

Sockets

—

Programación Distribuida y Tiempo Real

¿Qué es un Socket?

—

Sockets

Conceptos generales

- Mecanismo para comunicación entre procesos
 - Dos procesos de una misma PC
 - Dos procesos en PC diferentes unidas por la red
 - A priori, un socket no implica TCP/IP para conexión
 - Generalmente utiliza una estructura Cliente/Servidor
-

Sockets en C

—

Cliente en C

Cliente

Sockets en C

- `Socket()`
 - Devuelve un file descriptor del socket que luego podemos usar para llamadas al sistema.
 - Si devuelve -1 se produjo un error
- `Connect()`
 - Se usa para conectarse a una dirección IP/puerto específica

Cliente

Sockets en C

- Read()
 - Lee una determinada cantidad de datos desde el socket
- Write()
 - Escribe una determinada cantidad de datos en el socket

Servidor en C

Servidor

Sockets en C

- Socket()
 - Devuelve un file descriptor del socket que luego podemos usar para llamadas al sistema.
 - Este socket no se conecta con el cliente, se encarga de crear las conexiones.
 - Bind()
 - Asocia un socket (file descriptor) con un puerto de la máquina.
-

Servidor

Sockets en C

- Listen()
 - Se usa para esperar una determinada cantidad de conexiones entrantes en un socket.
- Accept()
 - Se usa para esperar las conexiones desde los clientes.
 - Devuelve un nuevo socket que es donde se realiza la conexión.
 - Es transparente a nosotros
 - Es un nuevo File Descriptor

Servidor

Sockets en C

- Read()
 - Lee una determinada cantidad de datos desde el socket
 - Write()
 - Escribe una determinada cantidad de datos en el socket
-

Ejemplo en C

http://www.linuxhowtos.org/C_C++/socket.htm

Sockets en Java

—

Paquetes

Sockets en Java

- Se utilizan los siguientes paquetes:
 - `java.io.*`
 - `java.net.*`
 - Clases:
 - `Socket()` – Cliente
 - `ServerSocket()` – Servidor
 - `InputStream`
 - `OutputStream`
-

Cliente en Java

Cliente

Sockets en Java

- Socket()
 - Crea directamente la conexión
 - Lleva como parámetros la IP y el Puerto donde escucha el Servidor



Cliente

Sockets en Java

- Read()
 - No lee directo desde el Socket
 - Se debe utilizar un InputStream relacionado al socket
 - Write()
 - No escribe directo en el Socket
 - Se debe utilizar un OutputStream relacionado al socket
-

Servidor en Java

Servidor

Sockets en Java

- `ServerSocket()`
 - Crea el socket donde se va a esperar la conexión con los clientes
 - Lleva el puerto donde se escucha como parámetro
 - `Accept()`
 - Mantiene el socket a la espera de una conexión
-

Servidor

Sockets en Java

- Read()
 - No lee directo desde el Socket
 - Se debe utilizar un InputStream relacionado al socket
 - Write()
 - No escribe directo en el Socket
 - Se debe utilizar un OutputStream relacionado al socket
-

Ejemplo en Java

<http://docs.oracle.com/javase/tutorial/networking/sockets/>

¿Realmente es
Cliente/Servidor?

—

¿Realmente es Cliente/Servidor?

- Las conexión no implica específicamente una petición, o una respuesta a una petición, solo es comunicación
- Para ser cliente/servidor se deben cumplir los pasos de conexión:
 - Inicialización
 - Envío/recepción de peticiones
 - Finalización
- Protocolo de transporte
 - Socket Stream – Orientados a la conexión – TCP (C/S)
 - Socket Datagram – No orientados a la conexión - UDP

Profundización

—

Temas relacionados

- Tipos de Servidores
- Concurrencia (varios clientes)
- Seguridad en la comunicación

¡Muchas gracias!
¿Preguntas?

