

INE5646 – Programação para Web

Unidade I

Introdução à Programação para Web

Prof. Frank Siqueira – Turma A

Prof. Leandro Komosinski – Turma B

Conteúdo

- Introdução
- Modelo cliente-servidor
- Arquitetura da Web
- Servidores Web: instalação, configuração, administração e serviços
- Páginas estáticas e dinâmicas
- Protocolos HTTP e Websockets
- Segurança: HTTPS e controle de acesso

Introdução

“A World Wide Web é uma iniciativa para recuperação de informação hipermídia em longa distância que tem como objetivo prover acesso a um amplo universo de documentos”

Tradução da definição de *World Wide Web* contida no primeiro Web site criado (<http://info.cern.ch/>)

Introdução

- Definições
 - Hipertexto: documento de texto associado a outros documentos através de ligações (*hyperlinks*)
 - Hipermídia: hipertexto enriquecido com outras mídias (imagens, sons, vídeos, etc.)
 - Termos definidos pelo sociólogo e filósofo americano Ted Nelson em 1965
 - Usados inicialmente em repositórios centralizados
 - Hipermídia + Internet = *World Wide Web* !!!

Introdução

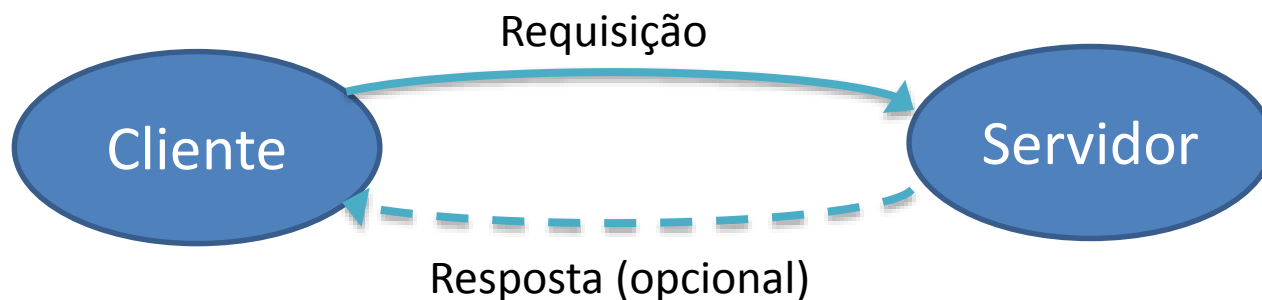
- Histórico da *World Wide Web*
 - Idealizada pelo físico inglês Tim Berners-Lee, à época pesquisador do CERN em Genebra, Suíça
 - Arquitetura proposta em 1989 e aperfeiçoada no ano seguinte por Berners-Lee e Robert Cailliau
 - O software do primeiro servidor Web e um navegador baseado em texto foram disponibilizados em 1991
 - NCSA Mosaic: 1º navegador gráfico; lançado em 1993
 - O *World Wide Web Consortium* (W3C), dirigido por Berners-Lee, foi criado em 1994 com o intuito de definir padrões e conduzir a evolução da Web

Modelo Cliente-Servidor

- Utilização
 - Modelo mais utilizado para interação entre computadores interligados em rede
 - Adotado pela Web e por diversos outros serviços da Internet, como e-mail, FTP, Telnet, NFS, etc.
 - Empregado na construção de diversos tipos de sistemas de informação, como sistemas de banco de dados, CRM, ERP, etc.

Modelo Cliente-Servidor

- Entidades do modelo:
 - Cliente: requisita a execução de um serviço do qual necessita
 - Servidor: executa o serviço e responde ao cliente



Modelo Cliente-Servidor

- Lógica de execução
 - Apenas o servidor precisa conhecer a lógica de execução do serviço e/ou ter acesso aos dados necessários para sua execução
 - O cliente se limita a solicitar a execução do serviço, fornecendo parâmetros de execução quando necessário, e a exibir, armazenar ou utilizar o resultado enviado pelo servidor, quando for o caso

Modelo Cliente-Servidor

- Dinâmica do Modelo Cliente-Servidor
 - O cliente solicita a execução de um serviço a um servidor
 - O servidor executa o serviço e responde ao cliente
 - O cliente recebe o resultado da execução do serviço e o utiliza como desejar
- Exemplo: *download* de um arquivo
 - Cliente especifica o arquivo que deseja baixar
 - Servidor lê arquivo e envia conteúdo ao cliente
 - Cliente recebe e salva ou abre o arquivo

Modelo Cliente-Servidor

- Considerações sobre o modelo
 - Vários clientes podem ter acesso ao servidor
 - Vários servidores podem executar o mesmo serviço
 - É preciso adotar um protocolo de comunicação entre cliente e servidor, que defina o formato e a semântica das mensagens trocadas pela rede

Arquitetura da Web



Arquitetura da Web

- Navegadores Web (*Web Browsers*)
 - Acessam servidores Web e solicitam conteúdos
 - Exibem para o usuário o conteúdo acessado
 - Permitem que o usuário navegue para outros conteúdos da Web acionando *hyperlinks*
- Servidores Web
 - Disponibilizam conteúdo na Web
 - Conteúdo pode ser armazenado estaticamente em arquivos ou gerado dinamicamente

Arquitetura da Web

- Características Gerais da Implementação
 - A Web emprega o modelo Cliente-Servidor
 - HTTP (*HyperText Transfer Protocol*) é o protocolo de comunicação padrão, mas vários outros protocolos são suportados (ftp, news, mailto, ...)
 - Permite vários formatos de conteúdos, sendo o principal o HTML (*HyperText Markup Language*)
 - Navegadores podem ser implementados em diversas plataformas (PC, celular, tablet, TV, etc.)
 - Servidores foram desenvolvidos para vários SOs

Arquitetura da Web

- *Web Caching*
 - Em geral, navegadores armazenam em uma *cache* local cópias de conteúdos obtidos recentemente
 - O navegador, antes de contatar o servidor, deve verificar se o conteúdo requisitado está disponível no *cache* local, e nesse caso deve perguntar ao servidor se o conteúdo foi alterado
 - Caso o conteúdo não tenha sido modificado, a cópia mantida em *cache* pode ser usada
 - Usuário pode configurar o tamanho do *cache*

Arquitetura da Web

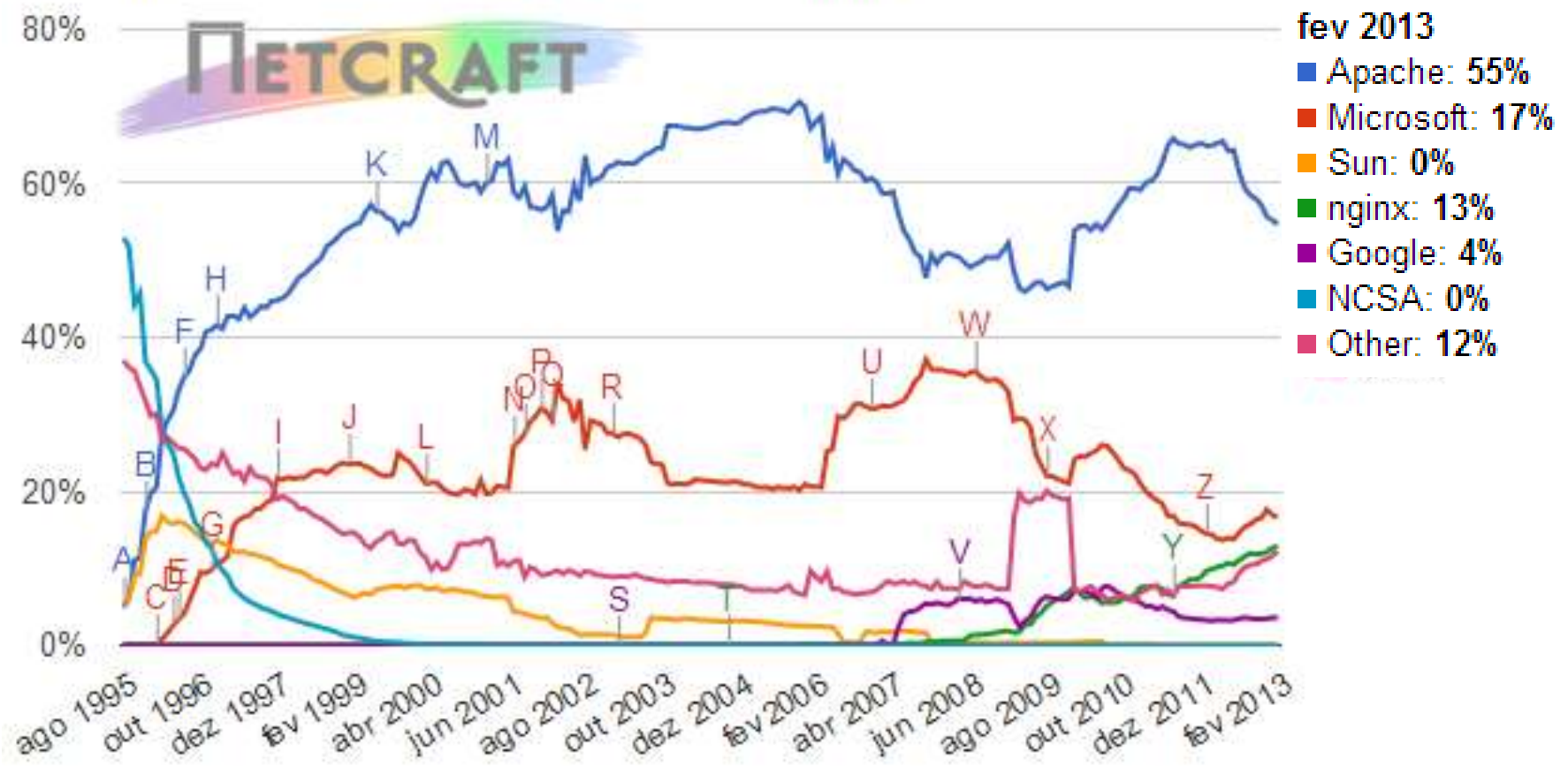
- *Web Proxy*
 - Clientes com ou sem acesso à Internet podem acessar a Web através de um servidor de *proxy*, que intermediará conexões com servidores Web
 - Servidores de *proxy* podem manter um *cache* para armazenar conteúdos acessados por seus usuários, com o intuito de reduzir tráfego na rede
 - Caso um usuário acesse um conteúdo mantido no *cache* do servidor de *proxy* e que não foi alterado no servidor, a cópia em *cache* poderá ser utilizada

Servidores Web

- Implementações
 - O Apache HTTP Server, disponível para diversas plataformas, é a implementação mais popular atualmente, sendo predominante em sistemas da família UNIX
 - O Microsoft IIS (*Internet Information Services*) é a implementação mais utilizada em servidores com sistema operacional Windows

Servidores Web

Market Share for Top Servers Across All Domains



Servidores Web

- Instalação
 - Geralmente o servidor é instalado como *daemon*/serviço do sistema operacional e permanece em execução indefinidamente
 - Alguns servidores (ex.: Apache) são distribuídos com seu código-fonte, e precisam ser compilados para a plataforma alvo
 - Em geral os servidores permitem definir durante a instalação/compilação quais funcionalidades serão suportadas, adicionando/removendo módulos

Servidores Web

- Instalação (cont.)
 - Durante o processo de instalação determina-se o local de instalação do servidor (*Server Root*)
 - Pode ser necessário fornecer informações de configuração iniciais:
 - Nome do servidor (*Server Name*)
 - Diretório com o conteúdo servido (*Document Root*)
 - Arquivos de configuração, erro, *log*, etc.
 - Em geral, essas configurações podem ser alteradas posteriormente

Servidor Web

- Configuração
 - Apache: arquivo *httpd.conf*
 - IIS: utilitário *IIS Manager* do Painel de Controle
- Principais opções de configuração:
 - *Document Root*: diretório que mantém os conteúdos disponibilizados pelo servidor
 - *Server Name*: nome do servidor
 - *Server Alias*: nome(s) alternativo(s) do servidor
 - *Virtual Host*: permite hospedar vários sites em uma mesma máquina

Servidor Web

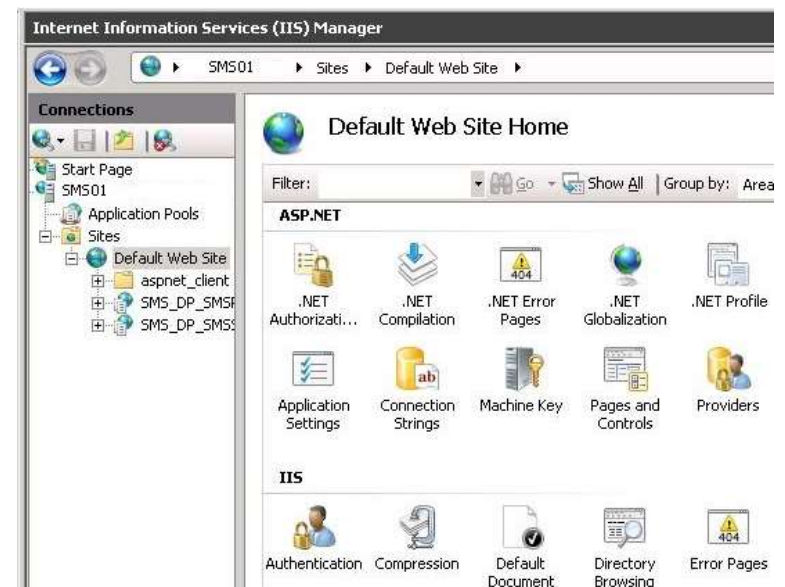
- Configuração – Erros e *Logging*:
 - O administrador pode definir em quais arquivos serão registrados os acessos ao servidor (arquivo de *log*) e os eventuais erros que venham a ocorrer
 - Mensagens urgentes podem ser enviadas ao administrador por e-mail ou por outros meios (SMS, IM, VoIP, etc.) suportados pelo servidor
 - As páginas ou mensagens enviadas ao cliente em caso de erro também podem ser definidas na configuração do servidor

Servidor Web

- Administração
 - Comandos
 - Ex.: `httpd -k start|restart|stop|...` (válido p/ Apache)
 - Ferramentas fornecidas pelo sistema operacional
 - Ex.: Apache pode ser instalado como serviço do Windows e controlado usando o comando NET ou o utilitário *Serviços* do Painel de Controle

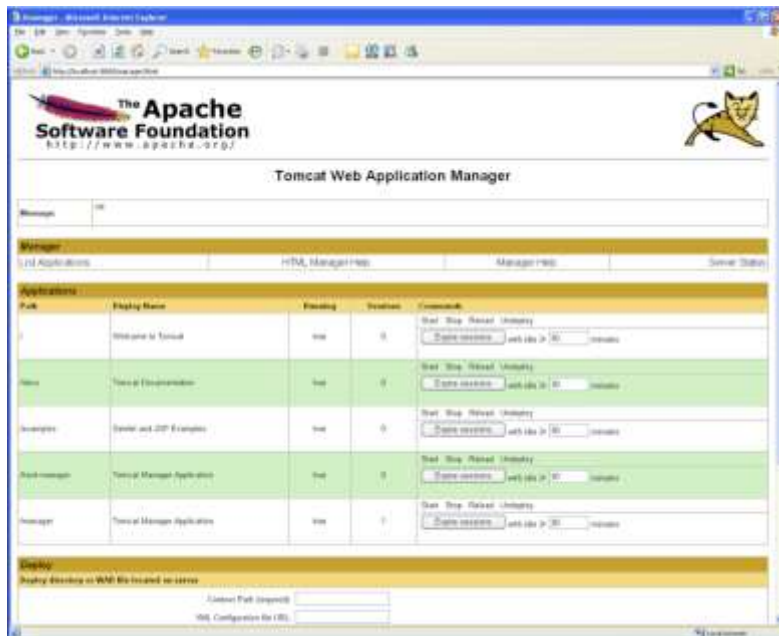
Servidor Web

- Administração (cont.)
 - Programa utilitário fornecido pelo servidor
 - Ex.: *Apache Service Monitor*, *IIS Manager* (no Windows)



Servidor Web

- Administração (cont.)
 - Interface Web de administração do servidor
 - Ex.: Tomcat Manager; Glassfish Admin Console



Servidor Web

- Serviços
 - Ao escolher um entre os vários servidores Web existentes, é importante considerar os serviços (funcionalidades) implementados pelo servidor
 - Ex.: protocolos, linguagens, controle de acesso, etc.
 - Em geral, quanto mais serviços fornecidos pelo servidor Web, maior o consumo de recursos de hardware e pior o tempo de resposta do servidor
 - Serviços podem ser fornecidos por módulos, que são adicionados ao servidor quando necessários

Servidor Web

- Serviços (cont.)
 - Por exemplo, o Apache HTTP Server possui módulos para:
 - Protocolos SSL, FastCGI, ...
 - Linguagens Perl, PHP, Python, ...
 - Controle de acesso via LDAP, Kerberos, ...
 - Compressão
 - ... e vários outros serviços/funcionalidades.

Servidor Web

- Serviços (cont.)
 - Os chamados Servidores de Aplicação hospedam programas/sistemas que geralmente possuem uma interface Web – ou seja, podem ser acessados através de um cliente/navegador Web
 - Em geral, servidores de aplicação consomem mais memória e ciclos de CPU que os servidores Web

Servidor Web

- Serviços (cont.):
 - Servidores de Aplicação JavaEE:
 - Apache Tomcat e Jetty são servidores JavaEE “leves”, que fornecem suporte apenas para Java Servlets e JSP
 - JBoss, Glassfish, JOnAS (gratuitos), IBM WebSphere e Oracle WebLogic (pagos) são servidores completos, com suporte a EJB, JMS e às demais APIs do JavaEE

Páginas Estáticas e Dinâmicas

- Páginas Estáticas x Dinâmicas
 - Páginas estáticas são mantidas no sistema de arquivos do servidor Web e disponibilizadas para acesso dos clientes, que ao acessar o endereço correspondente visualizarão o mesmo conteúdo
 - Páginas dinâmicas, quando acessadas, são parcial ou totalmente geradas através da execução do código de programas, podendo apresentar conteúdo diferenciado para cada cliente

Páginas Estáticas e Dinâmicas

- Páginas Estáticas
 - Principal formato: HTML (*HyperText Markup Lang.*)
 - Outros formatos: XHTML (baseado em XML), ...
- Páginas Dinâmicas
 - Código em uma linguagem de programação para Web é executado ao acessar o endereço da página
 - Instruções podem ser adicionadas a páginas (X)HTML ou invocadas a partir delas, gerando o conteúdo dinâmico
 - Exemplos de linguagens para Web: JSP, PHP, ASP(.NET), Python, Perl, Ruby (on Rails), etc.

Páginas Estáticas e Dinâmicas

- Introdução ao HTML
 - HTML é uma linguagem de marcação, que define o aspecto de um documento que será exibido no navegador do usuário
 - O código HTML pode ser criado de várias maneiras:
 - Escrito pelo desenvolvedor usando um editor de texto
 - Gerado por um editor gráfico (ex.: Dreamweaver)
 - Convertido a partir de outro formato (ex.: DOC→HTML)

Páginas Estáticas e Dinâmicas

- Histórico do HTML
 - Versão 1.0: definida informalmente p/ Berners-Lee
 - Versão 2.0: especificada pela IETF em Nov. 1995
 - Versão 3.2: definida pelo W3C em Jan. 1997
 - Versão 4.0 (atual): publicada em Dez. 1997
 - Depreciou vários elementos de formatação, que não serão mais suportados na versão 5, obrigando que a formatação seja feita por CSS (*Cascading Style Sheets*)
 - Versão 5.0: rascunho publicado em Jan. 2008; versão final esperada para 2014

Páginas Estáticas e Dinâmicas

- Estrutura de um Documento HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Título da Página</title>
    ...
  </head>
  <body bgcolor="lightblue">
    ... Conteúdo da Página...
  </body>
</html>
```

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML

- Parágrafo:

- ```
<p>texto</p>
```

- Quebra de linha:

- ```
<br />
```

- Títulos:

- ```
<h1>Título 1</h1>
```

- ```
<h2>Título 2</h2>
```

- ```
<h3>Título 3</h3>
```

- ```
...
```

Saída:

texto

Título 1

Título 2

Título 3

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML

- Negrito:

- `Negrito`

- Itálico:

- `<i>Itálico</i>`

- Subscrito:

- `X_i`

- Sobrescrito:

- `X²`

Saída:

Negrito

Itálico

X_i

X^2

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML

- Citação:

```
<blockquote>O computador nasceu para  
resolver problemas que não existiam  
antes.</blockquote>
```

O computador nasceu para resolver problemas
que não existiam antes.

- Código (texto monoespaçado):

```
<code>int a = 10;</code>
```

```
int a = 10;
```

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML

- Texto pré-formatado:

```


```

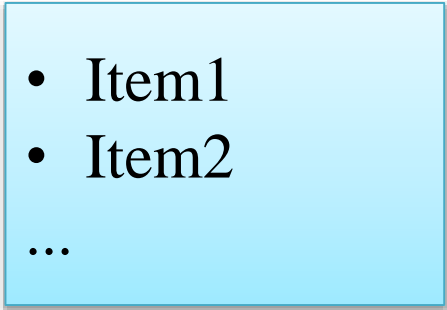
```

      _||_|_||_|_
    [ o o o ]
  _____
 \ o o o o o o /
~^~^~^~^~^~^~^~^</pre>
```

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML
 - Lista Não-ordenada (com *bullets*):

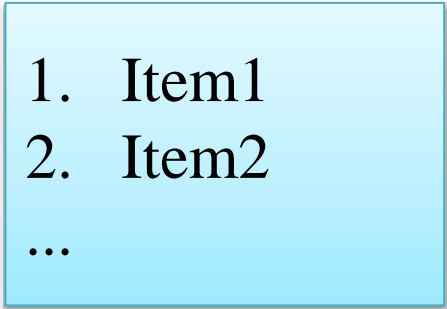
```
<ul>  
  <li>Item1</li>  
  <li>Item2</li>  
  ...  
</ul>
```



- Item1
- Item2
- ...

- Lista Ordenada (numerada):

```
<ol>  
  <li>Item1</li>  
  <li>Item2</li>  
  ...  
</ol>
```



1. Item1
2. Item2
- ...

Páginas Estáticas e Dinâmicas

- Principais marcadores do HTML

- Imagens

```

```



- Links

```
<a href="destino.html">Clique aqui</a>
```

[Clique aqui](#)

Páginas Estáticas e Dinâmicas

- Tabelas

```
<table>
  <caption>Médias Finais</caption>
  <tr>
    <th>Aluno</th>
    <th>Média</th>
  </tr>
  <tr>
    <td>Fulano</td>
    <td>9.5</td>
  </tr>
  ...
</table>
```

Aluno	Média
Fulano	9.5
...	...

Páginas Estáticas e Dinâmicas

- Caracteres Especiais:

Código	Caractere
&	&
<	<
>	>
"	"
'	'
¼	¼
½	½
°	°
 	(espaço)
©	©
€	€

Páginas Estáticas e Dinâmicas

- Problemas do HTML
 - Dois *browsers* podem exibir uma mesma página de forma diferente
 - Alguns *browsers* criam *tags*/recursos proprietários, que não são aceitos por outros
 - *Browsers* podem não seguir fielmente a especificação, ou ser compatíveis com diferentes versões do HTML
 - Erros contidos no código da página podem ser tratados por *browsers* de formas diferentes
 - XHTML, por ser baseado em XML, tem regras mais rígidas de construção, evitando alguns problemas

Protocolo HTTP

- HTTP (*HyperText Transfer Protocol*)
 - Histórico
 - HTTP/0.9: versão implementada por Berners-Lee
 - HTTP/1.0: especificada pela IETF em Maio de 1996
 - HTTP/1.1: versão atual; definida pela IETF no RFC 2616, publicado em Janeiro de 1997.
 - Características Gerais
 - Permite a transferência de conteúdos localizados em um endereço da Web, chamado tecnicamente de URL
 - Permite o envio de mídias identificadas por um tipo MIME: (hiper)texto, imagens, áudio, vídeo, etc.

Protocolo HTTP

- URL (*Uniform Resource Locator*)
 - Identificador que determina a localização de um recurso (documento)
 - Especificada pela IETF no RFC 1738, em Dez. 1995
 - Alterado pelos seguintes RFCs, de Jan. 2005:
 - RFC 3986: define a URI (*Uniform Resource Identifier*):
 $URI = URL \cup URN$ (*Uniform Resource Name*)
 - RFC 3987: define a IRI (*Internationalized Resource Identifier*), que permite o uso de caracteres de diversos alfabetos, acentuação, etc.

Protocolo HTTP

- URL

- Caracteres permitidos:

A-Z a-z 0-9 - . _ ~

- Caracteres reservados:

: / ? # [] @ ! \$ & ' () * + , ; =

Obs.: Caracteres reservados podem ser codificados no formato *%nn*, onde *nn* é seu código ASCII em hexadecimal

Protocolo HTTP

- Formato Geral de uma URL:

esquema://servidor:porta/caminho?parâmetros#fragmento

- *esquema*: em geral, indica o protocolo usado para acessar o recurso (ex.: http, ftp, mailto, file, ...)
- *servidor*: nome de domínio ou endereço IP do servidor, caso exigido pelo protocolo
- *porta*: porta de comunicação usada pelo servidor (se for diferente da porta padrão do protocolo)
- *caminho*: caminho até o recurso no servidor (ex: pasta/subpasta/arquivo.html)
- *parâmetros*: dados passados ao servidor (ex: a=1&b=2&...)
- *fragmento*: parte interna do recurso (ex.: #secao1)

Protocolo HTTP

- MIME (*Multipurpose Internet Mail Extensions*)
 - Criado originalmente para permitir o envio de diversos tipos de mídia através de e-mail
 - Usado na Web pelo servidor HTTP para indicar o tipo de mídia enviado como resposta a uma requisição, de modo que o *browser* possa renderizá-la usando o mecanismo adequado
 - Formato: *tipo/subtipo*

Protocolo HTTP

- Principais Tipos MIME

Extensão	Tipo MIME	Descrição
.txt	text/plain	Texto não-formatado
.htm, .html	text/html	Hipertexto
.gif	image/gif	Imagem em formato GIF
.jpg, .jpe, .jpeg	image/jpeg	Imagem no formato JPEG
.png	image/png	Imagem no formato PNG
.avi	video/x-msvideo	Vídeo em formato AVI
.qt .mov	video/quicktime	Vídeo em formato Quicktime
.mpeg .mpg .mpe	video/mpeg	Vídeo em fomato MPEG
.wav	audio/x-wav	Áudio em formato WAV
.mp3	audio/mpeg3	Áudio em formato MPEG3
.exe .bin	application/octet-stream	Arquivo binário (executável)
.gz, .gzip	application/x-gzip	Arquivo compactado em formato GZIP
.zip	application/x-zip-compressed	Arquivo compactado em formato ZIP
.doc	application/msword	Documento do Microsoft Word
.pdf	application/pdf	Documento no formato Adobe PDF

Protocolo HTTP

- Estabelecimento de conexões HTTP
 - Emprega os protocolos TCP/IP da Internet, usando por padrão a porta 80
 - Nas versões 0.9 e 1.0, uma nova conexão era estabelecida para cada documento trocado entre um cliente e um servidor
 - A versão 1.1 permite que a conexão entre cliente e servidor seja mantida aberta para troca de vários documentos

Protocolo HTTP

- Dinâmica do Protocolo HTTP
 - Fases do protocolo
 1. Requisição: enviada pelo cliente ao servidor
 2. Resposta: remetida pelo servidor para o cliente
 - Formato das mensagens
 - Cabeçalho: informações de controle do protocolo
 - Corpo: dados trocados entre o cliente e o servidor (possivelmente vazio)

Protocolo HTTP

- Formato das mensagens de requisição

Método	Documento	Versão
Cabeçalho		
<linha em branco>		
Corpo da mensagem		

- Métodos de requisição
 - GET: solicita o documento especificado
 - HEAD: obtém o cabeçalho da resposta (*caching*)
 - POST: envia dados para o servidor processar
 - PUT: envia documento para ser armazenado
 - DELETE: remove o documento especificado

Protocolo HTTP

- Exemplo de mensagem de requisição

```
GET /index.html HTTP/1.1
Host: www.servidorweb.com
Connection: keep-alive
Cache-Control: no-cache
Accept: text/html,application/xhtml+xml,
application/xml;q=0.9,*/*;q=0.8
Pragma: no-cache
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/537.17 (KHTML, like Gecko)
Chrome/24.0.1312.57 Safari/537.17
Accept-Encoding: gzip,deflate,sdch
Accept-Language: pt-BR,pt;q=0.8,en-US,en;q=0.6
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
```

Protocolo HTTP

- Formato das mensagens de resposta

Versão	Código	Status
Cabeçalho		
<linha em branco>		
Corpo da mensagem		

- Códigos de status

- 1xx: informacional
- 2xx: sucesso
- 3xx: redirecionamento
- 4xx: erro do cliente
- 5xx: erro do servidor

Protocolo HTTP

- Exemplo de mensagem de resposta

```
HTTP/1.1 200 OK
Date: Wed, 20 Feb 2013 19:23:40 GMT
Server: Apache/2.4.3 (Win64)
Last-Modified: Mon, 30 Aug 2010 19:25:52 GMT
Accept-Ranges: bytes
Content-Length: 34394
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

```
<!DOCTYPE html>
```

```
<html>
```

```
...
```

```
</html>
```

Protocolo HTTP

- Exemplo de mensagem de erro

HTTP/1.1 404 Not Found

Date: Wed, 20 Feb 2013 19:23:40 GMT

Server: Apache/2.4.3 (Win64)

Content-Length: 201

Keep-Alive: timeout=5, max=100

Connection: Keep-Alive

Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML>

<html>

<head><title>404 Not Found</title></head>

<body><h1>Not Found</h1>

<p>The requested URL was not found.</p>

</body></html>

Protocolo HTTP

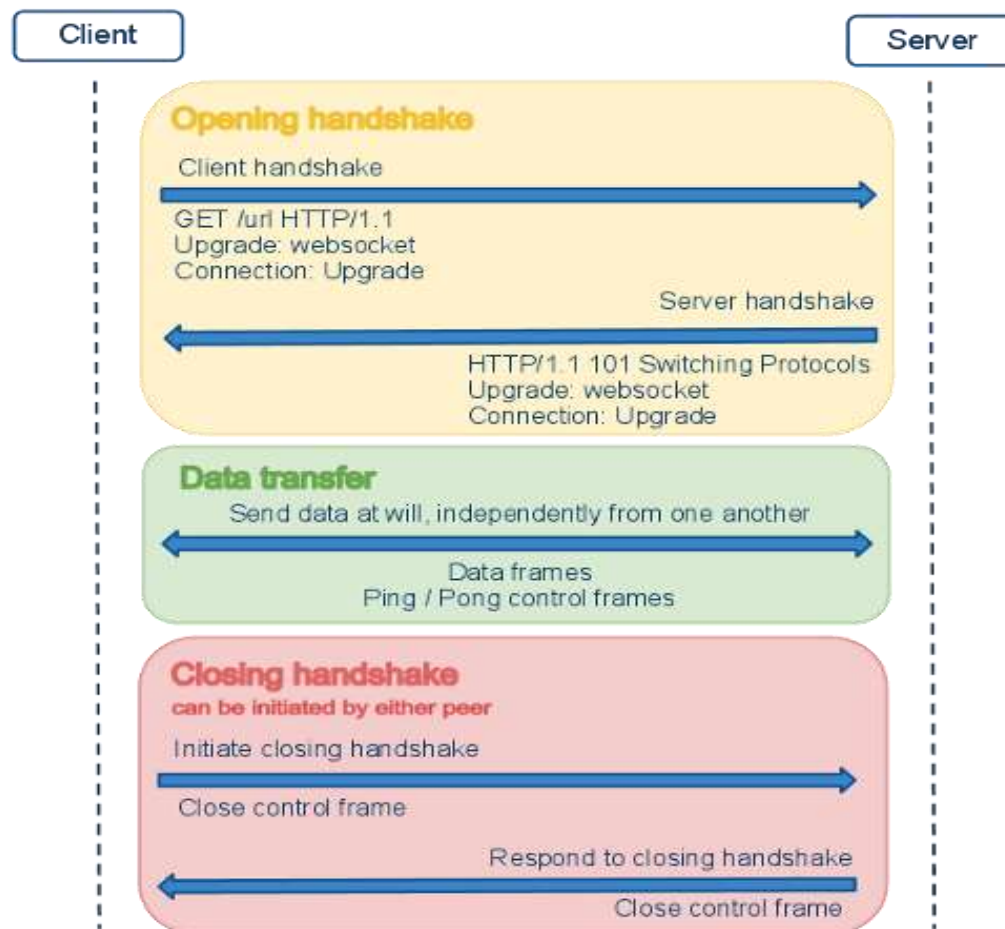
- O protocolo HTTP não garante a segurança do usuário em suas interações na Web
 - Dados trafegados entre *browser* e servidor podem ser interceptados por outros usuários da rede
 - Usuário pode ser direcionado a um servidor que se passa por outro para roubar dados/senhas
- O protocolo não impede ataques de negação de serviço (DoS)
 - Atacante pode estourar o limite de conexões simultâneas suportadas pelo servidor Web

WebSockets

- Protocolo WebSockets
 - Especificado pela IETF em 2011 no RFC 6455
 - API WebSockets integra a especificação do HTML5
 - Permite comunicação bidirecional entre navegador e servidor sobre uma conexão TCP
 - Atravessa *firewalls* facilmente: usa porta 80
 - Esquema URL: *ws://* ou *wss://* (seguro)
 - Suportado nas versões mais recentes dos principais navegadores Web

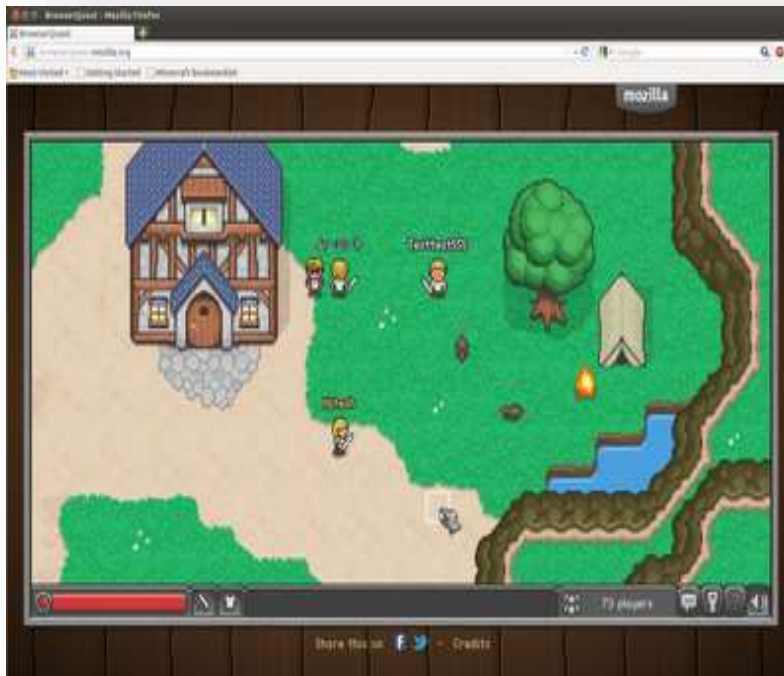
WebSockets

- Funcionamento



WebSockets

- Principais aplicações:
 - Jogos interativos *multiplayer* online
 - Transmissão de conteúdo em tempo real

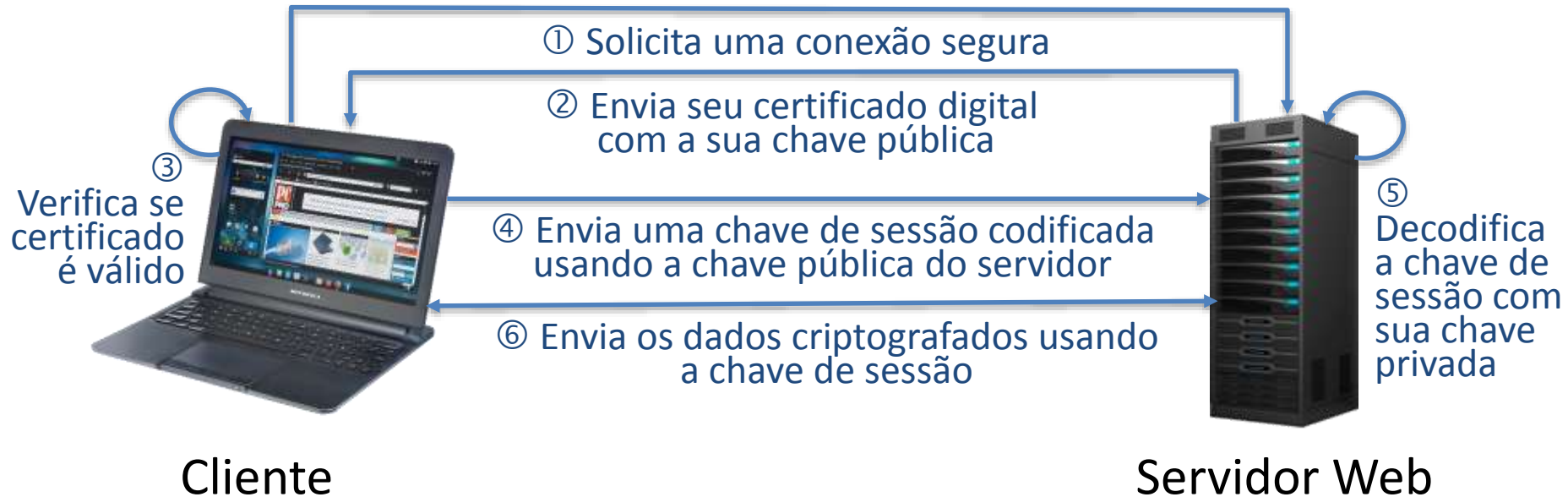


Segurança

- HTTP Seguro (HTTPS)
 - HTTP sobre SSL/TLS
 - Permite o estabelecimento de um canal de comunicação seguro entre *browser* e servidor
 - Dados trocados pela rede são criptografados
 - Servidor deve possuir um certificado digital que garanta a sua autenticidade
 - Esquema URL: *https://*
 - *Browser* mostra chave/cadeado e exibe aviso caso o servidor não possua um certificado digital válido

Segurança

- Funcionamento do HTTPS
 - Usa criptografia simétrica (chave de sessão) e assimétrica (chave pública + chave privada) para codificar os dados trocados entre o cliente e o servidor Web



Segurança

- Controle de acesso
 - Servidores podem limitar o acesso a todo ou a parte de um site com base no endereço do cliente
 - Exemplos – Configuração do Apache (httpd.conf):
 - Permitir acesso ao site somente de um domínio

```
Order deny, allow
Deny from all
Allow from ufsc.br
```
 - Bloquear o acesso a uma área a partir de um domínio

```
<Directory /messageboard>
Deny from spammers.net
</Directory>
```

Segurança

- Autenticação

- Pode ser exigida a autenticação do usuário para que este acesse um determinado site ou diretório
- Exemplos – Configuração do Apache (httpd.conf):

- Criar um arquivo de senhas e cadastrar um usuário

```
htpasswd -c /usr/local/apache/passwd/passwords fulano
```

- Cadastrar um novo usuário em um arquivo já existente

```
htpasswd /usr/local/apache/passwd/passwords beltrano
```

- Configurar o servidor para exigir autenticação

```
<Directory /private>  
AuthType Basic  
AuthName "Private area"  
AuthUserFile /usr/local/apache/passwd/passwords  
Require valid-user  
</Directory>
```