can call the commit function of the transaction object to commit your changes to Database.

```
t.commit();
```

Rollback Transaction

In case there are any exception, you can call the rollback function to cancel all the operation.

```
t.rollback();
```

Sample Code

The following is an example of using the transaction API.

```
private Course fireOK() throws PersistentException {     PersistentTransaction t = SchoolSystemPersistentManager.instance().getSession().beginTransaction();     try {Course lCourse = CourseFactory.createCourse();        lCourse.setTitle(getTitleTextField().getText());        lCourse.setDescription(getDescriptionTextField().getText());        lCourse.setTeacher(_teacher);        _teacher.save();        t.commit();     return lCourse;     } catch (Exception e) {        e.printStackTrace();        t.rollback();     }     return null;  }
```