

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAOUB

Objectif de l'atelier.....	1
Séparation des couches en projets.....	1
Adaptation de la couche métier à RMI.....	2
Création d'un serveur RMI	4
Maintenant c'est la classe ServerRmi qui va attendre les appels de la couche présentation. 5	
Adaptation de la couche présentation à RMI	5
Exécution des deux projets.....	6
Création d'un autre client de la couche métier.....	7

Objectif de l'atelier

L'objectif de cet atelier est de séparer la couche métier de la couche présentation. Il s'agit de créer un projet pour la couche métier et un autre projet pour la couche présentation.

Vous allez apprendre à:

- *Séparer la couche présentation de la couche métier*
- *Invoquer les services proposés par la couche métier à travers RMI (**Remote Method Invocation**)*
- *Implémenter le patron de conception Value Object.*
- *Implémenter le patron de conception MVC2.*

Séparation des couches en projets

Créer deux projets sur Eclipse :

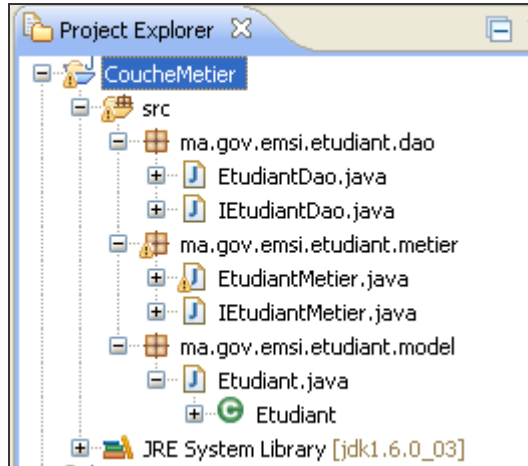
*Un projet **Java** portant le nom CoucheMetier*

*Un projet **Web** portant le nom CouchePresentation*

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAOUB



Dans le projet « CoucheMetier » vous devriez garder.

Le package `ma.gov.emsi.etudiant.dao` avec `IetudiantDao` et `EtudiantDao`

Le package `ma.gov.emsi.etudiant.metier` avec `IetudiantMetier` et `EtudiantMetier`

Le package `ma.gov.emsi.etudiant.model` avec la classe `Etudiant`

Ces classes sont déjà créées lors de notre dernier atelier.

Dans le projet « CouchePresentation » vous devriez avoir :

Le package `ma.gov.emsi.etudiant.presentation` avec `EtudiantServlet`

*Le package `ma.gov.emsi.etudiant.metier` avec `IetudiantMetier`. **Et surtout pas la classe `EtudiantMetier`***

Le package `ma.gov.emsi.etudiant.model` avec la classe `Etudiant`

*Dans ce projet « CouchePresentation » considéré Client de service, Nous aurons besoin du **contrat** et du **modèle** proposé par le fournisseur de service « CoucheMetier »*

Adaptation de la couche métier à RMI

Nous allons maintenant proposer maintenant les services de la couche métier en utilisant la technologie RMI.

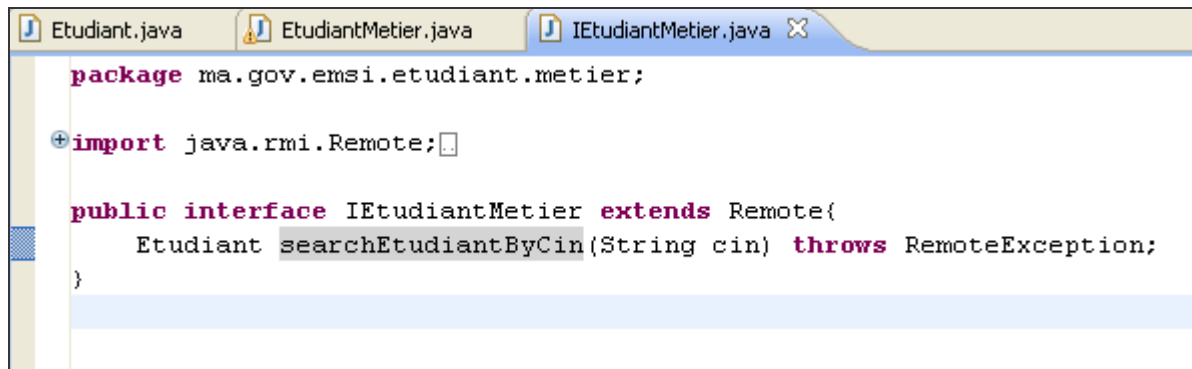
Modifions le contrat de la couche métier.

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAOUB

Dans l'interface *IEtudiantMetier* rajouter un lien d'héritage avec l'interface **Remote** (*java.rmi.Remote*) et ajouter **throws RemoteException** à la méthode *searchEtudiantByCin* proposée par la couche métier.



```
package ma.gov.emsi.etudiant.metier;

import java.rmi.Remote;

public interface IEtudiantMetier extends Remote{
    Etudiant searchEtudiantByCin(String cin) throws RemoteException;
}
```

N'oublier pas de mettre à jour le contrat déposé au niveau du projet « *CouchePresentation* »

Modifions l'implémentation de la couche métier.

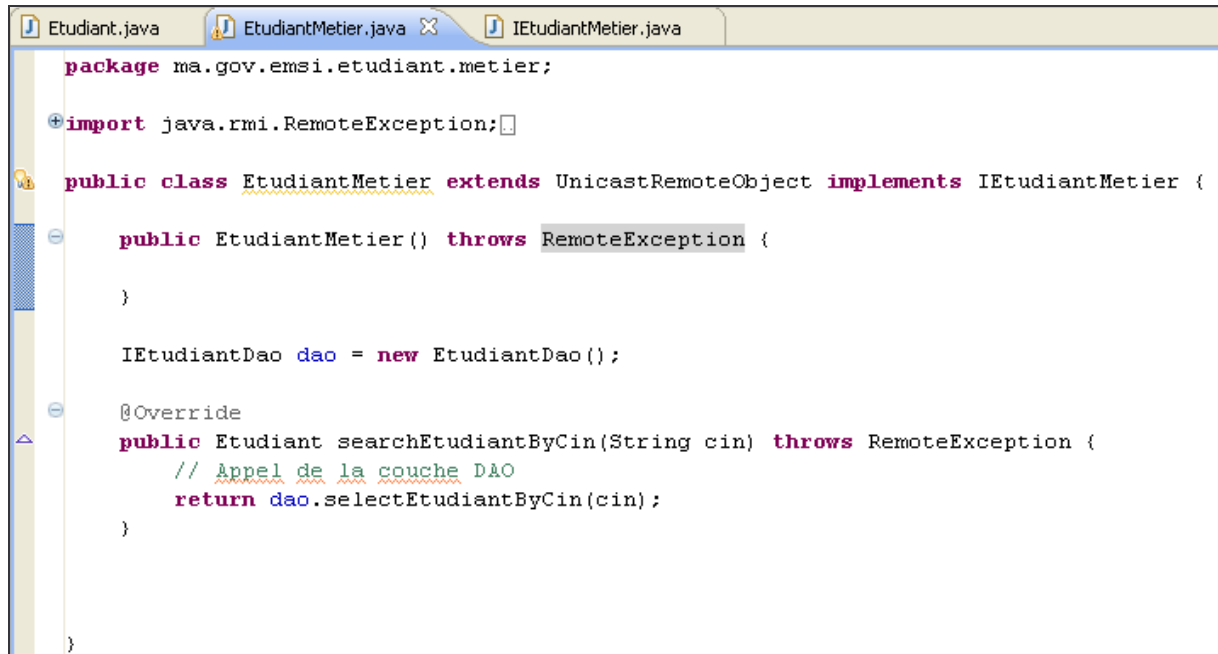
Dans la classe *EtudiantMetier* ajouter un lien avec la classe **UnicastRemoteObject**

Surcharger le constructeur de la classe *EtudiantMetier* pour ajouter le **throws RemoteException**

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAHOUB



```
package ma.gov.emsi.etudiant.metier;

import java.rmi.RemoteException;

public class EtudiantMetier extends UnicastRemoteObject implements IEtudiantMetier {

    public EtudiantMetier() throws RemoteException {

    }

    IEtudiantDao dao = new EtudiantDao();

    @Override
    public Etudiant searchEtudiantByCin(String cin) throws RemoteException {
        // Appel de la couche DAO
        return dao.selectEtudiantByCin(cin);
    }

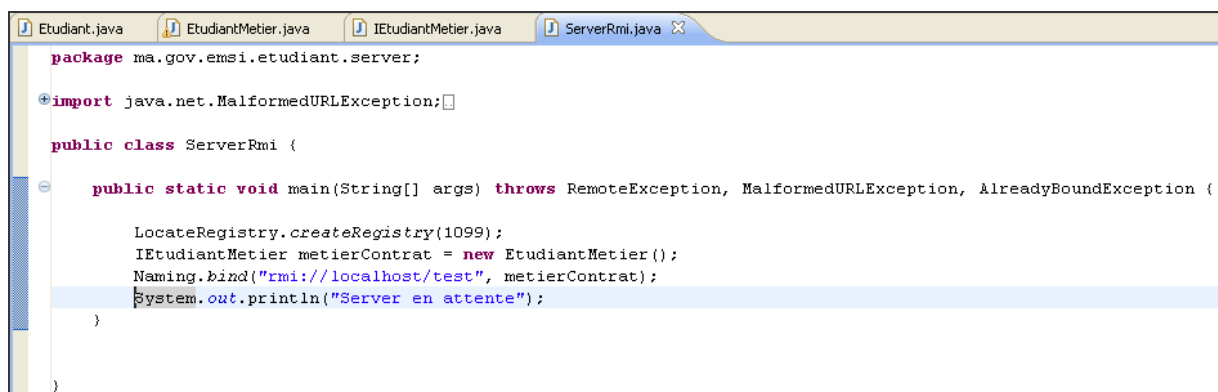
}
```

Création d'un serveur RMI

Au niveau de la couche métier créant un package `ma.gov.emsi.etudiant.server`
Avec une classe `ServerRmi` contenant la méthode `main`.

Dans cette classe :

- ✓ Réserver le port 1099 de RMI
- ✓ Créer un objet de la couche métier
- ✓ Mettre l'objet nouvellement crée dans l'annuaire RMI



```
package ma.gov.emsi.etudiant.server;

import java.net.MalformedURLException;

public class ServerRmi {

    public static void main(String[] args) throws RemoteException, MalformedURLException, AlreadyBoundException {

        LocateRegistry.createRegistry(1099);
        IEtudiantMetier metierContrat = new EtudiantMetier();
        Naming.bind("rmi://localhost/test", metierContrat);
        System.out.println("Server en attente");
    }

}
```

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAHOU

Maintenant c'est la classe `ServerRmi` qui va attendre les appels de la couche présentation.

Adaptation de la couche présentation à RMI

Dans le contrôleur de la couche présentation, modifier la méthode `doGet()` comme suit :

- ✓ *Avoir un objet de la couche métier à travers RMI*

```
IEtudiantMetier metier = (IEtudiantMetier)Naming.lookup("rmi://localhost/test");
```

- ✓ *Faire un appel au serveur RMI de la couche métier qui propose les services*

```
Etudiant model= metier.searchEtudiantByCin(cin);
```

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAHOUB

```
EtudiantServlet.java
package ma.gov.emsi.etudiant.presentation;
import java.io.IOException;

public class EtudiantServlet extends HttpServlet{

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        try {
            // 0- Création d'un objet de la couche metier
            IEtudiantMetier metier = (IEtudiantMetier)Naming.lookup("rmi://localhost/test");
            // 1 - Recupération de la cin à partir du formulaire
            String cin=req.getParameter("cinFom");
            // 2- Appel de la couche metier
            Etudiant model= metier.searchEtudiantByCin(cin);
            // 3- Mise à jour de la requete HTTP
            req.setAttribute("MonEtudiant", model);
            // 4- Redirection vers une autre vue
            //(etudiantVue.jsp par exemple)

        } catch (NotBoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        req.getRequestDispatcher("/vues/etudiantVue.jsp").
            forward(req, resp);
    }
}
```

Exécution des deux projets

Lancer la classe *ServerRmi* comme application java. Le serveur RMI sera ainsi en attente des appels des clients notamment les appels de la couche présentation.

Lancer le projet « *CouchePresentation* » sur le Serveur Tomcat.

Pour assurer la communication en RMI entre la couche présentation et la couche métier, la classe *Etudiant* doit être sérialisable

Ajouter un lien d'implémentation à la classe *Etudiant* de la couche métier et à celle de la couche présentation.

Refaire l'exécution.

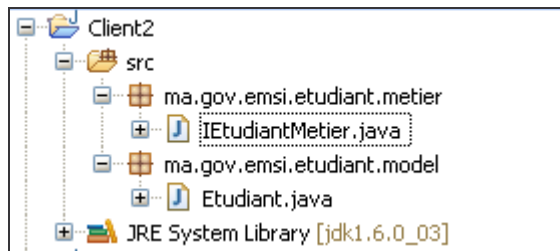
Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAHOU

Création d'un autre client de la couche métier

Créer un nouveau projet java sous le nom «Client2».



De la même façon, mettre dans ce projet :

Le package `ma.gov.emsi.etudiant.metier` avec `IetudiantMetier`. **Et surtout pas la classe `EtudiantMetier`**

Un client de la couche métier doit contenir le modèle et le contrat

Créer un autre package `ma.gov.emsi.etudiant.client` dans le projet « Client2 » avec une classe `Client` contenant la méthode `main`.

Mettre dans la méthode `main` le contenu d'un client RMI.

Formation Framework Struts1&2

Atelier 3 : Création d'une application web selon le patron de conception Value Object

Encadré par M.BOULCHAOUB

```
/**
 * @param args
 * @throws NotBoundException
 * @throws RemoteException
 * @throws MalformedURLException
 */
public static void main(String[] args)
throws MalformedURLException, RemoteException, NotBoundException {

    IEtudiantMetier metier =
        (IEtudiantMetier)Naming.lookup("rmi://localhost/test");
    // 1 - Recupération de la cin à partir du formulaire
    String cin="J300299";
    // 2- Appel de la couche metier
    Etudiant model= metier.searchEtudiantByCin(cin);

    System.out.println(model);
}
```