**Aluno: Tomás de Farias Ribeiro Caldas**

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.

**Método GET**

O método GET é usado para recuperar informações de um determinado servidor usando um determinado URI. Solicitações usando GET devem apenas recuperar dados e não devem ter nenhum outro efeito sobre os dados.

Uma solicitação GET recupera dados de um servidor da Web especificando parâmetros na parte da URL da solicitação. Este é o principal método usado para recuperação de documentos. O exemplo a seguir usa o método GET para buscar hello.htm:

GET /hello.htm HTTP/1.1

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

Host: www.tutorialspoint.com

Accept-Language: en-us

Accept-Encoding: gzip, deflate

Connection: Keep-Alive

A resposta do servidor em relação à solicitação GET acima será a seguinte:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT

ETag: "34aa387-d-1568eb00"

Vary: Authorization,Accept

Accept-Ranges: bytes

Content-Length: 88

Content-Type: text/html

Connection: Closed

<html>

<body>

<h1>Hello, World!</h1>

</body>

</html>

**Método HEAD**

Igual a GET, mas transfere apenas a linha de status e a seção de cabeçalho.

O método HEAD é funcionalmente semelhante ao GET, exceto que o servidor responde com uma linha de resposta e cabeçalhos, mas sem corpo de entidade. O exemplo a seguir usa o método HEAD para buscar informações de cabeçalho sobre hello.htm:

HEAD /hello.htm HTTP/1.1

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

Host: www.tutorialspoint.com

Accept-Language: en-us

Accept-Encoding: gzip, deflate

Connection: Keep-Alive

A resposta do servidor em relação à solicitação HEAD acima será a seguinte:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT

ETag: "34aa387-d-1568eb00"

Vary: Authorization,Accept

Accept-Ranges: bytes

Content-Length: 88

Content-Type: text/html

Connection: Closed

Você pode notar que aqui o servidor não envia nenhum dado após o cabeçalho.

**Método POST**

Uma solicitação POST é usada para enviar dados ao servidor, por exemplo, informações do cliente, upload de arquivos etc. usando formulários HTML.

O método POST é usado quando você deseja enviar alguns dados para o servidor, por exemplo, atualização de arquivo, dados de formulário, etc. O exemplo a seguir utiliza o método POST para enviar dados de formulário ao servidor, que serão processados ​​por um process.cgi e, finalmente, uma resposta será retornada:

POST /cgi-bin/process.cgi HTTP/1.1

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

Host: www.tutorialspoint.com

Content-Type: text/xml; charset=utf-8

Content-Length: 88

Accept-Language: en-us

Accept-Encoding: gzip, deflate

Connection: Keep-Alive

<?xml version="1.0" encoding="utf-8"?>

<string xmlns="http://clearforest.com/">string</string>

O script do lado do servidor process.cgi processa os dados passados ​​e envia a seguinte resposta:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT

ETag: "34aa387-d-1568eb00"

Vary: Authorization,Accept

Accept-Ranges: bytes

Content-Length: 88

Content-Type: text/html

Connection: Closed

<html>

<body>

<h1>Request Processed Successfully</h1>

</body>

</html>

**Método PUT**

Substitui todas as representações atuais do recurso de destino pelo conteúdo carregado.

O método PUT é usado para solicitar ao servidor que armazene o corpo da entidade incluído em um local especificado pelo URL fornecido. O exemplo a seguir solicita que o servidor salve o corpo da entidade fornecido em **hello.htm** na raiz do servidor:

PUT /hello.htm HTTP/1.1

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

Host: www.tutorialspoint.com

Accept-Language: en-us

Connection: Keep-Alive

Content-type: text/html

Content-Length: 182

<html>

<body>

<h1>Hello, World!</h1>

</body>

</html>

O servidor armazenará o corpo da entidade fornecido no arquivo **hello.htm** e enviará a seguinte resposta de volta ao cliente:

HTTP/1.1 201 Created

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Content-type: text/html

Content-length: 30

Connection: Closed

<html>

<body>

<h1>The file was created.</h1>

</body>

</html>

**Método DELETE**

Remove todas as representações atuais do recurso de destino fornecidas por um URI.

O método DELETE é usado para solicitar ao servidor que exclua um arquivo em um local especificado pelo URL fornecido. O exemplo a seguir solicita que o servidor exclua o arquivo **hello.htm fornecido** na raiz do servidor:

DELETE /hello.htm HTTP/1.1

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

Host: www.tutorialspoint.com

Accept-Language: en-us

Connection: Keep-Alive

O servidor excluirá o arquivo mencionado **hello.htm** e enviará a seguinte resposta de volta ao cliente:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Content-type: text/html

Content-length: 30

Connection: Closed

<html>

<body>

<h1>URL deleted.</h1>

</body>

</html>

**Método CONNECT**

Estabelece um túnel para o servidor identificado por um determinado URI.

O método CONNECT é usado pelo cliente para estabelecer uma conexão de rede com um servidor web por HTTP. O exemplo a seguir solicita uma conexão com um servidor da Web em execução no host tutorialspoint.com:

CONNECT www.tutorialspoint.com HTTP/1.1

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

A conexão é estabelecida com o servidor e a seguinte resposta é enviada de volta ao cliente:

HTTP/1.1 200 Connection established

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

**Método OPTIONS**

Descreve as opções de comunicação para o recurso de destino.

O método OPTIONS é utilizado pelo cliente para conhecer os métodos HTTP e outras opções suportadas por um servidor web. O cliente pode especificar uma URL para o método OPTIONS ou um asterisco (\*) para se referir a todo o servidor. O exemplo a seguir solicita uma lista de métodos suportados por um servidor web executado em tutorialspoint.com:

OPTIONS \* HTTP/1.1

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

O servidor enviará uma informação baseada na configuração atual do servidor, por exemplo:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Allow: GET,HEAD,POST,OPTIONS,TRACE

Content-Type: httpd/unix-directory

**Método TRACE**

Executa um teste de loopback de mensagem ao longo do caminho para o recurso de destino.

O método TRACE é usado para ecoar o conteúdo de uma solicitação HTTP de volta ao solicitante, que pode ser usado para fins de depuração no momento do desenvolvimento. O exemplo a seguir mostra o uso do método TRACE:

TRACE / HTTP/1.1

Host: www.tutorialspoint.com

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

O servidor enviará a seguinte mensagem em resposta à solicitação acima:

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache/2.2.14 (Win32)

Connection: close

Content-Type: message/http

Content-Length: 39

TRACE / HTTP/1.1

Host: www.tutorialspoint.com

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

**O que é um arquivo WSDL?**

**A Linguagem de Descrição de Serviços da Web (WSDL) é uma linguagem baseada em XML para descrever serviços da Web .**

Um arquivo WSDL é escrito em XML e define as operações em um serviço da Web, as mensagens usadas por cada operação e a aparência das mensagens.

Um arquivo WSDL válido contém todas as informações necessárias para enviar uma solicitação a um serviço da web. Pense nisso como um “plano” ou um manual de instruções para interagir com um serviço da web.

O arquivo WSDL geralmente é distribuído aos consumidores de um serviço da Web, diretamente como download do próprio serviço ou distribuído separadamente.

Um arquivo WSDL é projetado para que os computadores entendam. É um contrato, que pode ser entendido por máquinas.

É basicamente um manual ou receita que descreve um serviço web.

Você pode usar as informações em um arquivo WSDL para:

* Chame o serviço web remoto que o WSDL descreve – escrevendo algum código ou usando uma ferramenta de teste
* Escreva seu próprio serviço da web, que implementa as definições no WSDL
* Crie mensagens de solicitação ou resposta de amostra para o serviço