Modulo 01

O que é de fato PHP

* Uma linguagem de programação que foi desenvolvida para deixar o HTML dinâmico
* Linguagem de script e open source
* O foco de PHP é o desenvolvimento web
* Os programas são executados em server side (lado do servidor backend)
* PHP significa personal home page (P) e hypertext prepocessor (HP)

Uma breve história

* O criador da linguagem foi Rasmus Lerdorf
* O ano de lançamento foi 1994
* A linguagem já está na versão 8
* Diversas melhorias foram implementadas
* A ideia principal era deixar o HTML dinâmico não havia pretensão de ser o que é hoje
* Quase 80% dos sites hoje contém PHP

O que é localhost

* É a forma de acessar o servidor local da nossa máquina
* Equivale ao IP 127.0.0.1, ou seja, nosso PC
* Assim podemos simular como se o site ou arquivo que estamos acessando é processado em um servidor
* IP significa internet protocol

A sintaxe do PHP

* O PHP vai interpretar um bloco de código em nosso arquivo apenas se ele estiver entra as tags de PHP
* A abertura é: <?php
* O fechamento é: ?>
* Coloque ; a cada instrução
* Todo o código dentro destas tags será executado peo PHp e após isso sera impressona tela

PHP e suas dependências

* Podemos checar com o PHP e suas dependências estão instaladas no nosso computador
* O nome da função é phpinfo()
* Ela exibe as versões também dos pacotes instalados
* Útil para saber como o servidor está configurado

Modulo 02

Case sensitivity

* Significa sensibilidade a casas maiúsculas e minúsculas
* Para instruções PHP não temos essa diferença ou seja: echo == ECHO
* Porém, para variáveis são case sensitive: $name != $NAME

Instruções de código

* As instruções simples de PHP são separados por ;
* Instruções simples são instruções de uma linha
* Quando há uma instrução maior, como de condição ou repetição, a definição da mesma é dada por abertura e fechamento de chaves
* Nestes casos, não precisaremos de ponto e vírgula

Espaços em branco

* Para interpretação do código em PHP o espaço em branco é ignorado
* Isso acontece pois o mesmo é removido antes da execução
* A quebra de linha também é ignorada
* Porém se utilizada de má forma pode gerar erros inesperados no código

Comentários

* Os comentários servem para dar informações e direções importantes de como o código funciona
* Iniciamos um comentário com //
* Todo conteúdo que está em um comentário é ignorado na execução
* Não insira informações sensíveis nos comentários
* Outra forma de inserir comentários é com #
* Comentários multilinhas são feitos com /\* comentário \*/

Palavras reservadas

* Algumas palavras são reservadas da linguagem e já tem suas funcionalidades definidas, então não podemos utilizar em nossos programas
* Pois caso fosse possível, poderíamos substituir a sua função original
* Alguns exemplos são: echo, insteadof, else, interface, namespace, pow, \_\_DIR\_\_, \_\_FILE\_\_, endif, print, private, protected, and, require, public, as, break, case, for, final, switch, throw e etc

Modulo 3

Tipo de dados: inteiros (integers)

* Os inteiros são os números inteiros da matemática como: 1,2,15
* Incluindo números negativos
* Os números positivos não precisam de um sinal de + na frente
* Já os números negativos devem ser descritos assim ex: -12

Checando número inteiro

* Podemos validar se um dado é inteiro com a função is\_int()
* Caso um número seja inteiro, será retornado true (um outro tipo de dado)
* Caso não seja, receberemos um retorno de false (tipo de dado também)
* Precisamos utilizar uma estrutura if para validar o valor

Tipo de dados: números decimais (floats)

* Os floats são todos os números com casas decimais
* Como o padrão universal é a língua inglesa, temos a separação das casas decimais com . (ponto) e não , (virgula)
* PHP permite a soma de números de conjuntos diferentes ex: 10 + 50.12
* Exemplos de floats: 2.133, 0.04, 12.8

Checando se é float

* Podemos utilizar a função is\_float para verificar se um dado é um float
* A função recebe um valor como parâmetro
* Novamente recebemos true or false, dependendo do dado enviado
* Precisamos utilizar uma estrutura if para validar o valor

Tipo de dados: texto (strings)

* Os textos são conhecidos como strings
* Em PHP podemos escrever textos em aspas simples ou duplas, não há diferenã para texto puro
* As aspas duplas interpretam variáveis

Checando se é string

* Podemos utilizar a função is\_string() para verificar se um dados é uma string
* A função recebe um valor como parâmetro
* Novamente recebemos true or false, dependendo do dado enviado
* Precisamos utilizar uma estrutura if para validar o valor

Tipo de dados: booleanos

* O boolean é um tipo de dado que só possui dois valores
* True: verdadeiro
* False: falso
* Alguns valores são considerados como falso: 0, 0.0, “0”, [], NULL

Checando se é booleano

* Podemos utilizar a função is\_bool() para verificar se um dado é um boolean
* A função recebe um valor como parâmetro
* Novamente receberemos true or false, dependendo do dado enviado
* Precisamos utilizar uma estrutura if para validar o valor

Tipo de dados: array (conjunto, lista)

* O array é um tipo de dado que serve para agrupar um conjunto de valores
* Podemos inserir qualquer tipo de dado na lista
* A sintaxe é: [1,2,3,4,5 ...]
* Sempre entre [] dados separados por , (virgula)
* Print\_r(): função para imprimir o array completo
* Echo $array[1]: echo pode ser usado para imprimir valores expecificos do array

Tipo de dados: array associativo

* O array associativo é basicamente um array, porém com chave e valor
* A estrutura base é a mesma mas vamos construir dessa maneira
* $arr = [‘nome’=> ‘daniel’, ‘idade’=>26]
* Chave entre aspas, seta para apontar o valor

Tipo de dados: objetos

* o PHP possui o paradigma de orientação a objetos
* podemos criar classes e objetos, e o objeto é considerado um tipo de dado
* objetos possuem métodos que são suas ações e propriedades que são características

tipo de dados: null

* o tipo de dado null tem apenas um valor o NULL
* um caso de uno de null seria chacar se uma variável tem ou não valor
* podemos checare se um valor é null com iss\_null()

modulo 4

sobre as variáveis

* são a forma que temos para declarar um valor e salvá-lo na memoria
* uma variável em PHP tem o $ na frente do seu nome
* ex: $nome = ‘Daniel’;
* podemos salvar qualquer tipo de dado
* podemos alterar o valor de uma variável no decorrer do programa
* podemos imprimir o valor de uma variável com echo

variável de variável

* podemos criar uma variável por meio do nome de outra variável com um valor diferente
* o símbolo para esta função é o $$ (cifrão duplo)
* $a = ‘teste’;
* $$x = 5;
* Após a execução do código, a variável teste (conteúdo de $x), será criada com o valor

Variável por referencia

* Podemos criar uma variável com referencia a outra
* O símbolo é =&
* Se mudamos a variável de referencia a referenciada muda o valor e ao contratrio também gera a mudança
* $x = 2
* $y =& $x

Escopo

* Como em outras linguagens em PHP, também temos escopo de variáveis
* Local: variável declarada em uma função
* Global: variáveis declaradas fora de funções
* Static: variável declarada dentro da função, porem o seu valor permanece salvo entre chamadas da função
* Parâmetros de função: variáveis passadas para um função, podendo ser utilizadas ao longo da mesma

Escopo: Variável local

* A variável local tem seu escopo definido dentro de uma função
* Ela não é acessível fora da mesma
* O seu valor sempre é resetado quando a função é finalizada

Escopo: varialvel global

* A principal característica da variável global é ser declarada fora de funções
* Por comportamento padrão não são acessíveis dentro de funções
* Precisamos utilizar a palavra global para isso
* Essa função de variável global não ser acessível dentro de funções previne muitos problemas no software
* Para acessar variável global dentro de local, usar função global ex: global $teste;

Escopo: variavel estática

* A variável estática é declarada com a instrução static
* O valor da mesma é mantido e alterado a cada execução de uma função
* É interessante esse comportamento pois as variáveis de escopo local sempre são resetadas

Escopo: parâmetros de função

* Os parâmetros de função também são considerados tipos de variáveis
* Este recurso nos ajuda a criar funções com valores dinâmicos
* Podendo alterá-los a cada invocação da mesma
* Podemos passar mais de um parâmetro para uma função

Modulo 5

O que é uma expressão?

* Uma instrução de código que sera avaliada e resultara em um valor
* Uma simples impressão de um texto é uma expressão
* Uma soma ou operação matemática mais complexa tambe
* Na programaão realizaremos diversas expressões durante nosso código para formar nosso software