API REST

API (interfaz de programa de aplicación)

Una puerta de enlace entre los clientes y los recursos del software.

Modelo-Vista-Controlador (MVC)

- Modelo: Representa los datos de la aplicación y la lógica de negocio. Es responsable de gestionar los datos y las reglas que los rigen.

- Vista: Representa la capa de presentación y es responsable de mostrar los datos al usuario. Maneja los elementos de la interfaz de usuario y el renderizado.

- Controlador: Actúa como intermediario entre el Modelo y la Vista. Recibe la entrada del usuario, la procesa y gestiona el flujo de datos entre el Modelo y la Vista.

REST

"Representational State Transfer" (transferencia de estado representacional), es una arquitectura de software que impone condiciones sobre cómo debe funcionar una API.

Principios de la API REST

Interfaz uniforme

Todas las solicitudes de API para el mismo recurso deben tener el mismo aspecto, sin importar de dónde provenga la solicitud. La API REST debe garantizar que un mismo dato pertenezca a un único identificador uniforme de recursos (URI). Los recursos no deben ser demasiado grandes, pero deben contener toda la información que el cliente pueda necesitar.

(Mapping)

Desacoplamiento cliente-servidor

En el diseño de la API REST, las aplicaciones de cliente y de servidor deben ser completamente independientes entre sí. La única información que la aplicación de cliente debe conocer es el URI del recurso solicitado; no puede interactuar con la aplicación de servidor de ninguna otra manera. Del mismo modo, una aplicación de servidor no debe modificar la aplicación de cliente más que pasándole los datos solicitados a través de HTTP.

(Nuestras URIs: URLs)

Sin estado

Las API REST no tienen estado, lo que significa que cada solicitud debe incluir toda la información necesaria para procesarla. En otras palabras, las API REST no requieren ninguna sesión del lado del servidor. Las aplicaciones de servidor no pueden almacenar ningún dato relacionado con una solicitud de cliente.

Capacidad de almacenamiento en caché

Cuando sea posible, los recursos deben almacenarse en caché en el lado del cliente o del servidor. Las respuestas del servidor también deben contener información sobre si se permite el almacenamiento en caché para el recurso entregado. El objetivo es mejorar el rendimiento del cliente, al tiempo que aumenta la escalabilidad en el lado del servidor.

```
@Cacheable("numerito")
    public int computarNumero(int i) {

        // ...

    }
```

Esta cosa existe

Arquitectura del sistema en capas

En las API REST, las llamadas y respuestas pasan por diferentes capas. Como regla general, no asuma que las aplicaciones de cliente y de servidor se conectan directamente entre sí. Puede haber varios intermediarios diferentes en el bucle de comunicación. Las API REST deben diseñarse de modo que ni el cliente ni el servidor puedan saber si se comunica con la aplicación final o con un intermediario.

Código bajo demanda (opcional)

Las API REST suelen enviar recursos estáticos, pero en ciertos casos, las respuestas también pueden contener código ejecutable (como applets de Java). En estos casos, el código solo debe ejecutarse a petición.

~FIN ~