unesp

Desenvolvimento de Aplicações para Internet

Aula 04

Celso Olivete Júnior

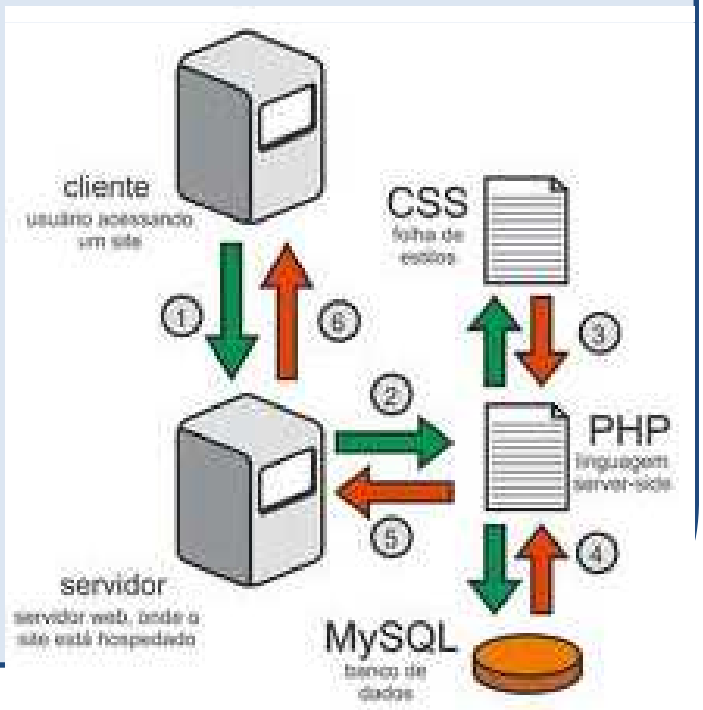
olivete@fct.unesp.br

Nas aulas anteriores...

- HTML
- CSS
- JavaScript

Na aula de hoje...

- Desenvolver aplicações para a Web utilizando as tecnologias **HTML**, ***JavaScript***, **CSS** e **PHP**.



Roteiro

- Conceitos básicos de programação PHP
- Variáveis
- Operadores: aritméticos/lógicos

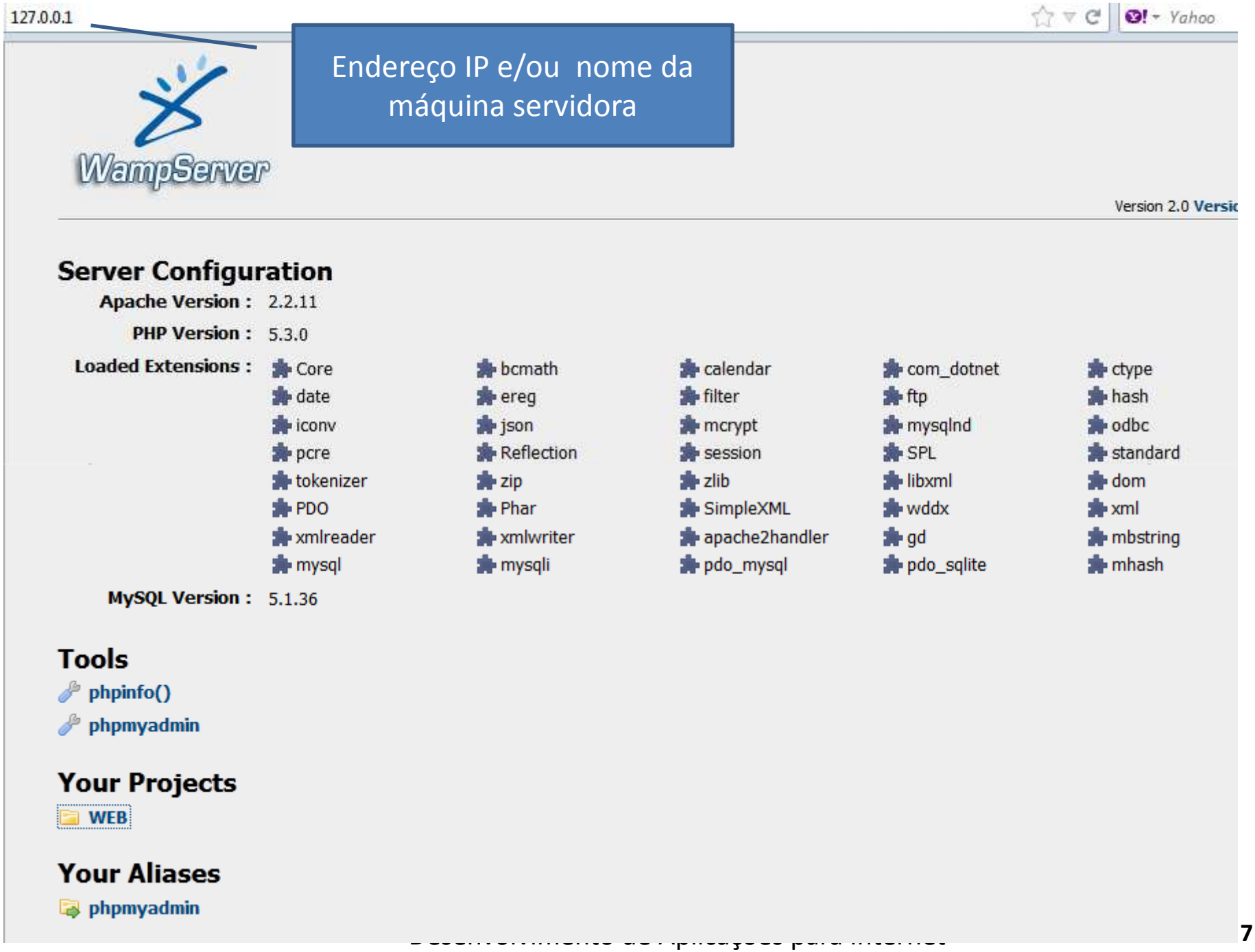
Requisitos necessários

- PHP: a linguagem de programação
- APACHE: servidor *Web*
- *MySQL*: gerenciador de banco de dados
- *PHPEditor*: editor PHP (facilitar a digitação do código)
- TopStyle: editor CSS
- Free JavaScript Editor (javascript, html, php,...)

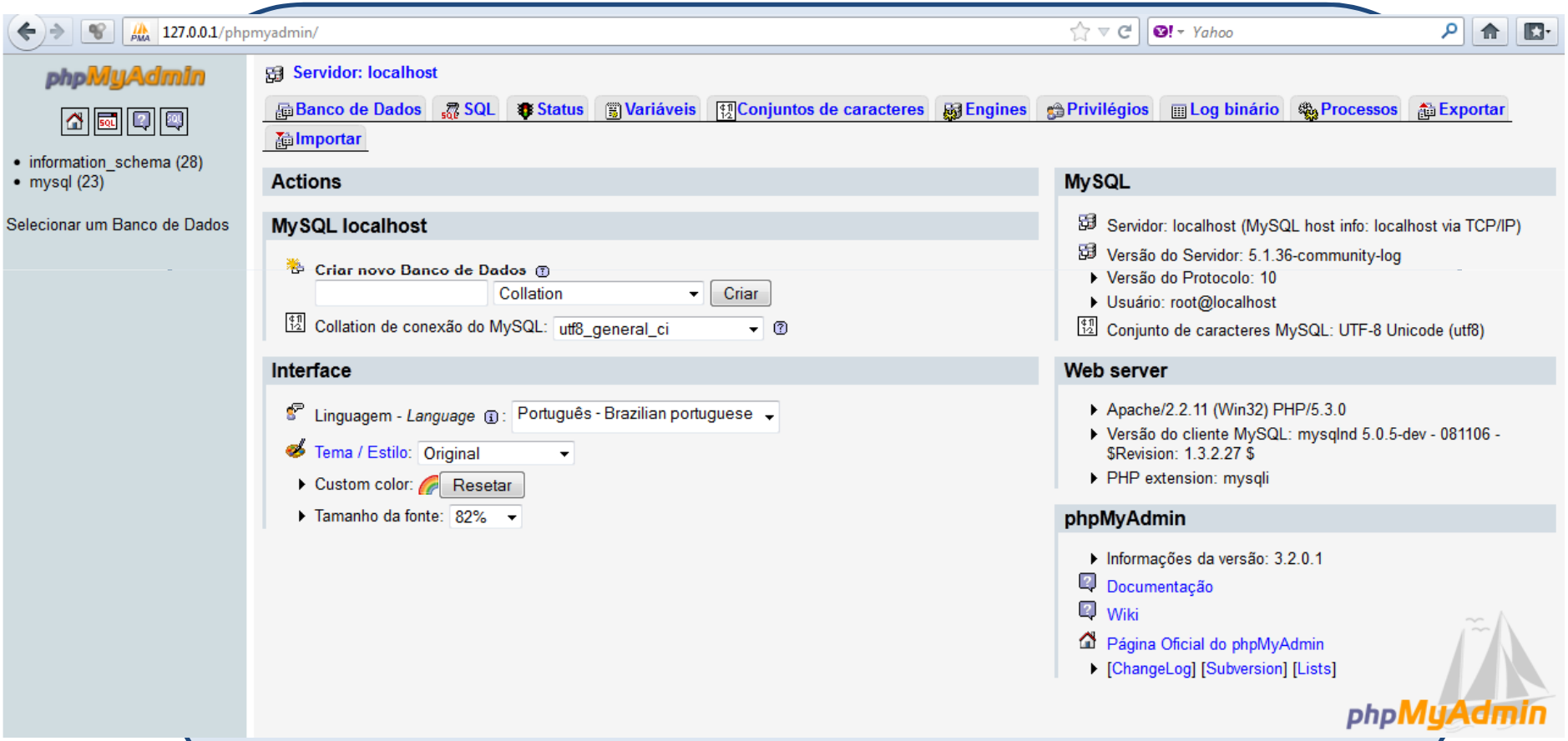
Requisitos necessários

- **Softwares**
  - WAMP
  - XAMPP

- **W** significa: Sistema operacional Windows
- **X** significa: Qualquer sistema operacional, Windows, Linux, Mac OS X,...
- **A** significa: Apache (Um famoso servidor HTTPd)
- **M** significa: MySQL (banco de dados)
- **P** significa: PHP (significa "PHP: Hypertext Preprocessor", que é uma linguagem de programação)
- **P** significa: Perl (uma linguagem de programação dinâmica)


127.0.0.1
Yahoo
Endereço IP e/ou nome da máquina servidora
WampServer
Version 2.0 Versic
Server Configuration
Apache Version : 2.2.11
PHP Version : 5.3.0
Loaded Extensions :
Core
bcmath
calendar
com_dotnet
ctype
date
ereg
filter
ftp
hash
iconv
json
mcrypt
mysqlnd
odbc
pcre
Reflection
session
SPL
standard
tokenizer
zip
zlib
libxml
dom
PDO
Phar
SimpleXML
wddx
xml
xmlreader
xmlwriter
apache2handler
gd
mbstring
mysql
mysqli
pdo_mysql
pdo_sqlite
mhash
MySQL Version : 5.1.36
Tools
phpinfo()
phpmyadmin
Your Projects
WEB
Your Aliases
phpmyadmin

PhpMyAdmin


127.0.0.1/phpmyadmin/
Yahoo
phpMyAdmin
Servidor: localhost
Banco de Dados
SQL
Status
Variáveis
Conjuntos de caracteres
Engines
Privilégios
Log binário
Processos
Exportar
Importar
information_schema (28)
mysql (23)
Selecionar um Banco de Dados
Actions
MySQL localhost
Criar novo Banco de Dados
Collation
Criar
Collation de conexão do MySQL:
utf8_general_ci
Interface
Linguagem - Language :
Português - Brazilian portuguese
Tema / Estilo:
Original
Custom color:
Resetar
Tamanho da fonte:
82%
MySQL
Servidor: localhost (MySQL host info: localhost via TCP/IP)
Versão do Servidor: 5.1.36-community-log
Versão do Protocolo: 10
Usuário: root@localhost
Conjunto de caracteres MySQL: UTF-8 Unicode (utf8)
Web server
Apache/2.2.11 (Win32) PHP/5.3.0
Versão do cliente MySQL: mysqlnd 5.0.5-dev - 081106 - $Revision: 1.3.2.27 $
PHP extension: mysqli
phpMyAdmin
Informações da versão: 3.2.0.1
Documentação
Wiki
Página Oficial do phpMyAdmin
[ChangeLog] [Subversion] [Lists]
phpMyAdmin

PHP – linha do tempo

- 1995 – Rasmus Lerdorf (PHP-FI – Personal Home Page – Forms Interpreter)
- Rasmus Lerdorf liberou o código da linguagem, com isto começaram a surgir novas funcionalidades
- 1997 – Primeira versão da linguagem PHP realmente pronta PHP3, escrita por Zeev Suraski e Andi Gutmans
- 1998 – Zeev Suraski e Andi Gutmans começam a trabalhar na versão 4.0
- 2000 – Lançamento oficial do PHP 4.0
- atual- PHP 5.4.16

Introdução

- O PHP (Hipertext Preprocessor) é uma linguagem de script voltada para o desenvolvimento de páginas dinâmicas para a Internet, permitindo ao programador inserir seus comandos diretamente nos códigos HTML de uma página.

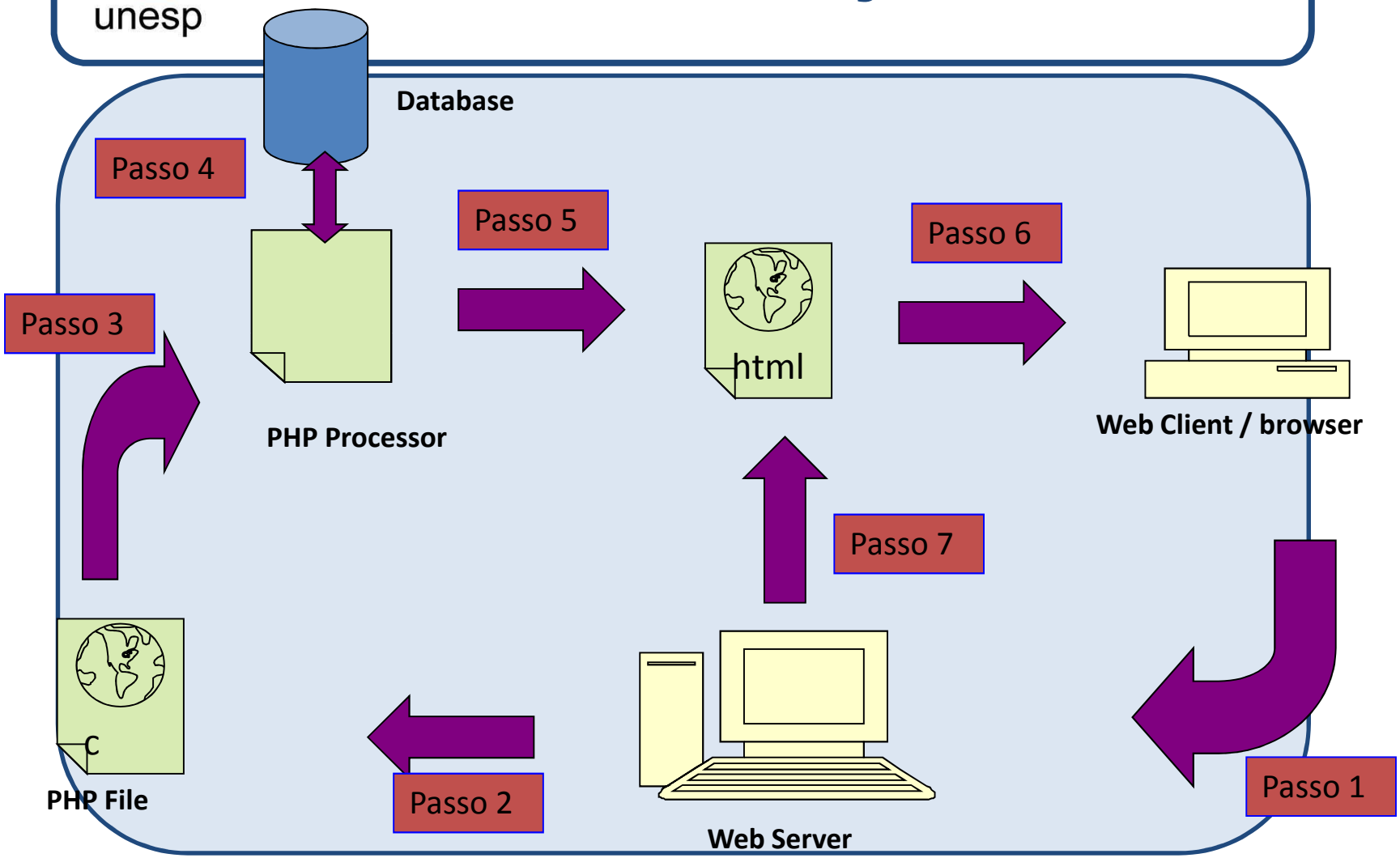
- Possui código aberto, e devido à suas excelentes características técnicas, vem ganhando milhões de adeptos.

Introdução

- O PHP oferece suporte de conexão com diversos bancos de dados, como Oracle, PostgreSQL, MySQL, etc. Outra vantagem é a possibilidade de ser executado em diversos sistemas operacionais como Linux, Windows, Unix, etc.

- O PHP foi desenvolvido para rodar no lado do servidor ➔ as páginas são processadas no servidor e enviadas ao cliente (em HTML) ao solicitar uma página por intermédio de seu navegador.

Introdução


unesp
Database
Passo 4
Passo 5
Passo 6
Passo 3
PHP Processor
html
Web Client / browser
Passo 7
C
PHP File
Passo 2
Web Server
Passo 1

Sites dinâmicos

- Imagine que você tivesse que desenvolver um site para uma editora que quisesse publicar todos os seus 670 títulos, ou uma livraria virtual com 20.000 títulos, ou até mesmo uma loja virtual de CDs.

- Certamente, utilizando somente as técnicas do HTML você jamais chegaria ao final do seu projeto, pois para cada livro uma nova página teria de ser criada, sem falarmos das alterações de preço, cancelamentos de livros, inclusão de novos títulos, etc.

Sites dinâmicos

- Como você já sabe, o HTML é uma linguagem de marcação de hipertextos e não uma linguagem de programação; portanto as páginas criadas para a Internet que utilizam somente o HTML são estáticas e funcionam como se fossem páginas de uma revista...

- Se analisarmos o caso da editora ou da livraria on-line, seria muito melhor se pudéssemos criar uma única página em branco e preenchê-la com informações vindas de um banco de dados, sempre que algum usuário solicitasse alguma informação. Em outras palavras, essas páginas passam a ser dinâmicas.

Sites dinâmicos

- Para transformarmos uma página estática em dinâmica, usamos um recurso que consiste na inserção de pequenos programas em um documento HTML. A esses programas damos o nome de scripts.

Sites estáticos

```
<html>
<head>
<title>FS Editora On-Line</title>
</head>
<body>
<b>Olá! Hoje é 16/03/2012 </b>
</body>
</html>
```

E amanhã??

Sites dinâmicos

```
<html>
<title>FS Editora On-Line</title>
<body>
<b>Olá! Hoje é </b>
<?php
echo date (“d/m/Y”);
?>
</body>
</html>
```

Sites dinâmicos

- Os scripts são escritos com a utilização de linguagens de programação adequadas para este fim, como por exemplo: JavaScript, PHP, etc.

Sites dinâmicos

- Basicamente existem duas maneiras de executar um *script*:
  - Os scripts são enviados pelo servidor da Web juntamente com os códigos HTML para o navegador do usuário, cabendo ao navegador interpretar esses scripts e executá-los.
  - Os scripts são interpretados e executados pelo próprio servidor da Web, o resultado desse processamento é inserido na página e enviado para o navegador do usuário no formato HTML.

Sites dinâmicos

- No primeiro caso, o navegador deve ser capaz de interpretar a linguagem de script usada na criação da página.
- Já no segundo caso, qualquer navegador exibe suas páginas, pois não precisa interpretar os scripts, uma vez que eles são executados no próprio servidor da Web.
- O PHP é uma linguagem de script executada pelo servidor, ao passo que o *JavaScript* é executado no cliente.

Sites dinâmicos

- A programação em JavaScript pode ser vista e copiada por qualquer pessoa. Para isto, basta escolher a opção Exibir/Código-fonte no menu do navegador.

- O PHP é exatamente o contrário, as linhas de programação não podem ser vistas por ninguém, já que elas são executadas no próprio servidor, e o que retorna é apenas o resultado (html) do código executado.

Sites dinâmicos

- Um exemplo simples:
  - Você já deve ter visto sites que exibem a data e a hora atual em suas páginas. Se essas informações forem escritas utilizando JavaScript, a data e a hora mostradas são retiradas do do seu computador. Agora, se a data e a hora forem escritas utilizando PHP, essas informações serão retiradas do servidor.

PHP - sintaxe

- Sintaxe geral

```
<?php
echo date (“d/m/Y”);
?>
```

- PHP é *case-sensitive*

PHP - sintaxe

- Comentários em código PHP

```
// comentário 1 linha

/*comentário de N
    linhas */
```

PHP - sintaxe

- Embutindo códigos HTML dentro de um script PHP
- Utilize echo ou print

```
<?php

echo "<b>Olá, Fulano!</b> Bem vindo!";

print "<b>Olá, Fulano!</b> <br> Bem
vindo!";

?>
```

PHP - sintaxe

- Variáveis – iniciam-se com **$**
  - As variáveis contidas em um programa PHP são voláteis e somente existem enquanto o script estiver sendo executado. Elas não precisam ser definidas para serem usadas, basta atribuir um valor diretamente a ela para ser criada.

```
$titulo = “PHP – 4”;
$totalEstoque = 2350;
$_autor;   $preco_item;
```

  - Errado: $1autor, $peso pessoa, $preco+item

PHP – primeiro exemplo

```
<?php
 //legal, estou escrevendo meu primeiro programa em PHP
echo "Olá Mundo!!!";
?>
```

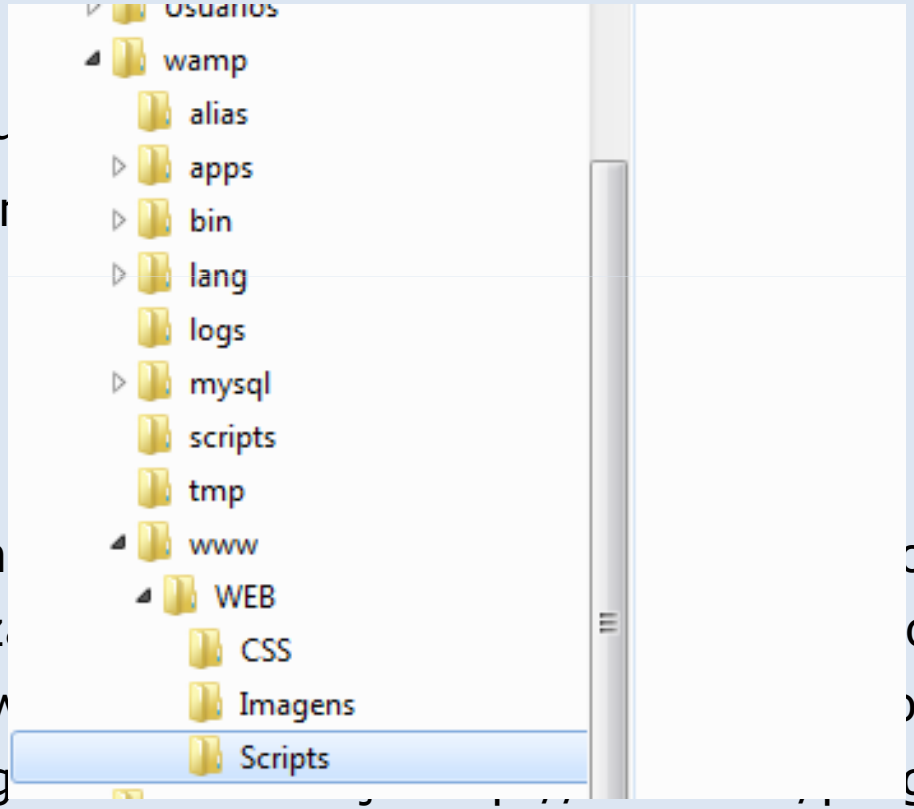
- Salve como prog1.php e envie para o diretório do servidor (no nosso caso, c:\wamp\www). Para ver o resultado, basta você acessar pelo navegador o endereço http://localhost/prog1.php.

Ou

127.0.0.1/prog1.php

PHP – primeiro exemplo

```
<?php
 //legal, estou                                em PHP
echo "Olá Mur
?>
```



- Salve com o que você está utiliz osso caso, c:\wamp\w ocê acessar pelo naveg g1.php.

PHP – primeiro exemplo

- Se você escolher a opção Exibir/Código-fonte em seu navegador, você verá o código que seu browser recebeu, que foi o seguinte:

```
<b>Olá, Amigo!</b> Bem vindo!
```

PHP – primeiro exemplo

- Exibindo resultados no navegador:
  - Para o navegador mostrar algum resultado é necessário que a página tenha pelo menos um comando ***echo*** ou ***print*** para escrever algo, ou então comandos HTML que escrevam os conteúdos da página

PHP – outro exemplo

- Veja o exemplo:

```
<html>
<body>
<?php
$dia = date (“d/m/Y”);
$base = 5.5;
$altura = 1;
$area = $base * $altura;
?>
</body>
</html>
```

Perceba que não há nenhum comando echo no programa, por isto seu navegador mostrará uma tela em branco.
Ao visualizar o código-fonte recebido pelo navegador, você verá apenas as tags do HTML

PHP - sintaxe

- Interpolação de variáveis

```
<html>
<body>
<?php
$time = “Corinthians”;
$ano1 = 2000;
$ano2 = 2012;
$frase1 = “O $time é o melhor time do mundo!”;
$frase2 = “O $time foi o campeão do mundo em $ano1 e $ano2”;
echo “<h3>$frase1</h3>”;
echo “<h3>$frase2</h3>”;
?>
</body>
</html>
```

PHP – tipos de dados

- São números inteiros, reais, decimais, octais, hexadecimais.
- Os dados numéricos são utilizados geralmente para efetuar cálculos
- 5, 4.012, 0xBC, 43000000

PHP – tipos de dados

- Alfanuméricos
  - Também conhecidos como strings. São sequências de caracteres, que podem ser delimitadas por aspas simples (`) ou aspas duplas (").

```
echo '<p align=center>Texto utilizando aspas simples</p>'
```

  - Utilizando aspas duplas pode-se fazer a interpolação de variáveis

```
<?php
$palavra = "teste";
$frase = "Isto é um $palavra";
echo $frase;
?>
```

PHP – tipos de dados

- Constantes
  - São valores que são predefinidos no início do programa e que não mudam ao longo de sua execução. Você pode definir suas próprias constantes usando o comando define.
  - define(<constante>,<valor>);

```
<html>
<body>
<?php
define(“meunome”, “Fulano”);
define(“profissao”, “estudante”);
echo “O meu nome é”.meunome;
echo “<br>”;
echo “A minha profissão é”.profissao;
?>
</body>
</html>
```

PHP – tipos de dados

- Concatenação
  - Pode-se concatenar quantos dados necessários. Todos serão exibidos como apenas uma sequência de caracteres.
  - O recurso foi utilizado no exemplo anterior, representado pelo ponto (.)
  - Ex:
  - Echo "A frase inicia com o texto = ".$variavel1." e termina com".$variavel2;

PHP – escopo de variáveis

- As variáveis existem somente no contexto em que são definidas. Podem existir funções com variáveis locais com os mesmos nomes.

- Para que uma variável mantenha o mesmo valor, deve-se defini-la como global.

```
global $salario_base = 1000;
global $nome = “Getulio Vargas”;
```

Variável global
Exemplo de uso

```
<?php
$a = 1;
$b = 2;

function Soma()
{
   global $a, $b;
   $b = $a + $b;
}
Soma();
echo $b;
?>
```

PHP – conversões

- Se tivermos uma string contendo somente números, o PHP somará normalmente esse valor com outra variável do tipo numérico. Se houver textos e números em uma string, o PHP utilizará somente a parte em que estão os números para efetuar operações aritméticas.

```
$string = “5”;
$numero = 3;
$texto = “3 vezes”;
```

- Se somarmos $numero+$string, o resultado será 8. Se somarmos $numero+$texto, o resultado será 6.

PHP – conversões

- Algumas vezes é preciso fazer a conversão manualmente para realizar certos tipos de cálculos.

```
<?php
$x = 50;
$y = 2.35;
$soma = (int)$y + $x;
?>
```

Conversor	Descrição
(int), (integer)	Converte para inteiro
(real), (float), (double)	Converte para ponto flutuante
(string)	Converte em string
(array)	Converte em array

PHP – operadores

- **Operadores Aritméticos:**
  - + - * / %
  - -oper : troca o sinal do operando
  - ++oper : pré-incremento. Incrementa o valor do operando e depois realiza a operação
  - --oper : pré-decremento. Decrementa o valor do operando e depois realiza a operação
  - oper++ : pós-incremento. Realiza a operação e depois incrementa o valor do operando
  - oper-- : pós-decremento. Realiza a operação e depois decrementa o operando.

PHP – operadores aritméticos

- Operadores Aritméticos:

```
<?php
$a = 1;
$b = 3;
$c = 5;
$res1 = ++$b - $a;
$res2 = $c-- + $a;
echo “a = $a<br>b = $b<br>”;
echo “res1 = $res1<br> res2 = $res2”;
?>
```

Res1 = 3
Res2 = 6

PHP – operadores

Comparação

Operador	Descrição
op1 == op2	V se op1 for igual a op2
op1 >= op2	V se op1 for maior ou igual a op2
op1 <= op2	V se op1 for menor ou igual a op2
op1 != op2	V se op1 for diferente de op2
op1 <> op2	Também representa diferença
op1 > op2	V se op1 for maior que op2
op1 < op2	V se op1 for menor que op2

O operador de comparação == pode ser usado tanto na comparação de números quanto na comparação de textos.

PHP – operadores

- Atribuição

Operador	Descrição
op1 = op2	op1 recebe o valor de op2
op1 += op2	Equivale a op1 = op1 + op2
op1 -= op2	Equivale a op1 = op1 – op2
op1 *= op2	Equivale a op1 = op1 * op2
op1 /= op2	Equivale a op1 = op1 / op2
op1 .= op2	Equivale a op1 = op1.op2
op1 %= op2	Equivale a op1 = op1 % op2

PHP – operadores

- Atribuição – exemplo

```
<?php
$soma=0;
$valor1=10;
$valor2=20;
$valor3=30;
$soma += $valor1; //$soma fica com 10
$soma +=$valor2; //$soma fica com 10+20 = 30
$soma *=$valor3; //$soma fica com 30 *30 = 900
$soma %=100; //$soma fica com 900%100 = 0
echo $soma;
?>
```

PHP – operadores lógicos

Operador	Descrição
!op1	V se op1 for falso
op1 AND op2	V se op1 e op2 forem verdadeiros
op1 OR op2	V se op1 ou op2 forem verdadeiros
op1 XOR op2	V se só op1 ou se só op2 for verdadeiro
op1 && op2	V se op1 e op2 forem verdadeiros
op1 || op2	V se op1 ou op2 forem verdadeiros

PHP – operadores lógicos

- Exemplo

```
<?php
...
if (empty($nome) OR empty($email) OR empty($cpf))
{
echo “Você deve preencher os campos nome, email e CPF!”;
}
...
```

Manipulação de Datas e Horas no Servidor

- Para mostrar um campo Data e Hora o caminho mais prático é a utilização da função DATE.

```
<?php
echo date('parametro1/parametro2/parametro3');
?>
```

- Ex: echo "hoje é dia". date('d/m/y');

Manipulação de Datas e Horas no Servidor

Letras chaves para usar com o comando date	
d	Dia do mês, numérico (01...31)
j	Dia do mês, numérico (1...31)
w	Dia da semana (0=Domingo...6=Sábado)
D	Nome do dia da semana abreviado (Sun...Sat)
M	Nome do mês abreviado (Jan...Dec)
F	Nome do mês completo (January...December)
Y	Ano, numérico, 4 digitos
y	Ano, numérico, 2 digitos
H ou g	Hora (00...23)
h	Hora (01...12)
i	Minutos, numérico (00...59)
s	Segundos (00...59)
A	AM ou PM
a	am ou pm

PHP – Manipulação de Strings

- Função STRLEN()

- Obtém a quantidade de caracteres de uma string.

```
<?php
$palavra = “Ciência da Computação”;
echo “A variável contém:“.strlen($palavra) . “caracteres”;
?>
```

PHP – Manipulação de Strings

- Função TRIM()
  - Obtém a quantidade de caracteres de uma string eliminando os espaços iniciais e finais.

```
<?php
$palavra = “Ciência da Computação “;
echo “A variável contém: “ .
strlen($palavra) . “caracteres” . “<br>”;
$palavra = trim($palavra);
echo “A variável agora contém: “ .
strlen($palavra) . “caracteres”;
?>
```

PHP – Manipulação de Strings

- Função **SUBSTR**(string, início, comprimento)

- Extrai partes de uma string.

```
<?php
$data = date(“m-d-Y”);
$dia = substr($data,3,2);
$mês = substr($data,0,2);
$ano = substr($data,6,4);
$novadata = $dia . “/” . $mês . “/” . $ano;
echo $novadata;
?>
```

PHP – Manipulação de Strings

- Função **EXPLODE**(delimitador, string)

- Extrai partes de uma string.

```
<?php
$data = date(“m-d-Y”);
$partes = explode(“-”, $data);
$parte1 = $partes[0];
$parte2 = $partes[1];
$parte3 = $partes[2];
$novadata = $parte2 . “/” . $parte1 . “/” . $parte3;
echo $novadata;
?>
```

PHP – Manipulação de Strings

- Funções strtoupper e strtolower
- strtoupper() converte todos os caracteres contidos em uma string em letras maiúsculas.
- strtolower() converte todos os caracteres contidos em uma string em letras minúsculas.

```
<?php
$A = “Olá pessoal! ”;
$B = strtoupper($A);
$C = strtolower($A);
echo $B .”<br>”;
echo $C;
?>
```

PHP – Operadores condicionais e loops

- Condicional **IF**
  - If (...condição...) {...ação...}
  - Uma ação somente será executada se a sua condição for verdadeira. Caso essa condição seja falsa, todos os códigos compreendidos entre as chaves serão ignorados pelo PHP.

PHP – Operadores condicionais e loops

```
if (condição) {
                    }
else if (condição) {
                    }
else {
                    }
```

PHP – Operadores condicionais e loops

- switch

```
switch (variável) {
        case condição : comandos;
                        break;
        case condição : comandos;
                        break;
        ...
        default: comandos;
        }
```

PHP – Operadores condicionais e loops

```
switch ($idade) {
      case 7 : case 8: case 9: case 10: case 11:
            echo “Você ainda é uma criança.<br>”;
                  break;
      case 12: case 13: case 14: case 15: case 16:
            echo “Te cuida...<br>”;
                  break;
      default:
            echo “Você já é um adolescente.<br>”;
            }
```

PHP – Operadores condicionais e loops

- FOR

for (inicialização; condição; operador)

{

...trecho a ser repetido...

}

```
for ($x = 1; $x <=10; $x++)
{
echo “Olá mundo!”;
}
```

PHP – Operadores condicionais e loops

- WHILE

```
while (condição) {
          }

$i = 0;
while ($i < 10) {
     echo “Esta é a linha $i. <br>”;
$i++;
}
```

PHP – Operadores condicionais e loops

- do while

```
do {
     } while (condição);

$i = 0;
do {
echo “Esta é a linha $i. <br>”;
$i++;
} while ($i < 10);
```

PHP – vetores e matrizes

- Vetores

```
$vetor[0] = 30;
$vetor[1] = 40;
```

- Se não colocarmos o índice do vetor entre colchetes, o PHP procurará o último índice utilizado e irá incrementá-lo, armazenando assim o valor na posição seguinte do array:

```
$vet[] = 1
$vet[] = 2
```

PHP – vetores e matrizes

- O índice também pode ser um texto, e nesses casos o texto é chamado de chave associativa:

```
$vetor[“turma”] = “Informática”;
$vetor[“ano”] = 2001;
```

PHP – vetores e matrizes

- Matrizes

```
$clube [“RS”] [“PortoAlegre”] = “Gremio”;
$clube [“RS”] [“Caxias”] = “Juventude”;
$clube [“MG”] [“BeloHorizonte”] = “Atlético”;
$clube [“SP”] [“São Paulo”] = “Corinthians”;
```

PHP – vetores e matrizes

- Outra forma de criar um vetor é por meio da função array do PHP:

```
<?php
$vetor = array(10,50,100,150,200);
echo $vetor[2].”<br>”;
?>
```

PHP – vetores e matrizes

- Funções básicas aplicadas aos vetores:
  - **Count()** – Informa quantos elementos o array possui.

```
echo count($vetor);
```

  - **Sizeof()** – idêntica à count, retornará um valor inteiro contendo o número de elementos de um array.

```
echo sizeof($vetor);
```

  - **Reset()** – Coloca o índice para o primeiro elemento do array e retorna o valor deste elemento.

```
echo reset($vetor);
```

PHP – vetores e matrizes

- Funções básicas aplicadas aos vetores:
  - **End()** – Coloca o índice para o último elemento do array e retorna o valor deste elemento.

```
echo end($vetor);
```

  - **Prev()** – A partir da posição do índice, desloca-se para o elemento anterior e retorna o valor deste. Caso esteja no primeiro elemento, o resultado será vazio.

```
echo prev($vetor);
```

PHP – vetores e matrizes

- **Next()** – A partir da posição do índice, avança para o próximo elemento e retorna o valor deste. Caso esteja no último elemento, o resultado será vazio.

```
echo next($vetor);
```

- **Pos()** – Retorna o conteúdo do elemento atual do array.

```
echo pos($vetor);
```

- **Key()** – Retorna o índice do elemento atual do array.

```
echo key($vetor);
```

PHP – vetores e matrizes

- **Sort()** – Coloca os elementos em ordem crescente.

```
sort($vetor);
echo $vetor[0].”|”.$vetor[1].”|”.$vetor[2];
```

- **Rsort()** – Coloca os elementos em ordem decrescente.

```
Rsort($vetor);
echo $vetor[0].”|”.$vetor[1].”|”.$vetor[2];
```

PHP – *function*

- Sintaxe:

```
Function nome_da_função (parâmetros)
{
...linhas de código da função
[return<expressão>]
}
```

PHP – *function*

- Exemplo

```
function comissao($valor)
{
$valor = $valor * 0.06;
Return $valor;
}
echo “Paulo vendeu R$ 25000 sua comissão é: R$ “ .
   comissao(25000);
echo “<br>”;
echo “Ana vendeu R$ 34540 sua comissão é: R$ “ .
   comissao(34540);
```

PHP – *function*

```
<?php
function clubes()
{
$clubes[] = “Grêmio”;
$clubes[] = “Palmeiras”;
$clubes[] = “Flamengo”;
$clubes[] = “Bahia”;
return $clubes;
}
$nomes = clubes();
for ($i=0; $i<sizeof($nomes); $i++)
{
echo $nomes[$i]. “<br>”;
}
?>
```

PHP – *function* passagem de parâmetros

- Quando passamos uma variável como argumento para uma função, por padrão estamos apenas passando o valor dela, e isso faz com que as alterações realizadas dentro da função não se reflitam sobre a variável quando terminar a execução da função. =>PASSAGEM POR VALOR

- Mas existem situações em que se quer que a variável a ser passada como argumento seja alterada conforme as alterações feitas durante a execução da função. Este tipo de passagem requer que seja colocado o símbolo & antes do nome da variável.

  => PASSAGEM POR REFERÊNCIA

PHP – *function* passagem de parâmetros

```
<?php
function dobro($valor)
{
$valor = 2 * $valor;
}
function duplica(&$valor)
{
$valor = 2 * $valor;
}
$valor = 5;
dobro ($valor);
echo $valor. “<br>”;
duplica($valor);
echo $valor;
?>
```

Exibe 5

Exibe 10 - referência

PHP – *function* passagem de parâmetros

- Há uma característica do PHP que permite a definição de valores-padrão. Se uma função possui determinado parâmetro e no momento da chamada esse parâmetro não é enviado, podemos utilizar valores-padrão.

- Para definir estes valores, basta colocar um operador de atribuição após o parâmetro definido na função, seguido pelo valor que deve ser considerado o padrão.

PHP – *function* passagem de parâmetros

```
<?php
function teste($time, $titulo = “Campeão
  Mundial”)
{
echo “O $time é $titulo<br>”;
}
teste(“Palmeiras”, “Campeão Taça Itália”);
teste(“Corinthians”);
?>
```

PHP – *function*
funções recursivas

- São aquelas que chamam a elas mesmas:

```
<?php
function lixo($valor)
{
if ($valor!=0)
{
echo “Foi chamada a função lixo passando o
  valor $valor<br>”;
lixo($valor-1);
}
}
lixo(7);
?>
```

PHP – vetores e matrizes exercício

- Usando matrizes para construir a página "Prato do Dia"
- Construir uma página que mostre para cada dia da semana o prato do dia e seu referido preço, conforme a tabela:

Elemento	Dia Semana	Prato Dia	Preço
0	Domingo	Lasanha	R$ 12,20
1	Segunda-feira	Frango	R$ 10,00
2	Terça-feira	Arroz à grega	R$ 9,40
3	Quarta-feira	Feijoada	R$ 11,20
4	Quinta-feira	Nhoque	R$ 8,50
5	Sexta-feira	Bacalhau	R$ 15,20
6	Sábado	Feijão branco	R$ 10,00

Mais exercícios

1. Lista de exercícios – Aula 4

www.fct.unesp.br/docentes/dmec/olivete/web/web_trabalhos_praticos.html