

# Introdução à Web e API



# Prazer, Analu.

Eu sou desenvolvedora Full-Stack, professora e co-fundadora da @brasamag.

Fui da primeira turma de Back-End da Reprograma

Hoje trabalho como Engenheira de Software no Banco Itaú.

Email: sampaioanaluiza@gmail.com



# Objetivos



Modelo Server/Client



API e API Rest



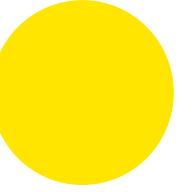
URI, URL, URN, Dominio, IP e DNS



JSON



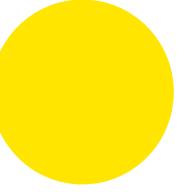
Protocolo HTTP



Consumindo APIs



Request e Response



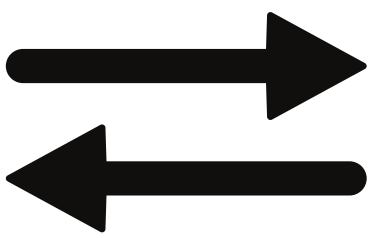
Para o lar

# Como funciona a Internet?

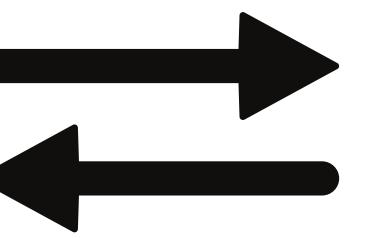
# Servidor/Cliente



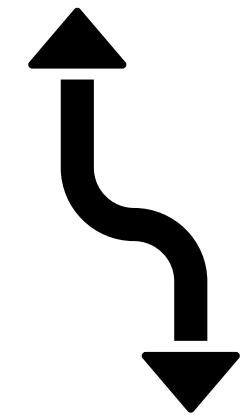
Usuários



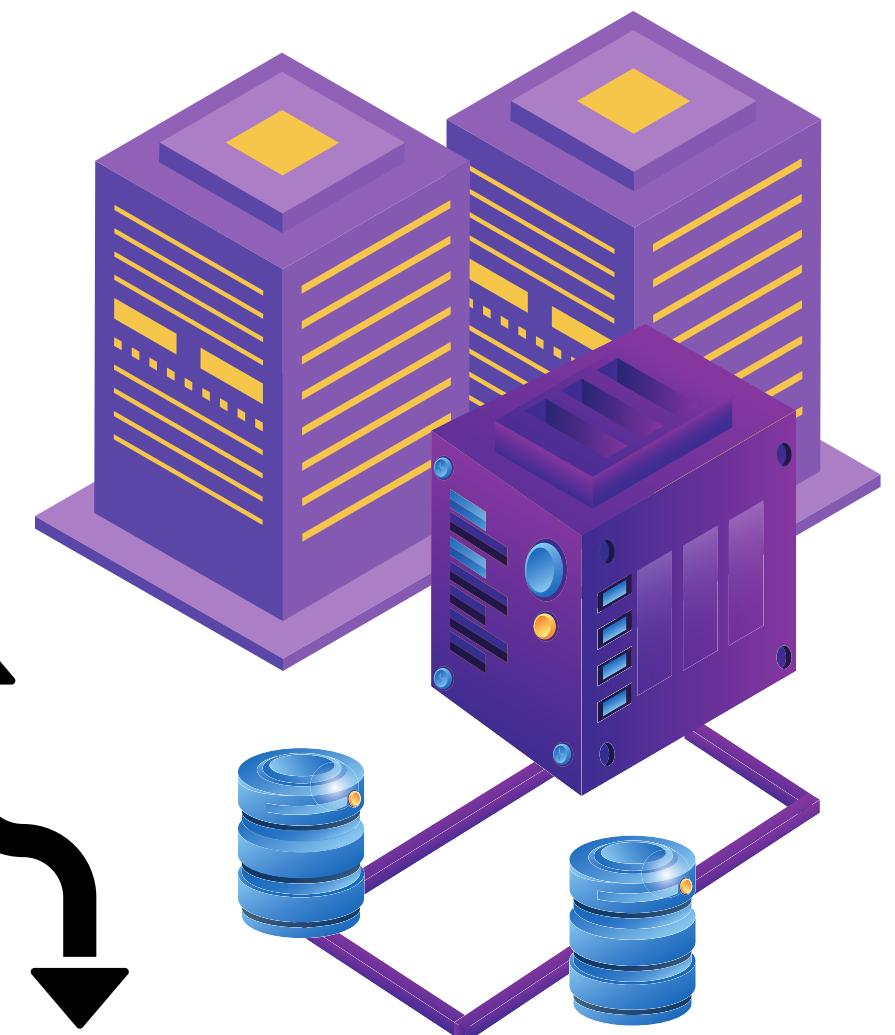
Client



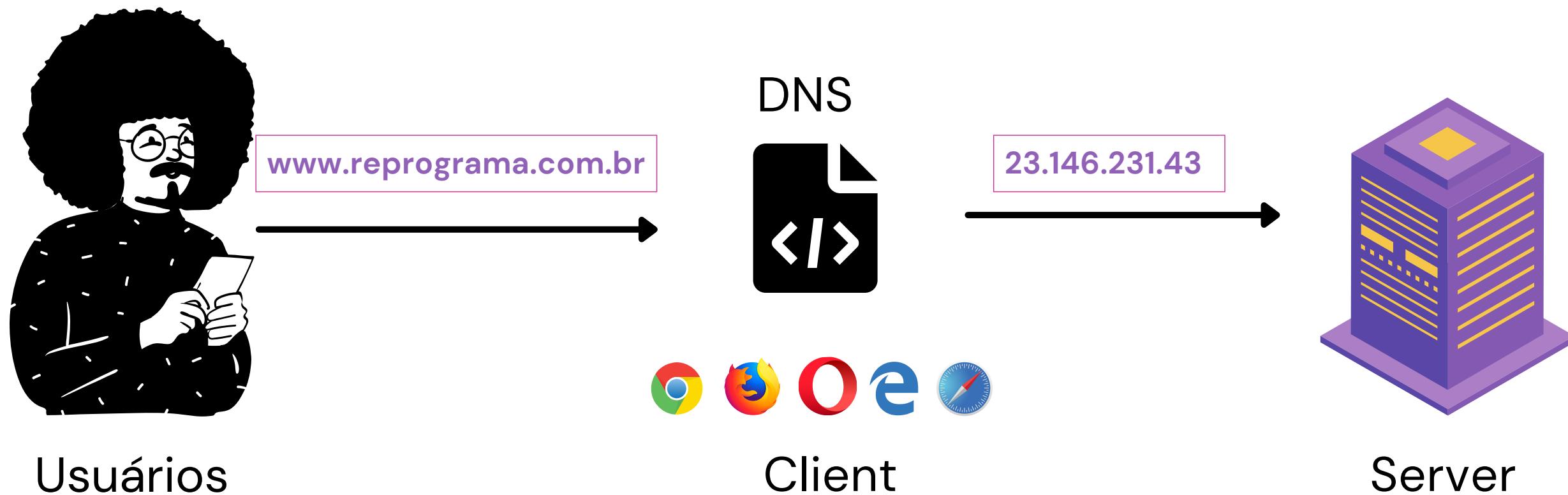
Server



Banco de dados



# Servidor/Cliente

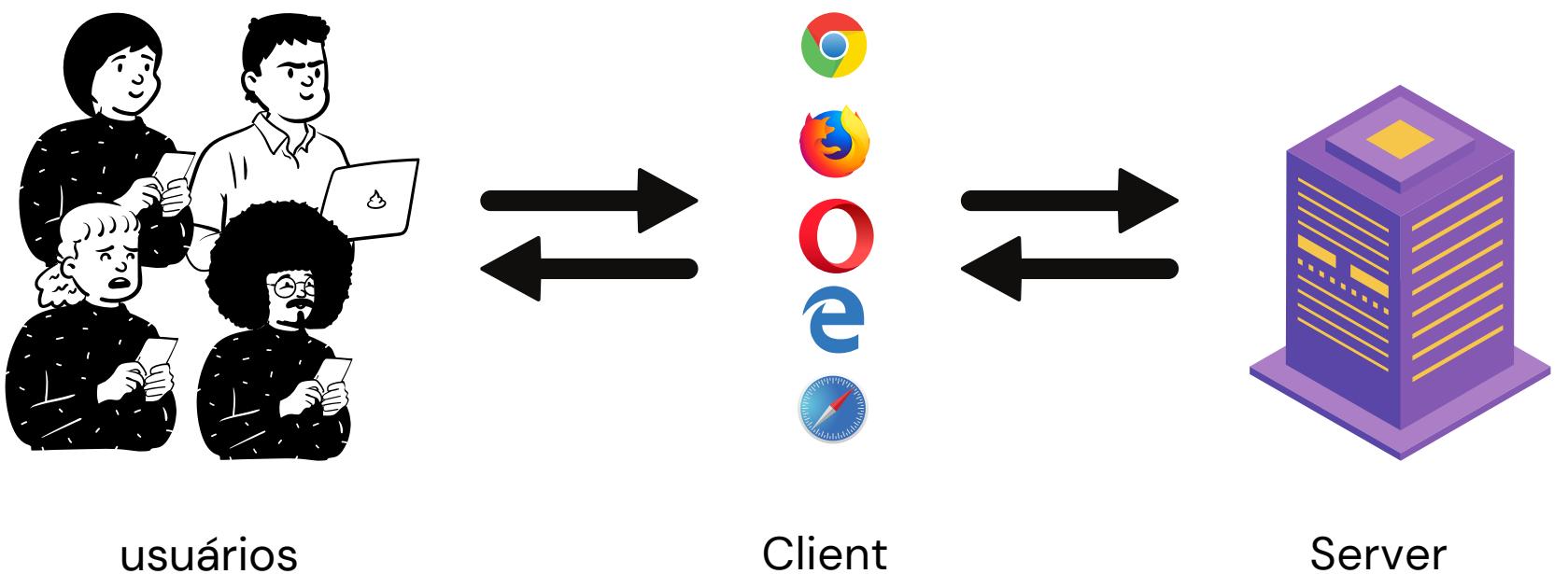


# Client

Client é a interface que os usuários interagem, é essa camada que é responsável de **solicitar** serviços e informações de um ou mais servidores.

Algumas tarefas a serem realizadas pelo Cliente:

- Manipulação de tela
- Interpretação de menus ou comandos
- Entrada e validação dos dados
- Recuperação de erro
- Manipulação de janelas
- Gerenciamento de som e vídeo (em aplicações multimídia)



# Server

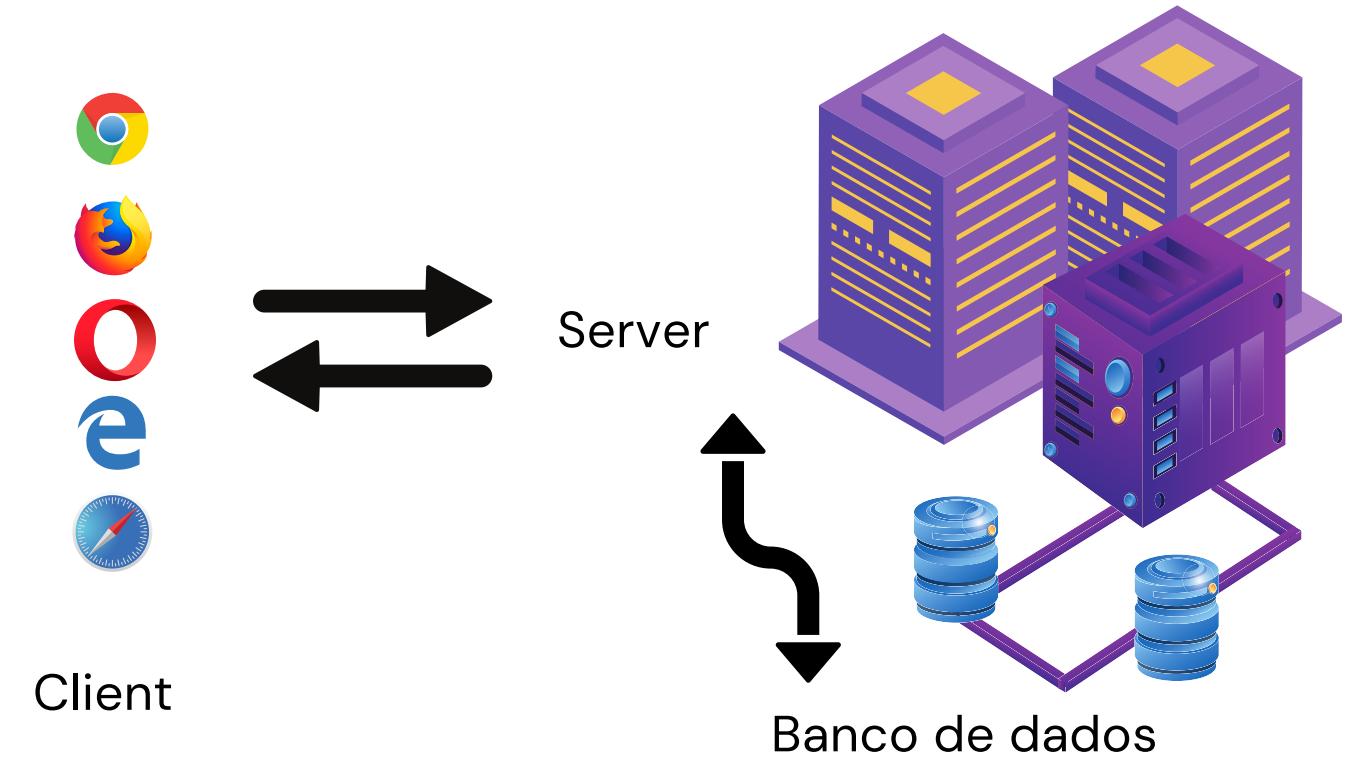
O Servidor é o responsável pelo processo, organização e gerenciamento das informações.

É ele que **responde às solicitações** feitas pelo Client.

Ele é um processo reativo, disparado pela chegada de pedidos de seus clientes

O processamento do servidor geralmente inclui:

- Acessar
- Organizar os dados compartilhados
- Fazer a comunicação com o Banco de Dados
- Atualizar dados previamente armazenados
- Gerenciamento dos recursos compartilhados.



# www.reprograma.com.br

O que acontece quando acessamos um site?



- 1 A URI é processada
- 2 É feita uma requisição
- 3 É dada uma Resposta
- 4 A pagina é renderizadas e aparece na tela

1

## A URI é processada



Todo site tem um **domínio**, normalmente é por ele que acessamos e conhecemos o Site.

Porém, no Server esse site não está registrado pelo nome de domínio, e sim pelo **endereço de IP**

Internet Protocol Address é o endereço exato de onde o site está dentro do servidor.

Então, antes de uma requisição ser feita o domínio deve virar o IP, e pra isso, usamos o **DNS**, o Domain Name System (Sistema de Nome de Domínio) que é como um grande dicionário de domínio para IP que já vem "de fábrica" no browser

1

## A URI é processada

Para entendermos uma URI precisamos entender a URL e a URN.



URL

**www.reprograma.com.br**

«endereço do servidor»

URL – Uniform Resource Locator (localizador de recurso uniforme). Ela representa o local/Host que estão localizados os recursos

1

A URI é processada



URN – Uniform Resource Name (Nome de Recursos Universal). Ela representa um recurso específico na web que está sendo acessado

/equipe.html

<recurso>

/alunas.html

<recurso>

URN

/parcerias.html

<recurso>

1

## A URI é processada



URI – Uniform Resource Identifier (Identificador de Recursos Universais). É o identificador contendo que une o protocolo http + o localizador do recurso (URL) + nome do recurso (URN)

Pode ser uma página html, imagem, video ou qualquer outro arquivo web tem um endereço dentro da internet, esse endereço é a **URI**

**protocolo**

**URL**

**URN**

**https://**

**www.reprograma.com.br**

**/equipe.html**

**<protocolo>**

**<endereço do servidor>**

**<recurso>**

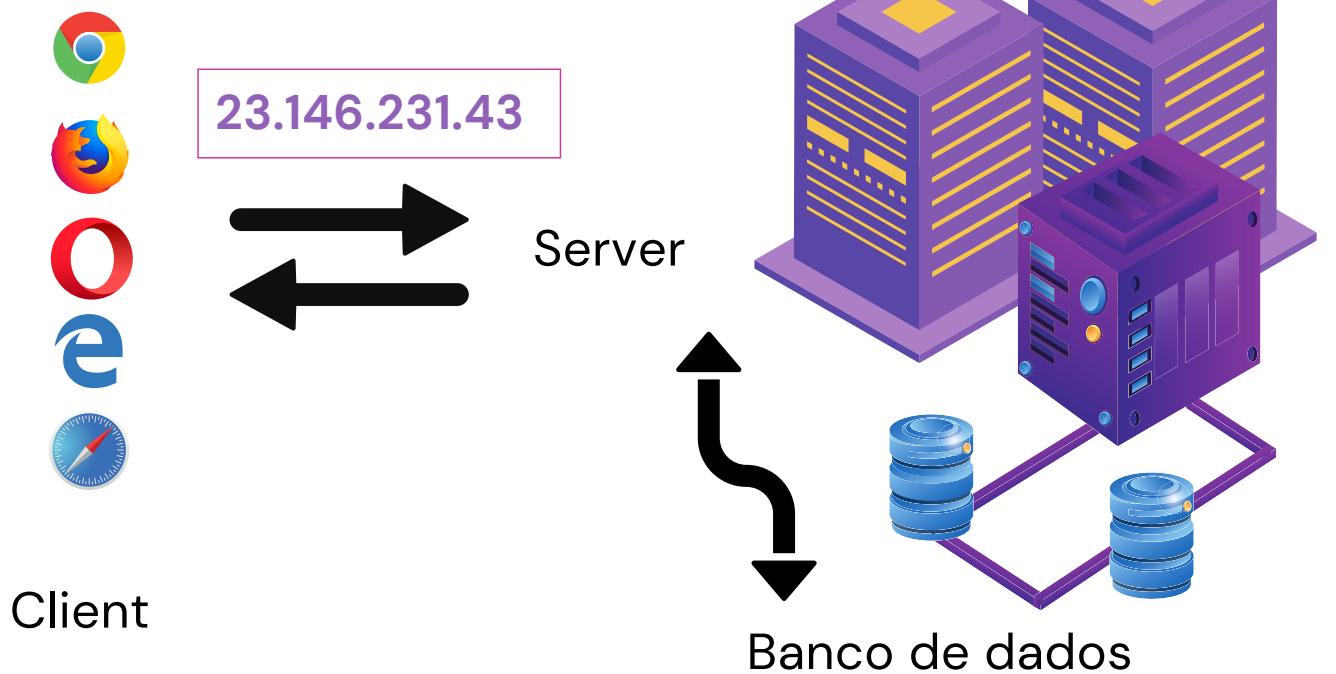
2

## O Request é enviado

Agora com endereço certo, o Client faz uma requisição, ou Request, cheio de informações desejadas.

Pra que isso aconteça, tanto o Server quanto o Client devem "falar a mesma língua".

Na maioria dos casos, essa comunicação entre Server e Client é feita a partir do **Protocolo HTTP**



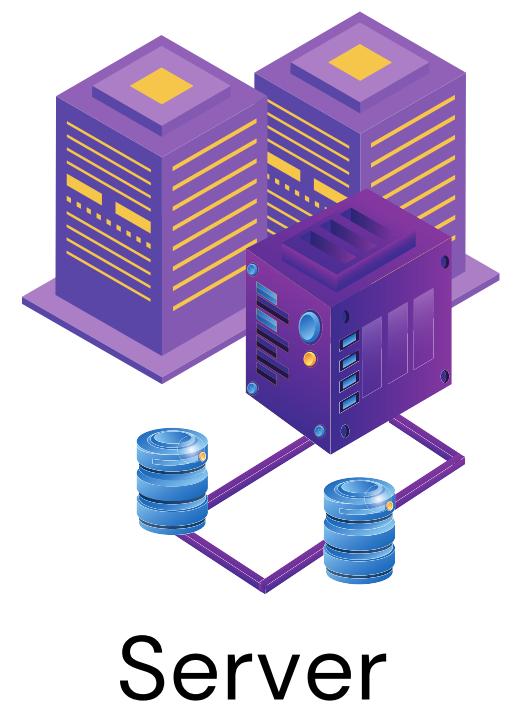
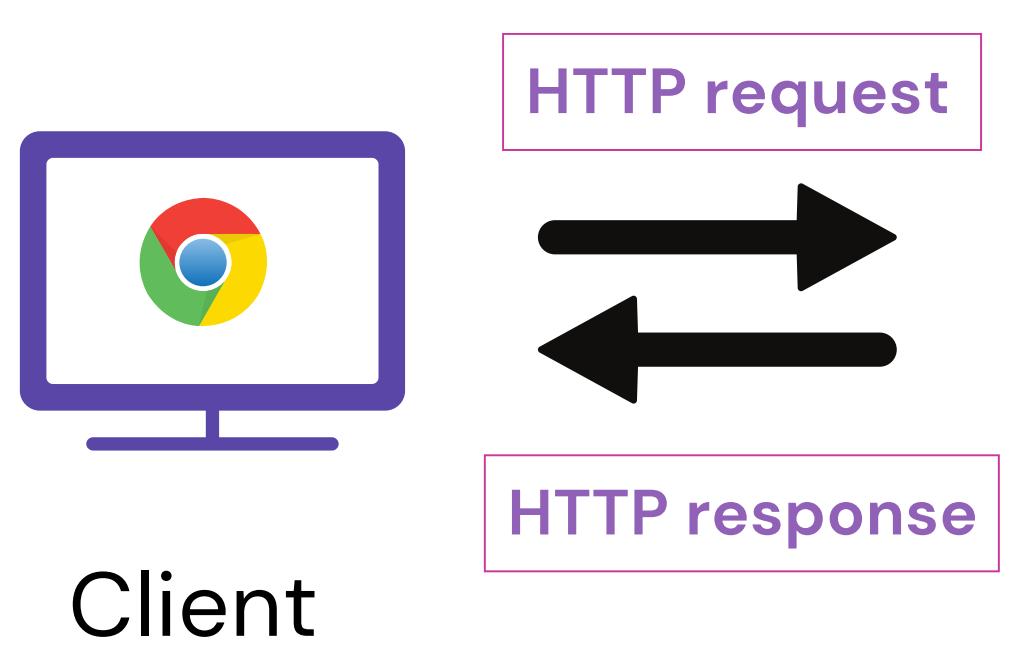
2

## O Request é enviado

Protocolo de Transferência de Hipertexto, o HTTP, é um protocolo usado dentro do modelo Client/Server é baseado em pedidos (requests) e respostas (responses).

O protocolo HTTP define um conjunto de métodos de requisição responsáveis por indicar a ação a ser executada.

Eles são chamados de **Verbos HTTP** ou **Métodos HTTP**.



2

## O Request é enviado

Os verbos HTTP mais utilizados são:

- GET
- POST
- PUT
- PATCH
- DELETE

Cada um deles corresponde a uma ação real no banco de dados.

**GET**

**POST**

**PUT**

**PATCH**

**DELETE**

**ler**

**criar**

**substituir**

**modificar**

**excluir**

3

## O Server responde

Quando o Client faz um Request o Server envia um **Response**.

E na resposta tem, além do resultado do que foi pedido, um código de status numerico padronizado

**código**

**100-199**

**200-299**

**300-399**

**400-499**

**500-599**

**tipo de resposta**

**informação**

**sucesso**

**redirecionamento**

**erro do cliente**

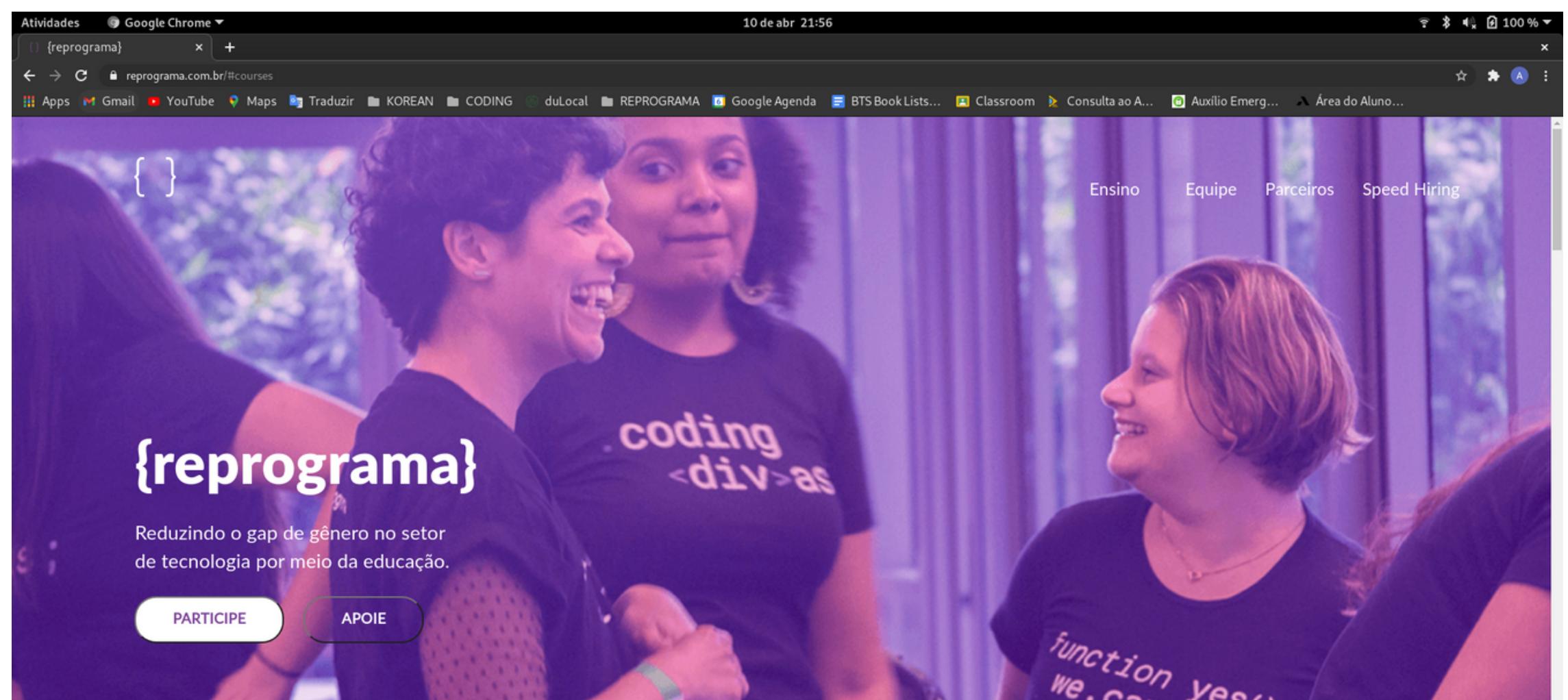
**erro de servidor**

4

O site aparece na tela



Finalmente!



# Ai, que canseira



# Dev Front

São as pessoas que são responsáveis muito mais do que construir as telas bonitas e funcionais.

Elas tem que criar aplicações preparadas para enviar Requests corretamente e receber as Responses, também disponibilizar elas para os usuários



# Dev Back

Dentro desse fluxo, são as pessoas que constroem toda a dinâmica do recebimento de Requests, o envio das Responses corretas, o tratamento das Responses, as execuções de ação no Banco de Dados e a disponibilização para a Dev Front.

Quando trabalhando construindo APIs somos as construtoras da ponte entre o front-end e o banco de dados



# Nós, as Devs Back-end



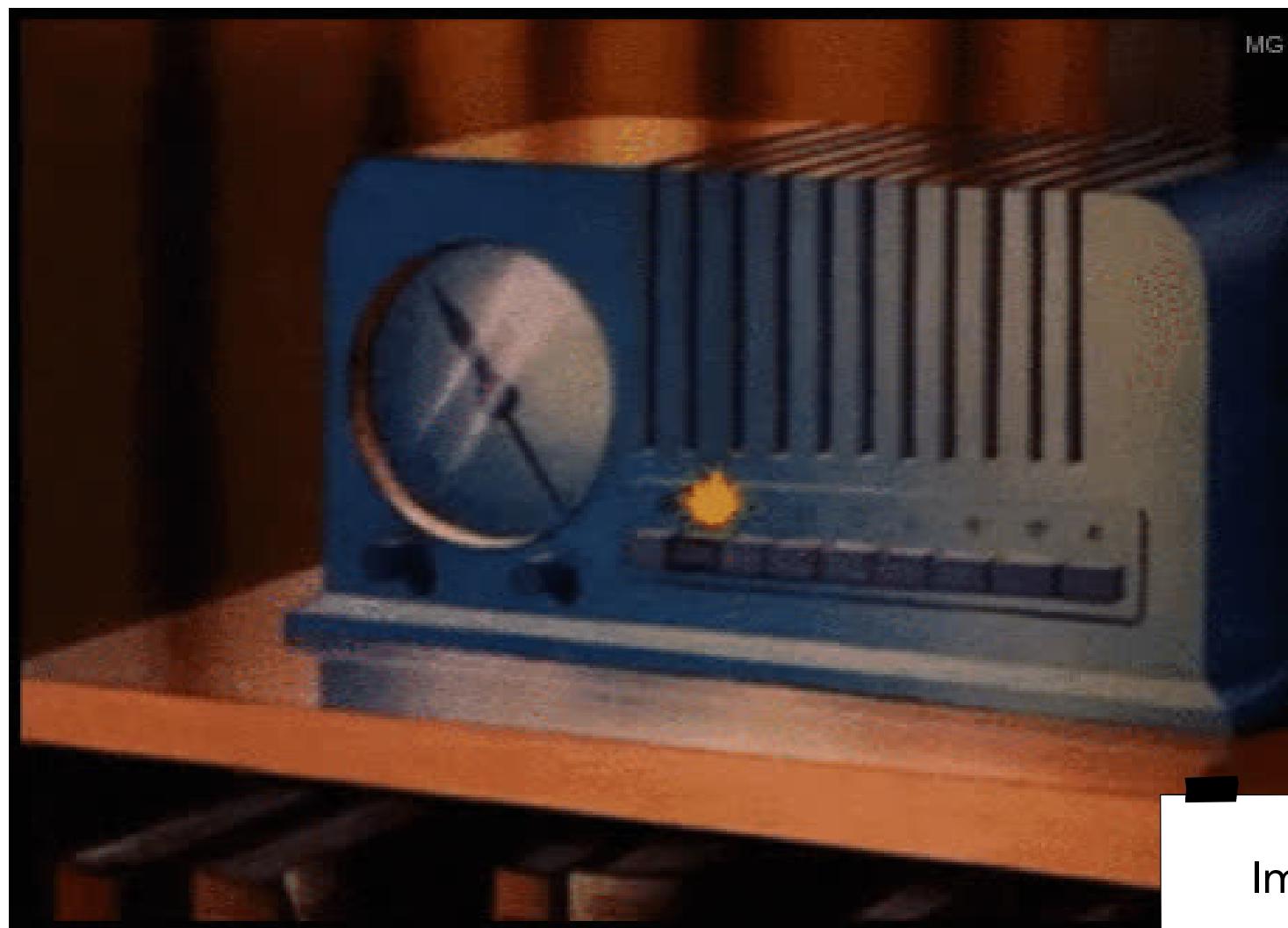
# API

tá, mas o que realmente é isso?

Interface de  
Programação de  
Aplicativos

# Interface

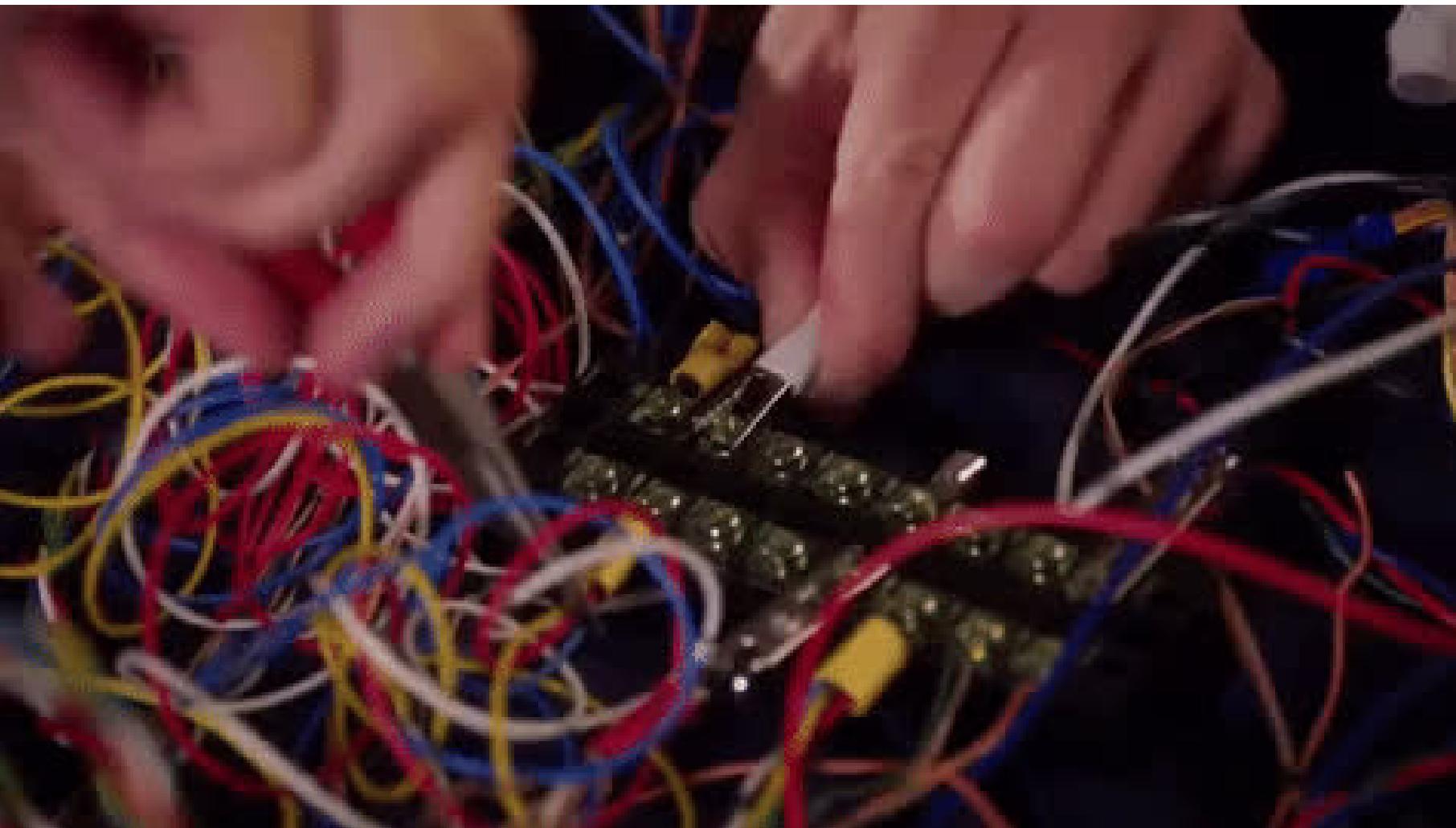
o I da API



Imagine um  
rádio

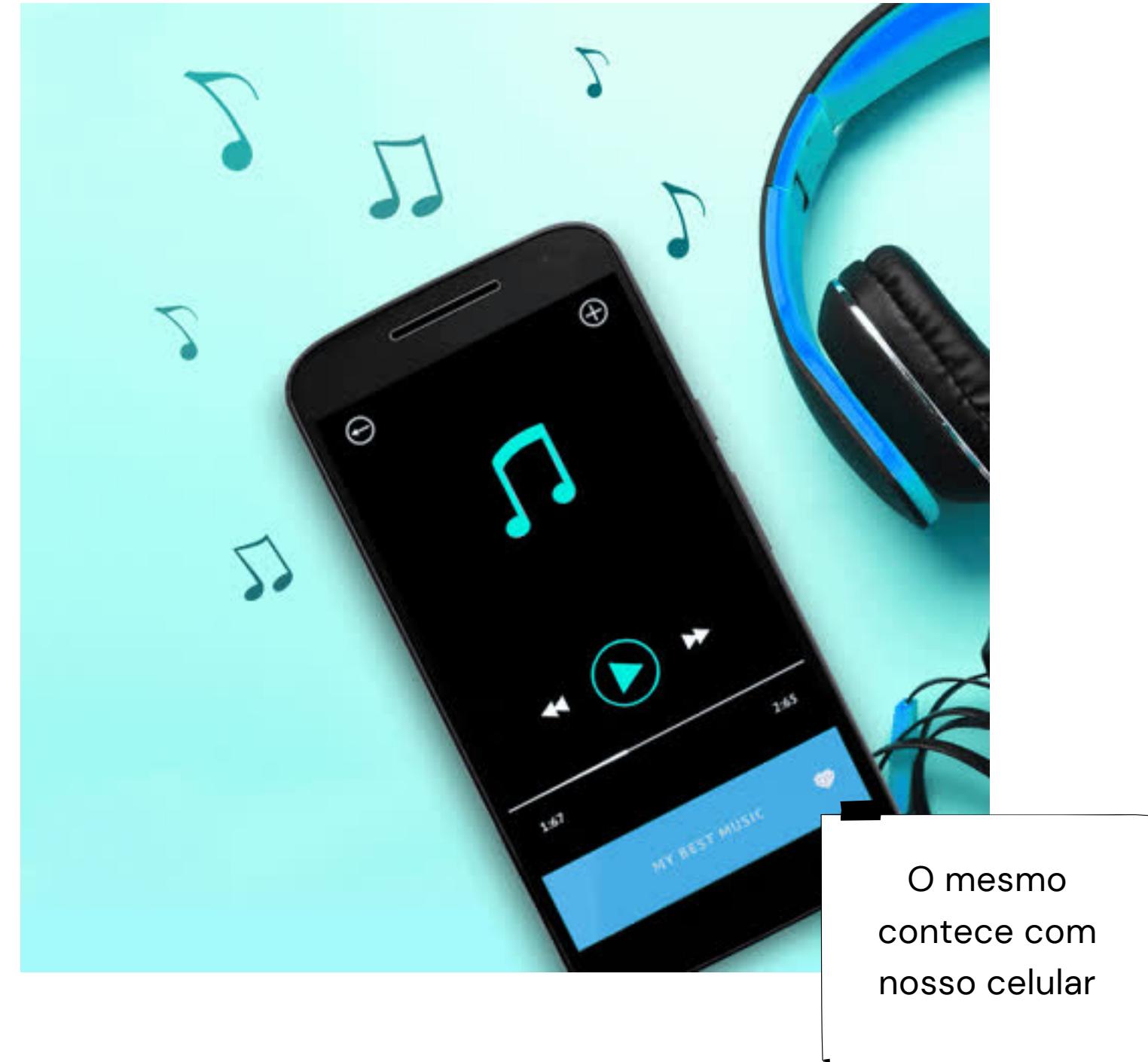
# Interface

o I da API



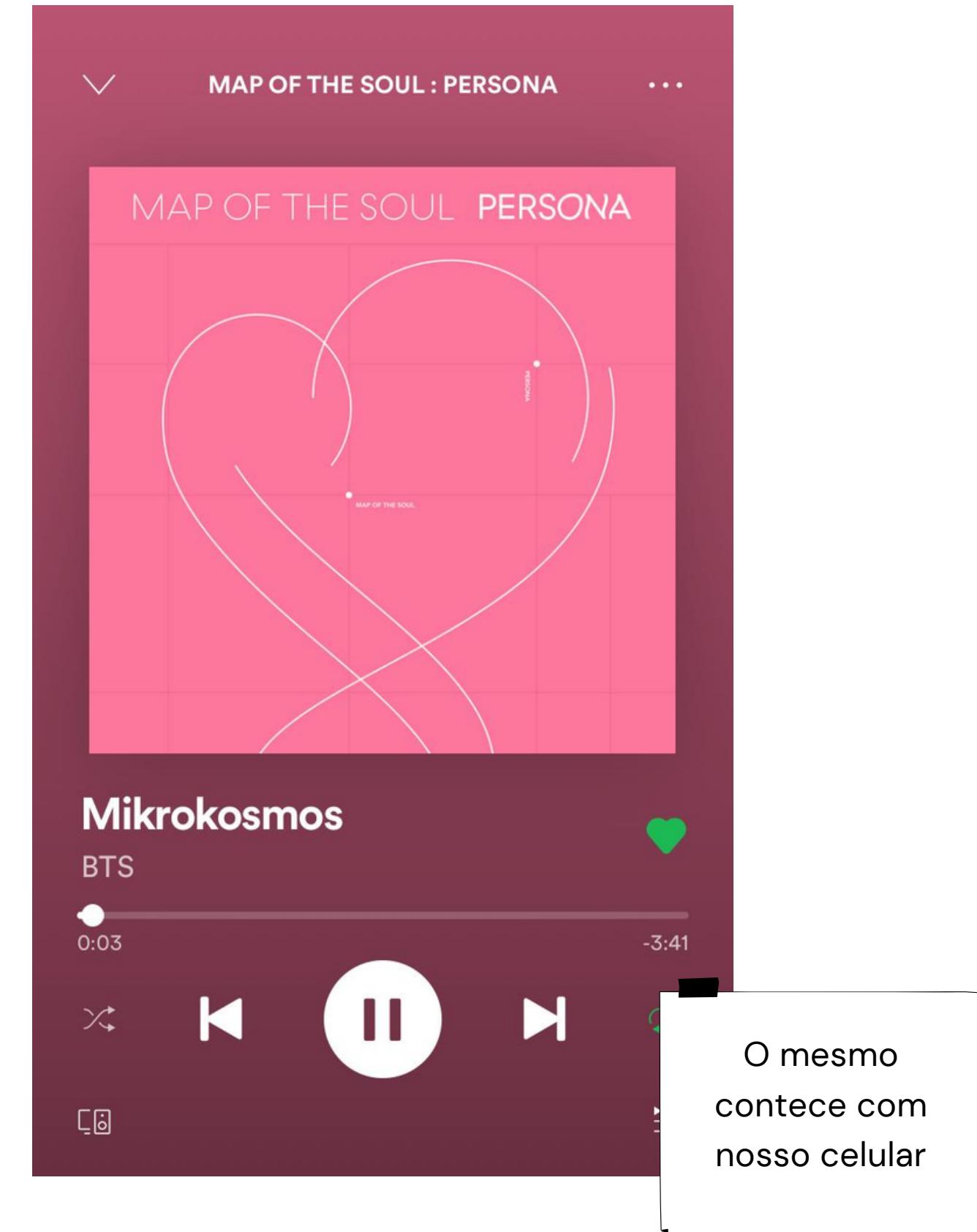
# Interface

o I da API



# Interface

o I da API



# Interface de Programação de Aplicativos

Assim como a interface do rádio, a API busca criar formas e ferramentas de se usar uma funcionalidade ou uma informação sem realmente ter que "reinventar a tal função."

Ela não necessariamente está num link na Web, ela pode ser uma lib ou um framework, uma função já pronta em uma linguagem específica, etc.

# Web APIs

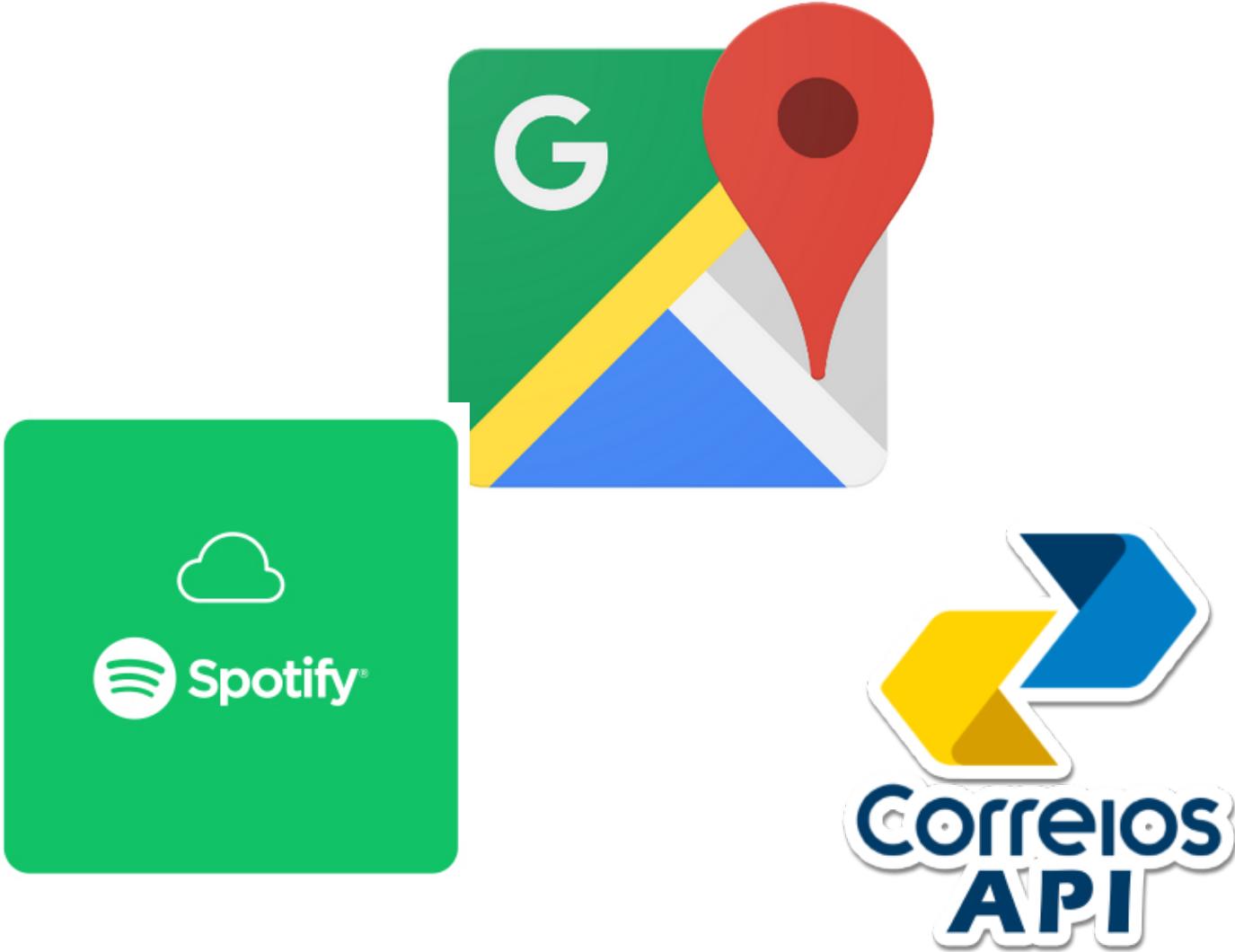
São um conjunto de instruções e padrões de programação para acesso a um aplicativo de software. Uma empresa de software lança sua API para o público de modo que outros criadores de software possam desenvolver produtos acionados por esse serviço.



# APIs Publicas

São aquelas que são disponibilizadas gratuitamente para desenvolvedoras e usuarios com restrição minima. Podem precisar de cadastro, o uso de API Key ou ser completamente abertas.

Elas estão relacionadas com uso externo de dados ou serviços.



# APIs Privadas

São oposto das APIs públicas. Elas estão ligadas à serviços sigilosos, dados sensíveis, transações de empresas privadas, comunicação e ferramentas interna da empresa, etc



# APIs REST e RESTfull

Trata-se de um conjunto de princípios e definições necessários para a criação de um projeto com interfaces bem definidas, Rest, que é a abreviatura de **Representational State Transfer**,

é um conjunto de restrições utilizadas para que as requisições HTTP atendam as diretrizes definidas na arquitetura.

# Objetivos



Modelo Server/Client



API e API Rest



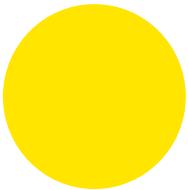
URI, URL, URN, Dominio, IP e DNS



JSON



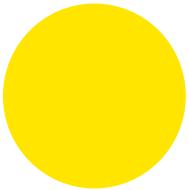
Protocolo HTTP



Consumindo APIs



Request e Response

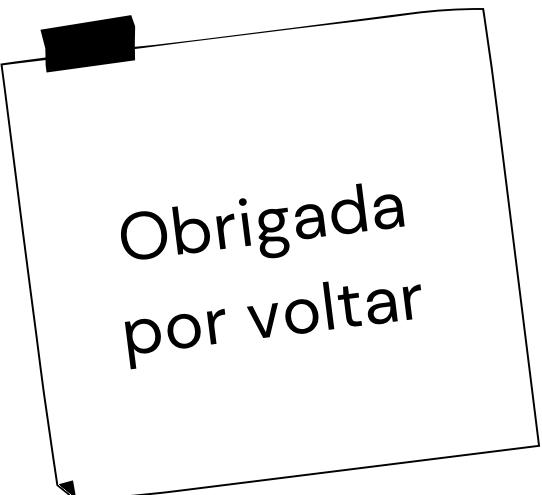


Para o lar

# Ai, que fome...



Nossa, você aqui  
de novo?



# JSON

Notação de Objetos JavaScript.

JSON é uma formatação leve de troca de dados.

É em formato de texto e completamente independente de linguagem, pois usa convenções que são familiares às linguagens C e familiares, incluindo C++, C#, Java, JavaScript, Perl, Python e muitas outras.

Estas propriedades fazem com que JSON seja um formato ideal de troca de dados.

# JSON

JSON é baseado em duas estruturas:

- Uma coleção de pares de nome / valor. Em várias linguagens, isso é realizado como um objeto, registro, estrutura, dicionário, tabela de hash, lista de chaves ou matriz associativa.
  - Uma lista ordenada de valores. Na maioria das linguagens, isso é realizado como um array, vetor, lista ou sequência.
- .

# JSON

Você pode incluir os mesmos tipos de dados básicos como em um objeto JavaScript padrão.

Porem, diferente das Arrays e Objetos os nomes das propriedades devem ser strings com aspas duplas e as vírgulas à direita são proibidas.

```
const data = [
  {
    "name": "Bulbasaur",
    "description": "Bulbasaur can be seen napping in bright sunlight. There is a seed on its back. By soaking up the sun's rays, the seed grows progressively larger. Bulbasaur can be seen napping in bright sunlight. There is a seed on its back. By soaking up the sun's rays, the seed grows progressively larger.",
    "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/001.png",
    "types": ["poison", "grass"]
  },
  {
    "name": "Charmander",
    "description": "The flame that burns at the tip of its tail is an indication of its emotions. The flame wavers when Charmander is enjoying itself. If the Pok  mon becomes enraged, the flame burns fiercely. The flame that burns at the tip of its tail is an indication of its emotions. The flame wavers when Charmander is enjoying itself. If the Pok  mon becomes enraged, the flame burns fiercely.",
    "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/004.png",
    "types": ["fire"]
  },
  {
    "name": "Charmeleon",
    "description": "Charmeleon mercilessly destroys its foes using its sharp claws. If it encounters a strong foe, it turns aggressive. In this excited state, the flame at the tip of its tail flares with a bluish white color. Charmeleon mercilessly destroys its foes using its sharp claws. If it encounters a strong foe, it turns aggressive. In this excited state, the flame at the tip of its tail flares with a bluish white color.",
    "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/005.png",
    "types": ["fire"]
  }
]
```

# Manipulando ele



# 1-exe-ghibli

Apresente os dados do JSON

```
exercicios-JSON > 01-exe-ghibli > JS script.js > ...
1 //Apresente no console cada um dos atributos desse JSON
2
3 const obj = [
4   {
5     "title": "Castle in the Sky",
6     "description": "The orphan Sheeta inherited a mysterious crystal that I
7   },
8   {
9     "title": "Grave of the Fireflies",
10    "description": "In the latter part of World War II, a boy and his sist
11  },
12  {
13    "title": "My Neighbor Totoro",
14    "description": "Two sisters move to the country with their father in or
15  },
16  {
17    "title": "Kiki's Delivery Service",
18    "description": "A young witch, on her mandatory year of independent lif
19  },
20  {
21    "title": "Only Yesterday",
22    "description": "It's 1982, and Taeko is 27 years old, unmarried, and ha
23  }
24 ]
25
26 // COMEÇA O EXERCÍCIO
27
```

# 2-exe-pokemon

Apresente os dados do JSON

```
exercicios-JSON > 02-exe-pokemon > JS script.js > ...
1  const data = [
2    {
3      "name": "Bulbasaur",
4      "description": "Bulbasaur can be seen napping in bright sunlight. There is a seed on its back. By soaking",
5      "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/001.png",
6      "types": ["poison", "grass"]
7    },
8    {
9      "name": "Ivysaur",
10     "description": "There is a bud on this Pok  mon's back. To support its weight, Ivysaur's legs and trunk gr",
11     "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/002.png",
12     "types": ["poison", "grass"]
13   },
14   {
15     "name": "Venusaur",
16     "description": "There is a large flower on Venusaur's back. The flower is said to take on vivid colors if",
17     "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/003.png",
18     "types": ["poison", "grass"]
19   },
20   {
21     "name": "Charmander",
22     "description": "The flame that burns at the tip of its tail is an indication of its emotions. The flame w",
23     "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/004.png",
24     "types": ["fire"]
25   },
26   {
27     "name": "Charmeleon",
28     "description": "Charmeleon mercilessly destroys its foes using its sharp claws. If it encounters a strong",
29     "art_url": "http://assets22.pokemon.com/assets/cms2/img/pokedex/full/005.png",
30     "types": ["fire"]
31   }
32 // COMECA O EXERCICIO
```

# 1-exe-ghibli



# 2-exe-pokemon

```
● ● ●

for(let i=0; i < data.length; i++){
    let pokemon = data[i];
    console.log(pokemon.name);
    console.log(pokemon.description);
    console.log(pokemon.art_url);

    let tipos = pokemon.types
    for(let i=0; i< tipos.length; i++ ){
        console.log(tipos[i])
    }
}
```

# Objetivos



Modelo Server/Client



API e API Rest



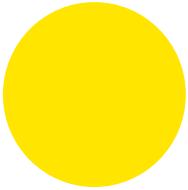
URI, URL, URN, Dominio, IP e DNS



JSON



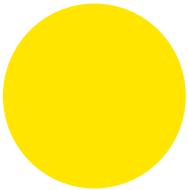
Protocolo HTTP



Consumindo APIs



Request e Response

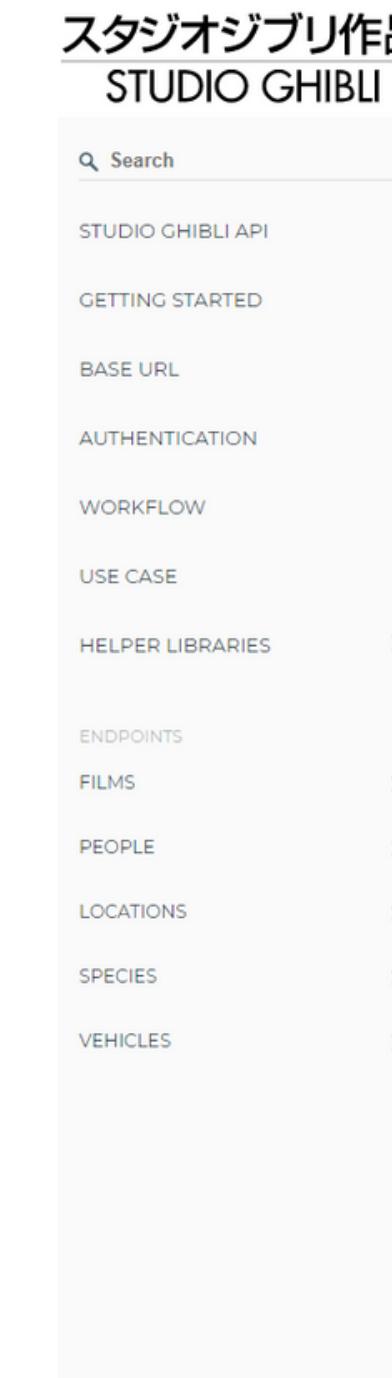


Para o lar

# Consumindo API

## Documentação

<https://ghibliapi.herokuapp.com/#>



### Studio Ghibli API (v1.0.1)

Download OpenAPI specification: [Download](#)

### Studio Ghibli API

The Studio Ghibli API catalogs the people, places, and things found in the worlds of Ghibli. It was created to help users discover resources, consume them via HTTP requests, and interact with them in whatever way makes sense. Navigation can be found on the left sidebar, and the right sidebar shows examples of returned objects for successful calls.

Users can raise an issue, ask for help, or find a contribution guide at the main repo: <https://github.com/janaipakos/ghibliapi>

### Getting Started

Requests can be made with `curl` or other helper libraries by following regular REST calls. For example, here is how to GET the resource for the film *My Neighbor Totoro*:

```
curl https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfclb1b49
```

Calling this resource will respond with the following object:

```
{
  "id": "58611129-2dbc-4a81-a72f-77ddfclb1b49",
  "title": "My Neighbor Totoro",
  "description": "Two sisters move to the country with their father in order to be closer to their hospitali",
  "director": "Hayao Miyazaki",
  "producer": "Hayao Miyazaki",
  "release_date": "1988",
  "rt_score": "93",
  ...
}
```

# Consumindo API

curl

Docs Overview Project Protocols Releases Tool Who and Why

curl / Docs / Tool Documentation / Man Page

## curl.1 the man page

---

<b>NAME</b>	<b>Related:</b>
curl - transfer a URL	<a href="#">File a bug about this man page</a> <a href="#">Manual</a> <a href="#">FAQ</a> <a href="#">HTTP Scripting</a>
<b>SYNOPSIS</b>	
curl [options / URLs]	
<b>DESCRIPTION</b>	
<p>curl is a tool to transfer data from or to a server, using one of the supported protocols (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, MQTT, POP3, POP3S, RTMP, RTMPS, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET and TFTP). The command is designed to work without user interaction.</p> <p>curl offers a busload of useful tricks like proxy support, user authentication, FTP upload, HTTP post, SSL connections, cookies, file transfer resume, Metalink, and more. As you will see below, the number of features will make your head spin!</p> <p>curl is powered by libcurl for all transfer-related features. See <i>libcurl(3)</i> for details.</p>	
<b>URL</b>	
<p>The URL syntax is protocol-dependent. You'll find a detailed description in <a href="#">RFC 3986</a>.</p> <p>You can specify multiple URLs or parts of URLs by writing part sets within braces and quoting the URL as in:</p> <p>"<a href="http://site.{one,two,three}.com">http://site.{one,two,three}.com</a>"</p>	

# Consumindo API

## Consumindo com curl



Requests podem ser feitas com `curl` ou outras libs que seguem o protocolo REST.  
Por exemplo, aqui tem como fazer um GET para o filme Me Visinho Totoro:

```
curl https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49
```

```
$ curl GET https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49
% Total    % Received % Xferd  Average Speed   Time   Time   Current
          Dload  Upload   Total Spent  Left  Speed
0       0     0      0      0      0      0 --:--:-- 0:00:01 --:--:-- 0curl: (6) Could not
resolve host: GET
% Total    % Received % Xferd  Average Speed   Time   Time   Current
          Dload  Upload   Total Spent  Left  Speed
100  1868  100  1868      0      0  2587      0 --:--:-- --:--:-- 2587{
  "id": "58611129-2dbc-4a81-a72f-77ddfc1b1b49",
  "title": "My Neighbor Totoro",
  "description": "Two sisters move to the country with their father in order to be closer to their hospitalized mother, and discover the surrounding trees are inhabited by Totoros, magical spirits of the forest. When the youngest runs away from home, the older sister seeks help from the spirits to find her.",
  "director": "Hayao Miyazaki",
  "producer": "Hayao Miyazaki",
  "release_date": "1988",
  "rt_score": "93",
  "people": [
    "https://ghibliapi.herokuapp.com/people/986faac6-67e3-4fb8-a9ee-bad077c2e7fe",
    "https://ghibliapi.herokuapp.com/people/d5df3c04-f355-4038-833c-83bd3502b6b9",
    "https://ghibliapi.herokuapp.com/people/3031caa8-eb1a-41c6-ab93-dd091b541e11",
    "https://ghibliapi.herokuapp.com/people/87b68b97-3774-495b-bf80-495a5f3e672d",
    "https://ghibliapi.herokuapp.com/people/d39deecb-2bd0-4770-8b45-485f26e1381f",
    "https://ghibliapi.herokuapp.com/people/591524bc-04fe-4e60-8d61-2425e42ffb2a",
    "https://ghibliapi.herokuapp.com/people/c491755a-407d-4d6e-b58a-240ec78b5061",
    "https://ghibliapi.herokuapp.com/people/f467e18e-3694-409f-bdb3-be891ade1106",
    "https://ghibliapi.herokuapp.com/people/08fffbce4-7f94-476a-95bc-76d3c3969c19",
    "https://ghibliapi.herokuapp.com/people/0f8ef701-b4c7-4f15-bd15-368c7fe38d0a"
  ],
  "species": [
    "https://ghibliapi.herokuapp.com/species/af3910a6-429f-4c74-9ad5-dfe1c4aa04f2",
    "https://ghibliapi.herokuapp.com/species/603428ba-8a86-4b0b-a9f1-65df6abef3d3",
    "https://ghibliapi.herokuapp.com/species/74b7f547-1577-4430-806c-c358c8b6bcf5"
  ],
  "locations": [
    "https://ghibliapi.herokuapp.com/locations/"
  ],
  "vehicles": [
    "https://ghibliapi.herokuapp.com/vehicles/"
  ],
  "url": "https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49",
  "length": null
}
```

# Consumindo API

Consumindo com Postman

The screenshot shows the Postman application interface. At the top, there are tabs for 'Import' and 'Builder' (which is selected), and 'Team Library'. On the right side of the header are icons for 'SYNC OFF', 'Sign In', and notifications. Below the header, there's a search bar with 'https://ghibliapi.herokuapp.com' and a dropdown menu showing 'No Environment'. The main area shows a 'GET' request to 'https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49'. The response status is '200 OK' and the time taken is '780 ms'. The 'Body' tab is selected, displaying a JSON response with a film object and a people array. The JSON is shown in 'Pretty' format, with line numbers 1 through 20 on the left.

```
1 {  
2   "id": "58611129-2dbc-4a81-a72f-77ddfc1b1b49",  
3   "title": "My Neighbor Totoro",  
4   "description": "Two sisters move to the country with their father in order to be closer to their hospitalized mother, and discover the surrounding trees are inhabited by Totoros, magical spirits of the forest. When the youngest runs away from home, the older sister seeks help from the spirits to find her.",  
5   "director": "Hayao Miyazaki",  
6   "producer": "Hayao Miyazaki",  
7   "release_date": "1988",  
8   "rt_score": "93",  
9   "people": [  
10     "https://ghibliapi.herokuapp.com/people/986faac6-67e3-4fb8-a9ee-bad077c2e7fe",  
11     "https://ghibliapi.herokuapp.com/people/d5df3c04-f355-4038-833c-83bd3502b6b9",  
12     "https://ghibliapi.herokuapp.com/people/3031caa8-eb1a-41c6-ab93-dd091b541e11",  
13     "https://ghibliapi.herokuapp.com/people/87b68b97-3774-495b-bf80-495a5f3e672d",  
14     "https://ghibliapi.herokuapp.com/people/d39deecb-2bd0-4770-8b45-485f26e1381f",  
15     "https://ghibliapi.herokuapp.com/people/591524bc-04fe-4e60-8d61-2425e42ffb2a",  
16     "https://ghibliapi.herokuapp.com/people/c491755a-407d-4d6e-b58a-240ec78b5061",  
17     "https://ghibliapi.herokuapp.com/people/f467e18e-3694-409f-bdb3-be891ade1106",  
18     "https://ghibliapi.herokuapp.com/people/08ffbcce4-7f94-476a-95bc-76d3c3969c19",  
19     "https://ghibliapi.herokuapp.com/people/0f8ef701-b4c7-4f15-bd15-368c7fe38d0a"  
20   ]  
}
```

# Consumindo API

Consumindo como um Front

The application displays three movie cards:

- Castle in the Sky**: The orphan Sheeta inherited a mysterious crystal that links her to the mythical sky-kingdom of Laputa. With the help of resourceful Pazu and a rollicking band of sky pirates, she makes her way to the ruins of the once-great civilization. Sheeta and Pazu must outwit the evil Muska, who plans to use Laputa's science to make himself ruler of the world.
- Grave of the Fireflies**: In the latter part of World War II, a boy and his sister, orphaned when their mother is killed in the firebombing of Tokyo, are left to survive on their own in what remains of civilian life in Japan. The plot follows this boy and his sister as they do their best to survive in the Japanese countryside, battling hunger, prejudice, and pride in their own quiet, personal battle.
- My Neighbor Totoro**: Two sisters move to the country with their father in order to be closer to their hospitalized mother, and discover the surrounding trees are inhabited by Totoros, magical spirits of the forest. When the youngest runs away from home, the older sister seeks help from the spirits to find her.

```
const app = document.getElementById('root');
const container = document.createElement('div');
container.setAttribute('class', 'container');

app.appendChild(container);

const request = new XMLHttpRequest();

request.open('GET', 'https://ghibliapi.herokuapp.com/films', true);

request.onload = function(){
  const data = JSON.parse(this.response);

  console.log(data)
  console.log(data[0].title)

  if(request.status >= 200 && request.status < 400){
    console.log("SUCESSO!!!")
    data.forEach(movie => {
      const card = document.createElement('div');
      card.setAttribute('class', 'card');

      const h1 = document.createElement('h1');
      h1.textContent = movie.title;

      const p = document.createElement('p');
      movie.description = movie.description;
      p.textContent = movie.description;

      container.appendChild(card);
      card.appendChild(h1);
      card.appendChild(p);
    });
  }else{
    console.log("erro :c")
  }
}

request.send()
```

# Objetivos



Modelo Server/Client



API e API Rest



URL, Dominio, IP e DNS



JSON



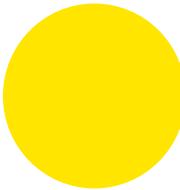
Protocolo HTTP



Consumindo APIs



Request e Response



Para o lar

# Para olar

UHUUUL