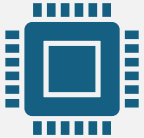
Introducción a REST API

¿Qué es una REST API?

Application Programming Interface (API)

Respresentational State Transfer (REST)



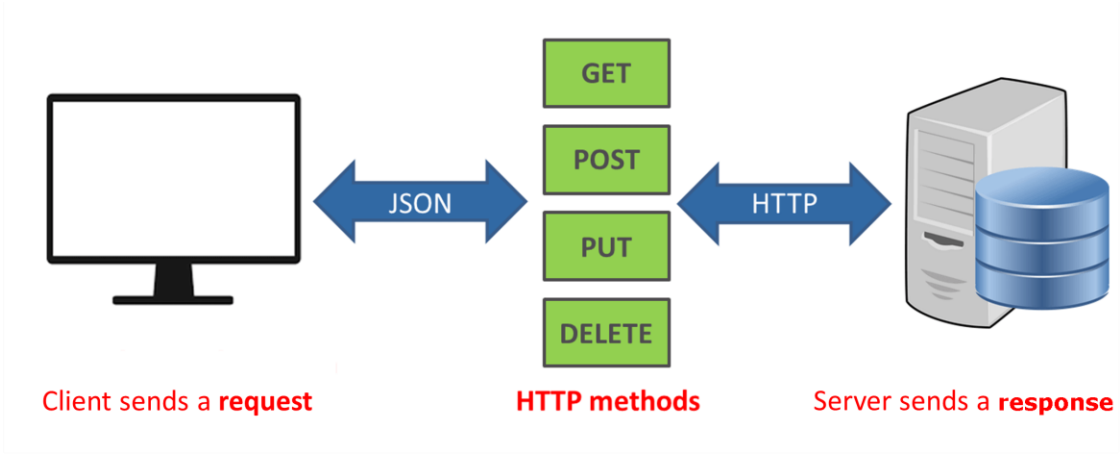
REST es una arquitectura de API que respeta unos principios de diseño - No es un protocolo

Las respuestas de la API representan el estado de un recurso en el momento que se realiza la solicitud

¿Qué debe cumplir una API para ser considerada REST?

ARQUITECTURA **CLIENTE-SERVIDOR** QUE TRABAJA SOBRE HTTP

LA COMUNICACIÓN ENTRE CLIENTE Y SERVIDOR ES **STATELESS**. CADA REQUEST ES INDEPENDIENTE Y NO GUARDA INFORMACIÓN DE LOS CLIENTES ENTRE REQUESTS



¿Qué debe cumplir una API para ser considerada REST?

Interfaz uniforme de todos los componentes, de este modo la información se transmite de foma estándar

- Los **recursos** deben ser **identificables**
- Las respuestas del servidor deben ser **descriptivas**. El cliente debe tener información suficiente para procesar las respuestas y construir nuevas solicitudes

Si una API implementa la arquitectura REST podemos decir que la API es **RESTful**

Pasos para diseñar una API REST

1. **Modelado de los objetos**: identificar los objetos que serán presentados como recursos
2. **Crear la URI de los recursos**: decidir las URI que se expondrán como endpoints de la API
3. **Representación de los recursos**: definir la estructura de los recursos que se retornarán a los clientes en cada URI
4. **Asignar verbos HTTP a cada URI**: de las operaciones presentes en la API, mapear las URI con verbos HTTP

Definición de URIs para los recursos

- Dado el modelo **auto** (car) como recurso de mas alto nivel
- Dado el modelo **service** como sub recurso de auto

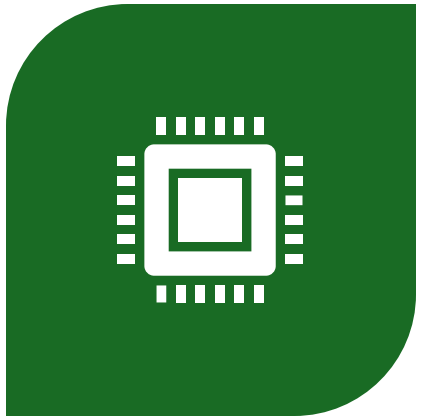
```
/cars
/cars/{id}

/services
/services/{id}

/cars/{id}/services
/cars/{id}/services/{serviceId}
```

Ejemplos de asignación de verbos HTTP

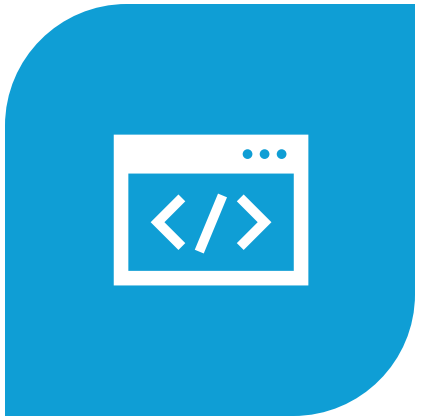
URI	Verbo	Acción
/cars	**POST**	Crear un nuevo auto
/cars	**GET**	Obtener todos los autos
/cars/{id}	**GET**	Obtener un auto por su id
/cars/{id}	**PUT**	Actualizar el auto con id = {id}
/cars/{id}/model	**PATCH**	Actualizar una propiedad especifica (modelo) del auto con id = {id}
/cars/{id}	**DELETE**	Eliminar el auto con id = {id}
/cars/{id}/services	**POST**	Crear un service y asignarlo al auto con id = {id}
/cars/{id}/services	**GET**	Obtener todos los services del auto con id = {id}

Mejores practicas al desarrollar una API

EL MODELO NO DEBE SER EXPUESTO. UTILIZAR DTOS, LOS CUALES LLAMAREMOS **CONTRATOS**



TENER UNA **CAPA DE SERVICIOS** PARA GESTIONAR TODA LA LÓGICA DEL NEGOCIO



LAS **INTERFACES** DE TODOS LOS COMPONENTES DEBEN SER **UNIFORMES**