Linguagem de Programação para WEB II (PHP)

Clenio Emidio

Senac
FACULDADE DE TECNOLOGIA E INOVAÇÃO

php

php

Introdução

- Carga horária disciplina: 60h
- Horários
- Celular
- Avaliações
- Manter comunicação
- Contato (clenio.Fonseca@df.senac.br)
- Metodologia: Atividades teóricas, Exercícios , trabalhos, prática em laboratório*, Projetos...

Alguns tópicos

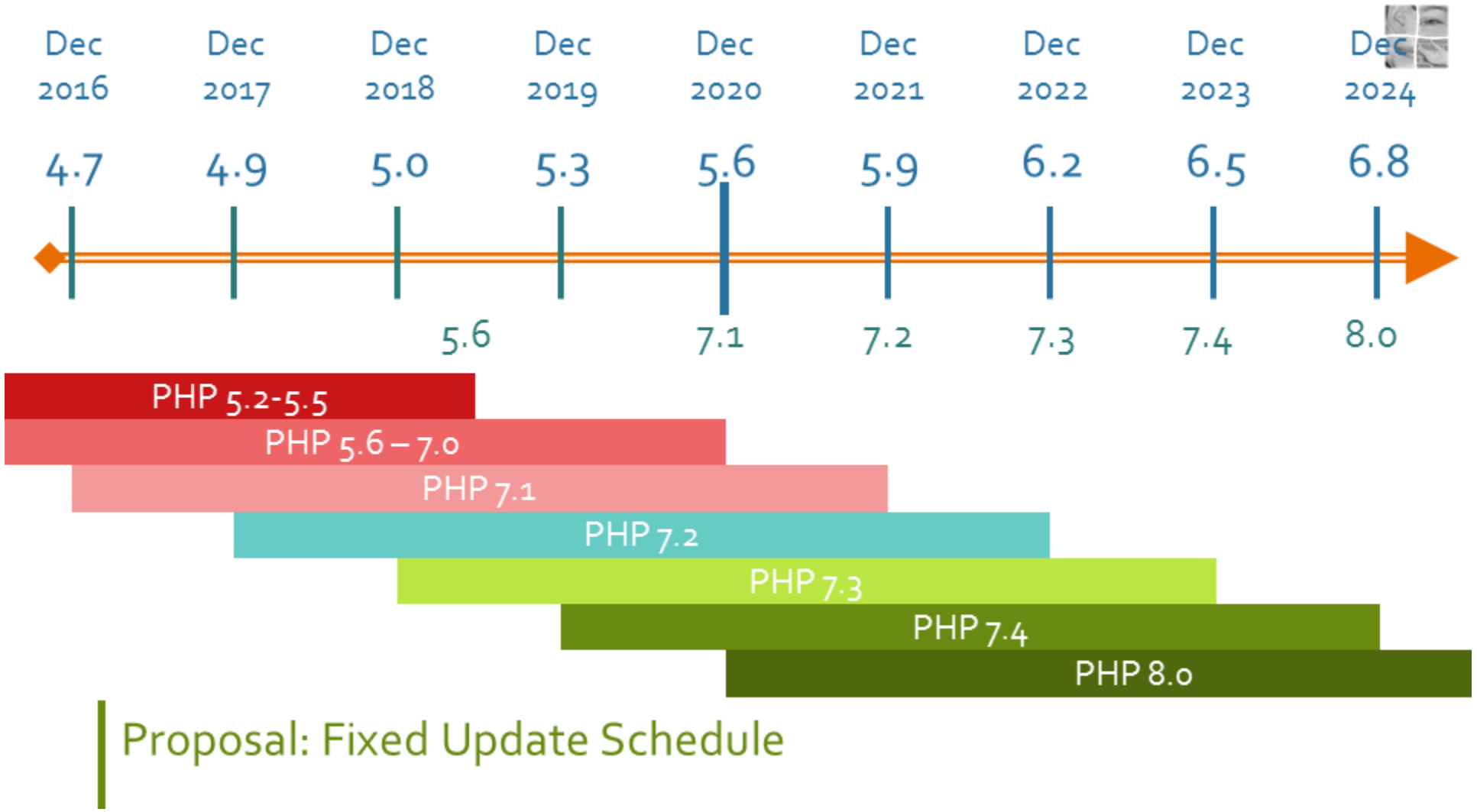
- Introdução ao PHP

- ✓ Sintaxe básica.
- ✓ Variáveis
- ✓ Comentários.
- ✓ Tipos de dados
- ✓ Operadores
- ✓ Estruturas de controle de decisão e repetição
- ✓ Vetores e Matrizes
- ✓ Funções
- ✓ Formulários HTML e PHP
- ✓ Uso de Include e Require
- ✓ PHP e Banco de dados
- ✓ Geração de Relatórios
- ✓ CRUD – PDO
- ✓ classes,
- ✓ Tratamento de exceções
- ✓ Desenvolvimento de uma aplicação com acesso a banco de dados.

História e Conceito

- O PHP (**P**HP **H**ypertext **P**reprocessor) é uma linguagem de programação de código aberto, utilizada principalmente para desenvolvimento web. Ele é frequentemente usado em conjunto com HTML e CSS para criar sites dinâmicos e interativos. Em comparação com outras linguagens de programação web como Java, C# e Python, o PHP é considerado uma opção mais acessível e econômica.
- O PHP foi criado por Rasmus Lerdorf em 1994 como um conjunto de scripts CGI para rastrear visitantes em seu site pessoal. Ele continuou a desenvolver e melhorar a linguagem ao longo dos anos, como suporte nativo a XML e a adição de recursos de programação orientada a objetos. Em 2004, a versão 5.0 foi lançada com ainda mais recursos orientados a objetos e melhorias significativas no desempenho.

Encerramento do Suporte de cada versão

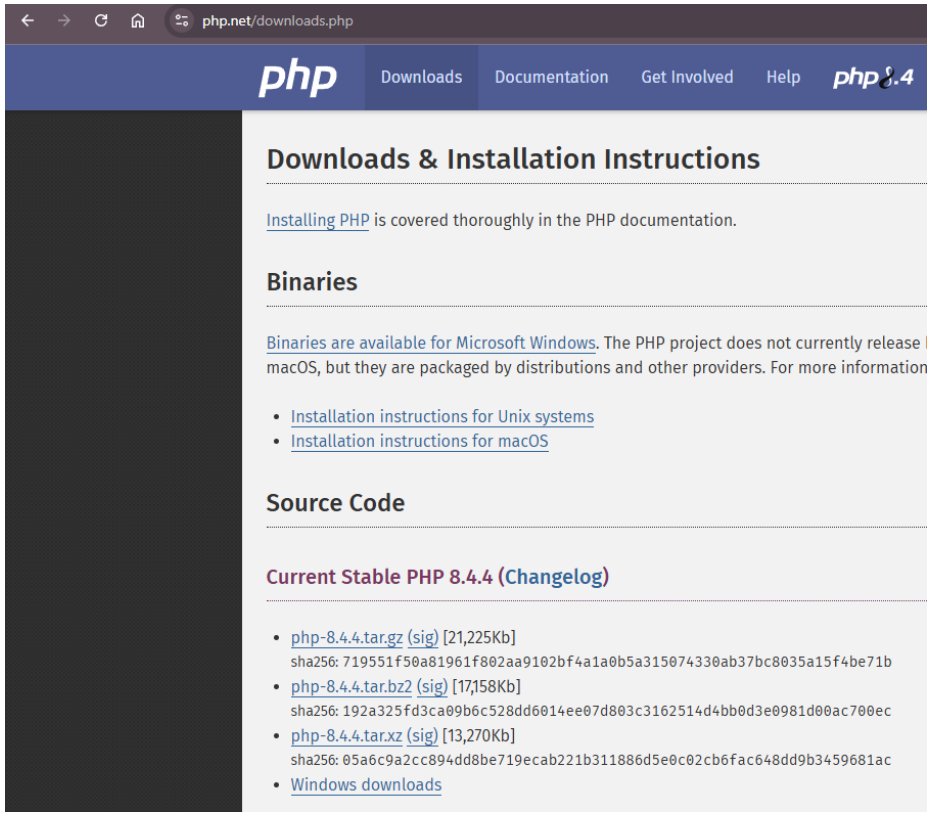


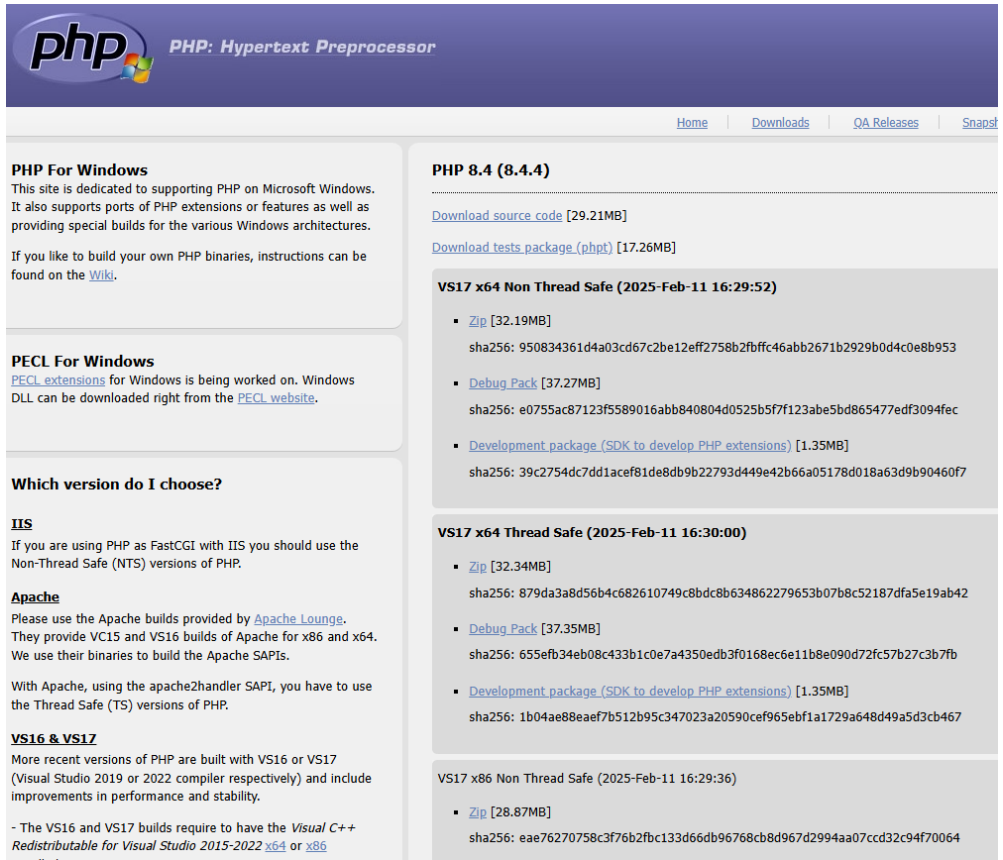
Proposal: Fixed Update Schedule

https://www.youtube.com/watch?v=gQ-P0yMBE9U&list=LL&index=3

Download

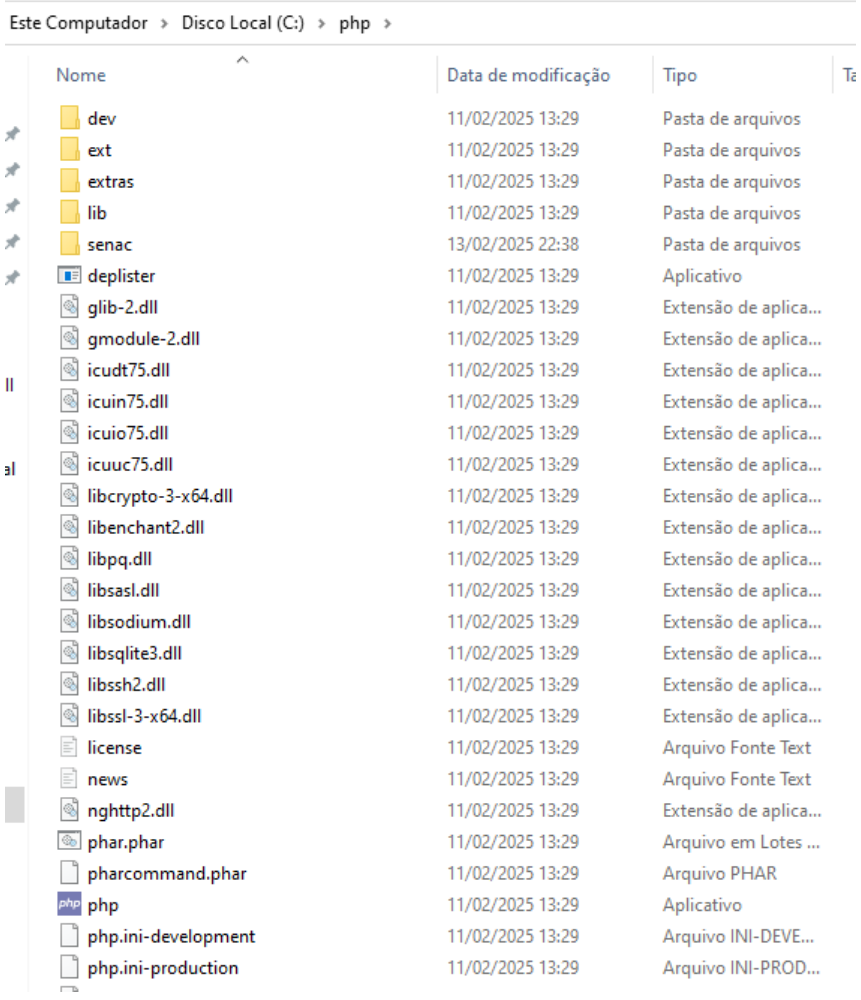
- https://www.php.net/downloads.php
- Clique em Windows Downloads
- Precisaremos instalar o C++
- Então clique em zip para baixar PHP e descompacte na pasta desejada .

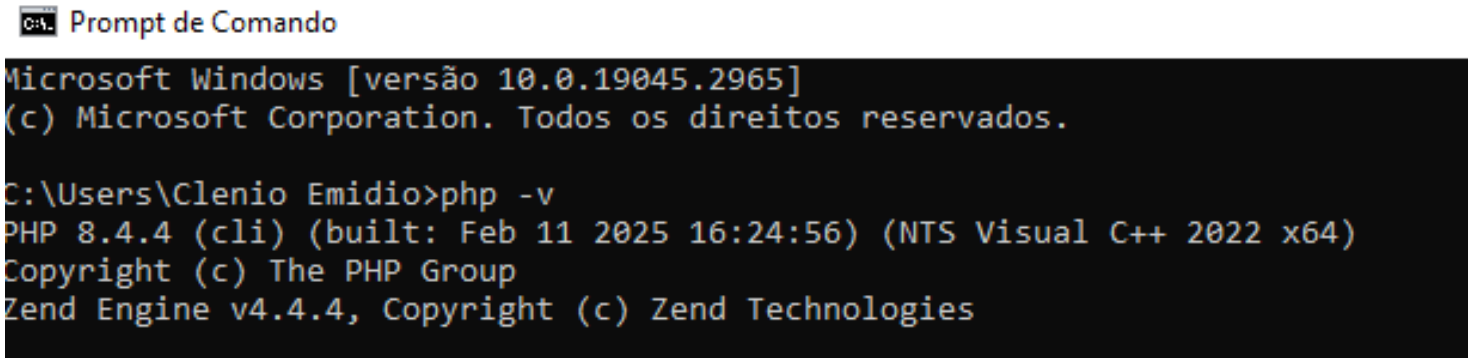




Instalação

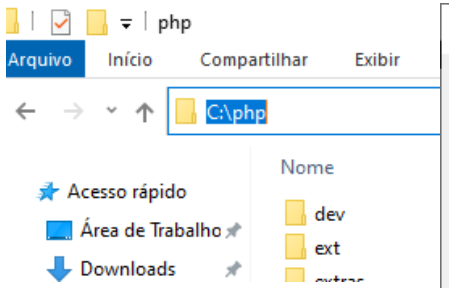
- Crie uma pasta preferencialmente com o nome PHP na unidade c: e descompacte o arquivo zip nesta pasta. Para testar a instalação, vá ao CMD e digite ‘php –v’

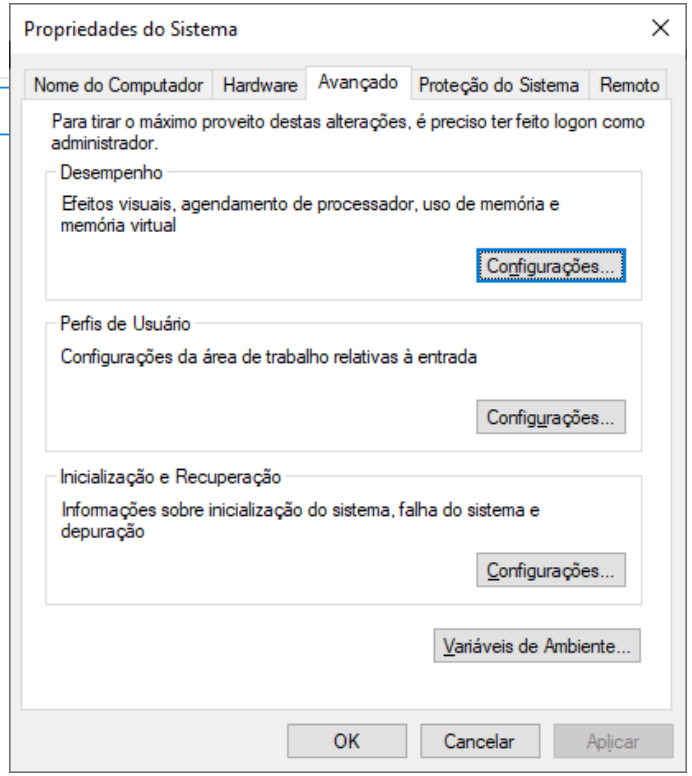


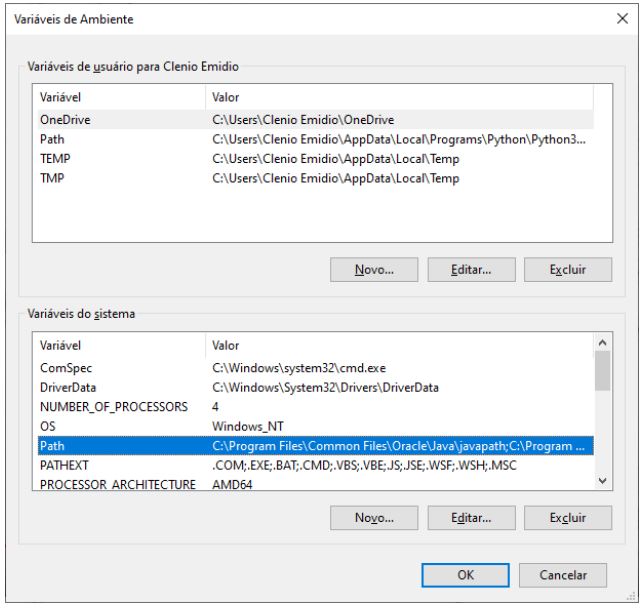


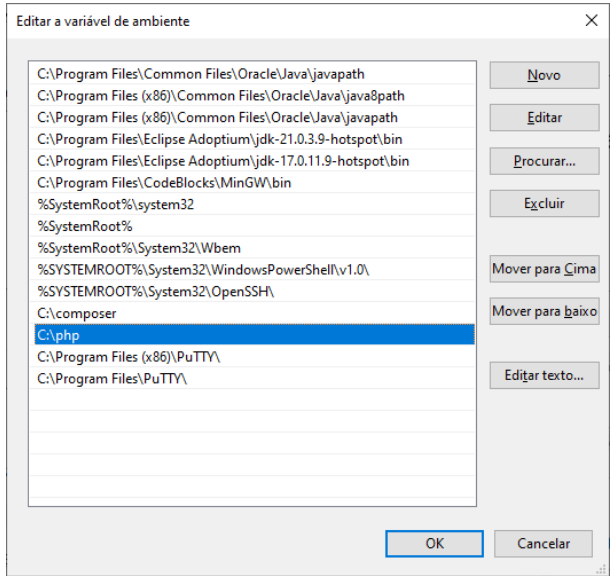
Caso não reconheça o PHP (Windows)

- Copie o endereço da pasta
- Vá em sistema e clique em configurações avançadas do sistema, Variáveis de ambiente, duplo clique em Path e clique em novo e cole o endereço copiado ‘c:\php’ e confirme 3x. Volte ao cmd e digite ‘php –v’ para verificar.



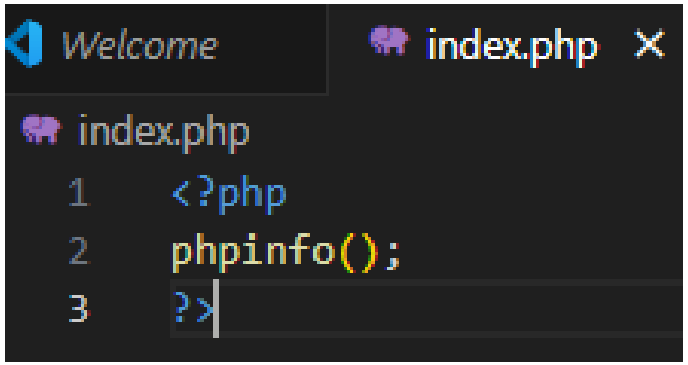


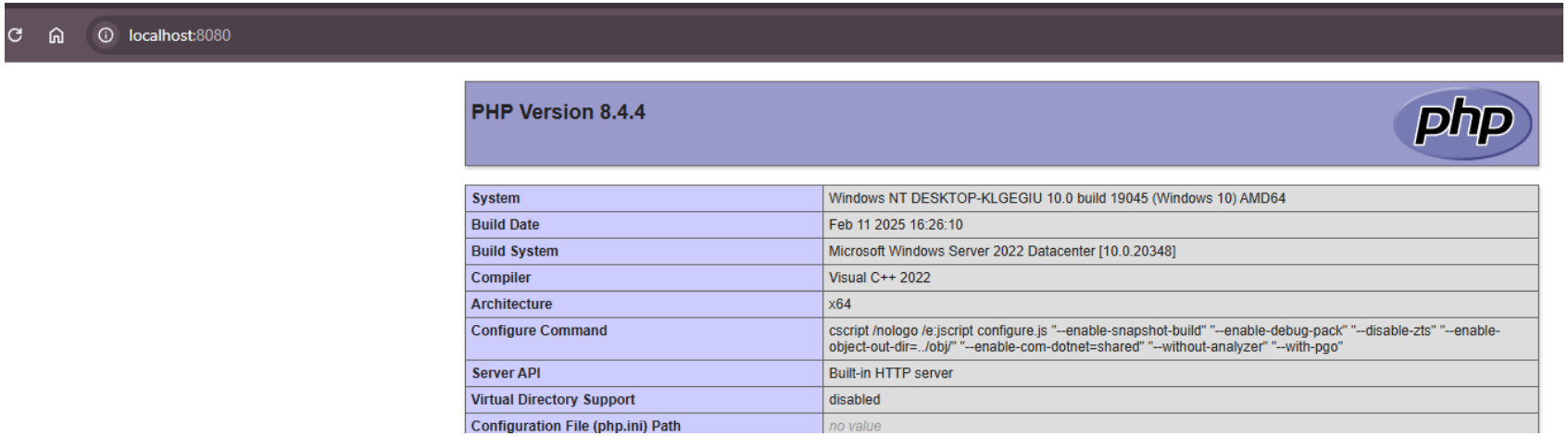




Executando

- Crie preferencialmente uma pasta para os testes ou projeto dentro da pasta PHP.
- Abra sua IDE vs code ou outra e crie um arquivo com extensão php chamada **index.php**.
- Podemos agora utilizar o próprio servidor embutido no PHP como o comando ‘php –S localhost:8080’.
- Em seguida, abra o navegador e digite ‘localhost:8080’



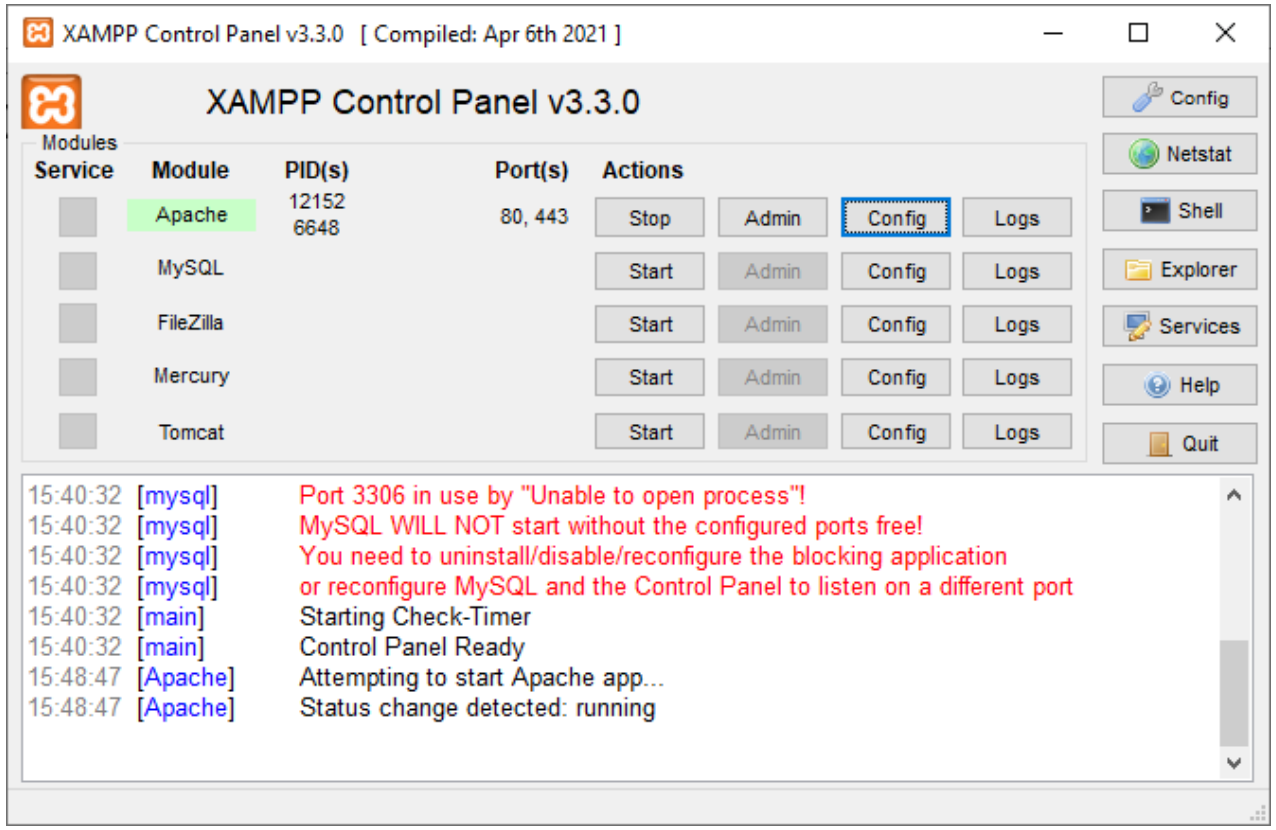


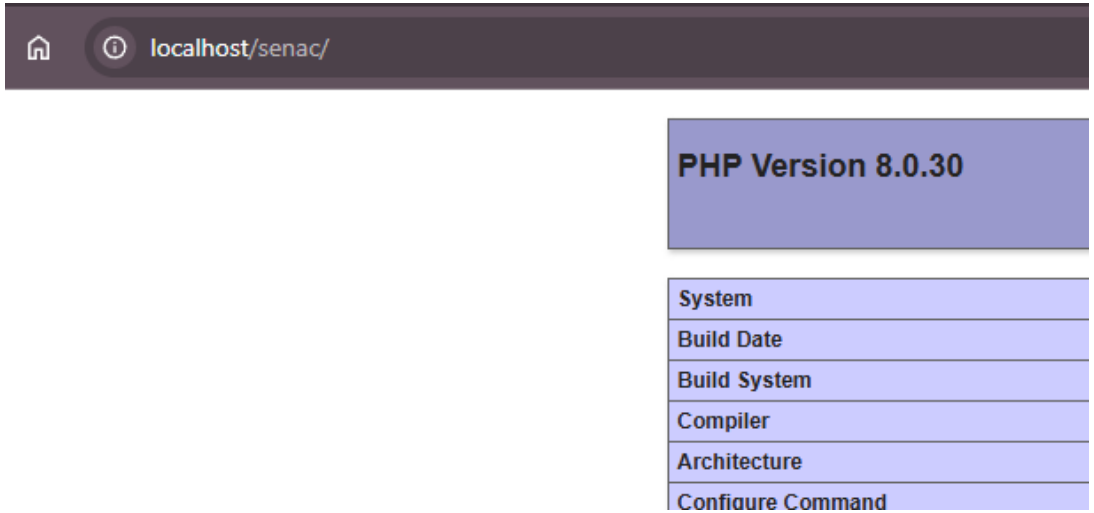
Utilizando servidor Externo

- Temos algumas opções:
  - Easy php
  - Xampp
  - wampServer

Utilizando o Xampp

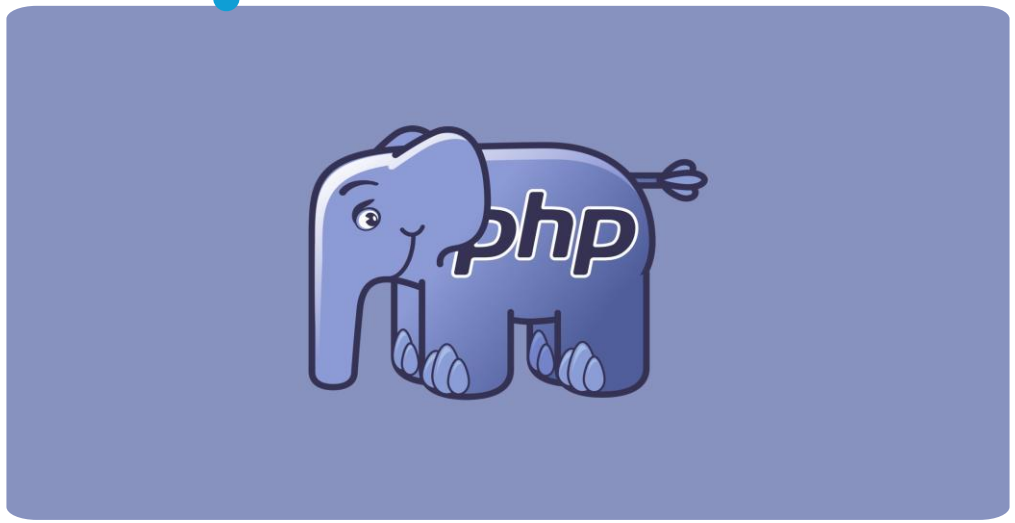
- Após instalar e abrir o Xampp, basta clicar em Start no servidor Apache.
- Criar o projeto na pasta ‘htdocs’ e chamar a pasta no navegador.





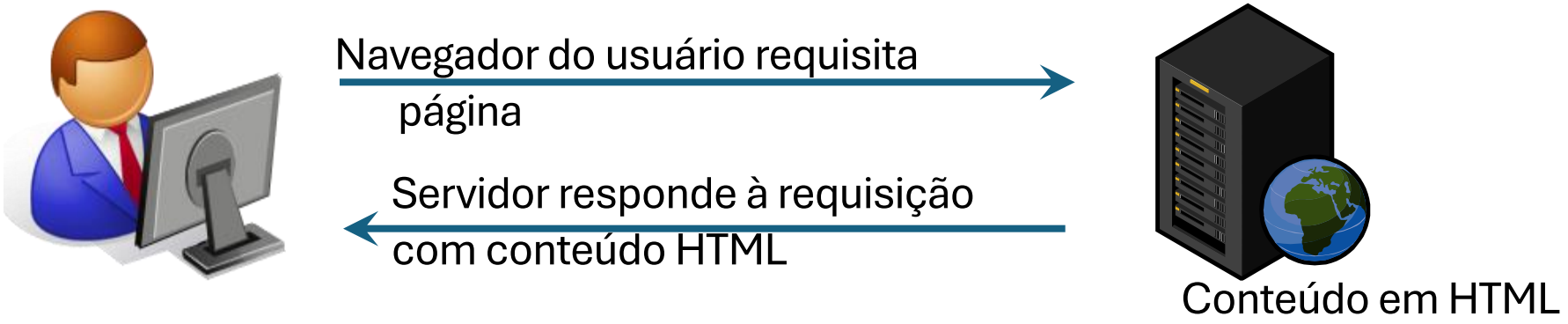
Conteúdo

- Linguagem Estática x Dinâmica
- O que é PHP?
- Sintaxe
  - Comentários
  - Declaração de variáveis
  - Constantes
  - Tipos de dados
  - Operadores
  - Estruturas de decisão
  - Estruturas de laço
  - Funções

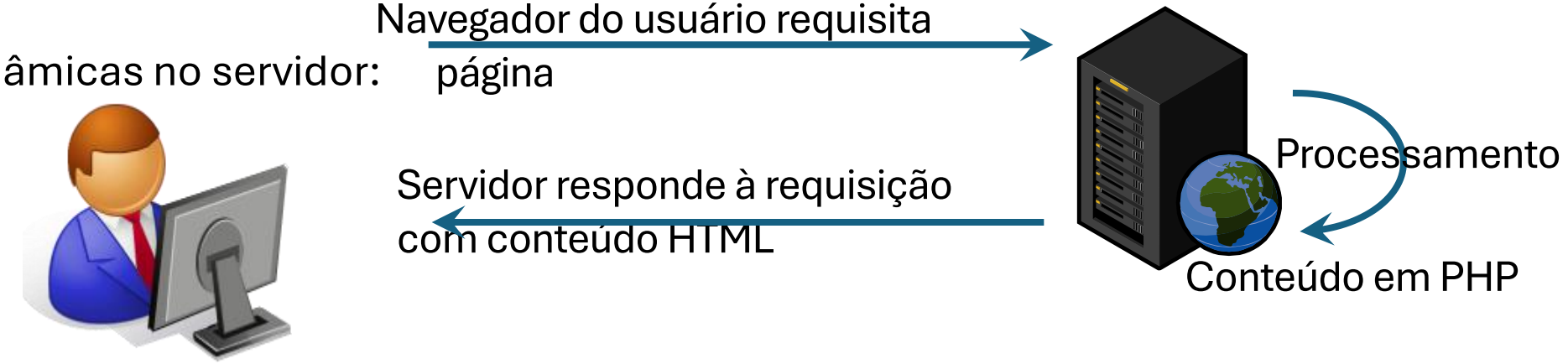


Linguagem Estática x Dinâmica

- Páginas estáticas no servidor:



- Páginas dinâmicas no servidor:



Linguagem Estática x Dinâmica

- Páginas **estáticas** são aquelas escritas utilizando a linguagem HTML, com nenhuma interação com o usuário, devido as próprias restrições da linguagem HTML.
- Páginas **dinâmicas** são aquelas escritas utilizando alguma linguagem de programação. As quais podem exibir seu conteúdo além do simples código escrito em HTML.
- Estas páginas possuem **interação** com o usuário, pois devido ao poder das linguagens de programação podemos tratar informações de formulários, acessar bancos de dados, e podem possuir várias funcionalidades.

O que é PHP?

- PHP **(um acrônimo recursivo para "*PHP: Hypertext Preprocessor*")** é uma linguagem de script Open Source de uso geral, muito utilizada e especialmente guarnecida para o desenvolvimento de aplicações web embútivel ao código HTML.
- É uma linguagem que permite criar sites web dinâmicos, possibilitando uma interação com o usuário através de formulários, parâmetros da URL e links.
- A diferença de PHP com relação a linguagens semelhantes a JavaScript é que o código PHP é executado no servidor, sendo enviado para o cliente apenas HTML puro.
- Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o usuário, sendo útil quando o sistema lida com senhas ou qualquer tipo de informação confidencial.

O que é PHP?

- Para que um página com código PHP possa ser executada corretamente pelo servidor, o arquivo da página deve conter a extensão .php.

- Se for requisitado um arquivo PHP diretamente ao servidor que não possua esta extensão, o servidor tratará todo o código-fonte como HTML.

- Quando o servidor PHP interpreta uma página, ele percorre o código-fonte do arquivo até encontrar a tag que indica o início do conteúdo PHP. O interpretador então executa todo o código que encontra, até chegar na tag de fechamento PHP.

- Este é o mecanismo que permite a inclusão de código PHP dentro do HTML. Isso significa que todo o código PHP pode estar escrito entre as tags HTML.

O que é PHP?

- Qualquer coisa fora das tags PHP é deixada como encontrado, enquanto tudo dentro é interpretado e executado.

- Há quatro conjuntos de tags que podem ser usadas para marcar blocos de código PHP. Duas delas estão habilitadas por padrão no PHP, as outras devem ser habilitadas manualmente.

- Habilitadas por padrão

  <?php . . . ?>

  <script language="php"> . . . </script>

- Habilitadas manualmente

  <? . . . ?>

  <%. . . %>

Sintaxe

- O código PHP fica embutido entre as tags HTML. O interpretador do PHP identifica quando inicia e termina uma instrução ou bloco de instruções em PHP quando são usadas as seguintes tags:

```
<?php
    Instruções
?>
<script language="php">
    instruções
</script>
```

Habilitados por padrão

```
<?
    instruções
?>
```

Forma abreviada da primeira opção, para funcionar deve-se habilitar no arquivo php.ini a opção *short_open_tag* para *on*

```
<%
    instruções
%>
```

Esta forma foi criada para desenvolvedores acostumados com a linguagem asp, para funcionar deve-se habilitar no arquivo php.ini a opção *asp_tags* para *on*

Sintaxe

- Ao escrever um código em php, da mesma forma da linguagem C, devemos colocar um ponto e vírgula (*;*) separando cada instrução.

```
<?php
    echo “meu código em php”;
?>
```

- No caso de apenas uma instrução ou da última instrução, antes do fechamento da tag PHP, o uso do ponto e vírgula (;) não é obrigatório.

```
<?php
    echo “instrução 1”;
    echo “instrução 2”
?>
```

- Por questão de padrões sempre utilizamos o ponto e vírgula ao fim de cada instrução.

Sintaxe

- Strings em PHP
  - Quando escrevemos uma string, podemos utilizar aspas (“ ”) ou apóstrofos (‘ ’) para delimitá-la.

```
<?php
    echo “exemplo string com aspas”;

    echo ‘exemplo string com apóstrofos’;
?>
```

  - Quando precisamos escrever uma aspa ou um apóstrofo dentro de uma string, precisamos “escapá-la”, utilizando uma contra-barra (\ )

```
<?php
    echo “exemplo aspas \“ dentro da string com aspas”;

    echo ‘exemplo apóstrofo \’ dentro da string com apóstrofos’;
?>
```

Sintaxe

- Strings em PHP
  - A diferença entre o uso de aspas e apóstrofo é que quando utilizamos aspas, o PHP consegue interpretar caracteres de escape especiais e ainda, faz a interpretação de variáveis.

```
<?php
    echo “exemplo string com \r retorno de carro”;

    echo “exemplo string fim de linha \n”;


    echo “texto”;


    $nome = “João”;
    echo “Seu nome é $nome”;

?>
```

Sintaxe - Comentários

- O PHP suporta 2 estilos de comentário.
  - Comentários de uma linha utilizando # ou //:
  - Comentários de mais de uma linha utilizando /* .. */

```
<?php
    # Comentário 1
    echo “exemplo de comentário”;

    // Comentário 2
    echo “exemplo de comentário”; // Comentário 3

    /* Comentário com mais de
    uma linha */
    echo “exemplo de comentário”;
?>
```

Sintaxe - Comentários

- Nossa primeira página em PHP:

```
<html>
   <head>
      <title>Primeira página em PHP</title>
   </head>
   <body>
      <?php
                  echo “Este é meu primeiro script em PHP”;
                  echo “<p>Posso escrever tags HTML”;
                  echo “dentro dos textos em PHP</p>”;
      ?>
   </body>
</html>
```

Sintaxe - Declaração de Variáveis

- O uso de variáveis no PHP requer algumas regras:
    - **Toda variável começa pelo caractere cifrão ($);** Após o cifrão ($) deve-se colocar uma string que deve começar por uma letra ou pelo caractere underline (_);
    - Após esses caracteres só podem ser inseridos letras ou números.
- No PHP as variáveis são *case sensitive*, ou seja, $var, $Var e $VAR são variáveis diferentes e podem possuir valores diferentes.
- O PHP possui conversão automática de tipo, chamada também de coerção de tipo automática. Assim, a definição de tipo explicita não é necessária na declaração de variáveis.

Sintaxe - Declaração de Variáveis

- O **tipo de uma variável é determinado pelo contexto em que a variável é utilizada**. Isto significa que, se você assimila um valor string para a variável $var, $var se torna uma string. Se você então assimila um valor **inteiro** para $var, ela se torna um **inteiro**.

- Exemplo de variáveis PHP:

```
$variavel = “valor”;
$_variavel = 0;
$nome = “João”
$cidade = “Belém”
$mes1 = “Janeiro”
```

Sintaxe - Constantes

- No PHP, a declaração de constantes também possui algumas regras:
  - Constantes devem ser declaradas em maiúsculas.
  - Constantes não podem ter um sinal de cifrão ($) antes delas;
  - Constantes só podem ser definidas utilizando a função define();
  - Constantes podem ser definidas e acessadas de qualquer lugar sem que as regras de escopo de variáveis sejam aplicadas;
  - Constantes não podem ser redefinidas ou eliminadas depois que elas são criadas;
  - Constantes só podem conter valores escalares.
  - O PHP já possui diversas constantes pré-definidas.
  - Constantes: TRUE, FALSE, _FILE_ , _LINE_, PHP_VERSION, PHP_OS, E_ERROR, E_WARNING, E_PARSE, E_NOTICE.

- Exemplo de declaração de constantes:

```
<?php
    define(“VARIAVEL”,"valor");
    echo VARIAVEL;
?>
```

```
define (“PI", 3.1415926536);
    $circunf = 2*PI*$raio;
```

Sintaxe - Tipos de Dados

- O PHP trabalha com os tipos de dados:
  - Boolean
  - Integer
  - Float
  - String
  - Array
  - Object
  - Resource
  - Null

Tipo	Exemplo	Descrição
Inteiro	100	Um número inteiro.
Double	10.25	Um número de ponto flutuante.
String	"Aprendendo PHP"	Uma coleção de caracteres.
Boolean	True	Um dos seguintes valores: true ou false.
Object		Estudaremos este tipo mais a frente.
Array		Estudaremos este tipo mais a frente.

Sintaxe - Tipos de Dados

- Booleanos
  - Para especificar um literal booleano, use as palavras chave TRUE ou FALSE. Ambas são insensitivas ao caso.

- Inteiros
  - Inteiros podem ser especificados em notação decimal (base 10), hexadecimal (base 16) ou octal (base 8), opcionalmente precedido de sinal (- ou +). O tamanho de um inteiro é dependente de plataforma, sendo um numero aproximado a 2 bilhões o valor mais comum (número de 32 bits com sinal).

  - Overflow de inteiros
    - Se você especifica um número além dos limites do tipo inteiro, ele será interpretado como um ponto flutuante.

Sintaxe - Tipos de Dados

- Números de pontos flutuantes
  - O tamanho de um número de ponto flutuante é dependente de plataforma, sendo o máximo com uma precisão de 14 decimais digitais.

- Strings
  - Uma string é uma série de caracteres. No PHP, um caracter é o mesmo que um byte, ou seja, há exatamente 256 caracteres diferentes possíveis. Não há nenhum problema se as strings se tornarem muito grandes. Não há nenhum limite para o tamanho de strings imposta pelo PHP, então não há razão para se preocupar com strings longas.

- Arrays
  - Um array no PHP é atualmente um mapa ordenado. Um mapa é um tipo que relaciona valores para chaves. Este tipo é otimizado de várias maneiras, então você pode usá-lo como um array real, ou uma lista (vetor), hashtable (que é uma implementação de mapa), dicionário, coleção, pilha, fila e provavelmente mais. Como você pode ter outro array PHP como um valor, você pode facilmente simular árvores.

Sintaxe - Tipos de Dados

- Objetos
  - Utilizado na orientação a objetos

- Resource
  - Resource (Recurso) é uma variável especial, mantendo uma referência de recurso externo. Resources são criados e utilizados por funções especiais. Por exemplo, quando criamos uma conexão com um banco de dados, o PHP retorna uma variável, a qual possui o status da conexão dentro do sistema.

- NULL
  - O valor especial NULL representa que a variável não tem valor.

Sintaxe - Operadores

- **Operadores aritméticos:**
  - $x + $y – Adição;
  - $x - $y – Subtração;
  - $x / $y – Divisão;
  - $x * $y – Multiplicação;
  - $x % $y – Módulo;

Exemplo	Nome	Resultado
-$a	Negação	Oposto de $a.
$a + $b	Adição	Soma de $a e $b.
$a - $b	Subtração	Diferença entre $a e $b.
$a * $b	Multiplicação	Produto de $a e $b.
$a / $b	Divisão	Quociente de $a por $b.
$a % $b	Módulo	Resto de $a dividido por $b.

- **Operadores de Atribuição:**
  - O operador de atribuição é o igual (=)
  - $x = ($y = 5) + 1; $y recebe o valor 5 e $ x recebe o valor da soma entre $y e 1. $x recebe o valor 6;

Sintaxe - Operadores

- **Operadores de comparação:**
  - $x == $y – igual;
  - $x === $y – idêntico;
  - $x != $y – diferente;
  - $x > $y – maior que;
  - $x < $y – menor que;
  - $x >= $y – maior ou igual a;
  - $x <= $y – menor ou igual a;

- **Operadores de controle de erro:**
  - O PHP suporta um operador de controle de erro: o caractere arroba (@). Quando ele precede uma expressão em PHP, qualquer mensagem de erro que possa ser gerada por aquela expressão será ignorada.
    - @gettype($variavel);

Sintaxe - Operadores

- **Operadores de incremento/decremento:**
  - ++$x – pré-incremento;
  - $x++ – pós-incremento;
  - --$x – pré-decremento;
  - $x-- – pós decremento;

Operadoress	Nome	Resultados
++$sum	Pré-incremento	echo $sum; // 11
$sum++	Pós-incremento	echo $sum; // 10
--$sum	Pré-decremento	echo $sum; // 9
$sum--	Pós-decremento	echo $sum; // 10

1. Os operadores Pré fazem a operação **ANTES** do retorno.
2. Os operadores Pós fazem o retorno **ANTES** da operação.
3. O comportamento é igual ao do *javascript*.

- **Operadores lógicos:**
  - $x and $y – verdadeiro se $x e $y forem verdadeiros.
  - $x or $y – verdadeiro se $x ou $y forem verdadeiros.
  - $x xor $y – verdadeiro se $x ou $y forem verdadeiros, mas não ambos.
  - !$x – verdadeiro se $x for falso.
  - $x && $y – verdadeiro se $x e $y forem verdadeiros.
  - $x || $y – verdadeiro se $x ou $y forem verdadeiros.

Sintaxe - Operadores

- **Operadores de texto (concatenação)**
  - Para concatenar textos utilizamos o operador **ponto (.)**

```
$ a = “a”; $b = “b”;
$c = $a.$b; //$c = “ab”;
```

  - Para atribuir e concatenar textos utilizamos o operador **ponto e igual (.=)**

```
$c = “ab”; $c .= “c”; //$c igual a “abc”
```

- **Operadores de array:**
  - $x + $y – união entre $x e $y;
  - $x == $y – igualdade – se possui os mesmos elementos;
  - $x === $y – identidade – se possui os mesmos elementos na mesma ordem;
  - $x != $y e $x <> $y – diferença;
  - $x !== $y – não possui a mesma identidade;

Sintaxe - Estruturas de decisão

- **if...else... elseif**
  - No PHP, assim como nas outras linguagens baseadas em C, os valores a serem comparados dentro deste tipo de estrutura devem estar entre parenteses.

```
if ($x == $y)
        echo “x é igual y”;
```

  - Quando utilizamos mais de uma instrução dentro deste tipo de estrutura, é necessário o uso de chaves para determinar o início e fim da estrutura.

```
if ($x > $y) {
        echo “x é maior que y;
        $x = $y;
}
```

Sintaxe - Estruturas de decisão

- **if...else...elseif**
  - Uso do else

```
if ($x == $y){
        echo “x é igual y”;
}else{
        echo “x é diferente de y”;
}
```

  - Uso do else if

```
if ($x == $y){
        echo “x é igual y”;
}else if ($x > $y){
        echo “x é maior que y”;
}else{
        echo “x é menor que y”;
}
```

Sintaxe - Estruturas de decisão

- **switch** – estrutura que substitui estruturas if aninhadas

```
switch($i){
  case 0:
            echo “i é igual a 0”;
            break;
  case 1:
            echo “i é igual a 1”;
            break;
  case 2:
            echo “i é igual a 2”;
            break;
  default:
            echo “i é diferente de 0, 1 e 2”;
}
```

Sintaxe - Estruturas de laço

- Estrutura **while** – executa um bloco de instruções enquanto uma condição for satisfeita. O bloco de instruções é executado somente após a verificação da condição.

```
<?php
    $i = 0;
    while ($i < 10){
                    echo $i;
                    $i++;
    }
?>
```

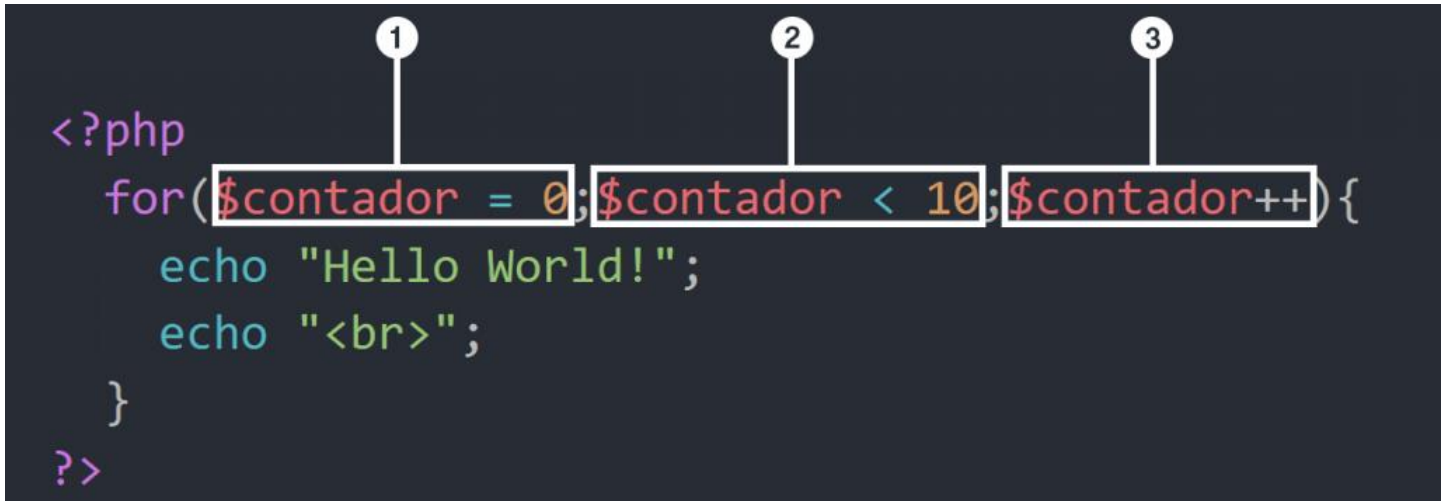
- Estrutura **do ... while** – parecida com a estrutura while. A diferença é que aqui o bloco de instruções é executado antes da verificação da condição.

```
<?php
    $i = 0;
    do {
        echo $i;
    }while ($i > 0);
?>
```

Sintaxe - Estruturas de laço

- Estrutura **for** – utilizada quando precisamos repetir um bloco de instruções por um determinado número de vezes.

```php
<?php
   for ($i = 0; $i <= 10; $i++){
      echo $i;
   }
?>
```



Sintaxe - Funções

- Funções são blocos de instruções criadas para realizar determinada tarefa dentro de um sistema.

- Para criarmos uma função devemos seguir a seguinte sintaxe:

```
<?php
    function nome_da_funcao(){
        bloco de instruções;
    }
?>
```

- Podemos ainda definir a necessidade de parâmetros para uma função:

```
<?php
    function nome_da_funcao($arg1, $arg2,...,$argN){
        bloco de instruções;
    }
?>
```

Sintaxe - Funções

- As funções podem retornar valores.

```
<?php
    function funcao_com_retorno(){
            bloco de instruções;
            return resultado_da_função;
    }
?>
```

- Para executarmos uma função basta chamar seu nome em qualquer parte do código, fora do bloco da função.

```
<?php
    function nome_da_funcao (){
            bloco de instruções;
            return resultado_da_função;
    }
    nome_da_funcao();
?>
```

Casas decimais em PHP

- $number = 1234.5600;
- $number = number_format($number, 2, '.', '');

- Inteiro 1 para 1.00
- number_format(1, 2, '.', '');

```php
<?php

$numero = 1234.56;

// Formato brasileiro
echo number_format($numero, 2, ',', '.');
// 1.234,56

// Formato frances
echo number_format($numero, 2, ',', ' ');
// 1 234,56

$number = 1234.5678;

// Formato inglês sem separador de milhar
echo number_format($numero, 2, '.', '');
// 1234.57
```

Bibliografia

- https://blog.schoolofnet.com/php-guia-completo-para-iniciantes/
- https://medium.com/trainingcenter/php-b%C3%A1sico-operadores-2ff49b34399f
- https://www.tecnoveste.com.br/php-hypertext-preprocessor-tudo-o-que-voce-precisa-saber-historico-caracteristicas-e-por-onde-comecar/