Entrega6_sockets

Versión: 3 de marzo de 2020

Objetivos

- Afianzar los conocimientos obtenidos sobre el desarrollo de aplicaciones cliente-servidor utilizando sockets TCP.

Descripción de la práctica

Esta práctica consiste en la modificación del proyecto de gestión de usuarios y quizzes desarrollado en clase para soportar su uso mediante un cliente remoto.

¡¡Nota importante!!: El proyecto proporcionado en esta práctica es el mismo que el proporcionado para la práctica Entrega5_BBDD. Ambas prácticas son independientes por lo que no es necesario implementar en ésta las nuevas funcionalidades desarrolladas en la Entrega5_BBDD. En esta entrega se evaluará únicamente que las funcionalidades del proyecto proporcionado funcionan desde un cliente remoto a través de un socket.

Descargar el código del proyecto

Es necesario utilizar la **versión 12 de Node.js** para el desarrollo de esta práctica. El proyecto debe clonarse en el ordenador desde el que se está trabajando:

```
$ git clone https://github.com/CORE-2020/Entrega6_sockets
```

A continuación se debe acceder al directorio de trabajo, instalar las dependencias y configurar la base de datos (migraciones y seeders). Entonces puede arrancarse el programa.

```
$ cd Entrega6_sockets
$
$ npm install
$
$ npm run migrate   ## En Windows: npm run migrate_win
$
$ npm run seed      ## En Windows: npm run seed_win
$
$ npm start     ##  or 'node main'
....
> h
```

```
Commands (params are requested after):
  > h              ## show help
  >
  > lu | ul | u    ## users: list all
  > cu | uc        ## user: create
  > ru | ur | r    ## user: read (show age)
  > uu             ## user: update
  > du | ud        ## user: delete
  >
  > lq | ql | q    ## quizzes: list all
  > cq | qc        ## quiz: create
  > tq | qt | t    ## quiz: test (play)
  > uq | qu        ## quiz: update
  > dq | qd        ## quiz: delete
  >
  > lf | fl | f    ## favourites: list all
  > cf | fc        ## favourite: create
  > df | fd        ## favourite: delete
  >
  > e              ## exit & return to shell
  >
```

Tareas

El alumno debe modificar el proyecto proporcionado para convertirlo en un servidor que atienda peticiones TCP en el puerto **8080**. Con la nueva versión, para utilizar el programa deberá arrancarse un cliente *telnet* (o *netcat*) conectado a la dirección y puerto del servidor. Usando dicho cliente deberán poderse ejecutar los mismos comandos que anteriormente y su comportamiento será idéntico, recibiendo los resultados de su ejecución también en el cliente. Además se exige el requisito de que puedan conectarse varios clientes y utilizar el programa de manera simultánea.

En resumen las tareas a realizar son las siguientes:

- Modificar el programa para que al ejecutarlo arranque un socket TCP escuchando peticiones en el puerto 8080.
- Modificar el programa para que en vez de leer las órdenes en la linea de comandos (`stdin`) lo haga en la entrada del socket.
- Modificar el programa para que en vez de escribir los resultados de la ejecución de las órdenes en la salida estándar (`stdout`) lo haga en la salida del socket.

- Nota: se debe seguir utilizando el módulo `readline` .

Una vez desarrolladas estas tareas, el comportamiento esperado es el siguiente:

- Al arrancar el programa con la orden `npm start` (o `node main`) éste debe quedarse esperando recibir conexiones en el puerto `8080` .
- Arrancando en otro terminal el programa *telnet* con la orden `telnet localhost 8080` (o `telnet 127.0.0.1 8080`) nos conectaremos al servidor y podremos ejecutar los comandos disponibles.
- Varios clientes deben poder conectarse y utilizar el programa de manera simultánea.

- Al cerrar la conexión de un cliente, el servidor debe seguir arrancado esperando nuevas conexiones.

Nota: El programa *telnet* no viene activado por defecto en Windows. Para activarlo deben seguirse los pasos indicados aquí

Nota 1: Alternativamente puede utilizarse el programa *netcat* en lugar de *telnet*. En ese caso para conectarse al servidor se usará la orden `netcat localhost 8080` (o `netcat 127.0.0.1 8080`).

Nota 2: Si durante el desarrollo de la práctica crees que has podido "romper" la base de datos o crear alguna inconsistencia siempre puedes reiniciar su estado incial eliminando el fichero `db.sqlite` y ejecutando de nuevo los comandos `npm run migrate` y `npm run seed`

Prueba de la práctica

Para ayudar al desarrollo, se provee una herramienta de autocorrección que prueba las distintas funcionalidades que se piden en el enunciado. Para utilizar esta herramienta debes tener node.js (y npm) (https://nodejs.org/es/) y Git instalados.

Para instalar y hacer uso de la herramienta de autocorrección en el ordenador local, ejecuta los siguientes comandos en el directorio del proyecto:

```
$ npm install -g autocorector     ## Instala el programa de test
$ autocorector                    ## Pasa los tests al fichero a entregar
.............................     ## en el directorio de trabajo
... (resultado de los tests)
```

También se puede instalar como paquete local, en el caso de que no se dispongas de permisos en el ordenador desde el que estás trabajando:

```
$ npm install autocorector     ## Instala el programa de test
$ npx autocorector             ## Pasa los tests al fichero a entregar
.............................  ## en el directorio de trabajo
... (resultado de los tests)
```

Se puede pasar la herramienta de autocorrección tantas veces como se desee sin ninguna repercusión en la calificación.

Instrucciones para la Entrega y Evaluación.

Una vez satisfecho con su calificación, el alumno puede subir su entrega a Moodle con el siguiente comando:

```
$ autocorector --upload
```

o, si se ha instalado como paquete local:

```
$ npx autocorector --upload
```

La herramienta de autocorrección preguntará por el correo del alumno y el token de Moodle. En el enlace https://www.npmjs.com/package/autocorector se proveen instrucciones para encontrar dicho token.

RÚBRICA: Se puntuará el ejercicio a corregir sumando el % indicado a la nota total si la parte indicada es correcta:

- **25%:** El servidor atiende conexiones en el puerto 8080
- **25%:** El servidor ejecuta las acciones de manera remota
- **25%:** El servidor admite varias conexiones simultáneas
- **25%:** El servidor cierra correctamente las conexiones

Si pasa todos los tests se dará la máxima puntuación.