Aplicación cliente-servidor con estado, XML y JSON usando correo electrónico

Descripción

En esta práctica se pide el diseño e implementación de una aplicación cliente-servidor con estado para proporcionar un servicio calculadora. Se parte de la base establecida en la práctica 2 y se pide que el protocolo de comunicación sea de correo electrónico, SMTP para el envío e IMAP para recepción.

Diseño

Sobre el diseño de la segunda práctica, mantenemos una estructura general parecida. En este caso, nuestra aplicación servidor lanzará un único hilo cuyo cometido será el de consultar periódicamente la bandeja de entrada del correo para comprobar si hay mensajes nuevos. Por otro lado, el cliente cada vez que desee realizar una petición, lo hará mediante el envío de un correo electrónico. A continuación, esperara a que el servidor responda con otro correo.

Para el desarrollo, se han creado dos cuentas de correo electrónico Gmail, *servidor.p4.ppc@gmail.com* para el servidor y *cliente.p4.ppc@gmail.com* para el cliente. Mediante estas cuentas la aplicación será capaz de usar el correo electrónico como protocolo de comunicación.

Cuando se desee enviar un correo, en cualquiera de los dos sentidos, la aplicación se conectará a la cuenta correspondiente, compondrá el mensaje y lo mandará al destinatario por medio del protocolo SMTP. En el caso contrario, cuando la aplicación desee consultar si se ha recibido algún mensaje nuevo, se conectará a la cuenta correspondiente con ayuda del protocolo IMAP.

Con respecto a la posibilidad de que existan varios clientes que realizan peticiones de forma simultánea, se ha tomado la decisión de diseño de que cada uno de ellos requiere de una cuenta de correo diferente. Es decir, no se soportan peticiones concurrentes de distintos usuarios que se conecten a una misma cuenta de correo.

Protocolo

A consecuencia de cambiar el protocolo de comunicación, es necesario realizar unas pequeñas modificaciones sobre el protocolo de la práctica 2 del que partíamos. Para la aplicación, se ha decidido que se puede prescindir de la longitud del mensaje y se ha separado el tipo del contenido en sí.

Ayudándonos del multiformato del correo electrónico, ahora se codifica en el campo *Asunto* el formato del mensaje (xml o json) y en el cuerpo es donde se encuentra el mensaje como tal.

Sabiendo esto, queda ver cómo están formados los mensajes JSON y XML, que como se puede ver en los esquemas inferiores, simplemente representan un objeto *Calculator*.

```
{
   "user":"String",
   "operand1":"String",
   "operator":"String",
   "operand2":"String",
   "result":"String"
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<calculator>
   <user>String</user>
   <operand1>String</operand1>
   <operator>String</operator>
   <operand2>String</operand2>
   <result>String</result>
</calculator>
```

Ejemplos de uso

A continuación, se ve como el cliente Jorge introduce directamente la petición que desea mandar al servidor y obtiene la respuesta. Como no se puede apreciar en una simple imagen, es necesario mencionar que, con esta forma de comunicación, existe un retardo apreciable en la recepción de un resultado. Esto es debido a que el servidor realiza el *polling* cada cinco segundos.

```
Enter user name:
Jorge
For the next query, XML or JSON
json
Enter the query or EXIT to end the connection:
1+1
Result: 1+1= 2.0
```

Ilustración 1: ejemplo de uso en el cliente Jorge

Como desde la propia aplicación no somos capaces de ver el funcionamiento del correo electrónico, podemos entrar al correo electrónico desde la aplicación web para ver los mensajes intercambiados.

Si accedemos primero a la cuenta del servidor, encontramos el correo de petición enviado por el cliente desde su cuenta de correo. Nótese el campo resultado a *null.*

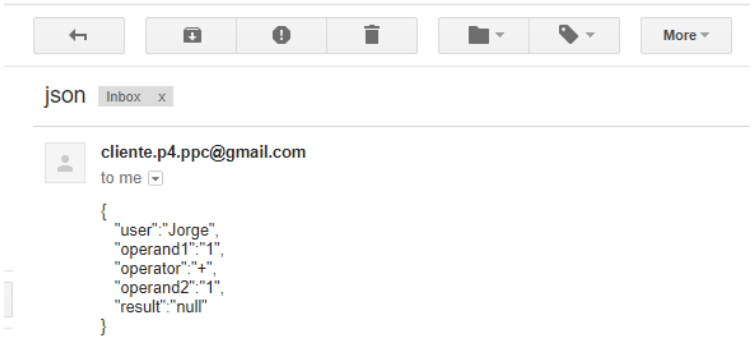

Ilustración 2: detalle del correo de petición

Ahora, entrando en la cuenta del cliente, podemos ver el correo de respuesta donde ya se encuentra el resultado de la operación.

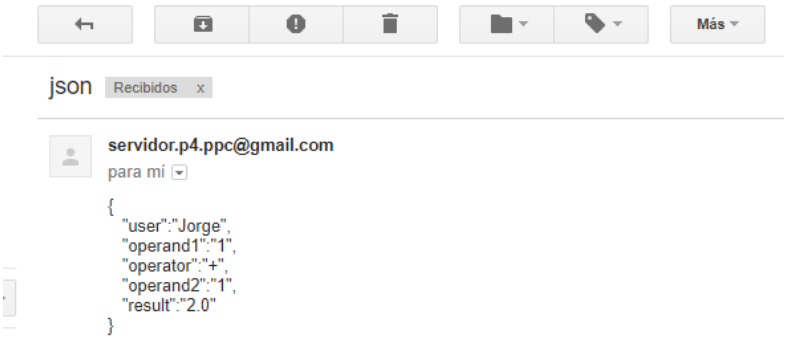

Ilustración 3: detalle del correo de respuesta