

Protocolo HTTP



Objetivo da Aula

- Apresentar o funcionamento do protocolo HTTP, seus métodos, cabeçalhos e códigos de status.
 - Demonstrar o uso de requisições HTTP na prática com **PowerShell**.
 - Realizar exercícios práticos para consolidar o aprendizado.
-



1. Definição do Protocolo HTTP



O que é HTTP?

- HTTP (*HyperText Transfer Protocol*) é um protocolo de comunicação para a transferência de dados na Web.
- Funciona no modelo **Cliente-Servidor** e é **stateless** (não mantém estado entre requisições).



Arquitetura HTTP

1. **Cliente** → Solicita recursos (Ex.: Navegador, API, Aplicações).
2. **Servidor** → Processa a solicitação e retorna uma resposta.



Exemplo Prático - Verificando uma Requisição HTTP

No PowerShell, execute:

```
Invoke-WebRequest -Uri "https://www.google.com" -UseBasicParsing
```



Saída esperada:

Retorno da resposta HTML da página inicial do Google.



2. Métodos HTTP

O que são Métodos HTTP?

Os métodos HTTP determinam a ação que será realizada sobre um recurso.

Principais Métodos HTTP

Método	Descrição
GET	Solicita um recurso do servidor (padrão em navegadores).
POST	Envia dados para o servidor.
PUT	Atualiza um recurso existente.
DELETE	Remove um recurso do servidor.

Exemplo - Testando Métodos HTTP com PowerShell

Teste de GET

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/posts/1" `
-Method Get `
-Headers @{ "Accept" = "application/json" } `
-UseBasicParsing
```

 **Saída esperada:** Um JSON com o post de ID 1.

Teste de POST

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/posts" `
-Method Post `
-Headers @{ "Content-Type" = "application/json" } `
-Body '{ "title": "Teste", "body": "Conteúdo do post" }' `
-UseBasicParsing
```

 **Saída esperada:** Resposta com **código 201** (Created).

3. Tipos de Passagem de Parâmetros

Como enviar dados em uma requisição HTTP?

- **Query String** → Parâmetros na URL (**GET**).

- **Body (corpo da requisição)** → Envio estruturado (`POST/PUT`).

✓ Exemplo - Enviando Parâmetros pela Query String

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/comments?postId=1" `
-Method Get `
-UseBasicParsing
```

📌 **Saída esperada:** Lista de comentários do post com ID 1.

📌 4. Cabeçalhos HTTP e Midia Types

🤔 HTTP - REQUEST RESPONSE

📖 O que são cabeçalhos HTTP?

- Informações adicionais enviadas no **request** e no **response**.
- Exemplos: `Content-Type` , `Authorization` , `Accept` , `User-Agent` .

✓ Exemplo - Testando Cabeçalhos no PowerShell

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/posts" `
-Method Get `
-Headers @{ "Accept" = "application/json"; "User-Agent" = "Mozilla/5.0" } `
-UseBasicParsing
```

📌 **Saída esperada:** JSON com a lista de posts.

📖 **Mídia Types (Content-Type)**

- `application/json` → JSON
- `text/html` → HTML
- `text/plain` → Texto puro
- `application/xml` → XML

5. Códigos de Status HTTP

O que são códigos de status?

- Informam o resultado da requisição.

Código	Categoria	Descrição	
200	Sucesso	Requisição OK.	
201	Criado	Recurso criado com sucesso.	
400	Erro do Cliente	Requisição inválida.	
401	Não Autorizado	Token inválido.	
404	Não Encontrado	Recurso não existe.	
500	Erro do Servidor	Erro interno no servidor.	

Exemplo - Verificando Código de Status


```
try {  
    $response = Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/invalid-url" -UseBasicParsing  
    $response.StatusCode  
} catch {  
    Write-Host "Erro: $($_.Exception.Message)"  
    if ($_.Exception.Response) {  
        Write-Host "Código de status: $($_.Exception.Response.StatusCode.Value__)"  
    }  
}
```

 Saída esperada: 404

6. Exercícios Práticos

1. Faça uma requisição GET para recuperar uma lista de usuários.

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/users" `
-Method Get `
-Headers @{ "Accept" = "application/json" } `
-UseBasicParsing
```

 **Pergunta:** Qual o nome do primeiro usuário?


◆ 2. Envie um novo post via POST e veja o código de resposta.

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/posts" `
-Method Post `
-Headers @{ "Content-Type" = "application/json" } `
-Body '{ "title": "Novo Post", "body": "Conteúdo" }' `
-UseBasicParsing
```

 **Pergunta:** Qual foi o código de status retornado?

◆ 3. Teste uma requisição DELETE e veja o código de status.

```
Invoke-WebRequest -Uri "https://jsonplaceholder.typicode.com/posts/1" `
-Method Delete `
-Headers @{ "Accept" = "application/json" } `
-UseBasicParsing
```

 **Pergunta:** O recurso foi realmente deletado?

Conclusão

 **O que aprendemos hoje?**

- ✓ O que é HTTP e como funciona.
 - ✓ Como usar métodos HTTP (GET, POST, PUT, DELETE).
 - ✓ Como passar parâmetros e interpretar respostas.
 - ✓ Como testar APIs com PowerShell.
-