Introdução aos Métodos HTTP e API

Por Escola Dnc

O que é uma API?

API (Application Programming Interface) é uma interface de programação de aplicações que permite a comunicação entre sistemas e aplicações diferentes. A API expõe rotinas e funcionalidades de um sistema para serem consumidas por outro sistema ou aplicação.

Por exemplo, quando você usa um aplicativo de celular que mostra a previsão do tempo, esse aplicativo está consumindo uma API de algum serviço de meteorologia que disponibiliza essas informações.

As APIs utilizam o protocolo HTTP para essa comunicação, enviando requisições HTTP com métodos como GET, POST, PUT, etc para realizar operações no sistema que disponibiliza a API.

Relacionamento entre Cliente e Servidor

A comunicação entre o cliente (quem consome a API) e o servidor (quem disponibiliza a API) é muito similar a um relacionamento entre pessoas.

Assim como em um relacionamento entre pessoas, onde ambos os lados precisam "ceder um pouco" e se comunicar de maneira adequada, no caso da API o cliente precisa fazer as requisições de maneira que o servidor entenda e vice-versa.

Por exemplo, se uma API espera receber o CEP em uma requisição GET para retornar informações sobre uma rua, o cliente precisa enviar o CEP corretamente, caso contrário o servidor não conseguirá responder adequadamente.

Então é muito importante que tanto o cliente quanto o servidor sigam um padrão de comunicação, que no caso das APIs é definido pelo protocolo HTTP.

Principais Métodos HTTP

Os principais métodos HTTP utilizados em APIs são:

- **GET**: Utilizado para realizar a leitura/consulta de informações no servidor.
- **POST**: Utilizado para criar/inserir novos recursos no servidor.
- **PUT**: Utilizado para atualizar/modificar recursos existentes no servidor.
- **DELETE**: Utilizado para excluir/deletar recursos do servidor.

Vamos explorar cada um desses métodos em mais detalhes:

GET

O método GET é utilizado para consultar informações no servidor. Você informa os parâmetros da consulta e o servidor retorna as informações correspondentes.

Exemplos:

- Consultar todos os usuários ativos:

```
GET /users?status=active
```

- Consultar dados de um usuário pelo ID:

```
GET /users/123
```

No GET você informa os filtros e informações que deseja buscar e o servidor retorna essas informações em caso de sucesso, geralmente no formato JSON.

POST

O método POST é utilizado quando você deseja criar/inserir um novo recurso no servidor.

Você envia os dados do novo recurso no corpo da requisição e o servidor irá processar esses dados, salvar em seu banco de dados ou estrutura de dados, e retornar uma resposta sobre o sucesso ou falha dessa operação.

Exemplo:

```
POST /users{    "name": "João Silva",    "email": "joao@email.com",
"phone": "9999-9999" }
```

No exemplo acima, enviamos via POST os dados para criar um novo usuário, com nome, e-mail e telefone. O servidor irá salvar esses dados e retornar a resposta sobre a criação desse usuário.

PUT

O método PUT é muito parecido com o POST, porém ao invés de criar um novo recurso, ele atualiza um recurso existente.

Para isso, é preciso informar o identificador único desse recurso (como um ID numérico) junto com os dados que devem ser alterados.

Exemplo:

```
PUT /users/123 {    "name": "João Silva",    "phone": "8888-8888"}
```

No exemplo acima, estamos atualizando o telefone do usuário de ID 123, mantendo o nome igual. O servidor irá identificar esse usuário pelo ID, alterar o telefone para o novo valor enviado e retornar uma resposta sobre o sucesso ou não dessa operação.

DELETE

O método DELETE, como o próprio nome diz, é utilizado para excluir um recurso existente no servidor.

Para isso, da mesma forma que no PUT, precisamos informar o identificador único desse recurso para que o servidor possa encontrá-lo e excluí-lo.

Exemplo:

```
DELETE /users/123
```

Nesse caso, estamos excluindo o usuário de ID 123 do sistema. O servidor irá buscar esse usuário, removê-lo do seu banco de dados ou estrutura de armazenamento, e retornar uma resposta sobre o sucesso ou não dessa operação.

Outros Métodos HTTP

Além dos principais métodos citados, existem também os métodos OPTIONS e PATCH que são menos utilizados.

O método OPTIONS serve para que o cliente possa consultar quais operações são permitidas em um determinado recurso ou rota. Ou seja, quais métodos HTTP aquele endpoint suporta.

Já o PATCH serve para atualizar apenas dados parciais de um recurso, sem a necessidade de enviar todos os dados como no PUT.

Status Codes HTTP

Além dos métodos HTTP, as APIs também utilizam códigos de status HTTP nas respostas para indicar o resultado de cada requisição.

Alguns dos principais códigos de status:

- **200 OK** - Requisição processada com sucesso.
- **201 Created** - Novo recurso criado com sucesso.
- **400 Bad Request** - Requisição inválida, sintaxe errada.
- **401 Unauthorized** - Usuário não autenticado
- **403 Forbidden**- Usuário não tem permissão de acesso
- **404 Not Found** - Recurso não encontrado
- **500 Internal Server Error** - Erro interno do servidor

O cliente pode utilizar esses códigos de status nas respostas para identificar se a requisição foi bem sucedida, falhou por algum erro de validação, não tem permissão de acesso, recurso não existe, ou algum outro cenário.

Isso facilita o tratamento das respostas nas integrações com APIs.

Vantagens de utilizar APIs

A utilização de APIs traz muitas vantagens, entre elas:

- **Modularidade** - As APIs permitem que sistemas sejam construídos de forma modular, reutilizando funcionalidades de outros sistemas por meio das APIs.
- **Escalabilidade** - Como as funcionalidades ficam separadas em diferentes sistemas, cada parte pode ser escalada de maneira independente.
- **Experiência do Usuário** - Por não precisar reinventar a roda, as aplicações podem oferecer uma melhor experiência, focando no seu diferencial.
- **Inovação** - Empresas podem disponibilizar inovações em seus sistemas de forma muito mais rápida por meio de APIs.
- **Novos Modelos de Negócio** - APIs abrem espaço para novos modelos de negócio baseados em compartilhamento e reutilização de funcionalidades entre aplicações.

Conclusão

As APIs e os métodos HTTP são a base para a integração e comunicação entre diferentes sistemas e aplicações na era digital moderna.

Entender esses conceitos é fundamental para desenvolvedores que desejam construir aplicações escaláveis, modularizadas e que possam oferecer uma excelente experiência ao usuário final.

Esperamos que este ebook tenha sido útil para introduzir os principais conceitos e métodos envolvidos na utilização de APIs. Nos próximos capítulos iremos aprofundar ainda mais nesses tópicos.