Entendendo o Método GET em APIs REST

Por Escola Dnc

Introdução

O método GET é um dos verbos HTTP mais utilizados em APIs RESTful para realizar consultas e obter dados de um servidor. Nesta aula, vamos entender em detalhes como funciona o GET, seus padrões de uso e como coletar informações utilizando esse método.

O que é o método GET?

O método GET serve para **coletar** informações de um servidor através de uma API REST. Quando fazemos uma requisição GET, estamos pedindo algum recurso ou dados do servidor.

A resposta geralmente retorna um dado em formato JSON. Ou seja, fazemos uma pergunta em formato de requisição GET e recebemos um JSON com a resposta.

JSON é um formato de texto padronizado para troca de informações entre cliente e servidor. O servidor entende requisições JSON e responde também em JSON.

Dessa forma, enviamos uma consulta via GET e recebemos um JSON com os dados solicitados ou o status da operação.

Padrões de uso do GET

Apesar de podermos fazer várias coisas com GET, algumas práticas são mais comuns e recomendadas:

- Consultar recursos (usuários, produtos, etc)
- Buscar informações (endereço pelo CEP, previsão do tempo, etc)
- Listar dados de uma coleção (todos os clientes, pedidos recentes, etc)
- Retornar metadados ou options de uma API

Ou seja, os principais usos são para **ler**, **consultar** e **listar** dados.

Outras funcionalidades como criar, editar ou excluir recursos devem utilizar métodos mais apropriados como POST, PUT e DELETE.

Padrão das rotas

As rotas em APIs REST seguem geralmente um padrão que facilita o entendimento:

```
https://api.servidor.com/v1/recursos
```

Onde:

- **api.servidor.com**: domínio da API
- **v1**: versão da api (opcional)
- **recursos**: o tipo de recurso que queremos acessar (usuários, produtos, pedidos, etc)

Alguns exemplos:

```
api.servidor.com/usersapi.servidor.com/v2/customersapi.servidor.com/p
```

Dessa forma fica intuitivo entender os recursos disponíveis apenas olhando as rotas.

Padrão de resposta

Geralmente o retorno de uma requisição GET é um JSON contendo uma coleção de recursos ou dados.

Por exemplo, ao fazer um GET em `api.servidor.com/users` podemos receber:

```
[  {    "id": 1,    "name": "João Silva",    "email":
"joao@email.com"  },  {    "id": 2,    "name": "Maria Souza",
"email": "maria@email.com"      }]
```

Um array JSON com dois objetos, representando dois usuários.

Outro exemplo seria buscar o clima pelo CEP:

```
GET api.servidor.com/weather/01001000
```

Retornando:

```
{  "city": "São Paulo",  "temp": 23,  "forecast": "Sol com algumas
nuvens"}
```

Um objeto JSON representando o clima na cidade de São Paulo.

Boas práticas

Algumas boas práticas ao trabalhar com GET:

- Retorne códigos HTTP corretos (200 OK, 404 Not Found, etc)
- Sempre retorne JSON, a menos que seja um arquivo para download
- Use nomes e padrões intuitivos para as rotas de recursos
- Tenha uma documentação clara e objetiva da sua API

Isso permite que qualquer cliente/dev consiga entender e consumir sua API sem muitas dificuldades.

Parâmetros de requisição

Até agora vimos como fazer GETs básicos para recuperar coleções de recursos completas.

Mas e se quisermos buscar um usuário específico, aplicar filtros mais avançados nas queries ou paginar os resultados?

Aí entram os **parâmetros da requisição**.

Fazendo GETs mais avançados

Podemos passar parâmetros na URL para termos mais controle sobre a resposta.

Por exemplo, para trazer apenas 1 usuário pelo ID:

```
GET api.servidor.com/users/35
```

Ou aplicar um filtro mais específico:

```
GET api.servidor.com/users?age=30&city=Rio de Janeiro
```

Nesse caso, traríamos usuários com 30 anos de idade no Rio de Janeiro.

Outros exemplos:

```
GET api.servidor.com/products?category=eletronics&page=2 GET
api.servidor.com/orders?limit=100&sort=desc
```

Vemos que com parâmetros conseguimos paginar, ordenar, filtrar e muito mais!

Isso permite consultas mais precisas e eficientes via GET.

Considerações Finais

Nesta aula, entendemos em profundidade o método GET para APIs REST:

- Sabemos que GET serve para consultar e obter dados
- Vimos padrões comuns de uso, rotas e respostas
- Aprendemos boas práticas ao projetar APIs
- Entendemos como passar parâmetros para criar queries avançadas

Dominar o GET é essencial para construir e consumir APIs, possibilitando obter dados de forma simples e eficiente.

Espero que este material complemente e reforce seu aprendizado!