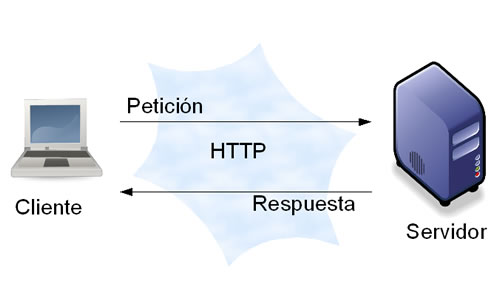
TALLER 1

**GET**

El concepto GET es obtener información del servidor. Traer datos que están en el servidor, ya sea en un archivo o base de datos, al cliente. Independientemente de que para eso tengamos que enviar (request) algún dato que será procesado para luego devolver la respuesta (response) que esperamos, como por ejemplo un identificador para obtener una noticia de la base de datos.

**POST**

Post sin embargo es enviar información desde el cliente para que sea procesada y actualice o agregue información en el servidor, como sería la carga o actualización en sí de una noticia. Cuando enviamos (request) datos a través de un formulario, estos son procesados y luego a través de una redirección por ejemplo devolvemos (response) alguna página con información.



Tanto GET como POST, justamente por ser métodos ambos de HTTP, ejecutan un request y response, y a mi criterio esto suele ser parte de la confusión sobre los objetivos reales de ambos métodos.

Ambos métodos solicitan una respuesta del servidor y ahí es donde parece que los conceptos son iguales ya que con ambos se podría lograr los mismos objetivos.

Generalmente usamos links para ejecutar llamadas GET ya que la idea del link es simplemente “solicitar” una información (página) al servidor y que sea devuelta como una respuesta. Mientras usamos formularios para actualizar datos de productos, clientes, noticias, etc, también teniendo en cuenta que por el método POST también se puede enviar mucha más cantidad de datos que por GET.

**PUT**

La petición HTTP PUT crea un nuevo elemento o reemplaza una representación del elemento de destino con los datos de la petición.

La diferencia entre el método PUT y el método POST es que PUT es un método idempotente: llamarlo una o más veces de forma sucesiva tiene el mismo efecto (sin efectos secundarios), mientras que una sucesión de peticiones POST idénticas pueden tener efectos adicionales, como envíar una orden varias veces.

POST y PUT son muy similares y hay bastante confusión respecto a su uso y sus diferencias. Las dos pueden utilizarse para crear y para actualizar un recurso, pero en situaciones diferentes.

**Delete**

El servicio de integración de datos utiliza el método HTTP Delete para eliminar los datos a través de un servicio web REST. Utilice el método Delete para quitar un recurso o un conjunto de recursos.

Cuando configure la transformación de consumidor de servicio web REST para utilizar el método Delete, debe configurar los puertos de entrada, la entrada del método, la salida del método y los puertos de salida.

**HEAD**

Este método se utilizar para obtener información sobre un determinado recurso sin retornar el registro. Este método se utiliza a menudo para probar la validez de los enlaces de hipertexto, la accesibilidad y las modificaciones recientes.

**CONNECT**

Se utiliza para establecer una comunicación bidireccional con el servidor. En la práctica no es necesario ejecutarlo, si no el mismo API de HTTP se encarga de ejecutarlo para establecer la comunicación previo a lanzar alguna solicitud al servidor.

**OPTIONS**

Este método es utilizado para describir las opciones de comunicación para el recurso de destino. Es muy utilizado con CORS (Cross-Origin Resource Sharing) para validar si el servidor acepta peticiones de diferentes origines.

Los métodos PUT, POST y DELETE se pueden utilizar para modificar los recursos de objeto de negocio y los recursos de estructura de objeto. Sin embargo, las reglas empresariales de un objeto pueden impedir que se actualice mediante una solicitud de la API de REST. Por ejemplo, una solicitud DELETE en un recurso de orden de trabajo puede fallar si las validaciones del objeto de negocio impiden las supresiones a causa del estado actual del objeto de negocio.

En algunas páginas sobre REST encontramos la siguiente definición de los métodos (ejemplo):

* GET para obtener un recurso del servidor
* POST para crear un recurso del servidor
* PUT para actualizar un recurso del servidor
* DELETE para eliminar un recurso del servidor

Y en algunas otras encontraremos exactamente lo contrario (ejemplo):

* GET para obtener un recurso del servidor del servidor
* POST para actualizar un recurso del servidor
* PUT para crear un recurso del servidor
* DELETE para eliminar un recurso del servidor

**Diferencia entre POST y PUT**

PUT se utiliza para poner un recurso en un lugar especificado (la URL). POST es mucho más general. POST simplemente envía una información al servidor para que éste la trate como considere oportuno. Puede crear un recurso nuevo, devolver una información, puede guardarlo en la base de datos, borrarlo, duplicarlo o modificarlo, puede crear 10 recursos de un tipo y dos de otro o puede no hacer absolutamente nada.