Protocolo HTTP

Eduardo Ferreira dos Santos

Ciência da Computação
Centro Universitário de Brasília – UniCEUB

Fevereiro, 2017

Sumário

O que é HTTP

- HTTP: *Hypertext Transfer Protocol*;
- Protocolo utilizado para transmitir dados através da *World Wide Web*;
- Na Web os dados transmitidos (arquivos, imagens, etc) são chamados de recursos;
- Web x Internet x redes TCP/IP;

Conexão HTTP

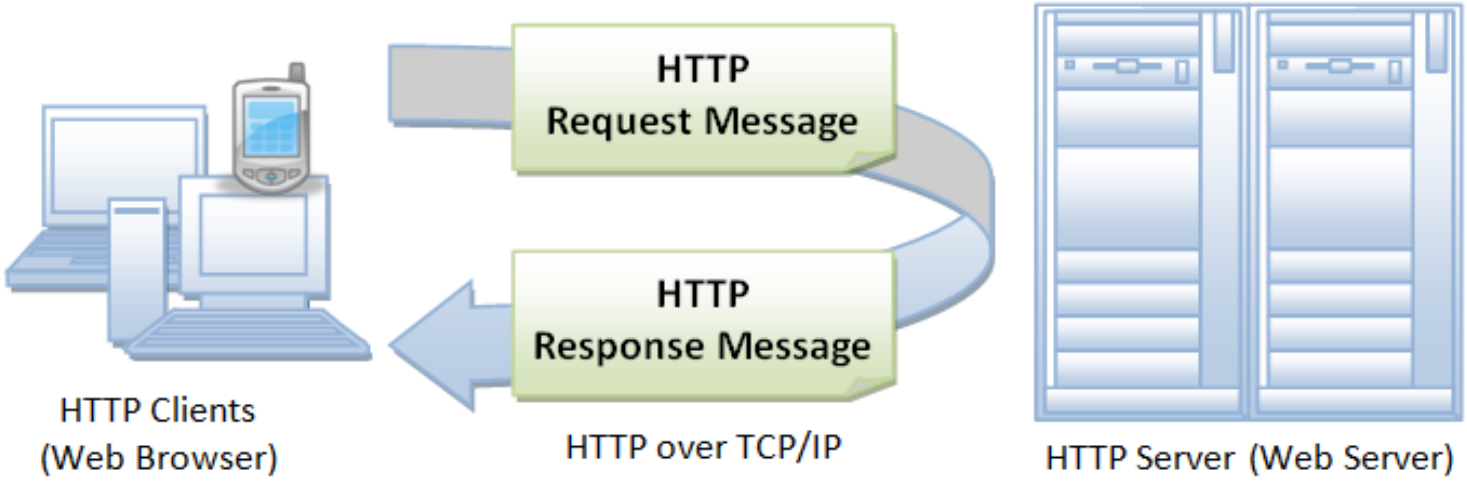

Figura 1.1: Modelo *request/response* do HTTP [Hock-Chuan, 2009]

Recursos

Um recurso é qualquer pedaço de informação identificado por uma URL. [Marshall, 2012]

URL – Uniform Resource Location

protocol://hostname:port/path-and-file-name

- A URL possui quatro partes:
  1. **Protocolo** Protocolo utilizado tanto por cliente quanto por servidor. Ex.: telnet, HTTP, etc.
  2. **Endereço** DNS do domínio ou endereço IP;
  3. **Porta** Porta que está escutando requisições dos clientes;
  4. **Caminho e arquivo** Nome e local do arquivo no diretório do servidor.

Estrutura das transações HTTP

- O protocolo HTTP utiliza o modelo client-servidor;
- Realiza um fluxo de operações para transmissão das informações:
  1. O cliente HTTP abre uma conexão com o servidor HTTP;
  2. Depois envia uma requisição;
  3. o servidor retorna uma resposta, normalmente contendo o recurso solicitado;
  4. Após o envio da resposta o servidor fecha a conexão.
- O protocolo HTTP é *stateless*, ou seja, não guarda informação sobre as conexões realizadas e os dados enviados.

Formato do HTTP [Marshall, 2012]

HTTP

<linha inicial, diferente para requisição e resposta>
Header1: value1
Header2: value2
Header3: value3

<corpo da mensagem>

Requisição

Linha inicial da requisição

```
GET /path/to/file/index.html HTTP/1.0
```

- GET representa o método http;
- A parte iniciada com */path* é chamada de caminho ou URI – *Uniform Resource Identifier*;
- A versão do protocolo HTTP é sempre representada no formato HTTP/x.x onde x.x representa a versão do protocolo.

Resposta

Linha inicial da resposta

HTTP/1.0 200 OK

- A versão do HTTP deve estar no mesmo formato da requisição (HTTP/x.x);
- O código de status é legível apenas por máquinas e representa uma resposta interpretável para a sua requisição;
- O texto após o código de status é para facilitar a visualização da resposta.

Outros componentes

- Cabeçalhos (*Header lines*);
- Corpo da mensagem (*Message body*);
- O cabeçalho descreve o conteúdo do corpo da requisição;
- Traz informações que facilitam o parsing da resposta HTTP.

Exemplo completo

```
eduardo@escritorio03:~$ telnet www.google.com 80
Trying 216.58.202.36...
Connected to www.google.com.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.1 302 Found
Cache-Control: private
Content-Type: text/html; charset=UTF-8
Location: http://www.google.com.br/?gfe_rd=cr&ei=rbaiV9vhJaTL8ge0soLQCg
Content-Length: 262
Date: Thu, 04 Aug 2016 03:29:49 GMT

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>302 Moved</TITLE></HEAD><BODY>
<H1>302 Moved</H1>
The document has moved
<A HREF="http://www.google.com.br/?gfe_rd=cr&amp;ei=rbaiV9vhJaTL8ge0soLQCg">here</A>.
</BODY></HTML>
```

Figura 2.1: Exemplo completo de uma requisição e uma resposta HTTP

Outros métodos HTTP

HEAD Solicita somente os Headers ao servidor Web;

POST Envia dados para o servidor;

RESTfull API Utiliza um novo conjunto de métodos HTTP para realizar operações diferentes.

Host: Header

- Utilização de múltiplos domínios no mesmo IP;
- É o único parâmetro que passa a ser obrigatório no HTTP/1.1.

Conexão utilizando host

GET /path/file.html HTTP/1.1
Host: www.host1.com:80

[linha em branco acima]

Chunked Transfer-Encoding

- O servidor Web começa o envio da resposta antes de saber seu tamanho total;
- Quebra a resposta em pequenos pedaços (*chunks*) e envia sequencialmente.

Resposta em pedaços (*chunks*)

```
HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: text/plain
Transfer-Encoding: chunked

1a; ignore-stuff-here
abcdefghijklmnopqrstuvwxyz
10
1234567890abcdef
0
some-footer: some-value
another-footer: another-value

[linha em branco acima]
```

Resposta normal (sem *chunks*)

```
HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: text/plain
Content-Length: 42
some-footer: some-value
another-footer: another-value


abcdefghijklmnopqrstuvwxyz1234567890abcdef
```

Conexões persistentes

- No protocolo 1.0 a conexão é fechada após o ciclo requisição/resposta estar completo;
- Abrir e fechar conexões TCP consome uma quantidade considerável de recursos computacionais;
- No protocolo 1.1 as conexões persistentes são o padrão:
  - Abra uma conexão;
  - Envia várias requisições;
  - Leia os resultados na sequência.
- A introdução do Header Connection: close indica que a conexão deve ser fechada imediatamente.

100 Continue

- O servidor indica que recebeu a primeira parte da requisição;
- Vai continuar enviando até terminar;
- Útil em conexões lentas;
- Gerenciadores de download.

Servidores HTTP/1.1

- Header Host obrigatório;
- Utilização de URL's absolutas;
- Chunked Transfer-Encoding;
- Conexões persistentes e o Header Connection-Close;
- Utilização do 100 Continue;
- Header Date:
- Header If-Modified-Since;
- Suporte aos métodos HEAD e GET (além de outros).

OBRIGADO!!!
PERGUNTAS???


Hock-Chuan, C. (2009).
Introduction to HTTP basics.
Disponível em https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html Acessado em 03/08/2016.


Marshall, J. (2012).
Http made really easy.
Disponível em http://jmarshall.com/easy/http Acessado em 03/08/2016.