

# Gestión de hoteles con APACHE-TOMCAT

Adrián Sánchez Carchano  
2º DAW CIPFP Batoi

## Sumario

Breve descripción del proyecto.....	3
Cliente Windows 7.....	4
Configuración de red.....	4
Servidor web Apache y servidor DNS, Ubuntu server.....	5
Configuración de red:.....	5
Servicios:.....	5
Servidor de aplicaciones Tomcat-1, MYSQL Server.....	8
Configuración de red:.....	8
Instalando Mysql server:.....	9
Conectando con la base de datos.....	9
Creando la tabla hoteles.....	11
Desarrollando nuestro primer Servlet de Java.....	13
Creando el servlet SearchHotel.....	14
Modificando máquina con servidor web APACHE, parte 1.....	21
Comprobación.....	22
Servidor de aplicaciones Tomcat-2.....	23
Configuración de red:.....	23
Configurando Tomcat:.....	23
Modificando máquina con servidor web APACHE, parte 2.....	28
Comprobación:.....	28
Servidor de aplicaciones Tomcat-3.....	30
Configuración de red:.....	30
Creando el servlet:.....	30
Modificando máquina con servidor web APACHE, parte 3.....	33
Comprobación:.....	33
Reflexión personal.....	35

## Breve descripción del proyecto

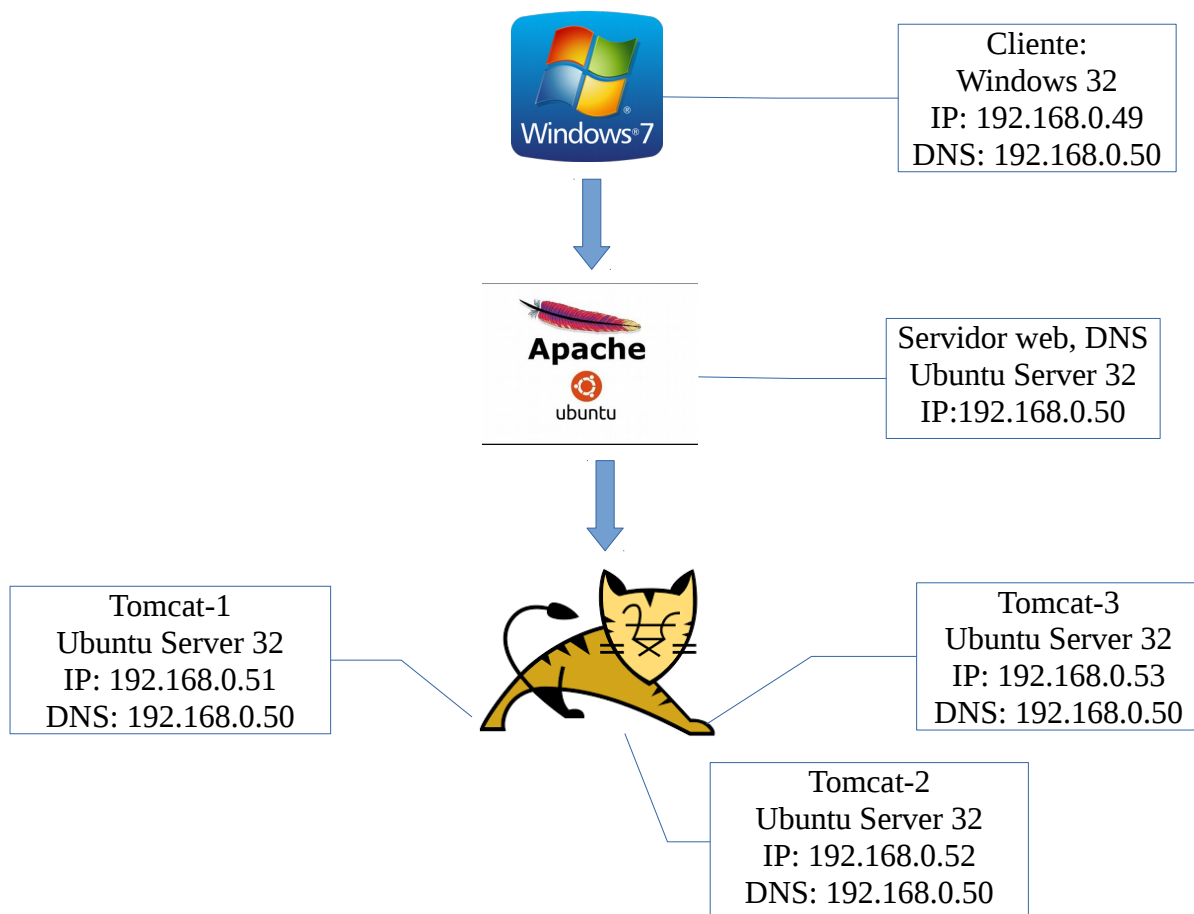
A continuación vamos a describir el proyecto que hemos realizado en la empresa de servicios Myhost.

Esta vez hemos tenido que utilizar cinco máquinas virtuales para el correcto desarrollo, ha sido una máquina cliente con sistema operativo Windows, una máquina con sistema operativo Ubuntu server, que hará la función de servidor web que ofrecerá varios servicios y tres máquinas con Tomcat también con sistema operativo Ubuntu server, cada una de las tres últimas nos ofrecerá varias posibilidades;

- **Buscador de hoteles**, esta aplicación nos mostrará información sobre los hoteles que hay almacenados en la base de datos.
- **Introducir nuevos hoteles**, esta aplicación nos permitirá introducir nuevos hoteles con identificación de usuario
- **Configuración de ofertas** a los clientes, con sesión de usuario de dos horas u mostrará las veces que un usuario se ha conectado.

Hay que remarcar que en el servidor web, donde tenemos alojado Apache, también hemos configurado un servidor DNS y en cliente hemos realizado algunas modificaciones, vamos a describirlas un poco a continuación.

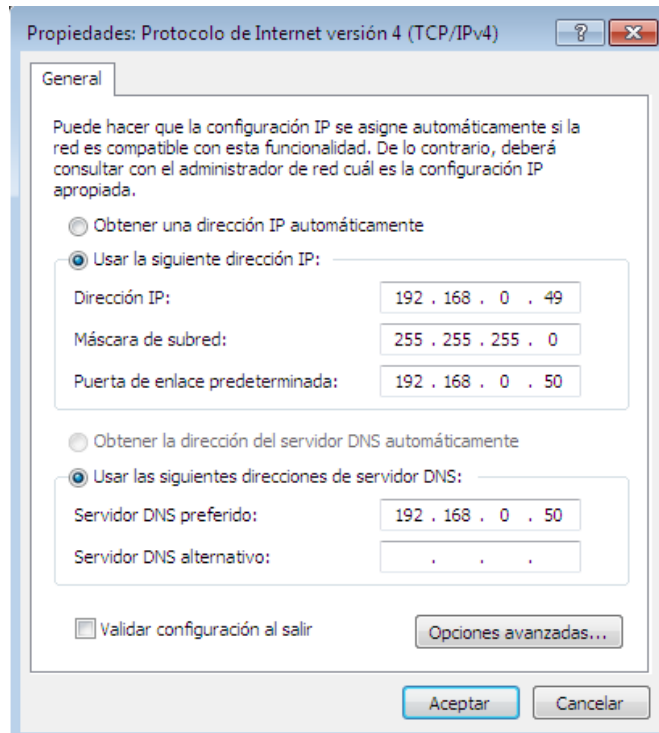
Infraestructura del proyecto.



A continuación vamos a describir los aspectos técnicos de las máquinas virtuales:

### Configuración de red

Para configurar el cliente, solo hemos tenido que asignarle una ip fija, la 192.168.0.49 y asignarle un DNS, 192.168.0.50 que hará referencia a la máquina virtual donde tendremos nuestro servidor DNS instalado.



Ahora comprobaremos mediante el uso “nslookup” que cuando le pedimos el dominio [www.hotel.es](http://www.hotel.es), el primer sitio donde va a buscarlo es al servidor DNS que configuraremos en nuestra máquina virtual con Bind9.

```
C:\Windows\system32\cmd.exe - nslookup

C:\Users\cliente>nslookup
Servidor predeterminado: UnKnown
Address: 192.168.0.50

> www.hotel.es
Servidor: UnKnown
Address: 192.168.0.50

Nombre: www.hotel.es
Address: 192.168.0.50

>
```

## Configuración de red:

Hemos configurado la ip fija, esta vez para esta máquina será la IP:192.168.0.50

```
ubuntu@ubuntu:/var/www/hoteles$ ifconfig -qa
ifconfig: opción '-qa' no reconocida.
ifconfig: '--help' le da información de como usar la orden.
ubuntu@ubuntu:/var/www/hoteles$ ifconfig -a
enp0s3      Link encap:Ethernet direcciónHW 08:00:27:5d:dc:8b
            Direc. inet:10.0.2.15 Difus.:10.0.2.255 Másc:255.255.255.0
            Dirección inet6: fe80::a00:27ff:fe5d:dc8b/64 Alcance:Enlace
            ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
            Paquetes RX:14527 errores:0 perdidos:0 overruns:0 frame:0
            Paquetes TX:5461 errores:0 perdidos:0 overruns:0 carrier:0
            colisiones:0 long.colaTX:1000
            Bytes RX:13922825 (13.9 MB) TX bytes:336944 (336.9 KB)

enp0s8      Link encap:Ethernet direcciónHW 08:00:27:51:1b:f7
            Direc. inet:192.168.0.50 Difus.:192.168.0.255 Másc:255.255.255.0
            Dirección inet6: fe80::a00:27ff:fe51:1bf7/64 Alcance:Enlace
            ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
            Paquetes RX:458 errores:0 perdidos:0 overruns:0 frame:0
            Paquetes TX:8 errores:0 perdidos:0 overruns:0 carrier:0
            colisiones:0 long.colaTX:1000
            Bytes RX:64769 (64.7 KB) TX bytes:648 (648.0 B)

lo          Link encap:Bucle local
            Direc. inet:127.0.0.1 Másc:255.0.0.0
            Dirección inet6: ::1/128 Alcance:Anfitrión
            ACTIVO BUCLE FUNCIONANDO MTU:65536 Métrica:1
            Paquetes RX:169 errores:0 perdidos:0 overruns:0 frame:0
            Paquetes TX:169 errores:0 perdidos:0 overruns:0 carrier:0
            colisiones:0 long.colaTX:1
            Bytes RX:12289 (12.2 KB) TX bytes:12289 (12.2 KB)
```

## Servicios:

Instalaremos Apache Web server:

```
ubuntu@ubuntu:/$ apache2 -version
Server version: Apache/2.4.18 (Ubuntu)
Server built: 2016-07-14T12:32:26
ubuntu@ubuntu:/$
```

Habilitaremos los módulos que vayamos a utilizar, estos son:

- proxy.conf
- proxy\_http.load
- proxy.load

```
ubuntu@ubuntu:/etc/apache2/mods-enabled$ ls
access_compat.load  authz_core.load  deflate.load  mime.load  proxy_http.load
alias.conf          authz_host.load  dir.conf     mpm_event.conf  proxy.load
alias.load          authz_user.load  dir.load     mpm_event.load  setenvif.conf
auth_basic.load     autoindex.conf  env.load     negotiation.conf setenvif.load
authn_core.load     autoindex.load  filter.load  negotiation.load status.conf
authn_file.load     deflate.conf     mime.conf    proxy.conf      status.load
ubuntu@ubuntu:/etc/apache2/mods-enabled$ _
```

A continuación vamos a crear nuestro fichero virtualhost, para ello haremos una copia del fichero default que nos proporciona apache como plantilla para desarrollar nuestras páginas web:

```
ubuntu@ubuntu:/etc/apache2/sites-enabled$ sudo cp 000-default.conf hoteles.es.conf
```

A continuación listaremos la carpeta para comprobar que los ficheros se encuentran correctamente:

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:/etc/apache2/sites-enabled$ ls -la
total 12
drwxr-xr-x 2 root root 4096 may 11 16:53 .
drwxr-xr-x 8 root root 4096 may 11 16:14 ..
lrwxrwxrwx 1 root root 35 may 4 14:47 000-default.conf -> ../sites-available/000-default.conf
-rw-r--r-- 1 root root 1504 may 4 17:25 hoteles.es.conf
ubuntu@ubuntu:/etc/apache2/sites-enabled$ _
```

A continuación vamos a editar el fichero de configuración de nuestro dominio:

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
GNU nano 2.5.3 Archivo: hoteles.es.conf
<VirtualHost 192.168.0.50:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.

ServerName www.hoteles.es
ServerAlias www.hoteles.es

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html/hoteles

<Directory /var/www/html/hoteles>
    DirectoryIndex index.html
    Options FollowSymLinks
    Require all granted
</Directory>

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
[ 39 líneas leídas ]
Ver ayuda  Guardar  Buscar  Cortar Text  Justificar  Posición
Salir  Leer fich.  Reemplazar  Pegar txt  Ortografía  Ir a línea
```

```
ServerName www.hoteles.es
ServerAlias www.hoteles.es

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html/hoteles

<Directory /var/www/html/hoteles>
    DirectoryIndex index.html
    Options FollowSymLinks
    Require all granted
</Directory>
```

Indicamos que los ficheros relacionados con esta página web están en la carpeta /var/www/html/hoteles y que la página principal es la de index.html. A continuación vamos a crear la carpeta que almacenará el index de nuestra página web:

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:/var/www/html$ mkdir hoteles
```

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:/var/www/html$ ls -la
total 24
drwxr-xr-x 3 root root 4096 may  4 16:26 .
drwxr-xr-x 3 root root 4096 may  4 16:29 ..
drwxr-xr-x 2 root root 4096 may  4 17:24 hoteles
-rwxr-xr-x 1 root root 11321 may  4 14:47 index.html
ubuntu@ubuntu:/var/www/html$
```

Y dentro de ella vamos a crear nuestro fichero index.html:

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
GNU nano 2.5.3 Archivo: index.html

<html>
  <head>
    <title> Hoteles - TOMCAT - ADRISC </title>
  </head>
  <body>
    <h1>Welcome to a diferent option to search a hotel</h1>
    <p>
      <a href="/Tomcat-1/SearchHotels/buscaHoteles">Search hotels</a>
    </p>
    <p>
      <a href="/Tomcat-2/NewHotel/">Insert a new hotel</a>
    </p>
    <p>
      <a href="/Tomcat-3/ConfigHotel/">Config hotel</a>
    </p>
  </body>
</html>
```

Observamos que en los diferentes enlaces no ponemos ninguna ip ni dirección reconocida, luego lo modificaremos.

Ahora para comprobar que todos los pasos anteriores se han llevado a cabo con éxito trataremos de acceder a esta página desde nuestro cliente realizando una petición en el navegador del mismo ha la dirección [www.hoteles.es](http://www.hoteles.es).



En cuanto a la configuración del servidor DNS no vamos a entrar en detalles debido a que ya lo configuramos en el proyecto anterior. Los archivos de configuración se incluirán en la presentación de este mismo proyecto.

## Servidor de aplicaciones Tomcat-1, MYSQL Server

### Configuración de red:

La IP que le hemos asignado ha sido esta vez la 192.168.0.51

```
Ubuntu-server-TOMCAT-1A [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:~$ ifconfig -a
eth0      Link encap:Ethernet  direcciónHW 08:00:27:a7:08:e6
          Direc. inet:192.168.0.51  Difus.:192.168.0.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fea7:8e6/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
          Paquetes RX:3551 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:1065 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:4494873 (4.4 MB)  TX bytes:177205 (177.2 KB)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO MTU:65536 Métrica:1
          Paquetes RX:96 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:96 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1
          Bytes RX:7488 (7.4 KB)  TX bytes:7488 (7.4 KB)

ubuntu@ubuntu:~$
```

Archivo resolv.conf

```
Ubuntu-server-TOMCAT-1A [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.0.50
ubuntu@ubuntu:~$ _
```

### -Problema-

Aquí he tenido un problema con las máquinas virtuales, no tenían acceso a internet, daba error al ejecutar el comando “sudo apt-get update”, después de andar desesperado durante un rato y comprobar varias posibilidades, he descubierto que cada vez que apago las máquinas virtuales se borra el contenido del fichero resolv.conf, he encontrado varias opciones posibles, una es cada vez que inicio las máquinas insertar su contenido o también podemos introducir el siguiente parámetro en el fichero /etc/network/interfaces la siguiente línea debajo de gateway:  
dns-nameservers 8.8.8.8.

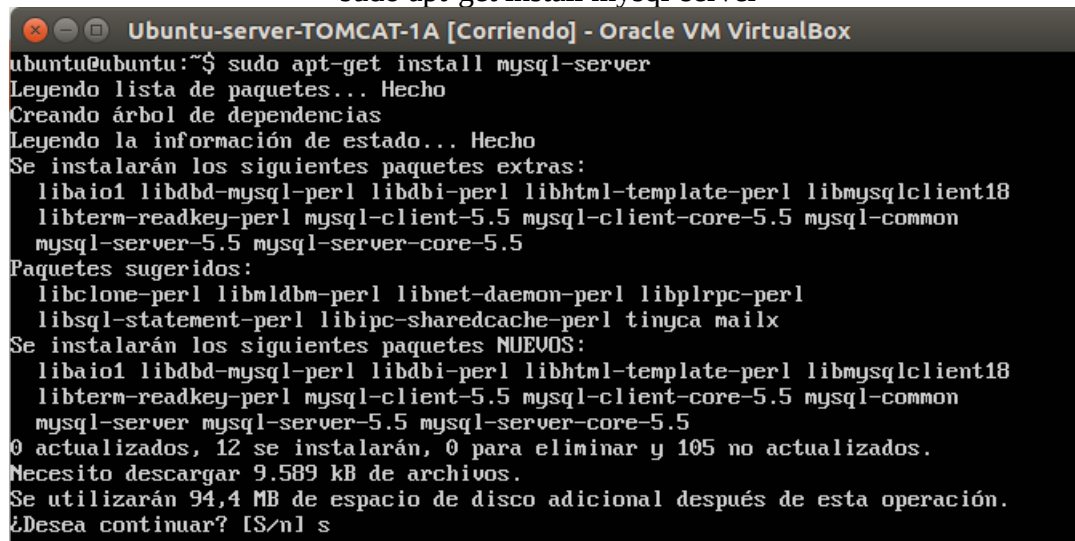
Fuente:<http://www.ubuntu-es.org/node/21309#.WRHaQrpOKbk>



## Instalando Mysql server:

Lo llevaremos a cabo mediante el siguiente comando:

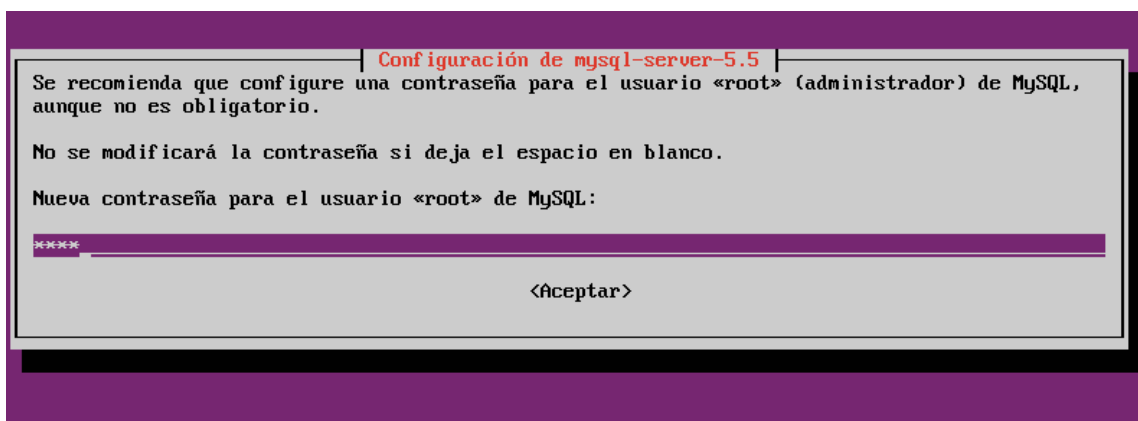
```
sudo apt-get install mysql-server
```



```
ubuntu@ubuntu:~$ sudo apt-get install mysql-server
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes extras:
  libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
  libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
  mysql-server-5.5 mysql-server-core-5.5
Paquetes sugeridos:
  libclone-perl libmldbm-perl libnet-daemon-perl libplrpc-perl
  libsql-statement-perl libipc-sharedcache-perl tinyca mailx
Se instalarán los siguientes paquetes NUEVOS:
  libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
  libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
  mysql-server mysql-server-5.5 mysql-server-core-5.5
0 actualizados, 12 se instalarán, 0 para eliminar y 105 no actualizados.
Necesito descargar 9.589 kB de archivos.
Se utilizarán 94,4 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

Seguidamente nos preguntará por el password para el usuario root:

password: 1234



## Conectando con la base de datos

Para acelerar este paso voy a utilizar mi máquina anfitrión para acceder a mysql y crear la base de datos desde Mysql Workbench, que es como estoy acostumbrado a hacerlo.

### - Problema -

Parece ser que al instalar mysql-server, no da acceso desde máquinas remotas, hay que configurar un fichero para hacer posible esta opción.

Fuente: <https://geekytheory.com/como-permitir-el-acceso-remoto-a-una-base-de-datos-mysql>

```

lc-messages-dir = /usr/share/mysql
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address          = 127.0.0.1
bind-address          = 192.168.0.51
#

mysql> GRANT ALL PRIVILEGES ON *.* TO root@'%' IDENTIFIED BY '1234';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> _

```

Creamos la conexión:

Setup New Connection

Connection Name:  Type a name for the connection

Connection Method:  Method to use to connect to the RDBMS

Parameters

Hostname:  Port:  Name or IP address of the server host - and TCP/IP port.

Username:  Name of the user to connect with.

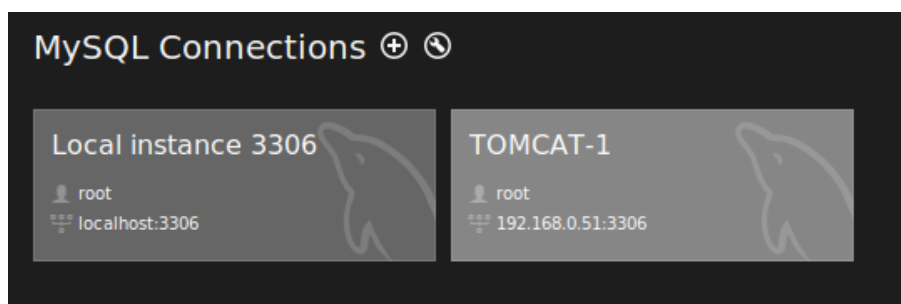
Password:   The user's password. Will be requested later if it's not set.

Default Schema:  The schema to use as default schema. Leave blank to select it later.

Testeamos la conexión para comprobar que funciona bien:

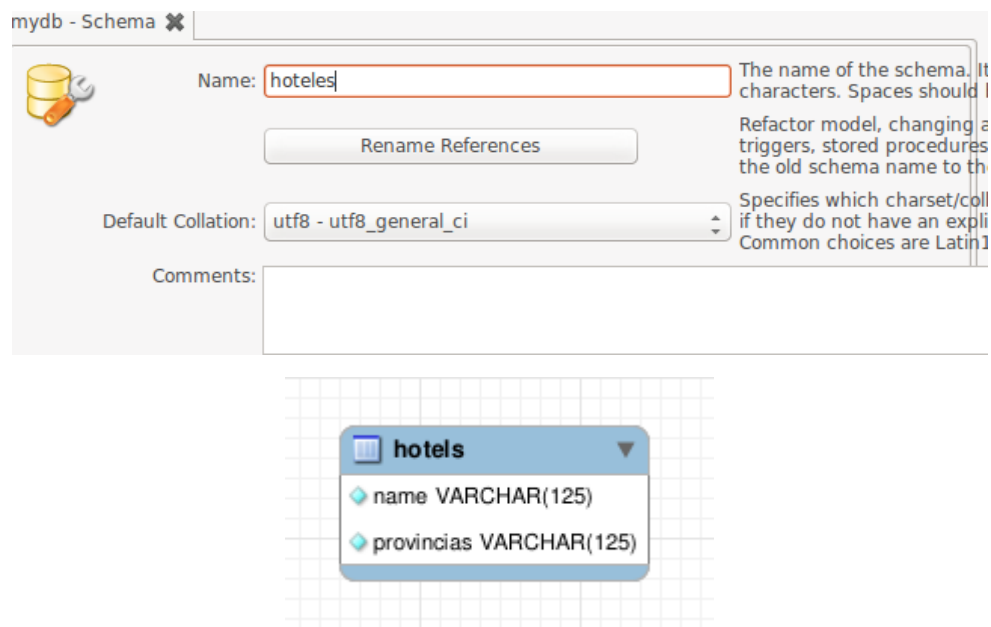


Ahora ya podemos acceder desde nuestro host principal a el servidor de base de datos de la máquina virtual TOMCAT-1.



## Creando la tabla hoteles

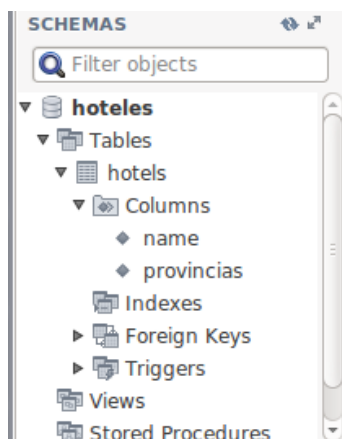
Vamos a crear una nueva tabla que constará dos campos, nombre y provincias. Para ello primero crearemos un nuevo modelo, luego exportaremos el script sql y lo ejecutaremos de manera remota en nuestra máquina de TOMCAT-1.



Una vez realizado este paso y con el script del sql de nuestra tabla, entramos en la conexión de nuestra máquina TOMCAT-1 y lanzamos el script:

❌	1	18:02:11	Apply changes to hoteles		
✅	2	18:11:01	SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_C...	0 row(s) affected	0,0018 sec
✅	3	18:11:01	SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS...	0 row(s) affected	0,0011 sec
✅	4	18:11:01	SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIO...	0 row(s) affected	0,00073 sec
✅	5	18:11:01	CREATE SCHEMA IF NOT EXISTS `hoteles` DEFAULT CHARACTE...	1 row(s) affected	0,0012 sec
✅	6	18:11:01	USE `hoteles`	0 row(s) affected	0,00059 sec
✅	7	18:11:01	CREATE TABLE IF NOT EXISTS `hoteles`.`hotels` ( `name` VA...	0 row(s) affected	0,035 sec
✅	8	18:11:01	SET SQL_MODE=@OLD_SQL_MODE	0 row(s) affected	0,00045 sec
✅	9	18:11:01	SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS	0 row(s) affected	0,00069 sec
✅	10	18:11:01	SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS	0 row(s) affected	0,00049 sec

Le damos a refrescar y deberá de aparecernos ya la nueva base de datos:



A continuación vamos a insertar algunos datos en la misma:

```
Query 2 x
1 • INSERT INTO `hoteles`
2   (`name`, `provincias`)
3   VALUES
4   ("Hospes Américo", "Alicante"),
5   ("Albahía", "Alicante"),
6   ("Alicante Golf", "Alicante"),
7   ("Eurostars Lucentum", "Alicante"),
8   ("Primus", "Valencia"),
9   ("Sorolla Palace", "Valencia"),
10  ("Casual Valencia de la Musica", "Valencia"),
11  ("The Westin Valencia", "Valencia"),
12  ("Expo Hotel", "Valencia"),
13  ("NH", "Castellon"),
14  ("Tryp Castellon Center", "Castellon"),
15  ("Hotel Luz", "Castellon"),
16  ("Hotel H2", "Castello"),
17  ("Hotel Jaime I", "Castello");
18
```

Y realizamos una consulta para comprobarlo:

Query 2 x SQL File 1\* x

```
1 • SELECT * FROM hoteles
```

Result Grid Filter Rows: Q

#	name	provincias
1	Hospes Américo	Alicante
2	Albahía	Alicante
3	Alicante Golf	Alicante
4	Eurostars Lucentum	Alicante
5	Primus	Valencia
6	Sorolla Palace	Valencia
7	Casual Valencia de la Musica	Valencia
8	The Westin Valencia	Valencia
9	Expo Hotel	Valencia
10	NH	Castellon
11	Tryp Castellon Center	Castellon
12	Hotel Luz	Castellon
13	Hotel H2	Castello
14	Hotel Jaime I	Castello

Ahora vamos a lanzar la misma consulta en nuestro servidor TOMCAT-1:

Primero indicamos que vamos a utilizar la base de datos hoteles:

```
mysql> USE hoteles
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

Luego lanzamos la SQL:

```
mysql> SELECT * FROM hotels;
+-----+-----+
| name                | provincias |
+-----+-----+
| Hospes Amérigo      | Alicante  |
| Albahía             | Alicante  |
| Alicante Golf       | Alicante  |
| Eurostars Lucentum  | Alicante  |
| Primus              | Valencia  |
| Sorolla Palace      | Valencia  |
| Casual Valencia de la Musica | Valencia  |
| The Westin Valencia | Valencia  |
| Expo Hotel          | Valencia  |
| NH                  | Castellon |
| Tryp Castellon Center | Castellon |
| Hotel Luz            | Castellon |
| Hotel H2            | Castello  |
| Hotel Jaime I       | Castello  |
+-----+-----+
14 rows in set (0.00 sec)
```

Hemos comprobado que nos proporciona el mismo resultado, por lo tanto la base de datos está correcta, ahora vamos a montar el servlet para que muestre estos mismos resultados.

### Desarrollando nuestro primer Servlet de Java.

A partir de ahora desarrollar los servlets de java vamos a utilizar netbeans, yo concretamente en su versión 7.4.

#### Nota:

Ahora vamos a crear lo que se denominan Servlets, estos utilizan lenguaje java pero se ejecutan en el servidor. También hay que destacar que para poder acceder a una base de datos desde una aplicación java, necesitaremos el conector que nos proporciona mysql, lo podemos descargar desde el siguiente enlace y agregarlo a nuestra aplicación, o bien utilizar el que nos proporciona netbeans, si queremos utilizar la última versión pincharemos en el enlace de abajo.

Descarga: <https://dev.mysql.com/downloads/connector/j/3.1.html>

Y para agregar el conector que hemos descargado seguiremos los pasos del siguiente enlace (en el mismo enlace se muestra como incluir el conector que nos proporciona netbeans).

Pasos a seguir: <http://pabletoreto.blogspot.com.es/2013/01/conectar-java-con-mysql.html>

Nosotros vamos a utilizar el que nos proporciona netbeans.

#### - Problema -

En el ordenador que trabajo habitualmente y utilizo para desarrollar los proyectos personales y de clase tenía instalada la versión 8 de Java, da algunos problemas a la hora de desarrollar proyectos para tomcat 7, así que tuve que instalar la versión 7 de Java.

Fuente: <http://www.ubuntu-guia.com/2012/04/instalar-oracle-java-7-en-ubuntu-1204.html>

Por otra parte, al tener dos versiones de Java en la misma máquina advertí que netbeans utilizaba por defecto la primera versión que encontró cuando lo instalé, esta era la versión 8 y yo necesitaba la versión 7 para desarrollar de manera correcta mis proyectos para Tomcat 7, para poder cambiar a la versión anterior de Java, tuve que seguir los pasos que se indican en el siguiente enlace:

<http://blog.jfexart.com/2012/02/seleccionar-que-version-de-java-usar.html>

Y otra vez comprobé que había otro error, por mucho que cambiáramos de versión por defecto de java había que indicarle a netbeans que debía de utilizar la versión anterior a la que el utilizaba, por lo que tras varios intentos fallidos

decidí descargarme una versión anterior de netbeans (la que utilizaba era la versión 8.1 con soporte para tomcat 8), así que me descargué la versión 7.4 con soporte para tomcat 7.

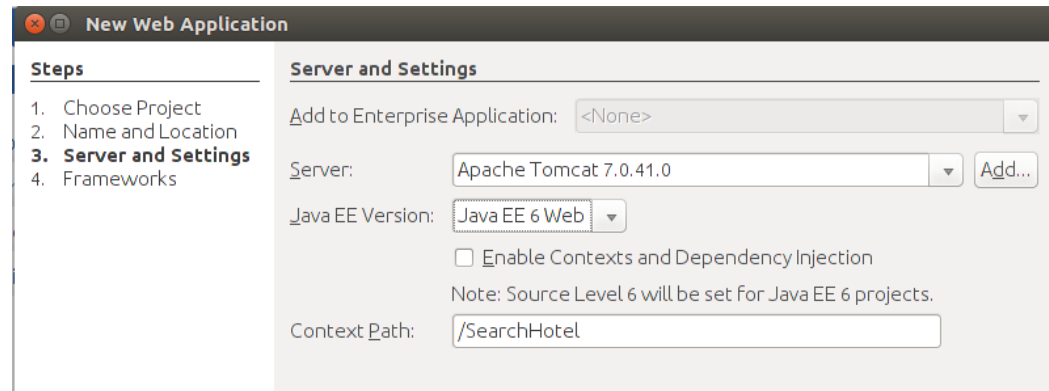
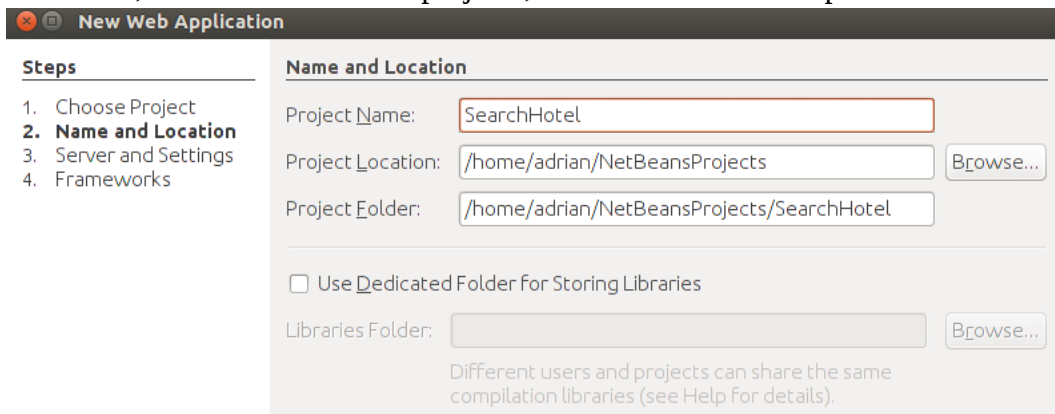
[http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/3/eclipse-jee-neon-3-linux-gtk-x86\\_64.tar.gz&mirror\\_id=96](http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/3/eclipse-jee-neon-3-linux-gtk-x86_64.tar.gz&mirror_id=96)

Y ya por último como todos los pasos anteriores parecían fáciles, resulta que no me dejaba desinstalar la versión que tenía de netbeans en mi ordenador, me daba un error porque en una carpeta oculta llamada “./netbeans”, había un fichero que se llamaba “lock”, al existir este fichero no me dejaba desinstalar netbeans, así que procedí a buscar este fichero y borrarlo de manera manual.

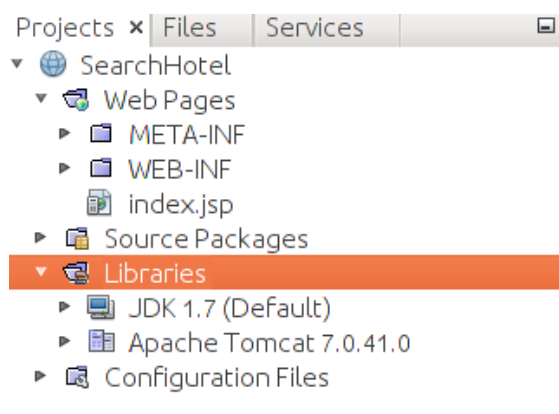
Una vez solucionados todos los problemas anteriores pude instalar la versión de Netbeans 7.4 con soporte para tomcat 7 y que utilizara por defecto la versión 7 de Java ya que me dejó elegir en el proceso de instalación que versión quería utilizar por defecto para desarrollar mis proyectos

## Creando el servlet SearchHotel

Abrimos Netbeans, seleccionamos “new project”, “Java web” → “web application”:

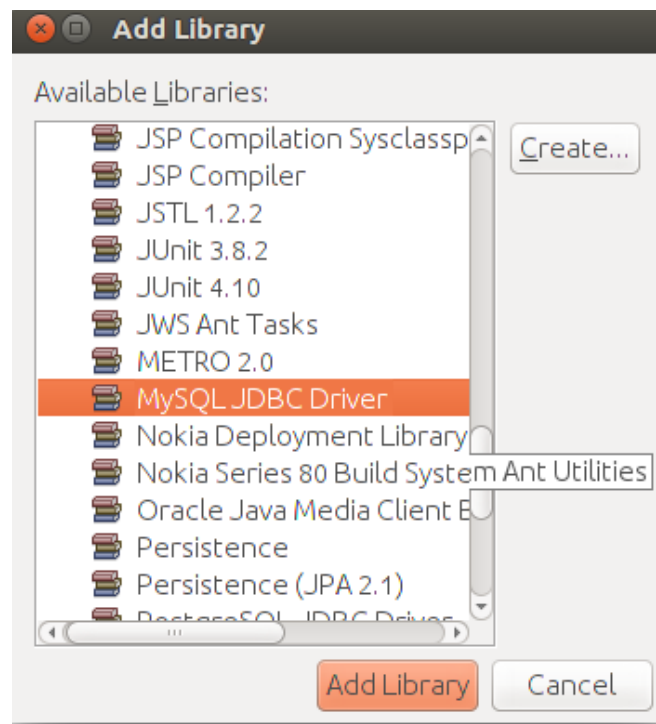


Ahora vamos a agregar el conector de mysql a nuestra aplicación.

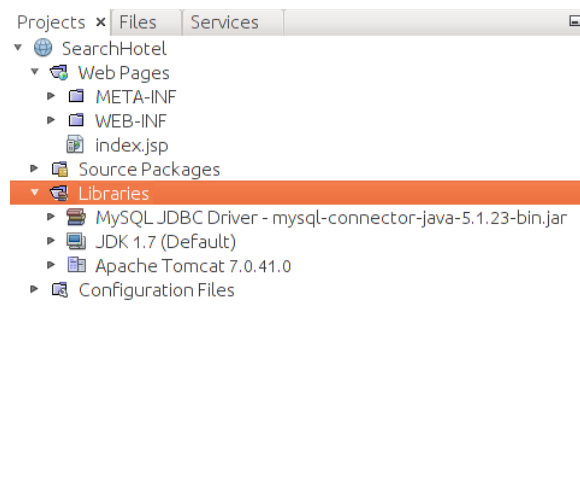


En la carpeta “Libraries”, pulsaremos con el botón derecho del ratón y seleccionaremos la opción “Agree library”.

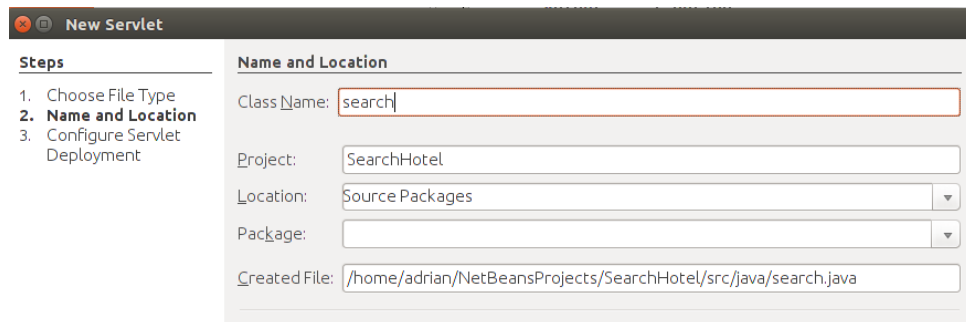
Luego buscaremos la librería que se llama “MySQL JDBC Driver” y pulsaremos sobre la opción de “Add Library”.



Esperaremos un poco que se instale la librería y ya la tendremos disponible para utilizar en nuestro proyecto.

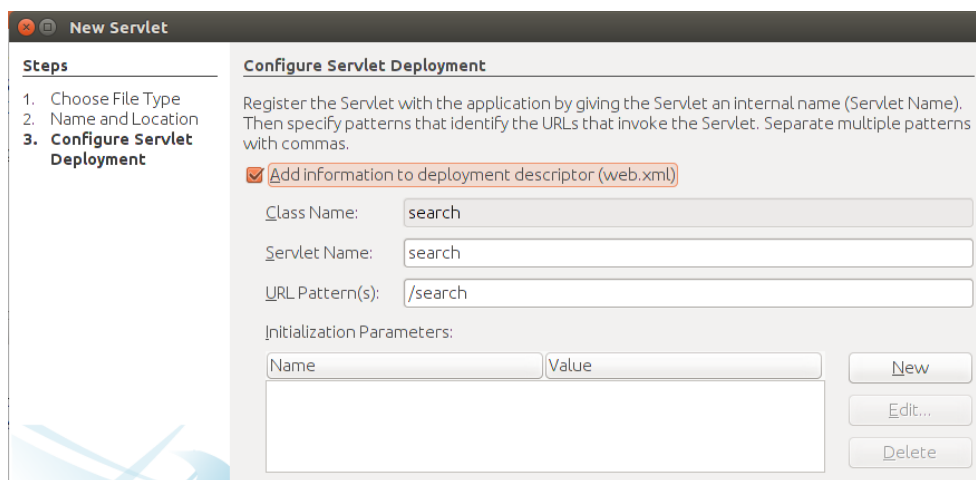


Ahora crearemos un servlet donde se va a desarrollar toda la acción, para ello seleccionamos la carpeta llamada “Source Packages” con el botón derecho del ratón y seleccionaremos la opción “New servlet”.



The screenshot shows the 'New Servlet' dialog box with the 'Name and Location' tab selected. The 'Steps' list on the left indicates the current step is '2. Name and Location'. The 'Class Name' field contains 'search'. The 'Project' field contains 'SearchHotel'. The 'Location' dropdown menu is set to 'Source Packages'. The 'Package' dropdown menu is empty. The 'Created File' field shows the path '/home/adrian/NetBeansProjects/SearchHotel/src/java/search.java'.

En el siguiente paso marcamos la opción de “Add information ...” para que el mismo agregue la información correspondiente a ruta, nombre de la clase y nombre del servlet al descriptor o fichero web.xml, de este modo no tendremos que realizarlo nosotros a mano.



The screenshot shows the 'New Servlet' dialog box with the 'Configure Servlet Deployment' tab selected. The 'Steps' list on the left indicates the current step is '3. Configure Servlet Deployment'. The 'Add information to deployment descriptor (web.xml)' checkbox is checked. The 'Class Name' field contains 'search'. The 'Servlet Name' field contains 'search'. The 'URL Pattern(s)' field contains '/search'. The 'Initialization Parameters' section is empty. There are buttons for 'New', 'Edit...', and 'Delete'.

A continuación vamos a agregar la información necesaria en nuestro servlet.  
Crearemos las variables que vamos a utilizar:

```
public class search extends HttpServlet {  
    /*  
        Zona de declaración de variables que vamos a utilizar en esta clase  
    */  
    Connection conn = null;  
    Statement stmt;
```



Creamos una instancia de la conexión:

```
@Override
public void init(ServletConfig config) {
    try {
        /*
         * Creamos la instancia del conector MySQL
         */
        Class.forName("com.mysql.jdbc.Driver").newInstance();
    }
}
```

Creamos la conexión con la base de datos:

```
/*
 * Creamos la conexión, incluyendo la ip de la maquina, nombre de
 * la base de datos, el usuario y el password.
 * la ip de la máquina es 127.0.0.1 porque es la misma donde se
 * está ejecutando el servlet donde esta la base de datos.
 */

conn = (Connection) DriverManager.getConnection("jdbc:mysql://127.0.0.1/hoteles",
        "root", "1234");
stmt = (Statement) conn.createStatement();
```

A continuación modificamos la información que se va a mostrar en la web.

```
try {
    /*
     * A continuación incluimos información de la cabecera de la web
     */
    out.println("<html>");
    out.println("<head>");
    out.println("<title>Hotels list</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Hotels:</h1>");
    /*
     * Y la información la mostramos en una tabla
     */
    out.println("<table>");
```

Ahora realizamos los pasos que tocan para ejecutar la consulta a la base de datos:

```
/*
 * Ahora realizamos la consulta a la base de datos
 */
String query = null;
/*
 * Muestre todos los datos de la tabla hotels
 */
query = "SELECT * FROM hotels;";
ResultSet resultSet = null;

try {
    synchronized (stmt) {
        /*
         * Ejecuta la consulta y el resultado lo guardas en stmt
         */
        resultSet = stmt.executeQuery(query);
    }
}
```

Y el resultado lo mostraremos a de la siguiente manera:

```
/*
    Mientras existan resultados haz ...
*/
while (resultSet.next()) {
    /*
        Me creas una fila y cada campo de la base de datos me lo
        muestras en una celda
    */
    out.println("<tr>");
    out.println("<td>" + resultSet.getString(1) + "</td>");
    out.println("<td>" + resultSet.getString(2) + "</td>");
}
out.println("</tr>");
```

Si ocurre alguna excepción la procesamos de la siguiente manera, mostrándola por pantalla para poder orientarnos y de esa manera poder corregirla, finalmente como no queremos hacer nada más que listar el contenido de la consulta cerraremos la conexión una vez mostrados todos los valores de la base de datos:

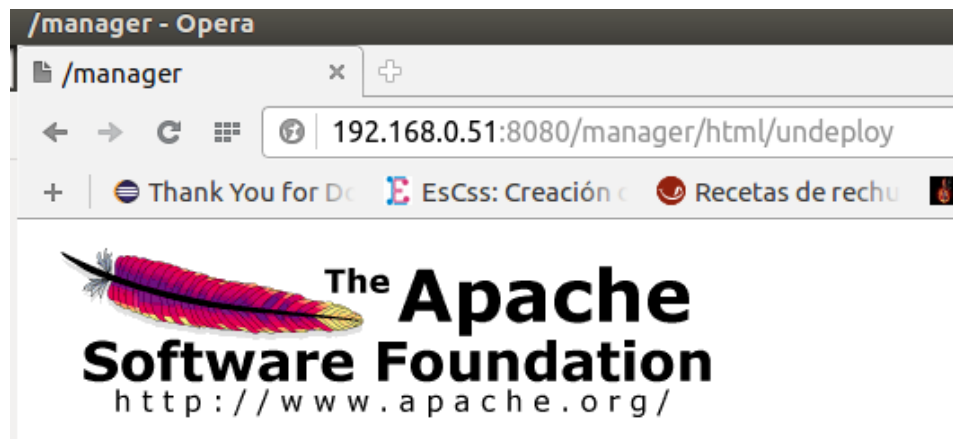
```
/*
    Si algo ha fallado en el proceso muestrame el error por pantalla
*/
} catch (SQLException ex) {
    out.println("Excepcion: " + ex);
} finally {
    if (resultSet != null) {
        try {
            resultSet.close();
        } catch (SQLException ex) {
            //Logger.getLogger(db.class.getName()).log(Level.SEVERE,
        }
    }
}
```

Por último cerraremos las etiquetas HTML y el programa en si:

```
    out.println("</table>");
    out.println("</body>");
    out.println("</html>");
} finally {
    out.close();
}
```

A continuación no falta limpiar y compilar el proyecto para poder tenerlo con extensión .war y de esa manera subirlo a nuestro servidor Tomcat.

Ahora ingresaremos la ip de nuestro servidor tomcat, en este caso de momento lo haremos desde la máquina anfitrión, en este caso la ip de nuestro primer servidor tomcat es la 192.168.0.51 y el puerto es el 8080



Deberemos buscar la sección donde dice Desplegar - Archivo War desplegable:

**Desplegar**

Desplegar directorio o archivo WAR localizado en servidor

Trayectoria de Contexto (opcional):

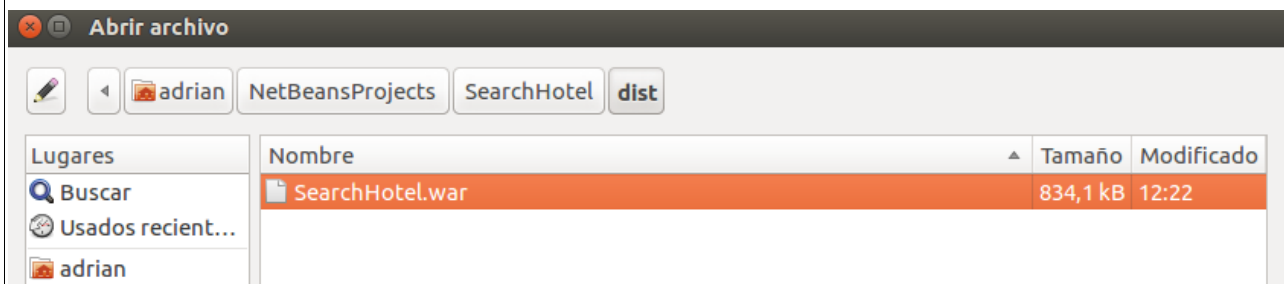
URL de archivo de Configuración XML:

URL de WAR o Directorio:

**Archivo WAR a desplegar**

Seleccione archivo WAR a cargar  Ningún archivo seleccionado

Seleccionaremos nuestro archivo con extensión war, en mi caso está en la carpeta:



Una vez subido tendremos que seleccionar la opción de “Desplegar”:

Seleccione archivo WAR a cargar  SearchHotel.war

Una vez realizado este paso podemos observar que nos muestra en la lista de aplicaciones:

Aplicaciones			
Trayectoria	Versión	Nombre a Mostrar	Ejecutándose
/	Ninguno especificado		true
/SearchHotel	Ninguno especificado		true
/docs	Ninguno especificado	Tomcat Documentation	true
/examples	Ninguno especificado	Servlet and JSP Examples	true
/host-manager	Ninguno especificado	Tomcat Host Manager Application	true
/manager	Ninguno especificado	Tomcat Manager Application	true

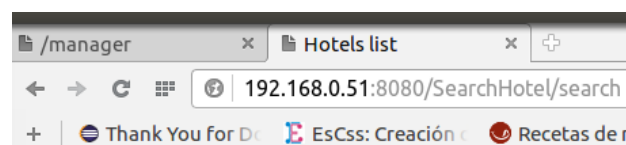
Ahora ya tenemos nuestro servlet en marcha, solo debemos de ingresar la siguiente dirección en nuestro navegador:

192.168.0.51:8080/SearchHotel/search

192.168.0.51:8080 – Indica la dirección Ip de la maquina virtual donde se está ejecutando Tomcat y el puerto que utiliza

/SearchHotel/search – indica el nombre de la aplicación y el nombre de el servlet que hemos creado para que muestre la lista de hoteles que se encuentra almacenada en la base de datos de la misma máquina virtual.

El resultado es el siguiente:



## Hotels:

Hospes Amérigo	Alicante
Albahía	Alicante
Alicante Golf	Alicante
Eurostars Lucentum	Alicante
Primus	Valencia
Sorolla Palace	Valencia
Casual Valencia de la Musica	Valencia
The Westin Valencia	Valencia
Expo Hotel	Valencia
NH	Castellon
Tryp Castellon Center	Castellon
Hotel Luz	Castellon
Hotel H2	Castello
Hotel Jaime I	Castello

## Modificando máquina con servidor web APACHE, parte 1.

### - Atención -

Este apartado es el más tedioso y difícil de entender a mi parecer, se trata de que nuestro servidor apache redireccione a otras máquinas virtuales que ofrecen aplicaciones hechas en tomcat, pero claro, se podría realizar de manera fácil metiendo la ip en la etiqueta href del enlace, pero no se trata de eso, así que lo haremos como indica en los apuntes que nos han ofrecido en clase, concretamente esta información se encuentra a partir de la página 212.

Se trata de que desde el cliente (windows), podamos acceder a estas aplicaciones que hemos creado a partir de el dominio que hemos configurado con anterioridad. Ahora debemos de modificar un poco nuestro servidor Apache para que efectúe el redireccionamiento a esta aplicación.

Para ello nosotros vamos a utilizar el mod de apache proxy y el proxy http, estos módulos los hemos activado ya con anterioridad en nuestro servidor apache.

Hay que realizar una pequeña modificación en el fichero de configuración de proxy.conf debido a que sobre este fichero se apoya el proxy\_http.

Ahora debemos de editar el fichero de nuestro virtualhost.

```
ServerName www.hoteles.es
ServerAlias www.hoteles.es

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html/hoteles

#configuracion para redireccionamiento para searchHotel
ProxyPass /Tomcat-1/SearchHotels/buscaHoteles http://192.168.0.51:8080/SearchHotel/search
ProxyPassReverse /Tomcat-1/SearchHotels/buscaHoteles http://192.168.0.51:8080/SearchHotel/s$

<Directory /var/www/html/hoteles>
    DirectoryIndex index.html
    Options FollowSymLinks
    Require all granted
</Directory>

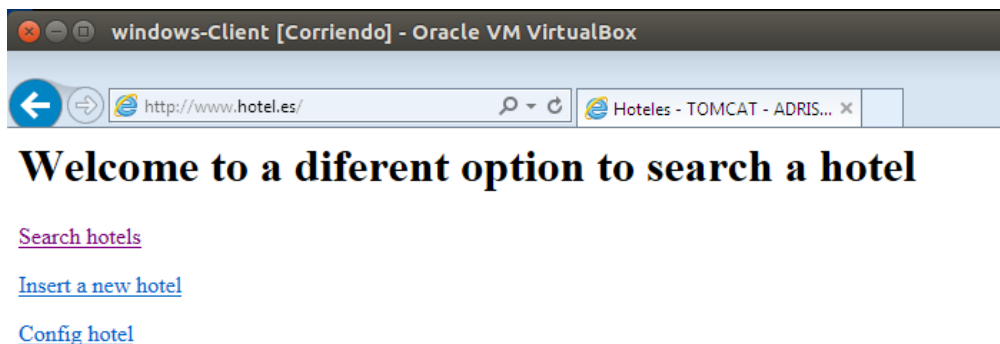
<Location /Tomcat-1/Searchhotels/buscaHoteles>
    Order allow,deny
    Allow from all
</Location>
```

Primero le indicamos la configuración de proxypass para gestionar las peticiones recibidas sobre buscaHoteles, deberá de redireccionar a la máquina donde se está ejecutando el servlet de tomcat, osea la que tiene la ip 192.168.0.51, en el puerto 8080.

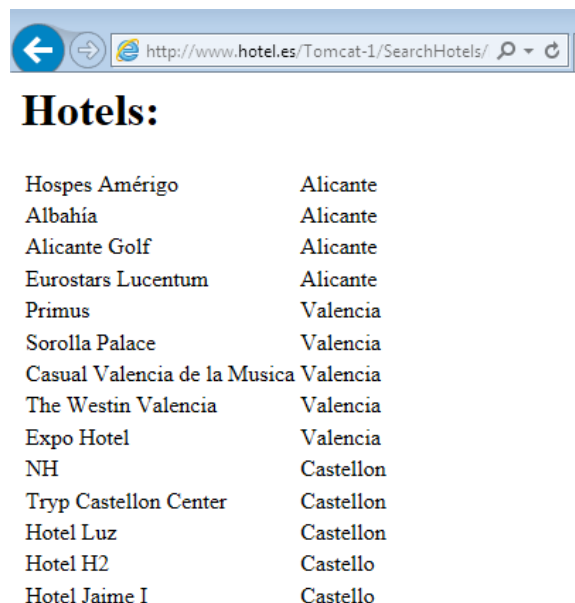
A continuación debemos reiniciar el servicio de apache:

```
ubuntu@ubuntu:/etc/apache2/sites-enabled$ sudo service apache2 restart
[sudo] password for ubuntu:
ubuntu@ubuntu:/etc/apache2/sites-enabled$
```

Comprobación:



Ahora ya tenemos el redireccionamiento completado, si vamos al cliente (windows) y buscamos el dominio [www.hoteles.es](http://www.hoteles.es) y nos aparece lo siguiente: Y al seleccionar la primera opción de search, nos redirecciona a la máquina virtual donde Tomcat está ofreciendo la lista de hoteles de nuestra base de datos.



### Configuración de red:

```

Ubuntu-server-TOMCAT-2A [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:~$ ifconfig -a
eth0      Link encap:Ethernet  direcciónHW 08:00:27:2b:0a:98
          Direc. inet:192.168.0.52  Difus.:192.168.0.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe2b:a98/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
          Paquetes RX:8 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:6 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:800 (800.0 B)  TX bytes:508 (508.0 B)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO MTU:65536 Métrica:1
          Paquetes RX:32 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:32 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1
          Bytes RX:2368 (2.3 KB)  TX bytes:2368 (2.3 KB)

ubuntu@ubuntu:~$

```

Esta vez se trata de ofrecer al cliente una página donde existan dos campos en los cuales podamos insertar un nuevo hotel y su provincia, pero solo podrá entrar el usuario administrador.

### Configurando Tomcat:

Creando el usuario con rol “databaseadmin”.

#### - Anotación -

Para llevar a cabo el proceso de crear un usuario y asignarle un rol para poder utilizar el método de autenticación, me he basado en los apuntes que nos han facilitado en clase, concretamente se encuentran a partir de la página 179.

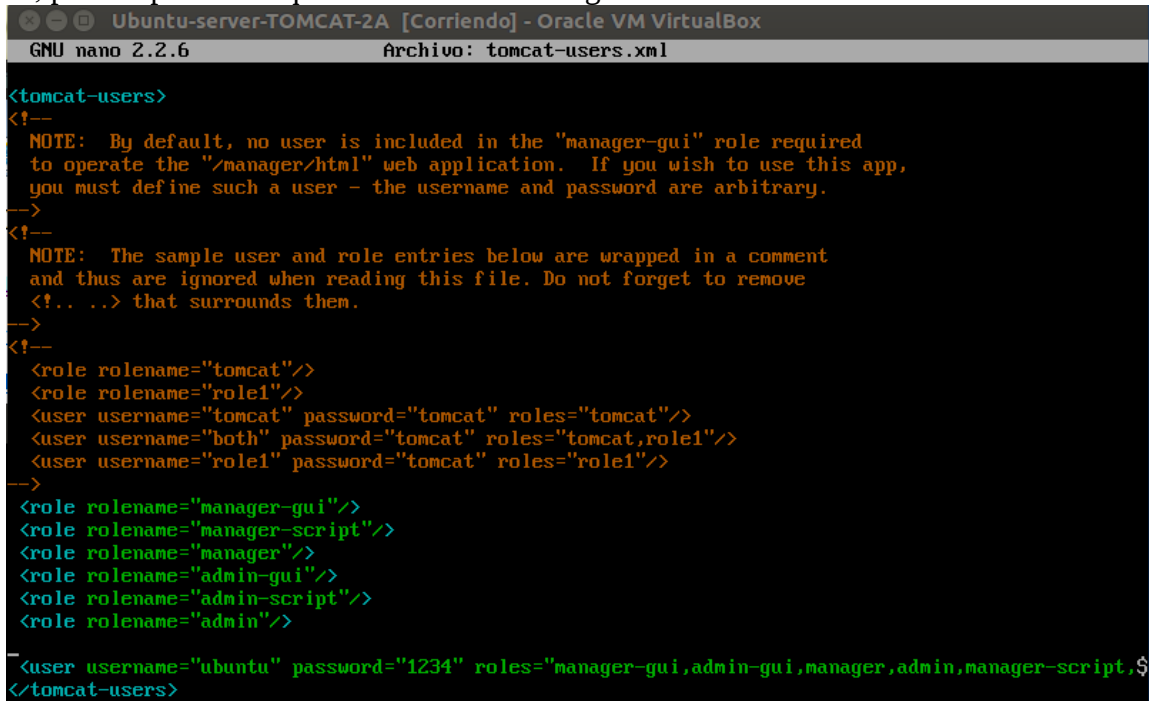
Para ello debemos de ir a los ficheros de configuración de nuestra máquina donde vamos a desplegar la aplicación tomcat (en este caso es la segunda máquina virtual con Tomcat, con la Ip 192.168.0.52), estos se encuentran en: /etc/tomcat/users.xml

```

Ubuntu-server-TOMCAT-2A [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:/etc/tomcat7$ ls -la
total 208
drwxr-xr-x  4 root root    4096 may  2 14:47 .
drwxr-xr-x 106 root root    4096 may 12 15:51 ..
drwxrwxr-x  3 root tomcat7  4096 may  2 14:47 Catalina
-rw-r----- 1 root tomcat7  6426 abr  5 20:27 catalina.properties
-rw-r----- 1 root tomcat7  1394 ene 25  2014 context.xml
-rw-r----- 1 root tomcat7  2370 mar 31 13:34 logging.properties
drwxr-xr-x  2 root tomcat7  4096 may  2 14:51 policy.d
-rw-r----- 1 root tomcat7  6500 abr  5 20:27 server.xml
-rw-r----- 1 root tomcat7  1831 may  2 15:00 tomcat-users.xml
-rw-r----- 1 root tomcat7 164104 abr  5 20:27 web.xml
ubuntu@ubuntu:/etc/tomcat7$ _

```

Observamos que por defecto el ya ha creado el usuario que utilizamos para gestionar nuestro Tomcat, pero se puede ver que la contraseña no la guarda cifrada.

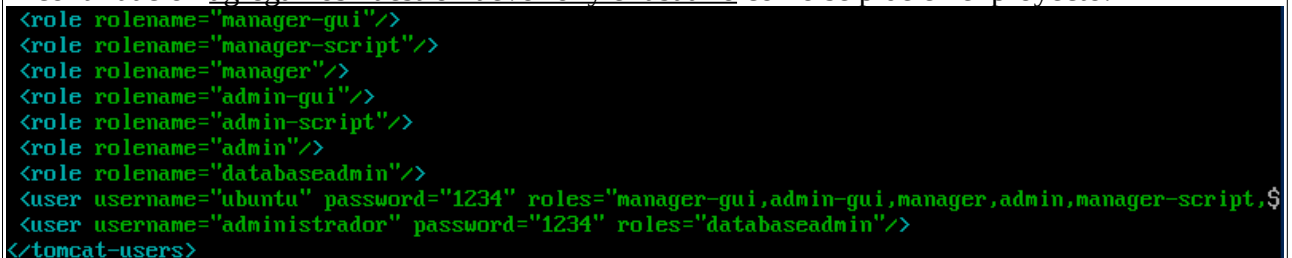


```
GNU nano 2.2.6 Archivo: tomcat-users.xml

<tomcat-users>
<!--
  NOTE: By default, no user is included in the "manager-gui" role required
  to operate the "/manager/html" web application.  If you wish to use this app,
  you must define such a user - the username and password are arbitrary.
-->
<!--
  NOTE: The sample user and role entries below are wrapped in a comment
  and thus are ignored when reading this file. Do not forget to remove
  <!-- ... --> that surrounds them.
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="both" password="tomcat" roles="tomcat,role1"/>
  <user username="role1" password="tomcat" roles="role1"/>
-->
  <role rolename="manager-gui"/>
  <role rolename="manager-script"/>
  <role rolename="manager"/>
  <role rolename="admin-gui"/>
  <role rolename="admin-script"/>
  <role rolename="admin"/>

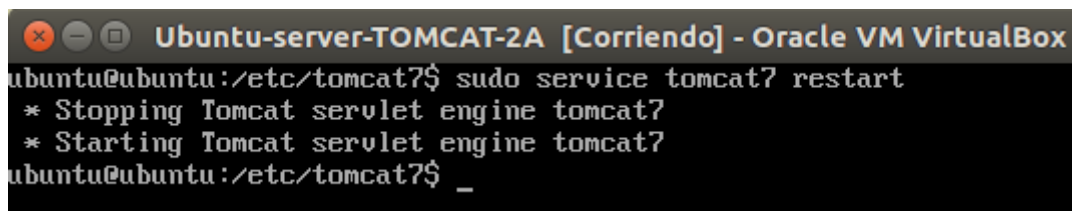
  <user username="ubuntu" password="1234" roles="manager-gui,admin-gui,manager,admin,manager-script,$
</tomcat-users>
```

A continuación agregamos nuestro nuevo rol y el usuario como se pide en el proyecto:



```
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="manager"/>
<role rolename="admin-gui"/>
<role rolename="admin-script"/>
<role rolename="admin"/>
<role rolename="databaseadmin"/>
<user username="ubuntu" password="1234" roles="manager-gui,admin-gui,manager,admin,manager-script,$
<user username="administrador" password="1234" roles="databaseadmin"/>
</tomcat-users>
```

Ahora debemos reiniciar el servicio de Tomcat para que se efectúen los cambios que acabamos de hacer:



```
ubuntu@ubuntu:/etc/tomcat7$ sudo service tomcat7 restart
* Stopping Tomcat servlet engine tomcat7
* Starting Tomcat servlet engine tomcat7
ubuntu@ubuntu:/etc/tomcat7$ _
```

#### - Anotación -

He estado un rato volviéndome loco debido a que siguiendo los apuntes facilitados en clase, el siguiente paso era modificar el fichero web.xml que se encuentra dentro de la carpeta web, pero resulta que este fichero no se crea si no creamos primero un servlet. Esto me ha ocurrido porque pensaba que podría realizar todas las acciones que se piden desde la página inicial.

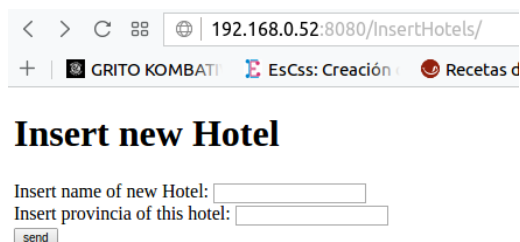
Como en esta parte voy un poco perdido, vamos a ir paso por paso, indicando todos los errores que vaya cometiendo, de esta manera cuando se lea este manual quedará mucho mas detallado y no se cometerán los mismo errores.



Hemos introducido un pequeño formulario en el fichero index.jsp, vamos a compilar el proyecto ya subirlo para de esta manera ir testeando conforme vamos avanzando.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Tomcat-2 AdriSC</title>
</head>
<body>
<h1>Insert new Hotel</h1>
<form action="action" method="POST" name="new_hotel">
<label for="new_hotel">Insert name of new Hotel:</label>
<input name="new_hotel">
<br />
<label for="new_hotel_provincia">Insert provincia of this hotel:</label>
<input name="new_hotel_provincia">
<br />
<input type="submit" value="send">
</form>
</body>
</html>
```

Observamos que al desplegar el fichero .war, si accedemos a el, nos muestra el formulario tal cual:



#### - Anotación -

En la máquina virtual anterior como toda la acción la desarrollábamos desde el servlet, conexión a la base de datos, mostrar la información del a consulta, etc, el redireccionamiento lo hacíamos directamente al servlet y no utilizábamos el archivo index.jsp para nada, en este caso, en el fichero index.jsp, presentaremos al usuario el formulario y trataremos los datos desde el servlet. Esta información no la he podido encontrar en los apuntes, pero he buscado por internet y he ido recopilando información que me va a servir de utilidad.

Fuente: <http://www.forosdelweb.com/f45/aporte-registro-login-usuarios-con-jsp-servlets-mysql-930805/>

En este caso en esta página había un ejemplo de como enviarse información de los ficheros con extensión jsp a los ficheros con extensión java,

Realizada la aclaración en la última anotación, procedemos a cambiar la información que había en el método action del formulario para que apunte a nuestra clase java y vamos a realizar una pequeña prueba, se trata de pasar los datos del fichero jsp y mostrarlos desde la clase java:

```
<body>
<h1>Insert new Hotel</h1>
<form action="InsertHotel" method="POST" name="new_hotel">
<label for="new_hotel">Insert name of new Hotel:</label>
<input name="new_hotel">
<br />
<label for="new_hotel_provincia">Insert provincia of this hotel:</label>
<input name="new_hotel_provincia">
<br />
<input type="submit" value="send">
</form>
</body>
```

```

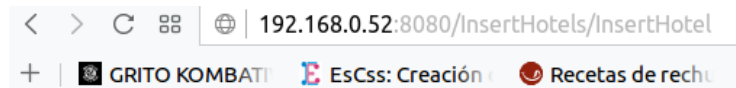
protected void processRequest(HttpServletRequest request, HttpServletResponse
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    String nameHotel = request.getParameter("new_hotel");
    try {
        /* TODO output your page here. You may use following sample code
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet InsertHotel</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Prueba de nuevo hotel " + nameHotel + "</h1>");
        out.println("</body>");
        out.println("</html>");
    } finally {
        out.close();
    }
}

```

## Insert new Hotel

Insert name of new Hotel:

Insert provincia of this hotel:



## Prueba de nuevo hotelprueba

Como ya hemos comprobado que si que funciona, vamos a desarrollar lo que queda, realizaremos la conexión a la base de datos y crearemos la sentencia insert a la base de datos con los datos que se han insertado, esta vez la consulta la agregaremos dentro de un método sobrescrito llamado doPost, vamos a describirlo paso a paso:

Creamos la conexión con la base de datos, en el método sobrescrito init (Config config), la información sobre la base de datos hará referencia a la ip de la máquina donde está la base de datos, "192.168.0.51", el puerto será el "3306" y la base de datos será hoteles, usuario "root" y el pass será "1234":

```

@Override
public void init(ServletConfig config) throws ServletException {
    try{
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        conn = (Connection) DriverManager.getConnection("jdbc:mysql://192.168.0.51:3306/hoteles",
        stmt = (Statement) conn.createStatement();
    }catch (ClassNotFoundException ex) {
        Logger.getLogger(InsertHotel.class.getName()).log(Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        Logger.getLogger(InsertHotel.class.getName()).log(Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        Logger.getLogger(InsertHotel.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(InsertHotel.class.getName()).log(Level.SEVERE, null, ex);
    }
}
}

```

Ahora en el método doPost trataremos la información que recibimos de index.jsp y haremos la consulta de tipo insert:

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    //processRequest(request, response);
    String nameNewHotel = request.getParameter("new_hotel");
    String provinciaNewhotel = request.getParameter("new_hotel_provincia");
    String query;
    try{
        query = "INSERT INTO hotels (name, provincias) values ('" + nameNewHotel + "','" + provinciaNewhotel + "')";
        synchronized(stmt){
            stmt.executeUpdate(query);
        }
    }catch (SQLException ex){
    }
}
```

A continuación configuraremos el acceso al usuario que hemos creado antes, esto lo haremos en el fichero web.xml:

```
<servlet>
    <servlet-name>InsertHotel</servlet-name>
    <servlet-class>InsertHotel</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>InsertHotel</servlet-name>
    <url-pattern>/InsertHotel</url-pattern>
</servlet-mapping>

<!-- esto es lo que añadiremos para la seguridad básica -->
<!-- Primero una restricción de seguridad -->
<security-constraint>
    <web-resource-collection>
        <web-resource-name> El * significa que pedimos autenticación para toda la aplicación</web-resource-name>
        <url-pattern>/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
    </web-resource-collection>

    <auth-constraint>
        <role-name>databaseadmin</role-name>
    </auth-constraint>

    <user-data-constraint>
        <!-- Hay tres tipos CONFIDENTIAL, INTEGRAL y NONE -->
        <transport-guarantee>NONE</transport-guarantee>
    </user-data-constraint>
</security-constraint>

<login-config>
    <auth-method>DIGEST</auth-method>
</login-config>
<!-- HASTA AQUÍ -->
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
</web-app>
```

## Modificando máquina con servidor web APACHE, parte 2.

A continuación vamos a incluir la información de redireccionamiento en la máquina donde está el servidor apache:

```
#configuracion para redireccionamiento para insertNewHotel
ProxyPass /Tomcat-2/NewHotel http://192.168.0.52:8080/InsertHotels
ProxyPassReverse /Tomcat-2/NewHotel http://192.168.0.52:8080/Inserthotels
```

```
<Location /Tomcat-2/NewHotel>
    Order allow,deny
    Allow from all
</Location>
```

Ahora ya solo queda subir la nueva aplicación a la máquina que contiene el servidor Tomcat-2 y probar que todo funciona de manera correcta.

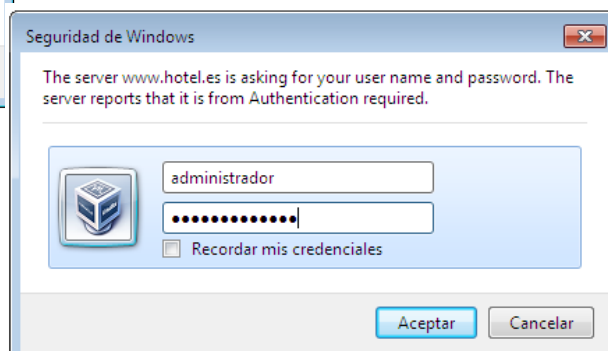
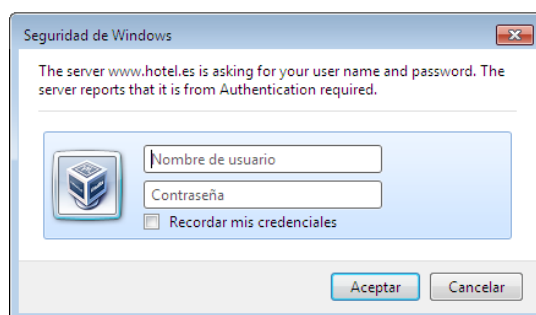
### - Error detectado -

Hemos comprobado que a lo hora de autenticarse con el método DIGEST no nos deja acceder, realizando pruebas y con el método basic si que nos deja autenticarnos con normalidad.

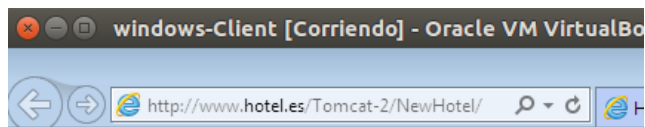
En los apuntes hacía referencia a que a lo mejor daba problemas al no tener el protocolo https activado, más tarde investigaremos sobre este error. De momento para prbar que funciona de manera correcta lo dejaremos en basic.

```
<login-config>
  <auth-method>BASIC</auth-method>
</login-config>
```

### Comprobación:



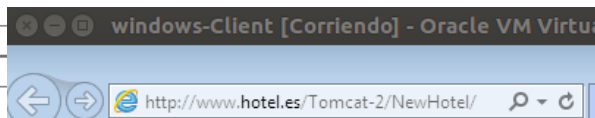
Al introducir el usuario que hemos creado y su contraseña, comprobamos que nos muestra el formulario que habíamos creado en el index.jsp de nuestro servlet, a continuación vamos a comprobar que podemos crear un nuevo hotel con normalidad:



## Insert new Hotel

Insert name of new Hotel:

Insert provincia of this hotel:



## Insert new Hotel

Insert name of new Hotel:

Insert provincia of this hotel:

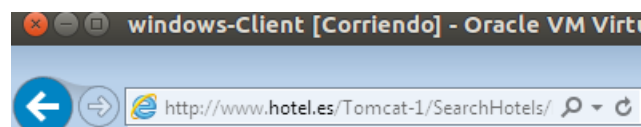
A continuación vamos a consultar la base de datos a través de mysql WorkBench y accediendo a la primera aplicación de tomcat para comprobar que se ha almacenado en la base de datos de manera correcta:

Query 1 ✕

SELECT \* FROM hotels

Result Grid

#	name	provincias
5	Primus	Valencia
6	Sorolla Palace	Valencia
7	Casual Valencia de la Musica	Valencia
8	The Westin Valencia	Valencia
9	Expo Hotel	Valencia
10	NH	Castellon
11	Tryp Castellon Center	Castellon
12	Hotel Luz	Castellon
13	Hotel H2	Castello
14	Hotel Jaime I	Castello
15	adri	prueba
16	adri	prueba



## Hotels:

Hospes Amérigo	Alicante
Albahía	Alicante
Alicante Golf	Alicante
Eurostars Lucentum	Alicante
Primus	Valencia
Sorolla Palace	Valencia
Casual Valencia de la Musica	Valencia
The Westin Valencia	Valencia
Expo Hotel	Valencia
NH	Castellon
Tryp Castellon Center	Castellon
Hotel Luz	Castellon
Hotel H2	Castello
Hotel Jaime I	Castello
adri	prueba
adri	prueba

## Servidor de aplicaciones Tomcat-3

### Configuración de red:

```
Ubuntu-server-TOMCAT-3A [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:~$ ifconfig -a
eth0      Link encap:Ethernet  direcciónHW 08:00:27:a7:08:e6
          Direc. inet:192.168.0.53  Difus.:192.168.0.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fea7:8e6/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
          Paquetes RX:18 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:6 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:1754 (1.7 KB)  TX bytes:508 (508.0 B)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Amfitrión
          ACTIVO BUCLE FUNCIONANDO MTU:65536 Métrica:1
          Paquetes RX:32 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:32 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1
          Bytes RX:2368 (2.3 KB)  TX bytes:2368 (2.3 KB)

ubuntu@ubuntu:~$
```

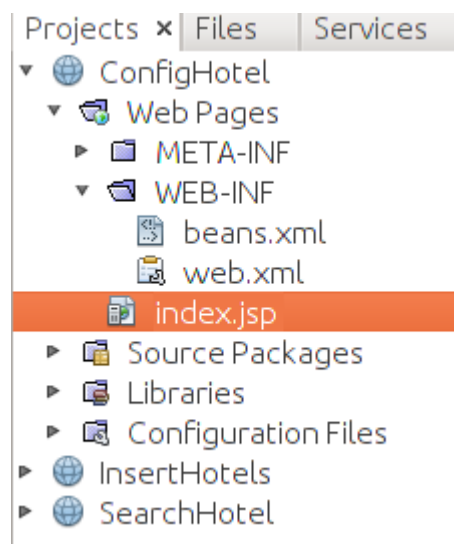
### Creando el servlet:

A continuación vamos a crear un servlet el cual al acceder a el podremos introducir información en unos campos para configurar nuestro hotel y al pulsar sobre el botón “enviar” nos mostrarán estos datos y el número de veces que nos hemos conectado, esta parte se trata de crear un nuevo servlet y utilizar las sesiones, esta por su parte tendrá una duración de dos horas.

#### - Anotación -

Para desarrollar esta parte del ejercicio he utilizado casi todo lo que se nos proporcionaba en los apuntes, concretamente en el pdf JavaEE-servlet a partir de la página 100. En los otros apuntes no encontraba nada relacionado con las sesiones y la información que encontraba por internet era un tanto difusa.

Crearemos una nueva aplicación y la llamaremos esta vez ConfigHotel y en el index.jsp incluiremos dos formularios:



El primer formulario servirá para que el usuario introduzca los valores que nos piden en el proyecto:

```
<h1>Config hotel information</h1>
<form action="ConfigHotel" method="POST" name="config_hotel">
  <label for="name_hotel">Insert name of Hotel:</label>
  <input name="name_hotel">
  <br />
  <label for="service_hotel">Insert service:</label>
  <input name="service_hotel">
  <br />
  <label for="discount_hotel">Insert discount:</label>
  <input name="discount_hotel">
  <br />
  <label for="persons_hotel">Insert persons:</label>
  <input name="persons_hotel">
  <br />
  <input name="accion" type="submit" value="send">
</form>
```

El segundo formulario tendrá un botón para cerrar la sesión:

```
<!--
  EL trozo que va a continuación lo he copiado tal cual de los apuntes
-->
<form method="get" action="ConfigHotel">
  <input name="accion" value="invalidar" type="hidden">
  <button>Invalidar sesion</button>
</form>
```

Ahora crearemos el servlet donde se mostrarán los datos introducidos por el usuario, además de un contador de veces que se conecta.

Primero crearemos un contador que mostrará las veces que se ejecuta el servlet y por lo tanto las veces que se conecta un mismo usuario.

```
int countTimesConnect = 1;
```

Ahora recogeremos los valores que recibimos desde el index.jsp y los incluiremos en variables.

```
String valor = request.getParameter("valor");
String action = request.getParameter("accion");

String nameHotel = request.getParameter("name_hotel");
String service = request.getParameter("service_hotel");
String discount = request.getParameter("discount_hotel");
String persons = request.getParameter("persons_hotel");
```

A continuación procesamos la información que recibimos, primero crearemos un objeto de tipo session, después comprobamos el valor de la variable que hemos recibido con nombre de “action”, si esta es “invalidar”, la sesión se cerrará y no se ejecutará nada más, por el contrario se guardarán los datos que previamente hemos guardado en las variables dentro de la sesión, en la parte que dice session.setAttribute(“nombre\_del\_atributo”, información). Acto seguido incluiremos toda la información dentro de nombresDeAtributos y mientras dentro exista información lo mostrará en forma de lista. Por último mostraremos un enlace que nos llevará al a página de index.jsp.

```
HttpSession session = request.getSession();
if (action.equals("invalidar")) {
    session.invalidate();
    out.println("<h1>Sesion invalidada:</h1>");
} else {
    session.setAttribute("name_hotel", nameHotel);
    session.setAttribute("service_hotel", service);
    session.setAttribute("discount_hotel", discount);
    session.setAttribute("persons_hotel", persons);
    out.println("<ul>");
    Enumeration<String> nombresDeAtributos = session.getAttributeNames();
    out.println("<p>Number connections :"+ (countTimesConnect++) +"</p>");
    while (nombresDeAtributos.hasMoreElements()) {
        String atributo = nombresDeAtributos.nextElement();
        out.println("<li><b>" + atributo + ": </b>" + session.getAttribute(atributo) +
    }
    out.println("</ul>");
}
out.println("<a href=/ConfigHotel/index.jsp>" + "Return</a><br/>");
//out.println(request.getQueryString());
out.println("</body>");
out.println("</html>");
finally {
    out.close();
}
```

Por último hay que configurar para que la sesión exista durante dos horas para los usuarios, esto lo haremos dentro del fichero web.xml.

```
<session-config>
    <session-timeout>
        120
    </session-timeout>
</session-config>
```

Realizados estos pasos vamos a proceder a subir nuestro servlet a la máquina tomcat-3 con ip 192.168.0.53.

/ConfigHotel	Ninguno especificado		true	0	<div> <div>Arrancar</div> <div>Parar</div> <div>Recargar</div> <div>Replegar</div> </div>
					Expirar sesiones   sin trabajar ≥ 120 minutos

Observamos que en la parte que pone “Expirar sesiones” cambia de 30 que tiene por defecto a 120, esto es lo que hemos modificado nosotros en el fichero web.xml.



### Modificando máquina con servidor web APACHE, parte 3.

Para esto vamos a nuestro fichero de virtualhost que hemos creado a nombre de hoteles.es.conf e incluimos las siguientes líneas:

```
#configuracion para redireccionamiento para configHotel
ProxyPass /Tomcat-3/ConfigHotel http://192.168.0.53:8080/ConfigHotel
ProxyPassReverse /Tomcat-3/ConfigHotel http://192.168.0.53:8080/ConfigHotel
```

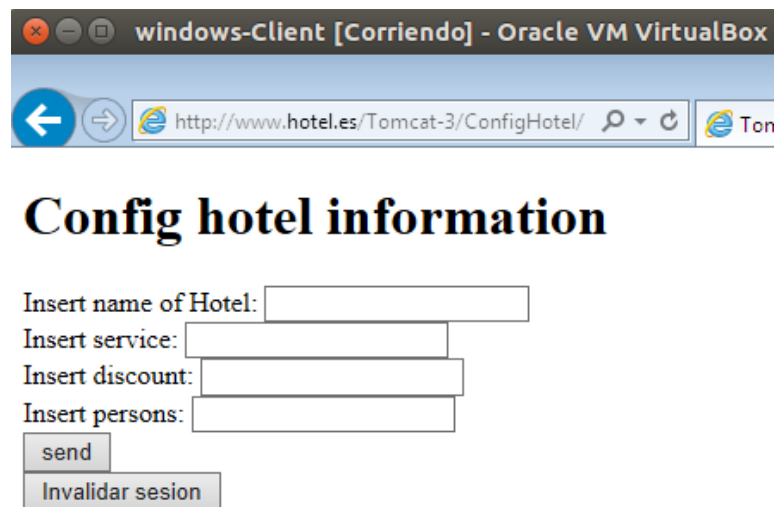
```
<Location /Tomcat-3/ConfigHotel>
    Order allow,deny
    Allow from all
</Location>
```

Ahora reiniciaremos nuestro servidor apache.

```
ubuntu-server-APACHE [Corriendo] - Oracle VM VirtualBox
ubuntu@ubuntu:/etc/apache2/sites-enabled$ sudo service apache2 restart
ubuntu@ubuntu:/etc/apache2/sites-enabled$ _
```

Ahora ya lo tenemos todo preparado para poder acceder desde el cliente a esta última aplicación a través del dominio que habíamos creado con anterioridad.

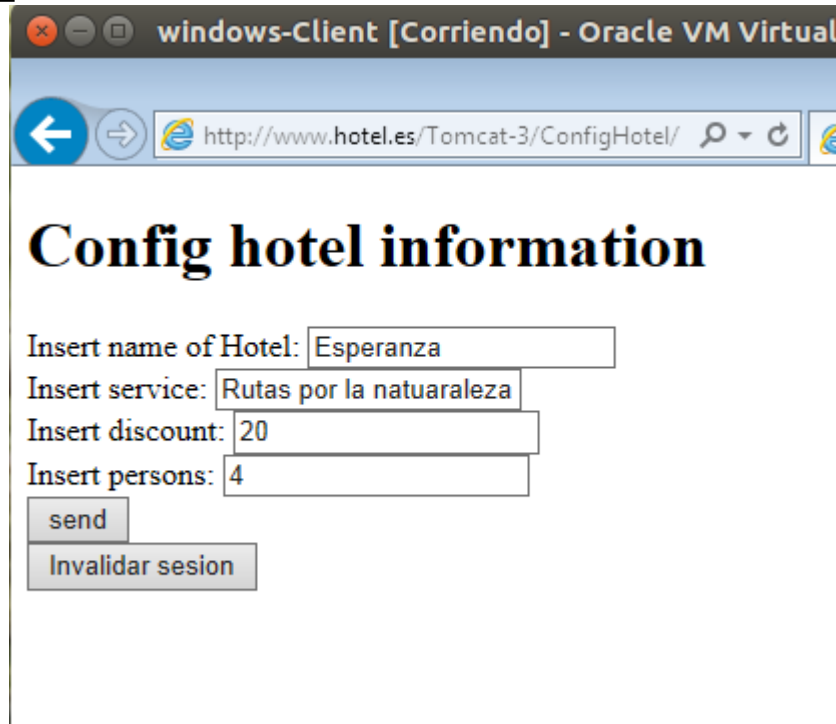
### Comprobación:



The screenshot shows a web browser window titled 'windows-Client [Corriendo] - Oracle VM VirtualBox'. The address bar displays 'http://www.hotel.es/Tomcat-3/ConfigHotel/'. The page content includes the heading 'Config hotel information' and a form with the following elements:

- Input field: 'Insert name of Hotel:'
- Input field: 'Insert service:'
- Input field: 'Insert discount:'
- Input field: 'Insert persons:'
- Button: 'send'
- Button: 'Invalidar sesion'

A continuación vamos a insertar datos y a comprobar que los reciben y nos indica las veces que nos hemos conectado.



Config hotel information

Insert name of Hotel:

Insert service:

Insert discount:

Insert persons:



ConfigHotel Information

Number connections :3

- **name\_hotel:** Esperanza
- **discount\_hotel:** 20
- **persons\_hotel:** 4
- **service\_hotel:** Rutas por la natuaraleza

[Return](#)

Este trabajo me ha costado bastante debido a que lo he tenido que hacer yo solo, en casa y no tenía casi conocimiento sobre Tomcat y los servlets de Java por mi imposibilidad de haber podido ir con normalidad a clase. Pero en cambio he de decir que creía que me iba a costar mucho más de lo que al final ha sido. Encuentro este mundo de los servlets de Java curioso por lo menos, pero a la vez que tedioso de llevar a cabo, habiendo aprendido un poco de php, para mi es mucho más fácil desarrollar un proyecto con este último lenguaje de programación mencionado. También me resultaba un poco raro que para mostrar algo mediante estos servlets se tuviera que incluir la línea típica de `out.println("contenido a mostrar")`, que utilizan las aplicaciones de java, me parece bastante incomodo.

En el fondo ha sido una práctica bastante divertida, he cometido muchos errores, los cuales he ido indicándolos como muestra de mi lucha por solucionar los errores y conflictos conforme me van surgiendo, además de tener errores y diferentes problemas con las máquinas virtuales que aún no entiendo porque me borraba el contenido del fichero `resolv.conf`, que cada vez que reiniciaba o arrancaba tenía que volver a introducir el dns para que funcionara toda la infraestructura requerida para este proyecto.