PHP e MySql

Sérgio S. Clemente Filho
sscf@cin.ufpe.br

PHP e MySql

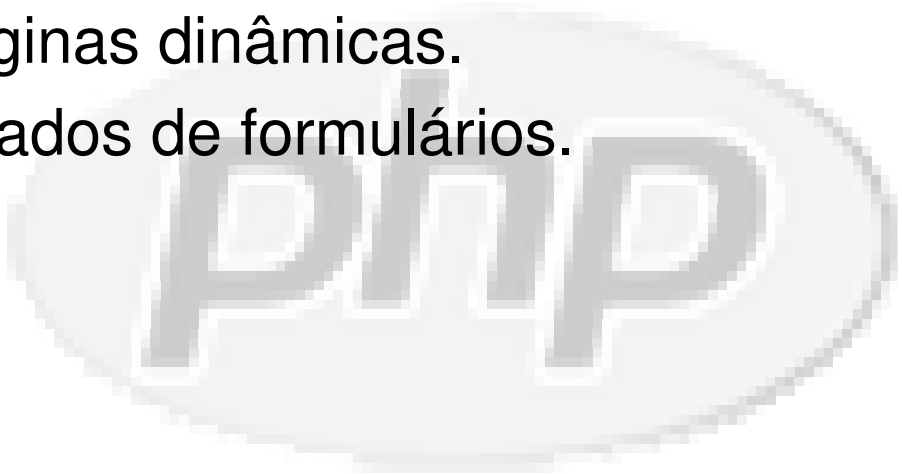
- Objetivo do curso
  - Tornar cada um capaz de fazer as seguintes aplicações:
    - Fotolog
    - Guest book
    - Site de comércio eletrônico
    - Site de pesquisa

Script x Linguagem de programação

- Compilado x Interpretado
- Tipagem estática x dinâmica
- Desempenho x Portabilidade

O que é e porque PHP? (1/2)

- Acrônimo recursivo: PHP Hypertext Processor.
- Server Side Scripting Language.
- Gerar páginas dinâmicas.
- Coletar dados de formulários.



O que é e porque PHP? (2/2)

- Roda em muitas plataformas.
- Facil aprendizado.
  - Flexibilidade de se alternar entre paradigmas de programação.
- Free e Open source.
- Suporta uma ampla gama de bancos de dados e padrões.
- ...

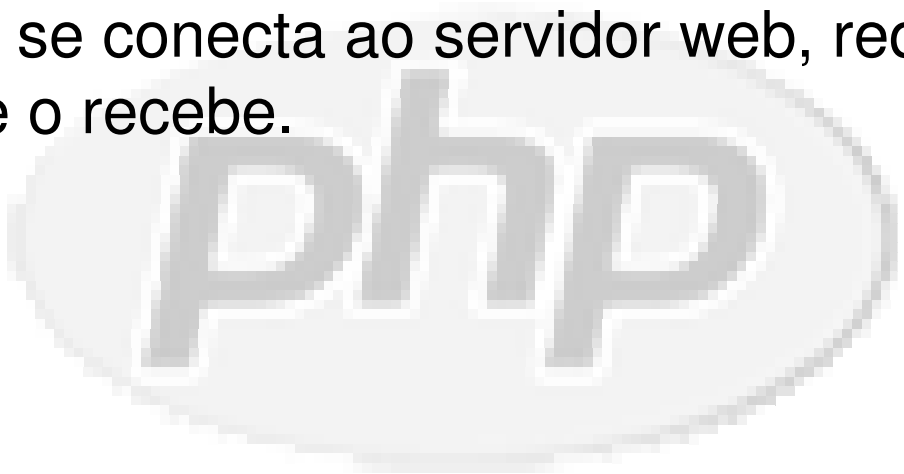
Trecho de código fonte do PHP Engine

```
ZEND_API zend_object_value
   zend_objects_new(zend_object **object,
   zend_class_entry *class_type TSRMLS_DC)
{
   zend_object_value retval;

   *object = emalloc(sizeof(zend_object));
   (*object)->ce = class_type;
   retval.handle = zend_objects_store_put(*object,
   (zend_objects_store_dtor_t)
   zend_objects_destroy_object,
   (zend_objects_free_object_storage_t)
   zend_objects_free_object_storage, NULL TSRMLS_CC);
   retval.handlers = &std_object_handlers;
   (*object)->in_get = 0;
   (*object)->in_set = 0;
   return retval;
}
```

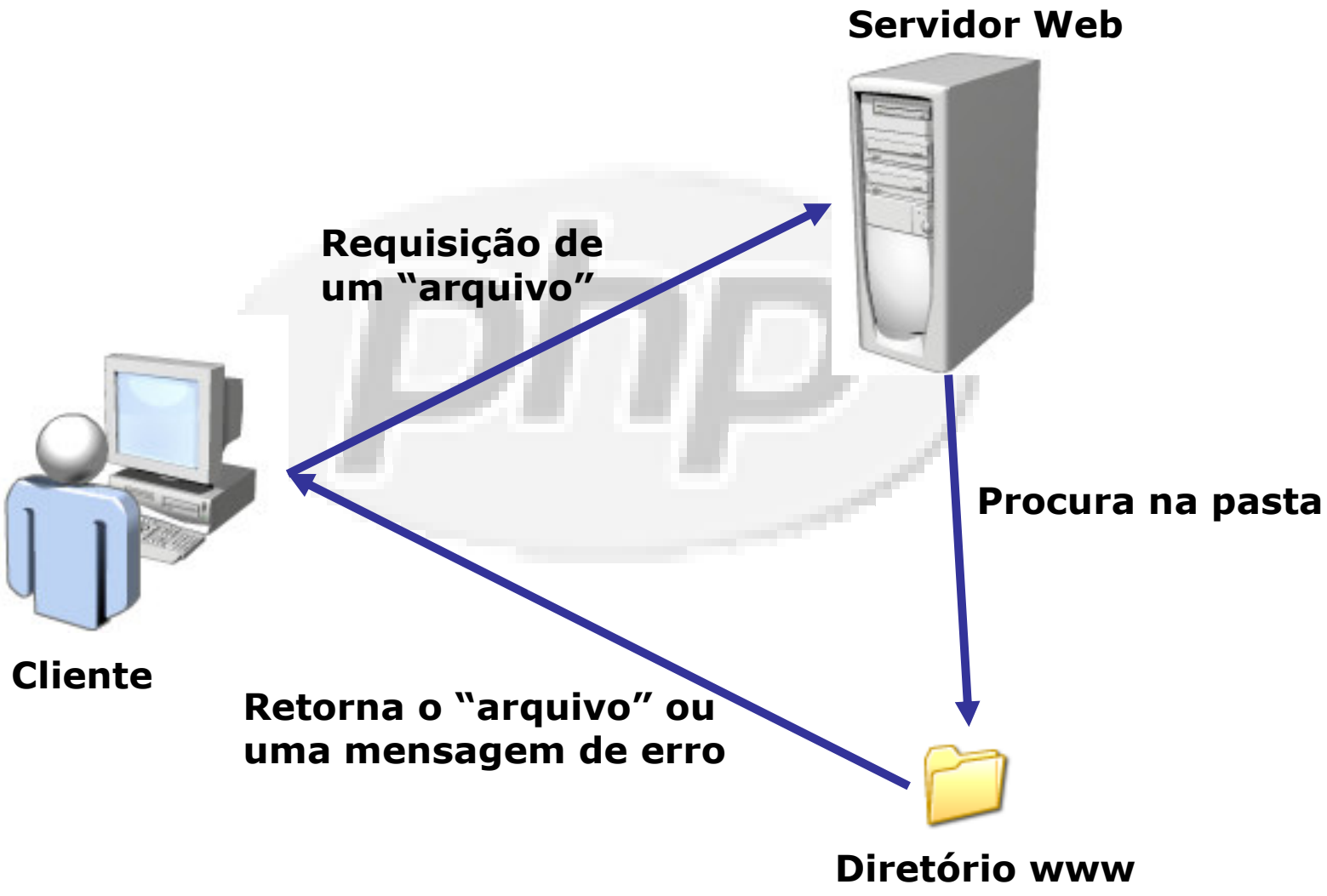
Fonte: www.php.net

Servidor Web

- “Servidor de arquivos”
  - Tem como função retornar um “arquivo”.
- O browser se conecta ao servidor web, requisita o “arquivo” e o recebe.

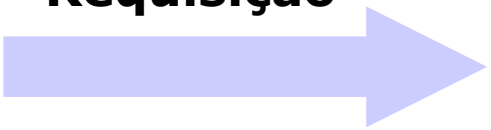


Servidor Web



Requisição & Resposta

Requisição



```
GET /index.html HTTP/1.1
From: sscf@cin.ufpe.br
User-Agent: IE/5.0
```

header (metadados)

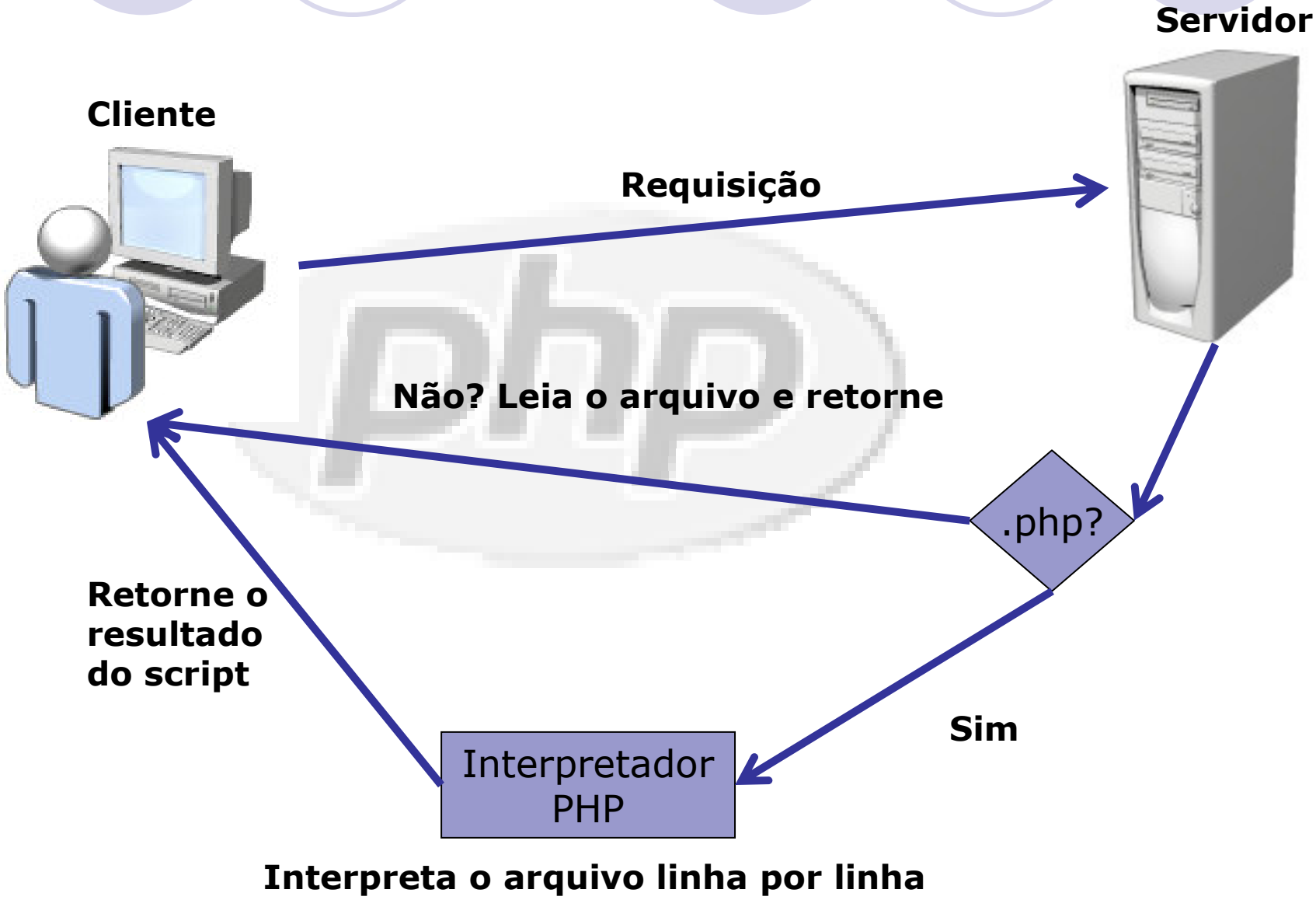
```
HTTP/1.1 200 OK
Date: Fri, 20 Apr 2004 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354
```

Resposta

corpo (arquivo)

```
<html>
<body>
<h1>Curso de PHP</h1>
...
</body>
</html>
```

Servidor Web e PHP



Código PHP mais simples possível

```
<html>
<head>
<body>
Hello World!
</body>
</html>
```

- É um script em php (pode ter a extensão *.php*)
- Não apresenta conteúdo dinâmico

Como é o codigo PHP?

- Código HTML + Código PHP
- O PHP é delimitado por tags iniciais e finais que lhe permitem pular pra dentro e pra fora do "modo PHP".

Forma canônica

```
...
<?php
    comandos;
?>
...
```

Forma abreviada

```
...
<?
    comandos;
?>
...
```

Hello World

```
<HTML>
<HEAD>
<TITLE>Hello</TITLE>
</HEAD>
<BODY>
<?php
    echo("Hello world!");
?>
</BODY>
</HTML>
```

Saída correspondente

Saída no browser

Hello world!

Código HTML gerado

```
<HTML>
<HEAD>
<TITLE>Hello</TITLE>
</HEAD>
<BODY>
Hello world!
</BODY>
</HTML>
```

Exercícios

- Exercício
  - Faça os exercícios 1, 2 e 3

O que eu preciso pra rodar PHP?

- Servidor Web (Apache, IIS, ...)*
  - http://www.apache.org
- Interpretador PHP (Atualmente na versão 5)*
  - http://www.php.net
- Banco de dados (MySql, Postgresql, Oracle, ...)
  - http://www.mysql.com

phpdev

- Tríade
  - Apache
  - MySql
  - PHP
- Plataforma suportadas atualmente
  - Win32
- Página do projeto
  - http://sourceforge.net/projects/phpdev5
- Free (GNU)

Exercício

- Baixar o phpdev

http://prdownloads.sourceforge.net/phpdev5/phpdev4_5NT.exe?download

- Descompactar em *c:\temp\phpdev*
- Configurar o *apache\conf\httpd.conf*
  - Substituir *c:\phpdev* por *c:\temp\phpdev*
- Rodar o *phpdev_start.bat*

PHP com Apache

- Normalmente PHP é instalado no apache
  - Pode rodar como módulo. (DLL)
    - Mais eficiente.
  - Pode rodar como CGI. (EXE)
    - Mais vulnerável.
- Linux & Apache & MySql & PHP quando utilizados conjuntamente são muito eficientes, além de ser free. (LIMP)

Pontos importantes

- Facil alternar entre o “modo php” e o “modo html”.
- Um arquivo php é basicamente um arquivo html (parte estática) com tags de php para gerar a parte dinâmica.

Funcionamento do Interpretador

Funcionamento do Interpretador

- Por default o interpretador vai varrer o arquivo no modo HTML. (Imprime o que é lido)
- Quando encontrar uma tag de inicio, entra no modo php.
- Quando no modo php, interpreta linha por linha e gera as saidas do script.
- Quando encontra a tag de fechamento, volta ao modo HTML.

Segundo exemplo

```
<?php
if ($expression) {
    ?>
     <strong>Isso é verdadeiro.</strong>
    <?php
} else {
    ?>
     <strong>Isto é falso.</strong>
    <?php
}
?>
```

Ponto e vírgula

- Em php precisamos colocar ponto e vírgula depois de cada instrução

```
<?php
      echo "ae, ";
      echo "td ";
      echo "blz?" // última instrução
?>
```

Inserindo comentários

```
<?php
    echo "Isto é um teste"; //Comentário de uma linha
/* Isto é um comentário de mais de uma linha
        e aqui temos outra linha */
    echo "Isto é um outro teste";
    echo "O último teste"; #Comentário no estilo Unix shell
?>
```

Comandos echo e print

- Server para “escrever” algo na saída HTML. (Browser)

```php
<?php
    echo("<h1>Introdução</h1>");
    print("<p>bla bla</p>");
    echo("Autor: Fulano")
?>
```

Variáveis

- Um espaço na memória para armazenar dados.
- São case sensitive.

  i.e. $nome e $Nome representam variáveis diferentes.
- Começam com *$* e pode conter letras, dígitos e sublinhados.
- O primeiro caractere não pode ser um número.

Variáveis

```
$idade $_nome $i
```
Nomes válidos

```
$3idade $primeiro-nome
```
Nomes inválidos

- Em PHP não é preciso declarar o tipo de uma variável antes de usá-la (Tipagem dinâmica).

```php
<?php
    $str = "José"; // $str armazena o tipo string.
    $i = 12; // $i armazena o tipo inteiro.
    $f = 3.1415; // $f armazena o tipo ponto flutuante.
?>
```

Variáveis

Tipo	Descrição
Inteiro	Um inteiro é um número do conjunto Z = {..., -2, -1, 0, 1, 2, ...}.
Pontos Flutuante	Números reais, como 4.12
Strings	Seqüência de caracteres
Booleano	Um booleano expressa um valor de verdade. Ele pode ser **TRUE** ou **FALSE**.
Array	Representa coleções
Objetos	Instancias de classes definidas.
Resource	Representa um recurso externo.
NULL	Representa uma variável sem valor.

Inteiro

- Representam os números inteiros da matemática
- Variam de -2.147.483.648 à 2.147.483.647

```php
<?php
    $a = 1; # número decimal
    $a = -2; # um número negativo
    $a = 0123; # número octal (83 em decimal)
    $a = 0xF; # número hexadecimal (15 em decimal)
    $a = 7*12 + 6/3; # saída: 86
?>
```

Pontos Flutuantes

- Possuem uma parte inteira e uma parte fracionária.
- Tamanho depende da plataforma.

```
<?php
    $a = 1.234;
    $b = 1.2e3;
    $c = 7E-10;
?>
```

- Cuidado o seguinte script gera a saída 1

```
<?= floor((0.1+0.7)*10) == 7 ?>
```

Strings

- (1 byte por caractere): Não suporta diretamento unicode.
- Podem ser delimitadas por “” (aspas duplas), ‘’ (aspas simples) e `` (crase).

Strings

- Delimitadas por “” (aspas duplas).
  - As variáveis internas são expandidas.

```
<?php
     $id = 12;
     $query = "SELECT * FROM FUNCIONARIO WHERE ID=$id";
     echo($query);
?>
```

Saída

SELECT * FROM FUNCIONARIO WHERE ID=12

Strings

- Delimitadas por ‘’ (aspas simples)
  - Variáveis internas não são expandidas.

```
<?php
     $id = 12;
     $query = 'SELECT * FROM FUNCIONARIO WHERE ID=$id';
     echo($query);
?>
```

Saída

SELECT * FROM FUNCIONARIO WHERE ID=$id

Strings

- Delimitadas ` ` (acentos graves)
  - Executa o conteúdo como um comando do shell e retorna a saída deste.

```
<?php
    $a = `time`;

    echo $a;
?>
```

Saida:

Hora atual: 16:00:57,61 Digite a nova hora:

Strings

- Caracteres especiais para string

Seqüência	Significado	“	‘
\n	Nova linha	OK	ERRO
\r	Retorno de carro	OK	ERRO
\t	Tab horizontal	OK	ERRO
\\	Barra invertida	OK	ERRO
\$	Cifrão	OK	ERRO
\’	Aspas simples	ERRO	OK
\”	Aspas duplas	OK	ERRO

Exercício

- Faça os exercícios 4 e 5

Strings

- Acessando um caractere da string.
- Coloca-se entre chaves o índice do caractere desejado (começando em 0).

```php
<?php
      $str = "abc";
      echo $str{1};
      echo "<br>";
      $str{1} = '2';
      echo $str;
?>
```

Saída gerada:

```
b
a2c
```

Strings

- Concatenação de strings

```php
<?php
    $str1 = "ab";
    $str2 = "c";
    $concat = $str1 . $str2;
?>
```

Strings

- Conversão de strings em inteiros
  - Tenta converter da esquerda para a direita, se não conseguir evolui para 0.

```
<?php
    $foo = 1 + "10.5";          // $foo é float (11.5)
    $foo = 1 + "-1.3e3";        // $foo é float (-1299)
    $foo = 1 + "bob-1.3e3";     // $foo é integer (1)
    $foo = 1 + "bob3";          // $foo é integer (1)
    $foo = 1 + "10 Small Pigs";   // $foo é integer (11)
    $foo = 4 + "10.2 Little Piggies"; //$foo é float(14.2)
    $foo = "10.0 pigs " + 1;      // $foo é float (11)
?>
```

Booleano

- Representam os conceitos de verdadeiro e falso.
- Use as palavras chave **TRUE** ou **FALSE**. Ambas são insensitivas ao caso.

```
<?php
    $foo = True; // assimila o valor TRUE para
$foo
?>
```

Booleano

- Normalmente utilizado em estruturas de controle junto com os operadores.

```
<?
    $bool = 3 > 0; // $bool = TRUE;
    if ($bool == True) {
        echo ("Entrou no if!");
    } else {
        echo ("Entrou no else!");
    }
?>
```

Arrays

- Encapsula uma coleção de dados.
- Podem ser indexados por números inteiros ou strings.

Arrays

- Não é necessário inicializar arrays

```
<?php
    $produtos[0] = "Pneu";     /* Equivalente a
$produtos[0] = "Pneu" e implicitamente o array
foi criado */
    $produtos[1] = "Oleo";
    $produtos[2] = "Luva";
?>
```

Arrays

- Indexados por números

```
<?php
    $produtos = array("Pneu", "Oleo", "Luvas");
    echo $produtos[1];
?>
```

Arrays

- Construtor
  - Utilizado para criar arrays.

```
array( [chave =>] valor
    , ...
    );
```

chave : pode ser int ou string

valor : pode ser qualquer coisa

Arrays

- Não é necessário inicializar arrays

```
<?php
    $produtos[] = "Pneu";    /* Equivalente a
$produtos[0] = "Pneu" e implicitamente o array
foi criado */
    $produtos[1] = "Oleo";
    $produtos[] = "Luva";
?>
```

Arrays

- Arrays Associativos

```php
<?php
    $precos = array( "Pneu"=>100, "Oleo"=>10, "Luva"=>4);

    $precos["Capo"] = 300;

    echo $precos["Oleo"];
?>
```

Arrays

- Ordenando arrays

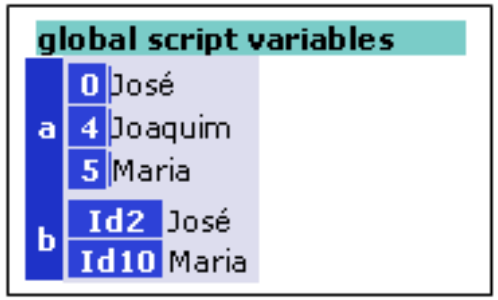
```php
<?php
    $produtos = array("Pneu", "Oleo", "Luvas");
    $precos = array( "Pneu"=>100, "Oleo"=>10, "Luva"=>41);

    sort($produtos);

    asort($precos); // ordena por preco
    ksort($precos); // ordena por nome
?>
```

Arrays

- Exemplo final:

```
<?php
    $a = array("José", 4=>"Joaquim", "Maria");
    $b = array("Id2" => "José", "Id10" => "Maria");
?>
```

- A memória fica mapeada como:



Resource

- Representa um recurso “externo”.
- Por exemplo:
  - Uma conexão com mysql.
  - Um arquivo aberto.
  - ...
- Exemplo:

```
<?php
    $handle = fopen (“curso.txt", "r");
?>
```

NULL

- Representa uma variável sem valor ou não inicializada.
- É insensível ao caso (null == Null).

- Exemplo:

```
<?php
    $raiz_quadrada = ($i > 0) ? sqrt($i) : null;
?>
```

Variáveis variáveis

- O conteúdo de uma variável pode virar o nome de uma variável.

```php
<?php
    $a = "gambiarra";

    $$a = "Isso é ceboso!";

    echo $gambiarra;
?>
```

Conversões entre tipos

- O nome do tipo desejado é escrito entre parênteses.

```
<?php
    foo = "123"; // $foo eh uma string
    $bar = (int) $foo; // $bar eh um inteiro
?>
```

Conversões entre tipos

- Semelhante a C, C++, porém bem mais poderoso...
- As conversões permitidas são:
  - (int), (integer) - molde para inteiro
  - (bool), (boolean) - molde para booleano
  - (float), (double), (real) - molde para número de ponto flutuante
  - (string) - molde para string
  - (array) - molde para array
  - (object) - molde para objeto

Conversões entre tipos

- No exemplo abaixo, entrará em todos os ifs.

```php
<?php
   $dez = 10;
   $str_dez = "10";
   $dez_e_meio = 10.5;
   $dez_e_meio_conv = (int) $dez_e_meio;
   if ($dez_e_meio_conv === $dez) {
      echo('$dez_e_meio_conv e $dez são idênticos.<br>');
   }
   $str1 = "$dez";
   $str2 = (string) $dez;
   if ($str1 === $str2) {
      echo('$str1 e $str2 são idênticos.<br>');
   }
   if ($dez == $str1) {
    echo('$str1 e $dez são equivalentes.<br>');
   }
?>
```

Conversões entre tipos

- Conceito de booleano em PHP
  - 0, para false.
  - 1, para qualquer coisa diferente de 0

```
<?php
    $foo = 10;          // $foo eh um inteiro
    $bar = (boolean) $foo; // $bar eh um booleano

    if ($foo) {
        echo("teste");
    }
?>
```

Operadores

- Aritméticos (*, /, +, -)
- Comparação (==, !=, <, >, <=, >=, ===)
- Concatenação (.)
- Lógicos (!, &&, ||, and, xor, or)
- Atribuição (=, .=, ++, --, +=, -=)

Estrutura de controle

- if
- else
- while
- do..while
- for
- foreach
- switch

If

- Ela permite a execução condicional de fragmentos de código.

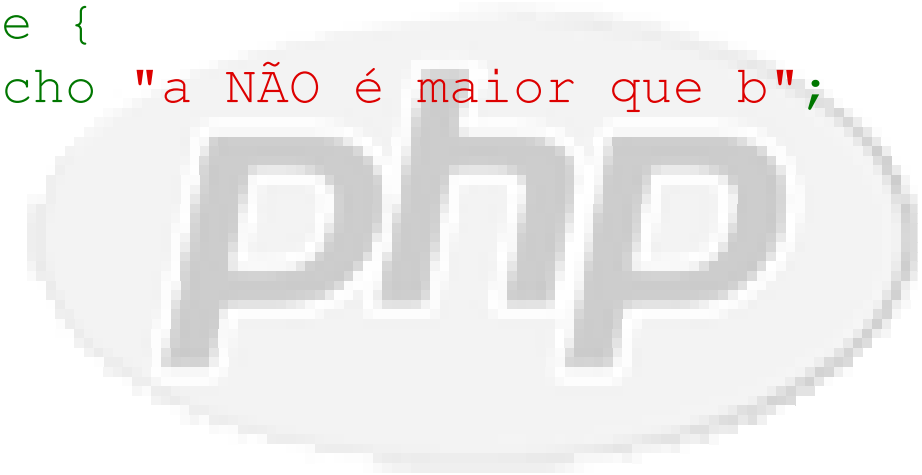
```
if (expressao) {
    instrucoes;
}
```

- Exemplo:

```
if ($a > $b) {
   echo "a é maior que b";
}
```

If else

```php
if ($a > $b) {
    echo "a é maior que b";
} else {
    echo "a NÃO é maior que b";
}
```



While

- Estrutura de repetição

```
while (expressao) {
    instrucoes;
}
```

- Exemplo 1:

```
$i = 1;

while ($i <= 10) {
    echo $i++;  /* o valor impresso será
                   $i depois do acréscimo
                   (post-increment) */
}
```

Do while

```php
$i = 10;
do {
    echo $i;
    $i++;
} while ($i < 10);
```

For

- Estrutura de repetição

```
for (expr1; expr2; expr3) {
    instrucoes;
}
```

- Exemplo 1:

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

For

- Exemplo2

```
$i = 1;
for (; ; ) {
    if ($i > 10) {
        break;
    }
    echo $i;
    $i++;
}
```

Exemplo3

```
for ($i = 1; ;  $i++) {
    if ($i > 10) {
        break;
    } else if ($i % 2 = 0) {
        continue;
    }
    echo $i;
}
```

Foreach

- Iterar sobre uma coleção.
- Existe somente no php4 em diante.

```php
<?php
    $array = array("a", "b", "c");

    foreach($array as $valor) {
        echo($valor);
    }

    $mapa = array("banana" => 1, "carne" => 10);

    foreach($mapa as $chave => $valor) {
        echo "<BR>";
        echo $chave . " -> " . $valor;
    }
?>
```

Foreach

- Exemplo2: Iterar sobre matrizes bidimensionais.

```
<?php
    $a[0][0] = "a";
    $a[0][1] = "b";
    $a[1][0] = "y";
    $a[1][1] = "z";

    foreach ($a as $v1) { // Para cada linha
        foreach ($v1 as $v2) { // Para cada elemento da
linha
            echo "$v2\n";
        }
    }
?>
```

Switch

- Similar a várias instruções IFs seguidas

```
swich (expressao) {
    case valor1:
    Instrucao1;
    break;
case valor2:
    Instrucao2;
    break;
case valor3:
    Instrucao3;
    break;
default:
    InstrucaoDefault;
}
```

Switch

- Exemplo

```
if ($i == 0) {
    echo "i igual a 0";
} elseif ($i == 1) {
    echo "i igual a 1";
} elseif ($i == 2) {
    echo "i igual a 2";
}
switch ($i) {
    case 0: echo "i igual a 0";
        break;
    case 1: echo "i igual a 1";
        break;
    case 2: echo "i igual a 2";
        break;
}
```

Exercício

- Faça o exercício 6.

Funções

- Mecanismo básico de estruturação.
- Não necessita declarar o retorno.
- Não é necessário declarar antes de usar.

```php
<?php
    $a = criarArray();

    function criarArray() {
        return array("Jose", "Maria");
    }
?>
```

Funções

- Funções com argumentos

```
<?php
    $n = calcular(3);

    function fatorial($num) {
        return ($num)*fatorial($num-1);
    }
?>
```

Funções

- Passagem de parâmetros por cópia.
- Não suporta sobrecarga. (Duas funções com o mesmo nome)
- O nome da função é insensível ao caso.

Exercício

- Faça o exercício 7.

Funções variáveis

- Assim como as variáveis variáveis, as funções também se beneficiar do mesmo mecanismo
- Exemplo:

```php
<?php
  $func = 'foo';
  $func();        // Chama foo()

  $func = 'bar';
  $func('test');  // Chama bar()

  function foo() {
     echo "Chamou foo()<br>\n";
  }
  function bar($arg = '') {
     echo "Chamou bar(); com argumento '$arg'.<br />\n";
  }
?>
```

include e require

- Instruções elementares que servem para inserir um arquivo externo.
- É bastante útil para dividir o código em vários arquivos, cada um com funcionalidades específicas.

include e require

- Inserir o código php.

```php
<?php
    include("util.php");

    $str = char2String('1'); // $str="Sim";
?>
```

util.php

```php
<?php
    function char2String($c) {
        return ($c == '1') ? "Sim" : "Não";
    }
?>
```

include e require

- Incluindo código Html.

```
<?php
    ...
    if (!$senhaValida) {
        require("SenhaInvalida.html");
    }
?>
```

SenhaInvalida.html

```
<HTML>
<HEAD></HEAD>
<BODY> Usuário e/ou senha inválido(s). </BODY>
</HTML>
```

include e require

- Se for definida alguma variável no arquivo incluso, ela estará disponível no script original e vice versa.

test.php

```php
<?php
        echo "A $color $fruit";
        include 'vars.php';
        echo "A $color $fruit";
?>
```

vars.php

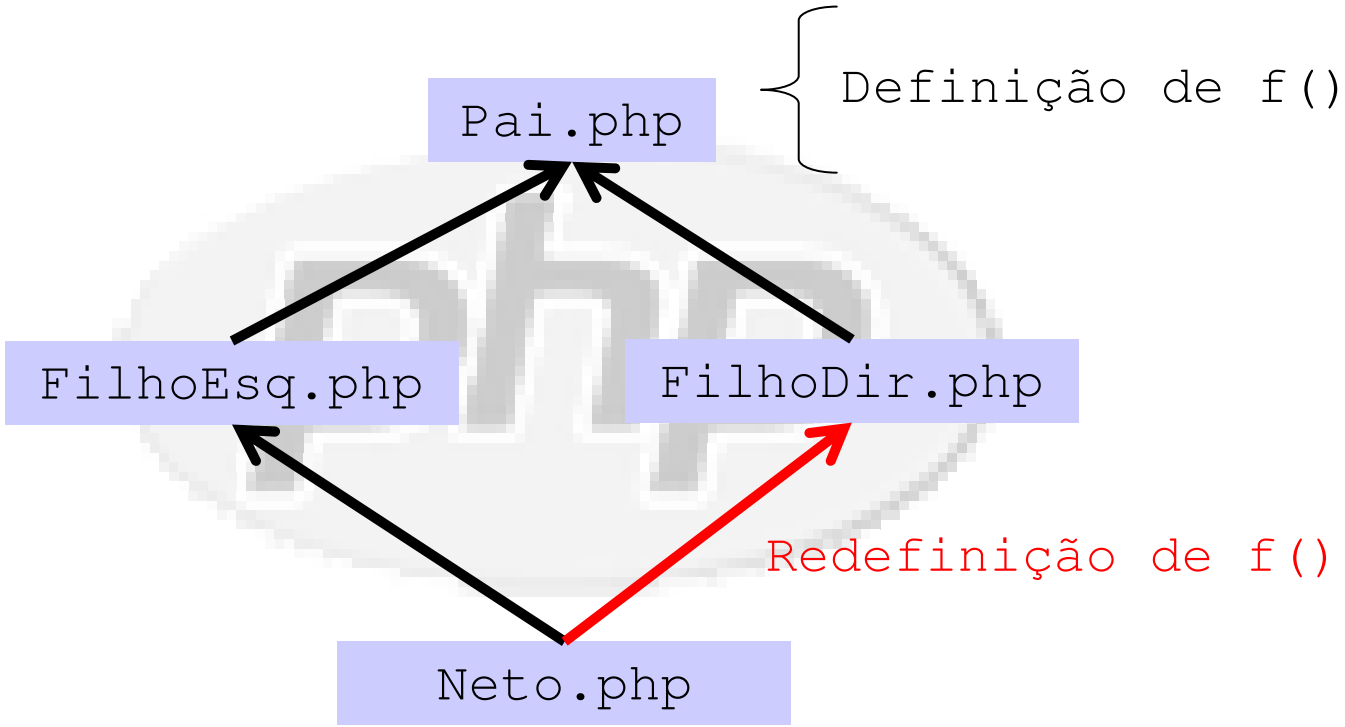
```php
<?php

$color = 'green';
$fruit = 'apple';

?>
```

include_once e require_once

```
include_once();
require_once();
```

- Inclui o arquivo apenas uma vez.
- Útil para evitar 2 definições de uma mesma função/classe, etc.
- Sempre use include_once ou require_once!

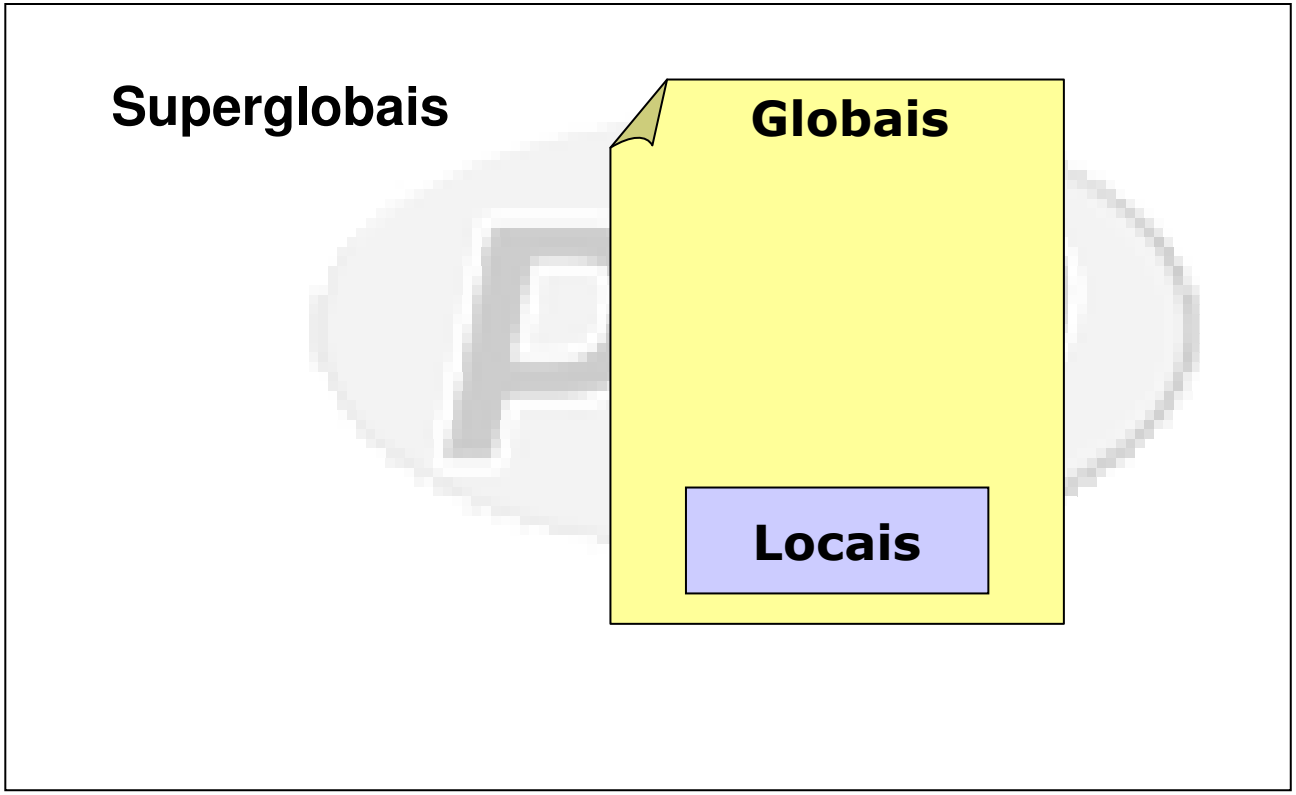
Caso em que é necessário o include_once e o require_once



Exercício

- Faça o exercício 8.

Escopo das variáveis



Escopo das variáveis

- Locais:
  - Declaradas dentro de uma função.
  - Seu escopo é restrito a função.

```
<?php
    function abc() {
        $a = 1;
    }

    abc();

    echo $a; /* Dá erro porque $a não foi definida no
escopo global*/
?>
```

Escopo das variáveis

- Global
  - Tem como escopo todo o arquivo ou sub-arquivos onde foi declarada.

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

    include "b.inc"; // $a pode ser referenciada

    echo $a;
?>
```

Escopo das variáveis

- Global

```php
<?php
    $a = 2;
    function imprime() {
        echo $a; // Em php 4x dá erro
    }
    echo $a; // Funciona pois a é global
    imprime();
?>
```

Escopo das variáveis

- Superglobals
  - Variáveis de ambiente (onipresentes).

- Principais
  - $_SESSION (Armazena dados de sessão)
  - $_POST, $_GET (Armazena dados de formulário)
  - $_FILES (Armazena dados vindos do upload)
  - $_SERVER (Informações de headers, ...)
  - $GLOBALS (Armazena variáveis globais)

$_SERVER

- Obtendo informações do cliente

```php
<?php
  $ipaddress = $_SERVER['REMOTE_ADDR'];
  $useragent = $_SERVER['HTTP_USER_AGENT'];
  $lang = $_SERVER['HTTP_ACCEPT_LANGUAGE'];

  echo $ipaddress . '<BR>';
  echo $useragent . '<BR>';
  echo $lang. '<BR>';
?>
```

Header

```
function header();
```

- Escreve no cabeçalho HTTP diretamente
  - Redirecionando para uma página

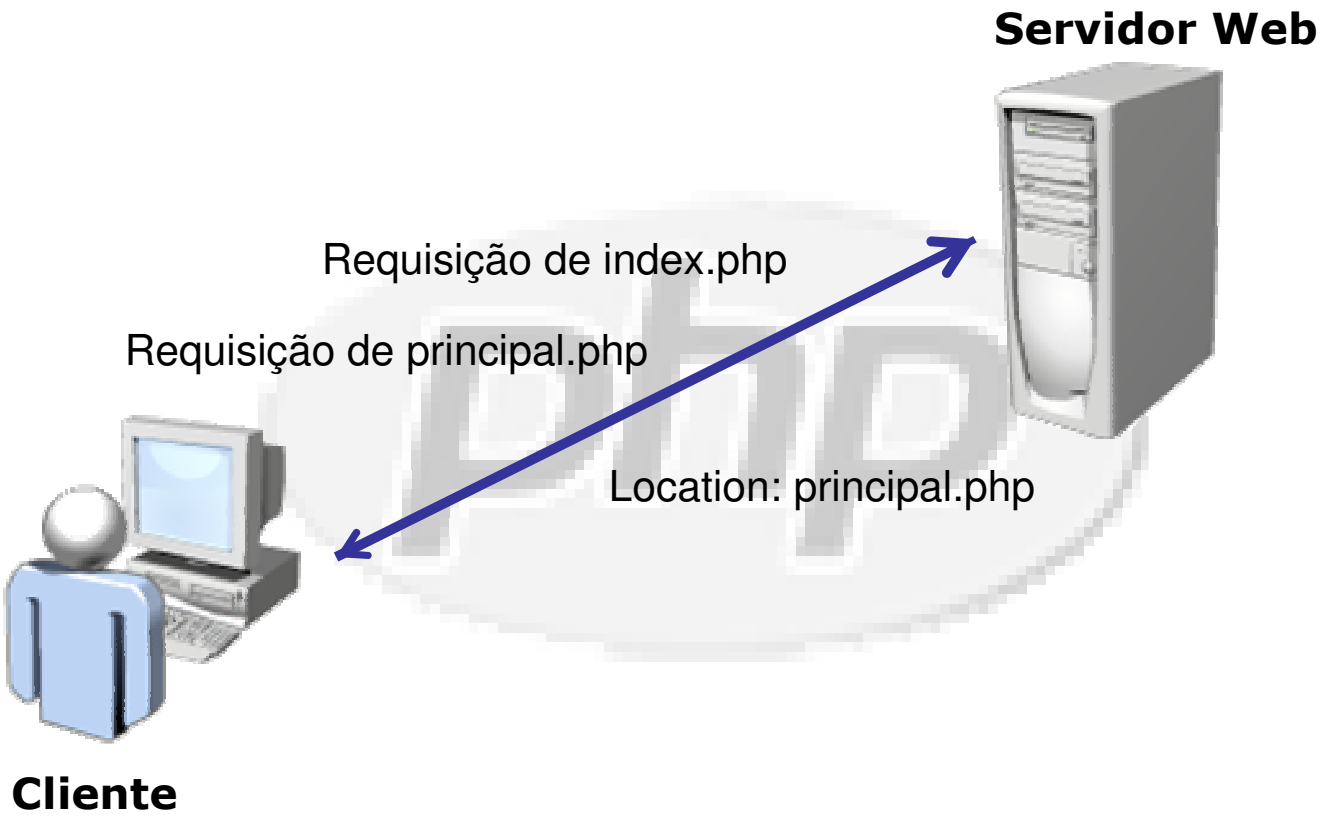
```
<?php
    header("Location: principal.php");
?>
```



```
GET /index.html HTTP/1.1
From: sscf@cin.ufpe.br
User-Agent: IE/5.0
Location: principal.php
```

Requisição

Funcionamento do redirecionamento



Header

- Simulando uma mensagem de erro.

```
<?php
    header("HTTP/1.0 404 Not Found");
?>
```

- Mudando o conteúdo da resposta

```
<?php
    header("Content-Type: image/jpeg");
?>
```

Header

- Um script php pode retornar uma imagem!

```php
<?php
  $conteudo = readfile($_GET['nome']);
  header("Content-Type: image/jpeg");
  echo $conteudo;
?>
```

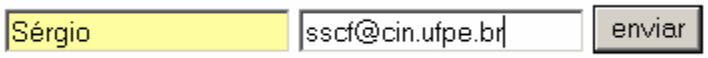
Exercício

- Faça o exercício 9 e 10

Formulários

- Permite passar dados para aplicação php.
- Cada elemento do form possui *nome, valor*.
- Cada elemento do form estará disponível através do *nome*, com o conteúdo *valor*.

Formulários



Ao clicar no botão:

Obrigado por enviar os dados sscf@cin.ufpe.br
Seu email é: Sérgio

```
<FORM ACTION= "processar.php" METHOD="post">
<INPUT TYPE=text NAME=email>
<INPUT TYPE=text NAME=nome>
<INPUT TYPE=submit NAME=botao VALUE=enviar>
</FORM>
```

processar.php

```
<?php
  echo "Obrigado por enviar os dados " . $_POST['nome'];
  echo "Seu email é: " . $_POST['email'];
?>
```

Exercício

- Faça o exercício 11 e 12

Componentes HTML

- Form
  - Todo form possui.
  - Tag “root”.

Arquivo que receberá os dados para processar

```
<form action="handlerCadastro.php" method="get">
</form>
```

Método Get ou Post

Componentes HTML

- TextField

José

```
<input name="nome" type="text">
```



```
$_GET['nome'] == "José";
```

- PasswordField

******

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



```
$_GET['senha'] == "123456";
```

Componentes HTML

- Button


```
<input type="submit" name="btn1" value="Enviar">
<input type="reset" name="btn2" value="Resetar">
```

Exercício

- Faça o exercício 13.

Componentes HTML

- CheckBox ☑

```
<input name="notificacao" type="checkbox" value="1">
```

```
$_GET['notificacao'] == "1";
```

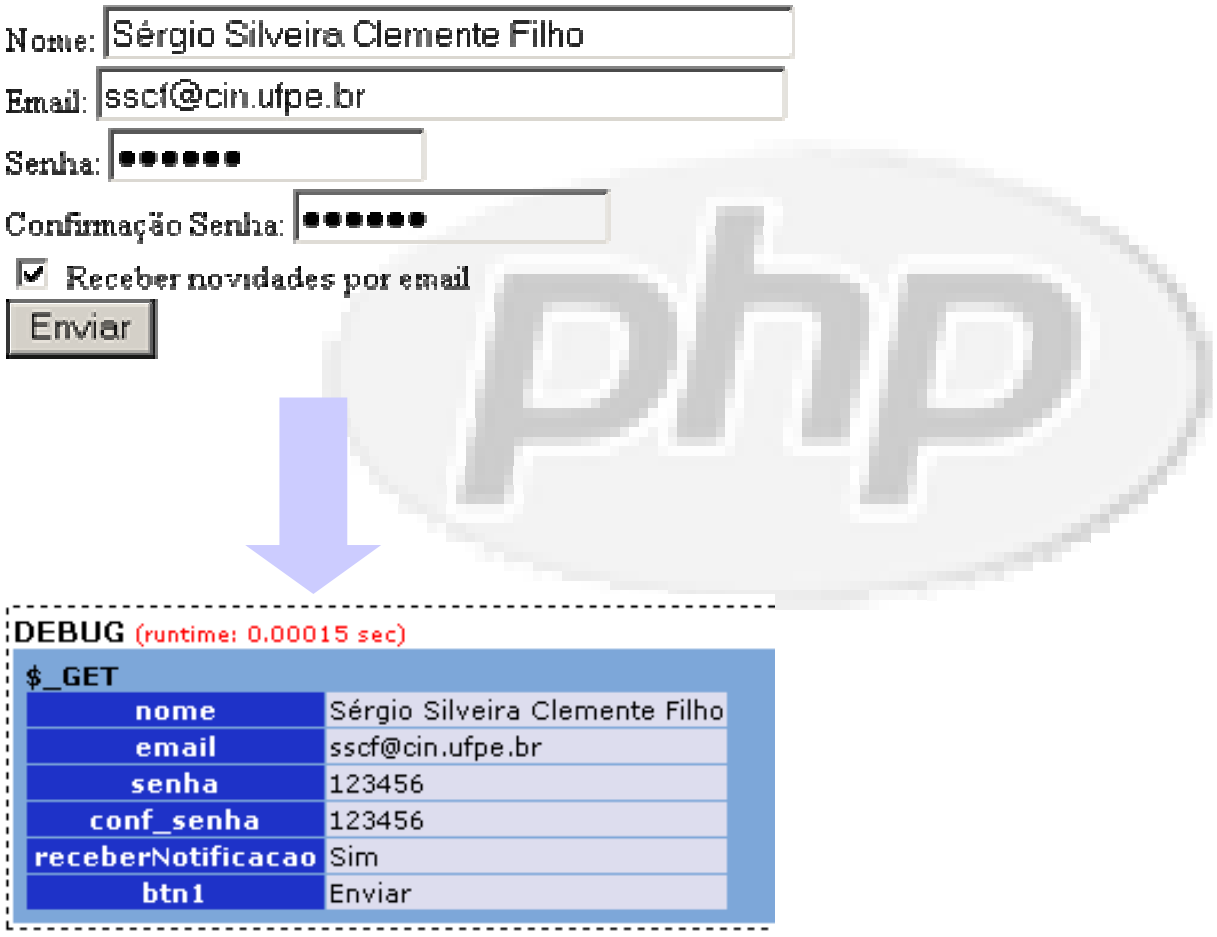
- Radio Button ◉

```
<input name="sexo" type="radio" value="m">
```

```
$_GET['sexo'] == "m";
```

Caso o o usuário não marque a opção, a variável estará indefinida!! (Restrição do protocolo HTTP)

Exemplo


Nome: Sérgio Silveira Clemente Filho
Email: sscf@cin.ufpe.br
Senha: ••••••
Confirmação Senha: ••••••
Receber novidades por email
Enviar
DEBUG (runtime: 0.00015 sec)
$_GET
nome Sérgio Silveira Clemente Filho
email sscf@cin.ufpe.br
senha 123456
conf_senha 123456
receberNotificacao Sim
btn1 Enviar

Exemplo


Nome: Sérgio Silveira Clemente Filho
Email: sscf@cin.ufpe.br
Senha:
Confirmação Senha:
Receber novidades por email
Enviar


DEBUG (runtime: 0.000159 sec)
$_GET
nome Sérgio Silveira Clemente Filho
email sscf@cin.ufpe.br
senha 123456
conf_senha 12345
btn1 Enviar

Componentes HTML

- Exemplo de action

```
<?php
    if (isset($_GET['notificacao']) {
        $notificacao = $_GET['notificacao'];    /* $notificacao
= "1"; */
        ...
    }

    if (isset($_GET['sexo'])) {
        $sexo = $_GET['sexo']; // $sexo = "m";
        ...
    }
?>
```

Componentes HTML

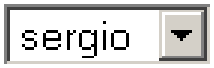
- TextArea

```
<textarea name="nome" cols="10" rows="5"></textarea>
```

```
$_GET['nome'] == "texto digitado";
```

Componentes HTML

- Select




```
<select name="logins[]" size="3" multiple>
    <option value="sscf">sergio</option>
    <option value="jts">jose</option>
    <option value="aas">antonio</option>
</select>
```

```
<select name="login">
    <option value="sscf">sergio</option>
    <option value="jts">jose</option>
    <option value="aas">antonio</option>
</select>
```

Componentes HTML

- Select
  - No exemplo anterior o action seria algo do tipo

```php
<?php
    ...
    $logins = $_POST['logins'];
    foreach($logins as $login) {
        ...
    }
?>
```

Exercício

- Faça o exercício 14.

Validação de campos

- Validar os campos é extremamente necessário.
- Usuários preenchem os campos distraídos ou maliciosamente.
- Precisamos garantir a robustez da aplicação!!
- A validação deve ser feita tanto no cliente (Javascript) quanto no servidor (PHP).
  - Na prática é realizada validação apenas no servidor, porém nunca apenas no cliente!

Validação de campos

- Exemplo
  - Formulário

Validação de campos

- Exemplo
  - Action

```php
<?php
	echo($_POST['nome']);

	echo("<br>");

	echo($_POST['habitos']);
?>
```

Validação de campos

- Exemplo:
  - Usuário entra com os “dados”.

Validação de campos

- Exemplo:
  - Action respectivo a entrada do usuário.

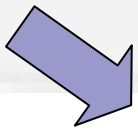
Exercício

- Faça os exercícios 15 e 16.

Componentes HTML

- Upload de arquivo
  - Formulário

```
<form enctype="multipart/form-data"
action="actionArquivo.php" method="POST">
<input type="hidden" name="MAX_FILE_SIZE" value="200">
Enviar este arquivo: <input name="userfile" type="file">
<input type="submit" value="Enviar Arquivo">
</form>
```



Componentes HTML

- Upload de arquivo
  - Action

```
<?php
  $uploaddir  = "c:\\temp\\uploads\\";
  $uploadfile = $uploaddir . $_FILES['userfile']['name'];
  print "<pre>";
  if (move_uploaded_file($_FILES['userfile']['tmp_name'],
$uploaddir . $_FILES['userfile']['name'])) {
     print "O arquivo é valido e foi carregado com
sucesso. Aqui esta alguma informação:\n";
  } else {
     print "Possivel ataque de upload! Aqui esta alguma
informação:\n";
  }
?>
```

Exercício

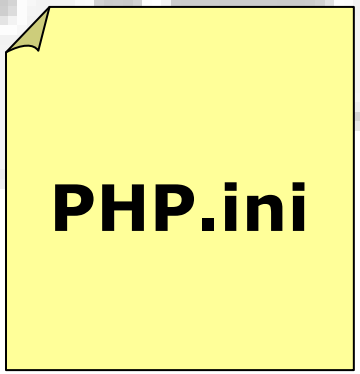
- Faça a questão 17

Get X Post

- Get
  - Maior número de browsers suportados.
  - Os parâmetros são enviados na própria url.
  - A url seria algo do tipo:
    processar.php?nome=jose&email=mer@mer.com
- Post
  - Os parâmetros são enviados encondidos junto com a requisição.
  - Não mostra os campos escondidos.
  - É mais elegante.

PHP.ini

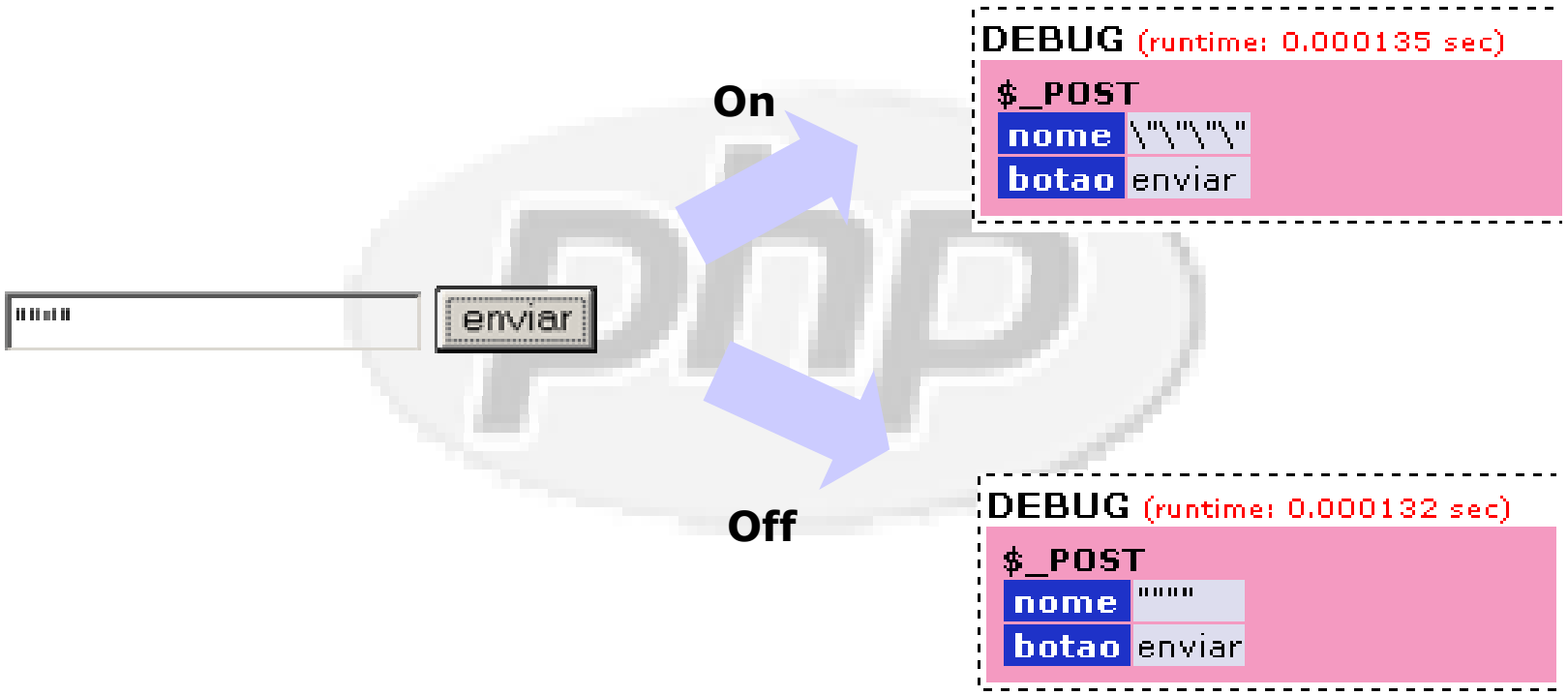
- Possui as principais configurações do interpretador PHP.
- É lido quando o PHP é iniciado.
- Normalmente é localizado na pasta c:\windows



PHP.ini

- Principais diretivas:
  - *register_globals*
  - *short_open_tag*
  - *include_path*
  - *SMTP*
  - *sendmail_from*
  - *Extension*
  - *upload_max_filesize*
  - *magic_quotes_gpc*

Magic Quotes



Enviando Email

```
bool mail ( string para, string assunto, string mensagem [,
string cabecalhos_adicionais])
```

- Retorna true se o email foi enviado, false caso contrário.

- Exemplo:

```
mail("destino@mer.com", "Assunto", "Ae mermao blz?");
```

Enviando Email

para : Nome do(s) destinatario(s).

assunto : Assunto da mensagem.

mensagem : Corpo da mensagem.

cabecalhos_adicionais :

- From
- BCC
- Reply-to
- Content-Type
- X-mailer

parametros_adicionais :

Criando uma função mail

- Criando uma função mail

```
<?php
    function enviarEmail($de, $para, $assunto, $corpo) {
            mail($para, $assunto, $corpo, "From: $de\r\nReply-To:
$de") or die('Erro ao enviar email');
    }
?>
```

Exercício

- Faça o exercício 17.

Cookies

- Informações que ficam armazenadas no cliente.
- Tem a forma *nome, valor.*
- O programa manda armazenar no cliente para num futuro reavê-la.

Cookie:caxibrema@google.com

```
Ocupacao estudante
```

Cookies

Set-Cookie: *Ocupacao=estudante*

Resposta

Cookie: Ocupacao=estudante

Requisição

www.google.com

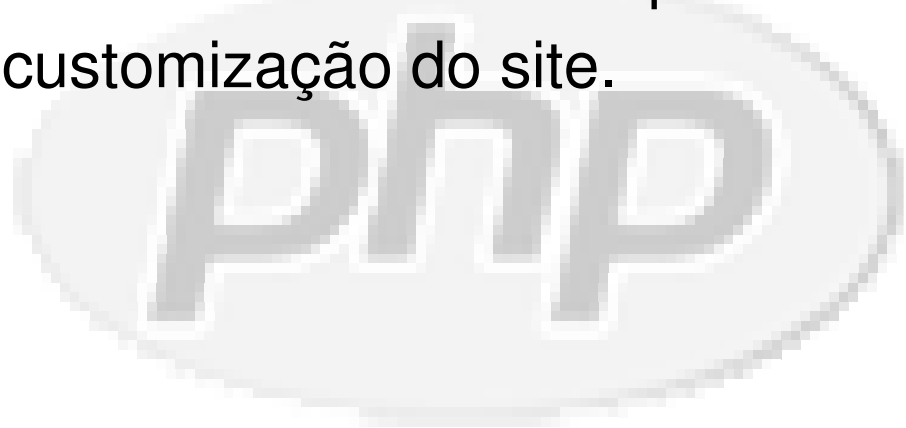
*Cookie:*google.com

Ocupacao estudante

```
$_COOKIE['Ocupacao'] == "estudante";
```

Cookies

- Os cookies trafegam no cabeçalho http.
- Cookies deixam a requisição maior, pois o cliente além da requisição manda os cookies daquele site.
- Permite a customização do site.



Cookies

- Enviando um cookie ao cliente

```
bool setcookie ( string nome [, string valor [, int
   tempoExpiracao]])

Nome : nome do cookie
Valor: valor do cookie
TempoExpiracao: tempo de vida do cookie
```

Precisam ser chamados antes de qualquer informação
seja impressa na tela. (Pois são enviados no cabeçalho do HTTP)

Cookies

Resposta

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1354
Set-Cookie: Ocupacao=estudante


<html>
<body>
<h1>Curso de PHP</h1>
...
</body>
</html>
```

Requisição

```
GET /index.html HTTP/1.1
From: sscf@cin.ufpe.br
User-Agent: IE/5.0
Cookie: Ocupacao=estudante
```

Cookies

- Exemplos:

```
<?php
    $value = 'aluno';

    setcookie ("Ocupacao", $value);
    /*expira no final da seção (Quando o browser fechar)*/

    setcookie ("Ocupacao",  $value,time()+3600);
    /* expira em uma hora */
?>
```

Cookies

- Quando um cliente requisita uma pagina na web, ele manda na requisição os cookies, as quais serão elementos da variável super global $_COOKIE.
- No exemplo anterior seria:

```
<?php
    if (isset($_COOKIE['Ocupacao']) &&
$_COOKIE['Ocupacao'] == "estudante") {
        enviarSpam();
    } else {
        efetuarDesconto();
    }
?>
```

Exercício

- Faça o exercício 18.

Sessões

- Pares *nome, valor* que o programa armazena no servidor, associados a um cliente que podem ser recuperados mais tarde.
- Permite manter o estado entre as páginas.
- Semelhante ao cookie, porém reduz o tamanho da requisição, pois apenas um identificador da sessão trafega na web.

Sessões

```
session_start();
```

- cria uma sessão ou
- resume a sessão atual baseada numa id de sessão sendo passada via um cookie ou uma variável GET.
- Essa função deve ser chamada antes de qualquer informação seja impressa na tela.

Sessões

- Guardando pares *nome, valor* na variável de sessão.

```php
<?php
    session_start();

    $_SESSION['nome'] = "José"; // Manda setar a variável
?>
```

Sessões

```
setCookie("PHPSESSID", "43b");
```

Requisição

Cookie: PHPSESSID=43b

www.google.com

*Cookie:*google.com

PHPSESSID 43b

Gera um número aleatório por exemplo '43b'.

```
session_start();

$_SESSION['nome'] = "José";
```

Nome José

Arquivo: sess_43b

Sessões

- Obtendo valores de volta

```php
<?php
    session_start(); // precisa ser chamada antes.

    echo "Seu nome é" .
        $_SESSION['nome']; /* recupera o valor que veio
junto com a requisição */
?>
```

Cookies x Seções

Cookies	Sessões
Necessita ficar trafegando junto com a requisição.	Apenas a Id do cliente trafega junto com a requisição. Toda a informação reside em um arquivo no servidor.
O cliente pode não aceitar cookie.	Caso o cliente não aceite cookie, a id do cliente pode trafegar via um campo get.
Evite utilizar cookies quando informações confidencias estiverem em jogo.	Sessões são ideais para armazenar login, senha e email de usuários quando os mesmo efetuam logon.
Cookies “vivem” por mais tempo	A Sessão “morre” quando você fecha o browser.

Aplicações

- Cookie
  - Evitar que um usuário vote mais de uma vez.
  - Contador de acessos do usuário.
- Sessões
  - Armazenar informações do usuário (login, senha, email,...) enquanto estiver checando email, ou comprando alguma coisa em um site.

Exercício

- Faça a questão 19.

Arquivos

- Principais funções
  - fopen();
  - fgets();
  - fwrite();
  - feof();
  - fclose();

Arquivos

- resource fopen(string arquivo, string modo);
  - Abre um arquivo especificado em arquivo.
  - Principais modos:
    - “r”: Somente leitura.
    - “w”: Somente escrita. Trunca caso já exista.
  - Exemplo:

```
<?php
     $handle = fopen ("info.txt", "r");
?>
```

Arquivos

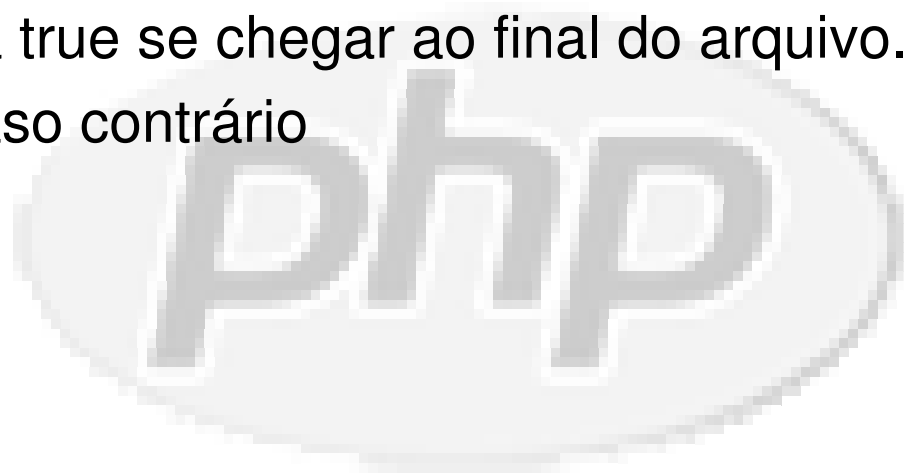
- fgets(resource arquivo[, int tamanho]);
  - Lê a próxima (última) linha.
  - Caso tamanho seja especificado, lê os próximos *tamanho* bytes.
  - Retorna false caso ocorra algum erro.

Arquivos

- int fwrite(resource arquivo, strint str);
  - retorna o número de bytes gravados ou FALSE em caso de erro.

Arquivos

- bool feof(resource arquivo);
  - Retorna true se chegar ao final do arquivo.
    false caso contrário



Arquivos

- bool fclose (resource arquivo);
  - Retorna true em caso de sucesso, false caso contrário.

Arquivos

- Exemplo de leitura:

```php
<?php
	$arq = fopen("teste.txt", "r");

	while (!feof($arq)) {
		$str = fgets($arq);
		echo $str;
		echo "<br>";
	}

	fclose($arq);
?>
```

Arquivos

- Exemplo de escrita:

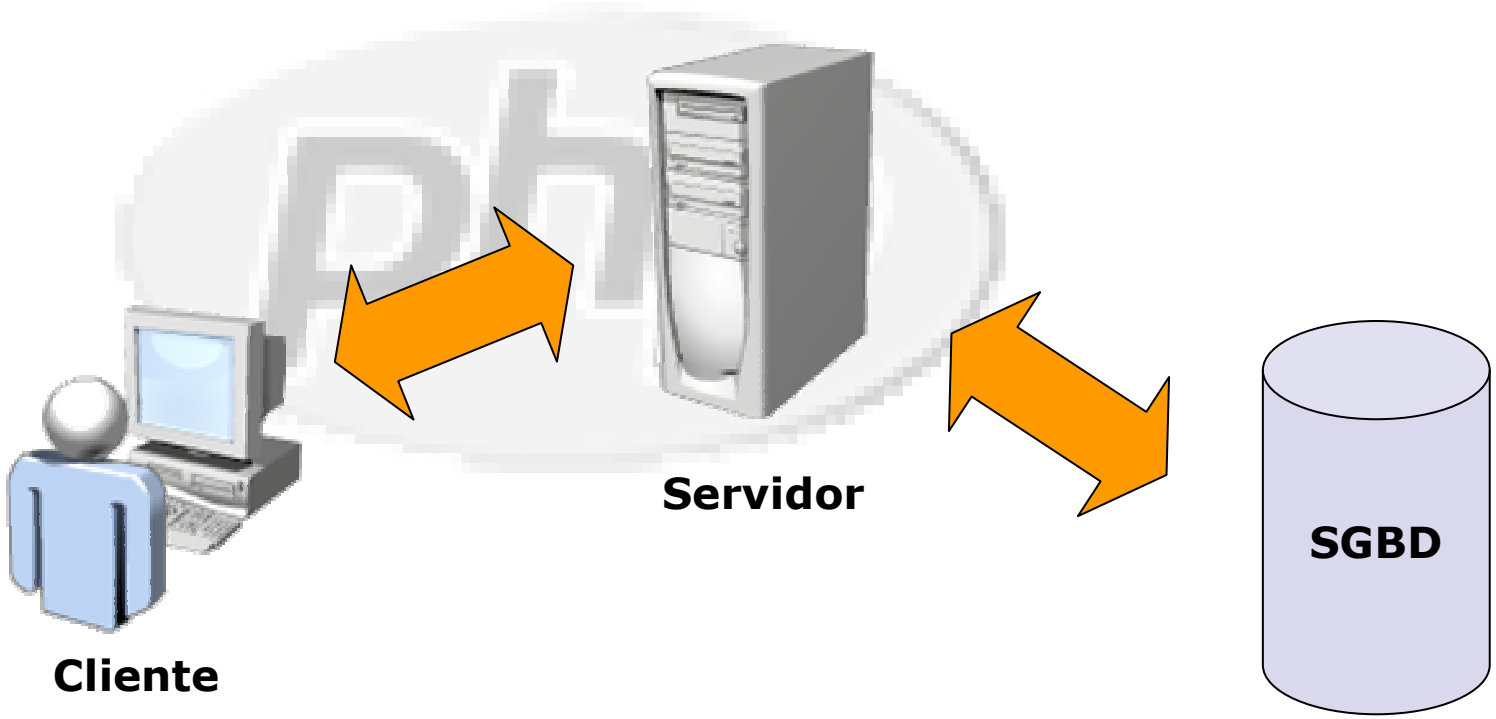
```php
<?php
	$arquivo = fopen("saida.txt", "w");
	fputs($arquivo, "ae vei\r\n");
	fputs($arquivo, "bla bla\r\n");
	fclose($arquivo);
?>
```

Exercício

- Faça a questão 20.

Conectividade com Banco de dados

- Porque?
  - Tornar mais dinâmico o conteúdo da página



SGBDR

- É um programa que armazena grandes quantidades de dados num formato de tabelas que é facilmente acessado via php.
- Um banco de dados é composto de uma ou mais tabelas, onde cada tabela é composta de uma ou várias colunas.

SGBD

- **Uma tabela de piadas**

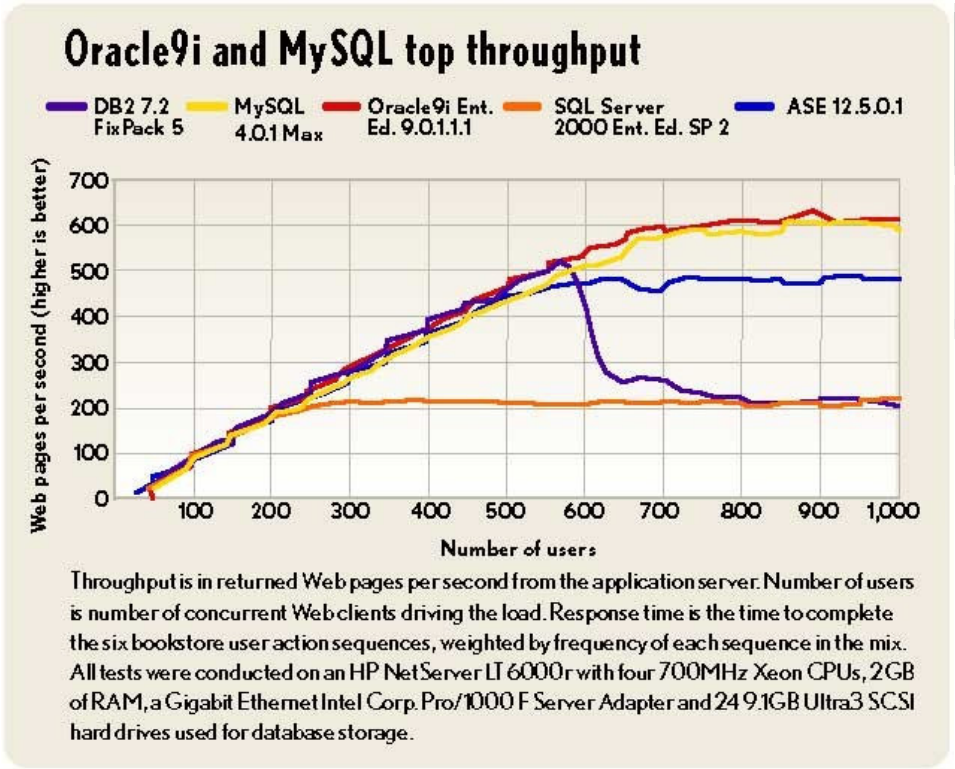
	Column ↓	Column ↓	Column ↓
	ID	**JokeText**	**JokeDate**
Row →	1	Why did the chicken...	2000-04-01
Row →	2	"Knock knock!" "Who's there?"	2000-02-22

SQL

- Structured Query Language.
- “Padrão” para interagir com banco de dados relacionais.
- Versão mais nova: SQL99.

MySql

- My Structured Query Language
- Trade-offs
  - Simples, fácil e extremamente veloz.
  - Suporta quase todos comandos do padrão Sql99



Fonte: www.mysql.com

Principais tipos de dados MySql

Nome	Característica	Exemplo
integer/int	Números inteiros	1,-12,141
Integer auto_increment	Número inteiros em sequência	1,2,3,4,5,6,...
double	Números reais	1.23, 3.14
varchar	Strings	‘abc’, ‘php’
text/blob	String longa	‘abracadabra’
time	'HH:MM:SS'	’12:11:04’
date	'YYYY-MM-DD'	‘1983-01-19’, ...
datetime	'YYYY-MM-DD HH:MM:SS'	‘1983-01-19 22:12:12’

Principais Comandos SQL

- Create Table
- Insert
- Select
- Update
- Delete

Create Table

- Cria uma tabela
  - Sintaxe

```
CREATE TABLE nome_tabela (
    nome_coluna tipo modificadores, ...
)
```

  - Exemplo

```
CREATE TABLE Jokes(
    ID INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    JokeText TEXT,
    JokeDate DATE NOT NULL
)
```

Insert

- Insere um registro em uma determinada tabela.
  - Sintaxe

```
INSERT INTO destino [(campo1, campo2,...)]
VALUES (valor1, valor2,...)
```

  - Exemplo

```
INSERT INTO Jokes (JokeText, JokeDate)
VALUES ('Why did the chicken cross theroad? To get to the
other side!',
'2000-04-01')
```

  - Vai inserir a piada com ID = 1, se executarmos o comando de novo irá inserir com ID = 2 e assim por diante...

Select

- Retorna um subconjunto de registros
  - Exemplo

```
SELECT ID, LEFT( JokeText, 20 ) ,
JokeDate
FROM Jokes


SELECT JokeText, JokeDate
FROM Jokes ORDER BY ID
```

Update

- Atualiza uma tabela
  - Sintaxe

```
UPDATE nome_tabela
SET nome_coluna = novo_valor
WHERE condicao
```

  - Exemplo

```
UPDATE Jokes
SET JokeDate = '1990-04-01'
WHERE JokeText LIKE '%chicken%'
```

Delete

- Deleta registro(s) de uma tabela
  - Sintaxe

```
DELETE FROM nome_tabela
WHERE condicao
```

  - Exemplo

```
DELETE FROM Jokes
WHERE JokeText LIKE '%chicken%'
```

Conectividade com SGBD

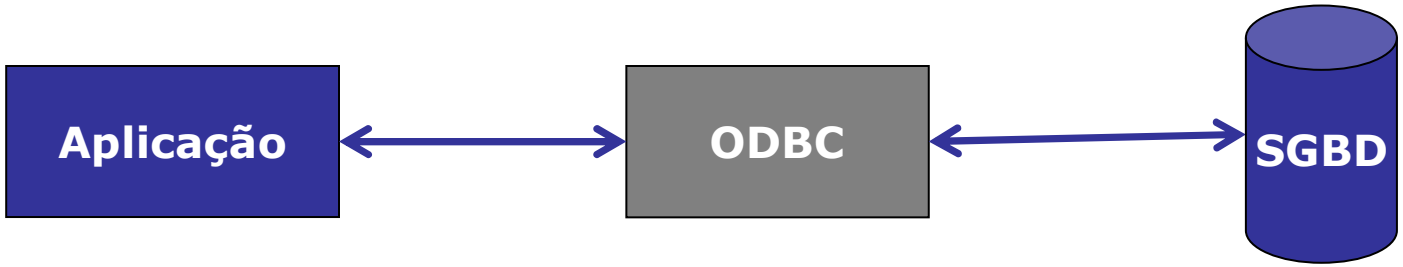
- Existe duas maneiras:
  - Driver nativo
  - ODBC

Driver nativo

- Fala a mesma “língua” do SGBD.
- Extremamente rápido.

ODBC

- Open Database Connectivity.
- Padrão para acesso a banco de dados.
- Abstrair o SGBD da aplicação.
- Traduz as chamadas genéricas (ODBC) em chamadas específicas do SGBD.
- Mais versátil.



Driver Nativo

- Principais funções

```
mysql_connect();
mysql_select_db();
mysql_query();
mysql_affected_rows();
mysql_insert_id();
mysql_num_rows();
mysql_fech_array();
```

Driver Nativo

```
resource mysql_connect(host, login, senha);
```

- Se conecta ao banco de dados.
- Retorna um identificador da conexão ou False se falhar

Exemplo:

```
<?php
    $con = mysql_connect("localhost", "user", "user");
?>
```

Selecionando a base de dados

```
bool mysql_select_db (string nome_base [, resource
id_conexao])
```

- Seleciona uma base de dados

Exemplo:

```
<?php
    $con = mysql_connect("localhost", "user", "user");
    mysql_select_db("nome_da_base");
?>
```

Executando Queries

```
resource mysql_query ( string sql [, resource
id_conexao])
```

- Executa uma query (Não deve conter ponto e vírgula)
- Em consultas select:
  - Retorna a id do resultado caso a execução com sucesso, caso contrário false.
- Em consultas insert, update, ...
  - True para sucesso, caso contrário false.

Executando Queries

Exemplo:

```
<?php
    $con = mysql_connect("localhost", "user", "user");
    mysql_select_db("nome_da_base");
    $resultado = mysql_query("SELECT * FROM usuario");
?>
```

```
<?php
    $con = mysql_connect("localhost", "user", "user");
    mysql_select_db("nome_da_base");
    mysql_query("UPDATE usuario SET nome='jose' WHERE
id=2");
?>
```

Executando Queries

```
int mysql_insert_id ( [resource
link_identifier])
```

- Recupera o ID gerado da operação insert anterior.

```
<?php
   mysql_query("INSERT INTO usuario (nome, endereco) values
('jose', 'avenida ...')");
   print("O ultimo registro incluído tem id %d\n" .
mysql_insert_id());
?>
```

Mostrando o resultado

```
int msql_fetch_array ( int id_resultado [, int
tipo_array])
```

- Carrega uma linha em um array.
- Retorna um ponteiro para o array caso haja mais registros, caso contrário retorna false.
- O segundo argumento pode assumir 3 valores
  - MSQL_NUM: Retorna um array indexado pelo índice da coluna.
  - MSQL_ASSOC: Retorna um array indexado pelo nome da coluna.
  - MSQL_BOTH : Ambos os anteriores. É a opção default.

Mostrando o resultado

- Exemplo

```
<?php
    $con = mysql_connect("localhost", "user", "user");
    mysql_select_db("nome_da_base");
    $resultado = mysql_query("SELECT nome,endereco FROM
usuario");

    while ($funcionario = mysql_fetch_array($resultado)) {
        echo $funcionario['nome'];
        echo $funcionario['endereco'];
    }
?>
```

Mostrando o resultado

```
int mysql_num_rows ( resource id_resultado)
```

- Retorna o número de linhas em um resultado

```
<?php
    $result = mysql_query("SELECT * FROM usuario");
    $num_rows = mysql_num_rows($result);
?>
```

Mostrando o resultado

```
int mysql_affected_rows ( [resource id_conexao])
```

- Devolve o número de linhas afetadas

```
mysql_query("DELETE FROM usuario WHERE nome like %jose%'");
print("Registros excluídos: %d\n" . mysql_affected_rows());
```

Exercício

- Faça as questões 21,22,23,24

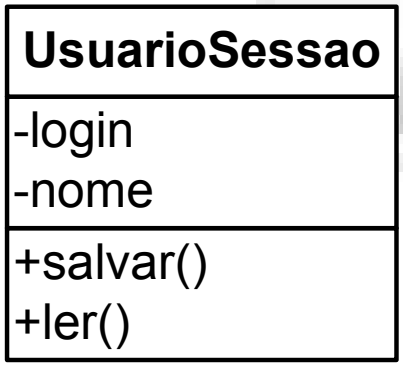
Objetos

- Necessidade de definir tipos mais complexos.
- Um objeto possui:
  - Comportamento: Operações que o objeto pode executar.
  - Estado: Informações sobre seu estado atual.
- Suportar hierarquia de tipos (Conta, Poupança)
- Exemplo: Minha conta, Minha bicicleta.

Classes

- É um modelo que define as variáveis e métodos comum a todos objetos de um determinado tipo.

- Exemplo: Classe para encapsular os dados de sessão.



Objetos em PHP

- PHP4:
  - Não possui um modelo de objetos perfeito. (Nojento?)
    - Não há encapsulamento (todos os atributos são públicos)
    - Não há interfaces nem exceções.
- PHP5:
  - Modelo de objetos totalmente refeito, incluindo até classes de reflexão.

Classes em PHP

- Elementos
  - Variáveis (Estado)
  - Funções (Operações)
  - Construtor

Sintaxe

- Criando uma classe

```
class nomeDaClasse {
    // variáveis

    // funções

    // construtores
}
```

Sintaxe

- Exemplo:

```
class UsuarioSessao {
    var $nome;
    var $login;

    function salvar() {
        session_start();
        $_SESSION['login'] = $this->login;
        $_SESSION['nome']  = $this->nome;
    }
    function ler() {
        session_start();
        $this->login = $_SESSION['login'];
        $this->nome  = $_SESSION['nome'];
    }
    function UsuarioSessao($login, $nome) {
        $this->login = $login;
        $this->nome = $nome;
    }
}
```

atributos

métodos

construtor

UsuarioSessao
-login -nome
+salvar() +ler()

Sintaxe

- Criando um objeto

```
$us = new UsuarioSessao("sscf", "Sergio");

$us->salvar();

echo $us->nome;
```

The Skin Pattern

- Separar a apresentação (skin) da lógica da aplicação.
- Facilitar o trabalho do designer
  - Ele não precisa saber programar para poder fazer o design de uma página.

Problema: O que acontece na prática é colocar código HTML dentro do código da aplicação, usando chamadas de métodos específicas (echo).

```php
<?php
    ...
        if ($_GET['operacao'] == "Cadastrar") {
        $coditarefa = -1; //gambirarra para o select la debaixo
        $agora = getdate();
        $dia   = $agora['mday'];
        $mes   = $agora['mon'];
        $ano   = $agora['year'];
        $nome      = "";
        $duracao   = "";
        $descricao = "";
    } else {
        $coditarefa    = $_GET['coditarefa'];
        $result_tarefa = $bd->executarQuery("select nome, descricao, EXTRACT(DAY FROM dataInicio) as dia,
EXTRACT(MONTH FROM dataInicio) as mes, EXTRACT(YEAR FROM dataInicio) as ano, DAYOFYEAR(dataFim) -
DAYOFYEAR(dataInicio) as duracao from tarefa where coditarefa=$coditarefa;");
        $tarefa = mysql_fetch_array($result_tarefa);

        $nome      = $tarefa['nome'];
        $dia       = $tarefa['dia'];
        $mes       = $tarefa['mes'];
        $ano       = $tarefa['ano'];
        $duracao   = $tarefa['duracao'];
        $descricao = $tarefa['descricao'];
    }
>?
<html><head><title>Tarefa</title>
<script language="JavaScript">
    function carregarCampos() {
        document.forms[0].nome.value = "<? echo $nome; ?>";
        document.forms[0].duracao.value = "<? echo $duracao; ?>";
        document.forms[0].ano.value = "<? echo $ano; ?>";
        document.forms[0].descricao.value = "<? echo $descricao; ?>";

        dropdown = document.forms[0].dia;
        dropdown.options[0].selected = false;
        dropdown.options[<? echo $dia; ?>].selected = true;
        ...

    }
 </script>
</head>
...
```

The Skin Pattern

- Outro exemplo:

```
<select name="logins[]" size=4 multiple>
<?
    if ($_GET['operacao'] == "Cadastrar") {
        $usuarios = $bd->executarQuery("select login,nome from usuario;");
        while (($usuario = mysql_fetch_array($usuarios))) {
            echo "<option value=" . $usuario['login'] . ">" . $usuario['nome'];
        }
    } else {
        $usuarios = $bd->executarQuery("SELECT u.nome, u.login, tu.coditarefa FROM
usuario as u LEFT JOIN tarefa_usuario as tu ON u.login = tu.login and
tu.coditarefa=$coditarefa;");

        while (($usuario = mysql_fetch_array($usuarios))) {
            $out = "<option value=" . $usuario['login'];
            if ($usuario['coditarefa'] != null) {
                $out .= " selected";
            }
            $out .= ">" . $usuario['nome'];
            echo $out;
        }
    }
?>
 </select>
```

The Skin Pattern

- Solução
  - Dividir o script em 2 partes:
    - Processamento
    - Apresentação

Exemplo Trivial

- Mecanismo:

```
processamento.php
<?php
        $msg = "Você já votou";
        include("template.htm");
?>
```



```
<html>
...
"Você já votou"
...
```

```
template.htm
<html>
...
<?php echo $msg ?>
...
```

The Skin Pattern

Trade-offs

- Separação da apresentação da parte lógica.
- Maior facilidade de alterar qualquer um dos dois.
- Maior legibilidade.
- Pode ser utilizado para localização, e personalização (webmail).

The Skin Pattern

- Vimos uma extremamente simples implementação do Skin pattern.
- Existem implementações mais reais:
  - HTML_Template_Flexy: http://pear.php.net/package/HTML_Template_Flexy
  - Smarty: http://smarty.php.net/
  - PHP Savant : http://phpsavant.com/

PEAR

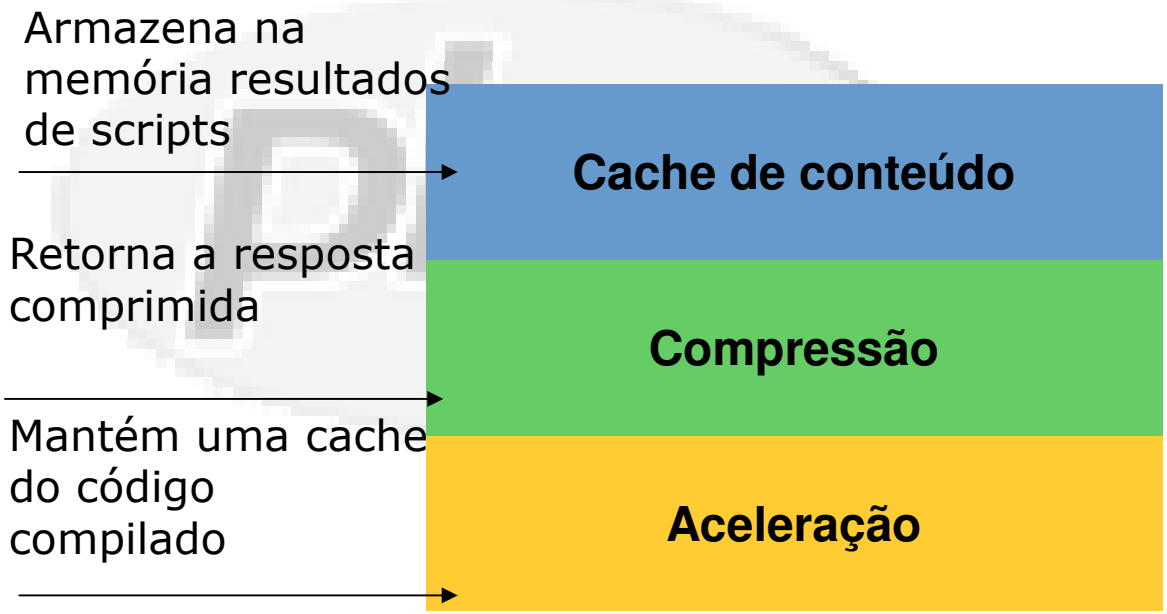
- PHP Extension and Application Repository
- Fornecer bibliotecas open-source para usuários PHP.
- Padrão de codificação.
  http://pear.php.net/manual/en/standards.php

PECL

- Repositório oficial das extensões de php
  - SQLite
  - Zip
  - Rar

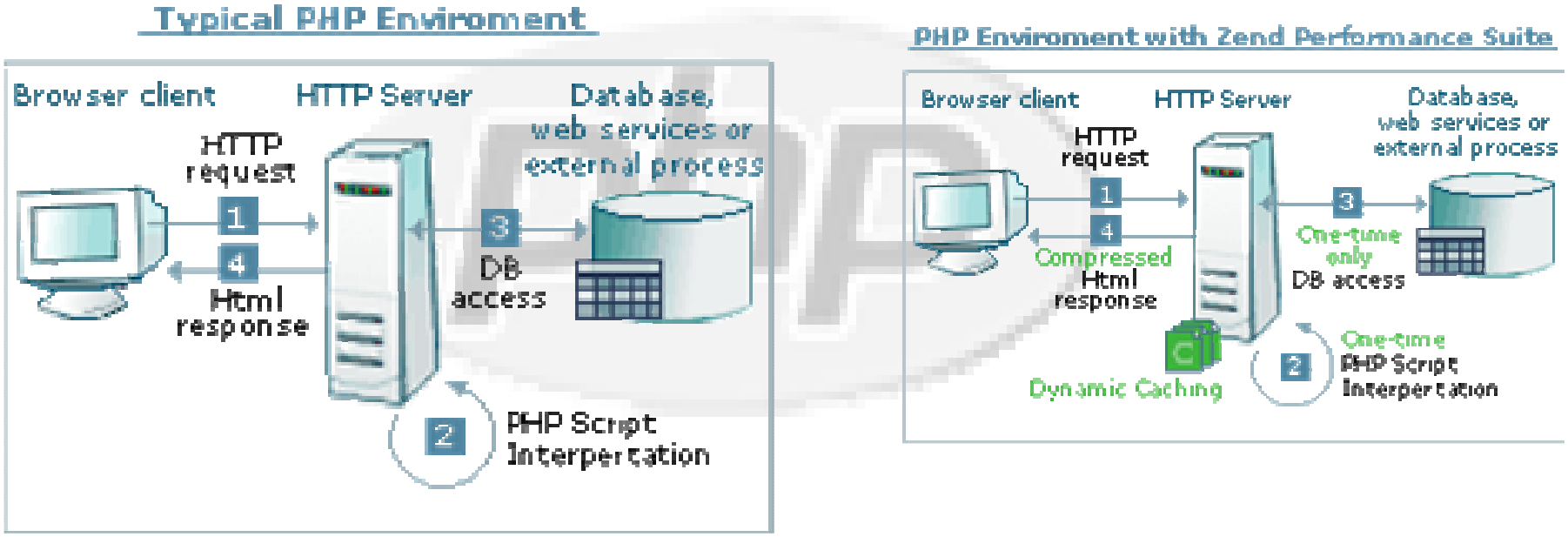
Zend Performance Suite

Componente opcional que melhora absurdamente a performance do servidor.

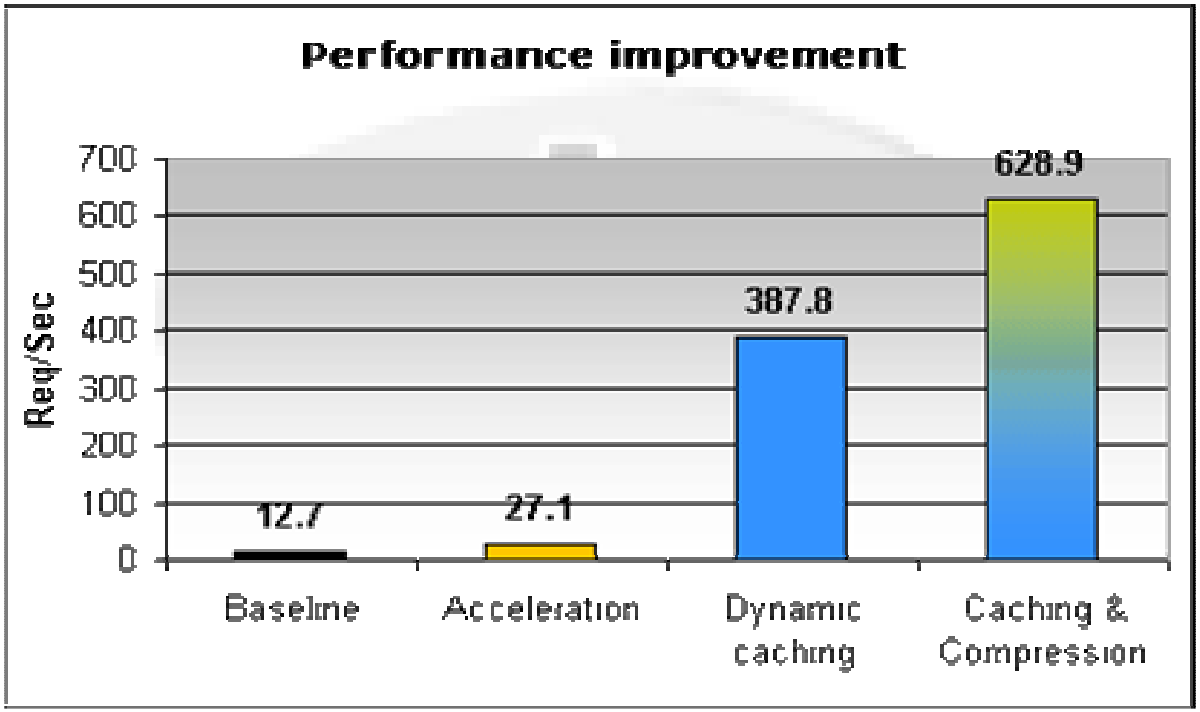


Fonte: http://www.zend.com/store/products/zend-performance-how-it-works.php#1

Zend Performance Suite (cont)



Zend Performance Suite (cont)



Instalação do PHP com apache

- Instalar o phpdev (Apache + MySql + PHP)
  - http://sourceforge.net/projects/phpdev5
  - Já configura o apache para quando houver uma requisição de um arquivo com extensão .php ele mandar o php efetuar o parser.
- Ou manualmente instalando o Apache, PHP, MySql e adicionando a seguinte linha no arquivo httpd.conf do apache.
- ```
  AddType application/x-httpd-php .phtml .php
  Action application/x-httpd-php /php/php.exe
  ScriptAlias /php/ "d:/php/"
  ```
  ```

Debugador de PHP

- IDE
  - NuSphere PhpED (Pago)

    http://www.nusphere.com/
    - Debuga até html

Referências

- How cookies works: http://computer.howstuffworks.com/cookie.htm/printable
- Zend http://www.zend.com/store/products/zend-performance-benchmark.php
- PHP vs .NET http://otn.oracle.com/pub/articles/hull_asp.html
- FreeMarker http://freemarker.sourceforge.net/
- LAMP: http://www.onlamp.com/
- HTTP: http://jmarshall.com/easy/http/
- Site oficial: http://www.php.net

Referências

- Using arrays in PHP: http://www.samspublishing.com/articles/article.asp?p=31840
- RFC2616 : HTML1.1 http://www.ietf.org/rfc/rfc2616.txt
- Object-Oriented Programming Concepts: http://java.sun.com/docs/books/tutorial/java/concepts/
- ODBC: http://www.webopedia.com/TERM/O/ODBC.html
- Strings in MySQL: http://dev.mysql.com/doc/mysql/en/String_syntax.html

Referências

- Building a Database-Driven Web Site Using PHP and MySQL http://dev.mysql.com/tech-resources/articles/ddws/10.html
- HTTP Cookies: http://wp.netscape.com/newsref/std/cookie_spec.html