

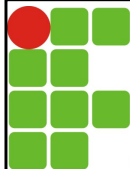
INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE



Redes de Computadores

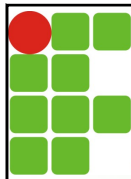
HTTP

Prof. Thiago Dutra <thiago.dutra@ifm.edu.br>



Agenda

- Definição de HTTP
- Hipertexto
- Características do HTTP
- O HTTP e a Web
- Conexões HTTP
- Mensagens HTTP
- Cookies
- Caches Web
- GET Condicional

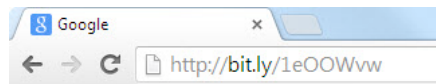


Definição de HTTP

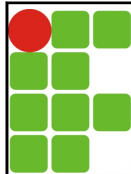
- HTTP = HiperText Transfer Protocol
 - Protocolo de Transferência de Hipertexto
 - Famoso nos endereços Web :



http://



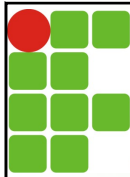
3



Hipertexto

- Texto em formato digital no qual se pode agregar outros conjuntos de informações
 - Blocos de textos, Imagens, Sons, etc.
- Qual a grande vantagem em relação ao texto comum ?
 - Qualquer conteúdo pode ser um **hiperlink** (ligação), ou apenas **link**, para outro conteúdo
 - Texto Dinâmico => Navegação

4

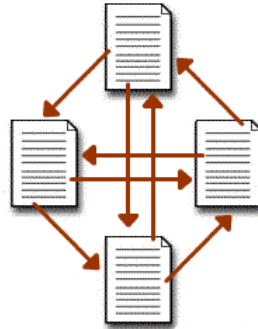


Hipertexto

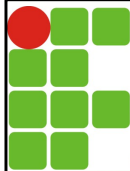
Texto normal



Hipertexto



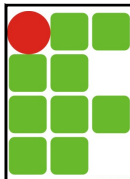
5



Características do HTTP

- Protocolo da camada de aplicação (TCP/IP)
- Possui 3 versões padronizadas :
 - HTTP 1.0 – 1996 [RFC 1945] (<http://tools.ietf.org/html/rfc1945>)
 - HTTP 1.1 – 1999 [RFC 2610] (<https://tools.ietf.org/html/rfc2616>)
 - HTTP 2.0 – 2015 [RFC 7540] (<https://tools.ietf.org/html/rfc7540>)
- HTTP 1.0 e HTTP 1.1 são compatíveis
 - Cliente 1.0 acessa Servidor 1.1 / Cliente 1.1 acessa Servidor 1.0
- HTTP 2.0 : é uma alternativa, mas que não torna obsoleta, a sintaxe das mensagens da 1.1 e mantém toda semântica existente
 - Nem todos clientes e servidores possuem suporte
 - 0,4% da Web adota HTTP 2.0 (W3Techs, 2015)

6

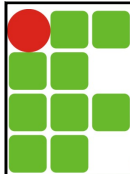


Características do HTTP

- Funciona no modelo **cliente-servidor**
 - Clientes e servidores “conversam” por meio da troca de mensagens HTTP
 - O HTTP define a **estrutura** destas mensagens e o **modo** como elas são enviadas
 - Clientes : Firefox, Chrome, Safari, Internet Explorer, Opera, ...
 - Servidores : Apache, Nginx, IIS, ...
- Utiliza o protocolo de transporte **TCP**
 - Porta **80**
- É um protocolo **stateless** (sem estado)
 - Não guarda informações sobre o cliente
 - As requisições são independentes



7



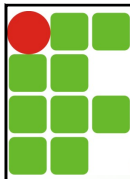
O HTTP e a Web

- “[...] protocolo da camada de aplicação da Web, está no coração da Web [...]” (Kurose, 2010)
- Boa parte do protocolo é responsável por controlar a transferência de **páginas Web** entre clientes e servidores
- **Página Web**
 - Constituída de objetos (arquivos html, jpg, png, mp3, ...)
 - Cada objeto possui uma URL

URL

substantivo masculino
(d1990) *inf* forma padronizada de representação de diferentes documentos, mídia e serviços de rede na internet, capaz de fornecer a cada documento um endereço único.

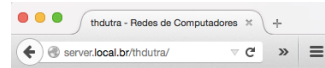
8



O HTTP e a Web

- Uma URL é composta de duas partes

- Nome do hospedeiro (servidor)
- Caminho do objeto



Olá Mundo !

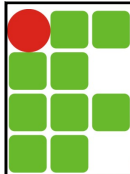
- Exemplo de página Web com 3 objetos

- http://nome_servidor/caminho_objeto
- <http://server.local.br/thdutra/index.html>
- <http://server.local.br/thdutra/mundo.jpg>
- <http://server.local.br/thdutra/google.png>

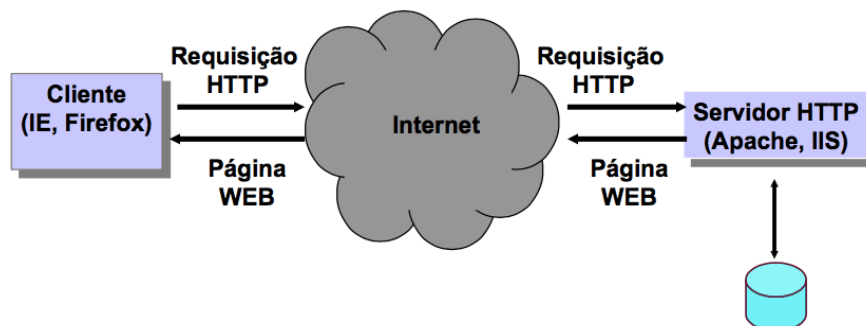


Google

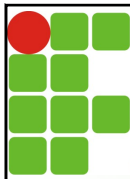
9



O HTTP e a Web



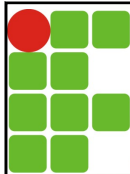
10



Conexões HTTP

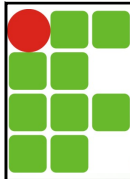
- Uma interação cliente-servidor precisa decidir se para cada par requisição/resposta irá utilizar uma conexão TCP **distinta** ou se todas as requisições e suas respectivas respostas utilizarão utilizarão uma **única** conexão TCP
 - distintas => **conexões não persistentes**
 - única => **conexões persistentes**

11



Conexões HTTP

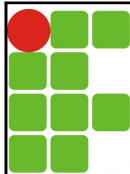
- Transferência de uma página web (**não persistente**)
 - <http://server.local.br/thdutra/>
 1. Cliente inicia conexão TCP com servidor server.local.br na porta 80
 2. Cliente envia uma **mensagem de requisição HTTP** ao servidor solicitando o objeto inicial da página Web (arquivo HTML base)
 3. Servidor recebe a requisição e responde com o objeto solicitado através de uma **mensagem de resposta HTTP**
 4. Servidor solicita encerramento da conexão TCP (a conexão só é realmente fechada quando o cliente confirmar que recebeu a mensagem perfeita)
 5. Cliente recebe a mensagem de resposta HTTP e a conexão TCP é encerrada. A mensagem indica que o objeto encapsulado é um arquivo HTML. O cliente extrai o arquivo HTML da resposta, analisa o seu conteúdo e encontra referências para outros 2 objetos
 6. Os passos 1 até 4 são repetidos para cada um dos objetos referenciados₁₂



Conexões HTTP

- Transferência de uma página web (**persistente**)
 - `http://server.local.br/thdutra/`
 1. Cliente inicia conexão TCP com servidor `server.local.br` na porta 80
 2. Cliente envia uma **mensagem de requisição HTTP** ao servidor solicitando o objeto inicial da página Web (arquivo HTML base)
 3. Servidor recebe a requisição e responde com o objeto solicitado através de uma **mensagem de resposta HTTP**
 4. Cliente recebe a mensagem de resposta HTTP e a **conexão TCP não é encerrada**. A mensagem indica que o objeto encapsulado é um arquivo HTML. O cliente extrai o arquivo HTML da resposta, analisa o seu conteúdo, encontra referências para outros 2 objetos e os requisita imediatamente, em paralelo, pela **mesma conexão TCP**

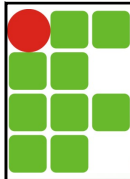
13



Conexões HTTP

- Transferência de uma página web
 - A medida que o cliente (navegador) recebe a página Web (objetos), ele a apresenta para o usuário
 - Dessa forma dois clientes distintos podem interpretar uma mesma página Web e exibi-las de modos ligeiramente diferentes
 - O HTTP **não tem nada haver com o modo como uma página Web é interpretada/exibida por um navegador**
 - O HTTP define apenas como deve ser realizada a comunicação (protocolo) entre clientes e servidores

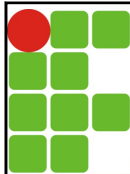
14



Conexões HTTP

- Conexões não persistentes (padrão do HTTP 1.0)
 - Para cada objeto de uma página
 - abrir conexão -> solicitar objeto -> transferir objeto -> fechar conexão
 - Páginas web atuais possuem centenas de objetos
 - Abrir **centenas de conexões sequencialmente é muito lento**; **em paralelo consumiria muitos recursos** do SO de clientes e servidores
- Conexões persistentes (padrão do HTTP 1.1 e 2.0)
 - Permite que várias solicitações e transferências de objetos sejam feitas **utilizando uma mesma conexão**
 - As requisições são feitas em paralelo para diversos objetos distintos
 - Desta forma é possível **atingir uma alta velocidade sem consumo exagerado de recursos** em servidores e clientes

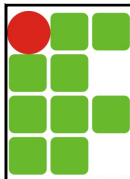
15



Mensagens HTTP

- São escritas em texto comum (código ASCII)
 - Podem ser interpretadas por qualquer ser humano
- Existem dois tipos de mensagens HTTP
 - De requisição (request)
 - De resposta (response)

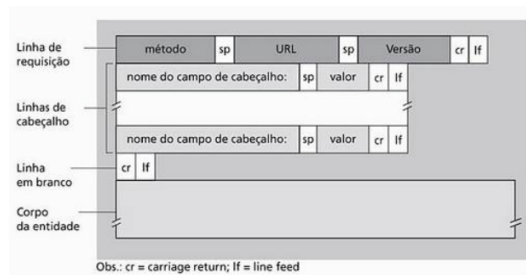
16



Mensagens HTTP

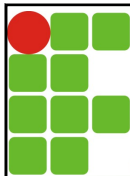
Request

- Linha de requisição
- Linhas de cabeçalho
- Corpo da entidade



Linha de requisição → GET /thdutra/ HTTP/1.1
Linhas de cabeçalho { Host: server.local.br
Connection: close
User-agent: Mozilla/5.0
Accept-language: pt-br

17



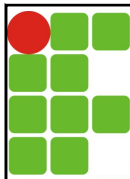
Mensagens HTTP

Analisando um Request

- GET /thdutra/ HTTP/1.1
 - GET -> método (tipo de requisição)
 - /thdutra/ -> caminho do objeto solicitado
 - HTTP/1.1 -> versão do protocolo
- Host: server.local.br
 - nome do servidor onde o objeto deve ser buscado
- Connection: close
 - fechar a conexão (não persistente)
- User-agent: Mozilla/5.0
 - tipo do cliente (navegador)
- Accept-language: pt-br
 - linguagem preferencial do objeto requisitado

```
GET /thdutra/ HTTP/1.1
Host: server.local.br
Connection: close
User-agent: Mozilla/5.0
Accept-language: pt-br
```

18



Mensagens HTTP

■ Analisando um Request

■ Linha de requisição -> Sempre única, obrigatória !

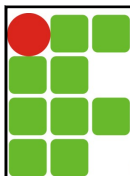
■ Métodos :

Método	Descrição
GET	Solicita a leitura de uma página da Web
HEAD	Solicita a leitura de um cabeçalho de página da Web
PUT	Solicita o armazenamento de uma página da Web
POST	Acrescenta a um recurso (por exemplo, uma página da Web)
DELETE	Remove a página da Web
TRACE	Ecoa a solicitação recebida
CONNECT	Reservado para uso futuro
OPTIONS	Consulta certas opções

■ Linhas de cabeçalho -> Opcionais

- Indicam opções relacionadas a requisição
- Existem cerca de 50 opções disponíveis no HTTP 1.1
 - Seção 14 da RFC2616

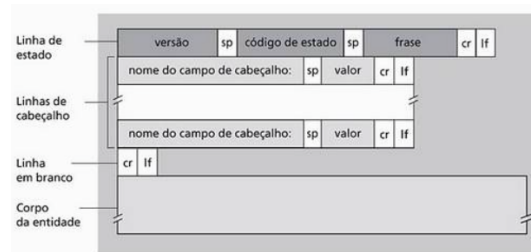
19



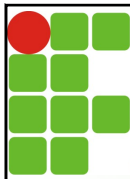
Mensagens HTTP

■ Response

- Linha de estado
- Linhas de cabeçalho
- Corpo da entidade



20



Mensagens HTTP

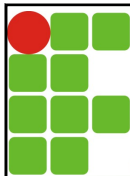
■ Analisando um Response

Diagram illustrating the structure of an HTTP response with labels and arrows pointing to the corresponding parts of the code block:

- Linha de estado** (Status line): `HTTP/1.1 200 OK`
- Linhas de cabeçalho** (Header lines):
 - `Date: Fri, 07 Aug 2015 04:53:54 GMT`
 - `Server: Apache/2.2.22 (Debian)`
 - `Last-Modified: Fri, 07 Aug 2015 04:53:36 GMT`
 - `ETag: "43fef-e1-51cb16b2c8017"`
 - `Accept-Ranges: bytes`
 - `Content-Length: 225`
 - `Vary: Accept-Encoding`
 - `Connection: close`
 - `Content-Type: text/html`
- Corpo da entidade** (Entity body):

```
<html>
  <head>
    <title>thdutra - Redes de Computadores</title>
  </head>
  <body>
    <center>
      <h1>01&aacute; Mundo !</h1>
       <br/>
      
    </center>
  </body>
</html>
```

21



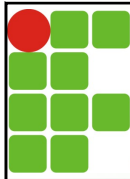
Mensagens HTTP

■ Analisando um Response

- **HTTP/1.1 200 OK**
 - **HTTP/1.1** -> versão do protocolo
 - **200** -> código de estado
 - **OK** -> mensagem do estado
- **Date: Fri, 07 Aug 2015 04:53:54 GMT**
 - data e hora da resposta
- **Server: Apache/2.2.22 (Debian)**
 - tipo do servidor
- **Content-Length: 225**
 - tamanho do arquivo na resposta
- **Content-Type: text/html**
 - tipo do arquivo na resposta
- **Last-Modified: Fri, 07 Aug 2015 04:53:36 GMT**
 - Data e hora de modificação do arquivo no servidor

```
HTTP/1.1 200 OK
Date: Fri, 07 Aug 2015 04:53:54 GMT
Server: Apache/2.2.22 (Debian)
Last-Modified: Fri, 07 Aug 2015 04:53:36 GMT
ETag: "43fef-e1-51cb16b2c8017"
Accept-Ranges: bytes
Content-Length: 225
Vary: Accept-Encoding
Connection: close
Content-Type: text/html
```

22



Mensagens HTTP

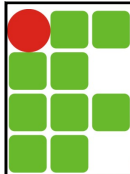
- Analisando um Response
- Linha de estado (*status*) -> **Sempre única, obrigatória !**

- Códigos :

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

- Linhas de cabeçalho
 - Opcionais, porém quase sempre presentes
 - Contém informações diversas sobre
 - O servidor
 - O conteúdo dos dados existentes na resposta
 - ...

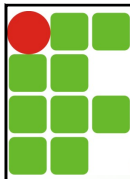
23



Cookies

- Problema : o HTTP é stateless, como então o servidor pode obter informações sobre os usuários para poder "interagir" com eles ?
- Solução : Cookies
 - Mecanismo que permite os servidores HTTP identificar (e monitorar) os seus usuários
 - Componentes
 - Linhas de cabeçalho utilizadas nas mensagens de requisição e resposta HTTP
 - Arquivos armazenados na máquina do usuário e gerenciados pelo navegador
 - Um banco de dados auxiliar mantido no servidor

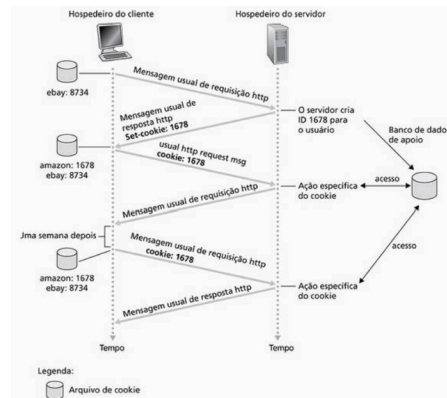
24



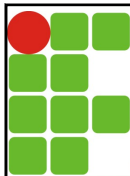
Cookies

■ Funcionamento :

1. Usuário acessa um site que use cookies pela primeira vez
2. Servidor responde incluindo em sua primeira resposta o cabeçalho Set-cookie: seguido de alguma identificação única.
 - Set-cookie: 1678453
3. O navegador armazena esta informação em um arquivo texto
4. Todas as vezes que o usuário voltar a acessar este site, o Navegador irá incluir em suas requisições a linha:
 - Cookie: 1678453



25

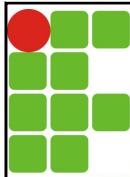


Cookies

- Os cookies podem trazer :
 - Autorizações, Cartões de compra, Recomendações, Estado de sessão do usuário, ...
- Privacidade
 - Cookies permitem que os sites saibam muito sobre você
 - Páginas visitadas, horários, nome, e-mail, ...
 - Suas informações pode ser repassadas/vendidas
 - Páginas personalizadas
 - Propagandas direcionadas
 - ...



26

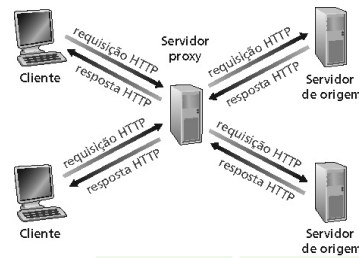


Caches Web

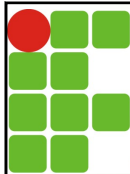
- É um **interceptador (proxy)** entre o cliente e o servidor web

- **Funcionamento :**

- O **usuário configura o navegador** para o acesso ser feito através de um proxy ou ele pode **existir "escondido"** na infraestrutura de instituição (**proxy transparente**)
- Navegador **envia todas as requisições HTTP para o proxy**
- Proxy **verifica se possui o objeto em cache**
 - Se possuir ele retornar o objeto
 - Senão, o proxy solicita o objeto ao servidor original e então o envia para o cliente armazenando uma cópia no seu cache



27

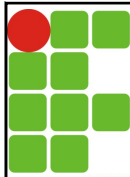


Caches Web

- Em geral são utilizados por provedores e instituições

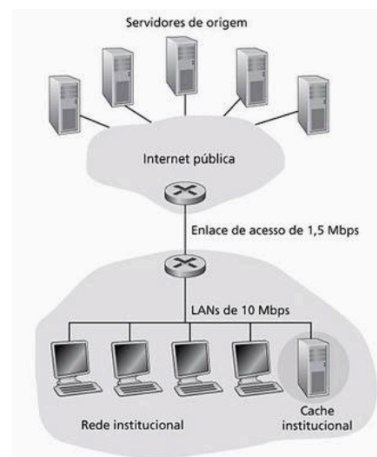
- **Reduz o tempo de resposta** para a requisição do cliente
- **Reduz o tráfego** no link de internet da instituição
- Possibilita um série de **controles de acesso** e **filtragens**
- Permite a geração de **estatísticas de uso** da Internet pelos usuários

28

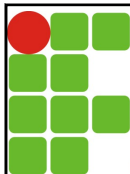


Caches Web

- Um cache bem configurado vai conseguir ter grande parte das páginas requisitadas armazenadas localmente
- O uso de cache web possibilita um uso otimizado do link de acesso externo (que normalmente tem um custo bastante elevado)

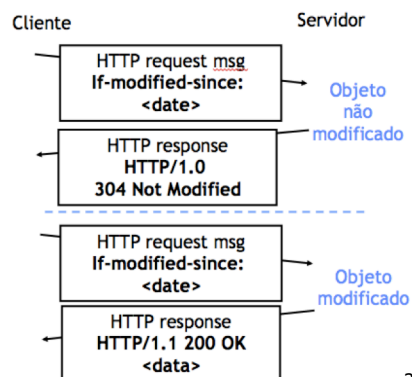


29

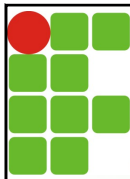


GET Condicional

- E se o cache do proxy estiver desatualizado ?
- Finalidade : não enviar objeto para o cliente se ele já tem a versão atualizada
- Cliente : especifica data da versão armazenada na requisição HTTP
 - If-modified-since
- Servidor : resposta não contém dados se a cópia esta atualizada
 - HTTP/1.0 304 Not Modified



30



GET Condicional

■ 1ª Requisição

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
```

```
HTTP/1.1 200 OK
Date: Sat, 7 Jul 2007 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Wed, 4 Jul 2007 09:23:24
Content-Type: image/gif

(data data data data data ...)
```

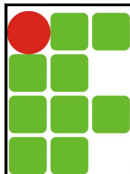
■ 2ª Requisição

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-modified-since: Wed, 4 Jul 2007 09:23:24
```

```
HTTP/1.1 304 Not Modified
Date: Sat, 14 Jul 2007 15:39:29
Server: Apache/1.3.0 (Unix)

(corpo de mensagem vazio)
```

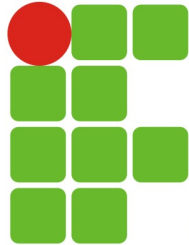
31



Referências

- KUROSE, J. F. e ROSS, K. - Redes de Computadores e a Internet – 5a Ed., Pearson, 2010.
- TANENBAUM, A. S. – Redes de Computadores – 5a Ed., Pearson, 2011.
- W³Techs – Web Technologies Surveys
 - <http://w3techs.com>
- Google
 - <https://www.google.com.br/search?q=define+url>

32



**INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE**



Redes de Computadores

HTTP

Prof. Thiago Dutra <thiago.dutra@ifm.edu.br>