Aluno: Marcelo Augusto

FAP – Softex

Faça um projeto explicando como funcionam os métodos HTTP com exemplos de requisições, sua utilização e em qual momento cada um dos métodos deve ser utilizado.

Além disso, descreva o que é WSDL, seu significado e sua função.

Resposta:

Nas requisições, é comum especificarmos o que chamamos de **método** HTTP. Na versão 1.1 do protocolo HTTP (que é a que todos usamos atualmente) temos 9 verbos diferentes.

GET

Essa é a requisição mais comum de todas. Através dessa requisição nós pedimos a representação de um recurso: que pode ser um arquivo html, xml, json, etc.

Um exemlo de requisição GET seria:

**GET /entendendo-o-net-core-parte-3-o-coreclr/ HTTP/1.1**

**User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)**

**Host:** [**http://marceloaugusto.com**](http://marceloaugusto.com)

**Connection: Keep-Alive**

POST

O método POST é utilizado quando queremos **criar** um recurso. Quando usamos POST, os dados vão no corpo da requisição e não na URI.

PUT

Requisita que um recurso seja "guardado" na URI fornecida. Se o recurso já existir, ele deve ser atualizado. Se não existir, pode ser criado.

DELETE

Exclui o recurso especificado.

TRACE

Devolve a mesma requisição que for enviada veja se houve mudança e/ou adições feitas por servidores intermediários.

OPTIONS

Retorna os métodos HTTP suportados pelo servidor para a URL especificada.

PATCH

Serve para atualizar **partes** de um recurso, e não o recurso todo.

CONNECT

Converte a requisição de conexão para um túnel TCP/IP transparente, geralmente para facilitar a comunicação criptografada com SSL (HTTPS) através de um proxy HTTP não criptografado.

HEAD

Retorna somente os cabeçalhos de uma resposta.

Idempotência

Alguns métodos, como o GET, podem ser chamados **diversas** vezes seguidas sem problema nenhum: a resposta será sempre a mesma. Isso porque quando fazermos uma requisição do tipo GET não estamos **alterando** nada no servidor, somente consultando informações.  
Já quando estamos fazendo um POST estamos **criando** um novo recurso.  
Sabe quando você manda atualizar a página do seu navegador e ele pergunta se você quer realmente atualizar essa página? Que ele vai reenviaar as informações e tal? Então, essa página está fazendo uma requisição do tipo POST.  
Se executarmos a mesma requisição POST duas vezes, corremos o risco de criar dois recursos iguais.  
Na maioria das vezes não queremos isso né?

Os métodos que não alteram nada no servidor e que podemos chamar várias vezes são o que chamamos de métodos **idempotentes**.

GET, OPTIONS, HEAD, PUT, TRACE, CONNECT e DELETE são idempotentes.