

Desenvolvimento Web

Agenda

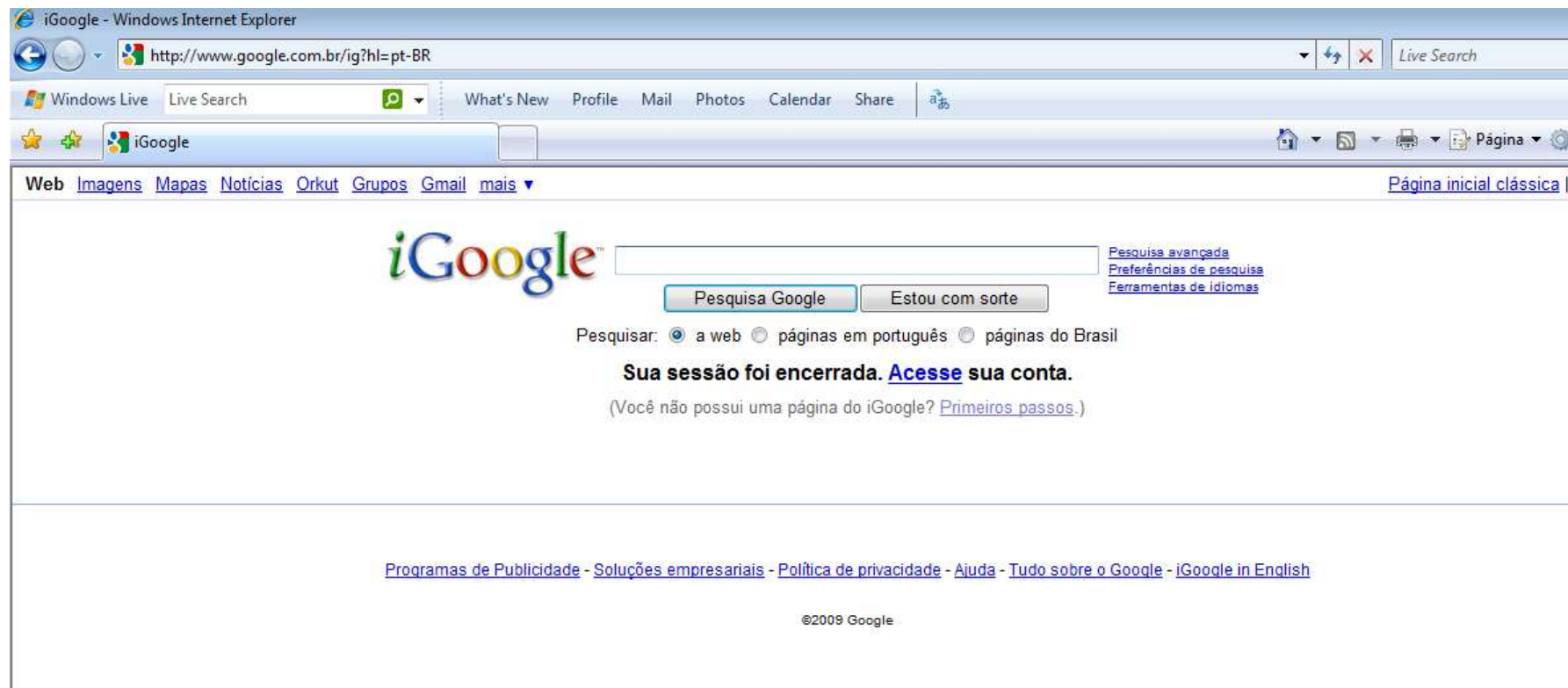
- Como funciona ?
 - Arquitetura Cliente-Servidor
 - Protocolo HTTP
- Como desenvolver ?
 - O lado do cliente
 - HTML / XML / XHTML
 - Javascript
 - CSS
 - O lado do servidor: Java, PHP, Ruby, Python, etc...

Agenda

- Web 2.0
 - O conceito
 - AJAX
 - Interface e RIA
- CMS e frameworks
- Conclusão

Servidor de HTTP

- **Servidor web**: Servidor responsável pelo armazenamento de páginas de um determinado site, requisitados pelos clientes através de browsers.



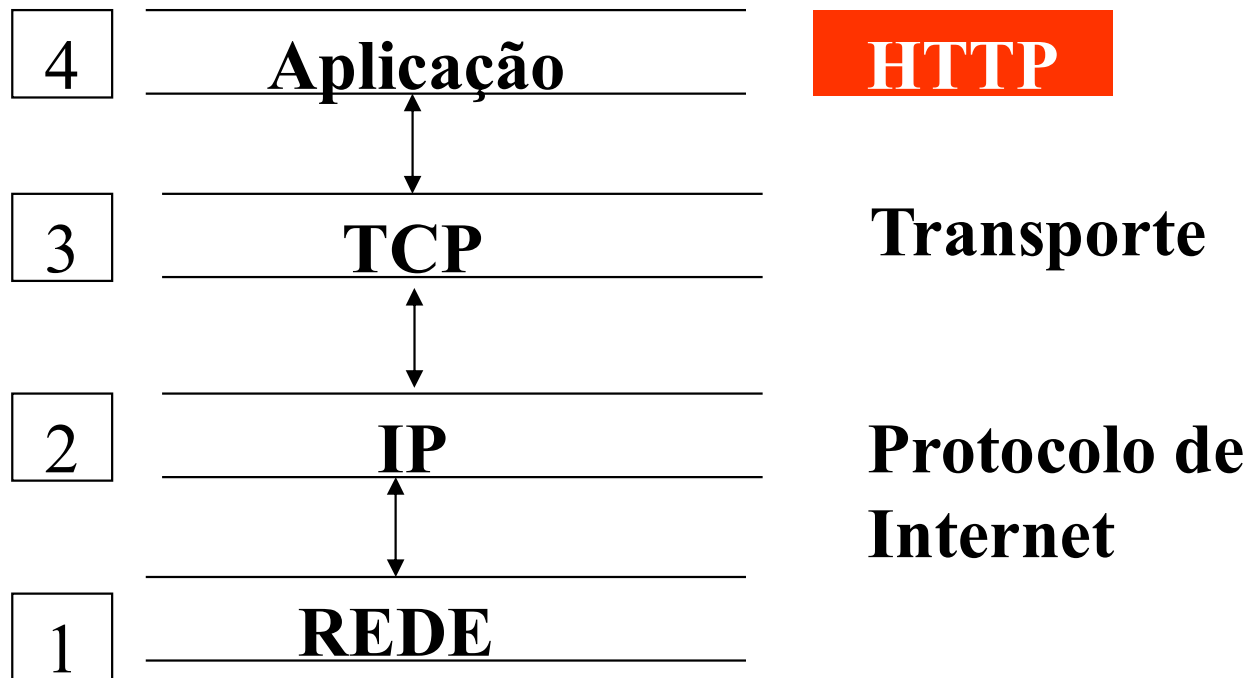
O protocolo HTTP

- (HyperText Transfer protocol) é o protocolo de rede utilizado para entrega virtualmente de todos os arquivos e outros dados sejam eles arquivos HTML, arquivos de imagens, resultados de consulta a banco de dados, arquivo de texto, ou qualquer outro tipo de recurso.

Sistema de comunicação

Arquitetura

Camadas



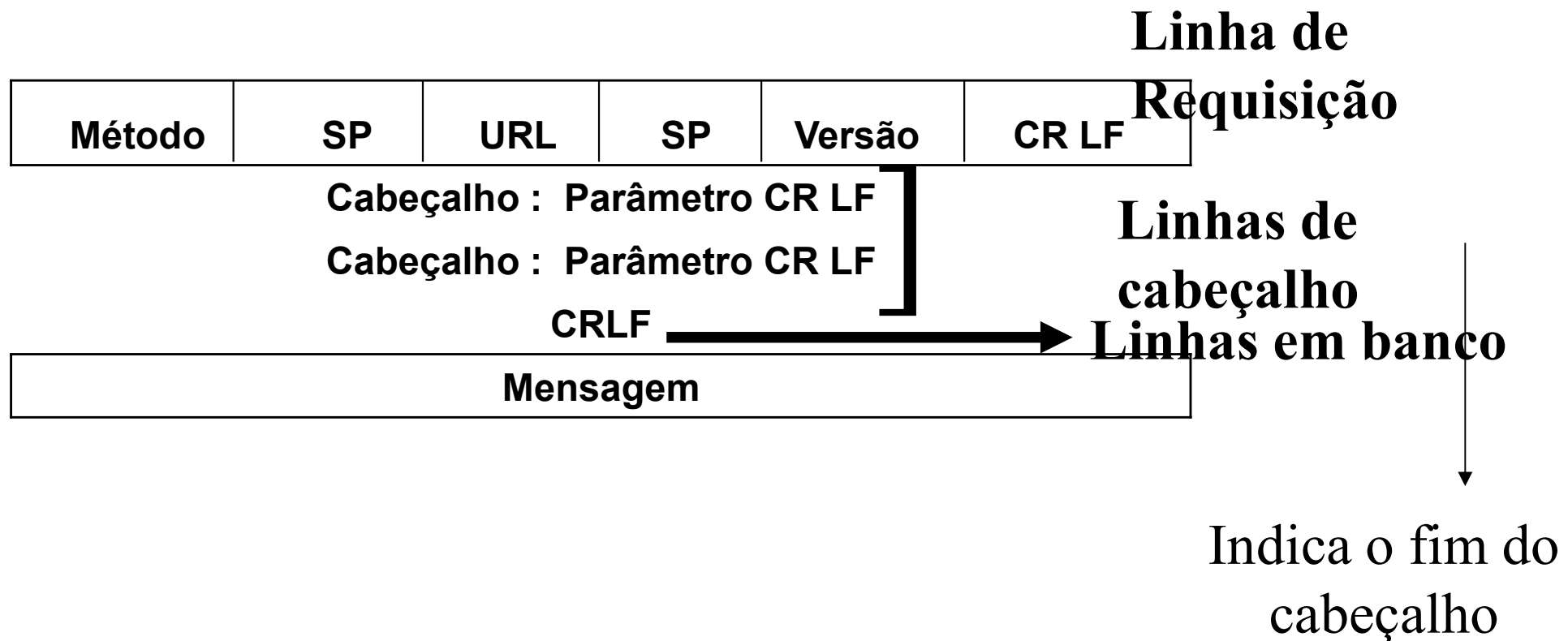
Transações HTTP

Uma transação consiste em:

- Estabelecimento da conexão;
- Requisição (pelo cliente);
- Resposta (pelo servidor);
- Fechamento da conexão.

Linha inicial de requisição

Método de envio de dados



URL – (User Resource locator)

Linha de resposta (linha de status)

Linha de status

Versão	SP	Status	SP	Motivo	CR LF
--------	----	---------------	----	--------	-------

Cabeçalho : Parâmetro CR LF

Cabeçalho : Parâmetro CR LF

CRLF

Linhas de Cabeçalhos

Linha em branco

Corpo da Entidade (Mensagem)

(HTML)

O código de status

- 1xx – Indica uma mensagem de informação apenas.
- 2xx – indica operação de algum tipo bem sucedida.
- 3xx – Redireciona o cliente para uma outra URL.
- 4xx – Indica um erro por parte do cliente
- 5xx – Indica um erro por parte do servidor

Protocolo TCP/IP

- Toda máquina tem um endereço IP
- Toda informação da web é identificada por uma URI.
 - URI: Universal Resource Identifier
 - URL: Uniform Resource Locator
 - URN: Uniform Resource Name

Exemplo de URI, URL e URN

- URI: conceito, identificador uniforme de recurso

<protocolo>://<localizacao>/<caminho>/<recurso>

- URL: localizador uniforme de recursos“

<http://www.dc.ufscar.br/>

- URN: nome uniforme de recurso
- <xsd:schema
xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:example"

esquema ou protocolo://domínio:porta/caminho/recurso?query_string#fragmento

- <http://example.org/absolute/URI/with/absolute/path/to/resource.txt>
- <ftp://example.org/resource.txt>
- file:///home/example/example.org/resource.txt
- urn:[issn](#):1535-3613

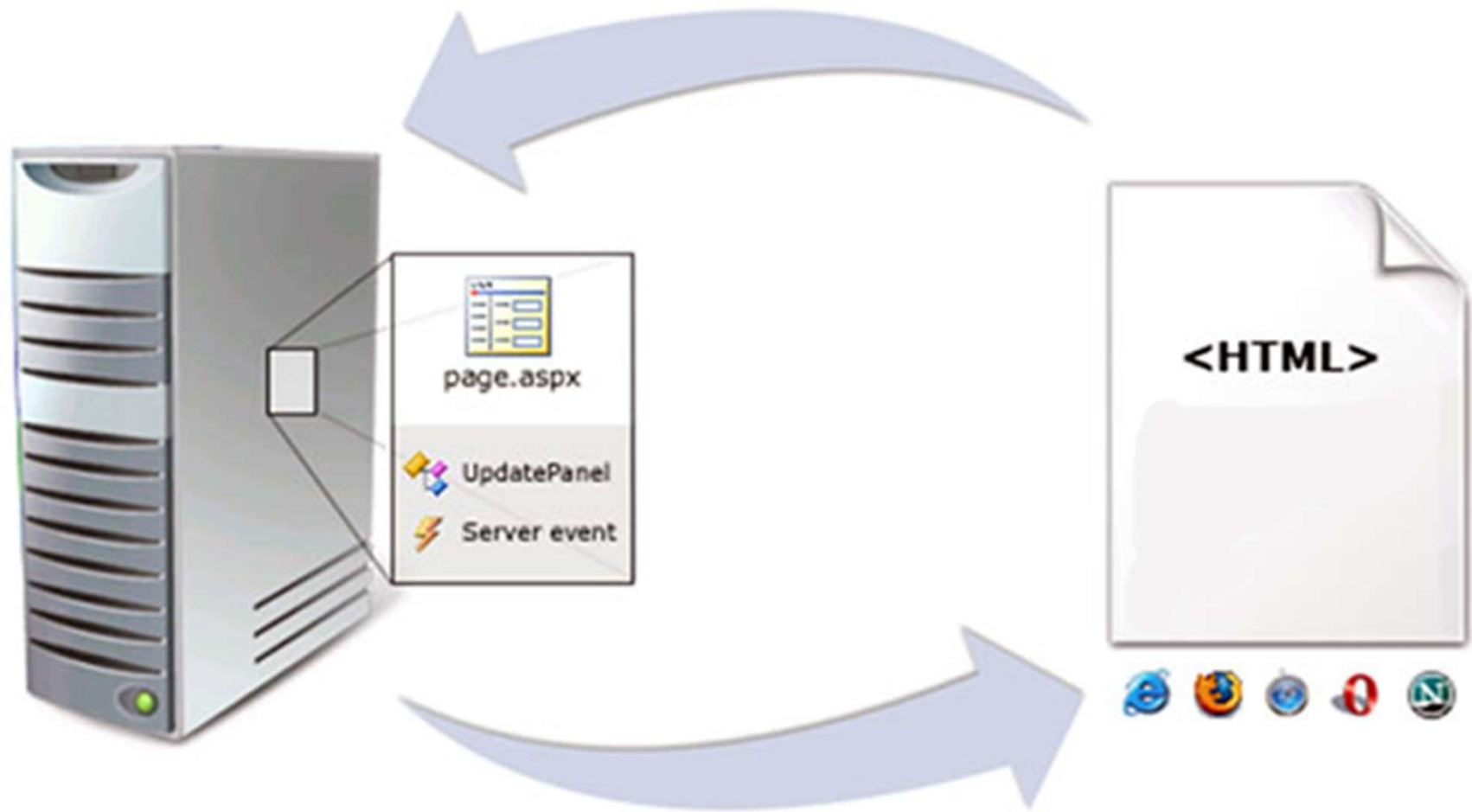
HTTP 1.1 – Na teoria

- HTTP GET: Requisição de informações especificadas pela URI
- HTTP POST: Requisita que o servidor aceite as informações enviadas para a URI
- HTTP PUT: Requisita que o servidor armazene as informações enviadas para a URI
- HTTP DELETE: Requisita que o servidor apague o recurso identificado pela URI

HTTP 1.1 – Na Prática

- As linguagens server-side aumentaram o nível de abstração da comunicação, restando só dois métodos HTTP realmente usados
- HTTP GET: Aparece na URI do browser
 - Ex: Busca do Google
- HTTP POST: Não aparece na URI do browser
 - Ex: A visualização do seu extrato no banco

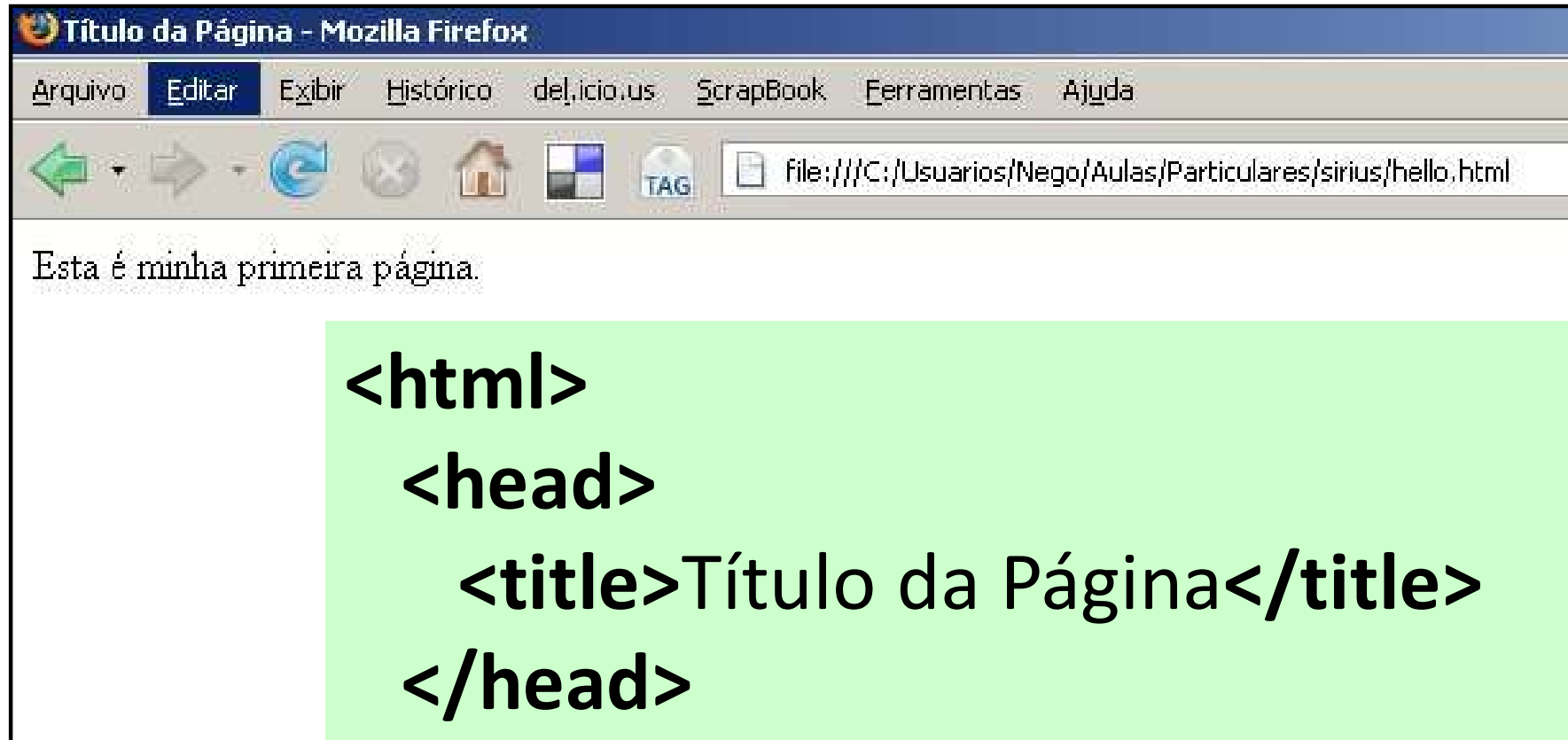
Cliente - Servidor



Como desenvolver?

- A resposta do servidor deve ser em formato que o cliente (browser) consegue entender
- O processamento é feito na linguagem do servidor (programação): Java, PHP, C#, etc...
- O output é feito em linguagem de marcação (HTML/XHTML), script (Javascript) ou de estilos (CSS)

HTML



```
<html>
  <head>
    <title>Título da Página</title>
  </head>
  <body>
    Esta é minha primeira página.
  </body>
</html>
```

HTML

Tags Básicas

Tag

[<html>](#)

[<body>](#)

[<h1> to <h6>](#)

[<p>](#)

[
](#)

[<hr>](#)

[<!-- -->](#)

Descrição

Define um documento HTML

Define o corpo de um documento

Define cabeçalhos 1 a 6

Define um parágrafo

Insere uma quebra de linha

Define uma linha horizontal

Define um comentário

HTML

Formulários

- HTML possibilita a criação de formulários online utilizando tags
- A tag <form> é a mais comum e permite a criação de um formulário de entrada de dados

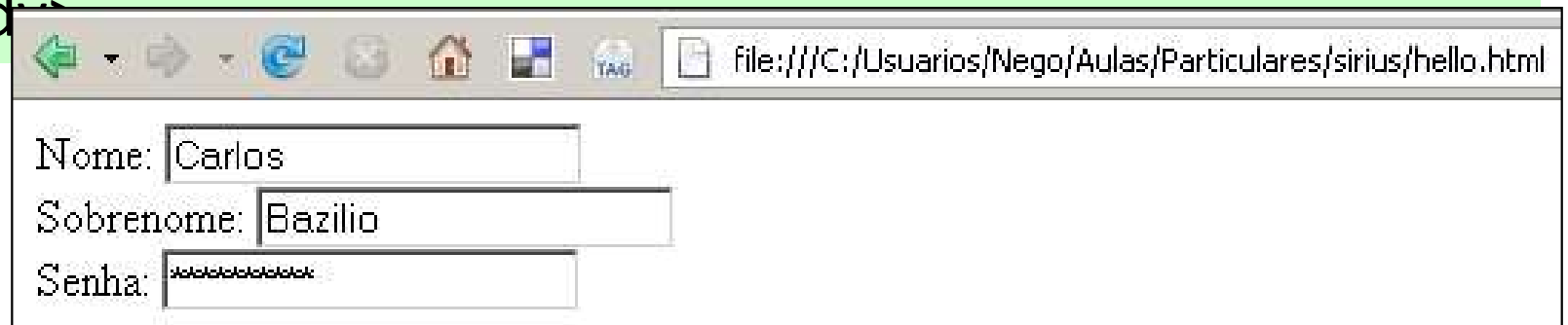
```
<body><form>
```

```
Nome: <input type="text" name="firstname"><br />
```

```
Sobrenome: <input type="text" name="lastname"><br />
```

```
Senha: <input type="password" name="senha">
```

```
</form></body>
```



A screenshot of a web browser window showing a rendered HTML form. The browser's address bar displays the file path: file:///C:/Usuarios/Nego/Aulas/Particulares/sirius/hello.html. The form contains three input fields: 'Nome' with the value 'Carlos', 'Sobrenome' with the value 'Bazilio', and 'Senha' which is masked with asterisks. The browser's toolbar includes navigation buttons (back, forward, home, etc.) and a search icon.

CSS

- *Cascading Style Sheets*
- Estilos definem como elementos html devem ser apresentados
- Permite a separação entre definição de conteúdo e formatação em HTML
- Style sheets externos são definidos através de arquivos CSS
- Atualmente é o padrão para inserção de estilo na construção de páginas html
- <http://www.csszengarden.com/>

CSS

- Formato geral:

seletor { propriedade: valor }

- Uma definição de estilo em CSS será composta por uma sequência de definições como esta acima
- Exemplos:
 - `body { color: black }`
 - `p { font-family: "Verdana"; text-align: center; color: red }`
 - `h1,h2,h3,h4,h5,h6 { color: green }`
 - `p {margin-left: 20px}`

CSS

- Classes em CSS permitem que um mesmo elemento seja exibido de diferentes formas

- Formato usando classes:

seletor.classe { propriedade: valor }

- Exemplos:

- p.direita {text-align: right} // Alinha à direita
- p.centro {text-align: center} // Centraliza
- .esquerda {text-align: left} // Aplicado a qualquer elemento html que contenha classe *esquerda*

```
<p class="direita">Este parágrafo será alinhado à direita</p>
```

```
<p class="centro">Este parágrafo será centralizado</p>
```

```
<p class="esquerda">Este será à esquerda</p>
```

Aplicações

- Com CSS podemos formatar:
 - Background
 - Textos
 - Fontes
 - Margens
 - Bordas
 - Listas
 - Tabelas
- http://www.w3schools.com/css/css_examples.asp

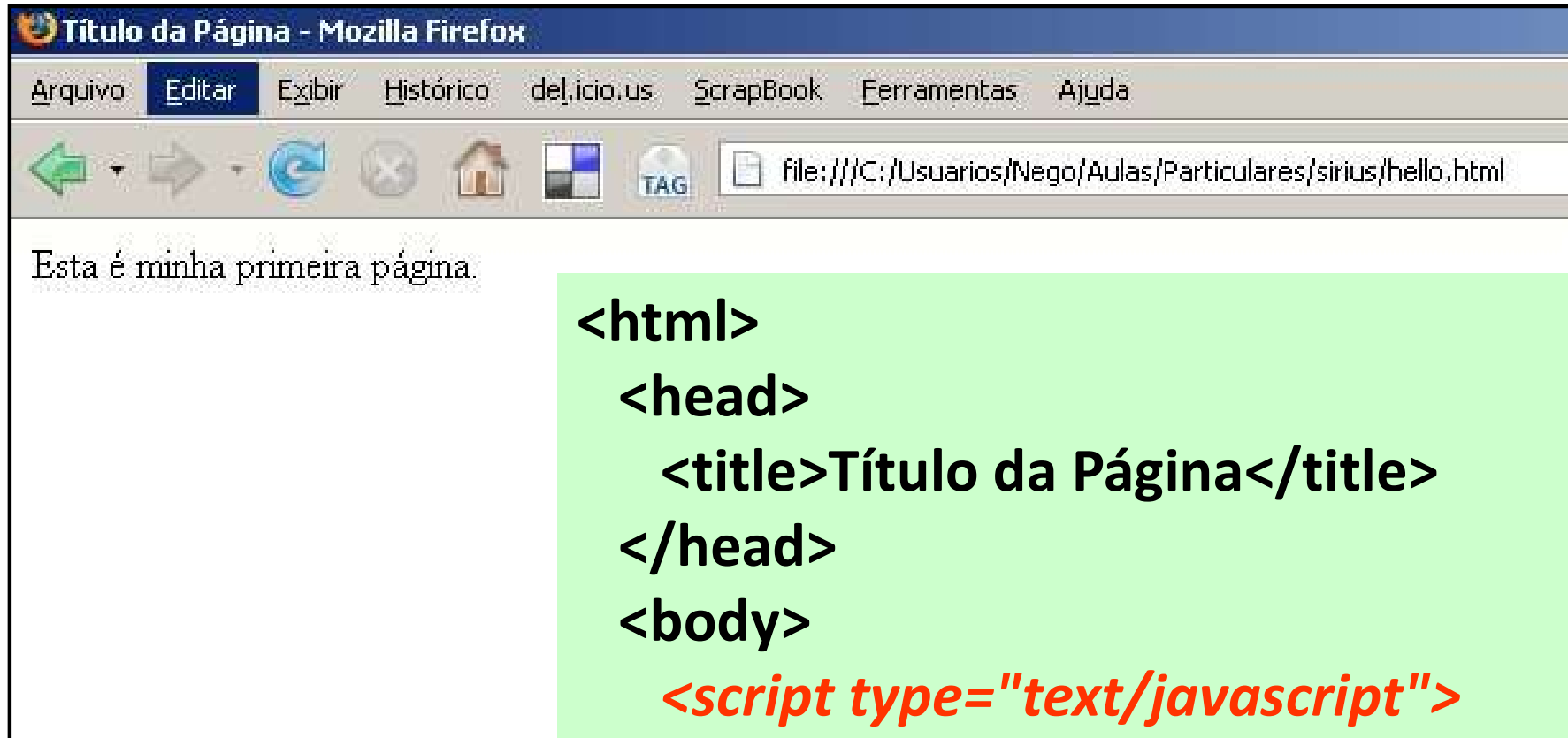
Tableless

- Padrão que vem sendo adotado na web para uso massivo de html+css
- Objetiva, principalmente, melhor acessibilidade de páginas web
- Propõe a não utilização tabelas html para a construção do layout da página (origem do nome)
- Naturalmente, este uso não deveria ser evitado se o que se deseja construir é uma tabela de fato

JavaScript

- É a linguagem de script utilizada por milhões de páginas web
- Foi projetada para aumentar interatividade das páginas web:
 - Validação de formulários, interação com o usuário (p.ex., tratamento de cliques de botões), detecção de navegadores, etc
- É reconhecida pela maioria dos navegadores
- Seu processamento é feito na máquina cliente (browser)
- Não há relação com Java

JavaScript - Exemplo



```
<html>
  <head>
    <title>Título da Página</title>
  </head>
  <body>
    <script type="text/javascript">
      document.write("Esta é minha
primeira página.");
    </script>
  </body>
</html>
```

JavaScript – Onde ocorrem

- Uma tag `<script/>` pode ser definida numa seção head, numa seção body e também pode ser definida externamente:
 - Na seção head, os scripts são executados quando são chamados ou quando algum evento ocorre;
 - Na seção body, os scripts são executados na carga da página web
 - Para definição externa, um arquivo “.js” precisa ser fornecido com as funções necessárias

JavaScript – Exemplo

```
<html>
  <head>
    <title>Título da Página</title>
    <script src="hello.js"></script>
  </head>
  <body onload="message()">
    <script type="text/javascript">
      document.write("<h1>Esta é minha primeira
página.</h1>");
    </script>
  </body>
</html>
```

JavaScript – Mais Usos

- Criação de cookies
- Validação de entrada
- Disparo de funções com retardo (tempo)
- Criação de objetos próprios

Linguagens do Servidor

- O servidor web é um programa que gerencia as requisições HTTP dos clientes
- O processamento e resposta é feito pelo software que controla o site (php, python, asp..)
- O software de controle do site é escrito em uma linguagem de programação, ou mesmo, HTML puro.

AJAX

- Asynchronous Javascript and XML
- Requisições assíncronas para o servidor
- Isso significa:
 - Rapidez de carregamento da página
 - Facilidade de navegação
 - Diminuição do tempo de espera do usuário

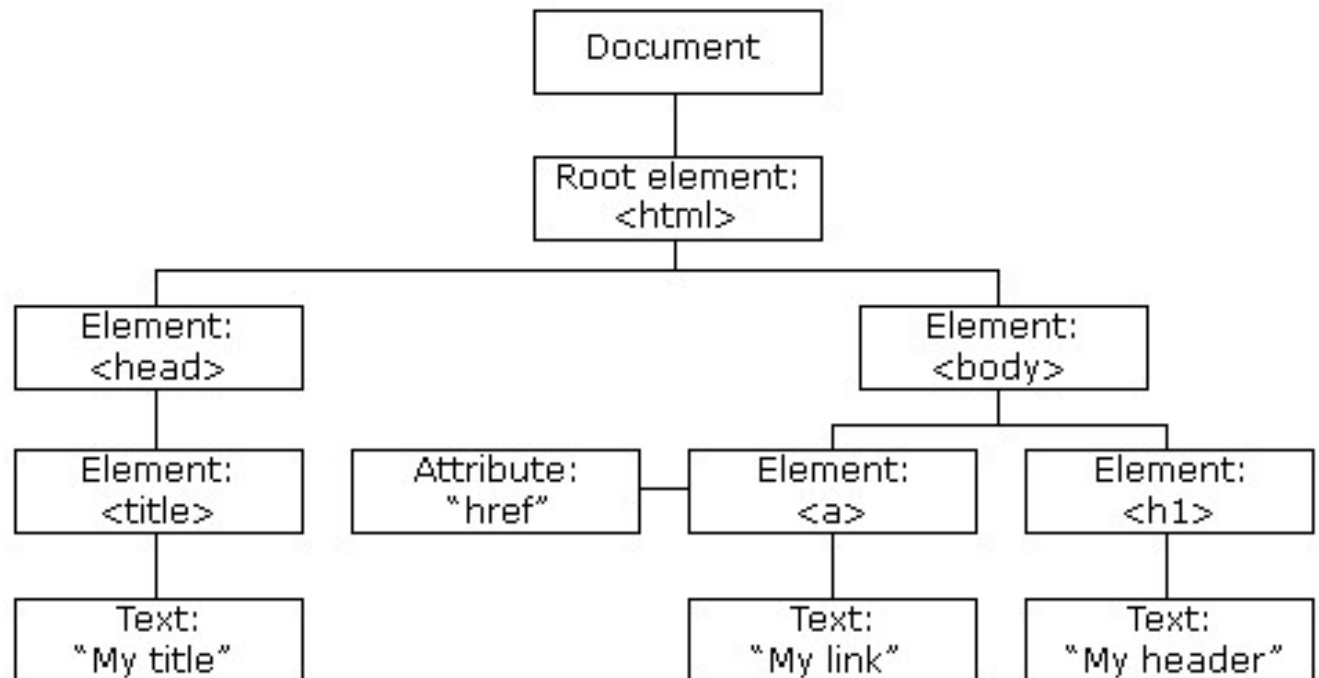
AJAX - Funcionamento

- Baseado no objeto XMLHttpRequest do Javascript
- O objeto XMLHttpRequest faz uma requisição para o servidor assim que ocorra um evento no DOM
- O servidor processa a requisição e envia a resposta
- No AJAX convencional, a resposta é XML
- Na prática, pode ser qualquer formato. O formato mais popular para AJAX é o JSON.

HTML DOM

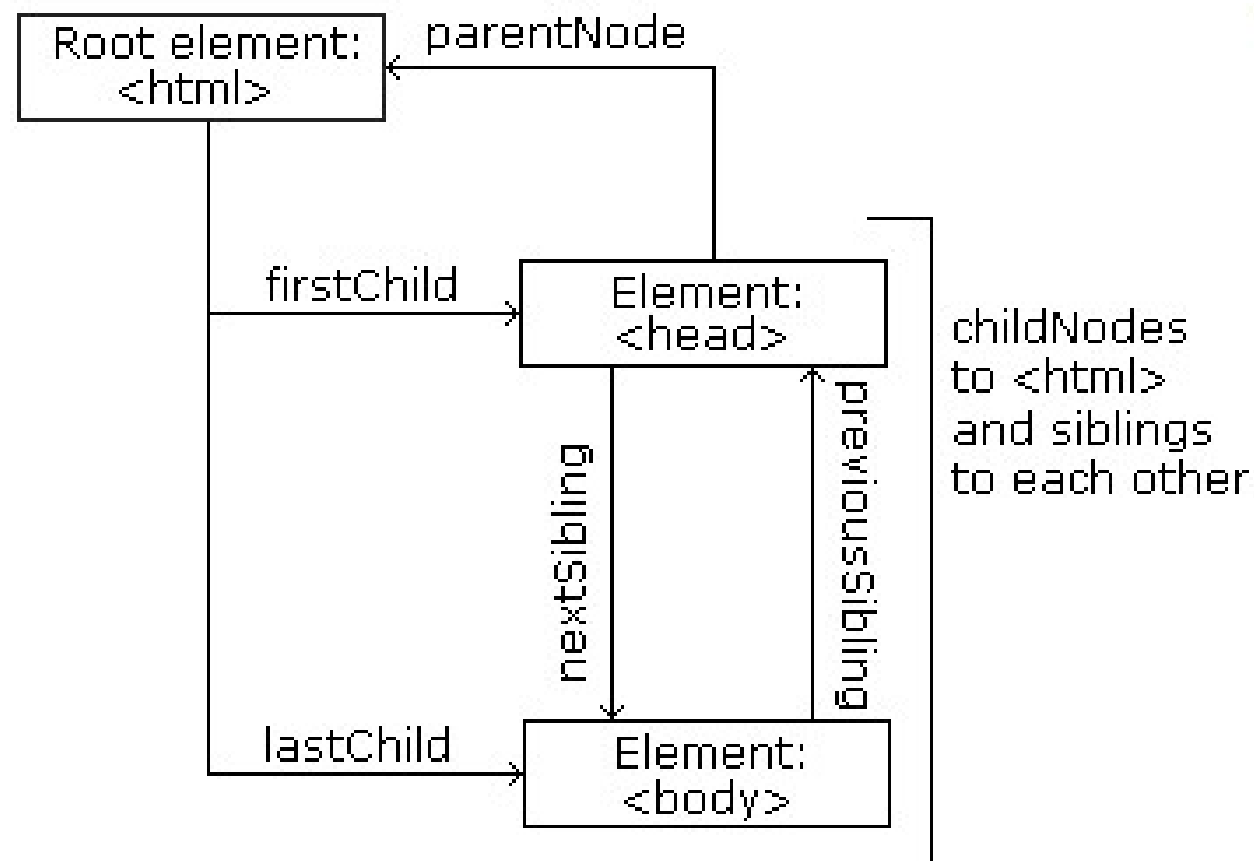
- Define um padrão para acesso a elementos HTML
- O DOM apresenta um documento HTML como uma estrutura em árvore

```
<html>
<head>
<title>My title</title>
</head>
<body>
  <a href="">My link</a>
  <h1>My header</h1>
</body>
</html>
```



HTML DOM

- Relacionamento entre nós numa árvore DOM



HTML DOM - API

- É definida por um conjunto de propriedades e métodos
- Algumas propriedades DOM
 - x.innerHTML: o valor text de x
 - x.nodeName: o nome do elemento x
 - x.nodeValue: o valor de x
 - x.nodeType: o tipo de x (*1 – elemento; 2 – atributo; 3 – texto; ...*)
 - x.parentNode: o nó pai de x
 - x.childNodes: os nós filhos de x
 - x.attributes: os nós atributos de x

HTML DOM - API

- Alguns métodos DOM
 - `x.getElementById(id)`: obtém o elemento com o *id* fornecido
 - `x.getElementsByTagName(name)`: obtém todos os elementos com a tag *name*
 - `x.appendChild(node)`: insere um nó filho *node* em *x*
 - `x.removeChild(node)`: remove o nó filho *node* de *x*

HTML DOM – API

Exemplo

```
<html>
<body>
  <p id="intro">Hello World!</p>

  <script type="text/javascript">
    txt=document.getElementById("intro").innerHTML;
    document.write("<p>Texto do parágrafo intro: " + txt + "</p>");
  </script>
</body>
</html>
```

Frameworks

- Provêm funcionalidades genéricas para um domínio
- Controlam o fluxo da aplicação
- Podem ser extendidos
- Diminuem a complexidade e o tempo de desenvolvimento

Frameworks - Exemplos

- Java: Spring, Struts
- PHP: Zend Framework, CakePHP, Symfony, Joomla, Drupal
- Python: Django
- Ruby: Ruby on Rails
- Javascript*: Prototype, Mootools, JQuery, Dojo, GWT, Script.aculo.us, Highslide, etc...

Frameworks - Java

► Spring

- *Framework Open Source* criado Rod Johnson;
- Tem como base:
 - Padrões de Projetos de inversão;
 - Injeção de dependência;
- Possui *Container* :
 - instancia classes de uma aplicação Java;
 - define as dependências entre elas por meio de um arquivo de configuração em formato XML;
- Fraco acoplamento;
- Arquitetura tem como base interfaces e POJOs (Plain Old Java Objects).

Linguagem de Programação PHP

Hypertext Preprocessor

Site oficial:<http://br.php.net/>

Versão 5.3.8 - <http://www.baixaki.com.br/download/easyphp.htm>

História do PHP

A linguagem surgiu por volta de 1994, como um pacote de programas CGI criados por Rasmus Lerdorf, com o nome *Personal Home Page Tools*, para substituir um conjunto de scripts Perl que ele usava no desenvolvimento de sua página pessoal.

Aplicação do PHP

O **PHP** se trata de uma linguagem de programação voltada para computadores que é interpretada, livre e é muito utilizada para gerar conteúdos no World Wide Web. Este tipo de linguagem surgiu em 94 com um pacote de programas, cuja principal função era substituir um conjunto de scripts que era utilizado no desenvolvimento de uma pagina pessoal. Esta é uma linguagem totalmente modularizada onde torna a instalação e o uso de servidores na web totalmente ideal.

Características

- Trata-se de uma linguagem extremamente modularizada, o que a torna ideal para instalação e uso em servidores web.
- É muito semelhante, em tipos de dados, sintaxe e mesmo funções, com a linguagem C e com a C++.
- Pode ser, dependendo da configuração do servidor, embarcada no código HTML.
- Case Sensitive (Difere maiúsculo de minúsculo).
- Interpretada e Case-sensitive.

A principal característica desta linguagem é:

1. Velocidade, robustez;
2. Sintaxe similar à Linguagem C/C++ e Perl;
3. Portabilidade com independência de plataforma.

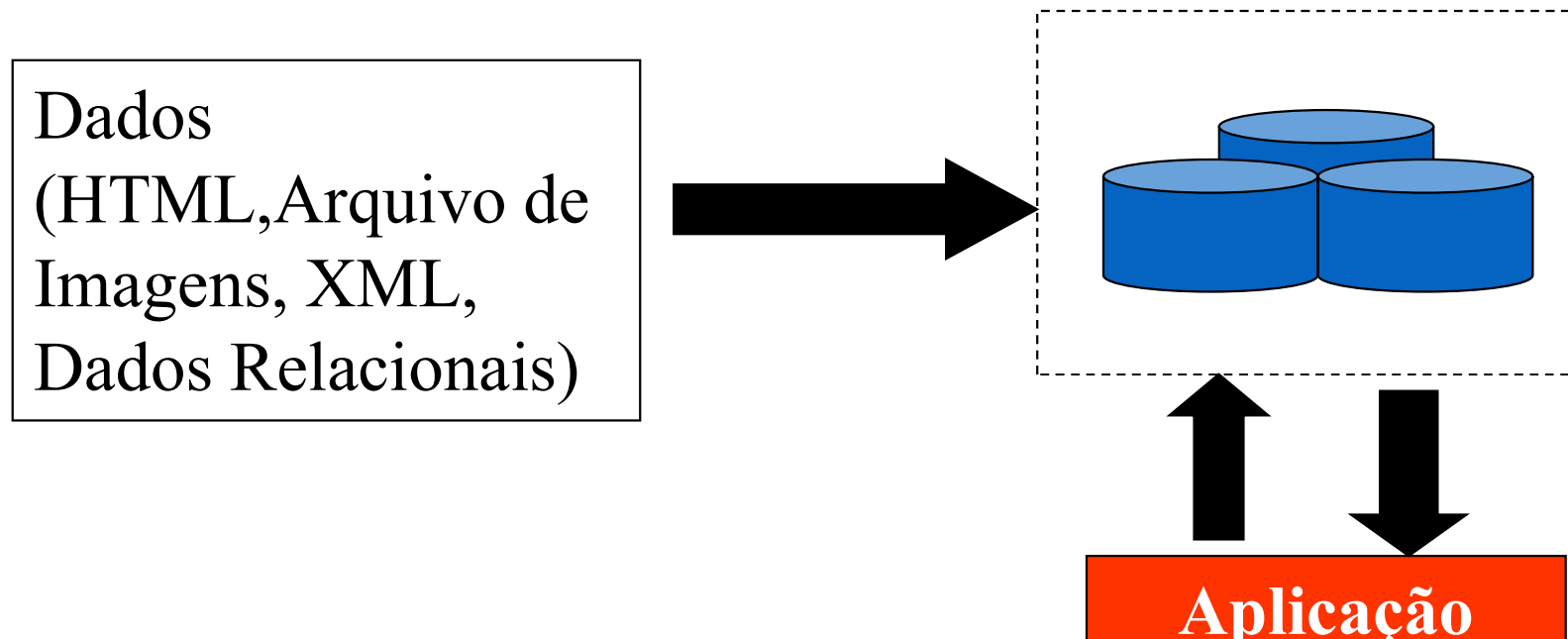
PHP – Conceito de Sistemas Distribuída

O que são?

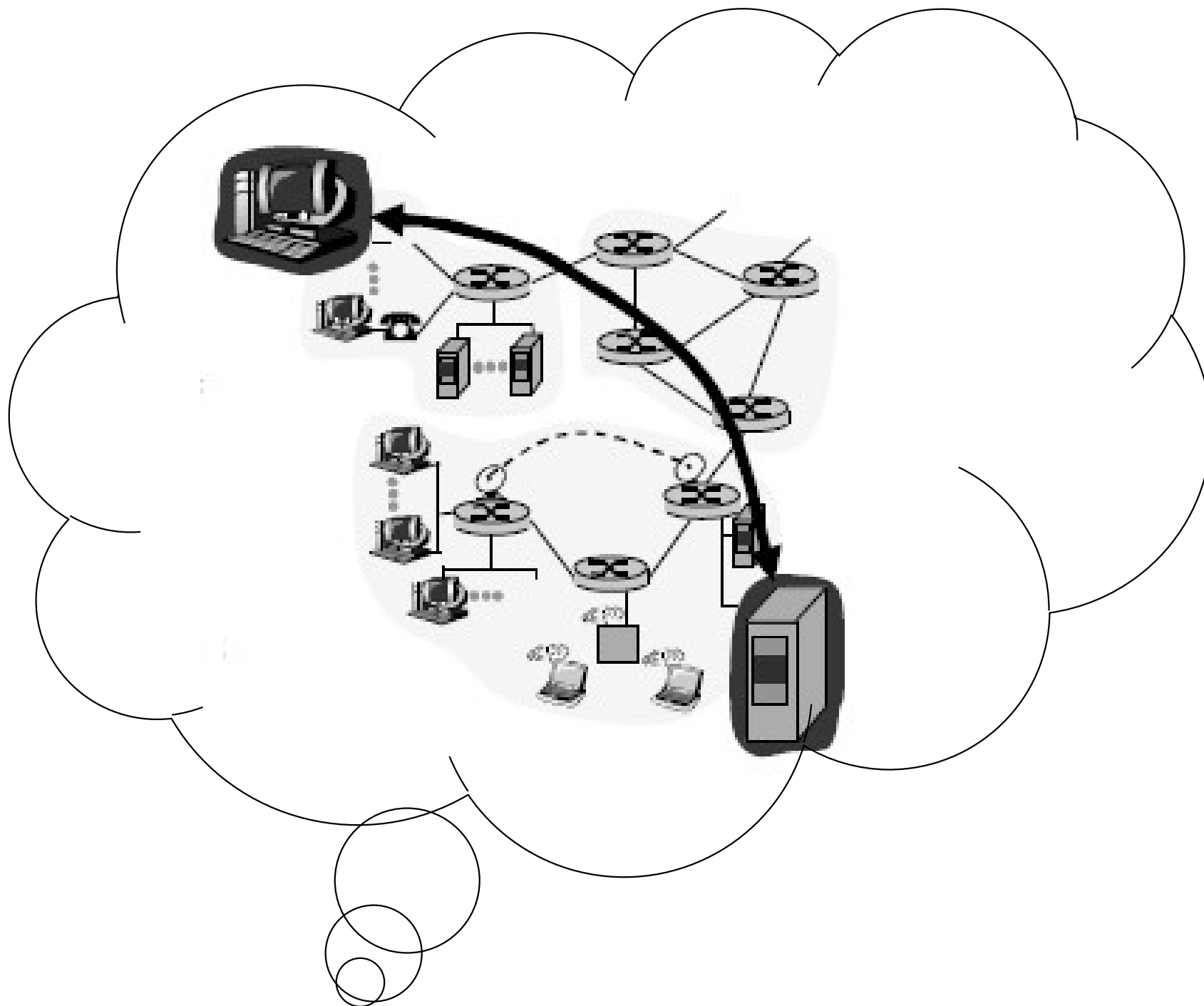
São sistemas compostos por diversas partes cooperantes que são executadas em máquinas diferentes interconectadas por uma rede.

Introdução a Programação Distribuída

- Tem como objetivo desenvolver ferramentas para aquisição e envio de dados em fontes remotas.



Estrutura Física (Distribuída)



Conceitos para um código em PHP
(Page *Hypertext Preprocessor*) - Página de Hipertexto
Pré-processada)

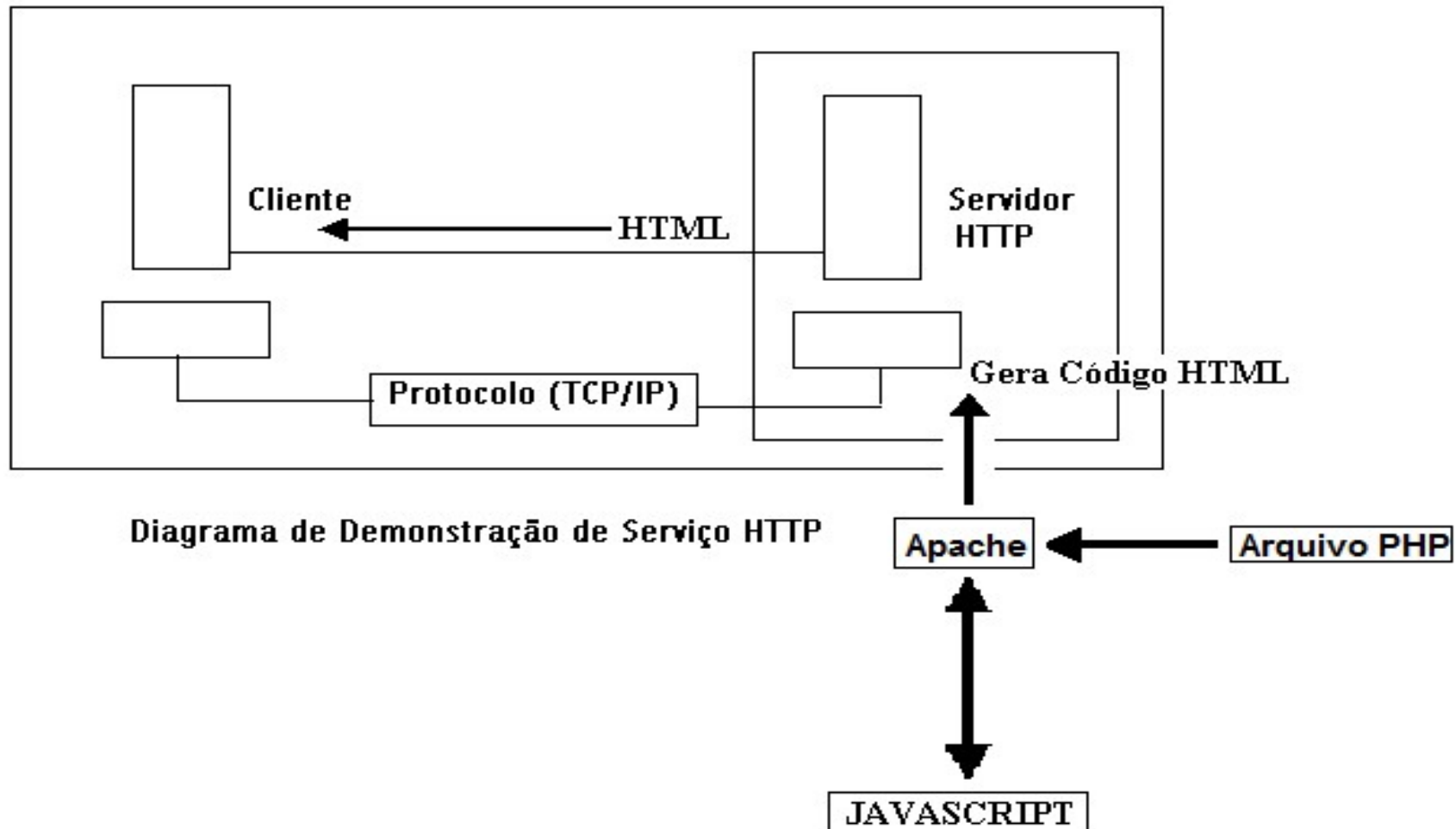
Um código escrito na linguagem PHP roda sempre no servidor nunca no cliente.

PHP- vem a ser um aplicativo que roda com a linguagem semelhante ao “C”. Embutida na estrutura de um código em HTML.

Exemplo de programa (php)

```
<?php
$fat = 1;
$n = $_GET["numero"];
for($i=1;$i<=$n;$i++)
{
    $fat*=$i;
    echo $fat;
}
?>
```

Diagrama de Funcionamento (Página com PHP - Apache)



Obs : Todo programa PHP tem como saída um código HTML.

Constante

São identificadoras criados sem permissão de troca de conteúdos.

Exemplo :

```
define ("pi",3.1415926536);
```

```
$area = 2*pi*$raio;
```

Operadores Aritméticos

+ Adição

- Subtração

* - Multiplicação

/ - Divisão

% - Resto da Divisão (Módulo)

Operadores de atribuição

= Atribuição simples

+= Atribuição com adição

-= Atribuição com subtração

*= Atribuição com multiplicação

/= Atribuição com divisão

%= Atribuição com módulo

.= Atribuição com concatenação

Operadores Lógicos

and - e lógico

Or - ou lógico

Xor - ou exclusivo

! - Não inverso

&& - e lógico

| | - ou lógico

Operadores Relacionais

= = - Igual

!= - Diferente de

< - Menor que

> - Maior que

>= - Maior ou Igual

<= - Menor ou Igual

Operadores de Incremento e decremento

++ - Incremento

-- - Decremento

Estruturas de controle

- Atribuição
- Condição
- Repetição

Exemplo de atribuição

\$base = 3;

\$altura = 4;

\$area = \$base * \$altura;

As variáveis devem ser precedidas do símbolo (\$) com exceção das constantes.

Condicional if()

```
if(<condição>) {  
    <Bloco-Instruções-1>  
}  
else {  
    <Bloco-Instruções-2>  
}
```

Condicional if()

```
if(<condição>)  
    <Instrução-1>  
else  
    <Instrução-2>
```

Exemplo de if()

```
$sexo = "M";  
if($sexo == "F")  
    echo "Ir ao cabelereiro";  
else  
    echo "Ir Jogo de futebol";
```


Comando switch case

Permite realizar testes para uma seqüência de condições sendo mais resumido que a instrução if().

Condicional Switch()

```
switch ($dado) {  
    case 0:  
        echo "i equals 0";  
        break;  
    case 1:  
        echo "i equals 1";  
        break;  
    case 2:  
        echo "i equals 2";  
        break;  
    default:  
        echo "i is not equal to 0, 1 or 2";  
  
}
```

Operador Ternário (?:)

Operador condicional de três termos.

\$varm = <condição> ? <Verdadeiro>:<Falso>;

Estrutura de repetição

As estruturas de repetição do PHP são similares as das linguagens de alto nível como C++, Java e outras.

Exemplo :

for() e while().

Sintaxe for

```
for(<inicialização>;<condição>;<incremento ou decremento>)  
    <comando>;
```

Ou

```
for(<inicialização>;<condição>;<incremento ou decremento>) {  
    <comando1>;  
        <comando2>;  
}
```

Ou

```
for(<inicialização>;<condição>;<incremento ou decremento>) :  
    <comando1>;  
        <comando2>;  
endfor;
```

Exemplo - 1 for()

```
<?php
    $fat = 1;
    for($i=1;$i<=5;$i++)
        $fat*=$i;
    echo $fat;
?>
```

Sintaxe for(): endfor;

```
<?php
    $fat = 1;
    $soma = 0;
    for($i=1;$i<=5;$i++):
        $fx = 2 * $i - 1;
        $soma += $fx;
        echo $fx."<br>";
    endfor;
    echo $soma."<br>";
?>
```

Estrutura de repetição comando while

```
while (expr)  
    statement
```

```
while (expr) {  
    statement  
}
```

```
while (expr):  
    statement  
endwhile;
```

```
do {  
    statement  
} while (expr);
```


Exemplo de while{}

```
$i = 1;  
while($i <= 10){  
    echo $i."<br>";  
    $i++;  
}
```

Exemplo de while:endwhile

```
$i = 1;  
while ($i <= 10):  
    echo $i;  
    $i++;  
endwhile;
```

Exemplo: do while

```
$i = 0;  
do {  
    echo $i."<br>";  
    $i++;  
} while ($i <= 5);
```

Bibliografia

- ▶ HTTP 1.1 : <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html>
- ▶ URI : <http://www.w3.org/Addressing/URL/uri-spec.html>
- ▶ Foundation Website Creation with CSS, XHTML and Javascript. Lane, J.; Moscovitz, M.; Lewis, J. EditoraApress, 2008.
- ▶ Professional Ajax, 2nd Ed. Zakas, N.; Fawcett, J.; MacPeak, J. EditoraWrox
- Notacoes de aula: Alessandro Cruz Marcelo Brandão Theodoro Júnior