|  |
| --- |
| Juan cruz gonzalez soler – maximiliano reino |
| Manual de Uso |
| Servidor de Autenticación |
|  |
| **Análisis y Diseño Orientado a Objetos** |
| **14/11/2014** |

|  |
| --- |
| El presente documento pretende ser una simple guía para la configuración inicial del servidor de autenticación. Para obtener más información consulte la documentación de Java adjunta. |

# Servidor xml

En este modo de uso: el servidor recibirá comandos en formato XML enviados remotamente (conexión TCP/IP), los ejecutará y devolverá SIEMPRE una respuesta en formato XML (de error o de aceptación).

Para que pueda conectarse a la base de datos donde realizará la autenticación, agregación, eliminación, modificación y listado de usuarios, debe ubicar el archivo **serverConfig.properties** (en la carpeta del proyecto) y editarlo de acuerdo a su configuración. En el mismo archivo podrá editar la contraseña del administrador del servidor.

Luego solo hace falta ejecutarlo y los clientes podrán conectarse. Lo único que necesitarán es su IP y el puerto donde el servidor esperara las peticiones (especificado en el archivo **serverConfigu.properties**).

Nota:Para ejecutar el servidor desde el IDE de Netbeans, recuerde agregar la librería de conexión a MySQL (adjunta).

## Comandos permitidos

Los comandos que podrán ejecutar los clientes son:

* **ADD:** Agrega un nuevo usuario a la base de datos. La estructura del mensaje XML es:

<MESSAGE TYPE=”ADD”>

<USERNAME>usuario<USERNAME>

<PASSWORD>contraseña</PASSWORD>

<ADM-PASS>contraseña administrador<ADM-PASS>

</MESSAGE>

* **REMOVE:** Elimina un usuario de la base de datos. La estructura del mensaje XML es:

<MESSAGE TYPE=”REMOVE”>

<USERNAME>usuario</USERNAME>

<ADM-PASS>contraseña administrador</ADM-PASS>

</MESSAGE>

* **MODIFY:** Modifica la contraseña de un usuario. La estructura del mensaje XML es:

<MESSAGE TYPE=”MODIFY”>

<USERNAME>usuario</USERNAME>

<PASSWORD>contraseña</PASSWORD>

<NEW-PASS>nueva contraseña</NEW-PASS>

</MESSAGE>

* **AUTHENTICATE:** Autentica un usuario. La estructura del mensaje XML es:

<MESSAGE TYPE=”AUTHENTICATE”>

<USERNAME>usuario</USERNAME>

<PASSWORD>contraseña</PASSWORD>

</MESSAGE>

* **LIST-USERS:** Devuelve una lista de los usuarios en la base de datos. La estructura del mensaje XML es:

<MESSAGE TYPE=”LIST-USERS”>

<ADM-PASS>contraseña administrador</ADM-PASS>

</MESSAGE>

* **LIST-AUT:** Devuelve la lista de autenticaciones realizadas por un usuario específico. La estructura del mensaje XML es:

<MESSAGE TYPE=”LIST-AUT”>

<USERNAME>usuario</USERNAME>

<ADM-PASS>contraseña administrador</ADM-PASS>

</MESSAGE>

## Respuestas

Para los mensajes de tipo **ADD, REMOVE, MODIFY** y **AUTHENTICATE** el servidor responderá con la siguiente estructura de mensaje XML:

<ACK STATUS=”OK|ERROR”>

<DESC>motivo del error</DESC>

</ACK>

* **OK:** Se pudo procesar el mensaje.
* **ERROR:** No se pudo procesar el mensaje. El motivo de error se indica en el tag “DESC”.

Para el mensaje **LIST-USERS** el servidor responderá con la siguiente estructura de mensaje XML:

<LIST-USERS>

<USER>

<USERNAME></USERNAME>

<TIMESTAMP></TIMESTAMP>

</USER>

<USER>

<USERNAME></USERNAME>

<TIMESTAMP></TIMESTAMP>

</USER>

……………

<LIST-USERS>

* El tag “TIMESTAMP” corresponde a la fecha y hora de creación del usuario.

Para el mensaje **LIST-AUT** el servidor responderá con la siguiente estructura de mensaje XML:

<LIST-AUT>

<AUT>

<HOST>###.###.###.###.###</HOST>

<TIMESTAMP>dd:mm:yyyy hh:mm:ss</TIMESTAMP>

</AUT>

<AUT>

<HOST>###.###.###.###.###</HOST>

<TIMESTAMP>dd:mm:yyyy hh:mm:ss</TIMESTAMP>

</AUT>

………….

</LIST-AUT>

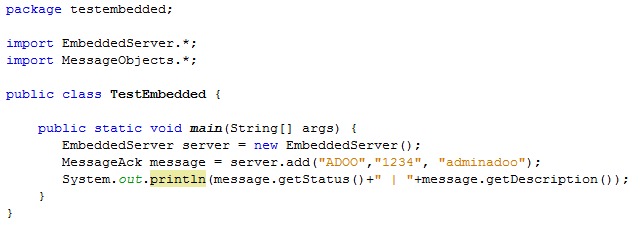
* El tag “HOST” indica la IP desde la cual se solicitó la autenticación.
* El tag “TIMESTAMP” indica la fecha y hora de la autenticación.

# Servidor embebido

En este modo de uso podrá embeber el servidor de autenticación en cualquier otra aplicación Java. Para ello deberá agregar el archivo ***ServerApplication.jar*** ubicado en ***Trabajo-Practico-ADOO\ServerApplication\dist.*** Luego deberá importar en su aplicación los paquetes ***EmbeddedServer*** y ***MessageObjects***.

A continuación se presenta un ejemplo de su funcionamiento con un comando **ADD:**

## Ejemplo



# Base de datos Mysql

La base de datos donde se almacenan los usuarios y sus autenticaciones deberá tener la siguiente estructura:

**Base de datos:** autenticación

**Tabla** usuarios

username: char(20)

password: char(20)

timestamp: datetime

**Tabla** autenticaciones

username: char(20)

host: char(15)

timestamp: datetime

# material adjunto

Adjunto con este documento se encuentran:

* Diagrama de clases
* Diagrama de paquestes
* Diagrama de secuencia
* Código fuente de la aplicación
* Javadoc