salesians
SARRIÀ

Sòcols i serveis

Sockets TCP/IP

Josep Gutiérrez

Departament d'informàtica
Salesians de Sarrià

TCP/IP

- TCP/IP és un protocol orientat a la connexió i amb verificació de trames. Quan el client va a transmetre, primer verifica que el servidor està escoltant i per tant s'intercanvien trames de control.
- Un cop s'ha establert la comunicació, encara hi ha més verificacions, ja que quan s'han transmès un cert volum de bytes, el client para la transmissió a l'espera que el servidor digui que tot el que s'ha rebut és correcte
- Quan finalitza la transmissió és necessari tancar la connexió.
- Les aplicacions que utilitzen TCP per comunicar-se no s'han de preocupar de la integritat de la informació, no han de fer cap tipus de control d'errors atès que poden assumir que tot el que reben és correcte, ja que el mateix protocol s'encarrega de les tasques de control de flux i d'errors.
- La fiabilitat associada a aquest protocol té un cost en la quantitat de recursos necessaris que el fan inadequat per alguns usos.

Sockets TCP/IP i .NET

- Per poder utilitzar **sockets**, caldrà referenciar la llibreria adequada:

```
Using System.Net.Sockets
```

- Existeixen **sockets** amb funció servidor i amb funció client.
- **TcpListener** : Actua com a **Servidor**.
  - És un **socket** que proporciona mètodes senzills per escoltar i acceptar sol·licituds de connexió entrants.
  - Es pot usar **TcpClient** per connectar amb **TcpListener**.
  - Es pot crear una classe **TcpListener** mitjançant un objecte **IPEndPoint** o una adreça IP i un número de port.
- **TcpClient**: Actua com a **Client**
  - És un **socket** que proporciona mètodes senzills per connectar, enviar y rebre fluxos de dades a través de una xarxa.
  - Per a que **TcpClient** pugui connectar-se i intercanviar dades, cal que existeixi un **TcpListener** escoltant per a sol·licituds de connexió entrants

Classe TcpListener

- El seu constructor més habitual és:

```
TcpListener(IPAddress, Int32);
```

- Les propietats més rellevants són:

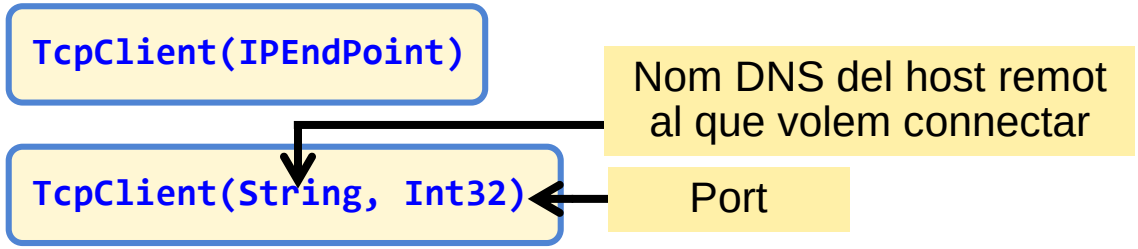
Active	Obté un valor que indica si TcpListener escolta activament les connexions de clients.

- Els mètodes més rellevants són:

AcceptSocket	Accepta una sol·licitud de connexió pendent. Presenta implementació asíncrona amb el mètode **AcceptSocketAsync**
Create	Crea una nova instància de TcpListener per escoltar al port especificat.
Pending	Determina si hi ha sol·licituds de connexió pendents.
Start	Comença l'escolta de sol·licituds de connexió entrants.
Stop	Tanca el TcpListener.

Classe TcpClient

- El seus constructors més habitual són:



- Les propietats més rellevants són:

Active	Obté un valor que indica si se ha establert la connexió.
Available	Obté la quantitat de dades que se han rebut de la xarxa i estan disponibles per a lectura.
Connected	Obté un valor que indica si el l'objecte socket associat està connectat a un host remot.
ReceiveBufferSize	Obté la mida del búffer de recepció.
SendBufferSize	Obté la mida del búffer d'enviament.

Classe TcpClient

- Els mètodes més rellevants són:

Close	Elimina aquesta instancia TcpClient i sol·licita que es tanqui la connexió TCP.
Connect	Connecta el client a un host TCP remot. Es pot fer proporcionant un **IPEndPoint** o la IP i el port.
GetStream	Retorna el **NetworkStream** usat per enviar i rebre dades.

- Per enviar i rebre dades cal utilitzar el mètode **GetStream** per tal d'obtenir un **NetworkStream**. A continuació cal cridar els mètodes **Write** i **Read** de **NetworkStream** per enviar i rebre dades del host remot.
- El mètode **Close** allibera recursos associats a **TcpClient**.

Servidor TCP/IP amb C#

- Per implementar un servidor senzill en TCP/IP cal seguir una sèrie de passos:
  - Crear i instanciar un objecte **TcpListener** i activar-lo per tal que resti a l'escolta

```
TcpListener Listener = new TcpListener(IPAddress.Any, portN);
Listener.Start()
```

  - Quan rep una petició de connexió, l'accepta creant un socket de tipus **TcpClient** preparat per a poder rebre i enviar dades. Aquest nou **TcpClient** es crea amb el mètode **AcceptTcpClient**

```
TcpClient client = Listener.AcceptTcpClient();
```

  - Es crea el **Networkstream** associat al socket client i es dimensiona el **buffer**.

```
NetworkStream ns= client.GetStream();
byte[] buffer = new byte[1024];
```

Servidor TCP/IP amb C#

- A partir d'aquí podem llegir el stream tal i com hem vist abans.
- Si el que necessitem és poder enviar algun missatge de tornada cap el socket client extern que ens ha demanat la connexió, ho podem fer amb el mètode **Write**. Només cal escriure una cadena i convertir-la en bytes utilitzant el mètode **GetBytes** de la llibreria **System.Text** amb el encoding apropiat i tot seguit enviar-ho amb el mètode **Write** del socket.

```
byte[] nouBuffer = Encoding.ASCII.GetBytes("Hola s2AM!!");
ns.Write(nouBuffer, 0, nouBuffer.Length);
```

- Quan vulguem aturar el servidor caldrà tancar l'objecte **TcpClient** i l'objecte **Networkstream** i aturar el **TcpListener**.

```
ns.Close();
client.Close();
Listener.Stop();
```

Client TCP/IP amb C#

- Per implementar un client senzill en TCP/IP cal seguir una sèrie de passos:
  - Cal realitzar les comprovacions pertinents per tal d'assegurar-se que el servidor al qual volem connectar-nos està actiu i escoltant.
  - Si és així podem crear un objecte **TcpClient** associat al servidor

```
TcpClient client = new TcpClient(server, port);
```

  - Convertim el missatge a enviar en un array de Bytes

```
Byte[] dades = Encoding.ASCII.GetBytes(txtMissatge.text);
```

  - Instanciem un **NetworkStream** per a lectura i escriptura a partir del mètode **GetStream**

```
NetworkStream ns = client.GetStream();
```

Client TCP/IP amb C#

- A partir d'aquí ja podem enviar el missatge al servidor

```
ns.Write(dades, 0, dades.Length);
```

- I el **socket** queda a l'escolta de la resposta del servidor

```
dadaResposta = new Byte[256];
String resposta = String.Empty;
Int32 bytes = ns.Read(dadaResposta, 0, dadaResposta.Length);
resposta = Encoding.ASCII.GetString(dadaResposta, 0, bytes);
```

Bibliografia

- TcpListener Class
- TcpClient Class
- How to set up TcpListener to always listen and accept multiple connections?
- Sending Files using TCP
- Creating Simple TCP/IP Server And Client to Transfer Data Using C#