SENAI CIMATEC SISTEMA FIEB
Federação das Indústrias do Estado da Bahia

2016.2

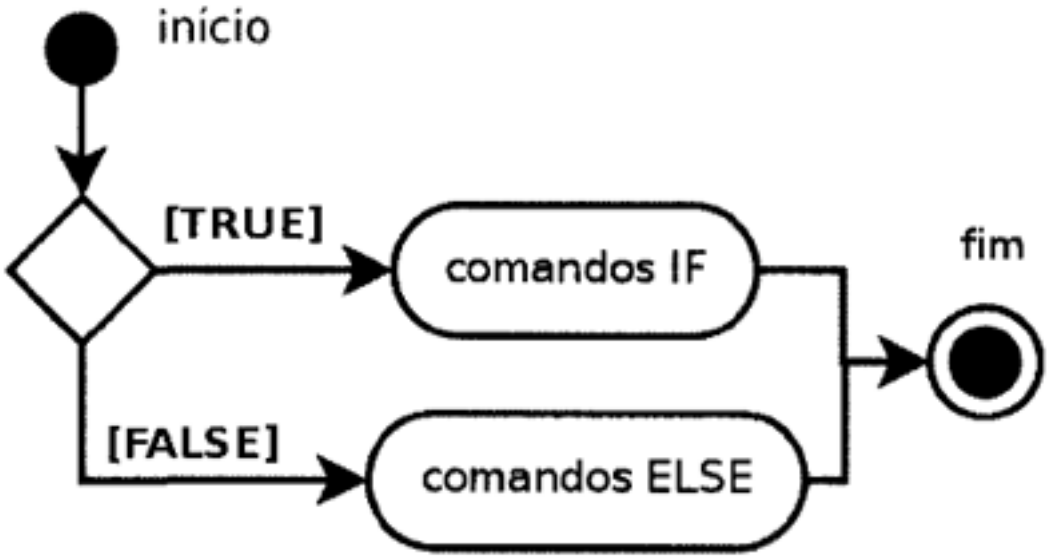
Estruturas de Controle em PHP e Variáveis de Sessão

Docente: Gil Jader – gil.jader@gmail.com

1 – Estruturas Condicionais

if, else e elseif

- Avalia uma expressão e dependendo do resultado é executado um conjunto diferente de instruções.



1 – Estruturas Condicionais

if, else e elseif

- Avalia uma expressão e dependendo do resultado é executado um conjunto diferente de instruções.

if (exp1)
{ bloco1 }

- Se a exp1 for verdadeira, execute o bloco1

elseif (exp2)
{ bloco2 }

- Senão se exp2 for verdadeira, execute o bloco2

else
{ bloco3 }

- Senão execute o bloco3

1 – Estruturas Condicionais

if, else e elseif

Observações:

- Podemos ter diversos **elseifs** durante uma avaliação condicional;
- Caso o bloco só tenha uma linha, então as chaves ({ }) são dispensáveis;
- Não é obrigatório o uso do **elseif** ou **else**. O **if** isoladamente ou **ifs** aninhados também pode ser usado.

Exemplo

```php
<?php
$salario         = 1020;     // R$
$tempo_servico   = 12;       // meses
$tem_reclamacoes = false;    // boleano
if ($salario > 1000)
{
    if ($tempo_servico >= 12)
    {
        if ($tem_reclamacoes != true)
        {
            echo 'parabéns, você foi promovido';
        }
    }
}

if (($salario > 1000) and ($tempo_servico >= 12) and ($tem_reclamacoes != true))
{
    echo 'parabéns, você foi promovido';
}
```

1 – Estruturas Condicionais

if, else e elseif

Pode acontecer casos em que seja necessário verificar mais de uma condição e executar uma ação para cada uma, neste caso, é possível utilizar **elseif**.

```
1
2   if ( primeira condição ) {
3       Primeira ação
4   } elseif ( segunda condição ) {
5       Segunda Ação
6   } elseif ( terceira condição ) {
7       Terceira Ação
8   } elseif ( Quantas condições precisar ) {
9       Ação a ser executada
10  } else {
11      Ação padrão
12  }
```

1 – Estruturas Condicionais

if, else e elseif

Exemplo:

```php

<?php
$a = 10;

if ( $a == 1 ) {
    echo '$a vale 1';
} elseif ( $a == 2 ) {
    echo '$a vale 2';
} elseif ( $a == 10 ) {
    echo '$a vale 10';
}
?>
```

1 – Estruturas Condicionais

if, else e elseif

- O PHP oferece uma sintaxe alternativa para algumas das suas estruturas de controle: if, while, for, foreach e switch.

- Em cada caso, a forma básica da sintaxe alternativa é mudar o sinal de abertura para dois-pontos (:) e o sinal de fechamento para endif;, endwhile;, endfor;, endforeach; ou endswitch;, respectivamente.

Exemplos:

```
$a=7;
if ($a == 5):
  echo "a igual a 5";
elseif ($a == 6):
  echo "a igual a 6";
else:
  echo "a não é nem 5 nem 6";
endif;
```

```
<div>
   <?php if ( isset( $a ) ): ?>
      <p>$a existe.</p>
   <?php else: ?>
      <p>$a não existe.</a>
   <?php endif; ?>
</div>
```

1 – Estruturas Condicionais

if, else e elseif

- IF e else em uma linha

condição ? **ação do if** : **ação do else**;

Exemplo:

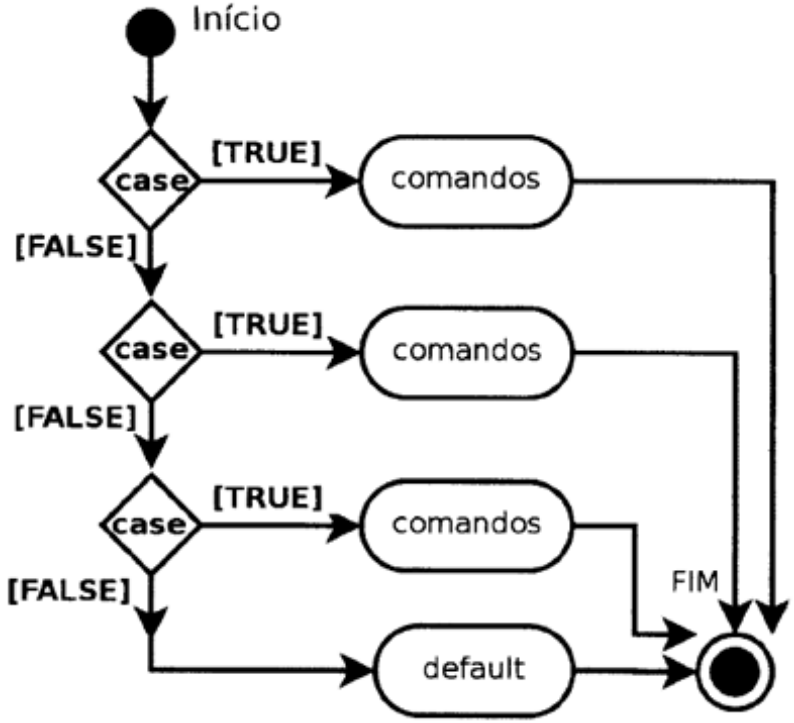
```php
<?php
$a = 10;
$b = isset( $a ) && !empty( $a ) ? $a * 2 : 10;

echo $b;
?>
```

1 – Estruturas Condicionais

Switch

A instrução switch é equivale a uma série de instruções **if's** seguidas, testando vários valores para uma mesma variável ou expressão.



1 – Estruturas Condicionais

Switch

A instrução switch é equivale a uma série de instruções **if's** seguidas, testando vários valores para uma mesma variável ou expressão.

```
$opcao = "";
switch($opcao){
        case 's':
                echo "Você escolheu SIM!";
                break;

        case 'n':
                echo "Você escolheu NÃO!";
                break;

        default:
                echo "A opção inválida";
                break;
}
```

1 – Estruturas Condicionais

If vs Switch

Uso do if

```
$numero = 2;

if($numero == 0){
        echo "O númeor é 0<br>";
}

elseif($numero == 1){
        echo "O númeor é 1<br>";
}

elseif($numero == 2){
        echo "O númeor é 2<br>";
}
```

Uso do switch

```
$numero = 2;

switch($numero){
    case 0:
        echo "O númeor é 0<br>";
        break;
    case 1:
        echo "O númeor é 1<br>";
        break;
    case 2:
        echo "O númeor é 2<br>";
        break;
}
```

2 – Manipulação de Strings

Uma string é uma cadeia de caracteres alfanuméricos. Para declarar uma string podemos utilizar aspas simples ‘’ ou aspas “”.

```
$variavel = 'Isto é um teste';
$variavel = "Isto é um teste";
```

A diferença é que todo conteúdo contido dentro de aspas duplas é avaliado pelo PHP. Assim, se a string contém uma variável, esta variável será traduzida pelo seu valor.

```
<?php
$fruta = 'maçã';
print "como $fruta";      // resultado 'como maçã'
print 'como $fruta';      // resultado 'como $fruta'
?>
```

2 – Manipulação de Strings

Também é possível declarar uma string literal com muitas linhas. Nesta caso é necessário escolher uma palavra chave para delimitar o início e fim da string.

Exemplo:

```php
<?php
$texto = <<< CHAVE
aqui nesta área
é possível escrever
textos com multiplas linhas
CHAVE;

echo $texto;
?>
```

2 – Manipulação de Strings

- **Concatenação**

Para concatenar strings, utiliza-se o operador “.” (ponto)

Exemplo:

```
<?php
$fruta = 'maçã';

// primeira forma
echo $fruta .' é a fruta de adão';   // resultado = maçã é a fruta de adão
```

2 – Manipulação de Strings

- **Concatenação**

Quando concatenamos uma variável que não seja string com uma outra string qualquer. O PHP automaticamente realiza a conversão entre tipos.

Exemplo:

```
<?php
$a = 1234;

echo 'O salário é ' . $a . "\n";
echo "O salário é $a \n";
?>
```

2 – Manipulação de Strings

- **Caracteres de escape**

Dentro de aspas duplas pode-se utilizar controles especiais interpretados diferentemente pelo PHP. Estes controles são chamados de caracteres de ecape (\).

Caractere	Descrição
\n	Nova linha, proporciona uma quebra de linha.
\r	Retorno de carro.
\t	Tabulação.
\\	Barra invertida "\\" é o mesmo que '\'.
\"	Aspas duplas.
\$	Símbolo de $.

2 – Manipulação de Strings

- **Caracteres de escape**

Exemplo:

```
<?php
        echo "Seu nome é \"Paulo\".";
?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **strtoupper**

Recebe um string como parâmetro e retorna o mesmo valor com todas as letras convertidas para maiúsculo.

Exemplo:

```
<?php
   /*string strtoupper (string conteudo)*/
   $nome = “gil jader";
   $nome_maisculo = strtoupper($nome);
   echo $nome_maisculo;
?>
```

SAÍDA:

GIL JADER

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **substr** substr(string conteudo, int inicio [,int comprimento])

Retorna uma parte de um string passado como primeiro parâmetro, a partir de uma posição e com certo comprimento.

Pode ser utilizada de três formas:

- informando apenas o início da parte
- informando o início e o comprimento
- informando o início e um valor negativo

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **substr**

Exemplos:

```php
<?php

    $nome = "Gil Jader";
    $parte = substr($nome, 3);
    echo $parte;

?>
```

```php
<?php

    $nome = "Gil Jader";
    $parte = substr($nome, 0, 5);
    echo $parte;

?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

  O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

  - **substr**

    Exemplos:

```php
<?php
    $nome = "Gil Jader";
    $parte = substr($nome, -5);
    echo $parte;

?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **str_pad**

Esta função recebe um string como parâmetro e retorna outro string com o conteúdo do primeiro complementado com alguns caracteres, de forma a atingir uma quantidade especificada.

  - STR_PAD_LEFT ou 0: preenche o string orinal à esquerda.
  - STR_PAD_RIGHT ou 1: complementa o string original à direita.
  - STR_PAD_BOTH: preenche ambos os lados do string original.

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **str_pad**

Exemplo:

```php
<?php

    $nome = "PHP";
    $nome_completo = str_pad($nome, 5, "*");
    echo $nome_completo;

?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **str_pad**

Exemplo:

```php
<?php
    $nome = "PHP";
    $left = str_pad($nome, 5, "*", STR_PAD_LEFT);
    $right = str_pad($nome, 5, "*", STR_PAD_RIGHT);
    $both = str_pad($nome, 5, "*", STR_PAD_BOTH);
    echo $left;
    echo "<br />";
    echo $right;
    echo "<br />";
    echo $both;
?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **str_replace**

Utilizada para substituir um trecho de um texto por outro valor.

```php
<?php

    $texto = "Olá, mundo.";
    $novo_texto = str_replace("mundo", "leitor", $texto);
    echo $novo_texto;

?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **stripslashes**

Retorna uma string com as barras invertidas retiradas.

Exemplo:

```php
<?php

$str = "\'DELETE FROM TABELA;'";

echo stripslashes($str);

?>
```

2 – Manipulação de Strings

- **Funções para manipulação de Strings**

O PHP disponibiliza uma variedade de funções para manipulação de cadeias de caracteres (strings).

- **Strip_tags**

Retira as tags HTML e PHP de uma string.

Exemplo:

```
<?php
$text = '<p>Test paragraph.</p><!-- Comment --> <a href="#fragment">Other text</a>';
echo strip_tags($text);

?>
```

3 – Manipulação de arrays

- Arrays em PHP são listas de valores que podem ter diferentes tipos de dados;
- Arrays são acessados mediante uma posição, por exemplo, um índice numérico. Para criar um array, pode-se utilizar a função array([chave=>] valor, ...);
- Pode crescer dinamicamente com adição de novos itens;

3 – Manipulação de arrays

Exemplo:

```php
<?php
        //Criação do Array
        $carros = array('Palio','Corsa','Gol', "siena" =>"Siena");

        echo $carros[1] . "<br>"; //Resultado Corsa

        echo $carros["siena"] . "<br>"; //Resultado Siena
?>
```

3 – Manipulação de arrays

Exemplo:

```
//Adicionando novos elementos
$carros[4] = "Clio";
$carros[11] = "Versa";
$carros[ ] = "Sandero";


echo $carros[4] . "<br>"; //Resultado Clio
echo $carros[5] . "<br>"; //Sem Resultado
echo $carros[11] . "<br>"; //Resultado Versa
echo $carros[12] . "<br>"; //Resultado Sandero
```

Usando o colchetes sem índice ([ ]) o PHP procurará o último índice utilizado e o incrementará

3 – Manipulação de arrays

- Também pode ser usada um string como índice, neste caso chamada de chave associativa

Exemplo:

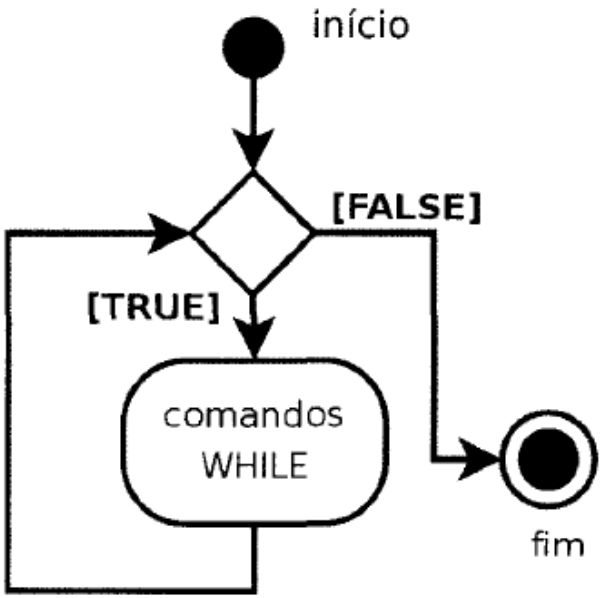
```
$carros["popular"] = "Fusca";
$carros["quantidade"] = 7;

echo $carros["popular"] . "<br>"; //Resultado Fusca
echo $carros["quantidade"] . " carros no array <br>"; //Resultado 7
```

4 – Laços de Repetição

while

Estabelece um laço de repetição que será executado repetitivamente enquanto a condição de entrada dada pela expressão for verdadeira.



4 – Laços de Repetição

while

- Sintaxe

```
while ( exp )
{
<comandos>
}
```

- Sintaxe alternativa

```
while ( exp ):
        <comandos>
endwhile;
```

Exemplo:

```
while($cont<100) {
        echo "O valor atual do contador é $cont <br>";
        $cont++;
}
```

4 – Laços de Repetição

do..while

- A única diferença entre o while e o do...while é que o while avalia a expressão no início do laço e o do...while ao final.

- Vai ser executado ao menos uma vez e caso utizar o while não seria executado nenhuma vez

Sintaxe

```
do
{
        <comandos>
} while ( exp )
```

4 – Laços de Repetição

do..while

Exemplo:

```
$numero = 1;

do
{
echo "O valor atual do contador é $cont <br>";
$numero++;

}while($numero<15);
```

4 – Laços de Repetição

for

- Sintaxe

```
for (inicialização ; condição; operador)
{
<comandos>
}
```

- Sintaxe Alternativa

```
for (inicialização ; condição; operador):
<comandos>
endfor;
```

4 – Laços de Repetição

for

Exemplos:

```
echo "Contagem Progressiva <br> <br>";

for($cont=0;$cont<10;$cont++){
        echo "A variável \$cont vale $cont
<br>";
}


echo "<br> Contagem Regressiva <br> <br>";
for($cont=13;$cont>0;$cont--){
        echo "A variável \$cont vale $cont
<br>";
}
```

4 – Laços de Repetição

for

Muito útil para realizar laços aninhados, por exemplo, para trabalhar com ***arrays*** bidimensionais.

Exemplo:

```
$vetor[0][0] = "elemento00";
$vetor[0][1] = "elemento01";
$vetor[1][0] = "elemento10";
$vetor[1][1] = "elemento11";

for($contI=0;$contI<2;$contI++){
        for($contJ=0;$contJ<2;$contJ++){
                echo "O valor do vetor é " . $vetor[$contI][$contJ];
                echo "<br>";
        }
}
```

4 – Laços de Repetição

foreach

Oferece uma maneira mais fácil de “navegar” entre os elementos de um array.

```php
<?php

foreach($nome_array as $elemento)
{
<comandos>
}

?>
```

Todos os itens de **$nome_array** serão visitados. A cada iteração o item da vez será armazenado em **$elemento**. Assim é possível trabalhar todos os elementos usando somente uma **variável.**

4 – Laços de Repetição

foreach

O PHP também permite realizar foreach conforme abaixo:

```php
<?php

 foreach($nome_array as $chave => $valor)
{
    <comandos>
}

?>
```

Essa segunda sintaxe funciona da mesma forma porém enquanto o elemento é adicionado $valor, o índice atual é atribuído a $chave

4 – Laços de Repetição

foreach

Exemplos:

```
$vetor = array (1,2,3,4,5);
foreach($vetor as $v)
{
print "O valor atual do vetor é $v. <br>";
}
```

```
$a = array ("um"=>1, "dois"=>2, "tres"=>3);
foreach($a as $chave => $valor)
{
print("\$a[$chave] => $valor.<br>");
}
```

5 – Break e Continue

Break

A instrução **break** finaliza a execução da estrutura for, foreach, while, do-while ou switch atual.

break aceita um argumento numérico opcional que diz quantas estruturas aninhadas deverá interromper. O valor padrão é 1, somente a estrutura imediata é interrompida.

5 – Break e Continue

Break

Exemplo:

```php
<?php
$i = 0;
while (++$i) {
    switch ($i) {
    case 5:
        echo "At 5<br />\n";
        break 1;  /* Sai somente do switch. */
    case 10:
        echo "At 10; quitting<br />\n";
        break 2;  /* Sai do switch e do while. */
    default:
        break;
    }
}
?>
```

5 – Break e Continue

Continue

Assim como **break**, a instrução **continue** permite que a execução do loop seja alterada, mas diferente do break não encerramos o loop, apenas informamos ao PHP para encerrar a interação atual e iniciar a próxima.

O continue também aceita um argumento numérico opcional que diz quantos níveis de laços aninhados deve pular. O valor padrão é 1, saltando para o final do laço atual.

5 – Break e Continue

Continue

Exemplo:

```php
<?php

for ($i=0; $i <= 10; $i++){
    if ($i%2 == 0) //se i for par (divisível por 2)
        continue;  //ignora o resto do for e vai para o próximo número
        echo ($i . " ");
}

?>
```

resulta em: 1 3 5 7 9

5 – Break e Continue

Continue

Outro Exemplo:

```php
<?php
$i = 0;
while ($i++ < 5) {
    echo "Fora<br>\n";
    while (1) {
        echo "  Meio<br>\n";
            while (1) {
                echo "  Dentro<br>\n";
                continue 3;
            }
        echo "Isto nunca será exibido.<br>\n";
    }
    echo "Nem isso.<br>\n";
}
?>
```

6 – Variáveis de Sessão

As variáveis de sessão podem ser utilizada para realizar o controle de acesso aos Sistemas desenvolvidos em PHP.

Uma variável de sessão é uma variável super global. Isto significa que ao ser utilizada, estará disponível em todos escopos pelo script PHP.

6 – Variáveis de Sessão

- Para iniciar a sessão:

session_start ()

- Para estabelecer e usar a variável de sessão:

$_SESSION[‘nome’]=“gil.jader”;

- Para encerrar a sessão:

unset($_SESSION['nome']);

Exercícios

7 – Referências

- Dall'Oglio, Pablo. Php 5 - PHP: Programando com Orientação a Objetos. 3ª Edição, São Paulo, Novatec, 2015.

- WALLACE, Soares. Php 5 - Conceitos, Programação e Integração com Banco de Dados. 7ª Edição, São Paulo, Erica, 2013.

- http://www.php.net/manual/pt_BR/index.php