APIs

API sígnica Application Programming Interface. Es una URL que devuelve información para que otro sistema lo consuma. Son desarrolladas para que dos sistemas puedan comunicarse.

Hay APIs públicas (RESTCountries), semipúblicas (Twitter), y privadas (Netflix).

**Endpoint:** es un punto de conexión donde necesitamos apuntar para obtener la información que queremos. Es decir, son las URLs que debemos utilizar para obtener la información de un servidor a través de una API.

**API REST:**

es un sistema a través del cual un front-end se puede comunicar con un back-end de una manera mucho más organizada y funcional. Siempre pensando en la optimización y velocidad del sistema, así como también en la facilidad de uso del mismo.

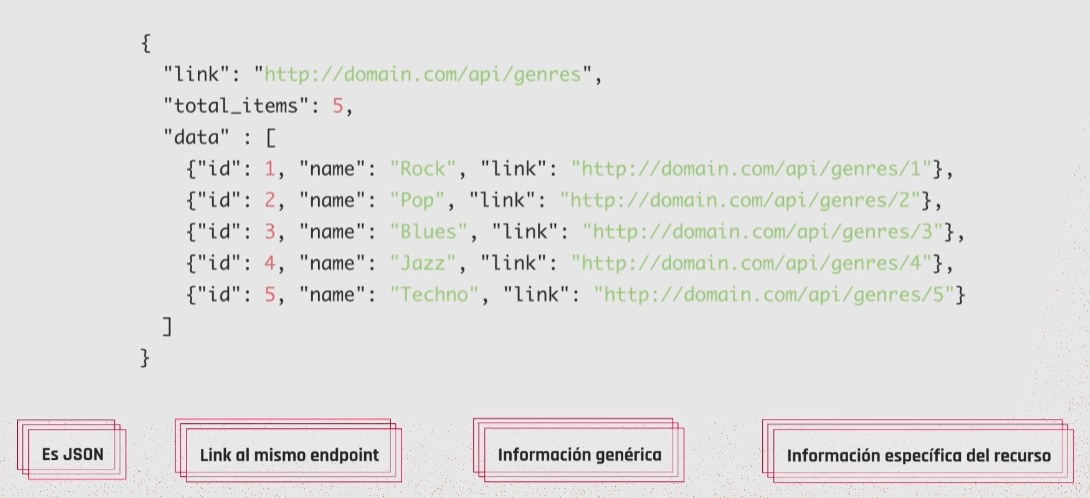
Un **sistema REST** busca implementar un esquema o protocolo que le permita a todos los sistemas que se comunican con él entender en qué forma lo tienen que hacer y bajo qué estructura deberán enviar sus peticiones para que sean atendidas.

**Características:**

1. Separa la aplicación web en dos: por un lado, lo que es la interfaz de usuario en una aplicación, y por el otro tener todo lo que la aplicación provee como servicio que la interfaz consume.
2. La ubicación de los recursos: dentro de la comunicación entre el cliente y el servidor surge muchas veces la necesidad del cliente de acceder a recursos del servidor. Lo que hace REST es que si por ejemplo queremos en Spotify acceder a las canciones solo podríamos llegar allí a través de una dirección que por ejemplo sea /canciones.
3. Stateless o “sin estado”: REST nos propone que el servidor no almacene ningún dato acerca de las peticiones que haga el cliente. De esta manera, cada solicitud será tratada como nueva y de forma independiente. No existe el concepto de sesión de usuario que recuerda cosas.
4. Cacheable: Si alguna información permanece sin modificarse por mucho tiempo, el cliente podría recordar esa respuesta para no tener que realizar constantemente el pedido, eliminando esa carga al servidor. Para que esto suceda el servidor debe responder enviando un encabezado que se llama “max-age”, cuyo valor es la cantidad de segundos que tiene de validez ese recurso. Una vez expirado el cliente debe volver a pedir de nuevo el recurso.

**DESARROLLO DE UNA API:**

1. **Identificar los objetos que serán presentados como recursos:** como por ejemplo, artistas, álbumes, canciones, géneros. Hay que identificarlos como sustantivos.
2. **Crear los identificadores de estos recursos – URL o endpoints:** /álbumes, /canciones, etc. Esto nos devuelve toda la información. Este tipo de recurso donde accedo a un conjunto de elementos se le llama **Colecciones**. Por separado se llaman **Recursos**. Podemos acceder a un recurso determinado por su id, ej: /álbumes/5, pero también podemos acceder desde un artista, ej: /artista/2/álbumes/5.
3. **¿Qué tiene que devolver el endpoint?** Devuelve un JSON con la información pedida. Contiene una clave link con la ruta al JSON que acabamos de consumir, devuelve información genérica como la cantidad de datos de géneros totales por ejemplo, e información específica del recurso.



1. **Asociar a los endpoints con los métodos HTTP:**