

# Sesión de Laboratorio L3

## Tema 3 (1): Sockets

### Lectura previa:


Kurose 2.7 hasta TCPServidor.java (pag. 159)

### Información de soporte:

- <http://download.oracle.com/javase/6/docs/api/index.html>
- 

- El API de los sockets
  - ¿Qué es un socket? ¿Cómo se identifica?
  - Tipos de sockets
- Los sockets en Java
  - Sockets TCP
    - La clase Socket. El método close.
    - Clientes TCP
      - Captura de excepciones mediante try ... catch
  - Cómo leer de un socket: el método `getInputStream`. Lectura de líneas de texto: la clase `Scanner`. Métodos: `next`, `nextInt`, `nextLine`, `hasNext` ....
    - Cómo leer del teclado.
  - Cómo escribir a través de un socket: el método `getOutputStream`. Escribir líneas de texto: la clase `PrintWriter`.
    - Métodos `print` y `println`. Diferencia (¡OJO, con los finales de línea dependientes de la plataforma!)
      - Necesidad de vaciar el buffer TCP al hacer una petición para evitar los bloqueos. El método `flush` (clases `OutputStream` y `PrintWriter`).
  - Direcciones IP en java: la clase `InetAddress`.
    - Cómo obtener información de la conexión establecida: métodos `getLocalPort`, `getPort`, `getLocalAddress` y `getInetAddress`.
    - Argumentos de un programa. Conversión de string a entero para indicar un número de puerto.

### Ejercicios:

- **Ejercicio 1:** Escribe un programa que conecte al puerto 80 del servidor “[www.upv.es](http://www.upv.es)”, imprima por pantalla el mensaje “¡Conectado!”, después cierre el socket y termine. No hace falta el tratamiento de excepciones.
- **Ejercicio 2:** Añade un método al programa del ejercicio 1. Este método realizará lo mismo que el del ejercicio 1 con 2 diferencias: el mensaje será ahora “¡Conectado de nuevo!” y se capturarán las posibles excepciones mediante una cláusula try ... catch.
  - Ejecútalo y comprueba que funciona correctamente.
  - Sustituye el puerto por el 81. Ejecuta el método y comprueba que ocurre.
  - Restaura de nuevo el puerto al valor 80. Añadir una w al nombre del servidor, ejecuta el método de nuevo y comprueba que ocurre.
  - Al intentar conectar un socket cliente con el de un servidor, ¿cuándo se genera una UnknownHostException? ¿y una IOException? 
- **Ejercicio 3:** Añade un nuevo método a tu programa que conecte al puerto 25 del servidor “smtp.upv.es”, lea la primera línea de texto que devuelve el servidor y la imprima por pantalla.
- **Ejercicio 4:** Cliente de daytime. Conecta al puerto 13 del servidor “ntp.upv.es” e imprime la respuesta del servidor por pantalla.
- **Ejercicio 5:** Cliente HTTP/1.0 básico en modo texto. Se conecta al puerto 80 del servidor “[www.upv.es](http://www.upv.es)” y envía la cadena “GET / HTTP/1.0\r\n\r\n”. Muestra la respuesta recibida del servidor por pantalla y termina.
- **Ejercicio 6:** Añade un nuevo método al programa. El método implementará un cliente que se conecta al puerto 25 del servidor SMTP de la UPV, y tras conectarse, muestra por pantalla el puerto local y el remoto, así como la dirección IP del servidor. Ejecútalo tres veces seguidas y comprueba cómo se modifican los valores mostrados.
- **Ejercicio 7(opcional):** Modifica el programa anterior para que acepte como argumentos del programa el nombre del servidor y el número de puerto en el que escucha.