Practica 2: Sockets y comunicaciones en .NET

Objetivo

Familiarizarse con el desarrollo de código C#, que se incluyan el uso de hilos y sockets así como de las clases que representan conexiones y algunas otras clases incluidas en los espacios de nombres System.Net, System.Net.Sockets y System.Text

Realización de la práctica

Se desarrollará un programa en C# aplicando los conocimientos vistos en clase de teoría correspondientes al tema 2.

El programa debe simular el funcionamiento de un servidor Web estático y a la vez reducido en sus posibilidades. Una vez desarrollado el programa, éste debe atender las solicitudes o peticiones que se envíen desde un navegador estándar (Mozilla Firefox, Internet Explorer ...)

El programa realizará las siguientes funciones:

- Habrá un socket a la escucha de peticiones Web en un determinado puerto. Éste puerto será determinado por linea de comandos. (no se podrá usar el puerto 80)
- Cuando el servidor reciba un cliente en ese socket, deberá leer lo que el cliente envía, procesar la petición y enviar una respuesta al cliente. A continuación cerrará el socket de datos con ese cliente y esperará a recibir más peticiones.
- El programa deberá trabajar de manera multihilo, es decir, que el servidor Web podrá atender 'n' peticiones simultáneamente. El número 'n' será determinado por línea de comandos al igual que el número de puerto.

Ver formatos de peticiones y otras cuestiones en el ANEXO

Documentación a entregar

E-mail a (ipeteira@uniovi.es) con un adjunto que contenga el código fuente con comentarios, para poder pasar a su revisión y a comprobar el correcto funcionamiento del mismo.

Evaluación

Realización íntegra de la práctica y ejecutándose correctamente, habiendo para ello hecho uso de las clases de Sockets → **Aprobado**

Además del apartado anterior, realización de otro programa similar en C# pero en vez de usar la librería de Sockets utilizando las clases que representan comunicaciones (TCPListener, TCPClient ...) → **Notable**

Ampliación de la práctica para servir ficheros que no estén en el directorio raíz y realización de un **conjunto de experimentos** para analizar el comportamiento del sistema cliente-servidor desarrollado a medida que crece el tamaño de la información enviada y recibida. Todos los experimentos se introducirán en una hoja Excel. En la primera columna debe ponerse el número de bytes enviados y recibidos en cada experimento. Se representarán los datos de forma gráfica. → **Sobresaliente**

La calidad del código generado, así como de los comentarios determinará en cada caso la nota numérica.

En caso de que la práctica se entregue pero no funcione correctamente, se podrá disponer de una semana adicional desde que al alumno le haya sido entregada la práctica corregida para poder subsanar los fallos y así aprobar esta práctica

La fecha de entrega será hasta el día 21 de noviembre de 2008

ANEXO 1

Formato de una petición HTTP

Un cliente (navegador) puede enviar diversos tipos de comandos en una petición HTTP (GET, POST, PUT, ...) Para esta práctica solamente se verá el comando GET. Una petición GET en HTTP tiene el siguiente formato

```
GET fichero http/1.1
Host: ip.del.host
```

- La primera línea comienza por la palabra "GET", después, tras un espacio, viene el nombre del fichero que se está solicitando. Este nombre debe incluir también la ruta hasta el mismo, partiendo del raíz donde se almacenan los documentos web. *Así, por ejemplo, si el cliente pide el fichero /index.html, se refiere al fichero "index.html" situado en la carpeta por debajo de la cual cuelgan los documentos accesibles en ese servidor.* Finalmente, la primera línea termina con el texto "HTTP/1.1" que identifica la versión del protocolo usada.
- En la línea siguiente, aparecerá la palabra "Host:" seguida del nombre del host del cual se quiere obtener el fichero. En nuestro caso el host coincidirá con la máquina en que se ejecuta nuestro servidor Web. *Se puede ignorar la línea, o no si se desea probar entre dos máquinas.*
- Finalmente aparecerá una línea en blanco.

Dada una petición de estas características el servidor Web tendrá como funciones las siguientes:

- Comprobar que la primera palabra es GET, si no lo es remitirse a la siguiente sección de este anexo, *Manejo de errores*.
- Comprobar que el fichero, existe en disco. Solamente, a no ser que se desee hacer la parte opcional de la práctica, se realizarán pruebas para la corrección de ficheros que se encuentren en el raíz, y además solamente se servirán ficheros de tipo HTML, GIF o JPEG (JPG).
- Si el fichero existe el servidor envía una respuesta al cliente de tipo:

```
HTTP/1.1 200 OK
Content-Type: text/html
```

Variable

- Si el fichero no existe el servidor remitirse al apartado *Manejo de errores*.
- Seguidamente debe aparecer una línea en blanco y los bytes tal cual se leen del fichero solicitado.

Manejo de errores

- Si el servidor Web creado debe responder con un error, la cabecera comenzara por una línea indicando el código del error. Será una de las siguientes:

`HTTP/1.1 404 Not Found` si se solicito un fichero en raíz, pero el fichero no existe, o bien
`HTTP/1.1 403 Forbidden` si se solicito un fichero fuera de raíz, o bien
`HTTP/1.1 400 Bad Request` si el comando solicitado no es "GET".

En cualquier caso, seguidamente ira una línea que diga "**`Content-Type: text/html`**" y una línea en blanco que pone fin a la cabecera. Tras esto, se enviara texto en HTML que explique el error (y que será lo que se vea en el navegador). Este texto explicativo puede tenerlo el servidor pre-almacenado en cadenas de texto, o leerlo de un fichero.