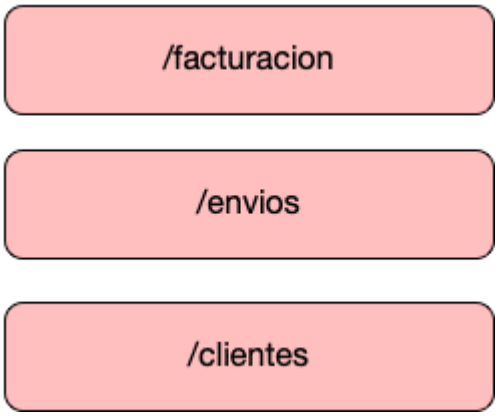
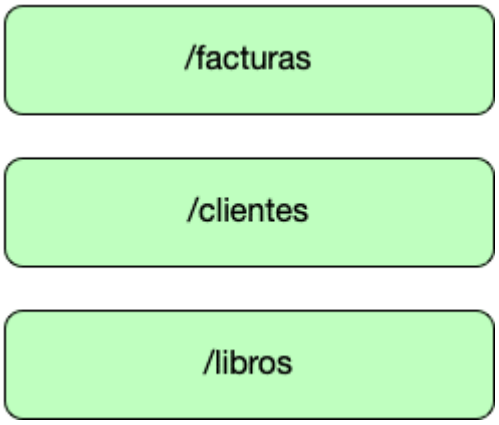
El principio de HATEOAS o Hypermedia as the Engine of Application State es un principio de diseño a nivel de APIS REST . ¿Como funciona exactamente? . Normalmente cuando diseñamos un API REST existen varios niveles en los cuales nosotros podemos diseñar el API

[ihc-hide-content ihc_mb_type=”show” ihc_mb_who=”4″ ihc_mb_template=”1″]

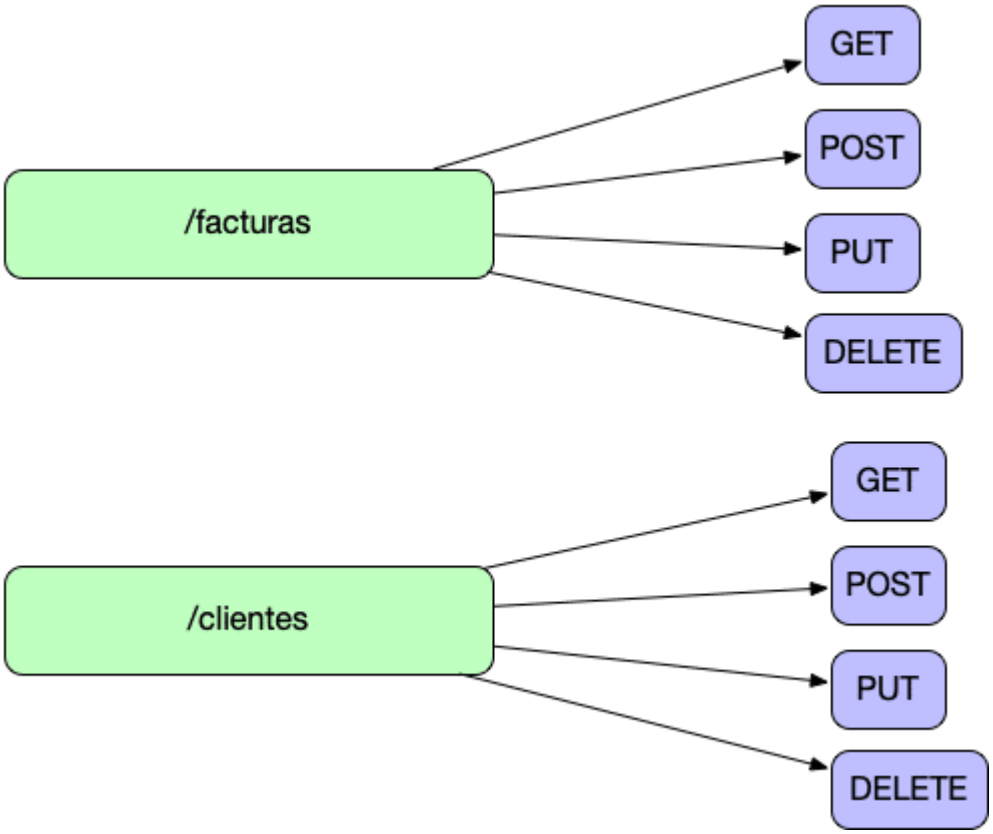
Nivel 0 (Swamp of Pox) : Este nivel hace referencia a cuando nosotros no tenemos ningún patrón o principio a la hora de publicar URLS y se admite cualquier situación caótica. Es decir una url se puede denominar /facturación otra /envios otra /clientes etc. No hay ningún tipo de principio o homogeneidad a la hora de trabajar.



Nivel 1 : Nivel de Recursos , en este nivel el diseño del API REST avanza y nos encontramos ante una situación en la cual definimos el concepto de Recurso un recurso es una entidad que puede ser seleccionada insertada buscada y borrada a nivel web. Recursos serían /facturas , /clientes , /libros etc. Este es un avance ya que empezamos a generar homegeneidad a la hora de como se comportan nuestras urls.



Nivel 2 : Verbos Http en este nivel nos encontramos con que se definen a traves de verbos HTTP las operaciones básicas contra un Recurso . Es decir si queremos realizar una operación de inserción es un POST y queremos selecionar registros es GET , si queremos Borrar es DELETE y si queremos actualizar es PUT. Cada uno de los verbos se encarga de una de las operaciones .



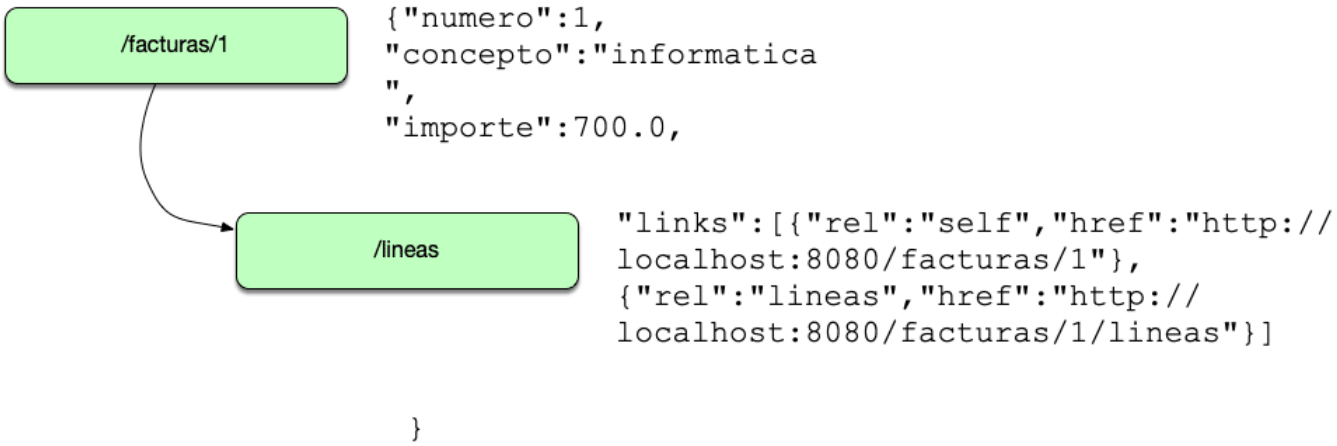
Nivel 3 : HATEOAS este nivel es el de Hipermedia y hace refecencia a como unos recursos REST se asocian con otros . Es decir normalmente cuando nosotros mostramos la información de una factura a nivel de REST nos encontraremos con datos en formato JSON que nos son devueltos con los contenidos de las facturas.

Veamos el codigo:

```
{"numero":1,"concepto":"informatica","importe":700.0}
```

Usando HATEOAS

Ahora bien hay situaciones y estructuras REST complejas que pueden sacar Partido del principio de HATEOAS y añadir a los resultados que se devuelven de los Recursos links a otros recursos adicionales como pueden ser links a las LineasFacturas .



Veamos como queda el retorno en un ejemplo clásico de Spring Hateoas

```
{"numero":1,"concepto":"informatica","importe":700.0,"links":[{"rel":"
self","href":"http://localhost:8080/facturas/1"},{"rel":"lineas","href
":"http://localhost:8080/facturas/1/lineas"}]}
```

De esta forma podemos navegar de una forma mucho más natural entre Recursos REST sin tener que conocer cada una de las URLS reduciendo el acoplamiento entre las diferentes partes.

Conclusiones

Eso sí a la hora de implementar HATEOAS tendremos siempre que tener en cuenta que también implica una mayor complejidad a la hora de construir y gestionar los Recursos que tenemos por lo tanto antes de implementarla hay que pensarlo a detalle y hacerlo desde el primer momento.

Otros artículos relacionados

- REST URL formatos y buenas prácticas.
- Diseño API REST , rendimiento y OverFetching
- Spring Boot REST JPA y JSON
- ¿Que es REST?

[/ihc-hide-content]