

# INTÉGRATION D'APPLICATIONS

TP1 Configuration de l'environnement  
pour l'intégration d'applications

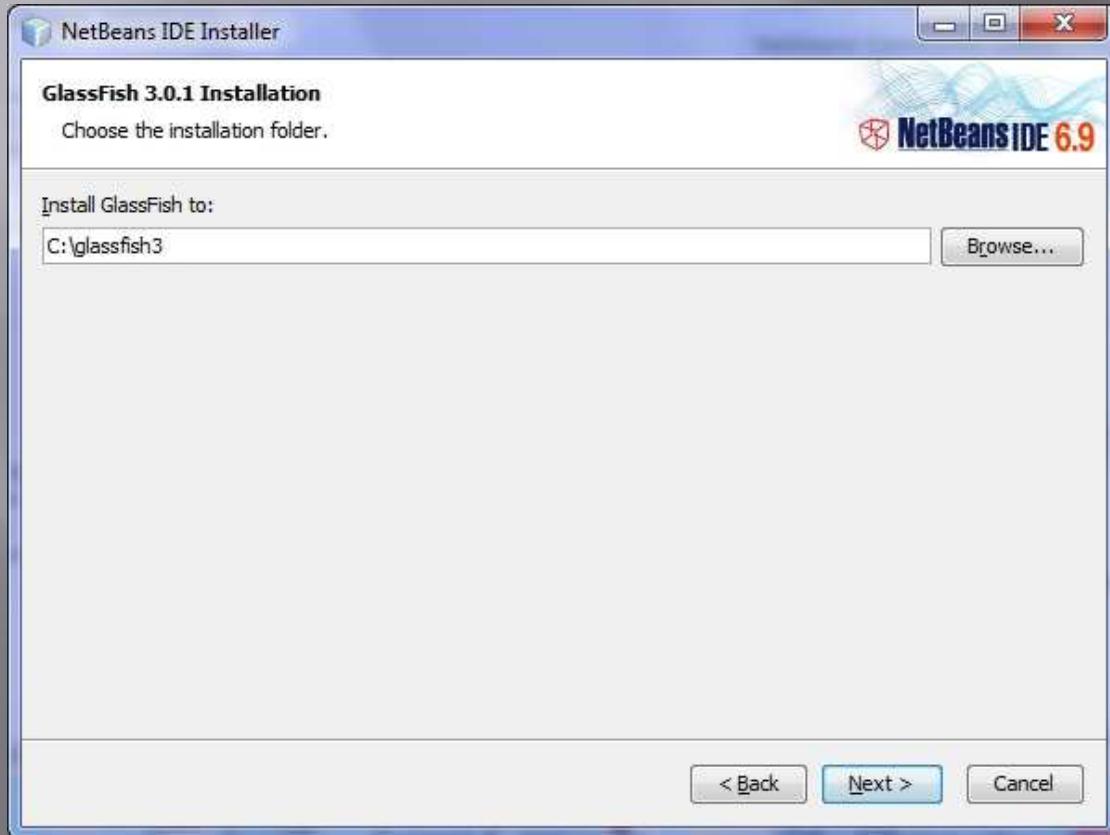
# Le IDE NetBeans

- ❑ Qu'est-ce qu'un IDE ? Et quels en sont les avantages (et inconvénients) ?
- ❑ Qu'est-ce que NetBeans ? Quels sont ses principales fonctionnalités ? Présenter son interface et le rôle des différents éléments.



# GlassFish

- Qu'est-ce que GlassFish ? Que permet son enregistrement dans NetBeans ?

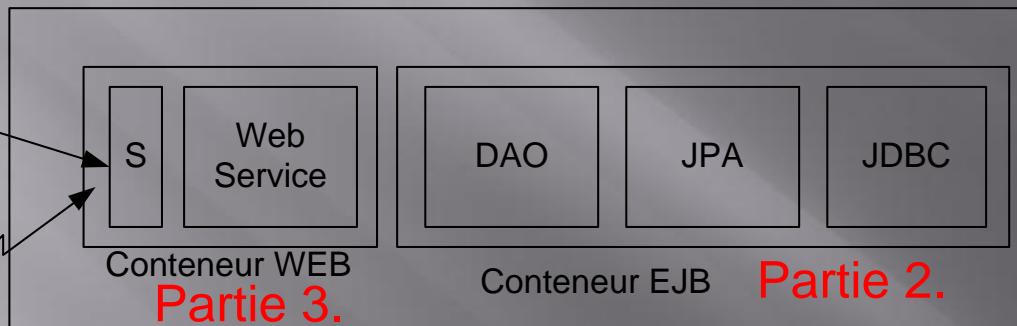


Penser à mettre  
un chemin  
d'accès sans  
espace

# Architecture d'application

Entreprise Application

Partie 4.

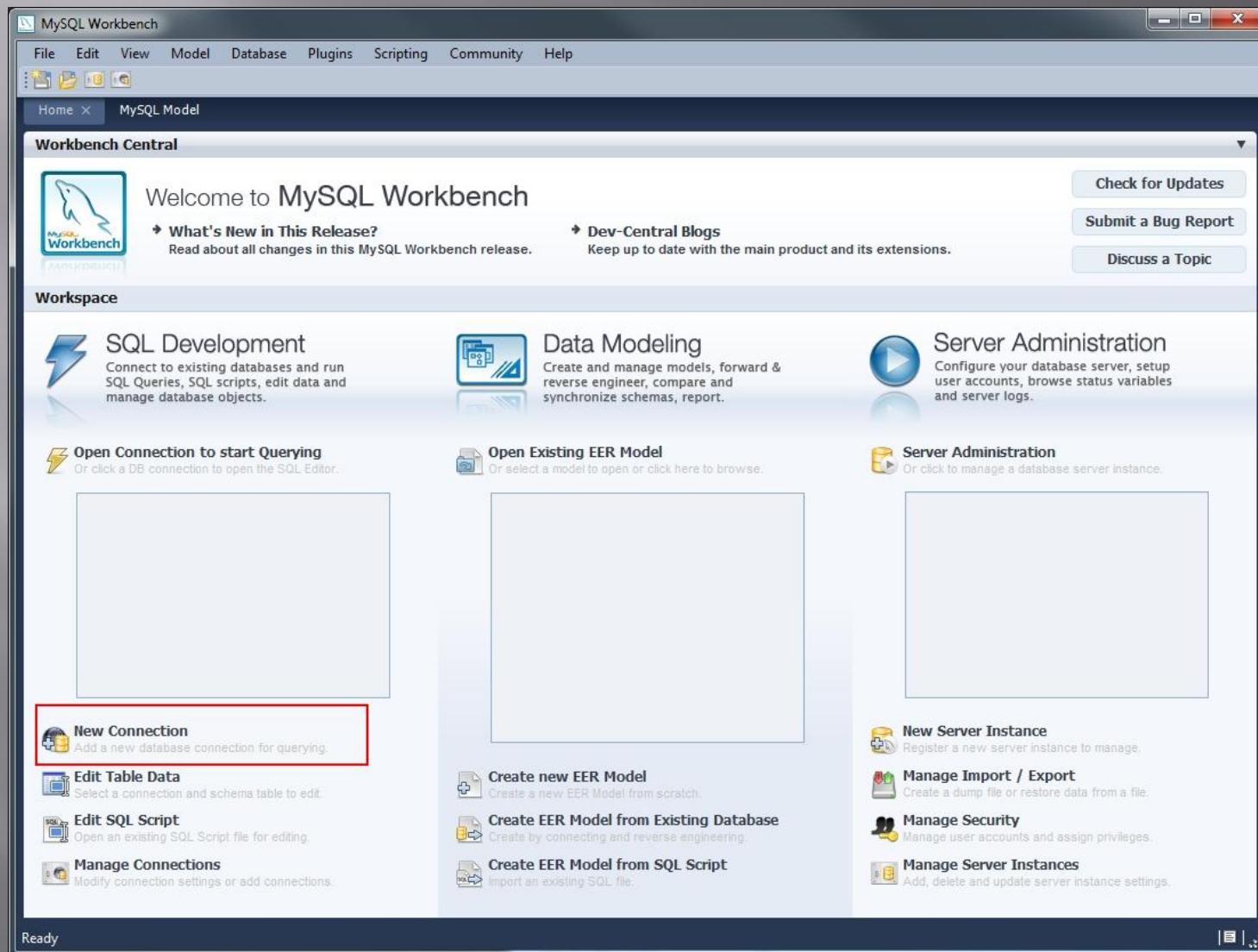


Partie 5.

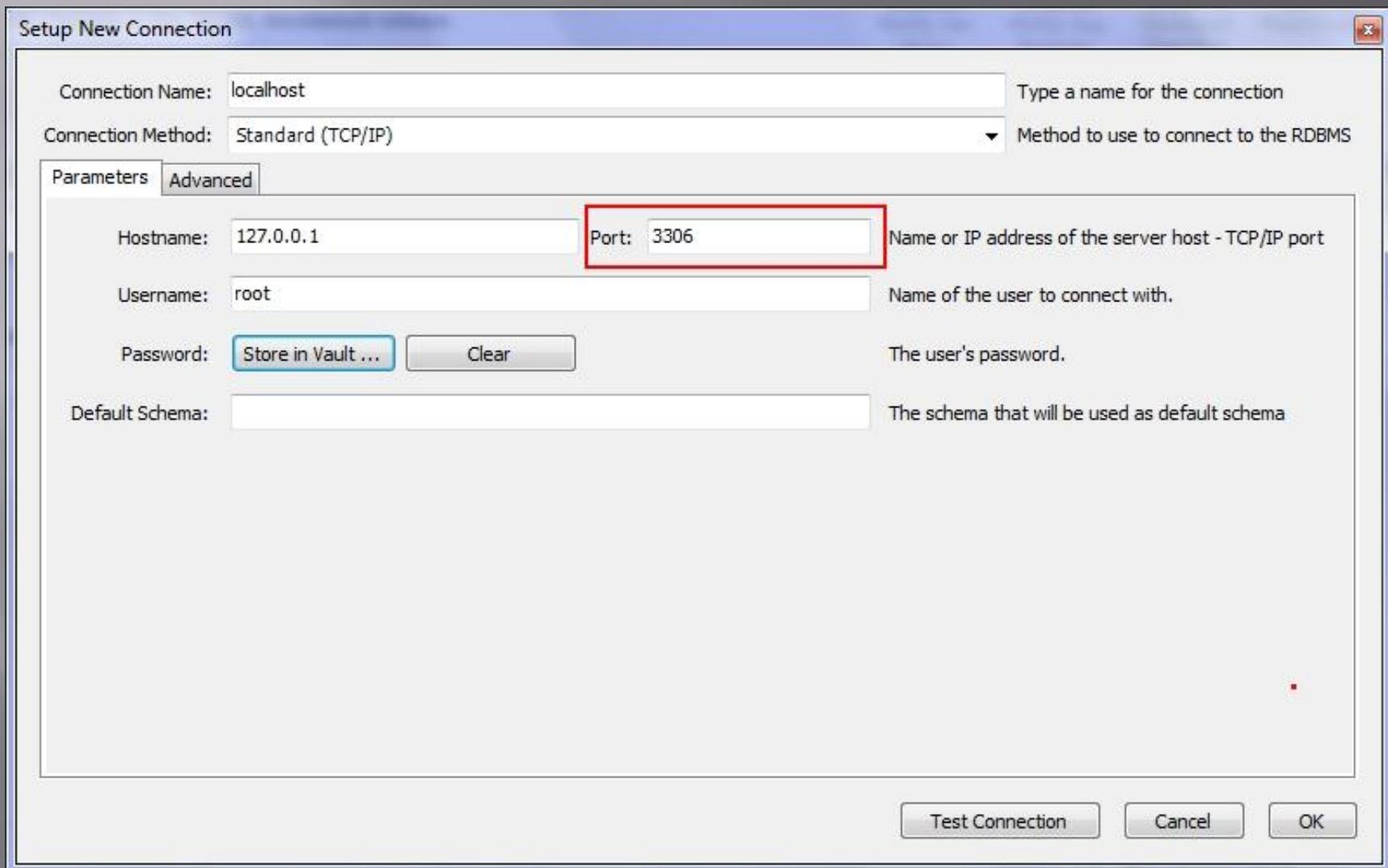
# Création d'une base de données

- ❑ Qu'est-ce que MySQL ?
- ❑ Pourquoi ce choix ?
- ❑ Quels sont les alternatives ?
- ❑ Qu'est-ce qu'un SGBD ?

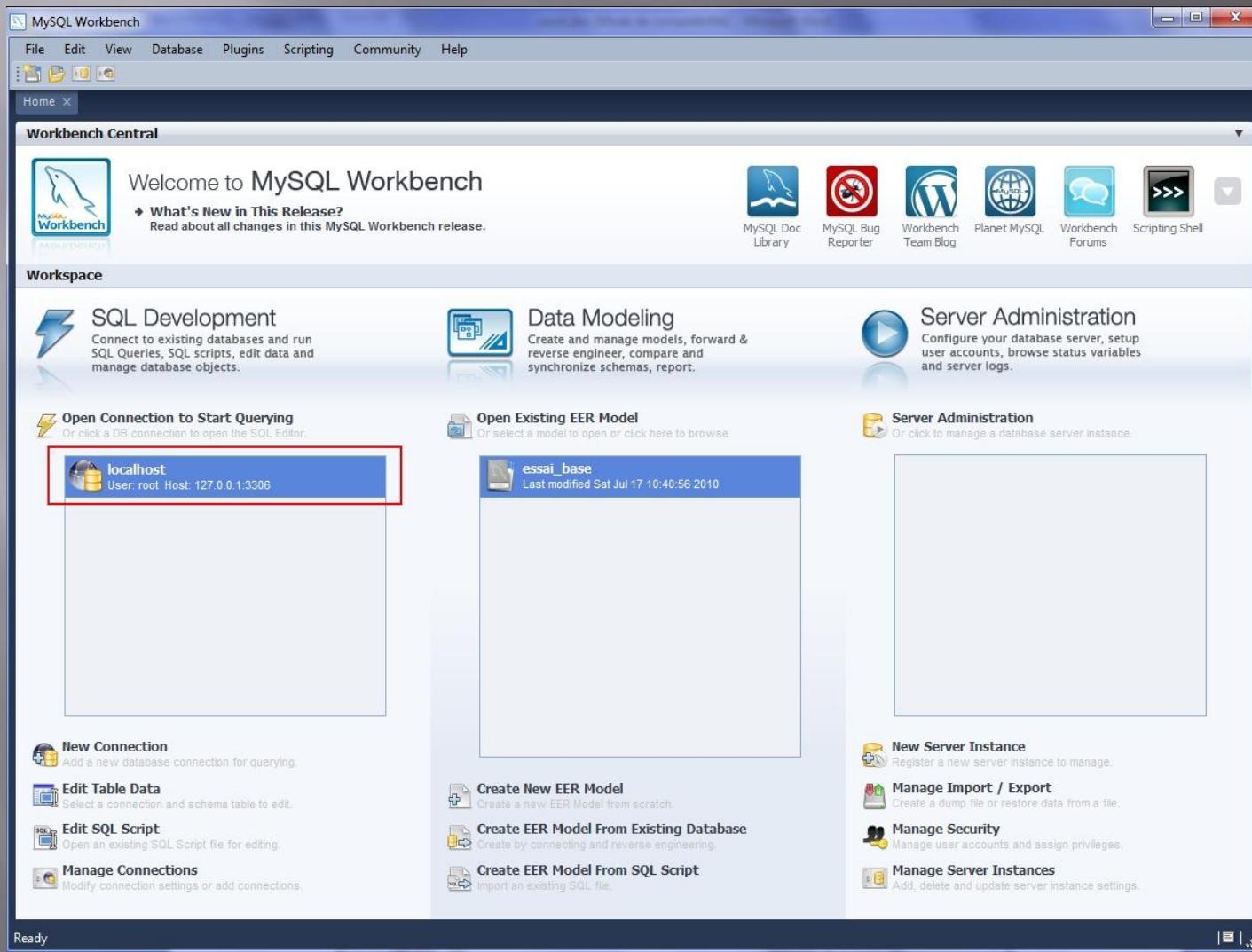
# Lancer MySQL Workbench



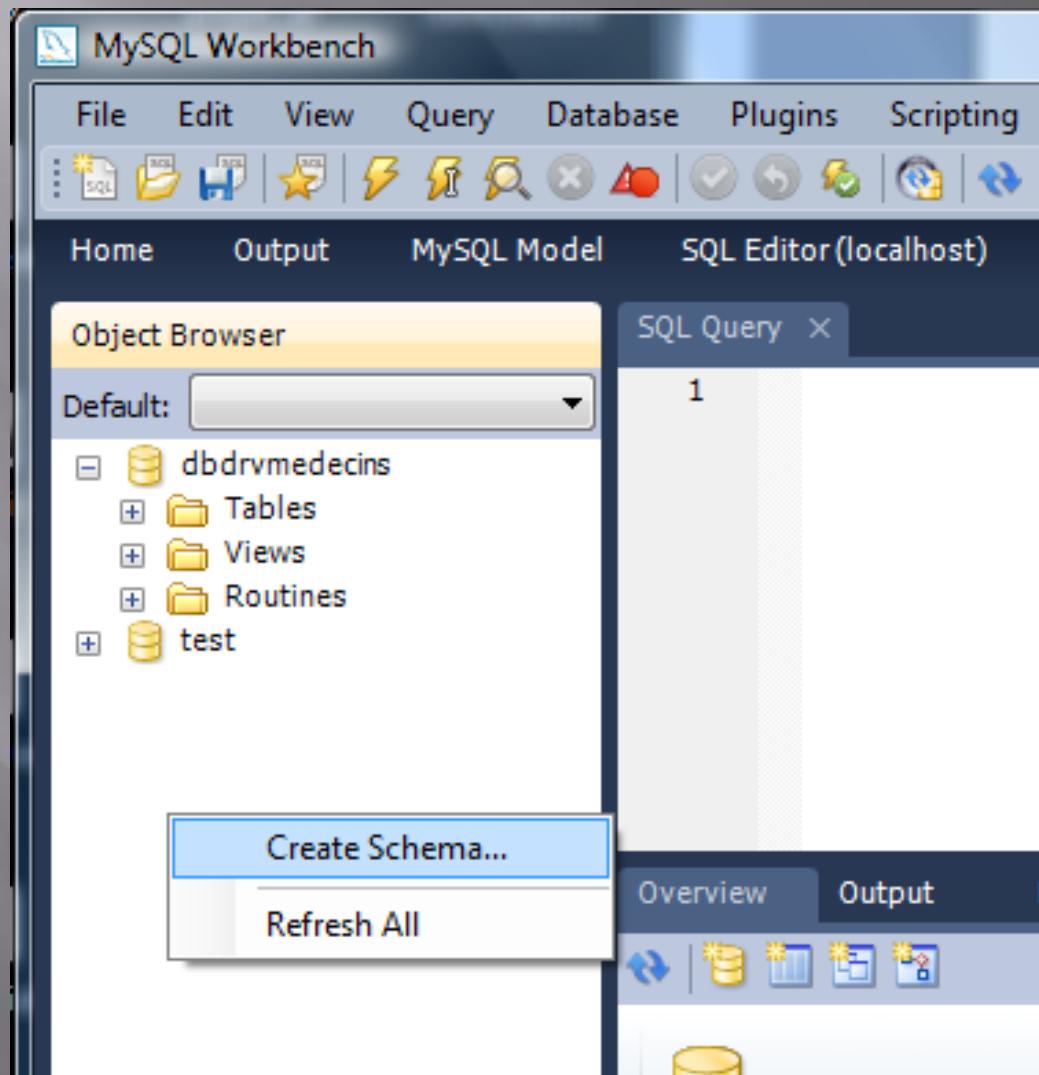
# Configurer le port



# Une nouvelle connexion est créée



# Créer un nouveau schéma



# Créer la nouvelle base médecine

The screenshot shows the MySQL Workbench interface. A modal dialog titled "new\_schema" is open in the foreground, prompting for a schema name ("base\_medecin") and collation ("Server Default"). The main workspace displays the MySQL Workbench window with the title bar "MySQL Workbench". The "Object Browser" panel on the left lists the newly created schema "base\_medecin" along with other schemas like "dbdrvmedecins" and "test". The "SQL Editor (localhost)" panel on the right contains the number "1".

new\_schema

Name: base\_medecin

Collation: Server Default

The name of the schema. It is recommended to use only alpha-numeric characters. Spaces should be avoided and be replaced by \_

Specifies which charset/collations the schema's tables will use if they do not have an explicit setting. Common choices are Latin1 or UTF8.

MySQL Workbench

File Edit View Query Database Plugins Scripting

Home Output MySQL Model SQL Editor (localhost)

Object Browser

Default:

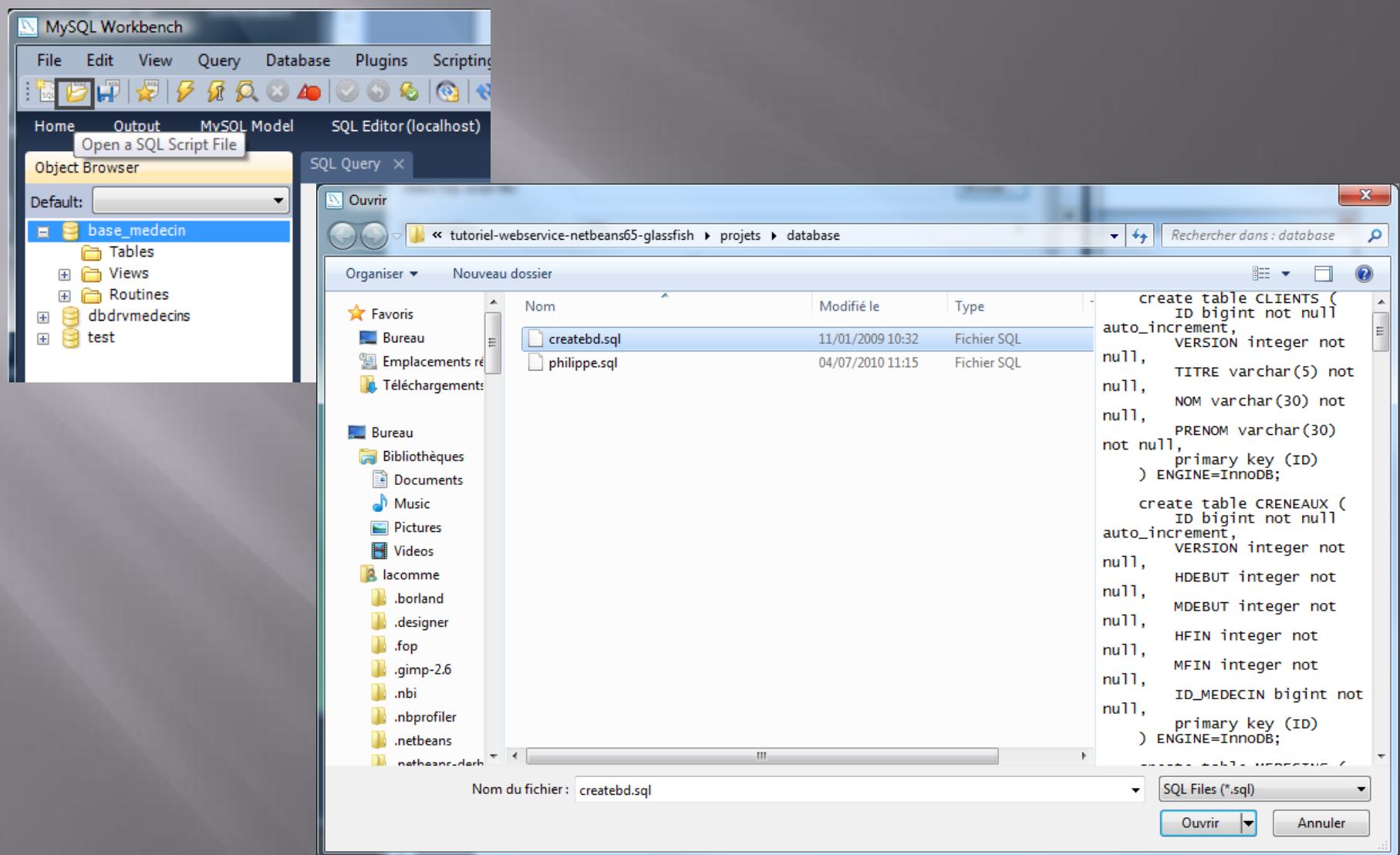
- base\_medecin
  - Tables
  - Views
  - Routines
- dbdrvmedecins
- test

1

Schema

DBMS feedback messages will go here upon applying changes.

# Lancer le script



# Créer un pool de connexion

- ❑ Qu'est-ce qu'un pool de connexion ?
- ❑ Quels sont les avantages (et inconvénients) ?  
Quelles sont les alternatives ?

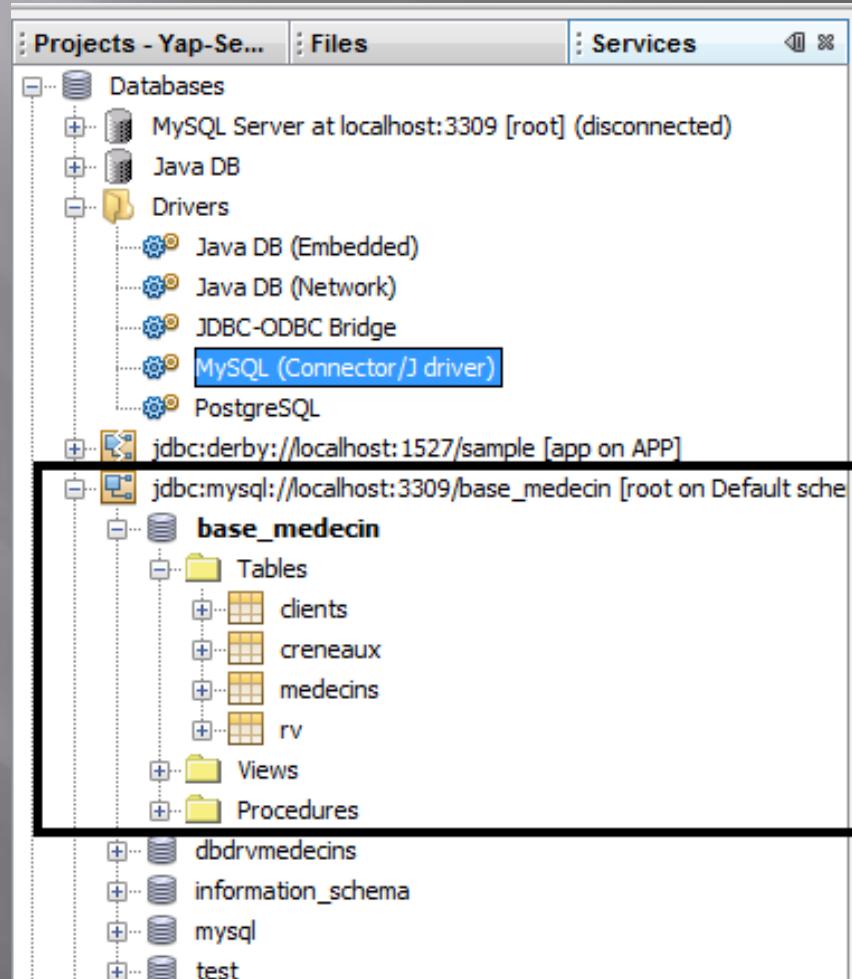
# Créer une nouvelle connexion à une base MySQL

The screenshot shows the GlassFish Admin Console interface. On the left, the navigation tree includes sections for Projects, Files, and Services. Under Services, the Databases section is expanded, showing Java DB, Drivers (with MySQL (Connector/J driver) selected), JDBC-ODBC Bridge, and PostgreSQL. Below this is a connection pool entry for jdbc:derby://localhost:1527/. The Servers section lists GlassFish v3 and GlassFish v3 (1). A Personal GlassFish v3 Domain is selected, showing Applications (EnterpriseApplication2, WebApplication2, WebApplication1, app1, hello) and Resources (JDBC, Connectors, JavaMail Sessions). The JDBC Resources section contains Connection Pools (TimerPool, DerbyPool, SamplePool). The Connectors section contains Connector Resources, Connector Connection Pools, Admin Object Resources, and JavaMail Sessions. At the bottom, Hudson Builders, Kenai Instances, and kenai.com are listed, along with Issue Trackers.

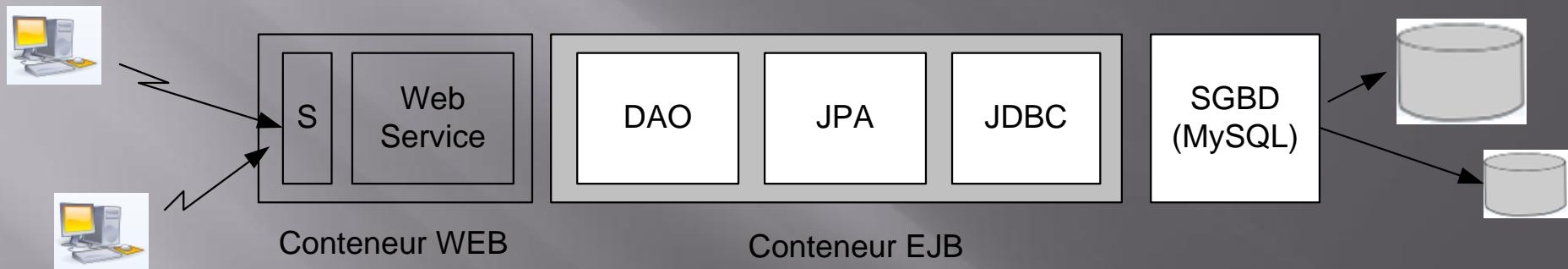
A context menu is open over the MySQL (Connector/J driver) entry in the Drivers list, with options: Connect Using..., Delete, and Customize.

The main window is titled "New Database Connection" and displays the "Basic setting" tab. The "Data Input Mode" is set to "Field Entry". The "Driver Name" is "MySQL (Connector/J driver)". The "Host" is "localhost", "Port" is "3309", and the "Database" is "base\_medecin". The "User Name" is "root" and the "Password" is masked. An optional "Display Name (Optional)" field is empty. A checkbox for "Remember password (see help for information on security risks)" is unchecked. An "Additional Props:" field is also empty. At the bottom right are "OK", "Cancel", and "Help" buttons.

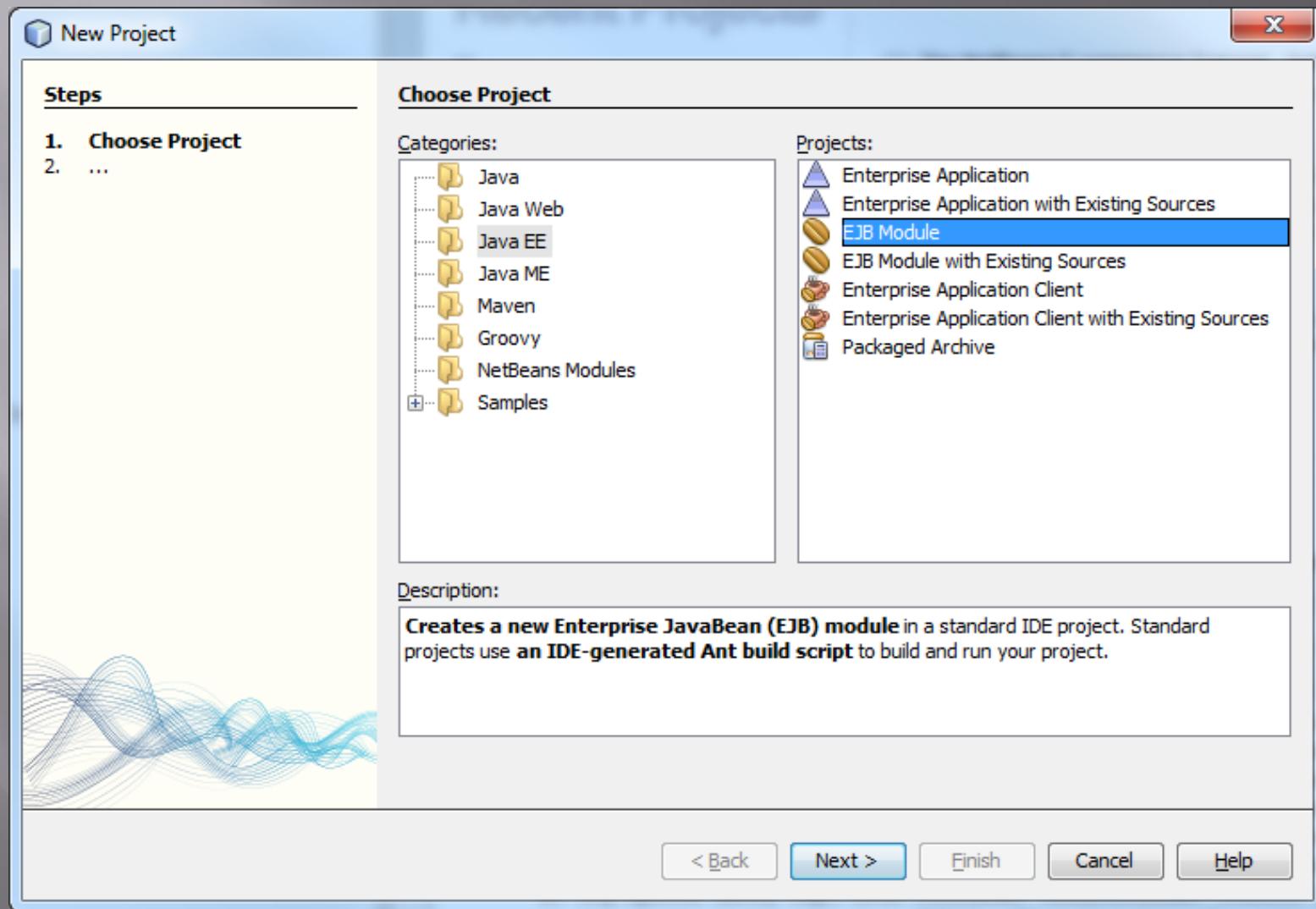
# La nouvelle connexion à base\_medecin



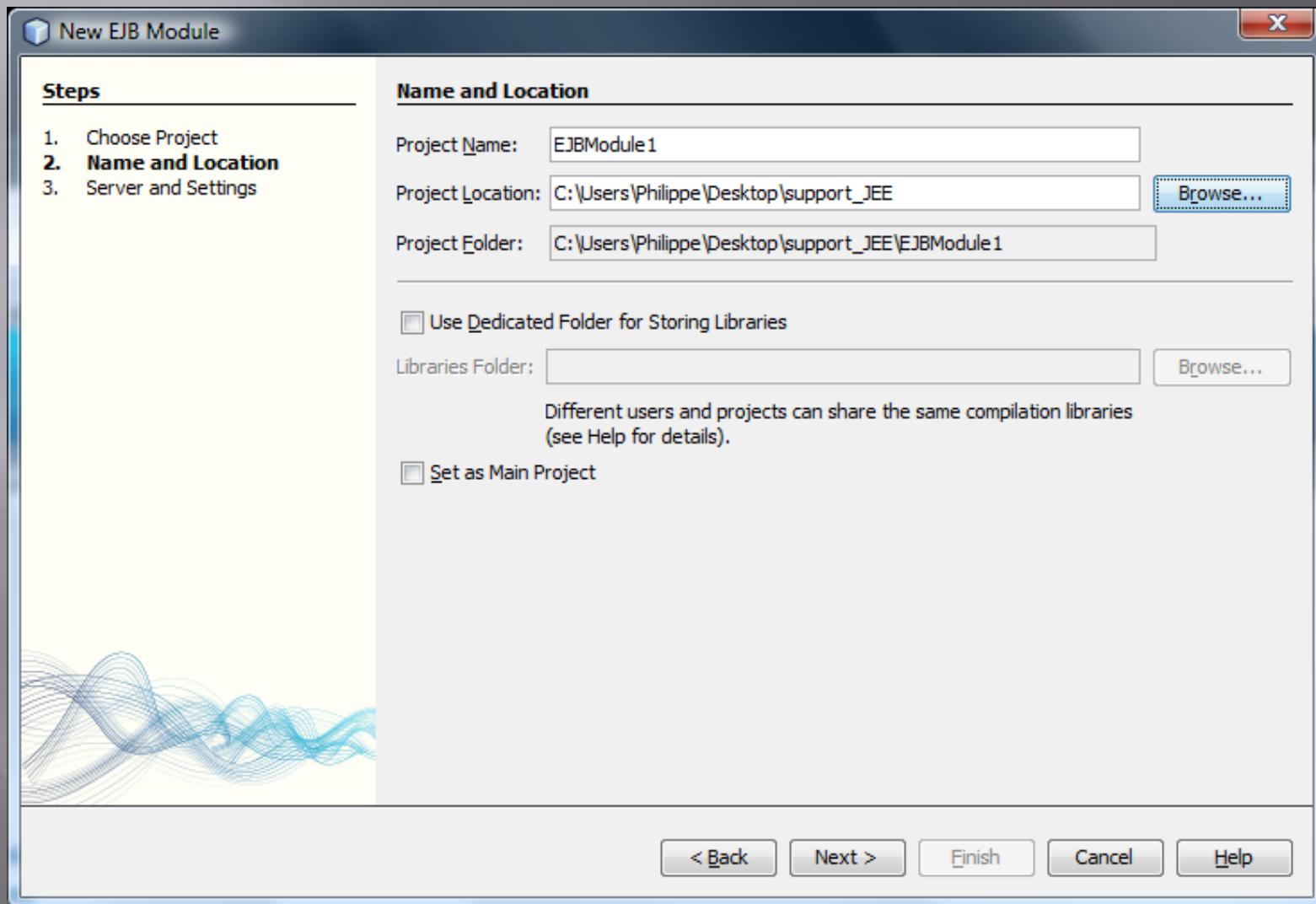
# Conteneur EJB



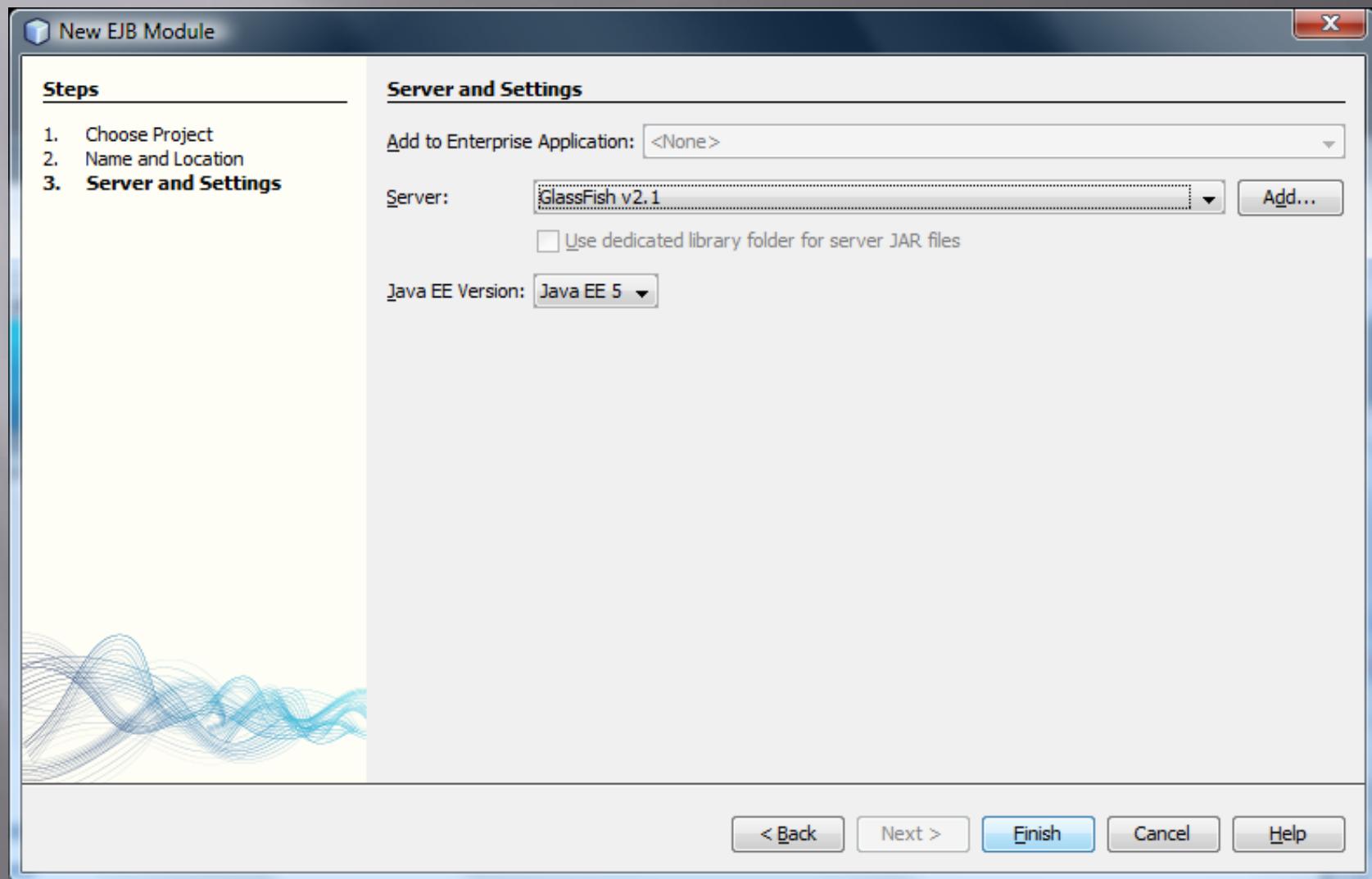
# Création d'un EJB module



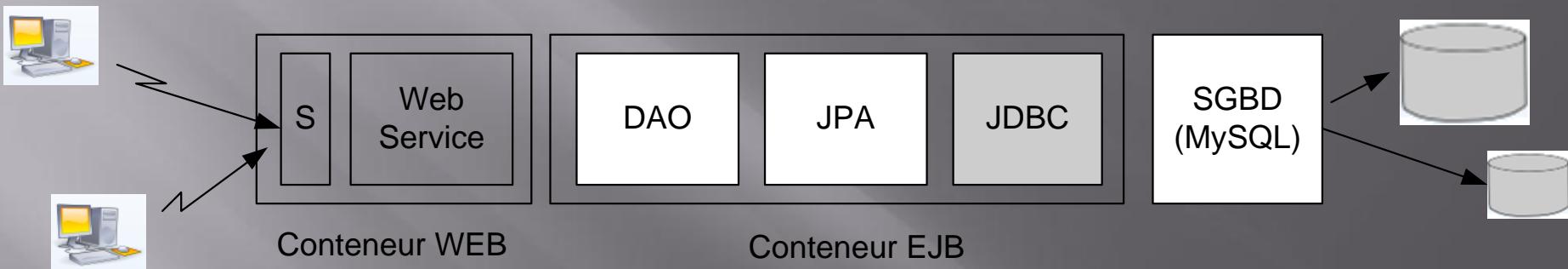
# Création d'un EJB module



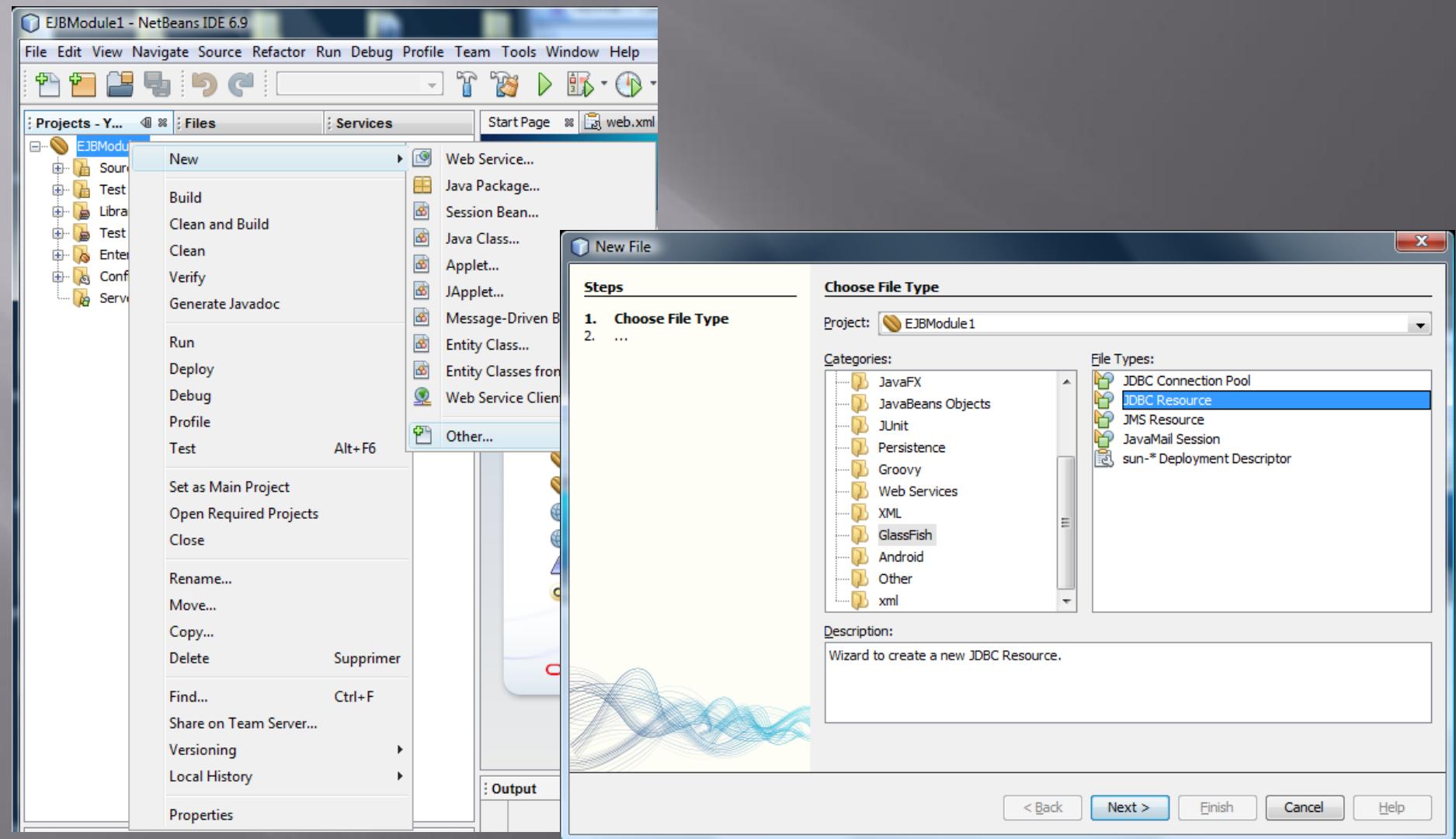
# Création d'un EJB module



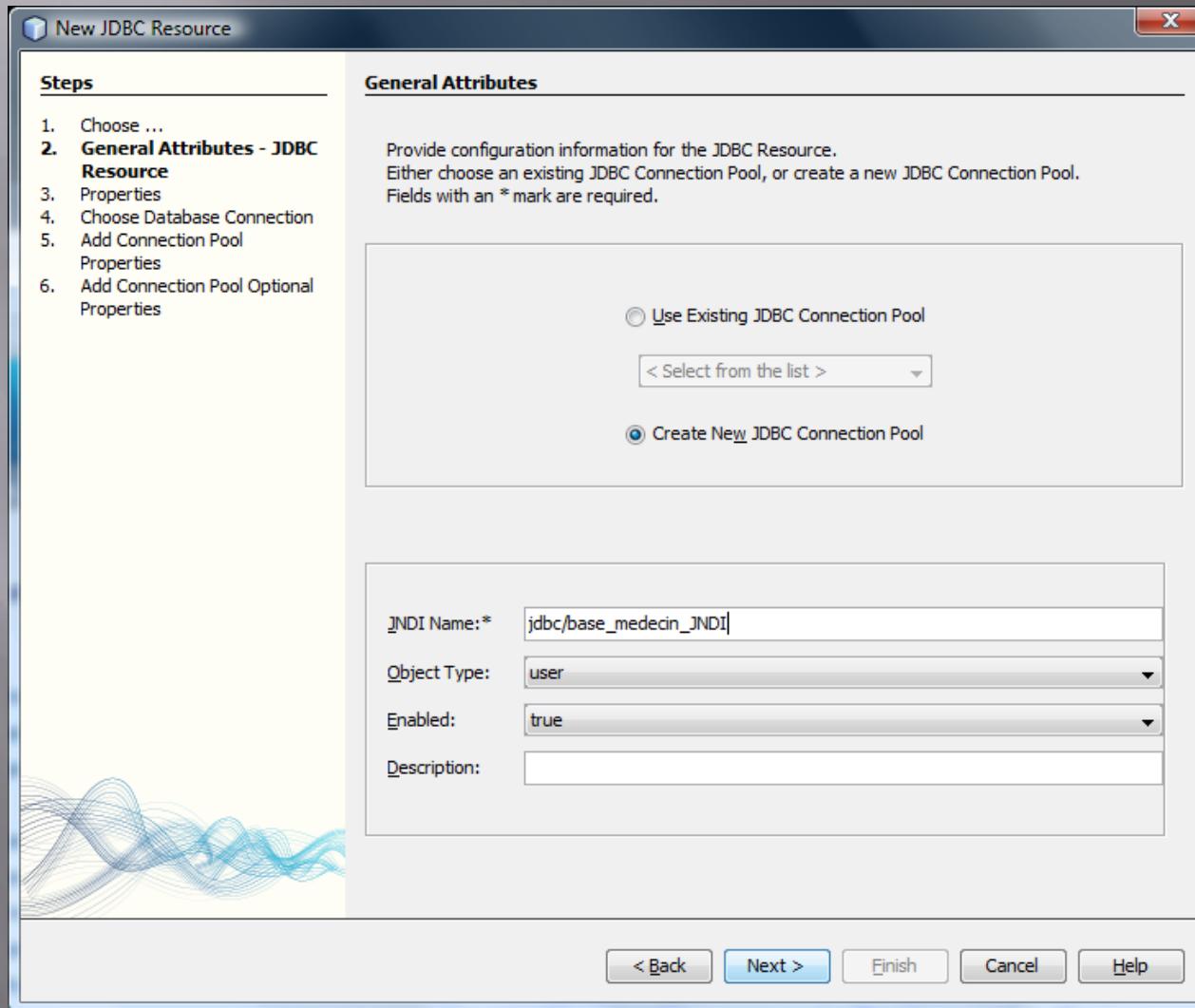
# Création d'une ressource JDBC (pool de connexion)



# Création d'une ressource JDBC (pool de connexion)



# Création d'une ressource JDBC (pool de connexion)



# Création d'une ressource JDBC (pool de connexion)

New JDBC Resource

**Steps**

1. Choose ...
2. General Attributes - JDBC Resource
3. Properties
- 4. Choose Database Connection**
5. Add Connection Pool Properties
6. Add Connection Pool Optional Properties

**Choose Database Connection**

Provide configuration information for the JDBC Connection Pool.  
Either choose an existing database connection to extract information, or enter the configuration information.  
Fields with an \* mark are required.

JDBC Connection Pool Name: \*

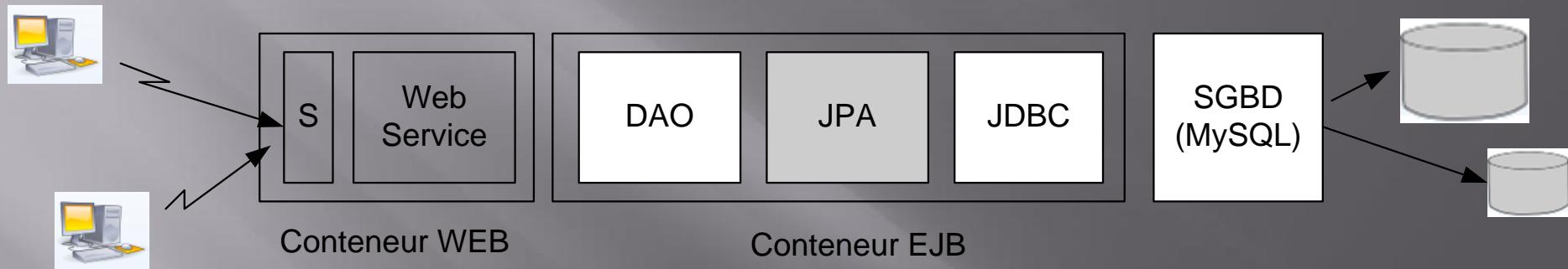
Extract from Existing Connection:

New Configuration using Database:

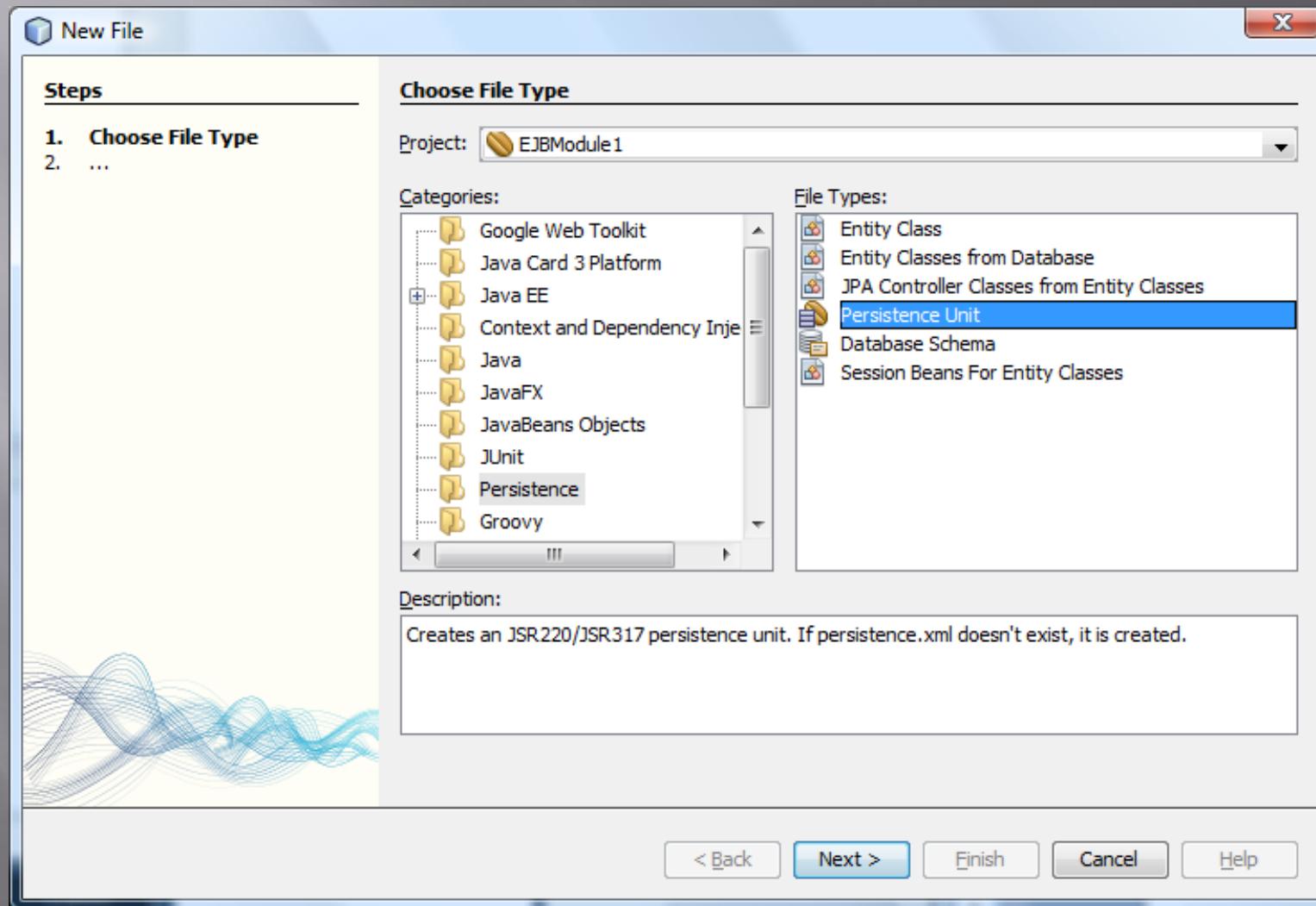
XA (Global Transaction)

< Back  Finish Cancel Help

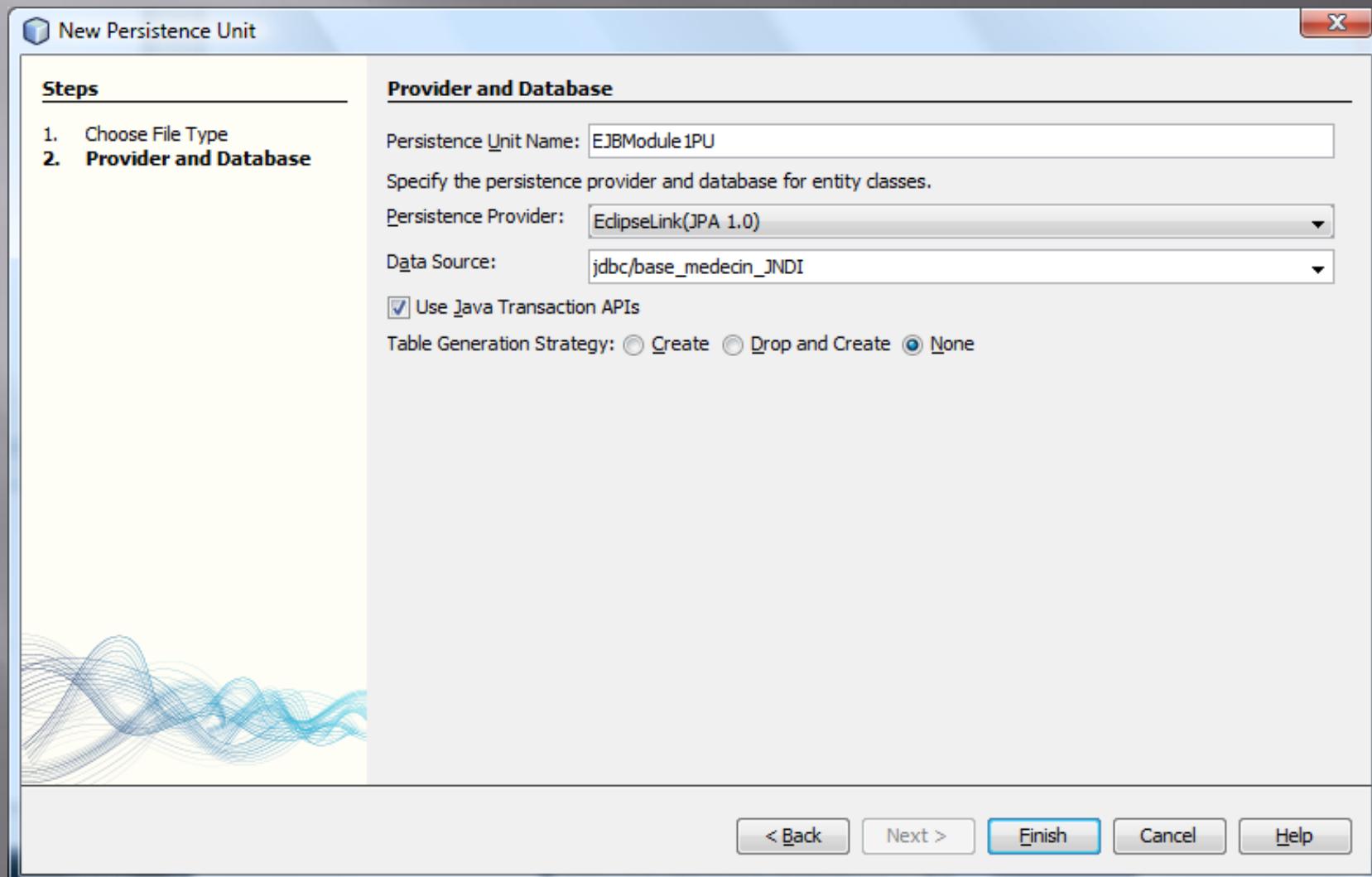
# Création d'une JPA (unité de persistance)



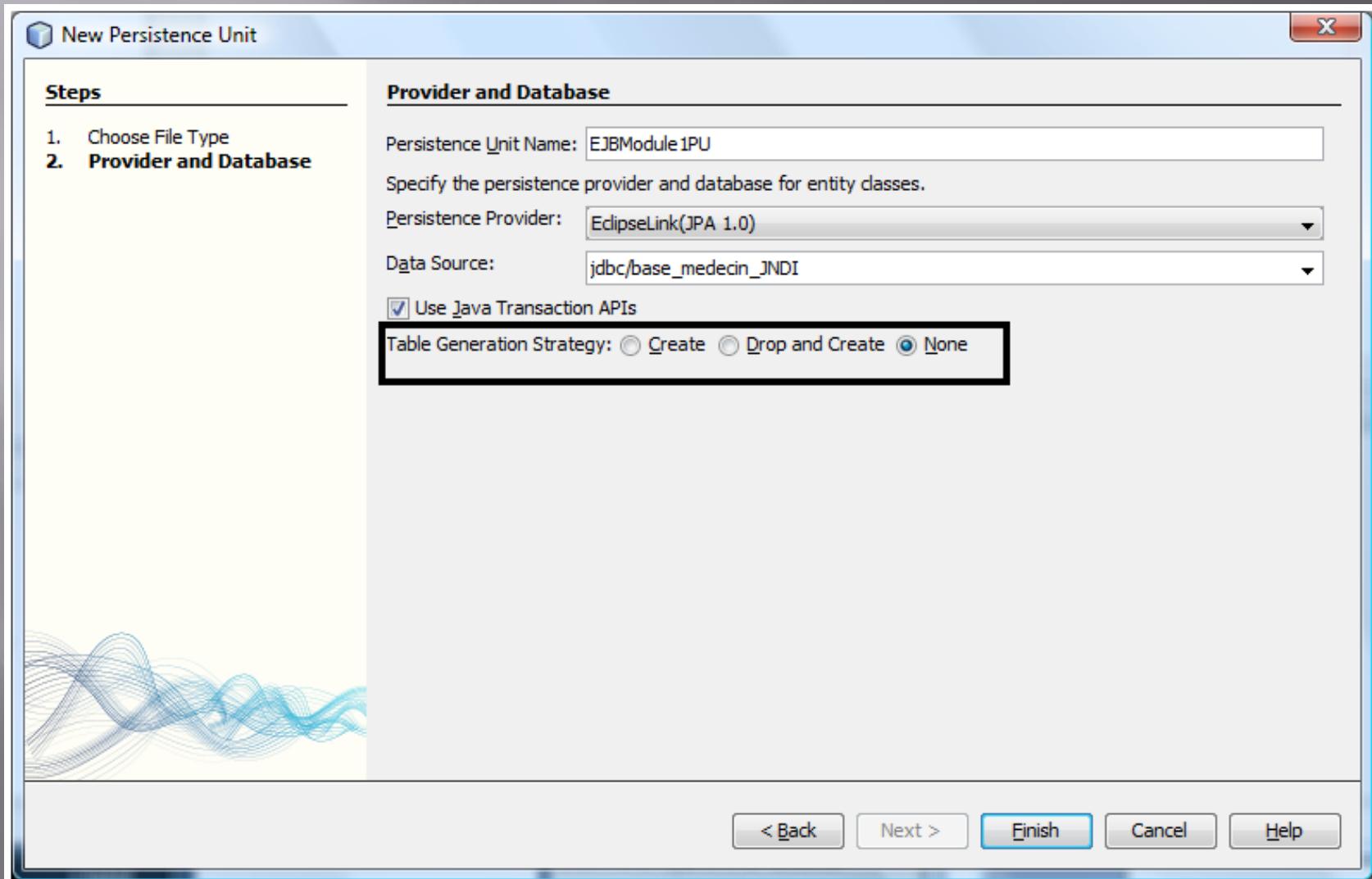
# Création d'une JPA (unité de persistance)



# Création d'une JPA (unité de persistance)



# Création d'une JPA (unité de persistance)



# Création des entités JPA

New File

**Steps**

1. Choose File Type  
2. ...

**Choose File Type**

Project: EJBModule1

**Categories:**

- Google Web Toolkit
- Java Card 3 Platform
- Java EE
- Context and Dependency Inje
- Java
- JavaFX
- JavaBeans Objects
- JUnit
- Persistence
- Groovy

**File Types:**

- Entity Class
- Entity Classes from Database
- JPA Controller Classes from Entity Classes
- Persistence Unit
- Database Schema
- Session Beans For Entity Classes

**Description:**

Creates Java Persistence API entity classes based on an existing relational database. Entity classes are used to represent objects whose lifespan is longer than a typical program execution. This template creates an entity class for each selected table, complete with named query annotations, fields representing columns, and relationships representing foreign keys.

< Back    Next >    Finish    Cancel    Help

# Création des entités JPA

New Entity Classes from Database

X

**Steps**

1. Choose File Type
2. **Database Tables**
3. Entity Classes
4. Mapping Options

**Database Tables**

Data Source: jdbc/base\_medecin\_JNDI

Database Schema: <no database schemas in the project>

Available Tables:

Selected Tables:

clients  
creneaux  
medecins  
rv

Add >  
< Remove  
Add All >>  
<< Remove All

Include Related Tables

< Back Next > Finish Cancel Help

# Création des entités JPA

New Entity Classes from Database X

**Steps**

1. Choose File Type
2. Database Tables
- 3. Entity Classes**
4. Mapping Options

**Entity Classes**

Specify the names and the location of the entity classes.

Class Names:	Database Table	Class Name	Generation Type
	clients	Clients	New
	creneaux	Creneaux	New
	medecins	Medecins	New
	rv	Rv	New

Project: EJBModule1

Location: Source Packages

Package:

Generate Named Query Annotations for Persistent Fields

 Provide a package name.

[< Back](#) [Next >](#) [Finish](#) [Cancel](#) [Help](#)

# Création des entités JPA

New Entity Classes from Database

**Steps**

1. Choose File Type
2. Database Tables
- 3. Entity Classes**
4. Mapping Options

**Entity Classes**

Specify the names and the location of the entity classes.

Class Names:

Database Table	Class Name	Generation Type
clients	Clients	New
creneaux	Creneaux	New
medecins	Medecins	New
rv	Rv	New

Project: EJBModule1

Location: Source Packages

Package: jpa

Generate Named Query Annotations for Persistent Fields

< Back Next > Finish Cancel Help

# Création des entités JPA

New Entity Classes from Database

X

**Steps**

1. Choose File Type
2. Database Tables
3. Entity Classes
4. **Mapping Options**

**Mapping Options**

Specify the default mapping options.

Association Fetch: `default`

Collection Type: `java.util.List`

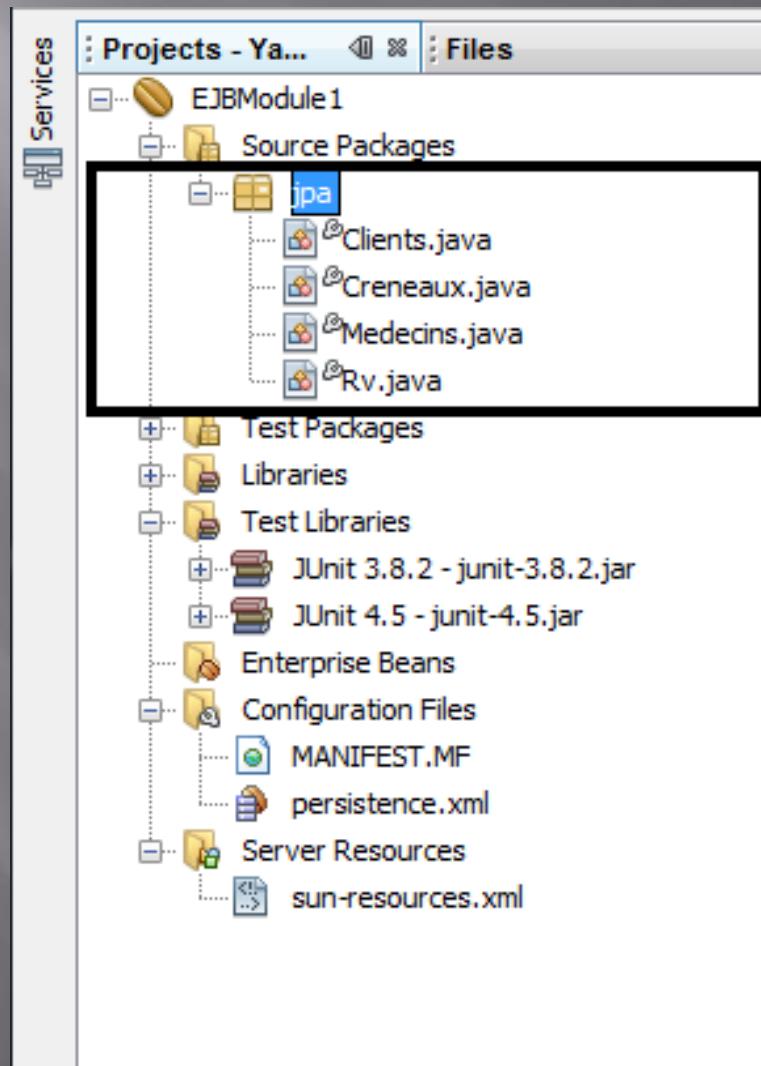
Fully Qualified Database Table Names

Attributes for Regenerating Tables

< Back Next > Finish Cancel Help



# Création des entités JPA : ajout d'un constructeur pour « RV »



# Création des entités JPA : ajout d'un constructeur pour « RV »

Clients.java

```
33     private String prenom,  
54     @OneToMany(cascade = CascadeType.ALL, mappedBy = "idClient")  
55     private transient List<Rv> rvList;  
56
```

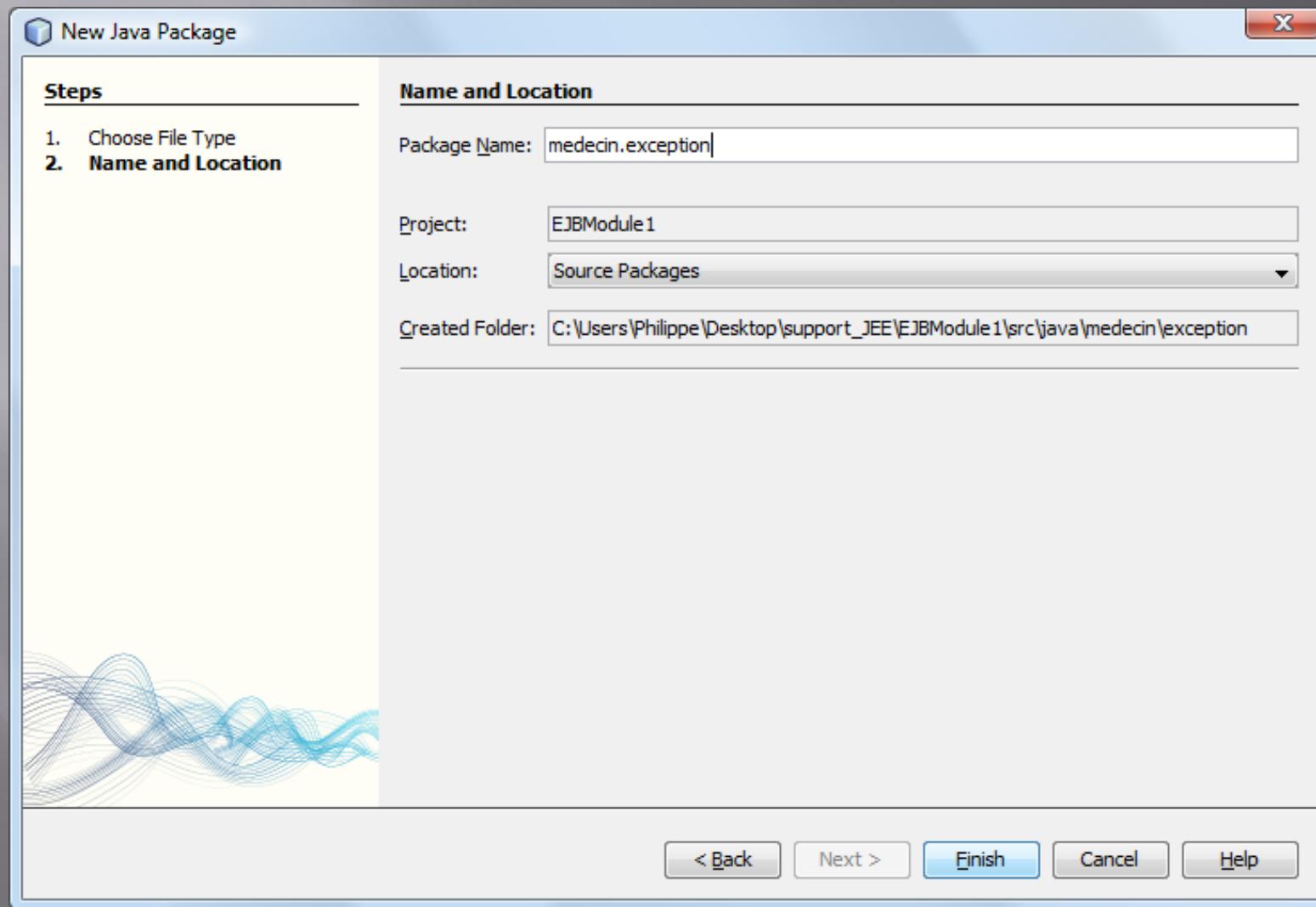
Medecins.java

```
54     @OneToMany(cascade = CascadeType.ALL, mappedBy = "idMedecin")  
55     private transient List<Creneaux> creneauxList;  
56
```

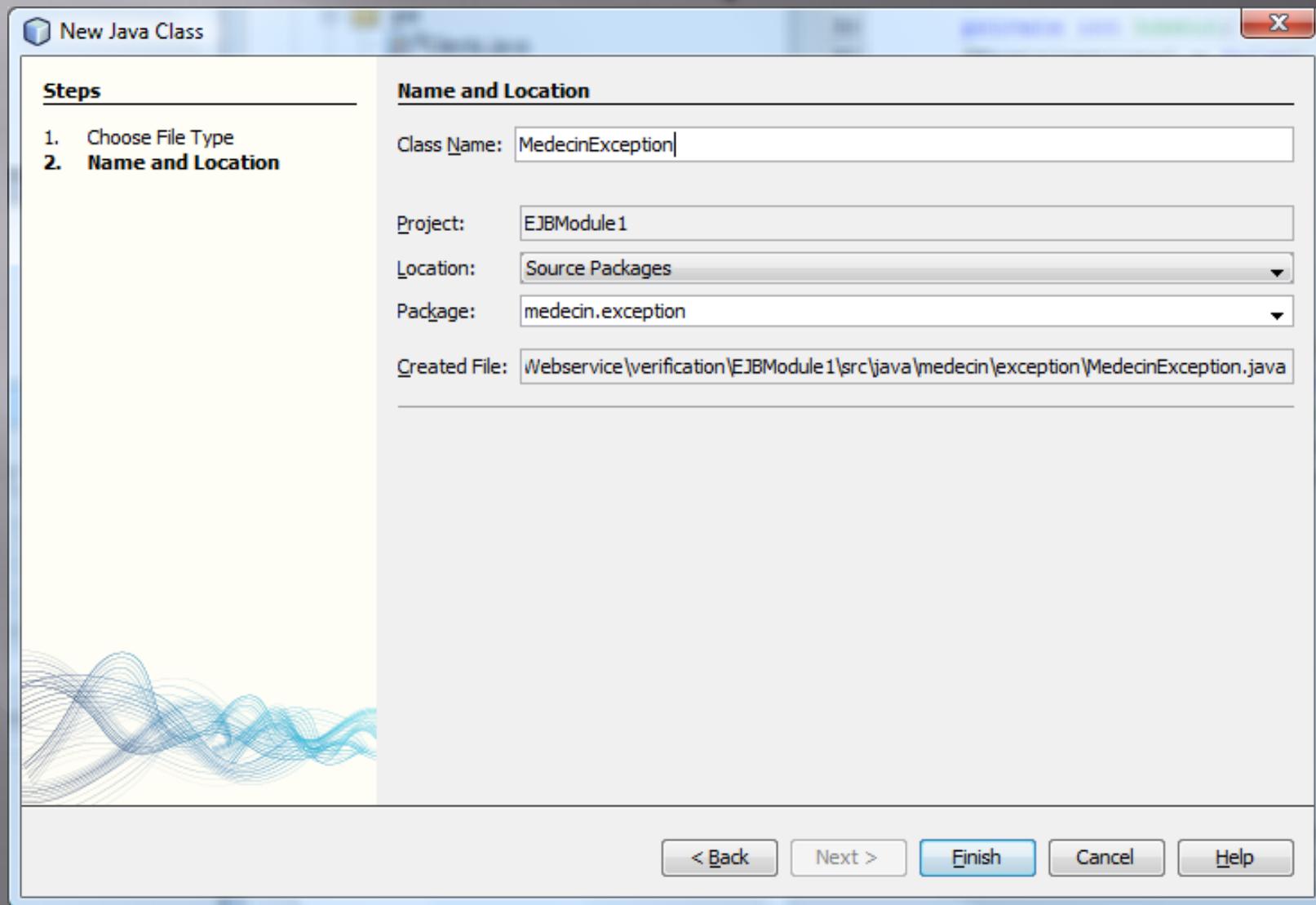
Creneaux.java

```
63     @OneToMany(cascade = CascadeType.ALL, mappedBy = "idCreneau")  
64     private transient List<Rv> rvList;  
65
```

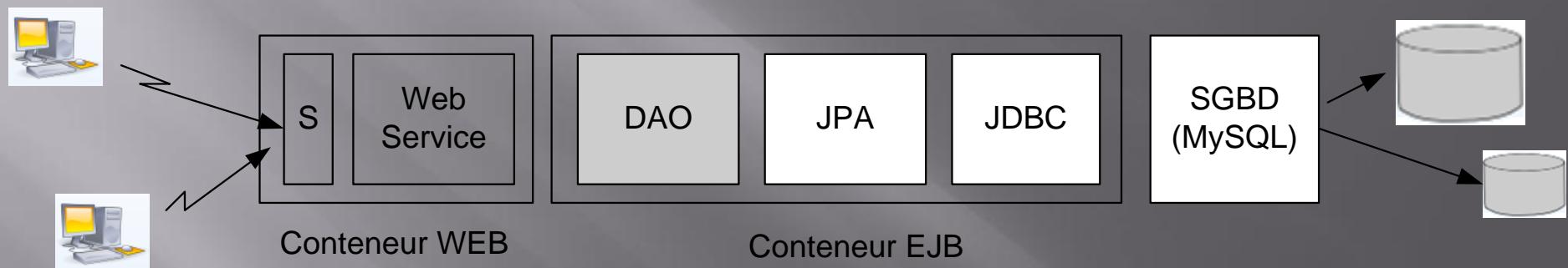
# Classe spécial pour le traitement des exceptions



# Classe spécial pour le traitement des exceptions



# Création de la couche DAO



# Création de la couche DAO

New File

**Steps**

1. Choose File Type
2. ...

**Choose File Type**

Project: EJBModule1

**Categories:**

- Google Web Toolkit
- Java Card 3 Platform
- Java EE
- Context and Dependency Inje
- Java
- JavaFX
- JavaBeans Objects
- JUnit
- Persistence
- Groovy

**File Types:**

- Session Bean
- Message-Driven Bean
- Service Locator
- Caching Service Locator
- Session Beans For Entity Classes
- Standard Deployment Descriptor

**Description:**

Creates Session EJB Facade based on Entity class. This template creates Session EJB for every Entity class with basic access methods.

< Back    Next >    Finish    Cancel    Help

# Création de la couche DAO

New File

**Steps**

1. Choose File Type  
2. ...

**Choose File Type**

Project: EJBModule1

**Categories:**

- Google Web Toolkit
- Java Card 3 Platform
- Java EE
- Context and Dependency Inje
- Java
- JavaFX
- JavaBeans Objects
- JUnit
- Persistence
- Groovy

**File Types:**

- Session Bean
- Message-Driven Bean
- Service Locator
- Caching Service Locator
- Session Beans For Entity Classes
- Standard Deployment Descriptor

**Description:**

Creates Session EJB Facade based on Entity class. This template creates Session EJB for every Entity class with basic access methods.

< Back    Next >    Finish    Cancel    Help

# Création de la couche DAO

New Session Beans For Entity Classes

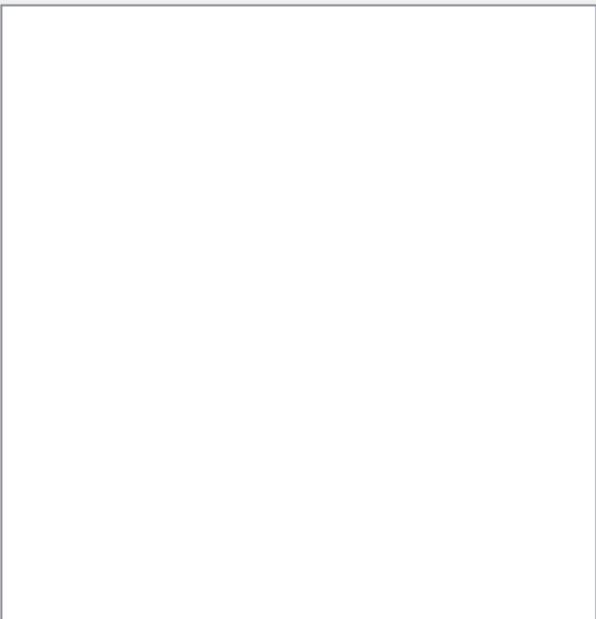
X

**Steps**

1. Choose File Type
- 2. Entity Classes**
3. Generated Session Beans

**Entity Classes**

Available Entity Classes:



Selected Entity Classes:

- jpa.Clients
- jpa.Creneaux
- jpa.Medecins
- jpa.RV

[Add >](#)

[< Remove](#)

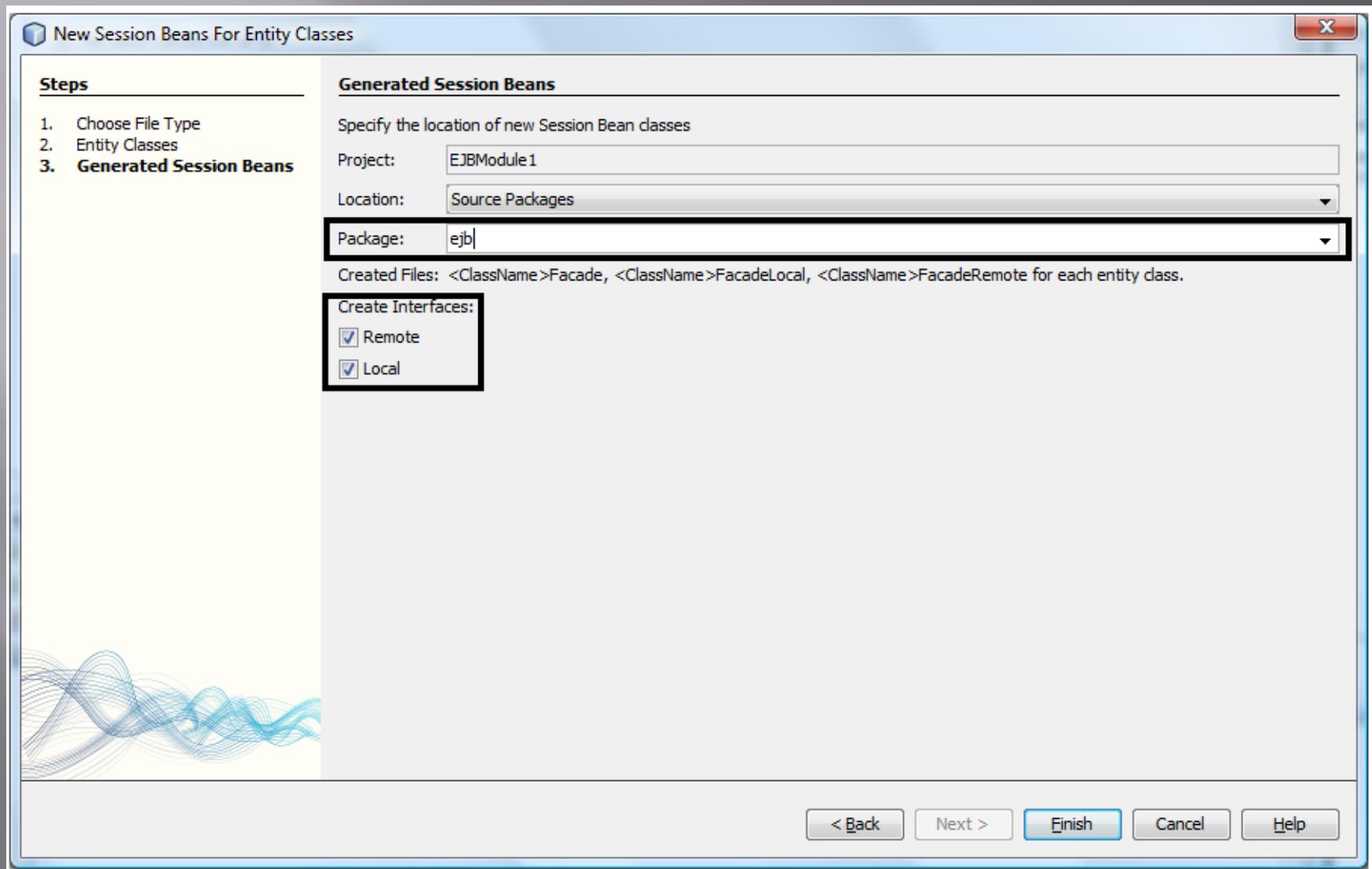
[Add All >>](#)

[<< RemoveAll](#)

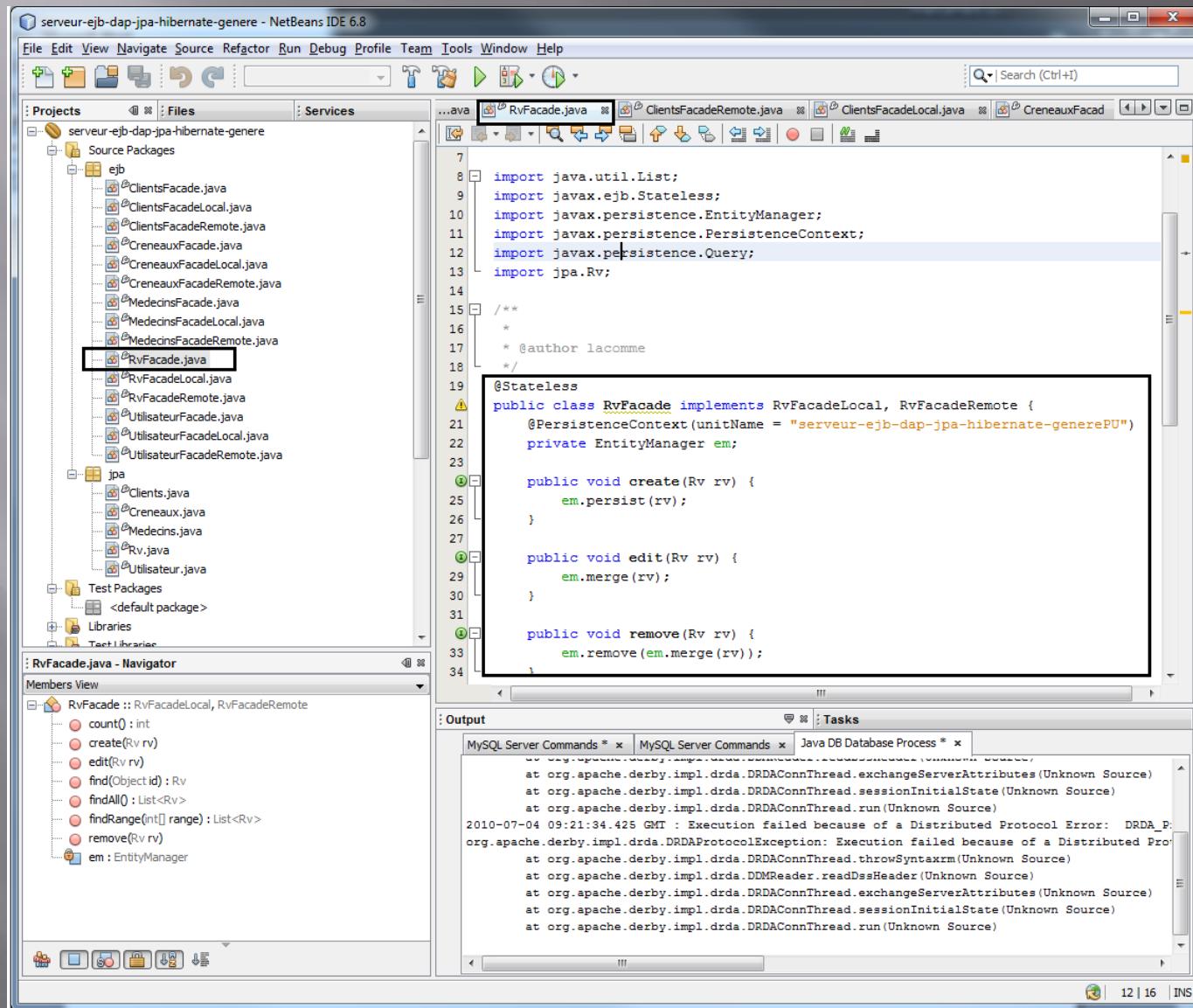
[Include Referenced Classes](#)

< Back [Next >](#) Finish Cancel Help

# Création de la couche DAO



# Création de la couche DAO



# Création de la couche DAO

New Java Package

Steps

1. Choose File Type  
2. Name and Location

Name and Location

Package Name: dao

Project: EJBModule1

Location: Source Packages

Created Folder: C:\Users\Philippe\Desktop\support\_JEE\EJBModule1\src\java\dao

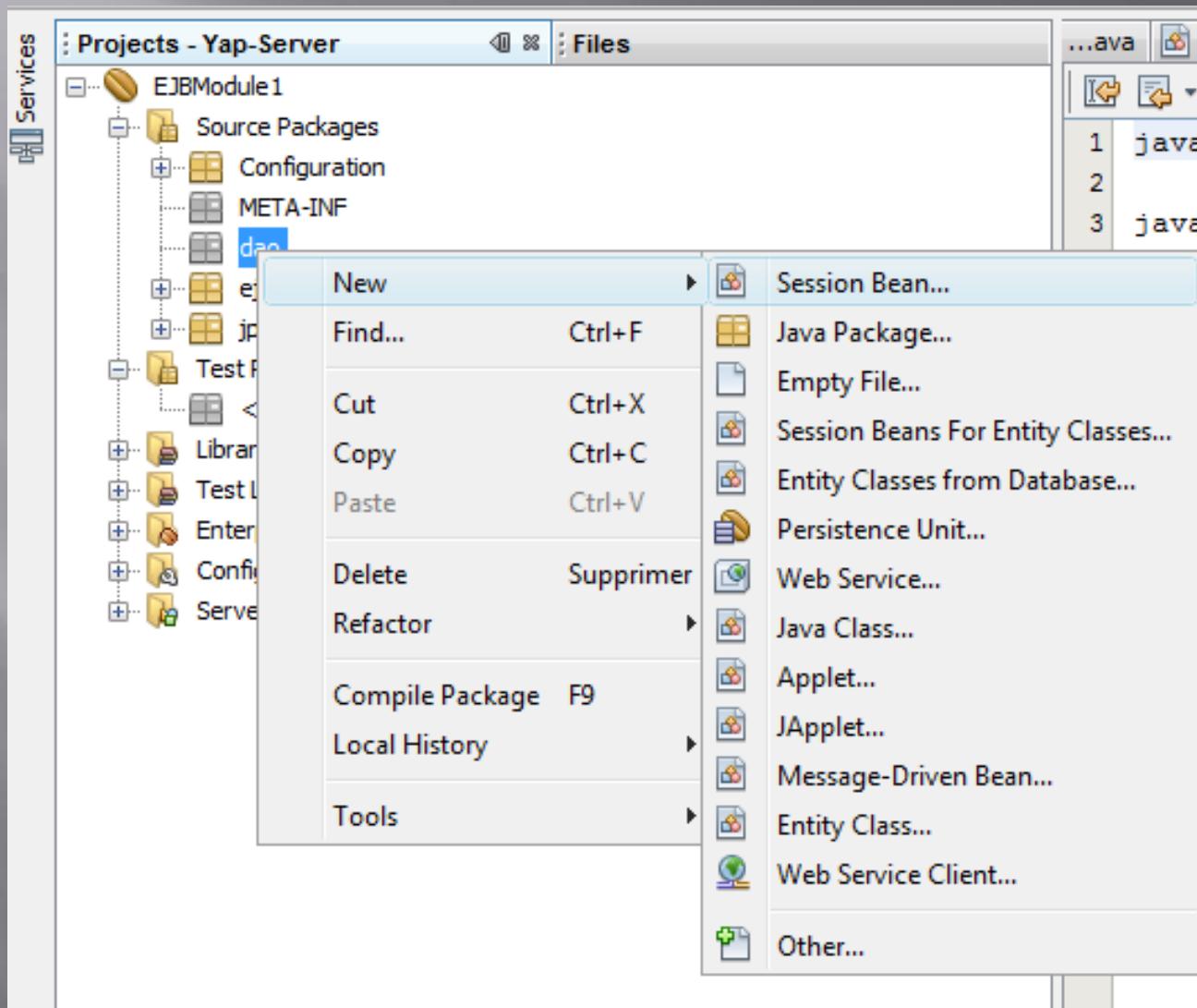
< Back Next > Finish Cancel

Projects - Yap-Se... Files

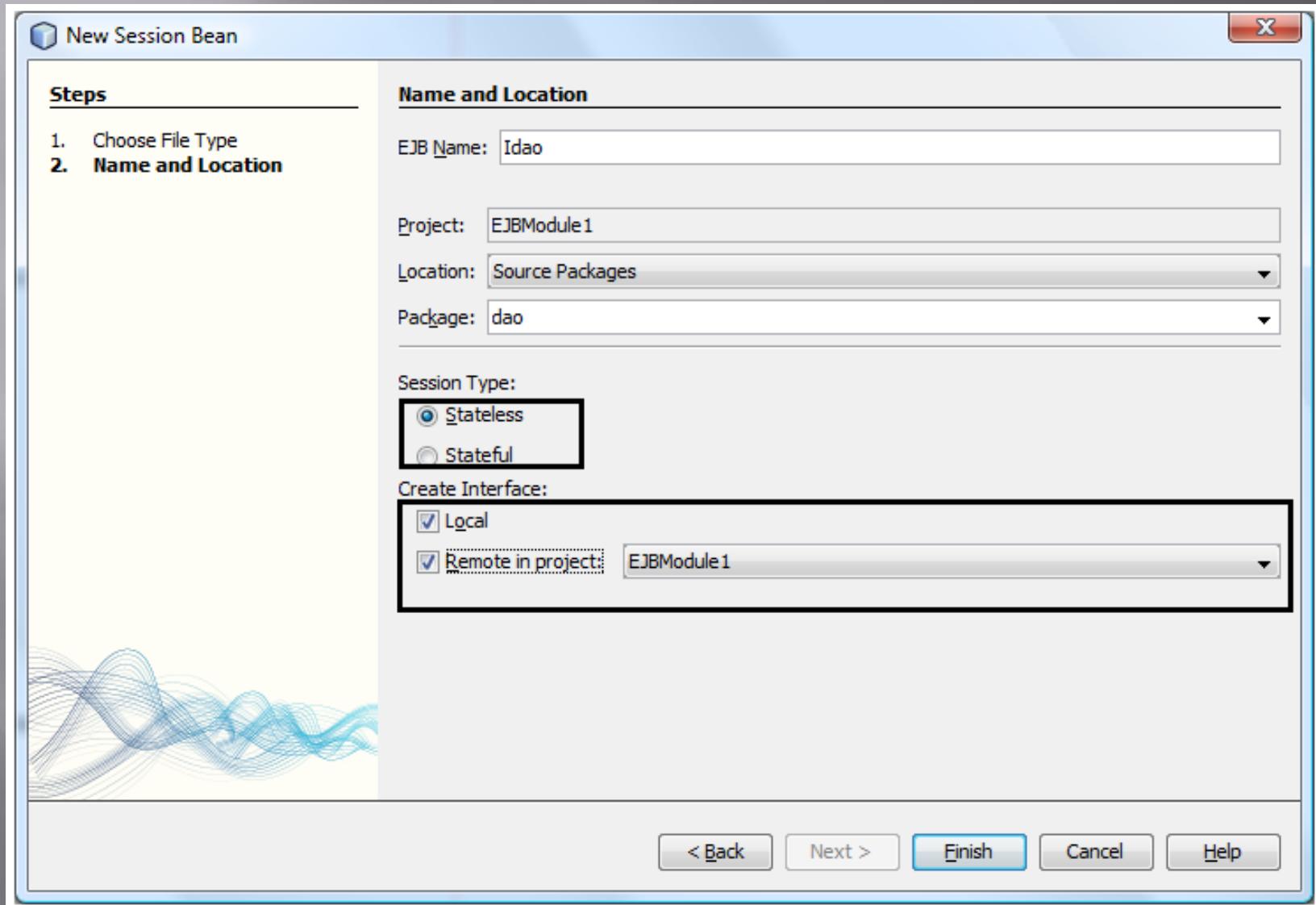
EJBModule1

- Source Packages
  - Configuration
  - dao
  - ejb
  - jpa
- Test Packages
- Libraries
- Test Libraries
- Enterprise Beans
  - ClientsFacade
  - CreneauxFacade
  - MedecinsFacade
  - RvFacade
- Configuration Files
  - MANIFEST.MF
  - persistence.xml
- Server Resources
  - sun-resources.xml

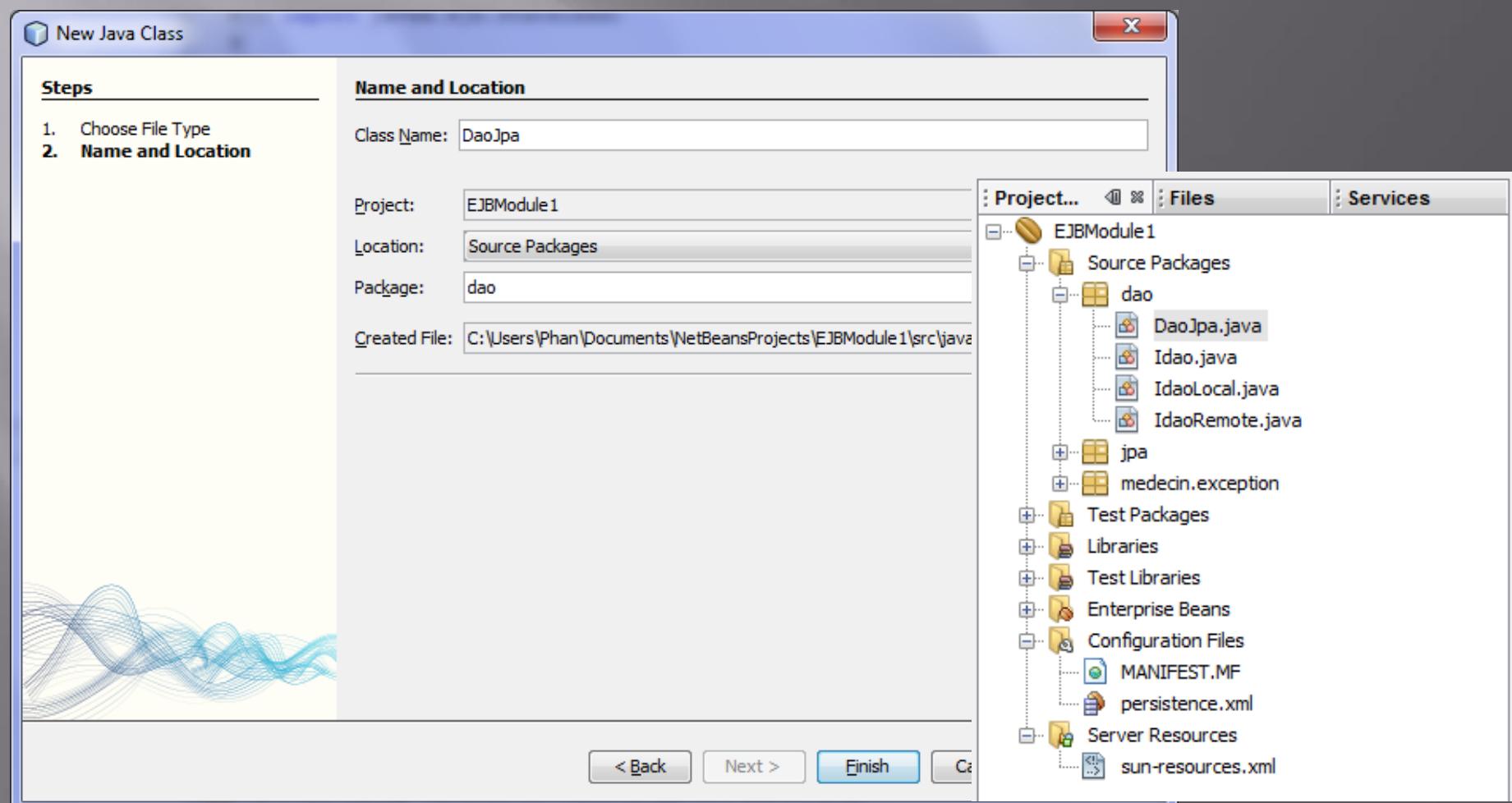
# Création de la couche DAO



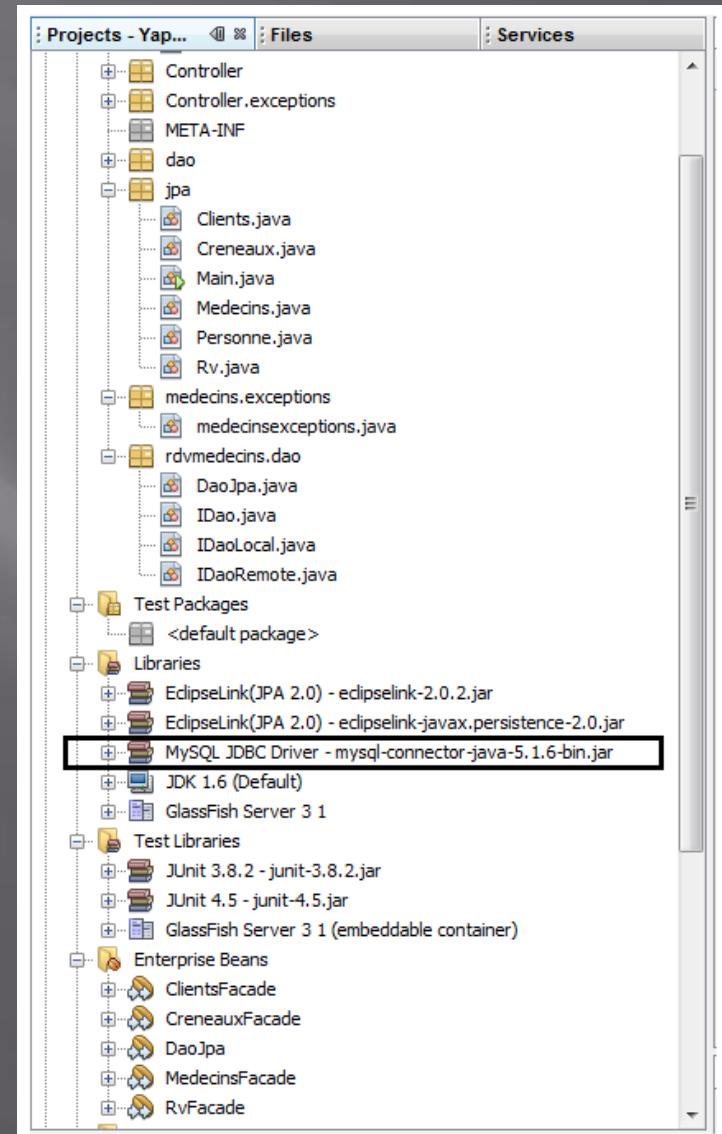
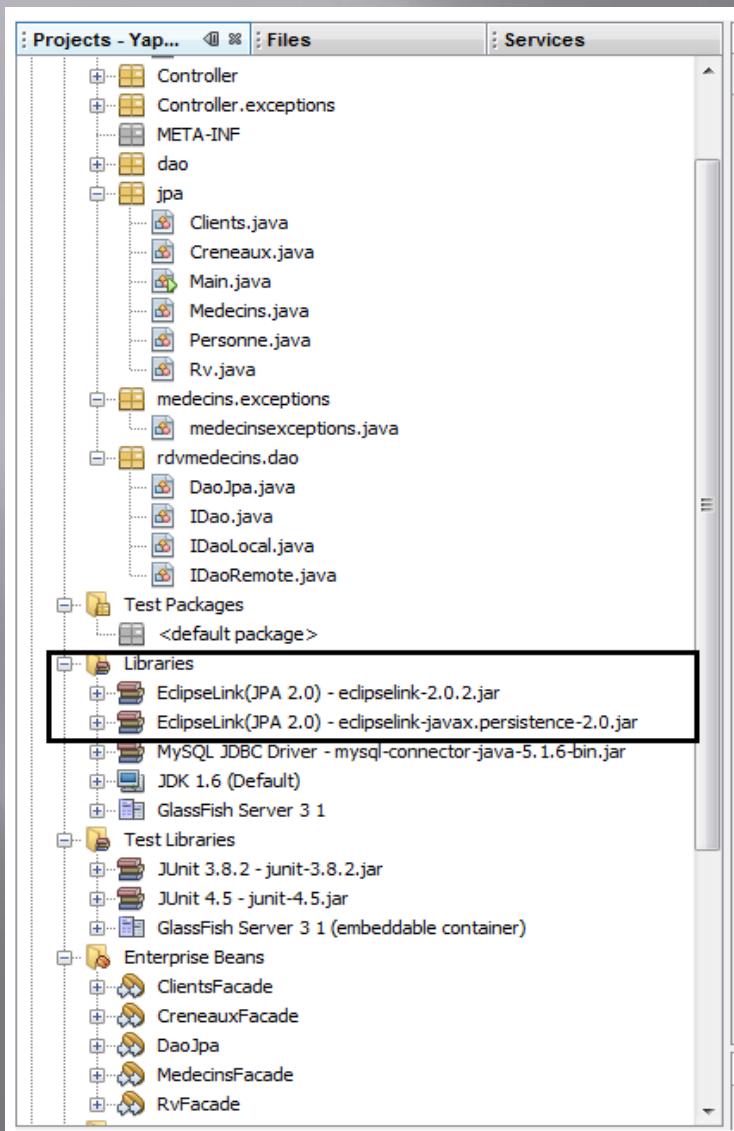
# Création de la couche DAO



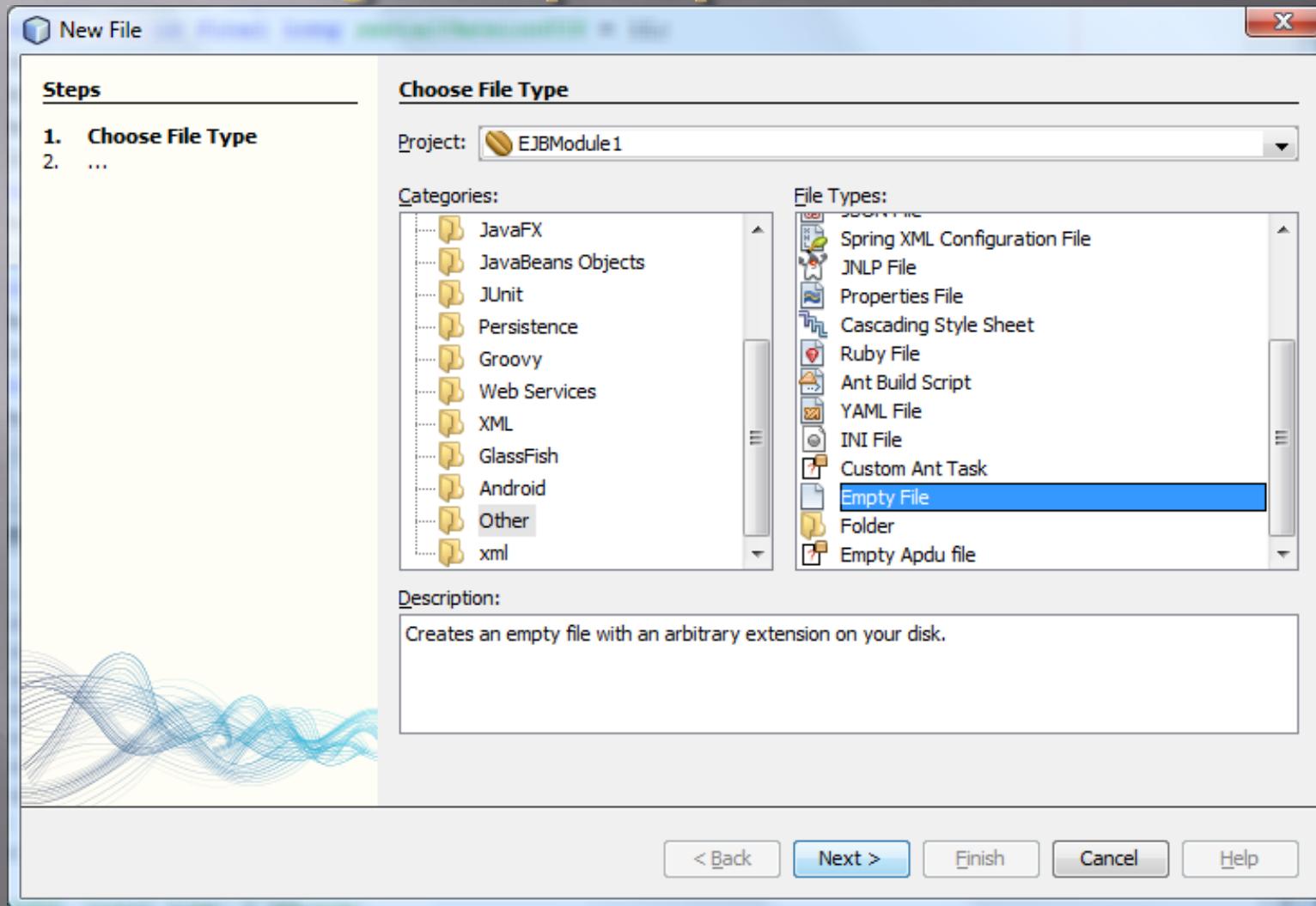
# Ajout d'une nouvelle classe pour implémenter IdaoLocal et IdaoRemote



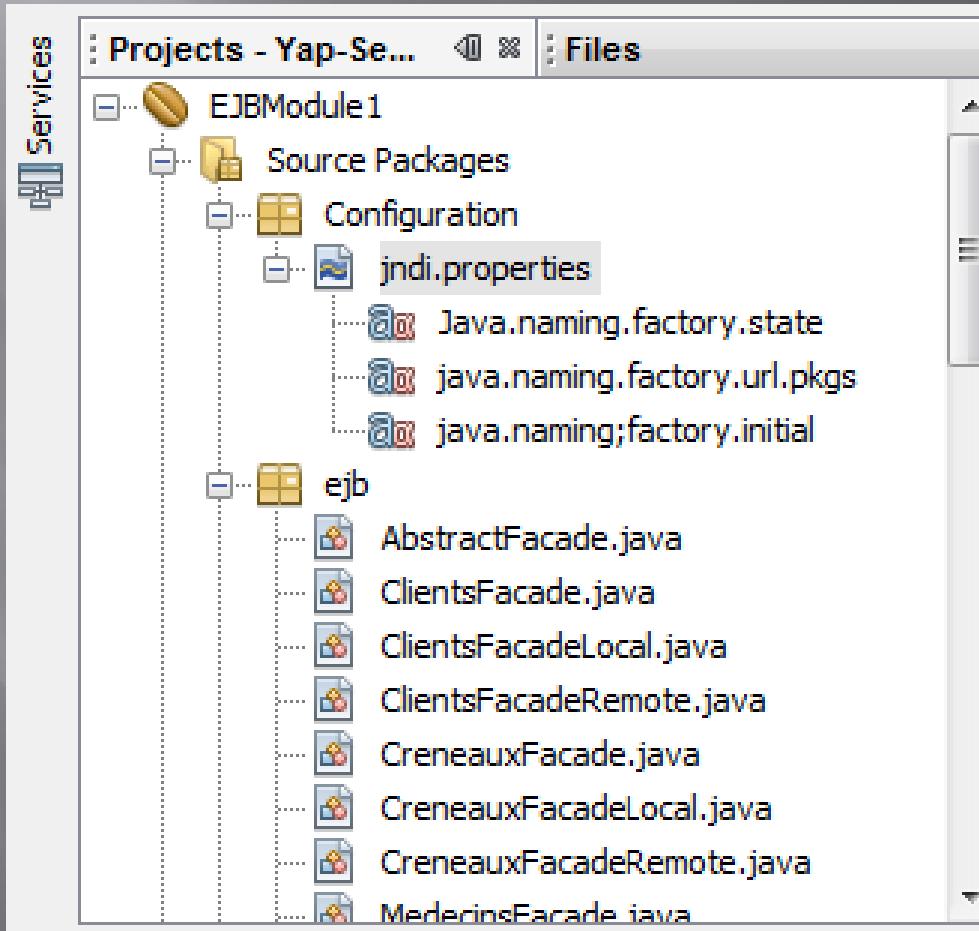
# Ajoute des librairies



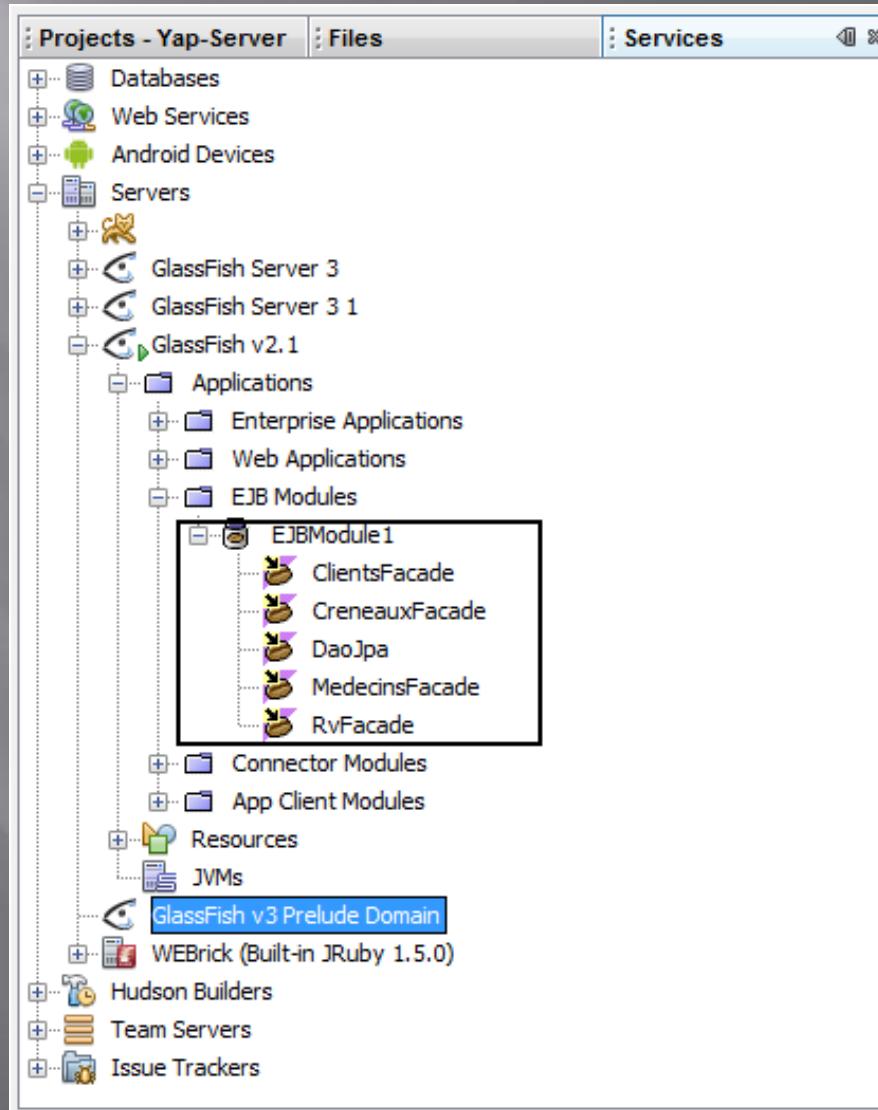
# Création du fichier de configuration JNDI nommé jndi.properties



# Création du fichier de configuration JNDI nommé jndi.properties

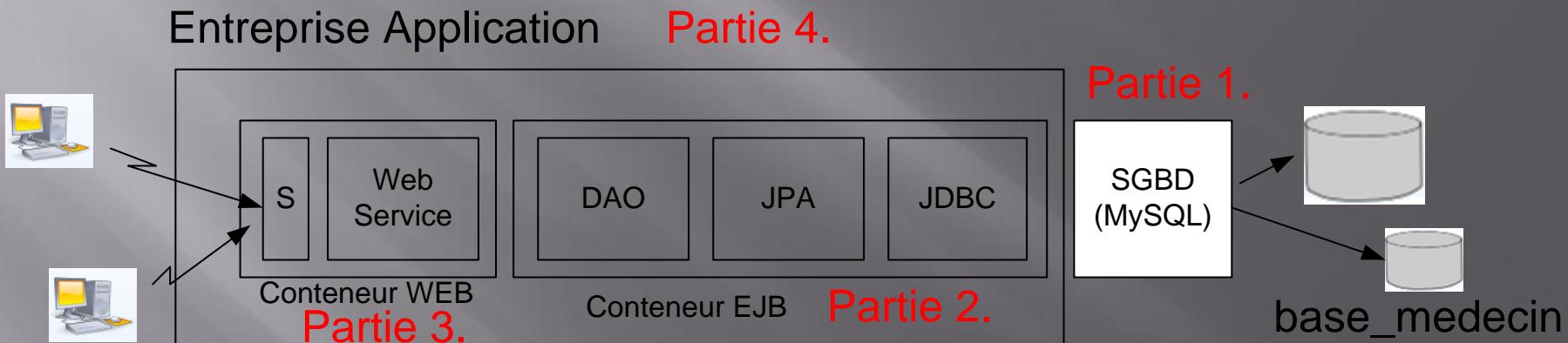


# Déploiement de l'EJB



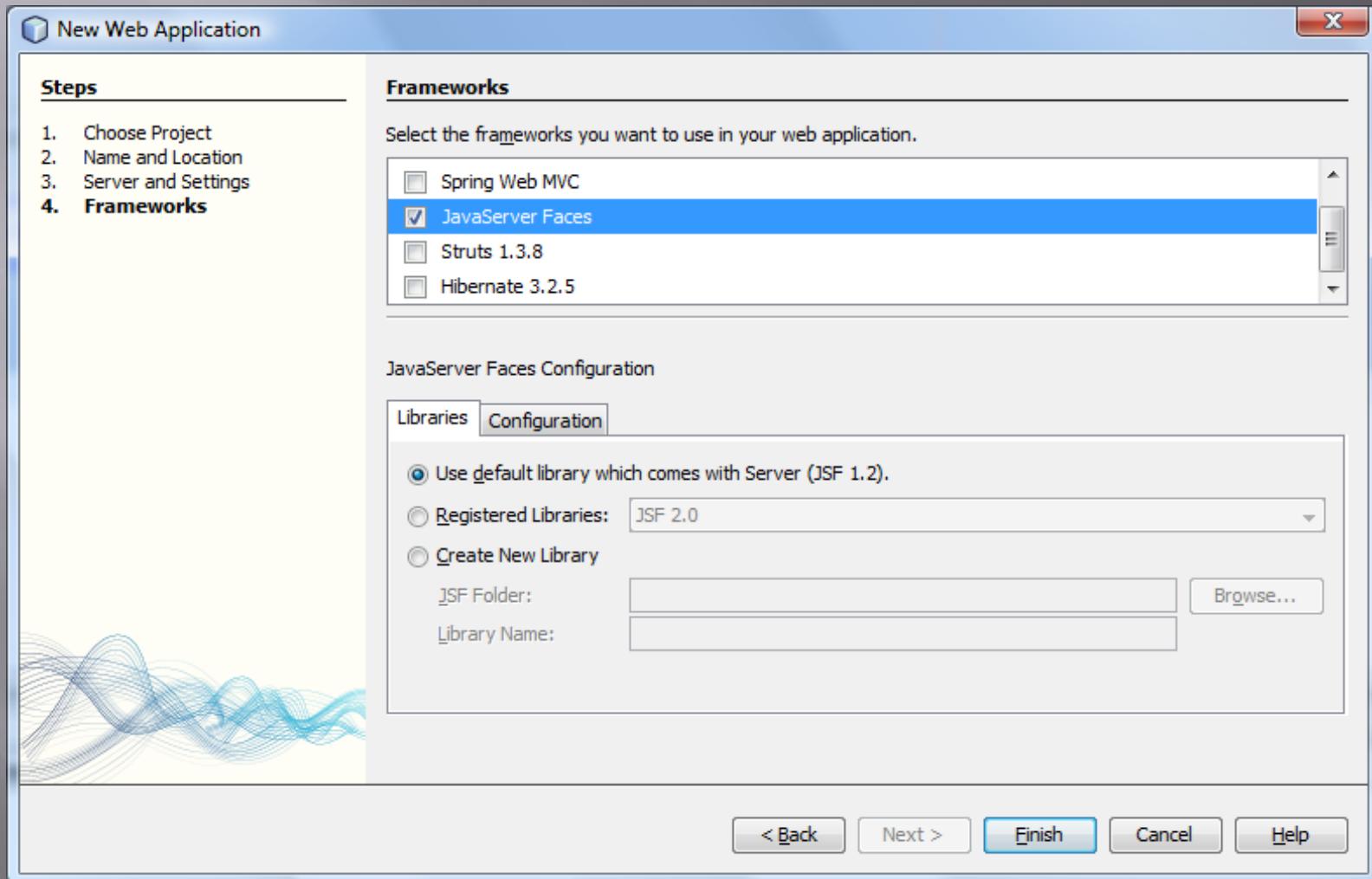
# Création d'un Web Service

- Qu'est-ce qu'un Framework ? Quels sont les exemples ?

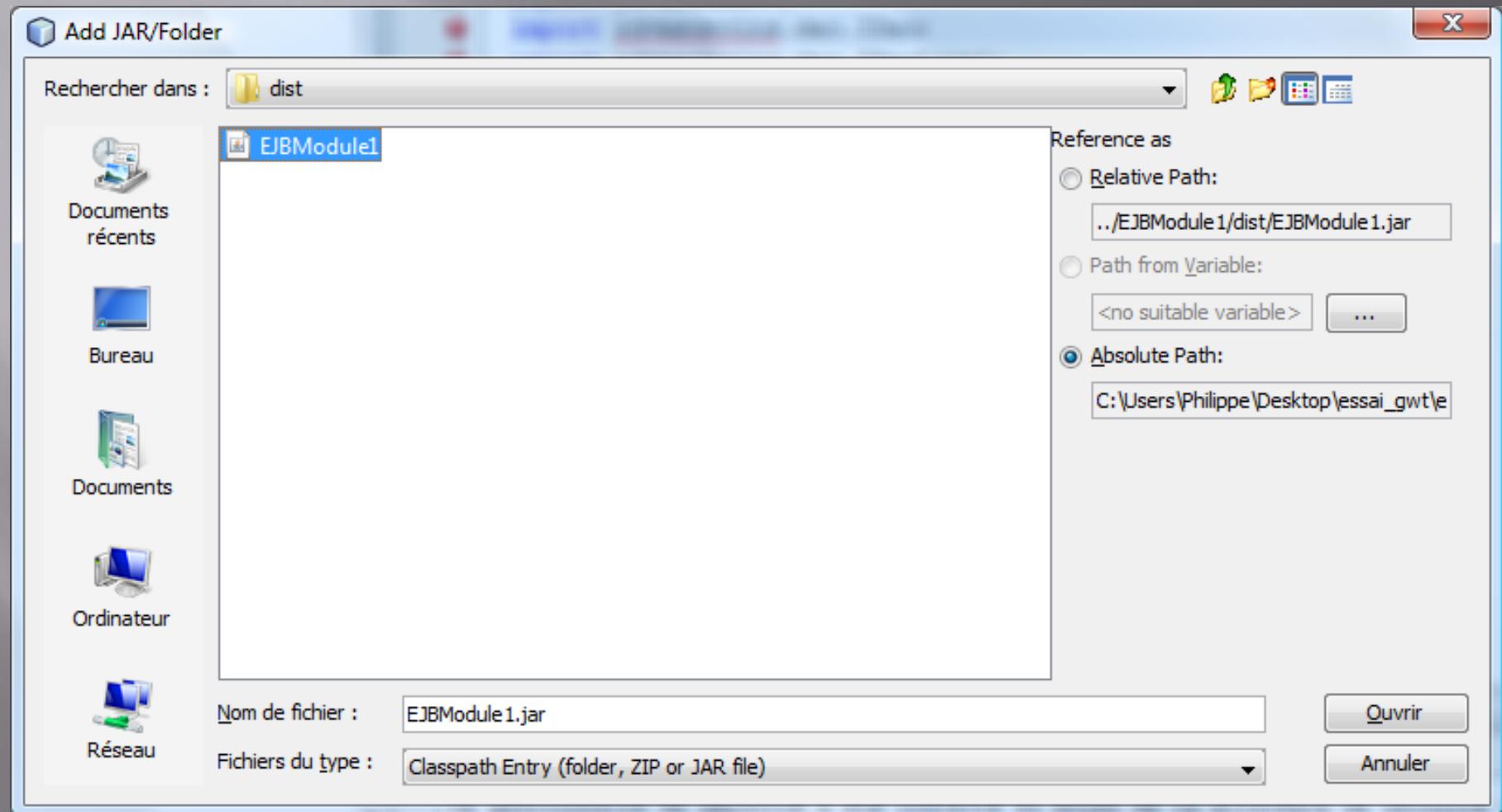


Partie 5.

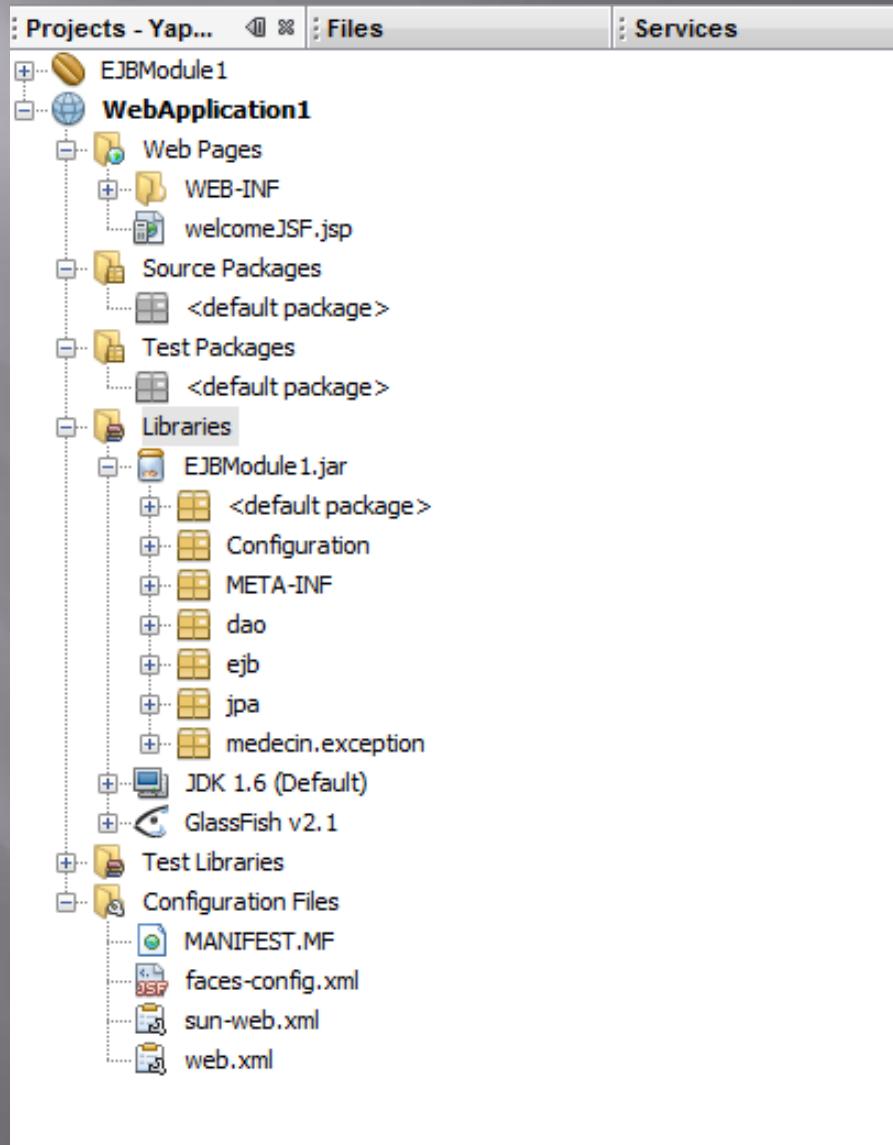
# Création d'un Web Service



# Ajouter l'EJB



# Le projet Web Application



# Création de WSDaoJpa

New Web Service

**Steps**

1. Choose File Type
2. Name and Location

**Name and Location**

Web Service Name:

Project:

Location:

Package:

Create Web Service from Scratch

Create Web Service from Existing Session Bean

Enterprise Bean:

Implement Web Service as Stateless Session Bean

i Select an Enterprise Bean.

< Back    Next >    Finish    C

Browse Enterprise Bean

Enterprise Beans:

- ... EJBModule1
  - Enterprise Beans
    - ClientsFacade
    - CreneauxFacade
    - DaoJpa
    - Idao
    - MedecinsFacade
    - RvFacade

OK    Cancel

# Déployer le Web Service

The screenshot shows a Java development environment with two main windows. On the left is the 'Projects' view, displaying a hierarchy of projects and modules. The 'EnterpriseApplication1' project is expanded, showing 'Java EE Modules' containing 'EJBModule1.jar' and 'WebApplication1.war', and 'Configuration Files'. The 'WebApplication1' module contains 'Web Pages', 'Source Packages', 'Test Packages', 'Libraries', 'Test Libraries', 'Web Services', and a selected 'WSD' item. A context menu for 'WSD' is open, with 'Test Web Service' highlighted. Other options include 'Open', 'Refresh...', 'Add Operation...', 'Edit Web Service Attributes', 'Configure Handlers...', 'Generate and Copy WSDL...', 'Generate SOAP-over-HTTP Wrapper', 'Delete', and 'Properties'. On the right is a Mozilla Firefox browser window titled 'WSDaoJpaService Testeur de service Web - Mozilla Firefox'. The address bar shows 'http://localhost:8080/WebApplication1/WSDaoJpaService?Tester'. The page content is a test form for the 'WSDaoJpaService'. It includes a heading 'WSDaoJpaService Testeur de service Web', a sub-instruction 'Ce formulaire vous permet de tester l'implémentation du service Web ([Fichier WSDL](#))', and a note 'Pour appeler une opération, renseignez les zones d'entrée des paramètres de la méthode, puis cliquez sur le bouton portant le nom de cette méthode.' Below this, there are four code snippets representing service methods:

```
public abstract java.util.List rendezvous.WSDaoJpa.getAllClients()
    getAllClients()
```

```
public abstract java.util.List rendezvous.WSDaoJpa.getAllMedecins()
    getAllMedecins()
```

```
public abstract java.util.List rendezvous.WSDaoJpa.getAllCreneaux(rendezvous.Medecins)
    getAllCreneaux( )
```

```
public abstract java.util.List rendezvous.WSDaoJpa.getRvMedecinJour(rendezvous.Medecins,java.lang.String)
    getRvMedecinJour( , )
```

At the bottom of the browser window, there is a 'Terminé' button.

# Déployer le Web Service

Method invocation trace - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?  
http://localhost:10397/WebApplication2/WsDaoJpaService?Tester  
Les plus visités Débuter avec Firefox À la une

Method invocation trace

## getAllClients Method invocation

---

### Method parameter(s)

Type	Value
------	-------

---

### Method returned

java.util.List : "[rendezvous.Clients@1e40c9f, rendezvous.Clients@f8bb7e, rendezvous.Clients@1dd57f3, rendezvous.Clients@b93e8f]"

---

### SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:getAllClients xmlns:ns2="http://Rendezvous/" />
    </S:Body>
</S:Envelope>
```

---

### SOAP Response

Terminé

# Déployer le Web Service

Method invocation trace - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?

http://localhost:10397/WebApplication2/WsDaoJpaService?Tester

Les plus visités Débuter avec Firefox À la une

Method invocation trace

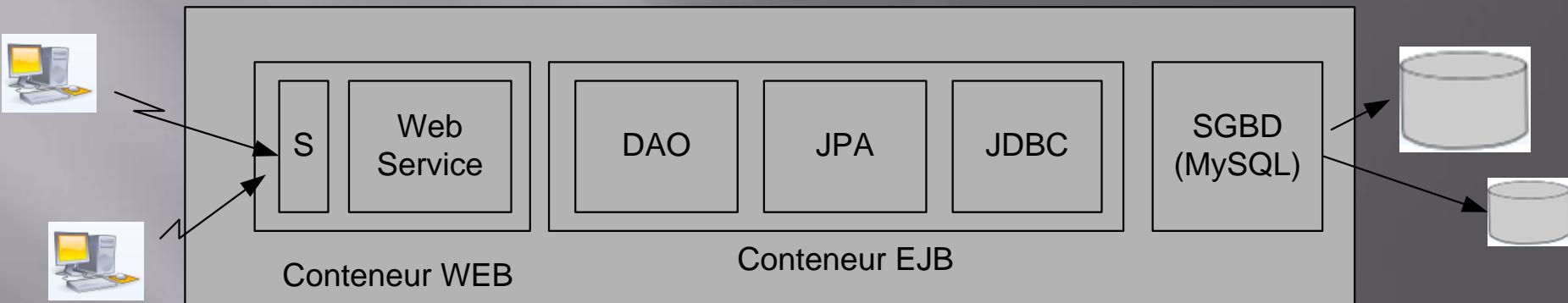
## SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Body>
        <ns2:getAllClientsResponse xmlns:ns2="http://Rendezvous/">
            <return>
                <id>1</id>
                <nom>MARTIN</nom>
                <prenom>Jules</prenom>
                <titre>Mr</titre>
            </return>
            <return>
                <id>2</id>
                <nom>GERMAN</nom>
                <prenom>Christine</prenom>
                <titre>Mme</titre>
            </return>
            <return>
                <id>3</id>
                <nom>JACQUARD</nom>
                <prenom>Jules</prenom>
                <titre>Mr</titre>
            </return>
            <return>
                <id>4</id>
                <nom>BISTROU</nom>
                <prenom>Brigitte</prenom>
                <titre>Melle</titre>
            </return>
        </ns2:getAllClientsResponse>
    </S:Body>
</S:Envelope>
```

Terminé

# Création d'une Enterprise Application

Enterprise Application



# Création d'une Enterprise Application

New Project

**Steps**

1. Choose Project
2. ...

**Choose Project**

**Categories:**

- Android
- Java
- JavaFX
- Java Web
- Java EE
- Java Card
- Java ME
- Maven
- PHP
- Ruby
- Groovy
- C/C++
- Native Modules

**Projects:**

- Enterprise Application
- Enterprise Application with Existing Sources
- EJB Module
- EJB Module with Existing Sources
- Enterprise Application Client
- Enterprise Application Client with Existing Sources
- Packaged Archive

**Description:**

**Creates a new enterprise application** in a standard project. You can also create an EJB module project and Web application project in the enterprise application. A standard project uses an **IDE-generated Ant build script** to build and run your projects.

< Back    Next >    Finish    Cancel    Help

# Création d'une Enterprise Application

New Enterprise Application X

**Steps**

1. Choose Project
- 2. Name and Location**
3. Server and Settings

**Name and Location**

Project Name:

Project Location:

Project Folder:

Use Dedicated Folder for Storing Libraries  
Libraries Folder:

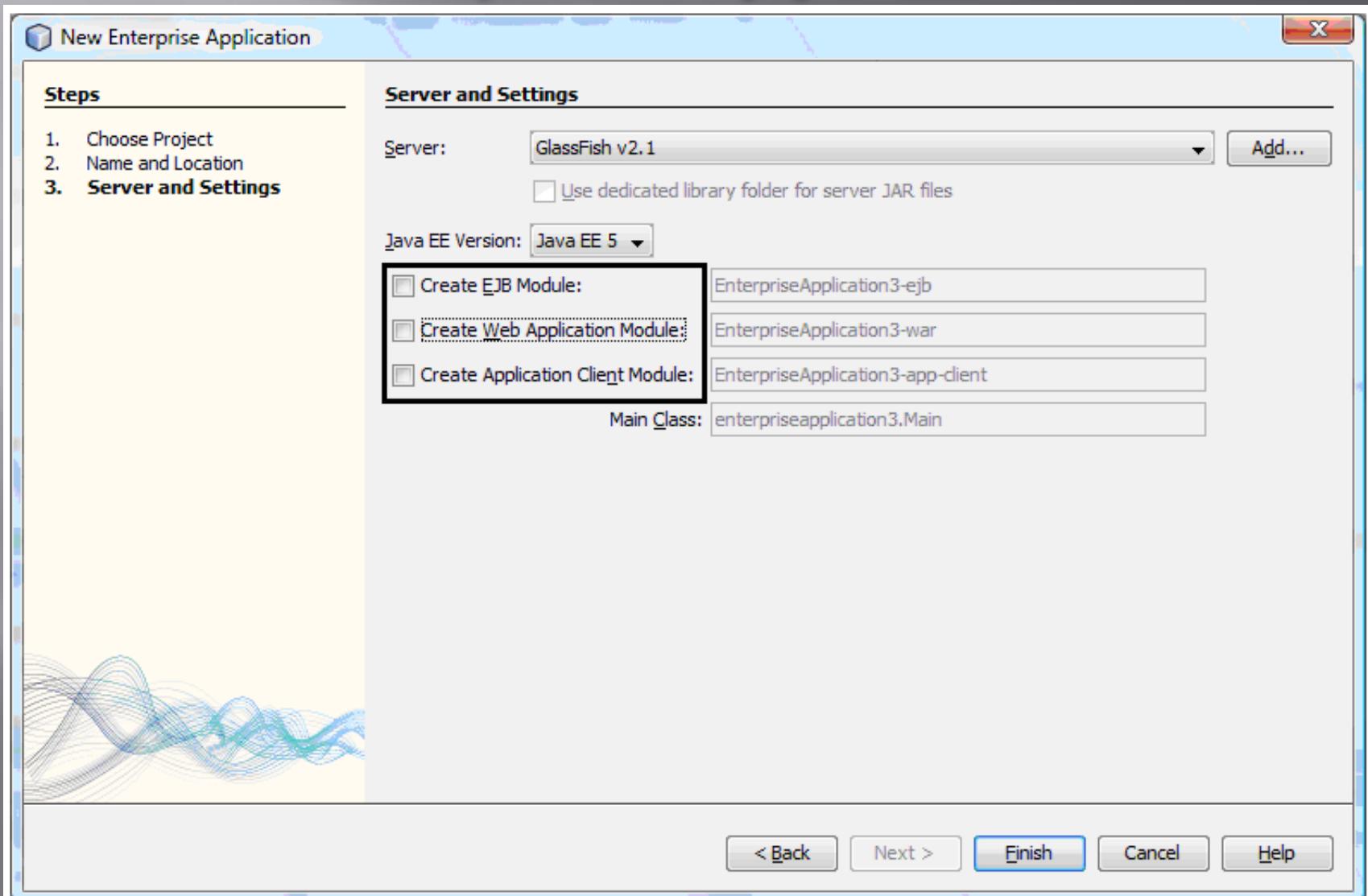
Different users and projects can share the same compilation libraries (see Help for details).

Set as Main Project

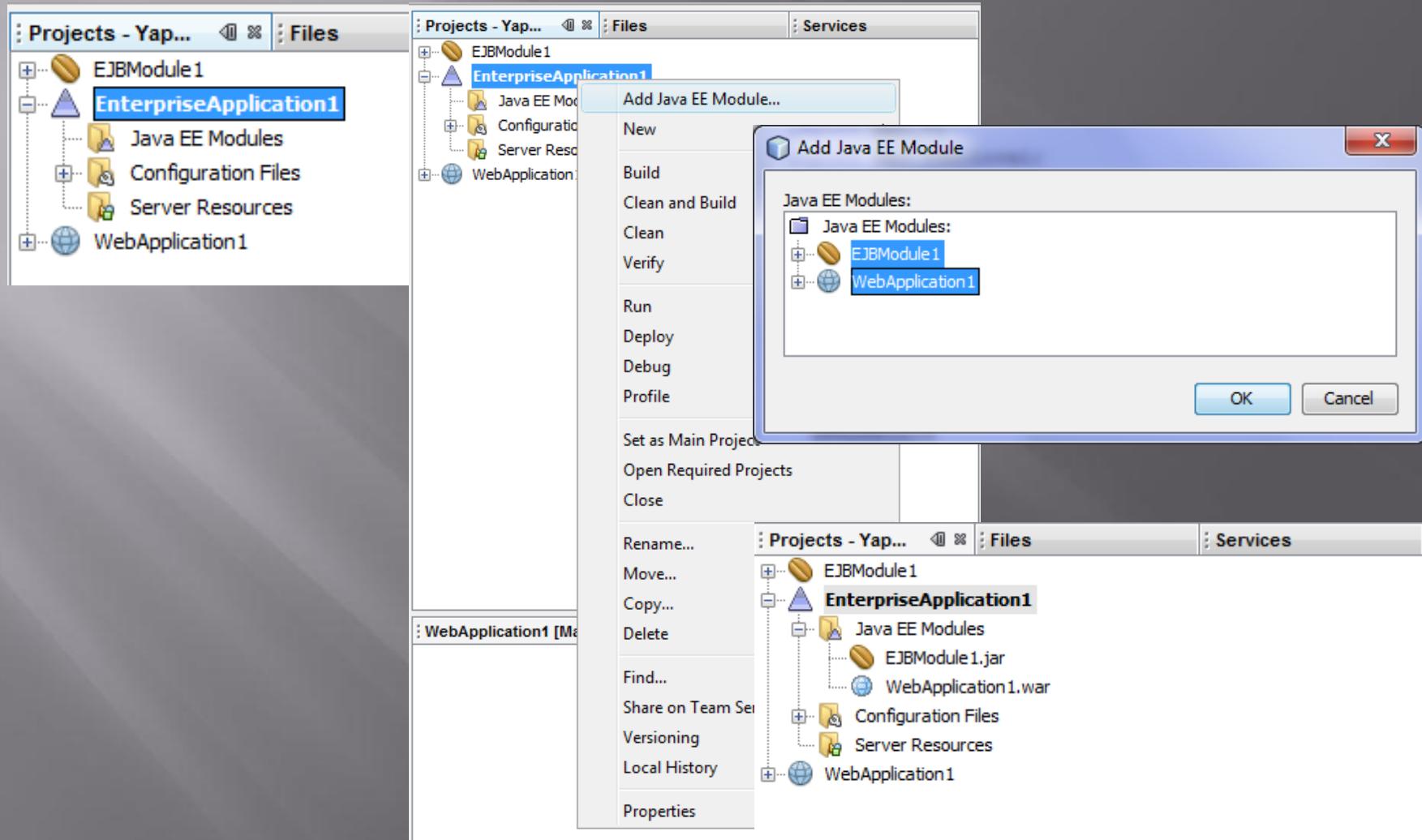


< Back Next > Finish Cancel Help

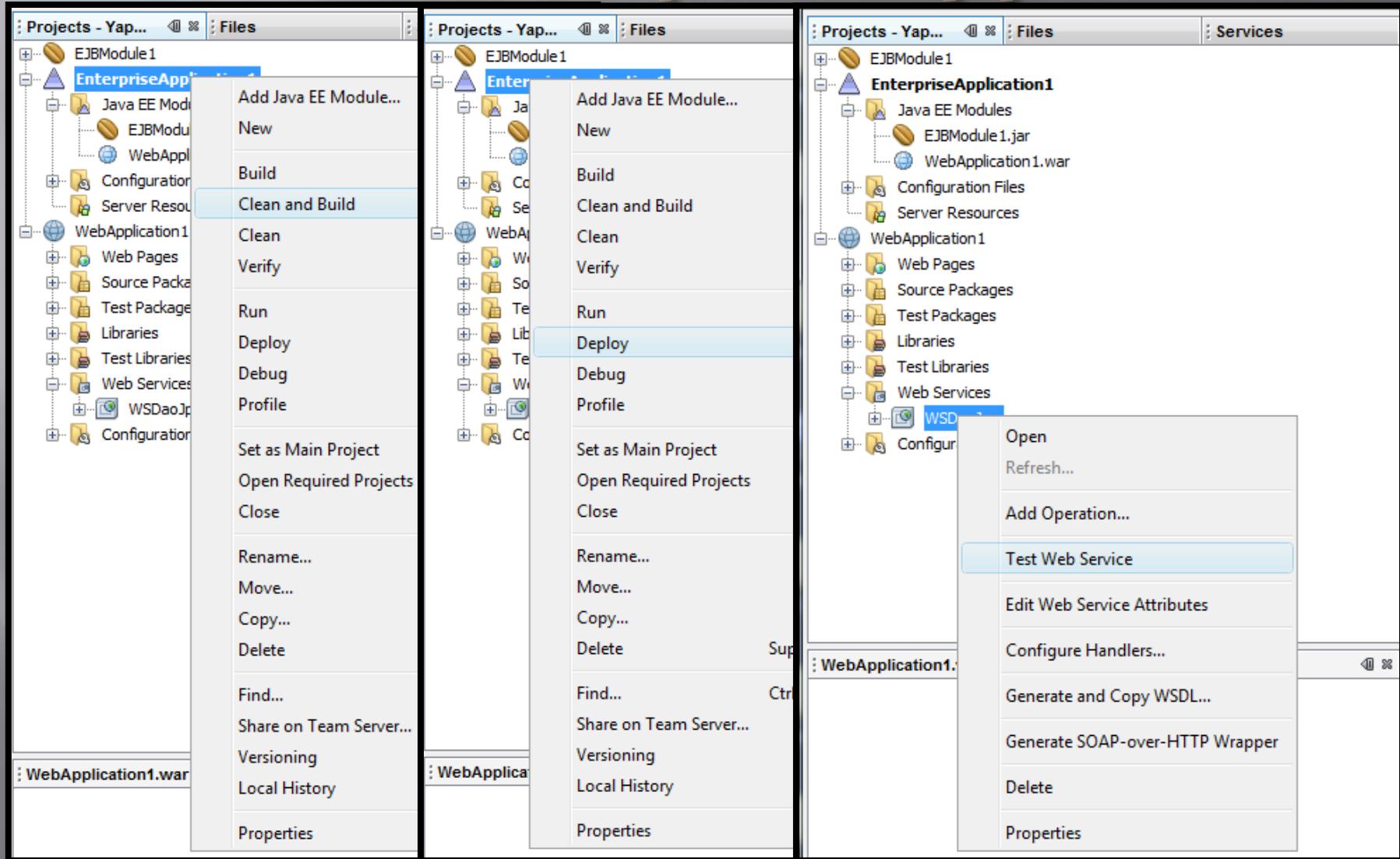
# Création d'une Enterprise Application



# Inclure l'EJB et la Web Application dans l'EnterpriseApplication



# Tester l'Enterprise Application



# Tester l'Enterprise Application

WSDaoJpaService Testeur de service Web - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?  
http://localhost:8080/WebApplication1/WSDaoJpaService?Tester  
Les plus visités Débuter avec Firefox À la une essai.html essai.html  
WSDaoJpaService Testeur de service...

## WSDaoJpaService Testeur de service Web

Ce formulaire vous permet de tester l'implémentation du service Web ([Fichier WSDL](#))

Pour appeler une opération, renseignez les zones d'entrée des paramètres de la méthode, puis cliquez sur le bouton portant le nom de cette méthode.

**Méthodes :**

public abstract java.util.List rendezvous.WSDaoJpa.getAllClients()  
**getAllClients ()**

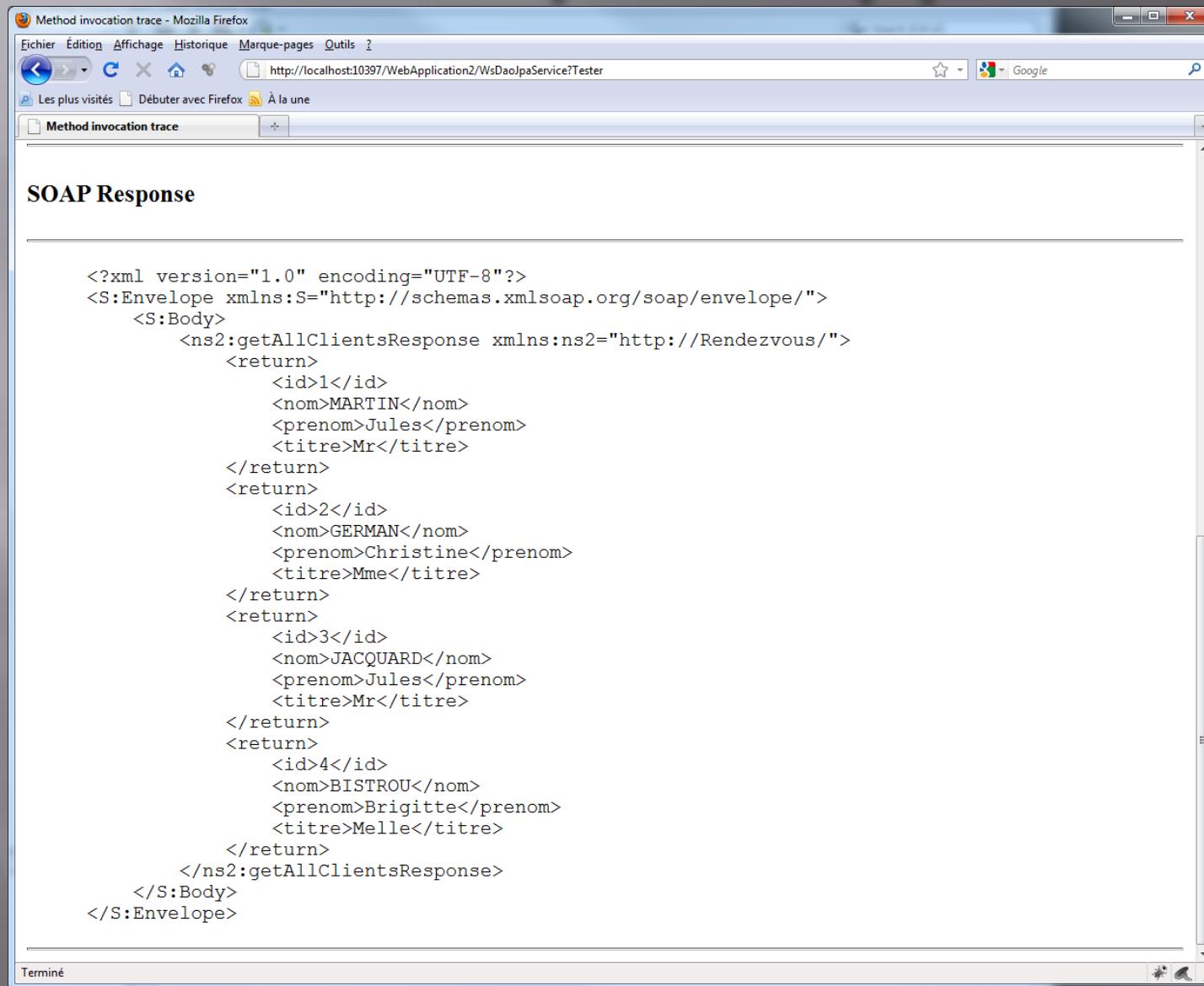
public abstract java.util.List rendezvous.WSDaoJpa.getAllMedecins()  
**getAllMedecins ()**

public abstract java.util.List rendezvous.WSDaoJpa.getAllCreneaux(rendezvous.Medecins)  
**getAllCreneaux ( )**

public abstract java.util.List rendezvous.WSDaoJpa.getRvMedecinJour(rendezvous.Medecins,java.lang.String)  
**getRvMedecinJour ( , )**

Terminé

# Tester l'Enterprise Application



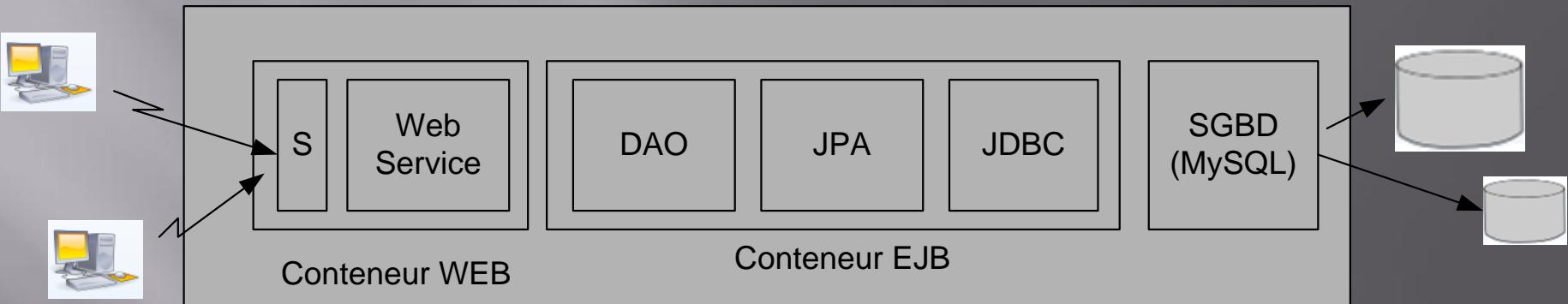
The screenshot shows a Mozilla Firefox window titled "Method invocation trace - Mozilla Firefox". The address bar displays the URL <http://localhost:10397/WebApplication2/WsDaoJpaService?Tester>. The main content area is titled "Method invocation trace" and contains the following XML response:

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getAllClientsResponse xmlns:ns2="http://Rendezvous/">
      <return>
        <id>1</id>
        <nom>MARTIN</nom>
        <prenom>Jules</prenom>
        <titre>Mr</titre>
      </return>
      <return>
        <id>2</id>
        <nom>GERMAN</nom>
        <prenom>Christine</prenom>
        <titre>Mme</titre>
      </return>
      <return>
        <id>3</id>
        <nom>JACQUARD</nom>
        <prenom>Jules</prenom>
        <titre>Mr</titre>
      </return>
      <return>
        <id>4</id>
        <nom>BISTROU</nom>
        <prenom>Brigitte</prenom>
        <titre>Melle</titre>
      </return>
    </ns2:getAllClientsResponse>
  </S:Body>
</S:Envelope>
```

The status bar at the bottom of the browser window shows the word "Terminé" (Completed).

# Création d'un client

Enterprise Application



# Création d'un client

New Project X

**Steps**

1. Choose Project  
2. ...

**Choose Project**

**Categories:**

- Java
- Java Web
- Java EE
- Java ME
- Maven
- Groovy
- NetBeans Modules
- Samples

**Projects:**

- Enterprise Application
- Enterprise Application with Existing Sources
- EJB Module
- EJB Module with Existing Sources
- Enterprise Application Client
- Enterprise Application Client with Existing Sources
- Packaged Archive

**Description:**

**Creates a new Enterprise application client project** in a standard IDE project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug the project.

< Back Next > Finish Cancel Help

# Création d'un client

New Enterprise Application Client X

**Steps**

1. Choose Project
- 2. Name and Location**
3. Server and Settings

**Name and Location**

Project Name:

Project Location:

Project Folder:

Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries  
(see Help for details).

Set as Main Project



# Création d'un client

New Enterprise Application Client

**Steps**

1. Choose Project
2. Name and Location
3. Server and Settings

**Server and Settings**

Add to Enterprise Application: EnterpriseApplication1

Server: GlassFish v2.1

Use dedicated library folder for server JAR files

Java EE Version: Java EE 5

Main Class: applicationclient1.Main

Projects - Yap... Files Services

- ApplicationClient1
  - Source Packages
    - applicationclient1
      - Main.java
  - Test Packages
  - Libraries
  - Test Libraries
  - Configuration Files
  - Server Resources
- EJBModule1
- EnterpriseApplication1
- WebApplication1

< Back

# Création d'un client

New File

**Steps**

1. Choose File Type  
2. ...

**Choose File Type**

Project: ApplicationClient1

**Categories:**

- AWT GUI Forms
- JUnit
- Persistence
- Groovy
- Hibernate
- Web Services
- XML
- GlassFish
- Android
- Other
- xml

**File Types:**

- Web Service Client
- Logical Handler
- Message Handler

**Description:**

Creates a web service client that is compliant with JSR-109.

< Back    Next >    Finish    Cancel    Help

# Ajout de la WSDaoJpa

New Web Service Client

**Steps**

1. Choose File Type
2. **WSDL and Client Location**

**WSDL and Client Location**

Specify the WSDL file of the Web Service.

Project:

Local File:

WSDL URL:

Specify a package name where the client java artifacts will be generated.

Project: ApplicationClient1

Package:

Client Style: JAX-WS Style

Generate Dispatch code

**i** Enter the URL of the service you wish to use.

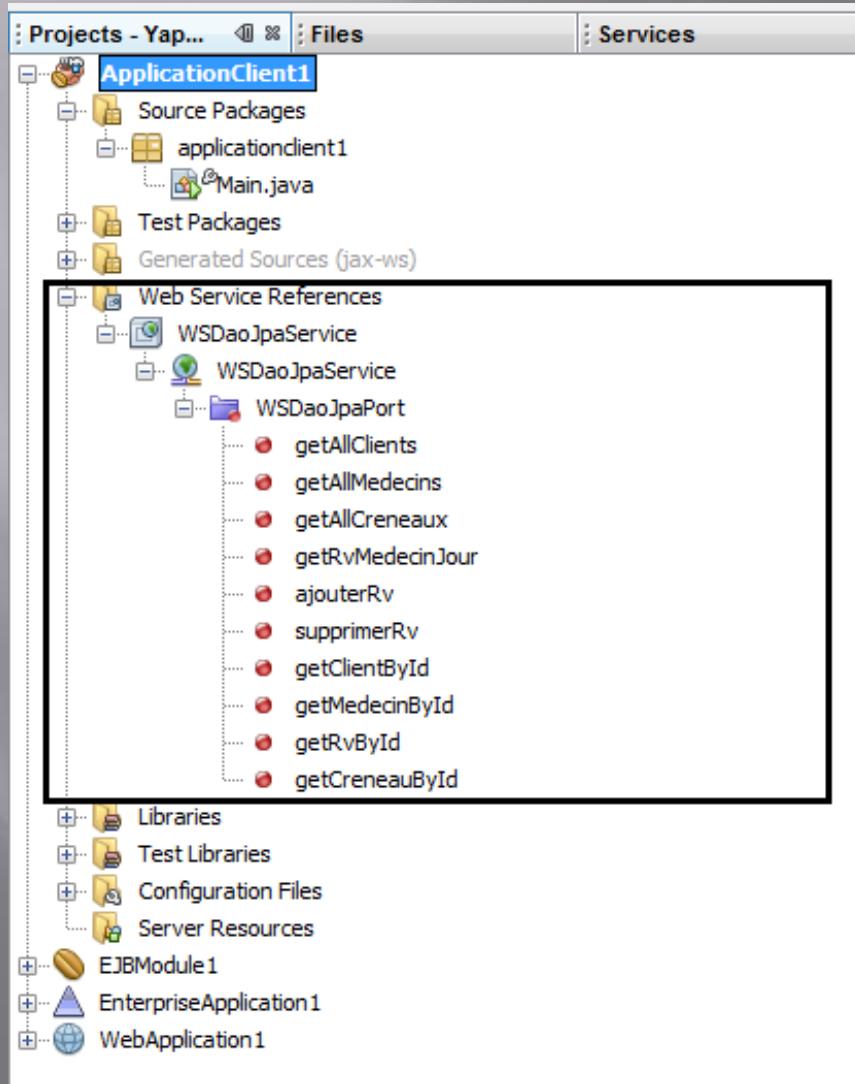
< Back

**Browse Web Services**

Web Services:

- WebApplication1
  - WSDaoJpa
    - getAllClients: List
    - getAllMedecins: List
    - getAllCreneaux: List
    - getRvMedecinJour: List
    - ajouterRv: Rv
    - supprimerRv: void
    - getClientById: Clients
    - getMedecinById: Medecins
    - getRvById: Rv
    - getCreneauById: Creneaux

# Ajout de la WSDaoJpa



- Modifier le fichier main.java pour qu'il instancie le WSDaoJpa

# Tester l'application cliente

```
: Output Tasks Usages
Personal GlassFish v3 Domain * x Retriever Output x Java DB Database Process * x ApplicationClient22 (run) x
-----
run-deploy:
Copying 1 file to C:\Users\lacomme.T3500-PC\Desktop\essai_gwt\JEE2\essai\ApplicationClient22\dist
run-tool2:
Copying 1 file to C:\Users\lacomme.T3500-PC\Desktop\essai_gwt\JEE2\essai\ApplicationClient22\dist\ApplicationClient22C]
8 juil. 2010 19:06:52 com.sun.enterprise.transaction.JavaEETransactionManagerSimplified initDelegates
INFO: Using com.sun.enterprise.transaction.jts.JavaEETransactionManagerJTSDelegate as the delegate
MARTIN
old-run-jar:
run-display-browser:
run:
BUILD SUCCESSFUL (total time: 7 seconds)
```