Livros de PHP

Aprendendo PHP (por David Sklar) – Ed. O'Reilly

Programing PHP: Creating Dynamic Web Pages (por Kevin Tatroe e Peter MacIntyre) – Ed. O'Reilly

Learn PHP 8 (por Steve Prettyman) – Ed. Apress

PHP 8 Quick Scripting Reference: A Pocket Guide to PHP Web Scripting (por Mikael Olsson) – Ed. Apress

Só o PHP é server-side?
Microsoft .NET = solução server-side para servidores window, linguagens como ASP.Net, C#

Linguagem Perl = 1978, linguagem multi plataforma, open source e roda no lado do servidor

Java Server Page = Java JSP

JavaScript = Deno, node e outras soluções online, Server Side JavaScript

Python = Django, Flask

Ruby = Ruby on Rails

PHP & MySQL: Server-Side Web Development (Jon Duckett) – Ed. Wiley

As versões do PHP

1995 = PHP1.0

- Personal Home Page;
- Funcionalidades simples para um site básico;

- Não tinha características de linguagem em si.

1997 = PHP 2.0

- Oficialmente nomeado PHP/FI 2.0;
- Linguagem standalone;
- Recursos limitados.

1998 = PHP 3.0

- Primeira versão colaborativa;
- características mais relacionadas a uma linguagem;
- Extension API.

2000 = PHP 4.0

- Zend Engine 1.0;
- Melhoria de performance;
- Aumento de modularização;
- Uso de super globais $_GET, $_POST, $_SESSION, etc.

2004 = PHP 5.0

- Zend Engine 2.0;
- Primeira Versão com Orientação PHP Data Objects (PDO);
- Operador de exponenciação;
- Suporte a Json;
- Namespaces, closures, garbage collection, exceptions handling;
- Novo depurador (phpdgb).

2005 = PHP 6.0

- Aumento da segurança e performance;
- Suporte nativo a Unicode;
- Até existiram algumas versões;
- Nunca recebeu uma release oficial.

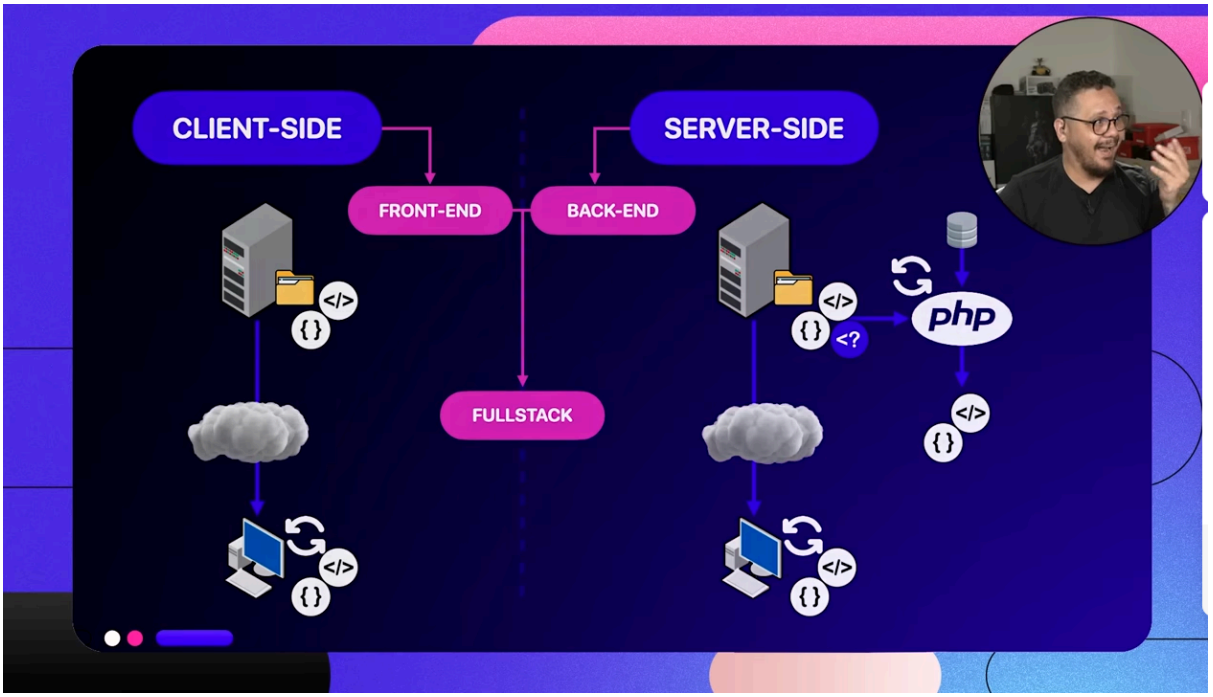
2015 = PHP 7.0

- Zend Engine 3.0;

- Performance até 9x maior;
- Escape de códigos Unicode;
- Operador null coalescing (??)
- Declaração escalar;
- Operador spaceship (<=>) ;
- Classes anônimas;
- Exceções mais modernas;
- Retorno de void;
- Libsodium;
- Foreign function interface.

2020 = PHP 8.0

- Zend Engine 4.0;
- Just-In-Time compilation;
- Locale-independent number convertions;
- Named arguments;
- Expressão match;
- Operador null safe;
- Enumerations;
- Readonly properties;
- Fibers.



Regras para nomes identificadores

- Variáveis sempre começam com o **símbolo $**
- O segundo pode ser **letra** ou **simbolo _**
- Aceita caracteres [**a-z**],[**A-Z**],[**0-9**] e [**_**]
- Aceita caracteres da tabela **ASCII** a partir de **128**
- Aceita caracteres acentuados como **á,õ,ç**
- A linguagem é **case sensitive** em relação aos nomes
- Nomes especiais com **$this** não podem ser usados

Recomendações para dar nomes

- Tente dar nomes **claros** e de **fácil** identificação
- Evite nomes muito **curtos** ou muito **longos**
- Defina um **padrão** e siga em todo o projeto
- Para **variáveis**, de preferências a letras **minúsculas**
- Para **constantes**, de preferência a letras **maiúsculas**
- Use **camelCase** para métodos e atributos
- Use **SNAKE_CASE** para nomear constantes

CATEGORIAS DOS TIPOS PRIMITIVOS

- ESCALARES
- COMPOSTOS
- ESPECIAIS

TIPOS PRIMITIVOS ESCALARES

- **STRING:** $sobrenome = “Silva” (sequência de letras, números e símbolos, sempre representadas entre aspas.)
- **INT ou INTEGER** $idade = 34 (um valor numérico inteiro, aquele que vem sem a parte decimal)
- **FLOAT, DOUBLE ou REAL** $peso = 85.9 (um valor numérico Real, que vem com a parte decimal, depois do ponto flutuante)

- **BOOL ou BOOLEAN** $casado = true (um valor lógico ou Booleano, que aceita apenas os valores verdadeiro ou falso (true ou false))

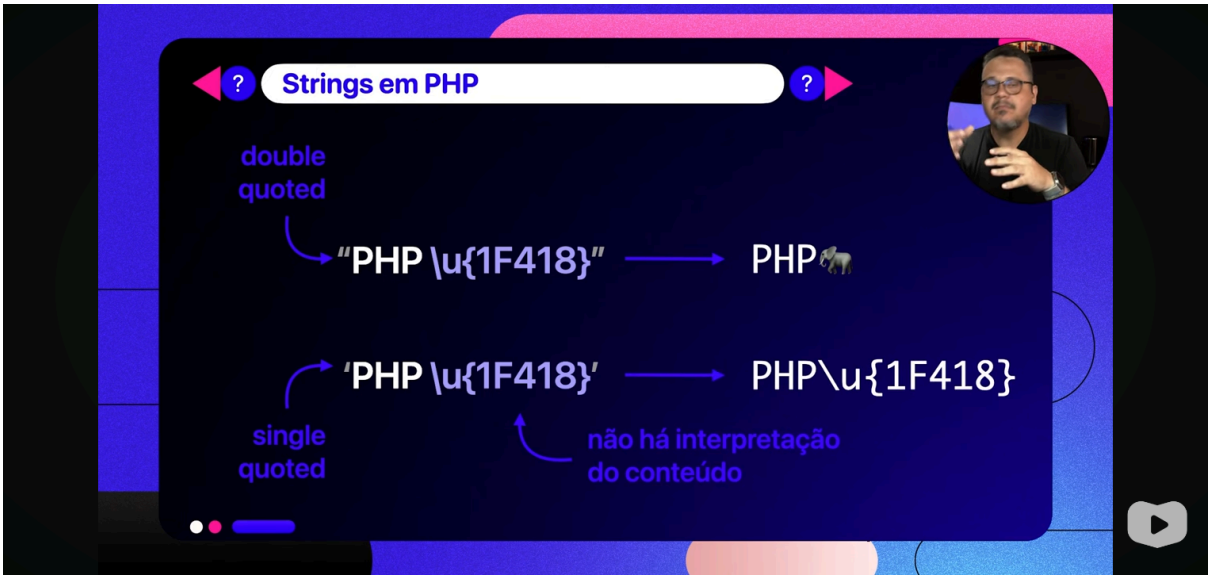
TIPOS PRIMITIVOS COMPOSTOS

- **Array**
- **Object**

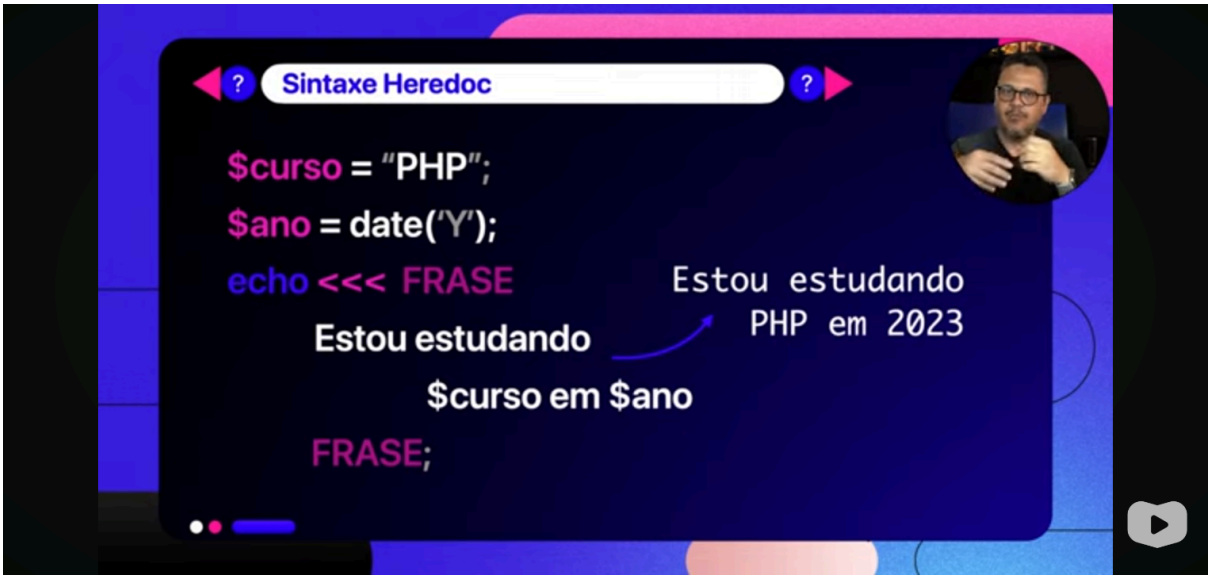
TIPOS PRIMITIVOS ESPECIAIS

- **null**
- **resource**
- **callabe**
- **mixed**

formatos de strings em PHP



double quoted = Aspas Duplas
single quoted = Aspas Simples
Heredoc = aceitam caracteres múltiplas linhas via terminal, estão mais associadas a double quoted por ter interpretação dentro dela



Nowdoc = aceitam caracteres múltiplas linhas via terminal, estão mais associadas a single quoted por NÃO ter interpretação

double quoted = "Curso" . "PHP"
concatenation operator = .(ponto)
double quoted = existe interpretação dentro das strings

single quoted = não há interpretação do conteúdo

\' = sequência de escape single quoted

Control + ; = transforma em comentário
Sequência de escapes double quoted

\n Nova linha
\t Tabulação horizontal
\\ Barra invertida
\$ Sinal de cifrão
\u{} Codepoint Unicode

obs: só funciona na saída textarea, saída HTML não funciona, testar no php online

Array SuperGlobal ($_GET, $_POST, $_REQUEST=junção das superglobais $_GET + $_POST + $_COOKIES)

. = Operador de concatenação

Operadores Aritméticos
+ = Operador de Adição
- = Subtração
* = Multiplicação
/ = Divisão
% = Módulo(resto da divisão)
** = Exponenciação

Ordem de Precedência
Expressões que estiverem entre () tem maior precedência, na ausência da mesma segue a ordem:
1° **= Exponenciação
2° *= Multiplicação ou /= Divisão ou %= Módulo (O que vier 1° na expressão)
3° += Adição ou -= Subtração (O que vier 1° na expressão)

Funções Aritméticas
abs = Valor Absoluto(valor sem o sinal de + ou -)
base_convert() [conversor de bases]
ceil() = função de arredondamento para cima
floor() = função de arredondamento para baixo
round() = função de arredondamento aritmético
hypot() = função para calcular a hipotenusa a partir do valor dos catetos
intdiv() = divisão inteira
min(), max() = mostra o valor mínimo e máximo de uma sequência
pi() = valor de PI 3,1415 ou constante M_PI
pow() = função de potência, usando esta função em vez do operador ** perde-se a ordem de precedência
sin(), cos(), tan() = Funções trigonométricas (Seno, Cosseno e Tangente)
sqtr() = função que calcula a raiz quadrada

valor ** (1/3) = fórmula para calcular raíz cúbica
calcular porcentagem = (($porcentagem * $preco)/100) + $preco;

Formatação de números e moedas
<input type="number" step="0.01" name="numero" id="numero"> // step="0.01" é necessário para que aceite valores decimais ex:1,50
number_format($valor, 2, ",",".") = formatação de números em php onde $valor(é a variável com o valor numérico), 2(quantidade de casas decimais), ","(separador decimal),"."(separador milhar)

$padrão = numfmt_create("pt_BR", NumberFormatter::CURRENCY); = Formatação de moedas com internacionalização / Biblioteca intl (Internallization PHP), "pt_BR"(informa em que idioma está o site) NumberFormatter::CURRENCY (tipo de número que vou formatar)

numfmt_format_currency($padrao, $valor, "BRL") = currency(valor monetário), $padrão (variável), $valor(variável que carrega o número), "BRL"(sigla internacional que vc quer formatar a moeda)

SuperGlobais

$_GET = gera um array associativo e todas as variáveis que foram passadas por parâmetro através de uma url | QUERY STRING = http://localhost/cursophp/exercicios/ex005/superglobais.php?nome=Mateus&idade=27&cep=0178350 , A partir da ? é considerado query string, uma solicitação na url

$_POST = Ela pega dados de formulários que foram enviados utilizando o método POST
$_REQUEST = É a junção do GET + POST

$_COOKIES = peça informações sobre os cookies da sua aplicação PHP(pequenas variáveis que são guardadas dentro do seu navegador por um determinado tempo)
$_FILES = resgata os dados de um possível upload de arquivo

$_SESSION = permite que variáveis de sessões que não são perdidas entre a navegação entre páginas possam ser usadas
$_ENV(enviroment significa ambiente) = Variáveis de ambiente do servidor, funciona no php online
$_SERVER =
$GLOBALS = A super global mais completa, pois a mesma mostra os dados das outras superglobais, joga dentro de um array os dados de todas as outras superglobais

shift + alt + seta para baixo = duplica os códigos que foram selecionados
alt + botão esquerdo do mouse, vc selecionar e alterar no código de uma vez só em vários locais
date('Y') pega o ano atual no formato 2024, date('y') pega o ano atual no formato 24

usando input type="range" no html
<label for="por">Qual será o percentual de reajuste? (?%)</label>
<input type="range" name="por" id="por" min="0" max="100" step="1" oninput="mudaValor()" value="<?=$porcentagem?>">
até aqui é dentro do forms

json_decode(); = tratar dados em Json

```
<script>
        mudaValor()

        function mudaValor(){
            p.innerText = por.value;
        }
    </script>
```

tag javascript para interação na tela

1 minuto = 60 segundos
1 hora = 60 minutos * 60 = 3.600 segundos

1 dia = 24 horas * 60 = 1.440 minutos * 60 = 86.400 segundos
1 semana = 7 dias * 24 = 168 horas * 60 = 10.080 minutos * 60 = 604.800 segundos

semana 2.000.000 / 604.800 = 3 semanas inteiras e sobra 185.600 segundos
dias 185.600 / 86.400 = 2 dias inteiros e sobra 12.800 segundos
horas 12.800 / 3.600 = 3 horas inteiras e sobra 2.000 segundos
minutos 2.000/60 = 33 minutos inteiros e sobra 20 segundos
segundos = 20 segundos