Compilar y ejecutar el servidor

Ya tenemos definido el servidor. Ahora tenemos que compilar sus clases mediante los siguientes pasos:

Compilamos el interfaz remoto. Además lo agrupamos en un fichero JAR para tenerlo presente tanto en el cliente como en el servidor:

```
javac InterfazRemota.java
jar cvf objRemotos.jar InterfazRemota.class
```

Luego, compilamos las clases que implementen los interfaces. Y para cada una de ellas generamos los ficheros Stub y Skeleton para mantener la referencia con el objeto remoto, mediante el comando rmic:

```
set CLASSPATH=%CLASSPATH%;.\objRemotos.jar;.
javac ClaseRemota.java
rmic -d . ClaseRemota
```

Observamos en nuestro directorio de trabajo que se han generado automáticamente dos ficheros .class (ClaseRemota _Skel.class y ClaseRemota _Stub.class) correspondientes a la capa stub-skeleton de la arquitectura RMI.

Para ejecutar el servidor, seguimos los siguientes pasos:

Se arranca el registro de RMI para permitir registrar y buscar objetos remotos. El registro se encarga de gestionar un conjunto de objetos remotos a compartir, y buscarlos ante las peticiones de los clientes. Se ejecuta con la aplicación rmiregistry distribuida con Java, a la que podemos pasarle opcionalmente el puerto por el que conectar (por defecto, el 1099):

```
start rmiregistry 1234
```

Por último, se lanza el servidor:

```
java -Djava.rmi.server.hostname=127.0.0.1 MiClaseRemota 1234
```

Compilar y ejecutar el cliente

Una vez que ya tenemos definido el cliente, para compilarlo hacemos:

```
set CLASSPATH=%CLASSPATH%;.\objRemotos.jar;.
javac MiClienteRMI.java
```

Luego, para ejecutar el cliente hacemos:

```
java MiClienteRMI 127.0.0.1 1234
```