

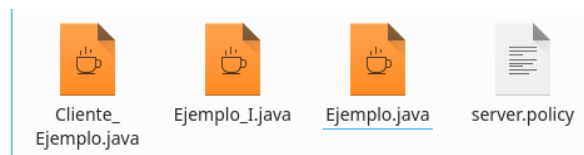
# Práctica 3. Primera parte: Implementación de los ejemplos

Noelia Escalera Mejías      Grupo DSD1

2 de abril de 2020

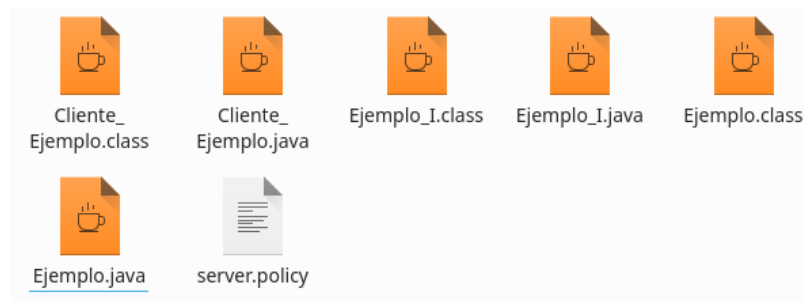
## 1. Ejemplo 1

1. Crear los archivos



2. Compilamos los *.java*.

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ javac *.java
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$
```



3. Invocamos *rmiregistry*

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ rmiregistry &
[1] 5394
```

4. Lanzamos el servidor

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo
Ejemplo bound
```

## 5. Lanzamos el cliente

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Cliente Ejemplo localhost 3
Buscando el objeto remoto
Invocando el objeto remoto
```

En el servidor obtenemos esto:

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy
Ejemplo
Ejemplo bound
Recibida petición de proceso: 3
Hebra 3
```

Si le pasamos 0 como argumento al cliente:

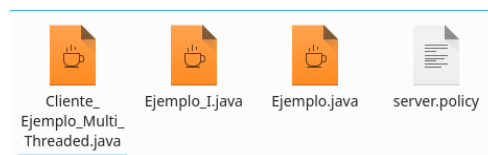
```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Cliente Ejemplo localhost 0
Buscando el objeto remoto
Invocando el objeto remoto
```

```
Recibida petición de proceso: 0
Empezamos a dormir
Terminamos de dormir
Hebra 0
```

Lo que está ocurriendo es que el cliente busca un objeto remoto, encuentra el que ha creado nuestro servidor y lo llama pasándole como argumento el que le hemos pasado nosotros. El servidor los interpreta como número de proceso y dependiendo de si es el proceso 0 o no hace un sleep o no.

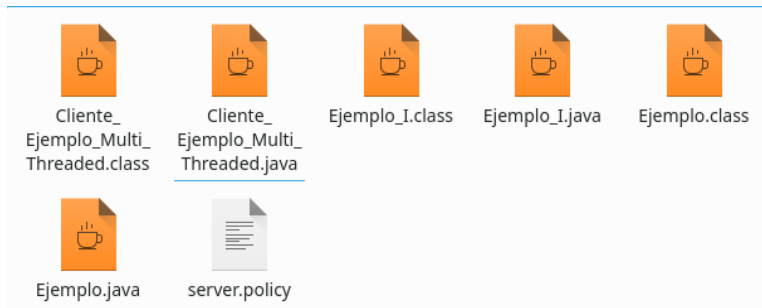
## 2. Ejemplo 2

### 1. Creamos los archivos



### 2. Compilamos los .java.

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 2$ javac *.java
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 2$
```



### 3. Invocamos *rmiregistry*

```
noelia@noelia-pc:~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 2$ rmiregistry &
[1] 7178
```

#### 4. Lanzamos el servidor

```
me@lapuola:pc~/.Escritorio/Universidad/3º/PSD/Prácticas/P3/Parte 1/Ejemplo$ java -cp . -Djava.rmi.server.codebase=file:/ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo
Ejemplo bound
```

## 5. Lanzamos el cliente

```
mellon@mla-pc:~/Escritorio/universidad37/DSD/actividades/P3/Parte 1/Ejemplo 5$ java -cp . -Djava.rmi.server.codebase=file://. -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy ClienteEjemplo.MultipThreaded localhost 5
```

En el servidor obtenemos esto:

```
root@kali:~/git-Escritorio/Universidad/17/BD/Practicas/P3/Parte 1/Ejemplo 2# java -cp -Djava.rmi.server.codebase=file:/ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo
@ejemplo bound
Entra Hebra Cliente2
Sale Hebra Cliente2
Entra Hebra Cliente0
Entra Hebra Cliente1
Sale Hebra Cliente1
Impezamos a dormir
Entra Hebra Cliente3
Sale Hebra Cliente3
Entra Hebra Cliente4
Sale Hebra Cliente4
Terminamos de dormir
Sale Hebra Cliente0
```

Lo que ocurre aquí es que al cliente le pasamos como argumento el número de hebras que se quieren lanzar. El servidor hace sleep a los procesos múltiples de 10.

Si añadimos ahora el modificador `synchronized`:

```
mal@kali:~$ cd /Escritorio/Universdad/37/OSD/Prácticas/P3/Parte 1/ Ejemplo 25 java -cp -Djava.rmi.server.codebase=file:/ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy Ejemplo
Ejemplo bound
Entra Hebra Cliente0
Imprimos a dorair
Terminamos de dorair
Sale Hebra Cliente0

Entra Hebra Cliente4
Sale Hebra Cliente4

Entra Hebra Cliente1
Sale Hebra Cliente1

Entra Hebra Cliente2
Sale Hebra Cliente2

Entra Hebra Cliente3
Sale Hebra Cliente3
```

### 3. Ejemplo 3

-  cliente.java     contador.java     icontador.java     server.policy     servidor.java

- ```
noelia@noelia-pc: ~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 3$ javac *.java
noelia@noelia-pc: ~/Escritorio/Universidad/3º/DSD/Prácticas/P3/Parte 1/Ejemplo 3$
```

### 3. Invocamos *rmiregistry*

4

#### 4. Lanzamos el servidor

```
huelleguella-pc:~/Escritorio/Universidad/3º/DSO/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy servidor
Servidor RemoteException | MalformedURLException preparado
```

Antes apareció el siguiente error:

```
huelleguella-pc:~/Escritorio/Universidad/3º/DSO/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy servidor
Exception: Port already in use: 1899; nested exception is:
java.net.BindException: la dirección ya se está usando (Bind failed)
```

Se ha solucionado cambiando el puerto en *servidor.java*

#### 5. Lanzamos el cliente

```
huelleguella-pc:~/Escritorio/Universidad/3º/DSO/Prácticas/P3/Parte 1/Ejemplo 1$ java -cp . -Djava.rmi.server.codebase=file:./ -Djava.rmi.server.hostname=localhost -Djava.security.policy=server.policy cliente localhost
Poniendo contador a 0
Incrementando...
Media de las RMI realizadas = 0.099 msecs
RMI realizadas = 1000
```

Lo que ocurre aquí es que el servidor sirve un contador y el cliente lo llama 1000 veces.