OAC : **public** ArrayList<Request> getViewRequests(String empOsName) {

ArrayList<Request> requestObj = **null**;

**try** {

Session session =SessionFactoryHelper.*getSession*();

// String query = "from Request where lower(requesterId)='"

// + empOsName.toLowerCase().trim()

// + "' order by requestId desc";

// Query queryObj = session.createQuery(query);

// if(!queryObj.list().isEmpty()){

// requestObj = (ArrayList<Request>) queryObj.list();

// }

Criteria criteria = session.createCriteria(Request.**class**);

criteria.add(Restrictions.*eq*("requesterId", empOsName.toLowerCase().trim()).ignoreCase());

ProjectionList proList = Projections.*projectionList*();

proList.add(Projections.*property*("requestId"),"requestId");

proList.add(Projections.*property*("raiseDt"),"raiseDt");

proList.add(Projections.*property*("reqFullName"),"reqFullName");

proList.add(Projections.*property*("individualName"),"individualName");

proList.add(Projections.*property*("empstatus"),"empstatus");

proList.add(Projections.*property*("cardType"),"cardType");

proList.add(Projections.*property*("firstTime"),"firstTime");

proList.add(Projections.*property*("status"),"status");

criteria.setProjection(proList);

criteria.addOrder(Order.*desc*("requestId"));

criteria.setResultTransformer(Transformers.*aliasToBean*(Request.**class**));

requestObj=(ArrayList<Request>) criteria.list();

} **catch** (Exception e) {

e.printStackTrace();

*logger*.info("get getViewRequests is " , e);

} **finally** {

SessionFactoryHelper.*closeSession*();

}

**return** requestObj;

}

There are times we have a class, we would like to fill with data according the data returned from a query. The class is a simple POJO and not an Hibernate entity, so Hibernate won’t recognize this class.

This can be done in Hibernate by using Transformers. Let’s have a look on a simple example, showing how Transformers can be used. First, let’s have a look at a simple POJO class named: “UserActivityStat”.

This class contains some statistical information. We would like to fill the statistical information of an instance, directly from running an Hibernate HQL.

public class UserActivityStat

{

private int totalPhotos;

private int totalViews;

public UserActivityStat() {}

public int getTotalPhotos() { return totalPhotos; }

public void setTotalPhotos(int totalPhotos) {this.totalPhotos = totalPhotos;}

public int getTotalViews() { return totalViews; }

public void setTotalViews(int totalViews) { this.totalViews = totalViews; }

}

Now, let’s have a look at a simple method, that uses hibernate HQL and the Transformers class to fill “UserActivityStat” instance with data:

public UserActivityStat getUserActivityStat(User user)

{

return (UserActivityStat)hibernateSession.createQuery("select count(\*) as totalPhotos, sum(p.views) as totalViews " + "from Photo p where p.user = :user " +

"p.dateCreated <= :now").setParameter("user", user).setTimestamp("now", new Date()).

setResultTransformer(Transformers.aliasToBean(UserActivityStat.class)).uniqueResult();

}

Note, that each of the 2 columns has an alias. This alias must be the name of the property on the “UserActivityStat” class. Also note for the use of the “setResultTransformer” along the “Transformers” class.