



PUC Minas

Protocolo HTTP

Introdução

Prof. Rommel Carneiro



Tópicos

- Definição e características
- Histórico de Versões
- Requisição e Resposta
- Métodos
- Cabeçalhos
- Novas versões



Protocolo HTTP

O *Hypertext Transfer Protocol* (HTTP) é um protocolo da camada de aplicação para sistemas distribuídos e colaborativos de informação no formato de hipertextos.

RFC - 2068

Características

- Requer a atuação de dois programas: Cliente e Servidor
- Atua na camada de aplicação da pilha TCP/IP
- A comunicação entre o cliente/servidor utiliza conexões TCP (e UDP no caso do HTTP v3.0)
- O servidor HTTP, por padrão, utiliza a porta 80
- Protocolo que não guarda estado do cliente (stateless)

Protocolo HTTP – Histórico de Versões

1991 • HTTP/0.9
1994 • HTTPS
1996 • HTTP/1.0
1999 • HTTP/1.1

2009 • SPDY 1.0

2015 • HTTP/2
2016 • QUIC
2018 • HTTP/3



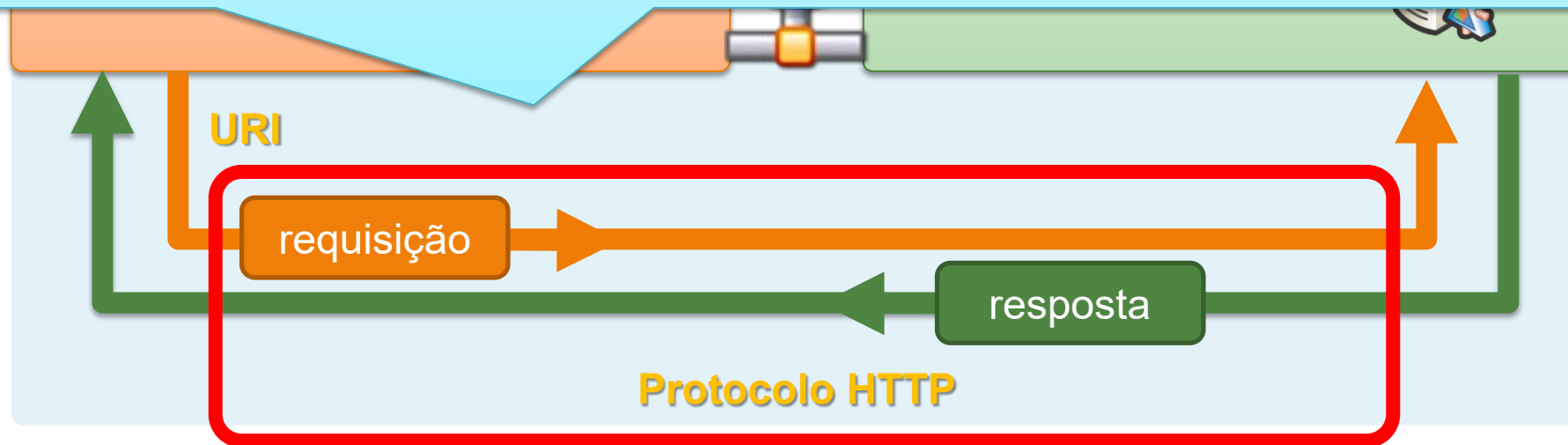
Protocolo HTTP – Histórico de Versões

- **1991** - O HTTP 0.9 é lançado
- **1994** - O HTTPS foi criado pela Netscape
- **1996** - O HTTP 1.0 foi lançado
 - Conceito de cabeçalhos
 - Códigos de Status
- **1999** - O HTTP 1.1 foi lançado
 - Conexões TCP persistentes
 - Suporte a Virtual Host (Cabeçalho Host)
 - Autenticação Digest
 - Controle de cache
 - Possibilidade de compressão de dados
- **2009** - Google propõe o SPDY
- **2015** - O HTTP 2.0 é lançado
 - Baseado no SPDY
 - Compressão de dados obrigatória
 - Cabeçalhos binários
 - Requisições paralelas
 - Envio apenas de cabeçalhos alterados nas próximas requisições
 - Priorização de requisições
 - Server PUSH – Envio automático de arquivos adicionais.
- **2018** – O HTTP 3 é lançado
 - Protocolo de transporte QUIC baseado em UDP

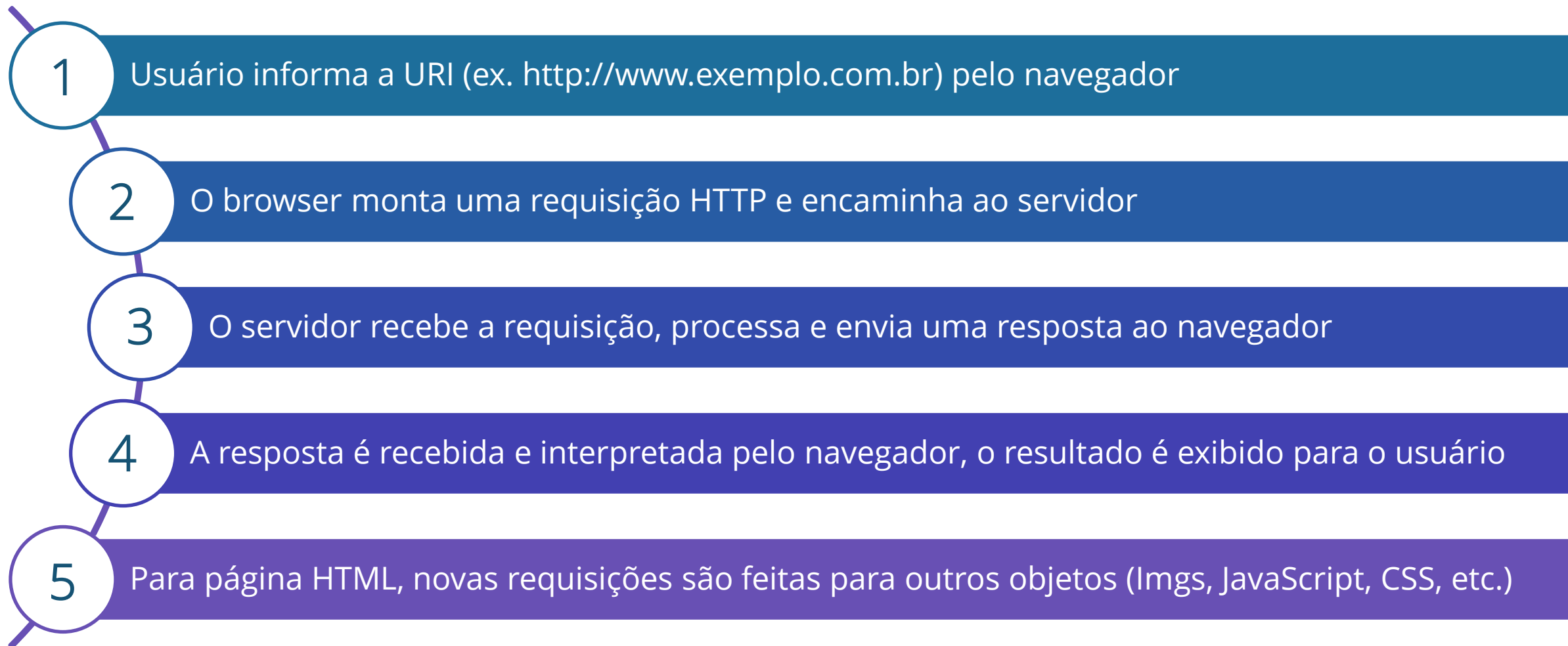
Arquitetura da Web – Protocolo HTTP

O protocolo HTTP é a forma como clientes e servidores se comunicam na rede. As requisições e as respostas obedecem aos padrões estabelecidos pelo protocolo HTTP.

A requisição HTTP é um pacote de dados enviado pela rede pelo Cliente Web para o Servidor Web e identifica o recurso solicitado. A resposta HTTP é formada por pacotes de dados enviados pelo Servidor Web para o Cliente Web com os recursos solicitados.



Protocolo HTTP – Processo de Navegação



Continua ...



PUC Minas

Protocolo HTTP

Requisição e Resposta

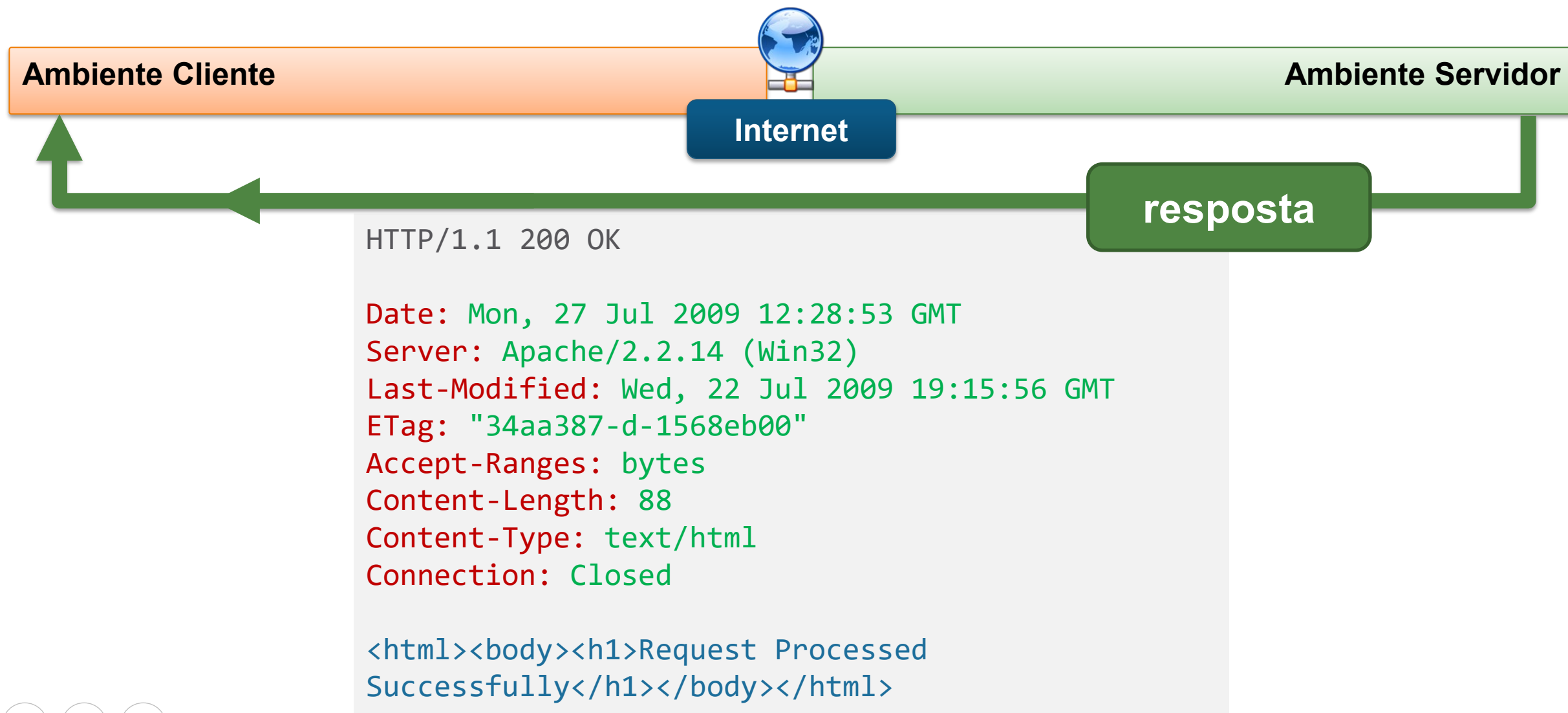
Prof. Rommel Carneiro



Arquitetura da Web – Protocolo HTTP



Arquitetura da Web – Protocolo HTTP



Protocolo HTTP – Requisição

Linha de Requisição

método

recurso

versão HTTP

```
POST /app/processamento HTTP/1.1
```

Linhas de Cabeçalho

campo cabeçalho:

valor

...

campo cabeçalho:

valor

```
User-Agent: Mozilla/4.0 (compatible...)  
Host: www.pucminas.br  
Content-Type: text/xml; charset=utf-8  
Content-Length: 88  
Accept-Language: en-us  
Connection: Keep-Alive
```

Corpo da entidade

```
<?xml version="1.0" encoding="utf-8"?>  
<string>Conteúdo do arquivo</string>
```



12



Protocolo HTTP

Protocolo HTTP – Resposta

Linha de Requisição -----

versão HTTP

code status

msg status

HTTP/1.1 200 OK

Linhas de Cabeçalho -----

campo cabeçalho:

valor

...

campo cabeçalho:

valor

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"
Accept-Ranges: bytes
Content-Length: 88
Content-Type: text/html
Connection: Closed

Corpo da entidade -----

```
<html><body>
<h1>Request Processed Successfully</h1>
</body></html>
```



13



Protocolo HTTP

Protocolo HTTP – Códigos de Retorno

Código	Propósito	Descrição
1xx	Informacional	Requisição recebida, processo em continuidade
2xx	Sucesso	A ação foi recebida, entendida e aceita
3xx	Redirecionamento	Ações adicionais devem ser executadas para completar o pedido
4xx	Erro no cliente	O pedido contém erro de sintaxe ou não pode ser completado
5xx	Erro no servidor	O servidor falhou em completar um pedido aparentemente válido

Exemplos mais comuns

- 200 – Ok
- 403 – Acesso negado
- 404 – Página não encontrada
- 500 – Erro interno do servidor



Continua ...



PUC Minas

Protocolo HTTP

Métodos

Prof. Rommel Carneiro



16



Protocolo HTTP – Métodos

Método	Propósito	Safe (readonly)	Idempotente
GET	Requisitar a representação de um recurso específico	Sim	Sim
POST	Enviar dados a serem processados por um recurso. Usado para incluir recursos ou submeter dados de processamento	Não	Não
HEAD	Similar ao GET, porém retorno deve ser somente do conjunto de cabeçalhos associados ao recurso solicitado	Sim	Sim
PUT	Requisitar a criação ou atualização de um recurso no servidor a partir dos dados no corpo da requisição	Não	Sim
DELETE	Excluir um recurso do servidor	Não	Sim
TRACE	Solicita ao servidor uma cópia (eco) da requisição. Usado para testar se a requisição foi alterada no caminho	Sim	Sim
PATCH	Utilizado para realizar alterações parciais de um recurso	Não	Não
OPTIONS	Usado pelo cliente para entender, ou descobrir, os métodos HTTP e outras opções suportadas por um servidor web	Sim	Sim
CONNECT	Usado quando o cliente estabelece uma conexão HTTPS com um servidor via um proxy	Não	Não

Protocolo HTTP – Métodos – GET

Método GET

- Tem por objetivo requisitar a representação de um recurso ao servidor
- **Por definição, não deve alterar o estado do servidor (*safe*)**
- As requisições podem ser mantidas em cache (favoritos ou bookmarks)
- Envia dados ao servidor via parâmetros na *query string* que ficam visíveis na URL
- Tem restrição quanto ao tamanho e ao formato das informações enviadas ao servidor
 - Formato: limitado a caracteres textuais (ASCII) incluídos na *query string*
 - Tamanho:
 - Apache: 4.000 caracteres
 - MS IIS: 16.384 caracteres
 - Tomcat: padrão 8.192 podendo chegar até 65.536 caracteres

Protocolo HTTP – Métodos – GET

Método GET

- Este é o método mais utilizado em aplicações Web.
- Ao informar uma URL em um navegador, o usuário está disparando uma requisição do tipo GET

Requisição

```
GET /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
Host: www.pucminas.br
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

Response

```
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"
Accept-Ranges: bytes
Content-Length: 88
Content-Type: text/html
Connection: Closed

<html>
  <body> <h1>Hello, World!</h1> </body>
</html>
```

Protocolo HTTP – Métodos – POST

Método POST

- Envia dados ao servidor para serem processados
- **Por definição tem objetivo de alterar o estado do servidor (*not safe*)**
- Pode enviar dados via query string ou via corpo da requisição
 - Os dados enviados pelo corpo **não** ficam visíveis na URL
 - Muito utilizado para envio de dados sensíveis como senhas de acesso
- Não podem ser ‘favoritados’ (bookmarked)
- Não possuem restrição quanto ao tamanho e ao tipo de dados a serem enviados ao servidor

Protocolo HTTP – Métodos – POST

Método POST

- Normalmente é utilizado em conjunto com formulários HTML
- Observe os dados enviados no corpo da Requisição

Requisição

```
POST /cgi-bin/process.cgi HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
Host: www.pucminas.br
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>
```

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"
Accept-Ranges: bytes
Content-Length: 88
Content-Type: text/html
Connection: Closed
```

```
<html><body><h1>Request Processed
Successfully</h1></body></html>
```

Protocolo HTTP – Métodos – HEAD

Método HEAD

Possui estrutura e objetivo similar às requisições de GET, porém o servidor deve enviar apenas o conjunto de cabeçalhos associados ao recurso informado.

Requisição

```
HEAD /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
Host: www.pucminas.br
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

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
```

Protocolo HTTP – Métodos – PUT

Método PUT

Requisita a criação ou atualização de um recurso no servidor a partir dos dados no corpo da requisição. Utilizado no upload de arquivos para servidores Web.

Requisição

```
PUT /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
Host: www.pucminas.br
Accept-Language: en-us
Connection: Keep-Alive
Content-type: text/html
Content-Length: 182
```

```
<html><body>
<h1>Hello, World!</h1>
</body></html>
```

Resposta

```
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>
```

Protocolo HTTP – Métodos – DELETE

Método DELETE

Solicita ao servidor a exclusão de dados ou representações associados ao recurso informado.

Requisição

```
DELETE /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
Host: www.pucminas.br
Accept-Language: en-us
Connection: Keep-Alive
```

Resposta

```
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>
```


Protocolo HTTP – Métodos – TRACE

Método TRACE

Usado para ecoar o conteúdo de uma requisição HTTP ao servidor. Usado para verificar se a requisição é alterada no caminho por agentes intermediários (servidores de cache ou proxy).

Requisição

```
TRACE / HTTP/1.1
Host: www.pucminas.br
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
```

Resposta

```
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.pucminas.br
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
```

Protocolo HTTP – Métodos – OPTIONS

Método OPTIONS

Usado pelo cliente para descobrir os métodos HTTP e outras opções suportados por um servidor web. O cliente pode especificar uma URL para o método de opções ou um asterisco (*) para se referir a todo o servidor.

Requisição

```
OPTIONS * HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
```

Resposta

```
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
```

Protocolo HTTP – Métodos – CONNECT

Método CONNECT

Usado pelo cliente para estabelecer uma conexão com o servidor web que pode ser via protocolo seguro (TLS). É utilizado no caso de requisições a proxies.

Requisição

```
CONNECT www.pucminas.br HTTP/1.1  
User-Agent: Mozilla/4.0 (compatible; MSIE5.01)
```

Resposta

```
HTTP/1.1 200 Connection established  
Date: Mon, 27 Jul 2009 12:28:53 GMT  
Server: Apache/2.2.14 (Win32)
```

Protocolo HTTP – Dados trafegados

- A transmissão via HTTP pode trafegar dados em formato texto ou binário
- Uma requisição deve especificar via cabeçalho **Content-Type**

MIME (Multipurpose Internet Mail Extensions)

- Os valores expressos no cabeçalho **Content-Type** seguem o padrão denominado MIME
- Abaixo são apresentados alguns exemplos de tipos MIME
 - **Image/jpg**: transmissão de imagens (jpe, jpg, jpeg, ...)
 - **text/html**: transmissão de textos em HTML
 - **x-application/java**: transmissão de classes java (.class)

GET vs POST

http://www.w3schools.com/tags/ref_httpmethods.asp

	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL



29



Protocollo HTTP

Continua ...



PUC Minas

Protocolo HTTP

Cabeçalhos

Prof. Rommel Carneiro



31



Protocolo HTTP – Cabeçalhos

Os cabeçalhos utilizados em requisições e respostas do protocolo HTTP carregam informações adicionais sobre a comunicação entre cliente e servidor.

Tipos de Cabeçalhos

- **Request header:** informações sobre a requisição feita ou sobre o cliente Web.
- **Response header:** informações sobre a resposta encaminhada ou sobre o servidor Web.
- **Entity header:** informações sobre o conteúdo da entidade trocada como tamanho e tipo.
- **General header:** Usado tanto em requisições quanto em respostas.

```
POST /app/processamento HTTP/1.1
```

```
User-Agent: Mozilla/4.0 (compatible...)
Host: www.pucminas.br
Content-Type: text/xml; charset=utf-8
Content-Length: 88
Accept-Language: en-us
Connection: Keep-Alive
```

```
<?xml version="1.0" encoding="utf-8"?>
<string>Conteúdo do arquivo</string>
```


Protocolo HTTP – Cabeçalhos de Requisição

Cabeçalho	Utilidade e exemplos	requisição
Accept	<p>Lista os tipos de mídia aceitáveis para a resposta. Indica que a solicitação está limitada a um pequeno conjunto de tipos desejados.</p> <p>Accept: application/json Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8</p>	
Accept-Charset	<p>Lista os conjuntos de caracteres que são aceitáveis para a resposta.</p> <p>Accept-Charset: utf-8, iso-8859-1;q=0.5</p>	
Accept-Encoding	<p>Lista conjuntos de codificações que são aceitáveis para a resposta.</p> <p>Accept-Encoding: gzip, deflate</p>	
Accept-Language	<p>Lista os conjuntos de idiomas naturais aceitáveis e preferidos pelo usuário para a resposta.</p> <p>Accept-Language: pt-BT, en;q=0.9, /*;q=0.8</p>	

Protocolo HTTP – Cabeçalhos de Requisição

Cabeçalho	Utilidade e exemplos	requisição
Authorization	Informa as credenciais de autenticação do User Agent Authorization: Basic SGxsdfRp32hgIKVrw5VzW1	
Host	Indica o host e a porta de onde o recurso está sendo solicitado Host: pucminas.br	
Referer	Informa a URL do recurso de origem, ou visitado antes da requisição atual e que, possivelmente, direcionou o usuário para este recurso referer: https://acesso.gov.br/login?id=acesso.gov.br	
User-Agent	Informa, ao servidor, detalhes sobre o <i>user agent</i> (cliente Web) que está enviando esta requisição user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36	

Protocolo HTTP – Cabeçalhos de Resposta

Cabeçalho	Utilidade e exemplos	resposta
Server	Informa detalhes do software que implementa o servidor Web Server: Apache/2.4.34 OpenSSL/1.0.2k-fips PHP/5.5.38	
Etag	Traz um identificador da versão do recurso que altera toda vez que este for alterado. É utilizado pelo controle de cache. ETag: "353-527867f65e8ad"	
Set-Cookie	Apresenta cookies a serem armazenados pelo cliente e que devem ser enviados ao servidor nas próximas requisições. set-cookie: MLPRICING=1; Domain=magazineluiza.com.br;	

Protocolo HTTP – Cabeçalhos de Resposta

Cabeçalho	Utilidade e exemplos	resposta
Location	Redireciona o cliente Web outra URI Location: https://www.pucminas.br/Paginas/main.aspx	
WWW-Authenticate	indica que o servidor requer a autenticação do usuário para ter acesso ao recurso e de que forma WWW-Authenticate: Basic realm="Site X", charset="UTF-8"	

Protocolo HTTP – Cabeçalhos de Entidade

Cabeçalho	Utilidade e exemplos	entidade
Content-Encoding	Indica uma modificação ao tipo de mídia empregado no conteúdo Content-Encoding: gzip	
Content-Language	Descreve a linguagem na qual o conteúdo foi criado (en, pt, etc) Content-Language: pt-br	
Content-Length	Indica a quantidade em número de bytes na notação decimal Content-Length: 17515	
Content-Location	Local alternativo para o recurso solicitado Content-Location: /index.htm	
Content-Type	Indica o tipo de mídia do conteúdo Content-Type: text/html; charset=utf-8	

Protocolo HTTP – Cabeçalhos de Entidade

Cabeçalho	Utilidade e exemplos	entidade
Expires	Informa a data de expiração do recurso recebido Expires: Sun, 31 Jul 2016 05:00:00 GMT	
Last-Modified	Informa a data e hora de última modificação do recurso no servidor Last-Modified: Tue, 06 Nov 2018 22:45:26 GMT	

Continua ...





PUC Minas

Protocolo HTTP

Novas Versões

Prof. Rommel Carneiro



40



Protocolo HTTP – Histórico de Versões

1991 • HTTP/0.9
1994 • HTTPS
1996 • HTTP/1.0
1999 • HTTP/1.1

2009 • SPDY 1.0

2015 • HTTP/2
2016 • QUIC
2018 • HTTP/3



Protocolo HTTP – Histórico de Versões

- **1991** - O HTTP 0.9 é lançado
- **1994** - O HTTPS foi criado pela Netscape
- **1996** - O HTTP 1.0 foi lançado
 - Conceito de cabeçalhos
 - Códigos de Status
- **1999** - O HTTP 1.1 foi lançado
 - Conexões TCP persistentes
 - Suporte a Virtual Host (Cabeçalho Host)
 - Autenticação Digest
 - Controle de cache
 - Possibilidade de compressão de dados
- **2009** - Google propõe o SPDY
- **2015** - O HTTP 2.0 é lançado
 - Baseado no SPDY
 - Compressão de dados obrigatória
 - Cabeçalhos binários
 - Requisições paralelas
 - Envio apenas de cabeçalhos alterados nas próximas requisições
 - Priorização de requisições
 - Server PUSH – Envio automático de arquivos adicionais.
- **2018** – O HTTP 3 é lançado
 - Protocolo de transporte QUIC baseado em UDP

Protocolo HTTP – HTTP/1.1 vs HTTP/2

HTTP 1.1	X	HTTP 2.0
Protocolo textual.		Protocolo binário.
Protocolo sequencial. Requer mais de uma conexão para simular o paralelismo de requisições		Protocolo assíncrono. Utiliza multiplexação para realizar requisições paralelas em uma única conexão
Não prioriza requisições		Possui priorização de requisições
Apenas o cliente pode iniciar uma requisição.		Possui o mecanismo de server push (servidor infere requisições futuras e realizar o envio antecipado)
Compressão de dados é opcional		Compressão de dados é padrão e obrigatória
Envia todos os dados de cabeçalho em cada mensagem		Comprime cabeçalhos para enviar apenas os dados que sofreram alteração ou são desconhecidos

Obrigado!

