¿Qué son los servicios REST?

Autor: José Guillermo Sandoval Huerta

Fecha: 08 noviembre 2023

Índice

Servicios REST

REST (Representational state transfer), también conocidos como Servicios Web RESTful es una arquitectura para el diseño de APIs sobre el protocolo HTTP. La arquitectura del estilo REST principalmente define reglas, comportamiento y restricciones sobre el funcionamiento de una API. Todas las APIs que siguen esta arquitectura son llamadas como “API RESTful”. Los servicios REST nacen como una alternativa a otras arquitecturas de servicios web más complejas, como los servicios SOAP (Simple Object Access Protocol). Donde el termino REST fue acuñado por Roy Fielding, en su tesis doctoral en 2000, quien propuso los principios de la arquitectura REST como una forma de abordar la comunicación en sistemas distribuidos de manera más simple y eficiente. Los servicios REST se inspiran en las características del Protocolo HTTP, que es el protocolo subyacente de la World Wide Web.

Principios REST

Los servicios REST se basan en los siguientes principios fundamentales:

1. **Protocolo HTTP**: Utilizan el protocolo HTTP (Hypertext Transfer Protocol) como protocolo de comunicación. Esto significa que se basan en las operaciones estándar de HTTP, como GET, POST, PUT y DELETE, para realizar acciones sobre los recursos.
2. **Recursos**: En una arquitectura REST, todo es un recurso. Los recursos pueden ser objetos, datos, servicios o cualquier cosa que pueda ser identificada de manera única a través de una URL. Cada recurso tiene una URL única que lo identifica, no puede existir dos o más recursos con el mismo identificador en la red y estos deben mantener una jerarquía lógica.
3. **Operaciones sobre recursos**: Las operaciones que se pueden realizar en los recursos se corresponden con las operaciones HTTP. Por ejemplo, se utiliza GET para obtener información sobre un recurso, POST para crear uno nuevo, PUT para actualizar un recurso existente y DELETE para eliminarlo.
4. **Representaciones**: Los recursos pueden tener múltiples representaciones, como JSON, XML, HTML, etc. El cliente puede especificar qué representación prefiere mediante encabezados HTTP como "Accept" o "Content-Type".
5. **Stateless (sin estado)**: Cada solicitud HTTP contiene toda la información necesaria para entenderla. No se almacena información de estado del cliente en el servidor. Cada solicitud es independiente y auto contenido.
6. **Sistema Cliente-Servidor**: La arquitectura REST sigue el principio de separación entre el cliente y el servidor, lo que permite una mayor escalabilidad y flexibilidad.
7. **Interfaz uniforme**: Se busca mantener una interfaz uniforme para los servicios REST. Esto significa que los recursos se identifican mediante URLs, y las operaciones se realizan utilizando los métodos HTTP estándar. La consistencia en la interfaz simplifica la interacción con los servicios.

Reglas de diseño REST

Las reglas principales del diseño son:

- Uso correcto de verbos y código de estado HTTP.
- Uso de sustantivos en puntos finales y nombres en plural.
- Uso correcto de filtrado, clasificación y paginación.

Verbos y códigos de estado HTTP

Las pautas de REST sugieren usar un verbo HTTP específico dependiendo de la particularidad de la llamada. REST debe respetar tanto los verbos y códigos de estado para cada operación.

Entre los verbos empleados, dependiendo de la acción a realizar en el servidor están:

- **GET**: Solicita información de recursos.
- **POST**: Creación de nuevos recursos.
- **PUT**: Actualiza un recurso existente en su totalidad.
- **PATCH**: Actualiza un recurso existente parcialmente.
- **DELETE**: Elimina un recurso existente.

Los códigos de estado utilizados, dependiendo de la situación:

- **1XX**: Respuestas informativas.
- **2XX**: Peticiones correctas.
- **3XX**: Redirecciones.
- **4XX**: Errores del cliente.
- **5XX**: Errores del servidor.

Entre los estados generados tenemos:

- **200 (OK)**: La solicitud ha sido recibida, entendida y procesada correctamente.
- **201 (Created)**: La solicitud se procesó correctamente y generó un nuevo recurso en el proceso.
- **400 (Bad Request)**: Solicitud no apropiada, faltan requisitos para ser válida.
- **401 (Unauthorized)**: No autorizado credenciales inválidas.
- **403 (Forbidden)**: Prohibido credenciales insuficientes, no tiene permisos para acceder.
- **404 (Not found)**: Recurso no encontrado.
- **500 (Internal server error)**: Hubo un error en el servidor y la solicitud no pudo ser completada.

Uso de los servicios REST

Entre las implementaciones de los servicios REST encontramos:

- **Desarrollo de API**: Los servicios REST son ampliamente utilizados en el desarrollo de API (Interfaces de Programación de Aplicaciones) para permitir la comunicación entre aplicaciones en la web y en dispositivos móviles.
- **Aplicaciones web y móviles**: Los servicios REST son una opción popular para la comunicación entre aplicaciones web y servidores, lo que permite acceder y modificar datos de manera eficiente.
- **Servicios en la nube**: Muchos servicios en la nube ofrecen interfaces RESTful para permitir a los desarrolladores interactuar con sus servicios y datos.
- **Intercambio de datos**: REST se utiliza para el intercambio de datos en formatos como JSON o XML, lo que facilita la interoperabilidad entre sistemas heterogéneos.

Ventajas y desafíos de los servicios REST

Ventajas	**Desafíos**
Simplicidad y facilidad de uso.	No es adecuado para todas las aplicaciones, especialmente aquellas que requieren transacciones complejas o seguridad adicional.
Escalabilidad, ya que no almacena información de estado del cliente en el servidor.	Puede carecer de ciertas características avanzadas de seguridad y autenticación.
Independencia del lenguaje de programación y la plataforma.	
Facilidad de la interoperabilidad y la comunicación entre aplicaciones.	
Ampliamente adoptado y soportado en una variedad de tecnologías y marcos de trabajo.	

Tabla 1. Ventajas y desafíos REST.

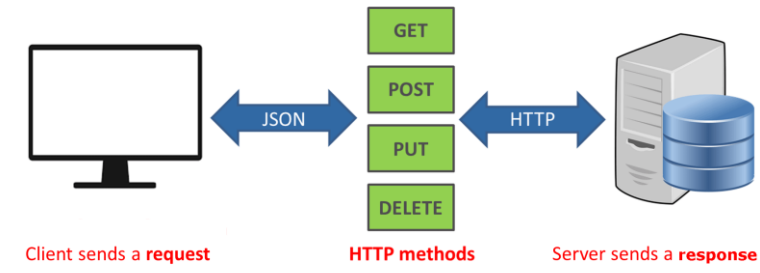

Imagen 1. Petición REST, cliente - servidor.

Referencias

[1] «Fundamentos de la arquitectura REST,» Medium, 22 Agosto 2022. [En línea]. Available: https://medium.com/@diego.coder/introducci%C3%B3n-a-las-apis-rest-6b3ad900acc9#:~:text=REST%20(Representational%20state%20transfer)%20es,el%20funcionamiento%20de%20una%20API.. [Último acceso: 8 Noviembre 2023].