

Equipo G76: Chat multiusuario en red, vía sockets.

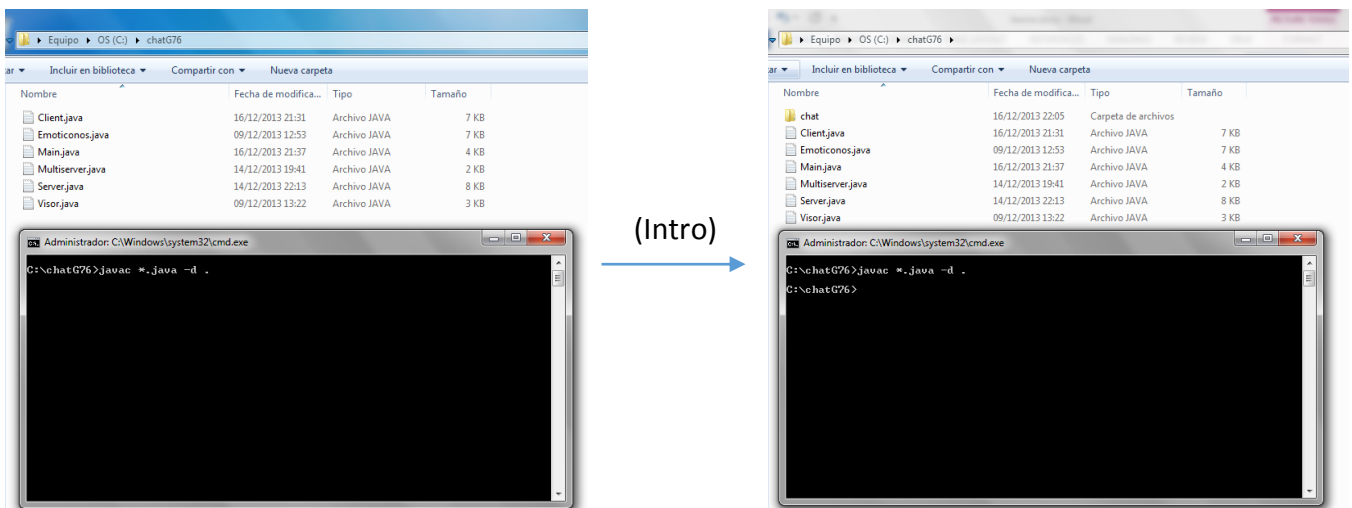
Resumen general:

Es un chat entre ordenadores en red (o en un mismo ordenador), con 3 partes principales: servidor, cliente y visor:

- El servidor recibe todos los mensajes de los clientes y los reenvía a los visores para ser vistos. Además, comprueba nombres de usuarios y contraseñas de los usuarios según se conectan.
- El cliente es la ventana donde se escriben los mensajes, se conecta al servidor y envía los mensajes al mismo. Además, en el sistema operativo Windows, abre un visor de manera automática, si el usuario quiere (en otros SO se puede abrir el visor manualmente).
- El visor recibe los mensajes del servidor y los muestra por pantalla, en tiempo real (si se hiciera en la misma ventana que client, solo se actualizaría después de que el usuario termine de escribir y pulse intro)

Compilación

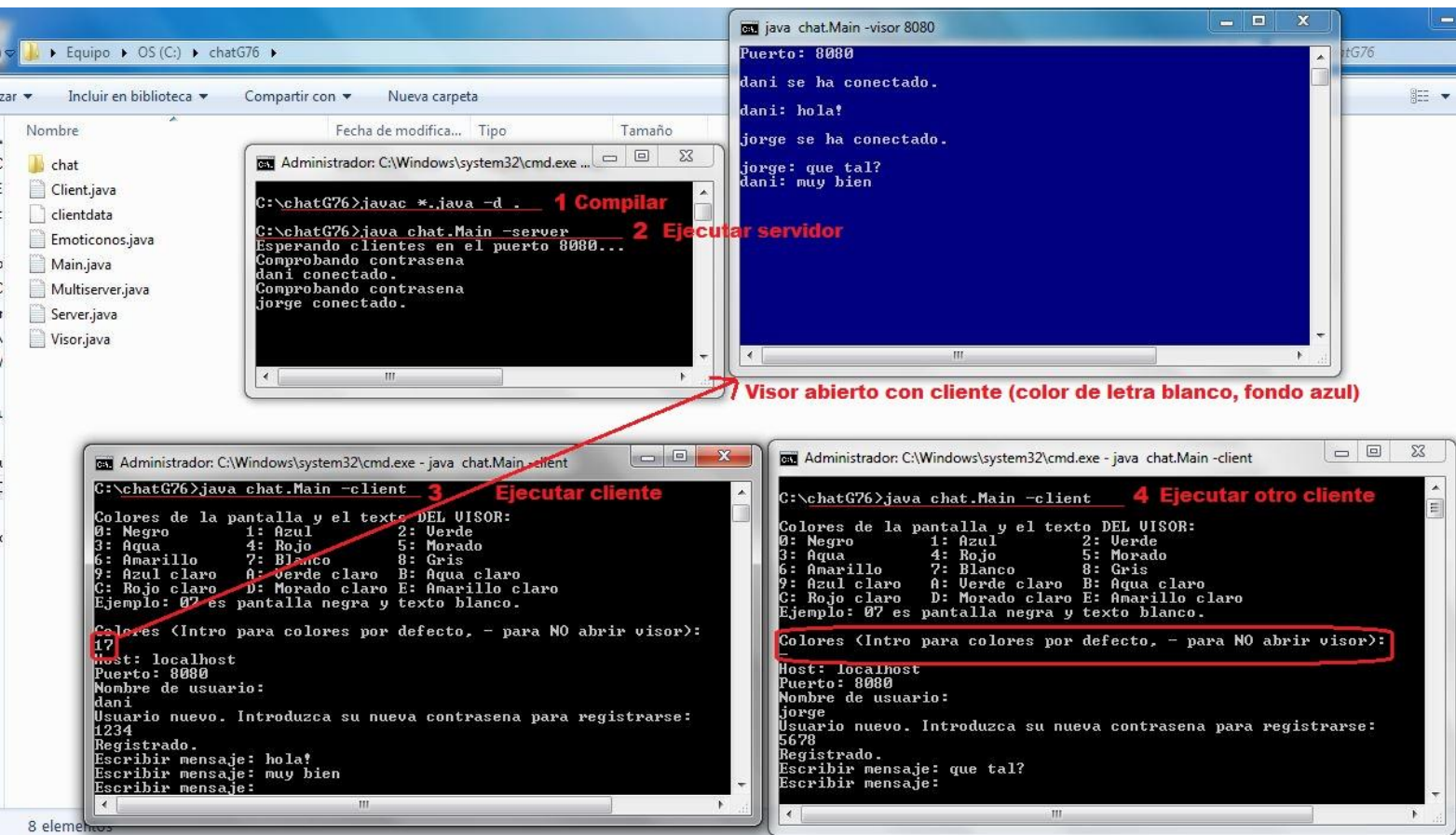
1. Descomprimir el Zip en una carpeta.
2. Abrir una ventana de comandos en dicha carpeta.
3. Ejecutar: `javac *.java -d .` (Aparecerá una carpeta 'chat' al lado del código fuente)



Ejecución para ver funcionamiento normal, en un mismo ordenador.

1. Desde la carpeta donde se descomprimió el zip, ejecutar PRIMERO:
`java chat.Main -server`
2. A continuación se pueden ejecutar en otras ventanas, desde la misma carpeta, tantos clientes y visores como se quiera:
`java chat.Main -client` (da la opción de abrir un visor automáticamente)

java chat.Main -visor (ejecucion manual, para SO distintos de windows, o si no se abrió vía el cliente)



Ejecución si se desean parámetros adicionales (en diferentes ordenadores, distintos puertos etc.):

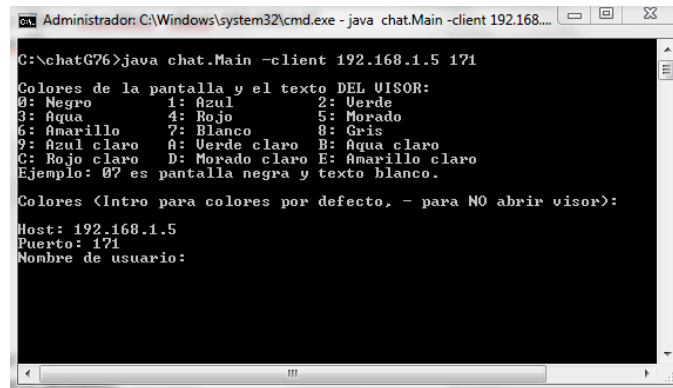
1. Para ejecutar en varios ordenadores, hay que ejecutar primero un servidor en uno de ellos.

java chat.Main -server

2. En otros ordenadores, (en el mismo ordenador sigue funcionando como antes), ejecutar client, dando la dirección IP o el nombre de host del ordenador con el servidor en espera (si no se sabe, se puede mirar ejecutando el comando ipconfig):



java chat.Main -client XXX.XXX.XXX.XXX (ej.: "java chat.Main -client 192.168.1.5" conectaría el cliente al servidor en esa dirección IP)



```
Administrador: C:\Windows\system32\cmd.exe - java chat.Main -client 192.168.1.5 171

C:\chatG76>java chat.Main -client 192.168.1.5 171

Colores de la pantalla y el texto DEL VISOR:
0: Negro      1: Azul      2: Verde
3: Aqua       4: Rojo      5: Morado
6: Amarillo   7: Blanco    8: Gris
9: Azul claro 0: Verde claro B: Aqua claro
C: Rojo claro D: Morado claro E: Amarillo claro
Ejemplo: 07 es pantalla negra y texto blanco.

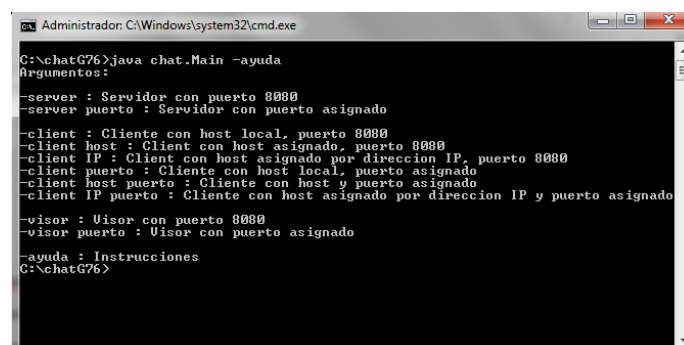
Colores (Intro para colores por defecto, - para NO abrir visor):
Host: 192.168.1.5
Puerto: 171
Nombre de usuario:
```

3. Si se quiere usar otro puerto, que no sea el 8080, se puede dar argumentos con el siguiente formato (sustituir "puerto" por el puerto deseado, IP por la IP o host por el nombre de host):

Argumentos chat.Main :

- server : Servidor con puerto 8080
- server puerto : Servidor con puerto asignado
- client : Cliente con host local, puerto 8080
- client host : Client con host asignado
- client IP : Client con host asignado por dirección IP
- client puerto : Cliente con host local, puerto asignado
- client host puerto : Cliente con host y puerto asignado
- client IP puerto : Cliente con host asignado por dirección IP y puerto asignado
- visor : Visor con puerto 8080
- visor puerto : Visor con puerto asignado

Se puede acceder a estas instrucciones dando argumento -ayuda (java chat.Main -ayuda).



```
Administrador: C:\Windows\system32\cmd.exe

C:\chatG76>java chat.Main -ayuda

Argumentos:

-server : Servidor con puerto 8080
-server puerto : Servidor con puerto asignado
-client : Cliente con host local, puerto 8080
-client host : Client con host asignado, puerto 8080
-client IP : Client con host asignado por dirección IP, puerto 8080
-client puerto : Cliente con host local, puerto asignado
-client host puerto : Cliente con host y puerto asignado
-client IP puerto : Cliente con host asignado por dirección IP y puerto asignado
-visor : Visor con puerto 8080
-visor puerto : Visor con puerto asignado
-ayuda : Instrucciones

C:\chatG76>
```

El puerto DEBE SER COMUN entre servidor, cliente y visor.

Ejemplos:

java chat.Main -server 171 (servidor con puerto 171)

java chat.Main -client 171 (en mismo ordenador, cliente con puerto 171)

java chat.Main -client 192.168.0.100 171 (desde otro ordenador, con servidor en esa IP, puerto 171)

java chat.Main -visor 171 (visor en cualquier ordenador, puerto 171)

Funciones adicionales

Emoticonos:

Al escribir :) , :(, :P , xD , >:(, :O , se muestran emoticonos ASCII en el visor.

Colores de Visor:

Al abrir un visor vía client, se puede seleccionar el color de letra y fondo de la ventana.

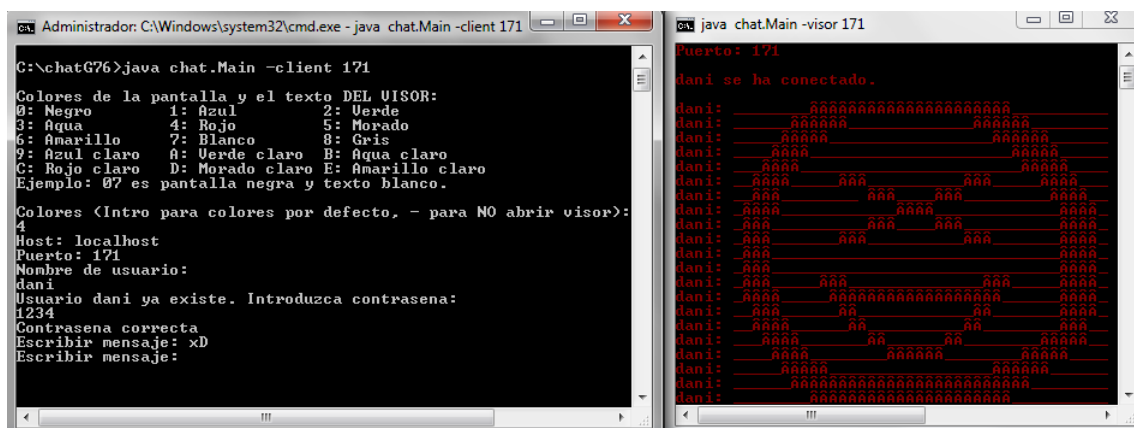
Nombres de usuario y contraseña:

Crea una base de datos (archivo clientdata) con usuarios y contraseñas que el servidor comprueba.

Si el usuario es nuevo lo registra en la base de datos con una contraseña.

Si el usuario está en la base de datos, pide la contraseña asociada. Si es correcta, permite el nombre de usuario. Si no, asigna la IP de su ordenador como nombre de usuario.

Si el usuario no escribe ningún nombre de usuario, le asigna la IP de su ordenador como nombre de usuario.



Para entender el funcionamiento de sockets y sockets multicast:

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

<http://docs.oracle.com/javase/tutorial/networking/datagrams/broadcasting.html>

Si hay cualquier duda o problema, enviar email a danipeiro@gmail.com