HINTS

APIS RESTFUL

Es una interfaz de programación de aplicaciones (API) que obedece las restricciones de la arquitectura REST, e interactúa con los servicios web RESTful.

La API RESTful también sigue los principios de la API REST, solo que son más escalables, tienen una vida útil más larga, y usa solicitudes HTTP para acceder y usar datos. La API REST y las API RESTful son lo mismo con diferencias mínimas.

DIFERENCIAS BÁSICAS ENTRE REST Y RESTFUL API

En el contexto de REST vs RESTful API, las diferencias básicas entre ellas se explican en las tablas a continuación.

Atributos	API REST	API RESTful
Definiciones	Se utiliza para desarrollar APIS que permiten la interacción entre el cliente y el servidor. Debe usarse para obtener un dato cuando el usuario conecta cualquier enlace a la URL en particular.	Es una aplicación web que sigue la infraestructura REST, y proporciona interoperabilidad entre diferentes sistemas en toda la red.
Servicios Web	El funcionamiento de la URL se basa en solicitud (request) y respuesta (response).	El funcionamiento de RESTful se basa completamente en aplicaciones REST.
Formato de datos	El formato de datos de REST se basa en HTTP.	El formato de datos de RESTful se basa en: JSON, HTTP y Text.
Adaptabilidad	Es altamente adaptable y fácil de usar para todas las empresas comerciales y TI.	Es demasiado flexible en comparación con los servicios web RESTLESS (SOAP, Simple Object Access Protocol, basado en XML).
Protocolo	El protocolo es sólido y hereda muchas medidas de seguridad, que son capas de arquitectura integradas.	Es multicapa, y tiene un protocolo de transporte que hace que el sistema sea menos seguro en comparación con REST.

Ancho de banda	Consume sólo un ancho de banda mínimo.	Consume menos ancho de banda.

REST VS RESTFUL: ARQUITECTURA

Aunque la arquitectura de REST y RESTful API es similar, varían con una ligera diferencia. La API REST sigue todas las reglas de la arquitectura REST, tiene un sistema de capas cliente-servidor, sin estado, almacenable en caché con una interfaz uniforme; mientras que las aplicaciones web RESTful tienen todas las características de la arquitectura REST, con características adicionales únicas. Además, la API REST tiene un sistema separado para manejar la información de la aplicación.

REST VS RESTFUL: CAPACIDAD DE CACHÉ

La API REST muestra la capacidad de almacenamiento en caché como almacenable en caché y no almacenable en caché. Esto proporciona a los clientes y la infraestructura la oportunidad de almacenar cuando sea posible, y permite aumentar el rendimiento del sistema, pues cada vez que el cliente no utiliza los datos almacenados, los datos que no se pueden almacenar en caché se desplazan. Por otro lado, en RESTful API, los clientes pueden acceder a la información de estado constante y almacenable en caché en cualquier momento y lugar.

REST VS RESTFUL: ESTABILIDAD

En la API REST, el cliente debe administrar todas las etapas de la aplicación pues el servidor REST no mantiene ningún tipo de estado del cliente, y debe proporcionar todos los datos necesarios para procesar la solicitud REST API. Cuando hay algún cambio en las API RESTful, es fundamental que el cliente vuelva al sistema de almacenamiento. El intercambio de datos y estado entre el cliente y el servidor establece la estabilidad de los sistemas REST. En los servicios RESTful, la implementación de datos está oculta, mientras que los demás datos no lo están.

REST VS RESTFUL: ARQUITECTURA MULTICAPA

La restricción en REST, es que los dispositivos del sistema no pueden ver más allá de la capa. Esto se supera combinando los balanceadores de carga y algún proxy, para elevar el rendimiento y la seguridad del sistema. En el caso de la API RESTful, los límites son fuertes, la separación entre ellos es clara, y esto se debe a que la arquitectura en capas construida sobre el cliente-servidor tiene límites sin estado. El flujo de datos a través de los límites es administrado por el cliente en función de sus requisitos, y es su responsabilidad mostrar o manipular los datos.

REST VS RESTFUL: INTERFAZ SIN DESVIACIONES

Es un requisito muy importante tener una interfaz sin desviaciones. Esta interfaz sin desviaciones entre los sistemas presentes en la arquitectura REST, actúa como una característica única para segregarla de otras arquitecturas independientes. Ambos servicios REST manejan datos como un recurso con un espacio de nombres único, e individual.

REST VS RESTFUL: LLAMADA DE PROCEDIMIENTO REMOTO EN SERVICIOS WEB

Cuando el servicio no está en REST, busca automáticamente los servicios de verbo HTTP o URI en funcionamiento. Esto se denomina conjunto uniforme de recursos, y denota la representación estructural de los datos REST. Esta separación entre cada capa se conoce como REST, y también se conocen como llamadas a procedimientos remotos. La consulta sobre HTTP `POST` y HTTP `GET` con un enlace URL se utiliza para publicar un archivo, y configurar su contenido. Las API RESTful utilizan servicios como: `PUT`, `DELETE`, `GET`, `POST` y `PATCH`, para realizar las acciones HTTP.

Operaciones CRUD de APIs RESTful:

OPERACIÓN	SQL	HTTP
Crear	INSERT	PUT/POST
Leer	SELECT	GET
Actualizar	UPDATE	PUT/PATCH
Eliminar	DELETE	DELETE

CRUD es un acrónimo de los comandos Crear, Leer, Actualizar y Eliminar. Estas cuatro funciones principales guían a los desarrolladores de software a interactuar con las bases de datos. A pesar de su origen en bases de datos, ahora mapea el principio de diseño de aplicaciones dinámicas como: HTTP, SQL y DDS. Las operaciones CRUD son cíclicas, en lugar de ser un sistema arquitectónico. A continuación, se muestran las tareas realizadas por ellas:

- **Crear:** es un procedimiento que genera nuevos registros.
- **Lectura:** es un procedimiento utilizado para leer/recuperar datos, en función de los parámetros de entrada deseados.
- **Actualizar:** este es un procedimiento utilizado para modificar registros (sin sobrescribir).
- **Eliminar:** es un procedimiento utilizado para eliminar (una o más) entradas por completo.

DIFERENCIA ENTRE LA API REST Y LA API SOAP

No existe una comparación directa entre las API SOAP y REST, pero si hay algunos puntos que ayudan a elegir mejor entre estos dos servicios web. Se enumeran a continuación:

1. SOAP significa Protocolo Simple de Acceso a Objetos; y REST significa Transferencia de Estado Representacional.
2. SOAP es un protocolo, sigue un estándar estricto para permitir la comunicación entre el cliente y el servidor; mientras que REST es un estilo arquitectónico que no sigue ningún estándar estricto, sino seis restricciones definidas por Roy Fielding en 2000, y son: interfaz uniforme, cliente-servidor, sin estado, almacenamiento en caché, sistema en capas, y código bajo demanda.
3. SOAP usa solo XML para intercambiar información en su formato de mensaje; mientras que REST no está restringido a XML, y es la elección de implementar qué tipo de medio usar, como: XML, JSON, texto sin formato. Además, REST puede usar el protocolo SOAP, pero SOAP no puede usar REST.
4. En el nombre de las interfaces de servicios para la lógica empresarial, SOAP usa `@WebService`; mientras que REST, en lugar de usar interfaces, usa URI como `@Path`.
5. SOAP es difícil de implementar, y requiere más ancho de banda; mientras que REST es fácil de implementar y requiere menos ancho de banda, como los teléfonos inteligentes.
6. SOAP tiene beneficio sobre REST, ya que contiene una transacción de cumplimiento ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad). Algunas de las aplicaciones requieren capacidad de transacción que es aceptada por SOAP, mientras que REST carece de ella.
7. Sobre la base de la seguridad, SOAP tiene SSL (Secure Socket Layer) y WS-security; mientras que REST tiene SSL y HTTPS. En el caso de contraseña de cuenta bancaria, número de tarjeta, entre otros, se prefiere SOAP a REST. El problema de la seguridad tiene que ver con los requisitos de su aplicación, pues debe crear la seguridad por su cuenta. Se trata de qué tipo de protocolo se usará.