Os métodos HTTP (Hypertext Transfer Protocol) são usados pelo protocolo da internet para controlar como informações são transferidas na web. Cada vez que um usuário visita um site, o seu navegador envia uma requisição para o servidor usando um desses métodos, e o servidor responde com o conteúdo da página.

Existem vários métodos HTTP diferentes, cada um com sua própria finalidade e utilização. Aqui estão alguns exemplos de métodos HTTP comuns e suas aplicações:

* O método **GET** é usado para solicitar dados de um recurso específico. Por exemplo, quando você visita um site, o seu navegador envia uma requisição **GET** para o servidor para solicitar o conteúdo da página.

EX: GET /index.html HTTP/1.1

Host: www.exemplo.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: pt-BR,pt;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

Neste código, a primeira linha especifica que a requisição é do tipo **GET** e está solicitando o arquivo **index.html** do servidor. As linhas seguintes incluem cabeçalhos adicionais com informações sobre o navegador que está fazendo a requisição, as linguagens e codificações de texto aceitáveis pelo navegador, e a conexão que está sendo usada.

Quando o servidor recebe esta requisição, ele enviará uma resposta com o conteúdo do arquivo **index.html** solicitado.

* O método **POST** é usado para enviar dados para o servidor. Por exemplo, quando você preenche um formulário online e clica em "Enviar", o seu navegador envia uma requisição **POST** para o servidor com os dados do formulário.

EX:

POST /formulario.php HTTP/1.1

Host: www.exemplo.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: pt-BR,pt;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Content-Type: application/x-www-form-urlencoded

Content-Length: 37

Connection: keep-alive

nome=Joao&idade=25&cidade=Sao+Paulo

Neste código, a primeira linha especifica que a requisição é do tipo POST e está enviando dados para o arquivo formulario.php no servidor. As linhas seguintes incluem cabeçalhos adicionais com informações sobre o navegador que está fazendo a requisição, as linguagens e codificações de texto aceitáveis pelo navegador, e o tipo e tamanho dos dados que estão sendo enviados.

Na última linha, os dados em si são enviados no formato de "chave-valor" (nome=Joao&idade=25&cidade=Sao+Paulo), onde cada par chave-valor representa um campo do formulário que foi preenchido pelo usuário.

Quando o servidor recebe esta requisição, ele processará os dados enviados e enviará uma resposta apropriada de volta para o navegador. Isso pode incluir uma mensagem de sucesso ou erro, dependendo do resultado da operação.

Parte superior do formulário

Parte inferior do formulário

* O método **PUT** é usado para atualizar um recurso existente no servidor. Por exemplo, se você quiser atualizar seu perfil em um site de rede social, o seu navegador enviará uma requisição **PUT** para o servidor com as novas informações do perfil.

EX:

PUT /perfil.php HTTP/1.1

Host: www.exemplo.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: pt-BR,pt;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Content-Type: application/x-www-form-urlencoded

Content-Length: 37

Connection: keep-alive

nome=Joao&idade=26&cidade=Rio+de+Janeiro

Neste código, a primeira linha especifica que a requisição é do tipo **PUT** e está enviando dados para o arquivo **perfil.php** no servidor. As linhas seguintes incluem cabeçalhos adicionais com informações sobre o navegador que está fazendo a requisição, as linguagens e codificações de texto aceitáveis pelo navegador, e o tipo e tamanho dos dados que estão sendo enviados.

Na última linha, os dados em si são enviados no formato de "chave-valor" (**nome=Joao&idade=26&cidade=Rio+de+Janeiro**), onde cada par chave-valor representa um campo do formulário que foi preenchido pelo usuário.

Quando o servidor recebe esta requisição, ele atualizará o perfil do usuário com as novas informações enviadas e enviará uma resposta apropriada de volta para o navegador. Isso pode incluir uma mensagem de sucesso ou erro, dependendo do resultado da operação.

* O método **DELETE** é usado para excluir um recurso do servidor. Por exemplo, se você quiser excluir sua conta em um site de rede social, o seu navegador enviará uma requisição **DELETE** para o servidor para excluir todos os seus dados.

EX:

DELETE /conta.php HTTP/1.1

Host: www.exemplo.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: pt-BR,pt;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

Neste código, a primeira linha especifica que a requisição é do tipo **DELETE** e está solicitando a exclusão de uma conta do arquivo **conta.php** no servidor. As linhas seguintes incluem cabeçalhos adicionais com informações sobre o navegador que está fazendo a requisição, as linguagens e codificações de texto aceitáveis pelo navegador, e a conexão que está sendo usada.

Quando o servidor recebe esta requisição, ele excluirá a conta do usuário e enviará uma resposta apropriada de volta para o navegador. Isso pode incluir uma mensagem de sucesso ou erro, dependendo do resultado da operação.

Esses são apenas alguns exemplos de métodos HTTP e suas aplicações. Existem outros métodos disponíveis, como **HEAD**, **OPTIONS** e **CONNECT**, cada um com suas próprias funções e usos específicos.

Em resumo, os métodos HTTP são usados pelo protocolo da internet para controlar como informações são transferidas na web. Cada vez que um usuário visita um site, o seu navegador envia uma requisição para o servidor usando um desses métodos, e o servidor responde com o conteúdo da página. É importante escolher o método adequado de acordo com a ação que você deseja realizar no servidor.

WSDL (Web Services Description Language) é uma linguagem de descrição de serviços web. Ele é usado para descrever os serviços web disponíveis em um determinado endereço da web, incluindo sua localização, os métodos que ele oferece, os parâmetros que cada método aceita e os tipos de dados que ele retorna.

Essa descrição é importante porque permite que outras aplicações encontrem e utilizem os serviços web de maneira fácil e consistente. Por exemplo, se uma aplicação precisar consumir um serviço web para obter dados de previsão do tempo, ela pode ler a descrição WSDL do serviço para saber como enviar uma requisição e interpretar a resposta.

WSDL é usado principalmente em conjunto com outras tecnologias de serviços web, como SOAP (Simple Object Access Protocol) e UDDI (Universal Description, Discovery and Integration). Ele é escrito em XML e geralmente é disponibilizado em um endereço da web em formato de arquivo, como **http://www.exemplo.com/servico.wsdl**.