

Jorge Ferrini

Elbert Rondón

Carolina Zapata

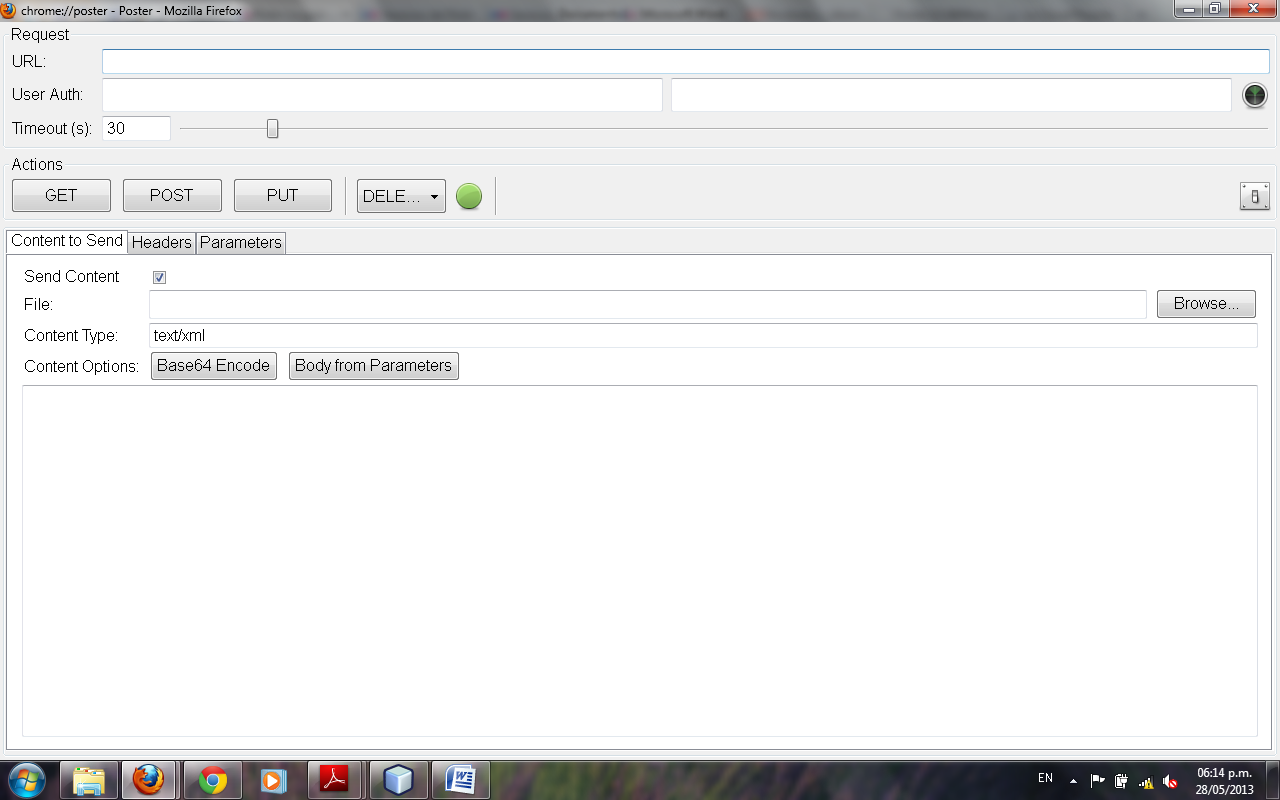
**Documentación del API de Servicio WEB**

Le Cloud Product es un portal WEB destinado para la compra de productos que estén publicados en el sitio. Para que esto sea posible el usuario debe estar registrado en sistema para que la operación sea exitosa.

Basados en los requerimientos del proyecto se procedió a realizar un API de servicios para que los productos que estén publicados en el portal puedan ser solicitados por otros sitios.

Para probar que el Servicio WEB funciona correctamente habrá que instalar un plugin de FireFox llamado “POSTER”. Este se puede instalar directamente desde el navegador (Menú Herramientas 🡪 Complementos) o a través del link:

<https://addons.mozilla.org/es/firefox/addon/poster/>



Interfaz de POSTER.

En el campo de texto llamado URL debemos colocar la dirección de nuestro portal WEB. En nuestro caso sería la siguiente: <http://localhost:3306/GrailsApplication3/productos/rest/>

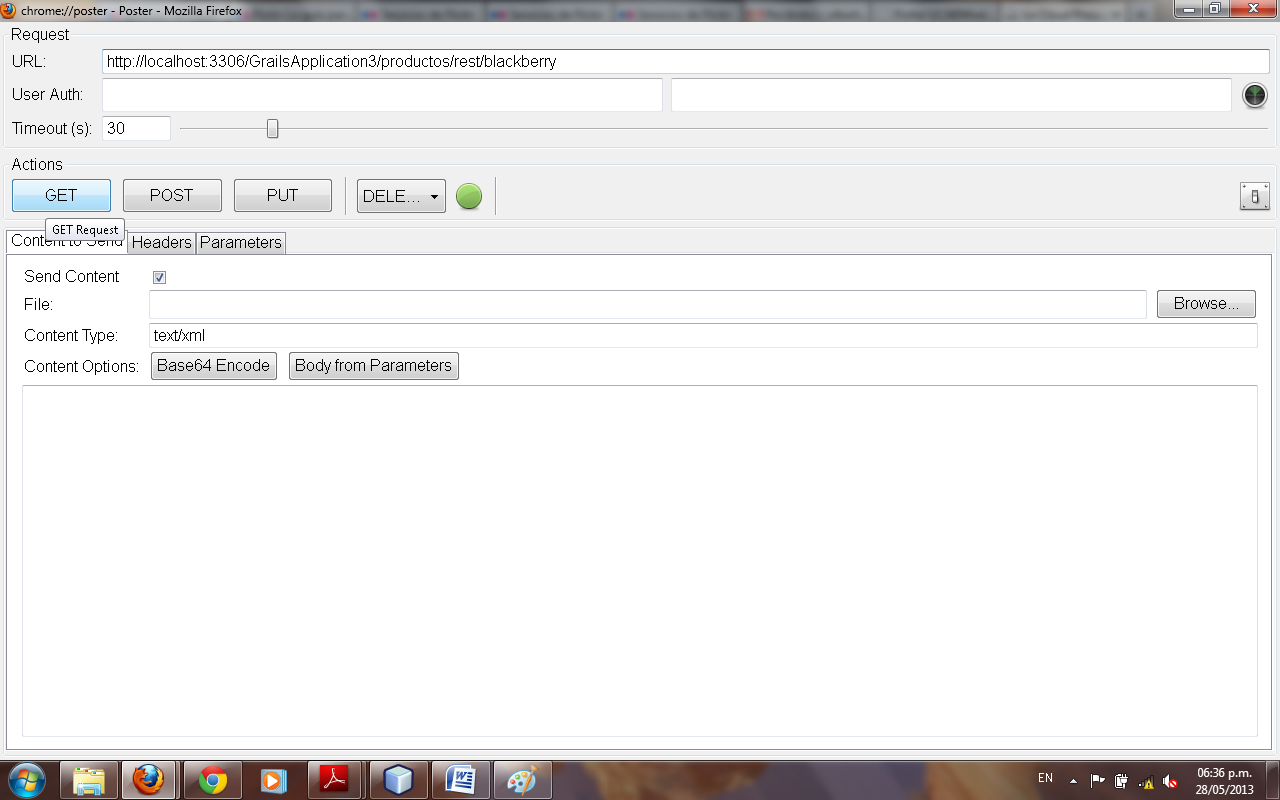
Para probar desde otra máquina habría que cambiar la sección que dice “localhost” por la dirección IP a la que estemos conectados, quedando de la siguiente manera:

[http://192.168.1.101:3306/GrailsApplication3/productos/rest/](http://192.168.1.101:3306/GrailsApplication3/productos/rest/reloj)

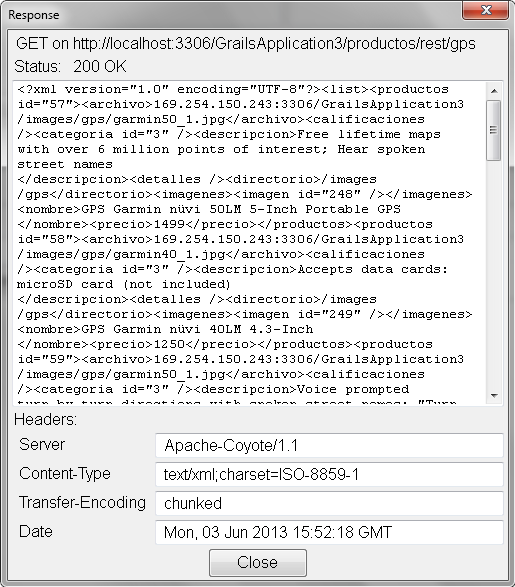
En nuestro API de servicios la búsqueda se realizará proporcionando el nombre del producto, dicho nombre deberá ser escrito al final del URL por ejemplo:

<http://192.168.1.101:3306/GrailsApplication3/productos/rest/gps>

Siendo “gps” el producto que se desea buscar. Finalmente debemos dar Click al botón GET.



Esta acción nos devolverá los productos que se encuentren registrados en nuestro sitio WEB bajo un esquema XML.



La Estructura de este archivo XML para cada producto es el siguiente:

<?xml version="1.0" encoding="UTF-8"?>

<productos id="57">

<archivo>

169.254.150.243:3306/GrailsApplication3/images/gps/garmin50\_1.jpg

</archivo>

<categoria id="3" />

<descripcion>

Free lifetime maps with over 6 million points of interest; Hear spoken street names

</descripcion>

<directorio>/images/gps</directorio>

<imagenes><imagen id="248" /></imagenes>

<nombre>GPS Garmin nüvi 50LM 5-Inch Portable GPS</nombre>

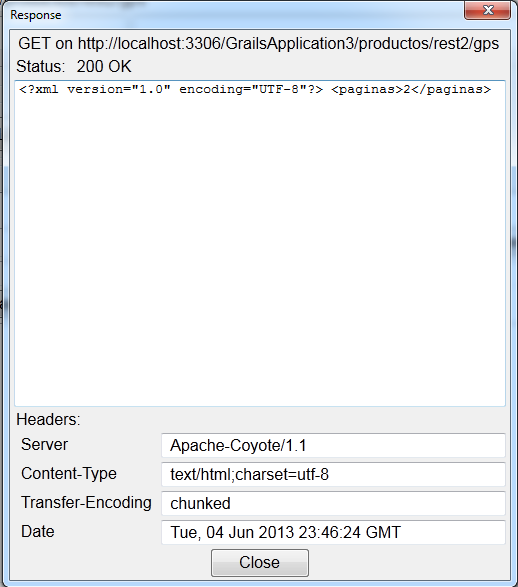
<precio>1499</precio>

</productos>

En nuestro servicio WEB decidimos paginar la búsqueda para que sea más fácil para el usuario observar los productos ubicando un máximo de 10 productos por cada resultado devuelto en la búsqueda. Es por esto que tenemos la opción de dar a conocer la cantidad de páginas con que contiene el producto que se desea buscar, todo esto a través del siguiente URL:

<http://localhost:3306/GrailsApplication3/productos/rest2/gps>

Al momento de hacer click en el método GET, el servicio devolverá un XML con la cantidad de páginas en las que se encuentra el producto “GPS”.



Una vez conocido el número de páginas podemos pasar ese número como parámetro en el URL para ver los productos que se encuentran en cada página. En nuestro caso en cada página se mostraran un máximo de 10 productos.

La Estructura de este archivo XML para la cantidad de páginas es el siguiente:

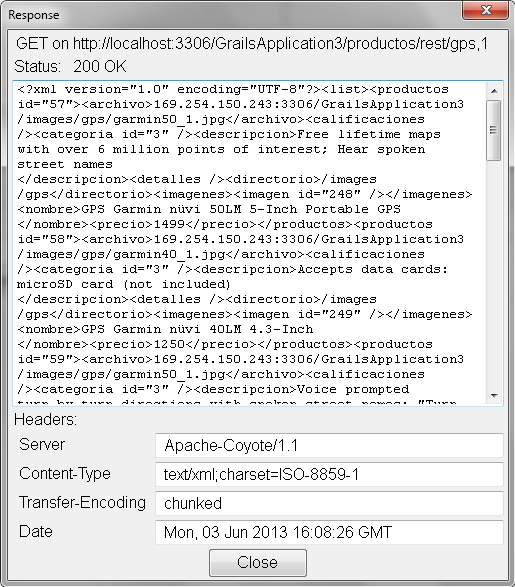
<?xml version="1.0" encoding="UTF-8"?>

<paginas>2</paginas>

El formato del URL para colocar el número de página a visualizar es el siguiente:

<http://localhost:3306/GrailsApplication3/productos/rest/gps,1>

Después de colocar el nombre del producto (gps) agregamos una coma (,) seguido del numero de pagina a observar. El servicio nos devuelve los 10 primeros productos de la pagina 1.



En el caso de poner como número de página “0”, el webservice devuelve la primera página de la búsqueda, en el caso de poner un número mayor al devuelto por el webservice de la cantidad de páginas, se muestra la última página de la búsqueda del producto.

Para este proyecto se implementó la arquitectura de servicios web REST, esta utiliza XML y HTTP. Cada URL representa un objeto sobre el que puedes realizar POST, GET, PUT y DELETE (las operaciones típicas del HTTP).

Algunas ventajas de REST:

* Ligero: no hace falta mucho XML de configuración.
* Resultados legibles.
* Fácil de implementar: no hacen falta herramientas específicas