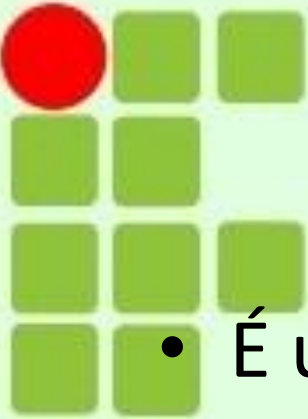


# LPW-PHP

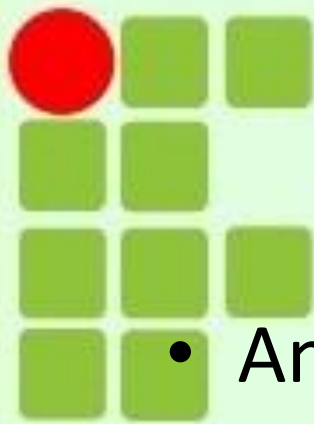
Eduardo Vasconcelos

[eduardo.vasconcelos@garanhuns.ifpe.edu.br](mailto:eduardo.vasconcelos@garanhuns.ifpe.edu.br)



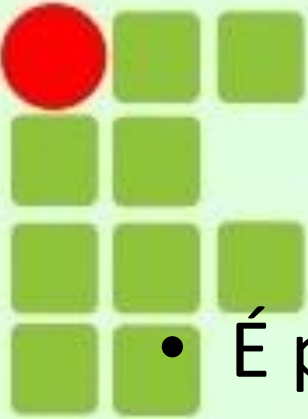
# PHP

- É uma linguagem de programação de código fonte aberto;
- Permite a construção de páginas dinâmicas;
- Similar ao JSP.



# PHP

- Antes de iniciar os estudos sobre PHP é preciso instalar o servidor que o interpreta, o Apache.
  - No linux, basta instalar o servidor por meio do gerenciador de pacotes;
  - No windows é possível baixar e instalar o servidor diretamente ou utilizar um dos diversos programas que facilitam a configuração:
    - Xampp, easyPhp etc.



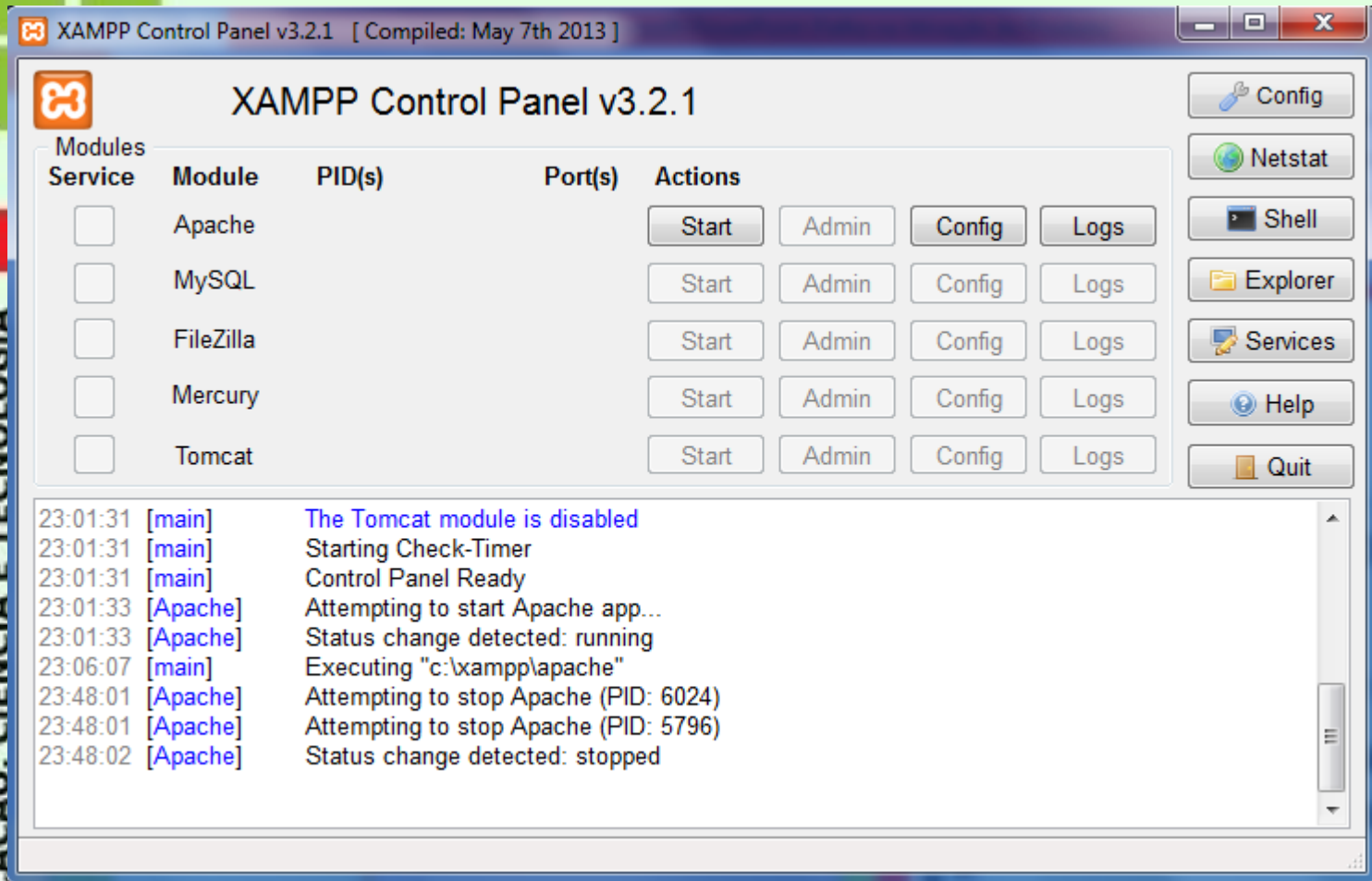
# PHP

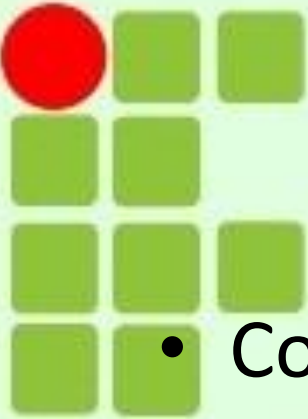
- É possível utilizar o NetBeans como IDE para programar em PHP.
  - Porém o NetBeans não possui o mesmo suporte para PHP que tem para JSP;
- Para usar o NetBeans é preciso baixar uma versão com suporte a PHP.

# PHP

- Configurando o servidor Apache com o xampp.
  - Baixe e instale o Xampp.
  - É possível instalar além do Apache outros serviços, inclusive o mysql.

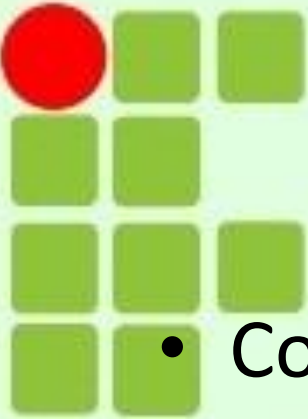
# PHP





# PHP

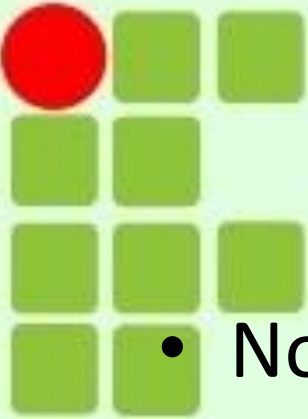
- Configurando o servidor Apache com o xampp.
  - É possível mudar a porta do apache por meio do arquivo de configuração httpd.conf.



# PHP

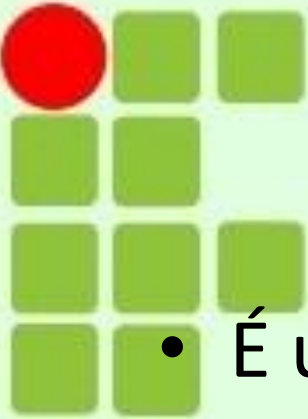
- Configurando o servidor Apache com o xampp.
  - Para acessar o arquivo httpd.conf, basta clicar no botão config do xampp e selecionar o arquivo.
  - Encontre a linha “listen 80” e a modifique para a porta que desejar:
    - Ex: listen 8080





# PHP

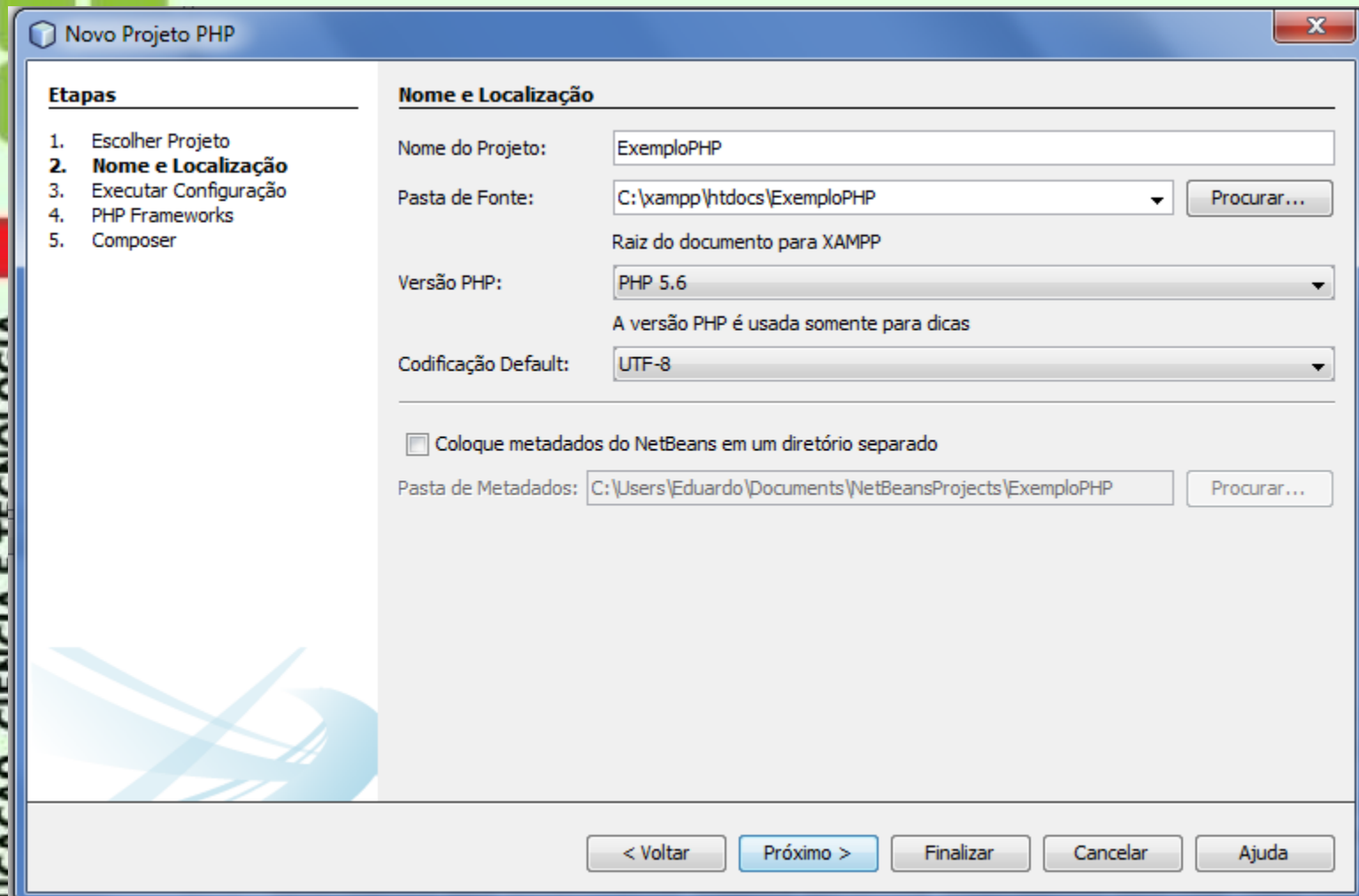
- No NetBeans é preciso fazer referência ao php instalado.
  - Vá em ferramentas/opções;
  - Clique em php;
  - Na aba geral procure pelo arquivo “C:\xampp\php\php.exe”



# PHP

- Quando for criar um projeto PHP, certifique-se, que o projeto está sendo criado na pasta “C:\xampp\htdocs”
  - Nesta pasta ficam todos os projetos executados pelo Apache.

# PHP



**Novo Projeto PHP**

**Etapas**

1. Escolher Projeto
- 2. Nome e Localização**
3. Executar Configuração
4. PHP Frameworks
5. Composer

**Nome e Localização**

Nome do Projeto:

Pasta de Fonte:

Raiz do documento para XAMPP

Versão PHP:

A versão PHP é usada somente para dicas

Codificação Default:

☐ Coloque metadados do NetBeans em um diretório separado

Pasta de Metadados:

# Sintaxe

- Páginas PHP são páginas HTML com código PHP dentro.
- O arquivo deve ter a extensão .php;
- Para inserir código php basta inserir o seguinte bloco de código.

```
<?php
```

```
//código php aqui
```

```
?>
```

# Sintaxe

- O comando echo permite escrever mensagens e valores de variáveis no código html;
  - Funcionamento:
    - Echo “mensagem”;

```
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
    <?php
      echo "primeiro exemplo php!";
    ?>
  </body>
</html>
```

# Sintaxe

- Comentários:
  - // comentário de uma linha;
  - # comentário de uma linha;
  - /\* comentário de múltiplas linhas\*/

```
<body>  
  <?php  
    //comentário  
    #comentário  
    /*  
    * comentário  
    *  
    */  
  ?>  
</body>
```

# Sintaxe

- Para comandos, o php não é sensível a caixa.

```
<body>
  <?php
    Echo "<h1>escreveu</h1>";
    eChO "<h1>escreveu</h1>";
    ECHO "<h1>escreveu</h1>";
  ?>
</body>
```

- Porém para variáveis, php é sensível.
  - Não funciona.

```
<body>
  <?php
    $var = "mensagem";
    echo $VAR;
  ?>
</body>
```

# Sintaxe

- Toda variável em PHP começa com o símbolo “\$”.

\$nome = “fulano”;

\$idade = 20;

\$dinheiroNoBolso = 0.5;





# Sintaxe

- Regras para criação de variáveis;
  - Variáveis devem começar com letras ou o caractere “\_”;
    - \$var, \$\_var;
  - Variáveis devem conter números, letras e/ou o caractere “\_”;
    - \$valor1, \$valor2, \$outro\_Valor;

# Sintaxe

- Variáveis em PHP são fracamente tipadas:
  - Isto significa que não é necessário declarar o tipo da variável;
  - A variável assume o tipo de acordo com o contexto em que está sendo usada.

```
<body>
    <?php
        $var1=10; //int
        $var2="20"; //String

        echo $var1+$var2; //int
        echo "<br>";
        echo $var1.$var2; //String
    ?>
</body>
```

# Sintaxe

- Variáveis em PHP são fracamente tipadas:
  - É possível fazer a conversão de dados de forma explícita.

```
<body>
    <?php
        $var1 = 10;
        $var2 = (String)$var1;

        echo gettype($var1);
        echo "<br>";
        echo gettype($var2);
    ?>
</body>
```



# Sintaxe

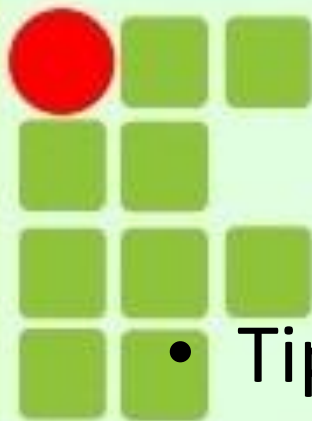
- Assim como o Javascript, php possui operadores de igualdade e identidade.
  - “==” verifica se os valores das variáveis são iguais;
  - “===” verifica se os valores das variáveis são idênticas;
  - “!=” verifica se os valores das variáveis são diferentes;
  - “!==” verifica se os valores das variáveis não são idênticas.

# Sintaxe

- Assim como o Javascript, php possui operadores de igualdade e identidade.

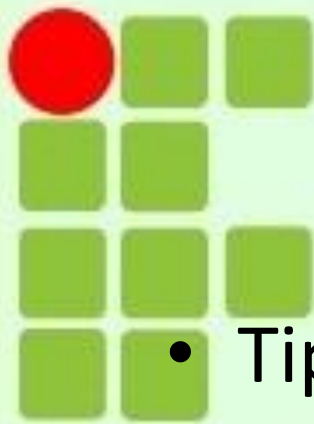
```
<?php
    $var1 = 10;
    $var2 = "10";

    if($var1==$var2){
        echo 'verdadeiro';
    }else{
        echo 'falso';
    }
    echo '<br>';
    if($var1=== $var2){
        echo 'verdadeiro';
    }else{
        echo 'falso';
    }
?>
```



# Sintaxe

- Tipos de dados:
  - Integer:
    - Pode ser nos formatos:
      - Decimal:
        - » \$n1 = 123;
      - Octal:
        - » \$n2=0123;
      - Hexadecimal;
        - » \$n3 = 0x12A;
      - Binário:
        - » Binde('11011');



# Sintaxe

- Tipos de dados:
  - Float:
    - \$n = (float) 45;
    - \$n = 45.0;
    - \$n = 2.2E5 //22000
    - \$n = 2.2E-1 //0.22

# Sintaxe

- Tipos de dados:
  - String.
    - Corresponde a cadeias de caracteres.
      - \$p = “palavra”;
      - \$p = ‘palavra’;
    - Com aspas duplas é possível incluir valores de variáveis dentro da string sem precisar de concatenação.

```
<?php
    $n = 20;
    $p = "o número é $n";
    $p2 = 'o número é $n';

    echo $p;
    echo "<br>";
    echo $p2;
?>
```



# Sintaxe

- Tipos de dados:
  - String.
  - Concatenação entre strings é feito pelo símbolo “.”;

```
<?php
    $n1 = 30;
    $n2 = 40;

    echo "a soma de ".$n1." e ".$n2." é " . ($n1+$n2);
?>
```

# Sintaxe

- Tipos de dados:
  - NULL:
    - É um tipo especial de dado que tem apenas um valor, “NULL”;
      - Variável que não possui valores associados.

```
<?php
    $a = NULL;

    echo gettype($a) ;
?>
```

# Sintaxe

- Constantes:
  - É possível definir constantes em PHP:
    - Sintaxe.
      - Define("nomeDaConstante","valor");

```
<?php  
  
    define("CONSTANTE","Isto é uma constante");  
  
    echo CONSTANTE;  
  
?>
```



# Sintaxe

- Escopos de variáveis:
  - Variáveis globais, são aquelas que podem ser utilizadas em todo o documento.
    - São declaradas no início do documento;
  - Variáveis locais são aquelas declaradas dentro de funções;



# Sintaxe

- Escopos de variáveis:
  - Para acessar uma variável local dentro de uma função podemos usar o termo “global” ou utilizar o array de variáveis globais “\$GLOBALS”.

# Sintaxe

```
<?php
    $var1 = 12; //variável global;
    $var2 = 13;

    function fun(){
        global $var1, $var2;
        return $var1+$var2;
    }

    function fun2($var1,$var2){
        return $GLOBALS["var1"]*$var1+$GLOBALS["var2"]*$var2;
    }

    echo fun();
    echo "<br>";
    echo fun2(1,2);
```

?>

# Sintaxe

- Para definir uma variável estática em uma função usa-se a palavra “static”.

```
<?php
    function fun() {
        static $a = 20;
        $a += $a;
        return $a;
    }

    echo fun() . "<br>";
    echo fun() . "<br>";
    echo fun() . "<br>";
?>
```

# Sintaxe

- Array:
  - Para definir um array utilizamos a seguinte estrutura:  
\$arr = array(1,2,3,4);
  - Neste caso, \$arr possuirá um array de 4 posições.

```
<?php
    $arr = array(1,2,3,4);

    for($i=0;$i<sizeof($arr);$i++){
        echo "$arr[$i] <br>";
    }
?>
```



# Sintaxe

- Array:
  - Para criar um array com intervalos de valores usamos a função range(min,max);

```
<?php
    $arr = range(1,10);

    for($i=0;$i<sizeof($arr);$i++){
        echo "$arr[$i] <br>";
    }
?>
```

```
<?php
    $arr = range('a','z');

    for($i=0;$i<sizeof($arr);$i++){
        echo "$arr[$i] <br>";
    }
?>
```

# Sintaxe

- Array:
  - Arrays podem ser indexados por palavras.
  - Para criar um array indexado por palavras, usamos a seguinte sintaxe:

```
<?php
    $arr = array("palavra1"=>"eita",
                "palavra2"=>"gota");

    foreach ($arr as $indice=>$valor) {
        echo "$indice e $valor <br>";
    }
?>
```

# Sintaxe

- Array:
  - Arrays multi dimensionais podem ser construídos da seguinte forma:

```
<?php
    $arr = array(array(10,20),array(30,40));

    echo $arr[0][1]+$arr[1][1];
?>
```

```
<?php
    $arr = array("array1"=>array(10,20),
        "array2"=>array(30,40));

    echo $arr["array1"][1]+$arr["array2"][1];
?>
```



# Sintaxe

- Manipulação de Strings:
  - Existem várias funções para manipulação de Strings em PHP.
  - Strlen(String):
    - Retorna a quantidade de caracteres de uma string;
  - str\_word\_count(String):
    - Retorna a quantidade de palavras de uma String.
  - strtoupper(String):
    - Converte a String em caixa alta;
  - strtolower(String):
    - Converte a String em caixa baixa.

# Sintaxe

- Manipulação de Strings:
  - implode(Array de String, separador);
  - Converte um array em uma String, separando os elementos do array com o separador.

```
<?php

    $arrLista = array("p1", "p2", "p3");
    $lista = implode($arrLista, "+");

    echo $lista;

?>
```

# Sintaxe

- Manipulação de Strings:
  - Explode(separador, array de string);
  - Torna uma String em um array.

```
<?php

    $frase = "Garanhuns, Caruaru, Belo Jardim";
    $palavras = explode(",", $frase);

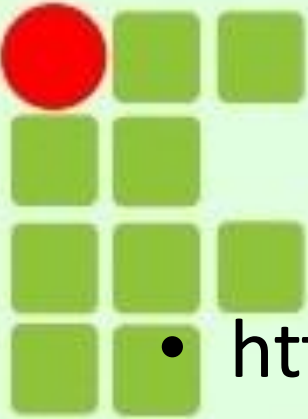
    echo $palavras[0];

?>
```



# Sintaxe

- Manipulação de Strings:
  - strip\_tags(string);
    - Retira eventuais tags de uma String.
  - str\_replace(antigo,novo,\$string);
    - Substitui as ocorrências da palavra “antigo” por “novo”, da \$string;
  - substr(String,int inicio,int fim)
    - Extrai da String uma substring iniciando em “inicio” e terminando em “fim”;
  - strcasecmp(String1, String2)
    - Compara duas Strings sem considerar a caixa.



# Referências

- <http://www.w3schools.com/php/default.asp>
- Lisboa, F. G. S., Zend Framework, Componentes poderosos para PHP, 2º ed. Novatec, 2013.