Blog sobre Java EE

API REST estilos y homogeneidad

5 marzo, 2020 Por Cecilio Álvarez Caules — 2 comentarios

El diseño de un API REST **siempre es complejo** y siempre surgen muchas muchas dudas . Esto es debido a que no siempre hay una única solución a una casuística determinada y se pueden aplicar varios enfoques . Vamos a hablar de un ejemplo sencillo que nos ayude a entender el concepto de homogeneidad y como esta nos pueda ayudar en nuestro día a día.

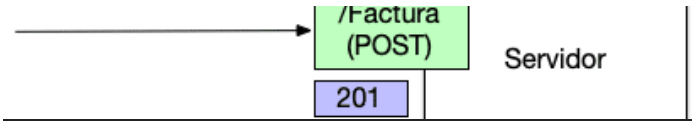
Normalmente cuando disponemos de un Servicio REST una de las operaciones más habituales es realizar una petición POST al servidor para insertar un recurso . Podríamos por ejemplo enviar una Factura vía JSON e insertarla a través del uso del verbo POST.

API REST y opciones

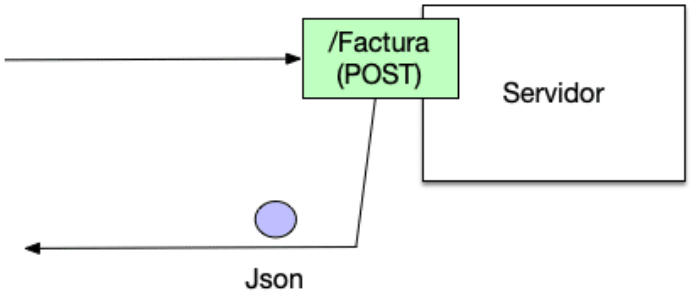
¿Ahora bien qué es lo que devolvemos una vez la inserción se ha realizado?. Existen varias opciones y todas ellas son posibles.

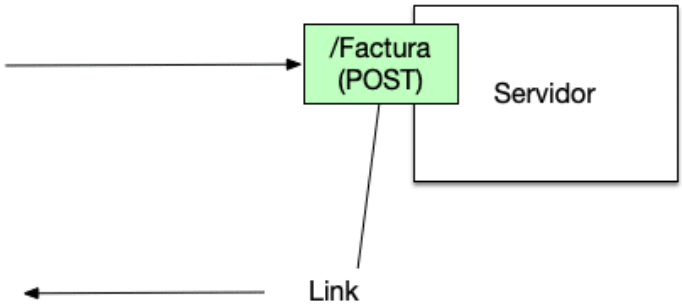
1. No devolver nada excepto un codigo 201 de que el recurso se ha creado correctamente.
2. Devolver la entidad en formato JSON de tal forma que podamos usarla en posteriores operaciones
3. Devolver un link como cuerpo del mensaje que nos ligue a la url del nuevo recurso.
4. Devolver en la cabecera un location header con un link al nuevo recurso

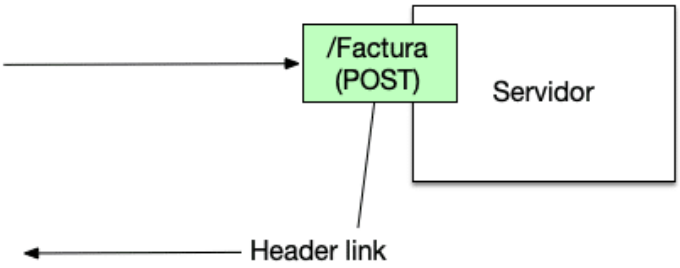
Todas las opciones son posibles



Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.
plugin cookies
ACEPTAR



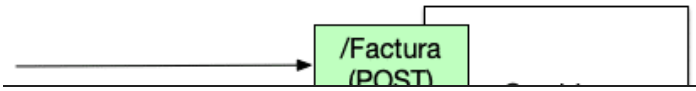




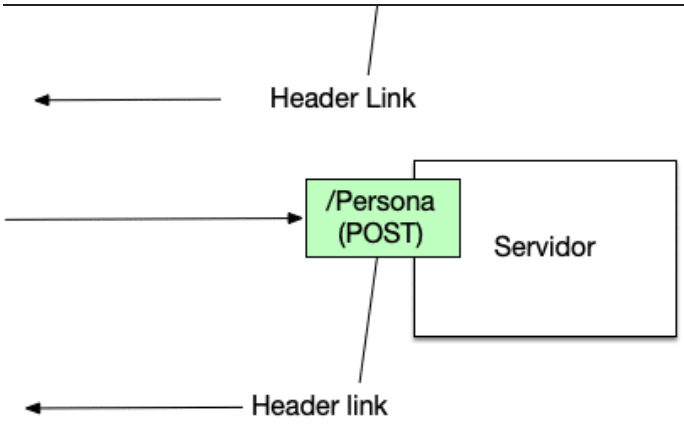
Es cierto que probablemente la cuarta opción es la que más ligada esta a lo que la especificación indica . Lo mejor es devolver un link en un location header. Sin embargo yo me he encontrado en situaciones en las que esta información pasa completamente desapercibida para el usuario del API ya que no dispone de los conocimientos para suponer que le devolvemos la entidad en el header.

¿Es entonces mejor devolver la propia entidad como parte de la respuesta en formato JSON ? . Me gustaría poder decir que es mejor .. o que es claramente peor . Pero la realidad es que ambas opciones son opciones “validas” .

Eso sí a la hora de diseñar el API REST lo que si es importante es el concepto de homogeneidad es decir una vez que hayamos elegido nuestra opción todos los servicios REST se deben



Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.
plugin cookies
ACEPTAR



El diseño de un API REST es algo que hay que pensar con calma ya que se genera un acoplamiento fuerte.

Otros artículos relacionados

- ¿Que es REST?
- REST API Design y Simplicidad
- REST Recursos Anidados
- REST

Descargar PDF

Cecilio Álvarez Caules

Cecilio Álvarez Caules Oracle Java Certified Architech

Archivado en: Sin categoría

Comentarios

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.
plugin cookies
ACEPTAR

7 marzo, 2020 al 0:42

Hola Cecilio,

Un comentario y una duda.

Respecto a la cuarta forma de devolver el link en el header, entiendo que si el api rest tiene documentación, no debería haber problema en que el cliente supiera consumirlo correctamente.

Y la duda es
En caso de devolver el Link, independientemente como... No implicaría que el back tuviera que saber la estructura de urls del front? En ese caso quedarían un poco acoplados, y en caso de cambio de urls en el front, implicaría cambiar también el back y al reves

Gracias

Responder

Cecilio Álvarez Caules dice
7 marzo, 2020 al 9:01

La url vendría en la cabecera de location . Es decir el cliente leería esa cabecera dinamicamente . Si en una nueva petición la cabecera se actualiza el cliente queda actualizado.

No se si me explico ?

Responder

Deja un comentario

Tu dirección de correo electrónico no será publicada.

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.

plugin cookies

ACEPTAR

Nombre

Correo electrónico

Web

He leído y acepto la Política de privacidad de esta web *

PUBLICAR EL COMENTARIO

Este sitio usa Akismet para reducir el spam. Aprende cómo se procesan los datos de tus comentarios.

BUSCAR

Buscar en esta web

LOS CURSOS
POR

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.
plugin cookies
ACEPTAR

Nuevo Curso
PROGRAMACIÓN REACTIVA SPRING WEBFLUX
50%

SPRING BOOT
50%
JAVA8
LAMBDA Y STREAMS
50%

Cursos Gratuitos

- Introducción AJAX
- Java Lambda
- Introducción a Spring Boot
- Introduccion a Angular
- Introduccion Typescript
- Introducción a JPA
- Java y Herencia

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.

plugin cookies

ACEPTAR

Ajax JSON y Promesas

Java 8 Lamdbas Streams y buenas prácticas 50%

Curso de Angular 8 con TypeScript

Curso de Spring Boot 50%

Curso de Typescript , un lenguaje moderno

Curso Arquitectura Java Solida con Spring

Curso Programacion Orientada Objeto Java

Curso de Java Standard Edition APIS

Curso Desarrollo Web en Java

Pack SpringBoot + Arquitectura Solida

Pack TypeScript+ Angular

CONTACTO

contacto@arquitecturajava.com

LO MÁS LEIDO

¿Qué es Spring WebFlux?

¿Javascript sincrono o asincrono?

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.

plugin cookies

ACEPTAR

CLASICOS

Uso de Java Generics (I)

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información.

plugin cookies

ACEPTAR

Copyright © 2020 · Cecilio Álvarez Caules, todos los derechos reservados. Aviso legal, política de privacidad y cookies