1. A Entrega Padrão:

- `Crie uma requisição do tipo GET.
- Tente buscar o CEP da nossa faculdade (ou um famoso, como o
- da Sé em SP: 01001000).

Pergunta: Qual foi o Status Code retornado? O JSON veio completo?

O status code retornado foi 200, JSON veio completo


```
{
  "cep": "01001-000",
  "logradouro": "Praça da Sé",
  "complemento": "lado ímpar",
  "unidade": "",
  "bairro": "Sé",
  "localidade": "São Paulo",
  "uf": "SP",
  "estado": "São Paulo",
  "regiao": "Sudeste",
  "ibge": "3550308",
  "gia": "1004",
  "ddd": "11",
  "siafi": "7107"
}
```


2. O Caso do CEP Inexistente:

- Tente fazer um GET para um CEP que não existe (ex: 99999999).

Desafio: Observe o JSON retornado. O ViaCEP retorna um erro 404 ou ele retorna um JSON com um campo "erro": true? Por que você acha que eles escolheram essa estratégia?


```
{
"erro": true
}
```


status code 200, O ViaCEP retorna **`"erro": true`** ao invés de 404 porque:
A requisição ela funcionou corretamente
Só que o CEP não existe

Isso facilita o tratamento no sistema.

Mudando o Formato (O Camaleão):o O ViaCEP é incrível porque ele aceita outros formatos além do JSON. Tente mudar o final da URL de /json/ para /xml/.

O que mudou na visualização dos dados? Qual formato parece mais fácil de ler no C#?

Veio em XML ao invés de JSON, o formato mais fácil de ler é o **JSON** Porque o C# trabalha muito bem com JSON usando o **System.Text.Json**

<?xml version="1.0" encoding="UTF-8"?>

<xmlcep>

<cep>01001-000</cep>

<logradouro>Praça da Sé</logradouro>

<complemento>lado ímpar</complemento>

<unidade></unidade>

<bairro>Sé</bairro>

<localidade>São Paulo</localidade>

<uf>SP</uf>

<estado>São Paulo</estado>

<regiao>Sudeste</regiao>

<ibge>3550308</ibge>

<gia>1004</gia>

<ddd>11</ddd>

<siafi>7107</siafi>

</xmlcep>

Passo 3: O Relatório do Engenheiro

Após realizar os testes, responda:

1. Verbo HTTP: Qual método você usou? Por que não usamos POST para consultar um CEP?

Usei o método GET.

Não usamos POST porque

GET ele consulta dados
POST ele envia dados

Consultar CEP não envia dados novos.

2. Análise de Dados: Copie o JSON do seu CEP e identifique:

- Qual é a Chave (Key) que guarda o nome da cidade?

  A chave KEY é **"localidade"**

- Qual é a Chave que guarda o nome da rua?

  A chave KEY é **"logradouro"**

3. Integração C#: Se você fosse criar uma classe em C# para receber esses dados, como ficaria a sua propriedade para a chave localidade?

Use o que aprendemos sobre [JsonPropertyName].