Sistemas Web II

Práctica 3

REST

Carlos Ambrosio Habela

David Recio Arnés

1.Introducción

La elección del proyecto del documento para trabajar con REST en el entorno de desarrollo es sobre un recetario y recetas como en el proyecto anterior. La elección se llevó a cabo por dos integrantes del grupo.

El proyecto se desarrolla en Java, en el entorno de desarrollo *Neatbeans*. Se ha usado también para el correcto desarrollo por parte de los dos integrantes del grupo el control de versiones *Github* para llevar un control y facilitar el reparto de tareas y el seguimiento del proyecto.

2.Resumen

En el caso del proyecto, como no se pide ninguna interfaz gráfica se prefirió decantarse por la terminal de salida sirviendo de un menú simple, para poder enforcase en el tema principal de la práctica.

Para darle un cierto control de la programación se decidió implementar un repositorio como se dijo anteriormente, además de definir una buena estructura, todo ello permitió trabajar paralelamente ya que se tocaban zonas de código diferente, e incluso poder hacer uso del *pair programing*.

El proyecto consta de tres partes diferenciadas o tres proyectos distintos, un proyecto servidor en web y dos proyectos que actúan de clientes, uno en Java y otro en web.

Los clientes le harán peticiones al servidor y este consumirá los recursos y los ofrecerá.

Los ficheros XML necesarios para el correcto funcionamiento del proyecto se encuentran en la carpeta *files* de los respectivos proyectos, es decir, en el caso por ejemplo del servidor cliente java la carpeta sería (Practica3RESTClienteJava/files/xml), el funcionamiento es similar con los otros proyectos.

Respecto a la ejecución, solo hay un dato importante, se debe crear o importar un recetario antes de crear las recetas, ya que este es el que actuará de almacén.

En este proyecto se necesita persistencia de los datos, por lo pide la creación de una base de datos, con todos los datos que se crean oportunos. La base de datos recibe el nombre de *pruebaRecetario* desarrollada en *MySQL* y cuyos scripts se ubican en una carpeta llamada Scripts BD Recetario, para poder meter todos los datos en ella, más adelante se explicará cómo se conecta y como se accede a ella.

Para poder tener acceso a la base de datos y poder conectarla posteriormente con el proyecto en *Netbeans* es necesario tener instalado MySQL SI PUEDE SER LA VERSION 5.7, aunque esto no es un requisito esencial ya que también se pude trabajar con versiones posteriores.

3.Desarrollo y despliegue

El proyecto cuyo nombre es *Practica3REST*, está desarrollado en Java y consta como se dijo anteriormente de tres proyectos separados, dos clientes (web y java) y el servidor. Cada uno de ellos tiene una función y unos paquetes con unas clases diferenciadas de las cuales se hablará más adelante. Además de la ya mencionada Base de Datos (*pruebaRecetario*).

El proyecto *Practica3RESTClass* es el servidor y consta de tres paquetes: *Funcionalidad, Recursos y servicioRest* con sus respectivas clases.

- El paquete Funcionalidad tiene dos clases Marshalling, ValidarXSD y *AccesoBBDD* que consta de todo lo necesario para el funcionamiento y la conexión con la Base de Datos *pruebaRecetario.*
- La clase *Marshalling* realiza el *marshalling* y *Unmarshalling*, que sirve para hacer volcados de objetos a XML o viceversa (*unmarshalling*). La clase *ValidarXSD* se usa, como su nombre indica para validar los XML correspondientes en la carpeta files y dentro de esta en la subcarpeta con nombre *xsd* y el fichero recetario. La clase *AccesoBBDD* como se indicó anteriormente será la encargada de gestionar la conexión, realizar consultas, crear tablas, cerrar la conexión, etc.

- El paquete Recursos tiene dos clases que son Receta y Recetario, que tiene la misma funcionalidad que en la practica1 siendo estas las clases con las que se va a trabajar a lo largo de los proyectos. Estas clases son idénticas a las de la primera práctica. Se le añaden algunas más, necesarias con la creación de la base de datos y la relación entre ellas, que son:

    - *Usuario*, o autentificación, que se encargara de que el usuario introducido y su contraseña sean correctos, para poder acceder a los recetarios disponibles.
    - *RecetarioReceta*, esta clase es el nexo de unión entre las recetas y los recetarios para poder buscarlos por *ids*, y poder trabajar con ellos.
    - *ConjuntoRecetario,* clase necesario para poder trabajar con los usuarios y poder asociarlos a recetarios en concreto.
    - *FileUSer:* clase que relaciona un fichero con su id

- El paquete *servicioRest* contiene una clase ApplicationConfig que se crea automáticamente.
- La clase *ServicioRestResource,* es una clase única donde se encuentran todos los recursos necesarios para el desarrollo de la práctica como pueden ser añadir Receta y recetario, borrarlas, listar recetas y recetarios, *exportarReceta*, *exportarRecetario*, etc.

El proyecto *Practica3RESTClienteJava* es el cliente Java y consta cuatro paquetes: *practica3restclientejava, Funcionalidad, Cliente y Recursos* con sus respectivas clases:

- El paquete *practica3restclientejava*, es como el *launcher* que se encarga de llamar al menú.
- El paquete *Recursos* contiene las clases necesarias para la realización del recetario ya mencionadas antes, estas son: *Receta, Recetario, RecetarioReceta, ConjuntoRecetarios, Usuario y FileUser.*
- El paquete *cliente*, tiene una única clase y es la clase *cliente* que es la encargada de establecer una correcta conexión con los servicios rest.
- El paquete *Funcionalidad* tiene dos clases, que son el menú y modelo que se encarga de hacer el trabajo pesado de la aplicación, donde participan las diferentes clases.

El proyecto *Practica3RESTClienteWeb* es el servidor web y consta de tres paquetes *beans*, en los que están las clases necesarias como en el paquete funcionalidad del servidor o el cliente Java, que son: *Receta, Recetario, ConjuntoRecetario, RecetarioReceta o FilerUser*, necesarias para las

servlets de las que se hablará posteriormente. Y el segundo paquete contiene las *servlets* implementadas.
Y el tercer paquete, llamado *cliente,* consta de una clase con el mismo nombre, que es creada automáticamente en el cliente Java (es el mismo) y se debe meter también en el cliente web para su correcto funcionamiento.
Se puede observar que el cliente web tiene como añadido las servlets implementadas, ya que es esta la que accede a los recursos del servidor
A continuación, se explicarán las clases *servlets* y los respectivos archivos *html* para su correcto funcionamiento.

- Modelo: Tiene la funcionalidad y se encarga de hacer el trabajo mas costoso de la aplicación, es necesario tal como está diseñado ahora el funcionamiento de las servlets
- *CrearRecetarioServlet:* Esta servlet crea recetarios.
- *CrearRecetaServlet:* Esta servlet crea recetas.
- *LeerRecetarioServlet:* Se obtiene un listado de recetarios disponible.
- *LeerRecetaServlet:* Se obtiene un listado de recetas disponible.
- *BorrarReceta:* Borra las recetas seleccionadas.
- *BorrarRecetario:* Borra los recetarios seleccionados.
- *LoginServlet:* Sevlet que conprueba si el usuario y la contraseña indicadas son correctas.
- *ValidarXSD:* Servlet que introduciendo el fichero lo valida contra un xsd.
- *ExportarRecetario:* Exporta el recetario elegido
- *ExportarReceta:* Exporta la receta elegido
- *ImportarRecetario:* Importa el recetario elegido
- *ImportarReceta:* Importa la receta elegida
- *AnnadirReceta:* Servlet que añade una receta al recetario que se elija

En la carpeta *web pages* dentro del proyecto de *PracticaRESTClienteWeb*, se encuentran unos archivos *html* que son necesarios que actúan como formularios para la creación o bien para obtener la lista de recetas y recetarios.

Cabe destacar, que, para la compilación y ejecución de este proyecto, se han de tener en cuenta que se debe tener desplegado el servidor y al menos un cliente ejecutándose y se tiene que seguir unos pasos para su ejecución. En primer lugar, se debe "*deployar"* o desplegar el proyecto del servidor, es decir, el *Practica3RESTClass.* Esto se hace de la siguiente manera (hay que pulsar botón derecho y pinchar en *Deploy*). Una vez hecho esto, ya se pueden compilar el resto del proyecto.

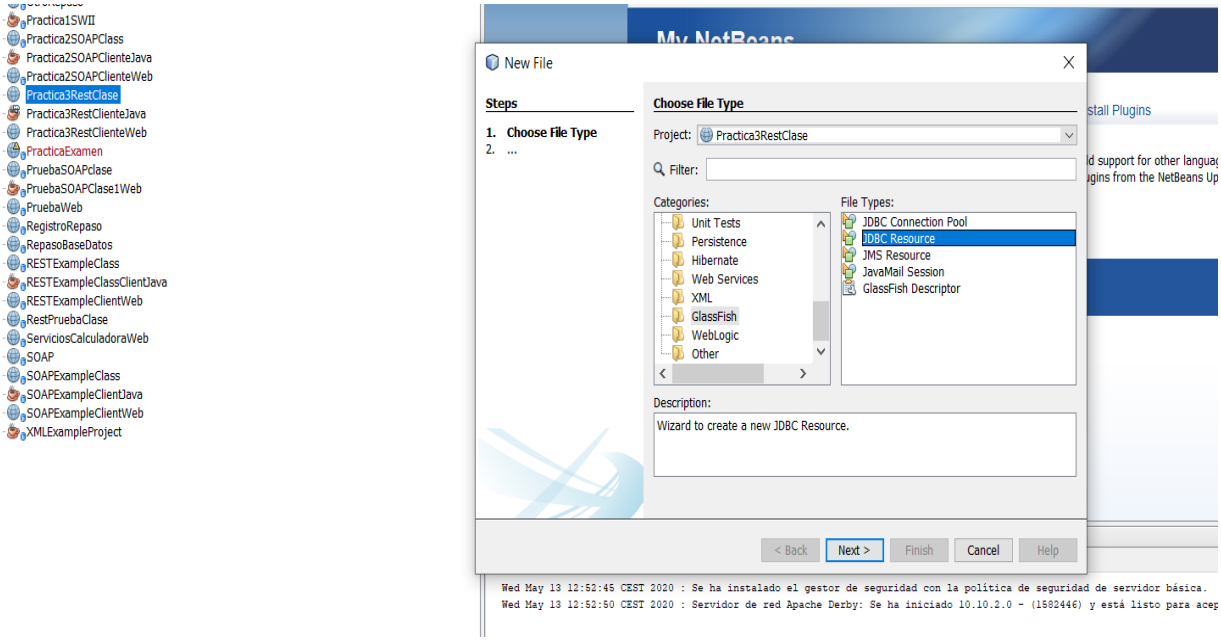
Para crear un cliente Java debes irte al cliente Java y ejecutar el *launcher*, una vez ahí simplemente debes importar un recetario o crear uno nuevo.

Para crear un cliente web debes ejecutar el *html* llamado *index* y una vez ahí, simplemente se debe importar un recetario o crear uno nuevo.

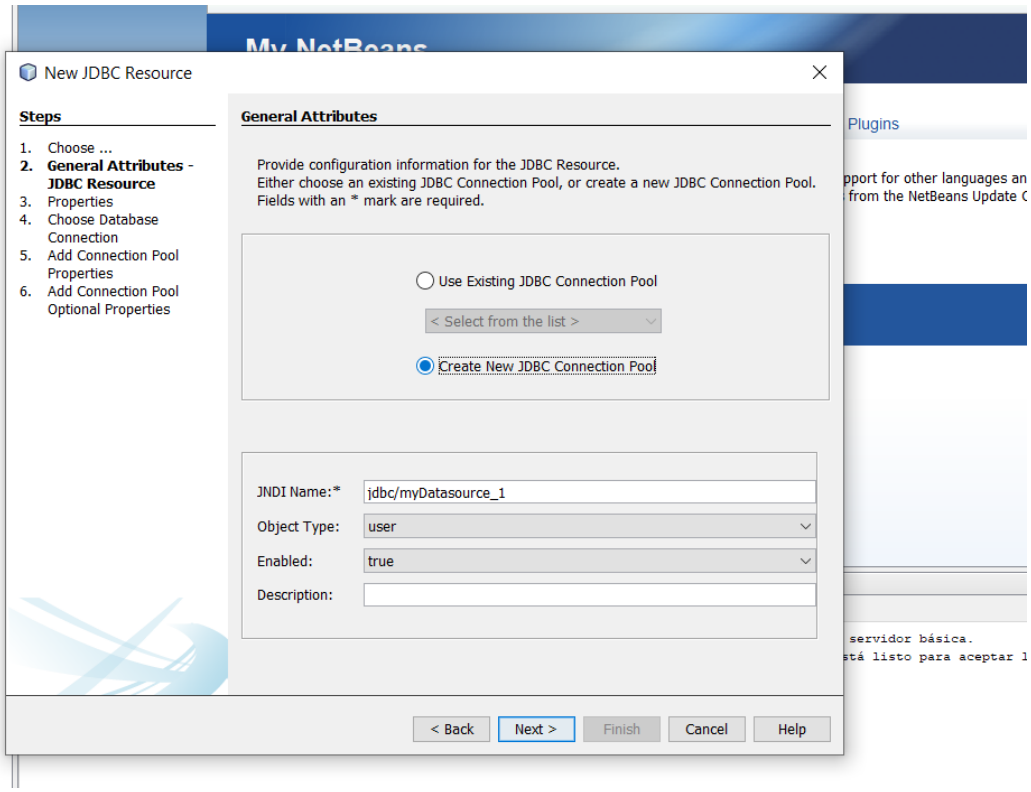
Cabe destacar que, para exportar recetas, se debe haber creado alguna, no hay problema de solapamiento a la hora de almacenar datos en el servidor temporalmente, ya que sus funciones son atómicas y se trabaja siempre sobre las mismas variables (almacenadas en el servidor), de tal forma que no se tiene de ellas ninguna copia local en el cliente, impidiendo así problemas de inconsistencia por falta de sincronía.

Si se requiere de algún recetario/receta que se quiera importar al servidor o que este lo mande, se trabajará bajo la carpeta *xml*, ubicada en el cliente. En el traspaso de ficheros de cliente a servidor, se realizará mediante una trasmisión de *arrays* de bytes.

En cuanto a la base de datos que se llama *pruebaRecetario*, como ya se dijo anteriormente, se debe tener instalado MySQL y tener el servicio activado. Se debe realizar un *pool* de conexiones a esta base de datos. Se debe conectar a la base de datos (*pruebaRecetario*) . Posteriormente en el cliente web pulsar botón derecho y en *New* buscar en *Others, en la carpeta Glassfish* elegir la segunda opción como se muestra en la figura (*JDBC Resource*).



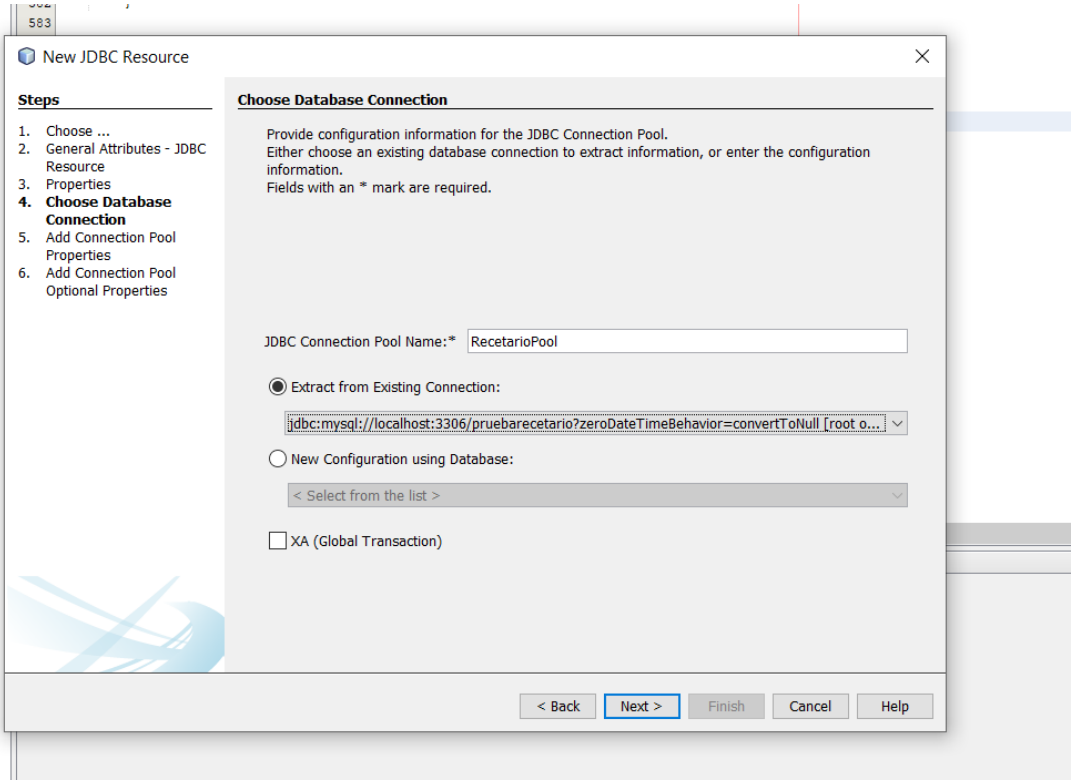
Posteriormente elegir *Create New JDBC Connection Pool* como se muestra en la siguiente figura.



Y ahí elegir el nombre que se quiera pulsar *Next.*

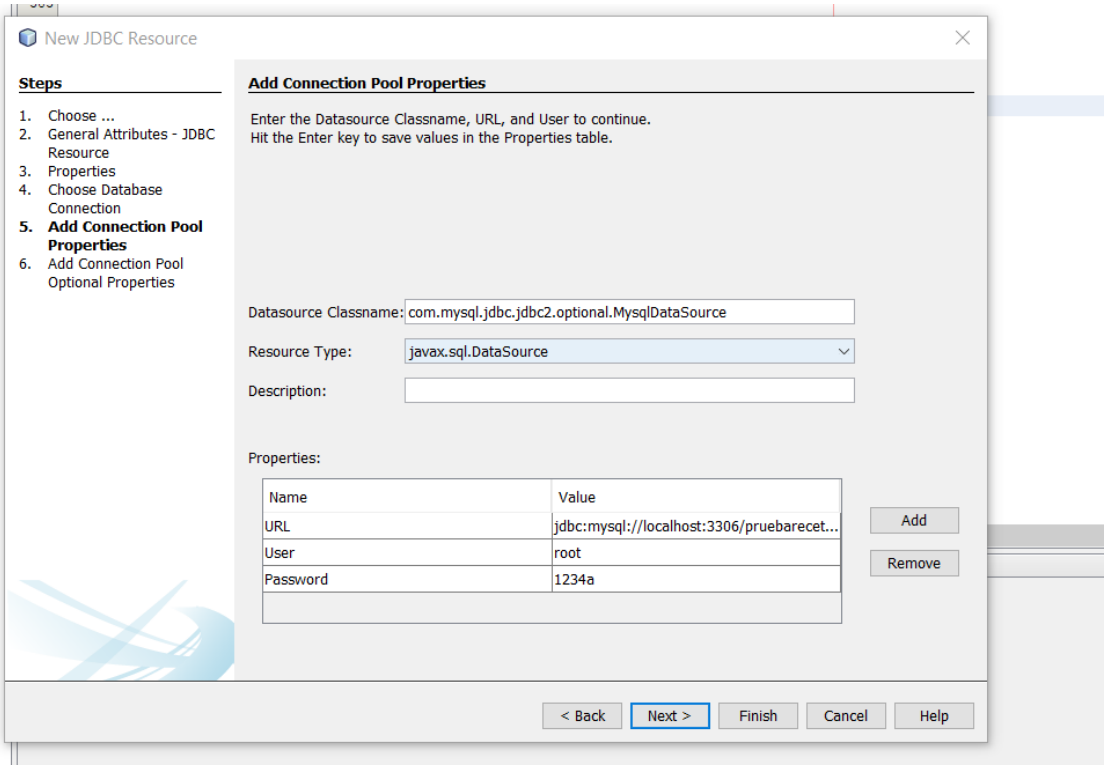
Nota: Si se elige el nombre que se quiera después se ha de cambiar en la clase *AccesoBBDD* en la función dedicada a la conexión, por el nombre elegido. Si no poner el mismo que se ha elegido por parte del grupo. (jdbc/myDatasource).

Posteriormente se pulsa Next y la pantalla nos muestra algo como lo que se muestra en la figura 3



Se elige un nombre para el *Pool* y se selecciona la base de datos que se quiere, en este caso *pruebaRecetario* para conectar el *pool* a la base de datos, se pulsa *Next.*

Para finalizar hay que realizar una última cosa antes de pulsar finalizar como se muestra en la figura 4



En *ResourceType* seleccionar lo que se muestra en la figura y pulsar *Next*, después en la siguiente imagen pulsar Finalizar.

Se puede ver el recurso que se ha creado en WEB INF>glassfish-resources.xml

Hay que tener cuidado, por algún motivo glassfish-resources.xml se crea en la carpeta equivocada

- Debemos moverlo manualmente desde /Proyecto/web/WEB-INF a / setup

- Finalmente, para usarlo en nuestro proyecto debemos añadir la etiqueta correspondiente al web.xml
  <resource-ref>
  <res-ref-name>pruebaRecetario </res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Shareable</res-sharing-scope>
  </resource-ref>

Todo lo necesario está en la carpeta *Scripts BD Recetario*, hay un archivo con consultas por si se quiere probar la base de datos, otro con datos de inserción, aunque se puede añadir los que se quieran para probarla, y también en el archivo *CreacionBBDD* está desde el borrado de la base datos, la creación de la base datos y la creación de las tablas necesarias con los atributos de cada tabla, por si fuera necesario. Simplemente se puede copiar todo lo que hay en *CreacionBBDD* en la consola que se abre de MySQL Client pulsar *Intro* e introducir datos con los *insterts* dados u otros nuevos y comprobar con las consultas dadas o probar otras distintas.