

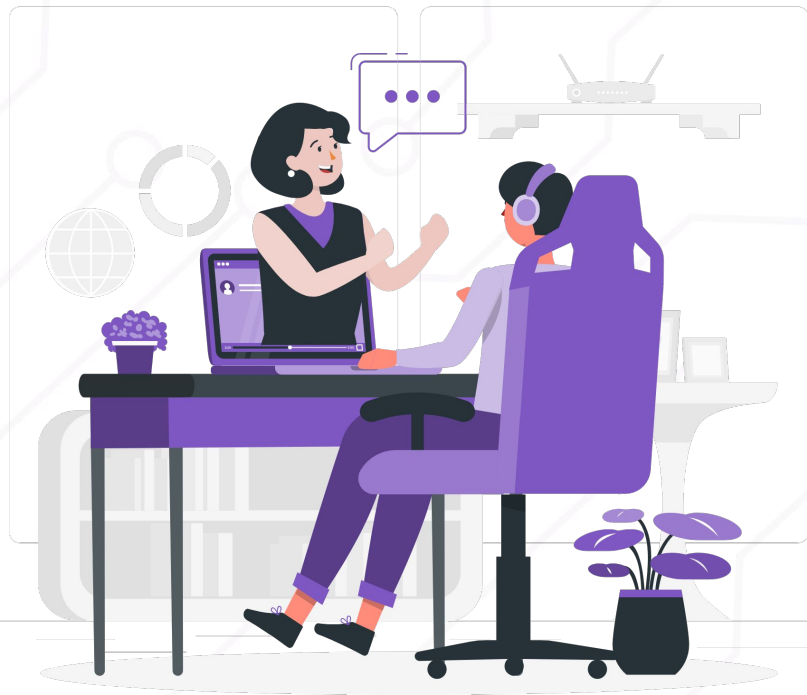
HTTP METHODS

DEV.F
DESARROLLAMOS(PERSONAS);

dev

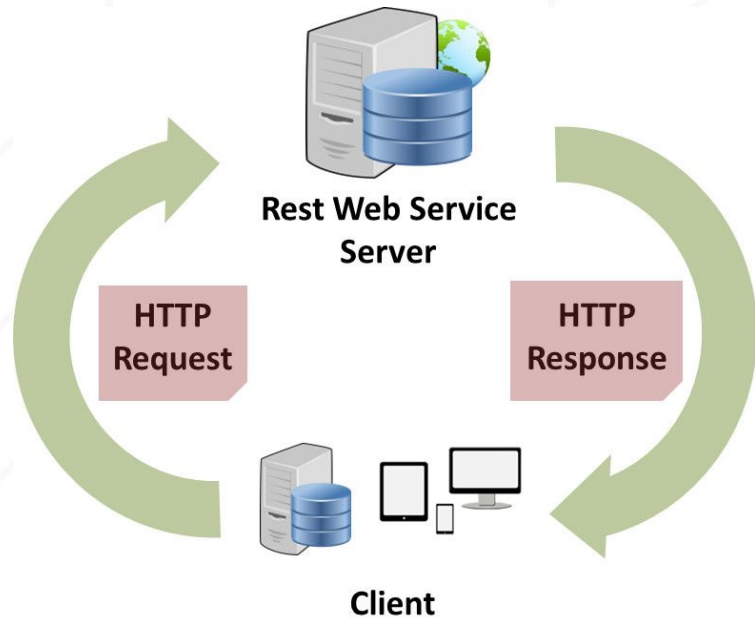
Objetivo de la sesión

- Veremos qué es HTTP
- Hablaremos de peticiones y tipos de peticiones en los métodos HTTP.
- Veremos Status Code y la importancia de ellos.
- Veremos Herramientas para hacer peticiones a una API y así mismo consumir una API



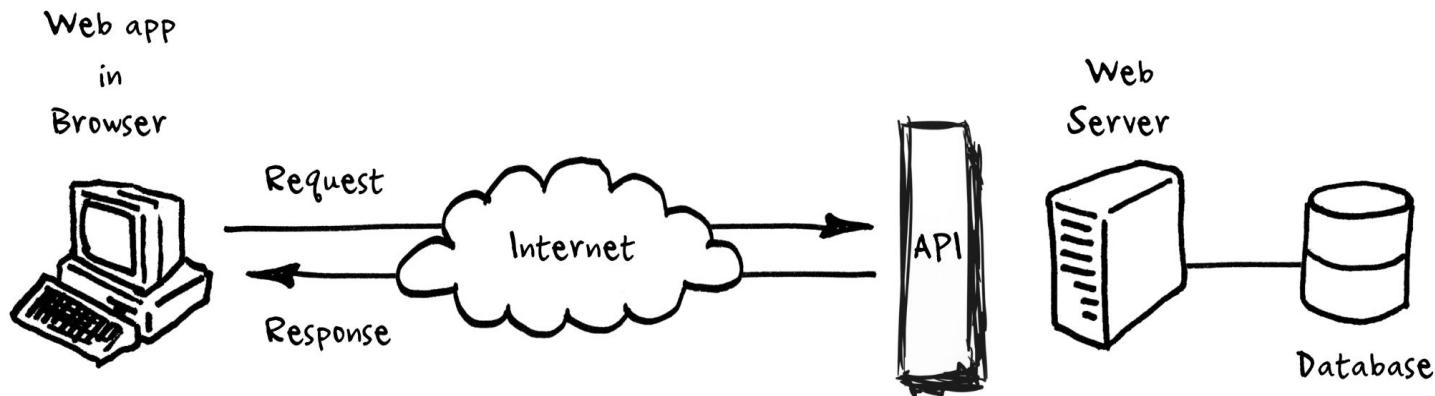
Métodos de petición HTTP

HTTP es un protocolo, o un conjunto definido de reglas, para acceder a recursos en la web. **Los recursos pueden ser cualquier cosa, desde archivos HTML hasta datos de una base de datos, fotos, texto, etc.**



Métodos de petición HTTP

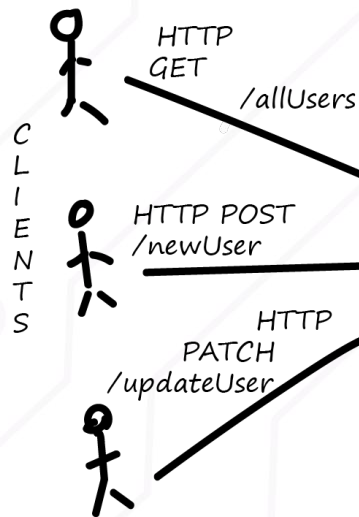
Estos recursos se ponen a nuestra disposición a través de una API y nosotros hacemos peticiones a estas APIs a través del protocolo HTTP. API significa interfaz de programación de aplicaciones. **Es el mecanismo que permite a los desarrolladores solicitar recursos.**



Mensajes HTTP

Los mensajes HTTP, son los medios por los cuales se intercambian datos entre servidores y clientes.

Rest API Basics



Our Clients, send HTTP Requests and wait for responses

Rest API

Recieves HTTP requests from Clients and does whatever request needs. i.e create users

Typical HTTP Verbs:
GET -> Read from Database
PUT -> Update/Replace row in Database
PATCH -> Update/Modify row in Database
POST -> Create a new record in the database
DELETE -> Delete from the database

Database



Our Rest API queries the database for what it needs

Response: When the Rest API has what it needs, it sends back a response to the clients. This would typically be in JSON or XML format.

Petición - Respuesta

La línea de inicio y las cabeceras HTTP, del mensaje, son conocidas como la cabeza de las peticiones, mientras que su contenido en datos se conoce como el cuerpo del mensaje.

Petición

Verbo Recurso Versión

↑ ↑ ↑

```
GET /index.html HTTP/1.1
Host: wikipedia.org
Accept: text/html
```

Primera línea

Encabezados

Cuerpo

Respuesta

Versión Código respuesta

↑ ↑

```
HTTP/1.1 200 OK
Server: wikipedia.org
Content-Type: text/html
Content-Lenght: 2026

<html>
...
</html>
```

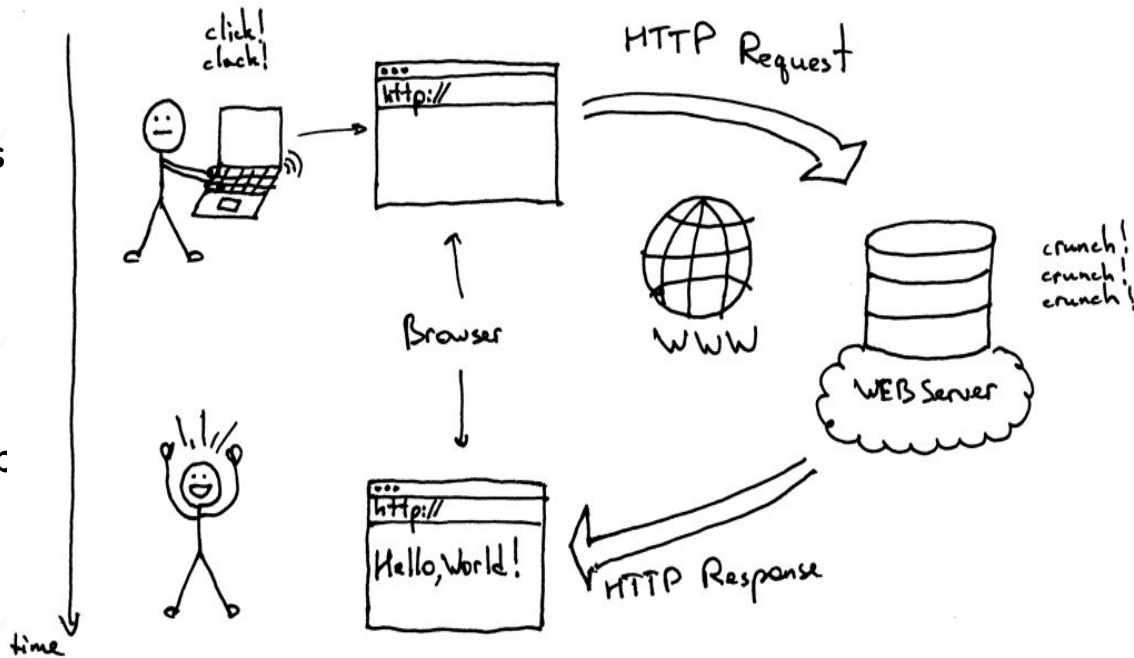
```
$ node index.js
Started at 3000
GET / 200 11.721 ms - 19
GET /favicon.ico 404 1.627 ms - 150
```

Hay paquetes como **morgan** (logger) que nos permiten ver esto mismo en consola

Repaso ARQ. CLIENTE - SERVIDOR

Arquitectura cliente-servidor

Para entender los métodos HTTP, es importante cubrir el concepto de arquitectura cliente-servidor. Esta arquitectura describe cómo funcionan todas las aplicaciones web y define las reglas de su comunicación.



Repaso ARQ. CLIENTE - SERVIDOR

Arquitectura cliente-servidor

Esta arquitectura ayuda a proteger cosas como las claves de la API, los datos personales, etc. Ahora, herramientas modernas como Next.js y Netlify permiten a los desarrolladores ejecutar código de servidor en la misma aplicación que su aplicación cliente, sin necesidad de una aplicación de servidor dedicada.



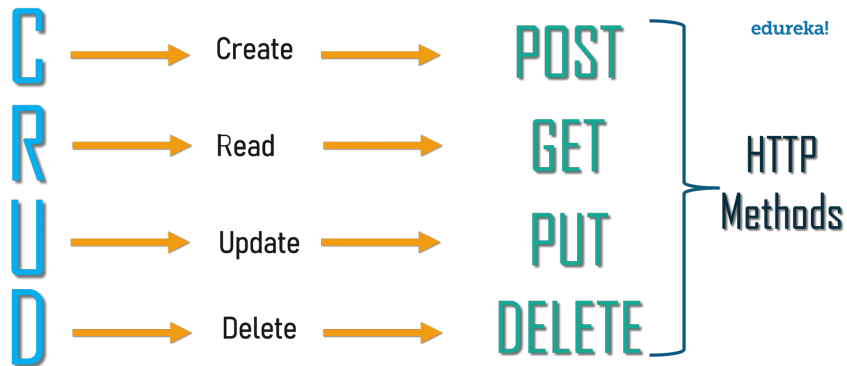
How to Deploy Next.js Sites to Netlify

Verbos HTTP

DEV.F
DESARROLLAMOS(PERSONAS);

dev

REST - http verbs



VERBOS HTTP

La primera línea de un mensaje de petición empieza con un verbo (también se le conoce como método). **Los verbos definen la acción que se quiere realizar sobre el recurso.**

Los verbos más comunes son:

- **GET:** Solicitar un recurso o varios.
- **POST:** Publicar un recurso.
- **PUT:** Reemplazar un recurso.
- **DELETE:** Eliminar un recurso.
- **PATCH:** Actualizar un recurso

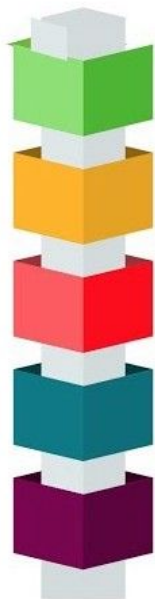
Nota: Cuando ingresas a una página desde un navegador, por debajo el navegador envía un mensaje GET, lo mismo cuando oprimes un vínculo a otra página.

Códigos de Respuesta (Status Code)

DEV.F
DESARROLLAMOS(PERSONAS);

dev

HTTP Status Codes



1XX
INFORMATIONAL

2XX
SUCCESS

3XX
REDIRECTION

4XX
CLIENT ERROR

5XX
SERVER ERROR

¿Qué Código de estado debe devolver el servidor?

200 La solicitud tiene éxito ya que el endpoint existe y realiza alguna validación interna, pero la respuesta debe incluir alguna información sobre el motivo por el que se deniega el acceso.

401 El acceso no está autorizado. No se requiere información adicional en la respuesta.

403 El acceso está prohibido y la respuesta incluye información sobre el acceso denegado.

503 No se puede hacer nada a menos que el acceso del usuario esté validado y autorizado (servicio no disponible).



Response Codes

La primera línea de un mensaje de respuesta tiene un código de 3 dígitos que le indica al cliente cómo interpretar la respuesta.

Los códigos de respuesta se dividen en cinco categorías dependiendo del dígito con el que inician:

- **1XX:** Información
- **2XX:** Éxito
- **3XX:** Redirección
- **4XX:** Error en el cliente
- **5XX:** Error en el servidor

¿Recuerdas el famoso error 404?

HTTP Cats

<https://http.cat/>

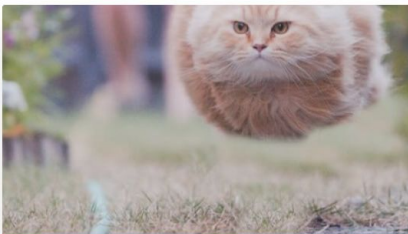


Usage:

```
https://http.cat/[status_code]
```



Note: If you need an extension at the end of the URL just add .jpg.



100

Continue



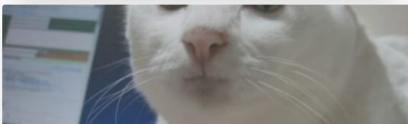
101

Switching Protocols



102

Processing



DEV.F

PRO TIP

Los códigos de respuesta son una convención no una regla.

Cada desarrollador puede asignar el código de respuesta que desee a cada petición, pero se recomienda seguir las convenciones (lo más posible)

API Rest

DEV.F
DESARROLLAMOS(PERSONAS);

dev

¿Qué es Rest?

REST es una interfaz para conectar varios sistemas basados en el protocolo HTTP y nos sirve para obtener, generar datos y operaciones, devolviendo esos datos en formatos muy específicos, como XML y JSON.

- Protocolo cliente/servidor "SIN ESTADO"
- Utiliza verbos http
 - GET (leer)
 - POST (crear)
 - PUT (editar)
 - DELETE (borrar)
- Devuelve
 - JSON
 - XML



HTTP STATUS
200 – OK
201 – CREATED
403 – FORBIDDEN
404 – NOT FOUND



Ejemplos:

```
GET    /rest/api/pedido/{id}
DELETE /rest/api/pedido/{id}
POST   /rest/api/pedido
PUT     /rest/api/pedido/{id}/producto/{idProducto}
```



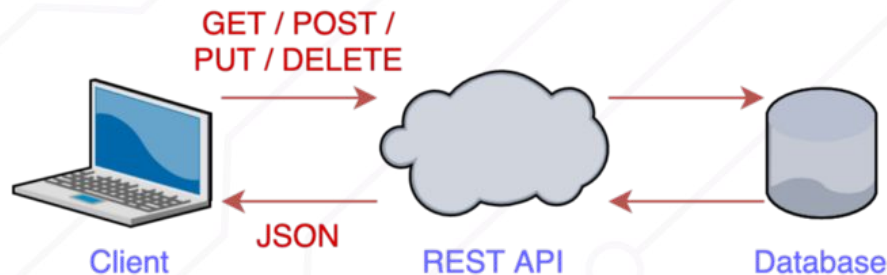
Devuelve
JSON/XML

```
{
  "nombre": "Pepito",
  "coches" : ["Ford", "Fiat", "Audi"]
}
```

API + Rest = API Rest

API: utiliza peticiones HTTP responsables de las operaciones básicas necesarias para la manipulación de datos (GET, POST, etc).

Rest: es un conjunto de restricciones que se utilizan para que las solicitudes HTTP cumplan con las directrices definidas en la arquitectura: cliente servidor, sin estado, con cache, etc.



API Rest es el conjunto de buenas prácticas utilizadas en las peticiones HTTP realizadas por una API en una aplicación web.

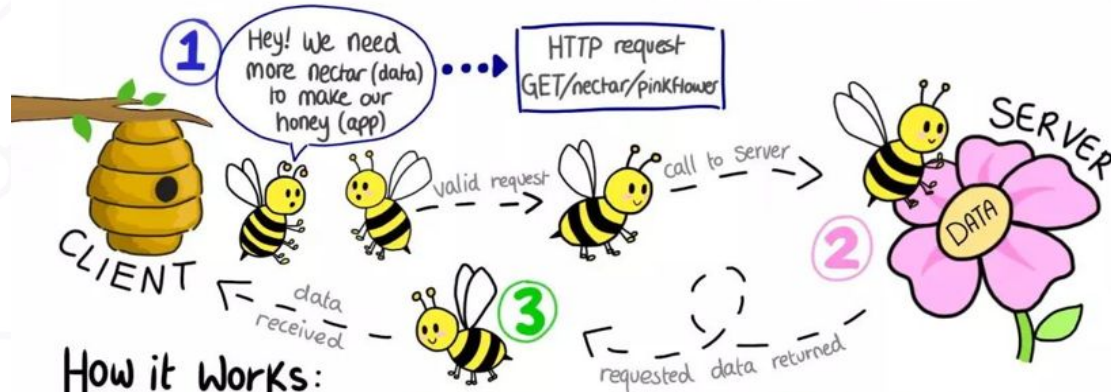
API REST

Es decir, cuando se habla de API Rest, significa utilizar una API para acceder a aplicaciones back-end, de manera que esa comunicación se realice con los estándares definidos por el estilo de arquitectura Rest.

What is an API ?

@Rapid_API 

An application programming interface allows two programs to communicate. On the web, APIs sit between an application and a web server, and facilitate the transfer of data.



How it Works:

1 Request

API call is initiated by the Client application via a HTTP request

2 Receive

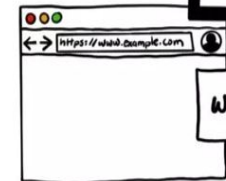
Our worker bee acts as an API, going to a Flower (server) to collect nectar (data)

3 Response

The API transfers the requested data back to the requesting application, usually in JSON format

WHAT HAPPENS* WHEN YOU TYPE IN A URL IN AN ADDRESS BAR IN A BROWSER?

*a brief overview



www.example.com

DNS

initiate TCP connection

HTTP Request

GET https://example.com HTTP/1.1

Server Response

1xx informational message
2xx success
3xx redirects
4xx client errors
5xx server errors

Cache hit Failed

Cache
Browser
Operating System
Router
ISP

What's the IP of "www.example.com"?

Root domain name Server

Top Level domain name Server

2nd Level domain name Server

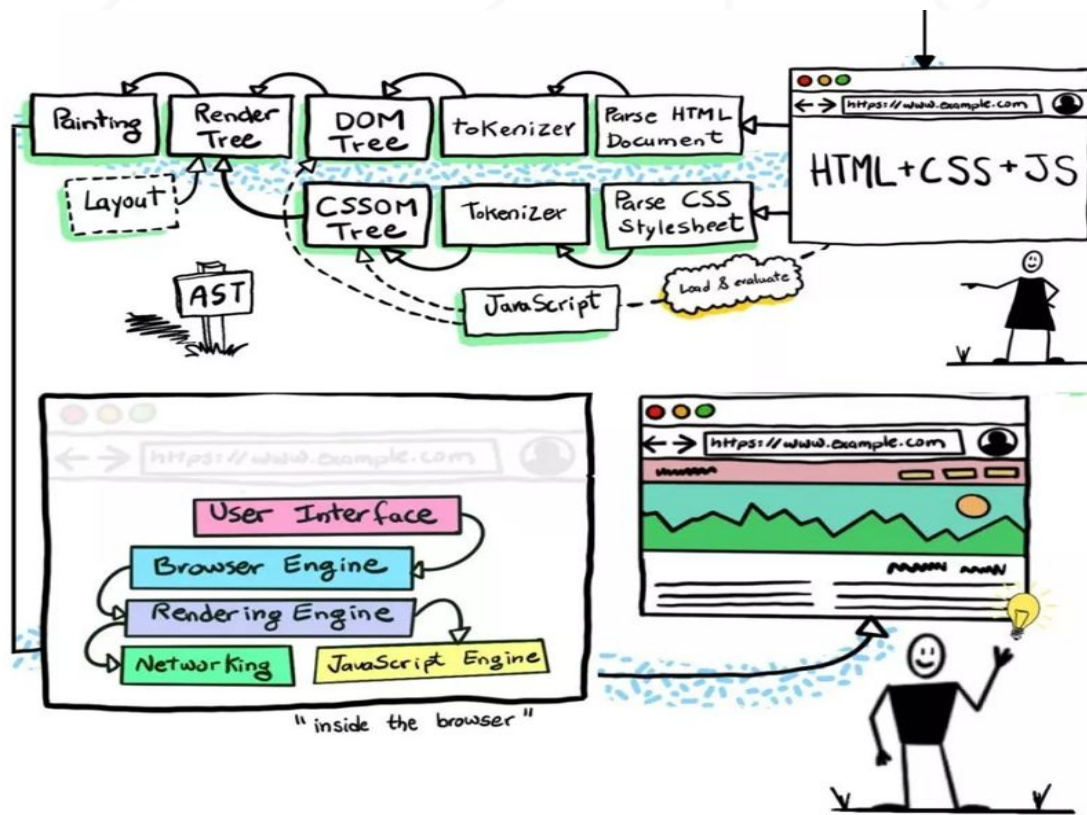
3rd Level domain name Server

93.184.216.34

TCP/IP 3 handshake

Client → SYN → Server
Server → SYN ACK → Client
Client → ACK → Server





Nombrando Endpoints en API Rest

/users // lista todos los usuarios

/users/123 // lista a un usuario en específico

/users/123/orders // lista los pedidos de un usuario específico

/users/123/orders/0001 // lista una orden específica de un usuario específico

Endpoints y Verbos Http

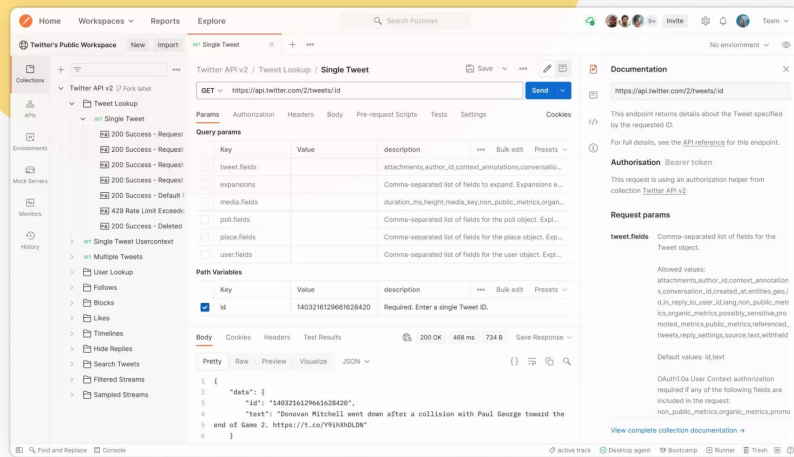
Un mismo Endpoint, hara diferentes acciones dependiendo el verbo http usado para accederlo

Recuros / Estado	POST	GET	PUT	DELETE
/casas	Crea una casa	Devuelve una lista de todos las casas	Modifica casas	Borra todas las casas
/casas/123	error	Devuelve los detalles de la casa 123	Modifica la casa	Borra la casa

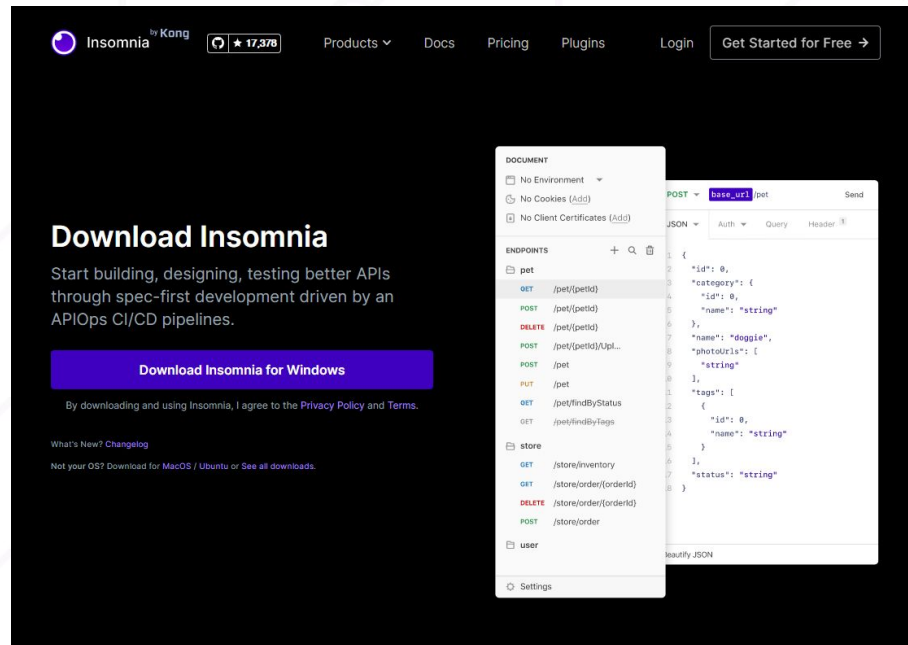
Herramientas para consumir una APIs

DEV.F
DESARROLLAMOS(PERSONAS);

dev



<https://www.postman.com/>




<https://insomnia.rest/>


DEV.F

APIs para probar

DEV.F
DESARROLLAMOS(PERSONAS);

dev

[Home](#)[About](#)[API v2](#)[GraphQL v1beta](#)



The RESTful Pokémon API

Serving over 250,000,000 API calls each month!

All the Pokémon data you'll ever need in one place,
easily accessible through a modern RESTful API.

[Check out the docs!](#)

Try it now!

Need a hint? Try `pokemon/ditto`, `pokemon/1`, `type/3`, `ability/4`, or `pokemon?limit=100&offset=200`.

Direct link to results: <https://pokeapi.co/api/v2/pokemon/ditto>

Resource for ditto

```
▼ abilities: [] 2 items
▼ 0: {} 3 keys
  ▼ ability: {} 2 keys
    name: "limber"
    url: "https://pokeapi.co/api/v2/ability/7/"
    is_hidden: false
    slot: 1
▼ 1: {} 3 keys
  ▼ ability: {} 2 keys
    name: "imposter"
    url: "https://pokeapi.co/api/v2/ability/158/"
    is_hidden: true
```

<https://pokeapi.co/>

SWAPI

The Star Wars API

(what happened to swapi.co?)

All the Star Wars data you've ever wanted:
Planets, Spaceships, Vehicles, People, Films and Species
From all **SEVEN** Star Wars films
Now with The Force Awakens data!

Try it now!

Need a hint? try `people/1`/or `planets/3`/or `starships/9`

Result:

```
{
  "name": "Luke Skywalker",
  "height": "172",
  "mass": "77",
  "hair_color": "blond",
  "skin_color": "fair",
  "eye_color": "blue",
  "birth_year": "19BBY",
  "gender": "male",
  "homeworld": "https://swapi.dev/api/planets/1/",
  "films": [
    "https://swapi.dev/api/films/2/",
    "https://swapi.dev/api/films/6/",
    "https://swapi.dev/api/films/3/",
    "https://swapi.dev/api/films/1/",
    "https://swapi.dev/api/films/7/"
  ],
  "species": [
```

<https://swapi.dev/>

DEV.F

GUÍA PARA PRINCIPIANTES

