
PRÁCTICA 2 - SEGUNDA PARTE
CALCULADORA - APACHE THRIFT

Tercer Curso - Grado en Ingeniería Informática
Curso 2019-2020

PEDRO SERRANO PÉREZ

Contents

1	Introducción	1
2	Solución	1
2.1	Servidor	1
2.2	Cliente	1
3	Excepciones creadas por el usuario en Apache Thrift	2
4	Capturas de pantalla	3

1 Introducción

En esta segunda parte de la práctica se va a desarrollar la misma calculadora, esta vez haciendo uso de la tecnología **Apache Thrift**. La calculadora continua con una arquitectura *Cliente-Servidor*:

- Servidor. El servidor se ha implementado en **Python**. Aporta las funciones *sumar*, *restar*, *multiplicar* y *dividir*. Estas operaciones las podrá utilizar el cliente para que, el servidor, realice la operación con los dos números enteros que el cliente desee.
- Cliente. El cliente se ha implementado en **Java**. Este se encarga de, por un lado, recibir la operación que desea calcular el usuario y, por otro, utilizar las funciones del servidor para obtener el cálculo.

2 Solución

Debido a que la funcionalidad implementada, tanto en el cliente como en el servidor, es prácticamente la misma que en la primera parte de la práctica, nos vamos a centrar en los distintos procedimientos que se han seguido para la implementación de la calculadora en Apache Thrift, utilizando un lenguaje para el servidor y otro para el cliente.

2.1 Servidor

Como ya se ha mencionado antes, el servidor se ha implementado en Python. Para generar los ficheros de Thrift necesarios para python usamos la orden:

```
thrift -gen py calculadora.thrift
```

Para ejecutar el servidor utilizamos la siguiente orden:

```
python servidor.py
```

Dentro de la carpeta que se genera, creamos el servidor, en el que se han implementado las operaciones matemáticas de suma, resta, multiplicación y división.

El servidor quedará escuchando en el puerto 9090, con la dirección 127.0.0.1 (localhost). La principal diferencia, en cuanto a funcionalidad, en comparación a la primera parte, es que el servidor enviará una excepción a los clientes cuando estos intenten dividir entre 0. Profundizaremos más sobre la creación de excepciones en Apache Thrift en la sección correspondiente a estas.

2.2 Cliente

El cliente se ha implementado en Java. Para generar los ficheros de Java, de forma similar a la vista para Python, ejecutamos la siguiente orden:

```
thrift -gen java calculadora.thrift
```

Para la compilación y ejecución en java se ha exportado la siguiente variable CLASSPATH:

```
export CLASSPATH=libs/libthrift-0.13.0.jar:libs/javax.annotation-api-1.3.2.jar:libs/slf4j-api-1.7.25.jar:libs/slf4j-log4j12-1.7.25.jar:libs/log4j-1.2.17.jar:ejecutables
```

Esta variable permite compilar utilizando las librerías de Apache Thrift para Java y, ejecutar el fichero Cliente que se generará en el directorio ejecutables.

Para compilar se ha usado la siguiente orden (situándonos en la carpeta "gen-java"):

```
javac -d ./ejecutables ./src/Cliente.java ./calculadora/Calculadora.java
      ./calculadora/DivideEntre0.java
```

Finalmente ejecutamos la siguiente orden (situándonos de nuevo en la carpeta "gen-java"):

```
java Cliente
```

El cliente recibirá la operación del usuario, comprobará que la operación introducida tiene un formato correcto y, llamará a las funciones del servidor que sean necesarias para resolver el cálculo.

El cliente debe ser capaz de atrapar la excepción que lanza el servidor cuando se intenta dividir algo entre 0.

3 Excepciones creadas por el usuario en Apache Thrift

Como hemos mencionado se hace uso de una excepción que hemos creado. Esta excepción permitirá al servidor alertar al cliente que se está intentando dividir entre 0.

Para que el servidor pueda comunicar esta excepción al cliente vamos a definirla en el fichero calculadora.thrift de la siguiente forma:

```
exception DivideEntre0 {
    1: i32 codigo,
    2: string cadena
}
```

A continuación debemos indicar en el mismo fichero las funciones que lanzarán la excepción. Nos quedaría lo siguiente:

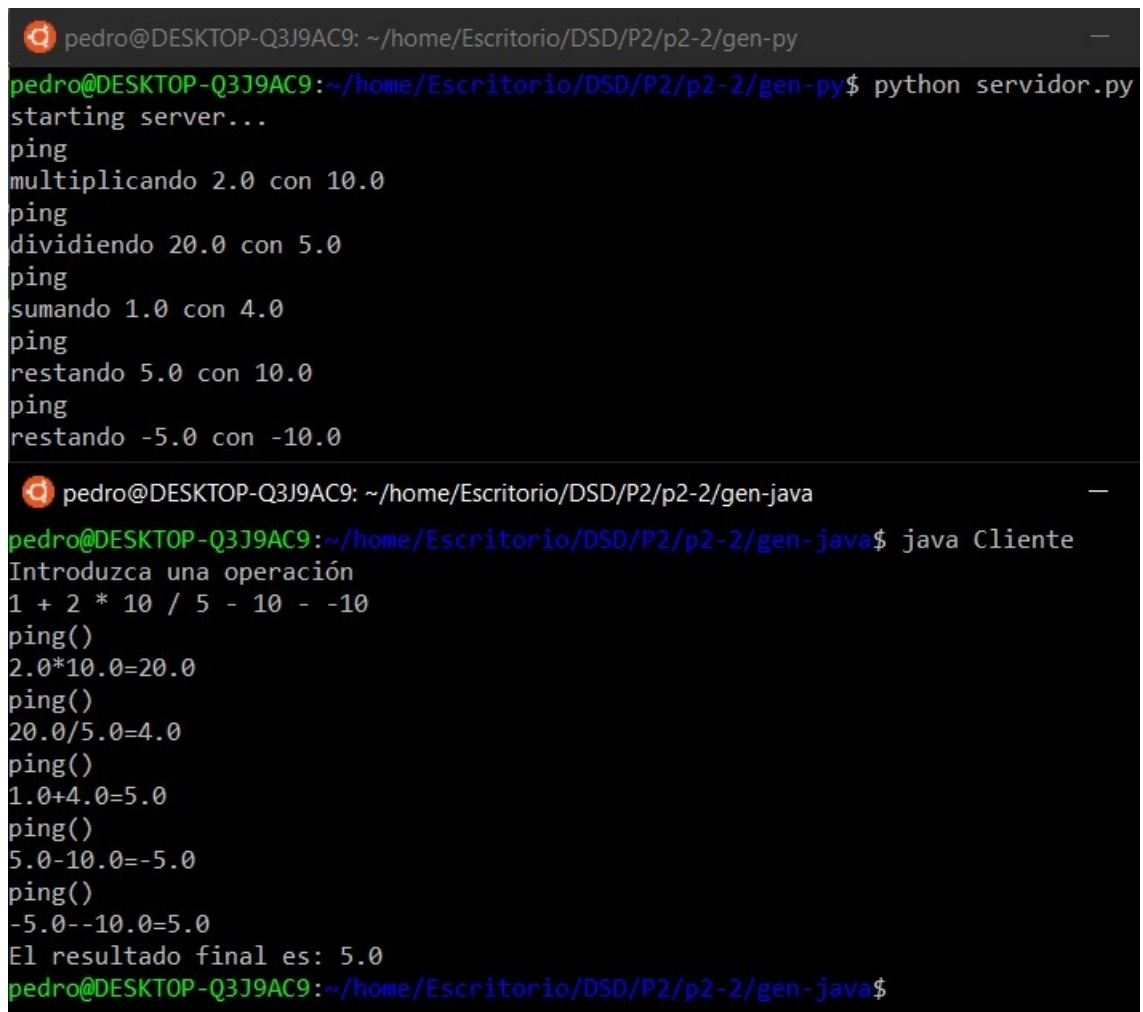
```
service Calculadora {
    void ping (),
    double suma (1: double num1 , 2: double num2 ),
    double resta (1: double num1 , 2: double num2 ),
    double multiplica (1: double num1 , 2: double num2 ),
    double divide (1: double num1 , 2: double num2 ) throws (1:DivideEntre0 div0),
}
```

Una vez definida la excepción se generará un fichero para la clase DivideEntre0 en java y, en Python, se encontrará en el fichero ttypes, por lo que será necesario incluir los respectivos ficheros en el cliente y en

el servidor. Finalmente, en Python, el servidor, se creará y lanzará la excepción cuando sea necesario (raise) y, en Java, el cliente, se capturará dicha excepción (Bloque try-catch).

4 Capturas de pantalla

A continuación se muestra una captura de pantalla que muestra el formato en el que es posible introducir la operación y la interacción entre el cliente y el servidor.



```
pedro@DESKTOP-Q3J9AC9: ~/home/Escritorio/DSD/P2/p2-2/gen-py
pedro@DESKTOP-Q3J9AC9:~/home/Escritorio/DSD/P2/p2-2/gen-py$ python servidor.py
starting server...
ping
multiplicando 2.0 con 10.0
ping
dividiendo 20.0 con 5.0
ping
sumando 1.0 con 4.0
ping
restando 5.0 con 10.0
ping
restando -5.0 con -10.0

pedro@DESKTOP-Q3J9AC9: ~/home/Escritorio/DSD/P2/p2-2/gen-java
pedro@DESKTOP-Q3J9AC9:~/home/Escritorio/DSD/P2/p2-2/gen-java$ java Cliente
Introduzca una operación
1 + 2 * 10 / 5 - 10 = -10
ping()
2.0*10.0=20.0
ping()
20.0/5.0=4.0
ping()
1.0+4.0=5.0
ping()
5.0-10.0=-5.0
ping()
-5.0--10.0=5.0
El resultado final es: 5.0
pedro@DESKTOP-Q3J9AC9:~/home/Escritorio/DSD/P2/p2-2/gen-java$
```

Captura de pantalla de ejecución de un ejemplo en el que el usuario trata de dividir entre 0:

```
pedro@DESKTOP-Q3J9AC9: ~/home/Escritorio/DSD/P2/p2-2/gen-py
ping
multiplicando 10.0 con 5.0
ping
dividiendo 1.0 con 0.0
Lanzando excepcion al cliente: No se puede dividir entre 0

pedro@DESKTOP-Q3J9AC9: ~/home/Escritorio/DSD/P2/p2-2/gen-java
pedro@DESKTOP-Q3J9AC9:~/home/Escritorio/DSD/P2/p2-2/gen-java$ java Cliente
Introduzca una operación
10 * 5 - 9 + -11 + 1 / 0
ping()
10.0*5.0=50.0
No se puede dividir entre 0
pedro@DESKTOP-Q3J9AC9:~/home/Escritorio/DSD/P2/p2-2/gen-java$
```