

# Técnico em Desenvolvimento de Sistemas

---

Desenvolvimento de sistemas web, mobile e desktop



# Sobre o Instrutor



- Formado em Ciências da Computação e pós-Graduado em Sistemas de Informação.
- Atuo como Gerente de TI do Coren-DF.
- Atuei por mais de 15 anos como professor universitário.
- Atuei por 5 anos como instrutor de cursos técnicos e de qualificação profissional no Senac.
- Grupo WhatsApp:  
<https://chat.whatsapp.com/lquYhphWzqI3aCg1wQQQPv>



# Sobre o curso

- Dividido em 12 Unidades Curriculares - UC



## TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

**Título do Curso:** Técnico em Desenvolvimento de Sistemas

**Competência:** Desenvolve competência.

**Tipo de Curso:** Habilitação Profissional Técnica de Nível Médio

**Eixo Tecnológico:** Informação e Comunicação

**Segmento:** Tecnologia da Informação

**Carga Horária:** 1.200 horas

**Código DN:** 1730

**Código CBO:** CBO associada – 3171-10 - Desenvolvedor de Sistemas de Tecnologia da Informação

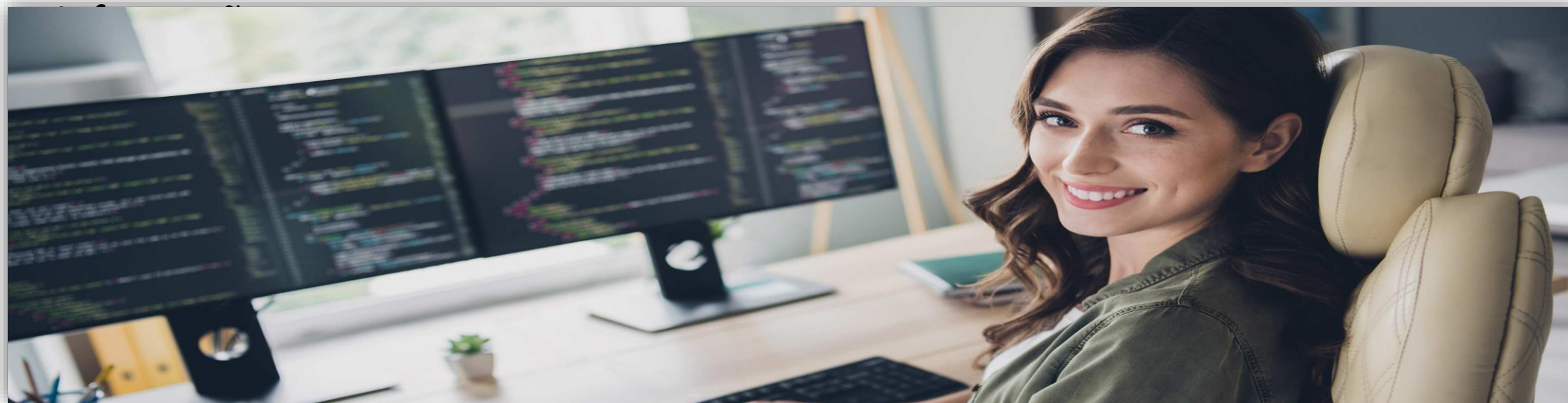
Unidades Curriculares		Carga horária
UC12: Projeto Integrador Desenvolvedor de aplicações (60 horas)	UC1: analisar requisitos e funcionalidades da aplicação	108 horas
	UC2: auxiliar na gestão de projetos de Tecnologia da Informação	60 horas
	UC3: Desenvolver algoritmos	108 horas
	UC4: analisar programação estruturada e orientada a objetos	48 horas
	UC5: Desenvolver aplicações <i>desktop</i>	140 horas
	UC6: Criar e manter Banco de Dados	108 horas
	UC7: Desenvolver aplicações <i>web</i>	140 horas
	UC8: Desenvolver aplicações <i>mobile</i>	140 horas
	UC9: Realizar operações de atualização e manutenção em aplicações desenvolvidas	96 horas
	UC10: Realizar testes nas aplicações desenvolvidas	108 horas
	UC11: Realizar operações de suporte junto ao usuário	84 horas
<b>Técnico em Desenvolvimento de Sistemas (UC1 a UC12)</b>		<b>1.200 horas</b>

# Sobre a Unidade Curricular 07

- Unidade Curricular 7: Desenvolver aplicações *web*
- Carga horária: 140 horas.
- Define IDE conforme aplicação a ser desenvolvida.
- Escreve linhas de código conforme requisitos do projeto.
- Aplica características de estilo da aplicação web.
- Realiza a conexão da aplicação com o banco de dados de acordo o Sistema Gerenciador de Banco de Dados definido.
- Testa o código desenvolvido conforme os requisitos do projeto.
- Versiona e disponibiliza a aplicação desenvolvida de acordo com as melhores práticas do mercado.

# Profissional

- O profissional Técnico em Desenvolvimento de Sistemas é responsável pelo desenvolvimento de sistemas web, mobile e desktop, e nesse contexto realiza a análise de requisitos, desenvolve, testa e publica as aplicações.
- Executa operações de manutenção de aplicações já desenvolvidas, fornece suporte junto ao usuário e auxilia na gestão dos projetos de Tecnologia da





# Marcas formativas

- Esse conceito orienta toda a formação e busca desenvolver, junto aos alunos, características que os diferenciam enquanto profissionais formados pelo Senac e pelas quais serão reconhecidos no mercado de trabalho, a saber: domínio técnico-científico; visão crítica; atitude sustentável; colaboração e comunicação; criatividade e atitude empreendedora; autonomia digital.



# Metodologia

- O núcleo da proposta metodológica que orienta a prática nos ambientes de aprendizagem Senac organiza-se a partir do conceito de ação-reflexão-ação, no qual se aprende fazendo e analisando o próprio fazer por meio de atividades que buscam articular a realidade do mundo do trabalho com as experiências prévias dos alunos, possibilitando uma aprendizagem significativa, que supera o paradigma tradicional, outrora focado na transmissão de conteúdo.



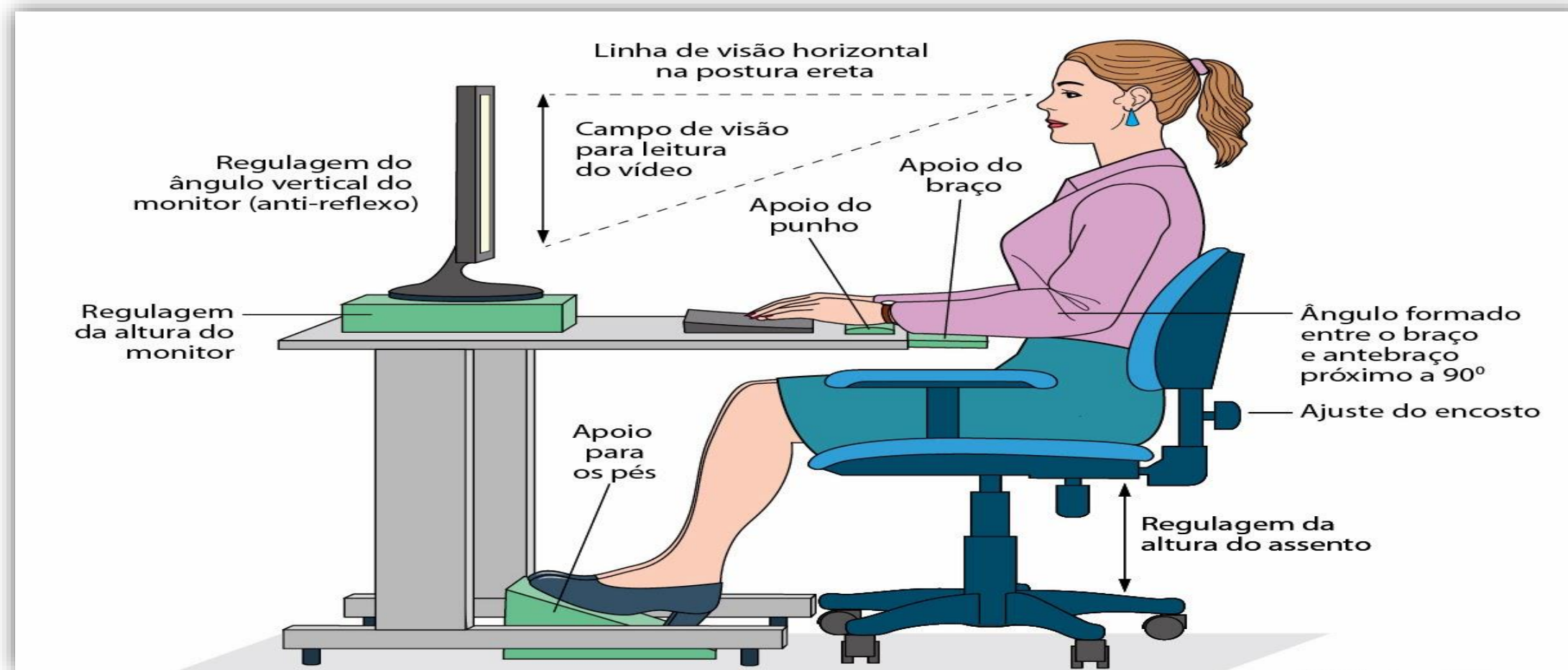
# Ligar e desligar o computador

- Ligar
  - Ligar o estabilizador
  - Ligar o computador
  - Ligar o monitor
- Desligar
  - Iniciar/Desligar
  - Desligar monitor
  - Desligar estabilizador
  - Colocar a cadeira no lugar





# Dicas Gerais – Boa Postura



# Desafios

- Desenvolver o Back-end do PI.

# Livros



# Livros e Autores

- Bibliografia Básica
  - Lacerda, Ivan Max Freire de. Programador Web: Um guia para programação e manipulação de bancos de dados. 3. ed. São Paulo: Senac São Paulo, 2016. 174 p. v. 1. E-book (174 p.). (Biblioteca Digital)
- Bibliografia Complementar
  - GOMES, Ana Laura. XHTML/CSS: Criação de páginas web. 3. ed. São Paulo: Senac São Paulo, 2010. 203 p. v. 1. E-book (203 p.). (Biblioteca Digital)
  - PUREWAL, Semmy. Aprendendo a desenvolver aplicações web. 1. ed. São Paulo: Novatec, 2014. 360 p. v. 1.

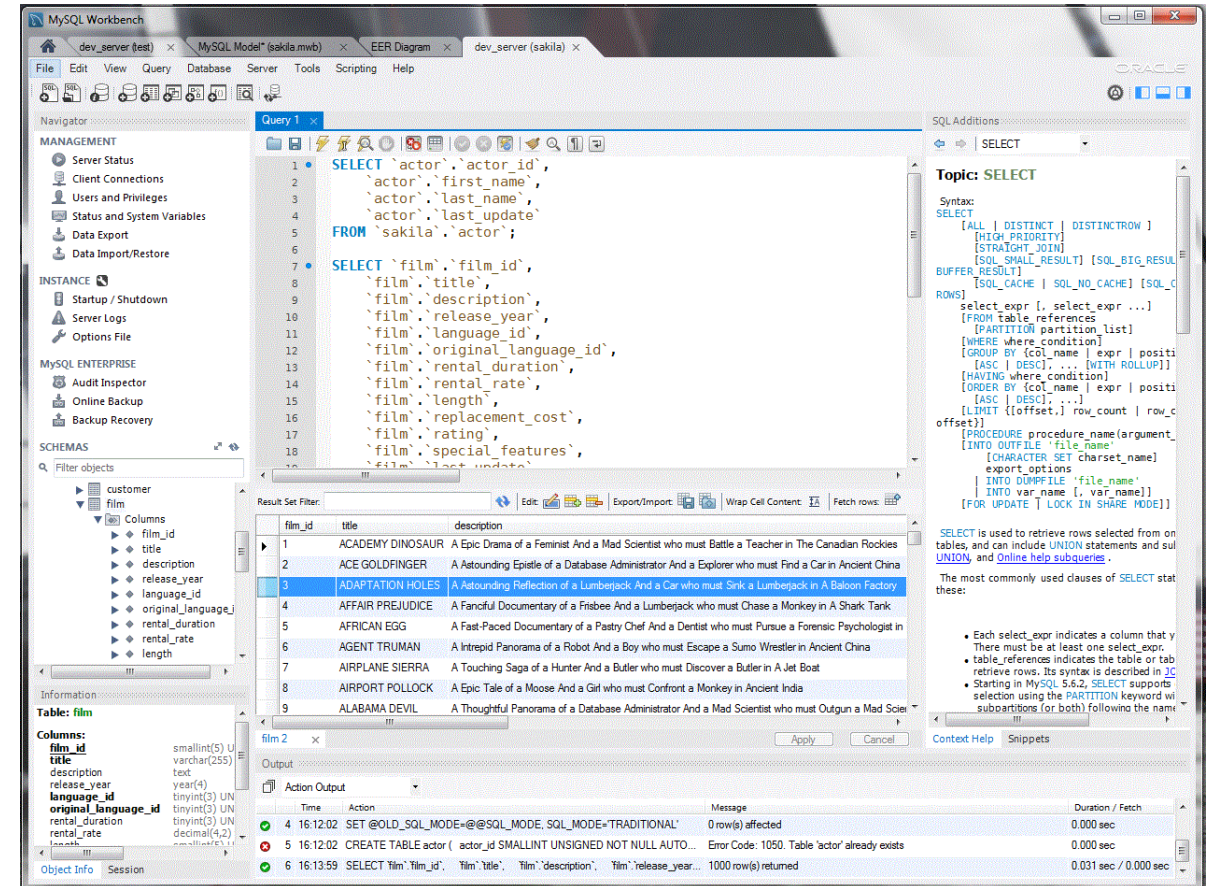
# Programas





# MySQL Workbench

- MySQL Workbench é uma ferramenta de design de banco de dados visual que integra desenvolvimento SQL, administração, design de banco de dados, criação e manutenção em um único ambiente de desenvolvimento integrado para o sistema de banco de dados MySQL.



# MySQL

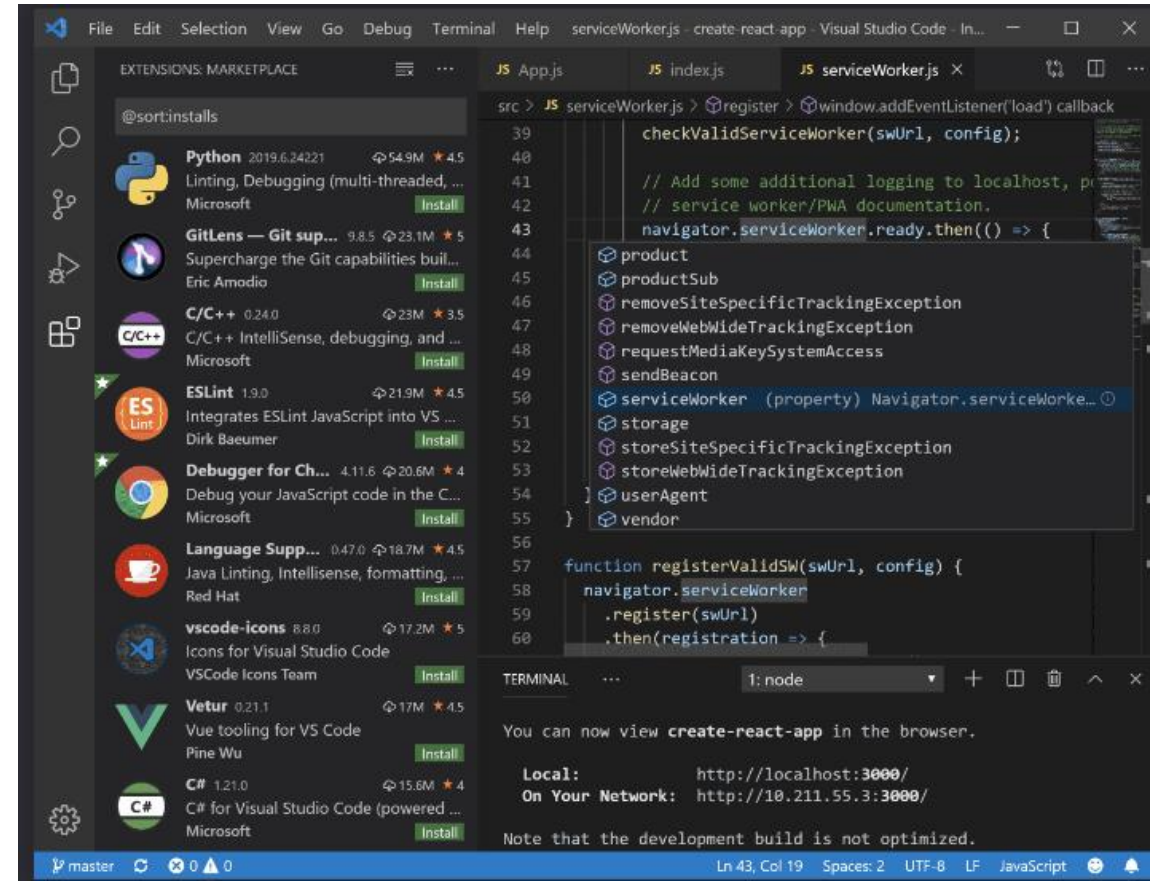
- O MySQL é um sistema de gerenciamento de banco de dados, que utiliza a linguagem SQL como interface.
- É atualmente um dos sistemas de gerenciamento de bancos de dados mais populares da Oracle Corporation, com mais de 10 milhões de instalações pelo mundo.
- XAMPP é um pacote com os principais servidores de código aberto do mercado, incluindo FTP, banco de dados MySQL e Apache com suporte as linguagens PHP e Perl.

[https://www.apachefriends.org/pt br/index.html](https://www.apachefriends.org/pt_br/index.html)



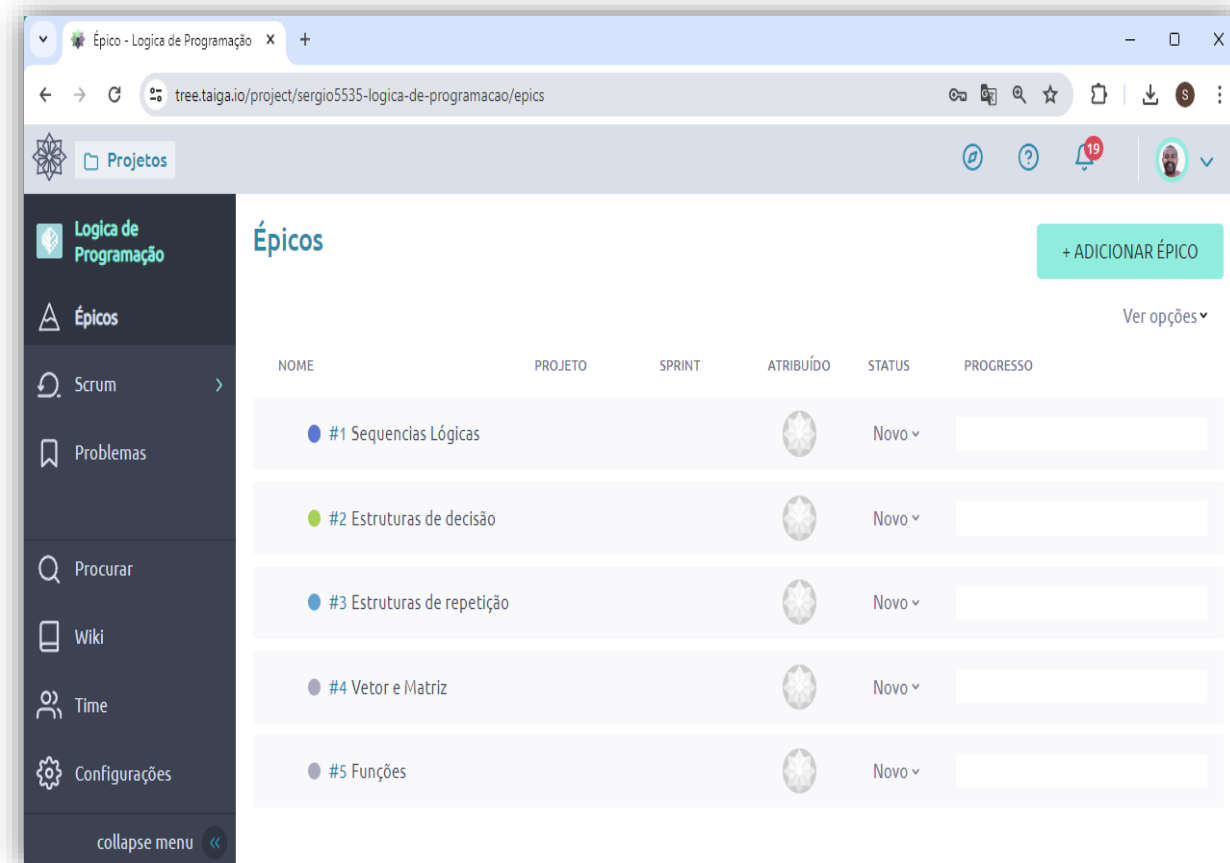
# Visual Studio Code (VS Code)

- O Visual Studio Code (VS Code) é um editor de código gratuito, leve e extensível para construção de aplicativos web, desktop e móveis.
- Ele suporta quase todas as principais linguagens de programação e oferece recursos como IntelliSense (sugestões de código inteligentes), depuração e integração com Git.
- Acesse: <https://code.visualstudio.com>



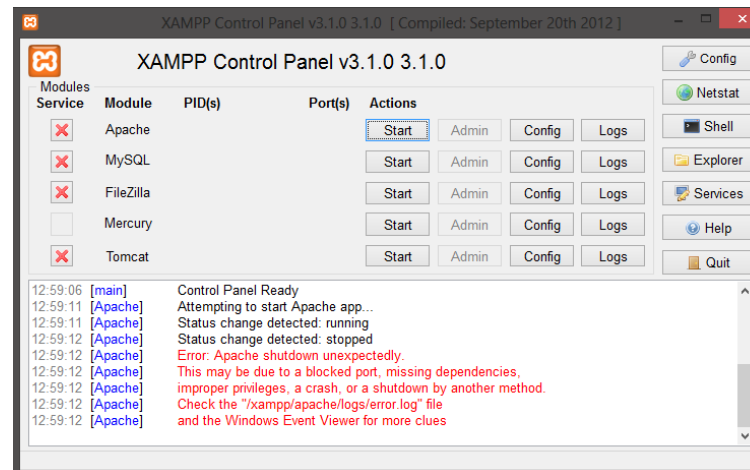
# Taiga.io

- O Taiga é um software de gerenciamento de projetos de código aberto projetado para equipes ágeis.
- Ele oferece recursos como planejamento, interação da equipe, insights, personalização e integrações com outras ferramentas.
- Com uma interface intuitiva, o Taiga é usado por equipes que trabalham com metodologias ágeis, como Scrum e Kanban.
- Acesse: <https://taiga.io>



# Servidor WEB XAMPP

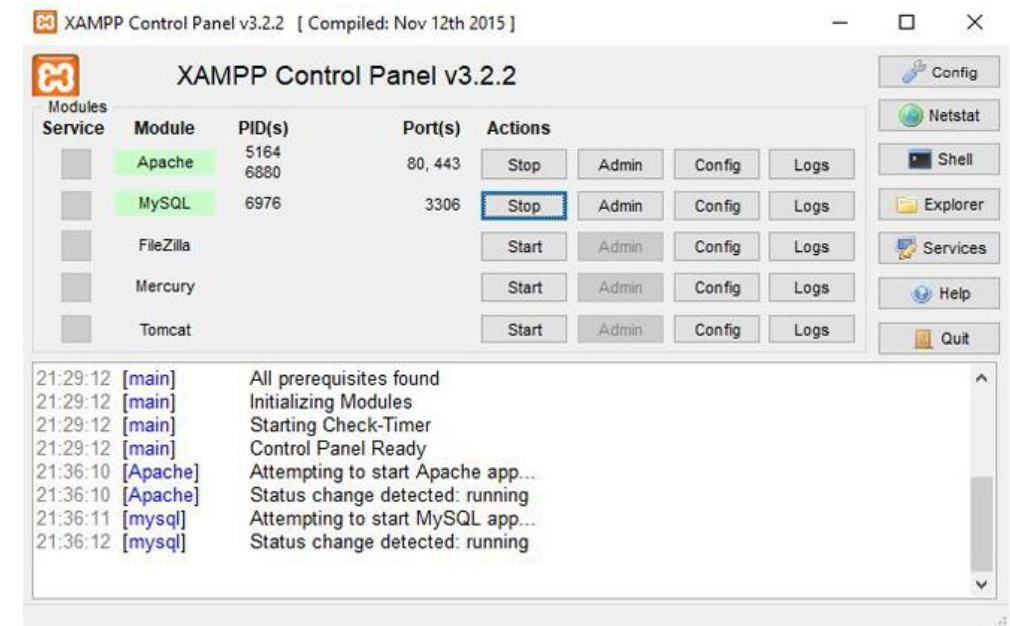
- XAMPP Apache + MariaDB + PHP + Perl
- XAMPP é o ambiente de desenvolvimento PHP mais popular.
- O XAMPP é gratuito, de fácil de instalar a distribuição Apache, contendo MySQL, PHP e Perl.
- O pacote de código aberto do XAMPP foi criado para ser fácil de instalar e de usar.





# Arquivos do projeto no XAMPP

- A pasta de projetos utilizando o XAMPP é:  
C:\xampp\htdocs
- Dentro dessa pasta crie uma pasta com o nome do seu projeto. Ex:  
C:\xampp\htdocs\PASTA\_DO\_PROJETO
- Agora arraste a pasta até a área de trabalho do VSCode IDE.
- Dê um START no XAMPP e vai no navegador e digite a (URL) :  
[http://localhost/PASTA\\_DO\\_PROJETO](http://localhost/PASTA_DO_PROJETO).
- OBS: é importante verificar se o Apache está iniciado (start).



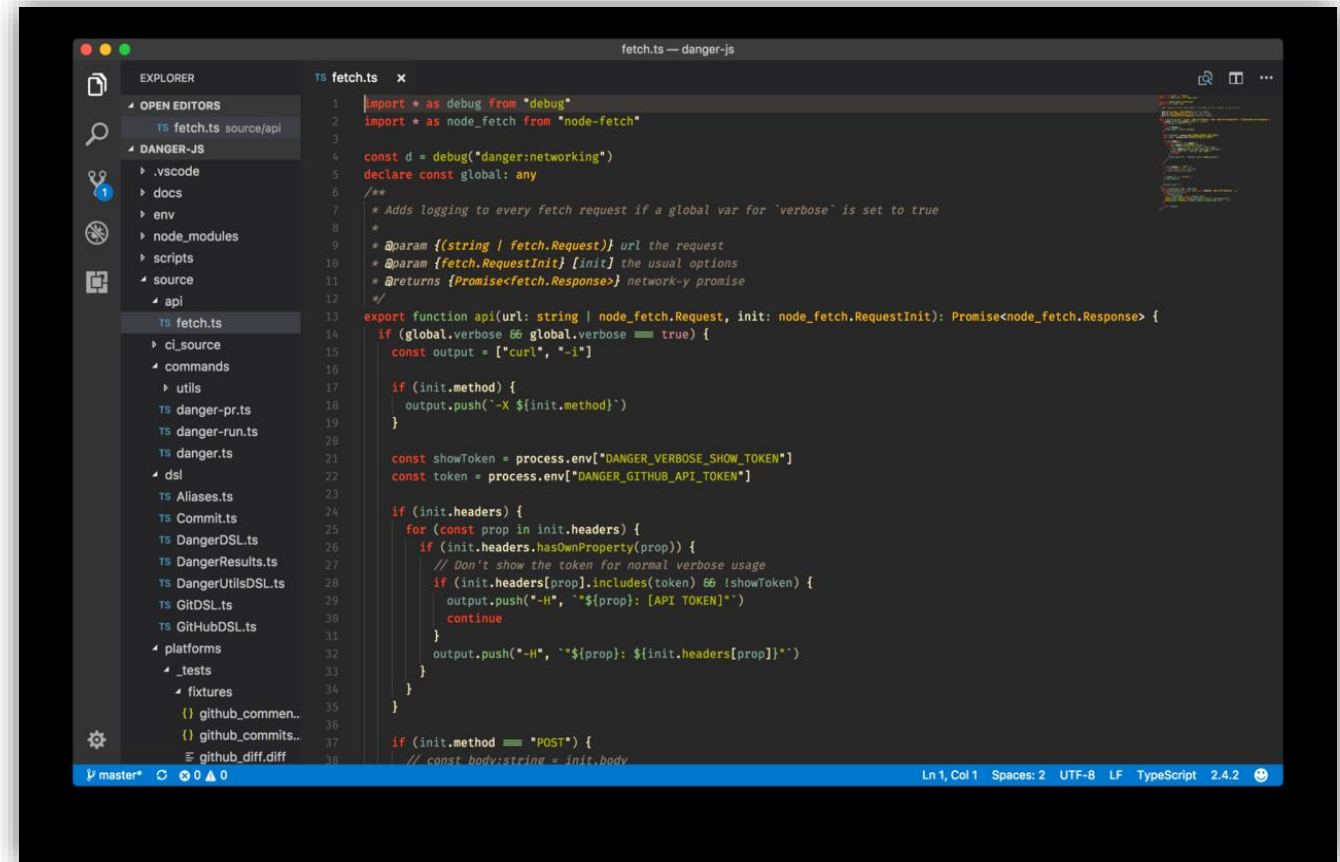


# Visual Studio Code VsCode



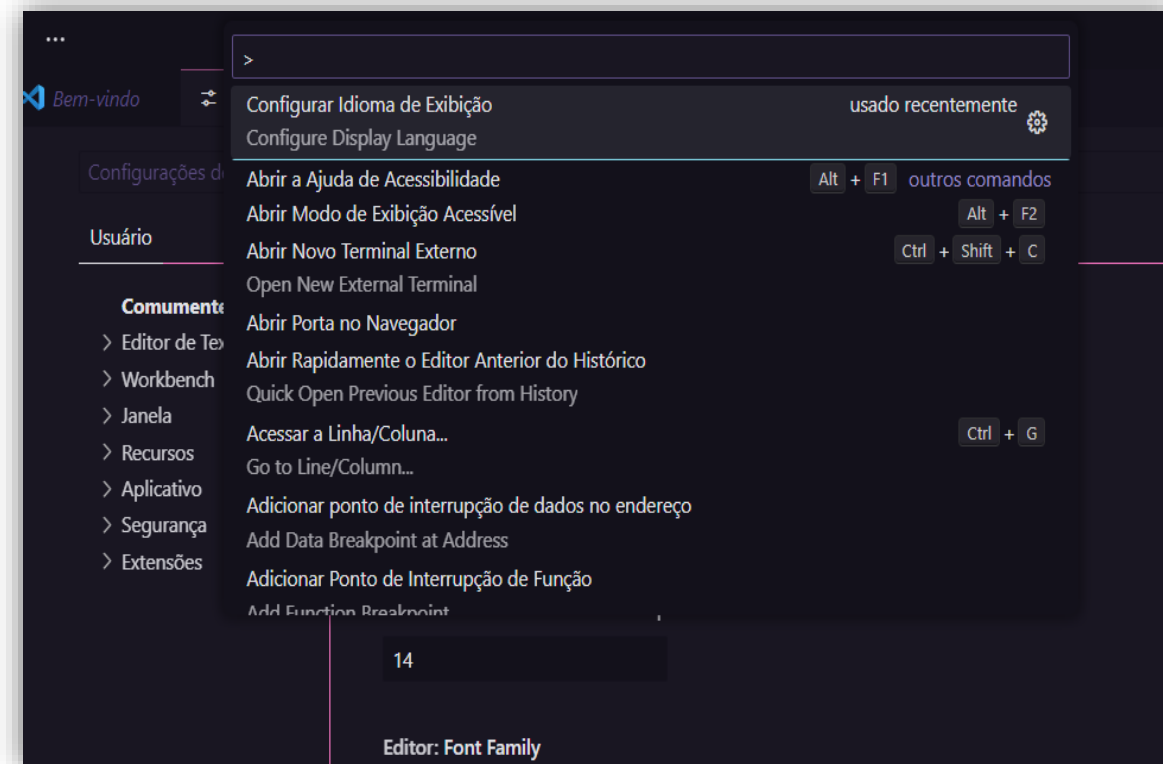
# Visual Studio Code (VsCode)

- O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS.
- Ele inclui suporte para depuração, controle Git incorporado, realce de sintaxe, complementação inteligente de código, snippets e refatoração de código.
- Versão online: <https://vscode.dev/>



# Configurações Gerais no VSCode

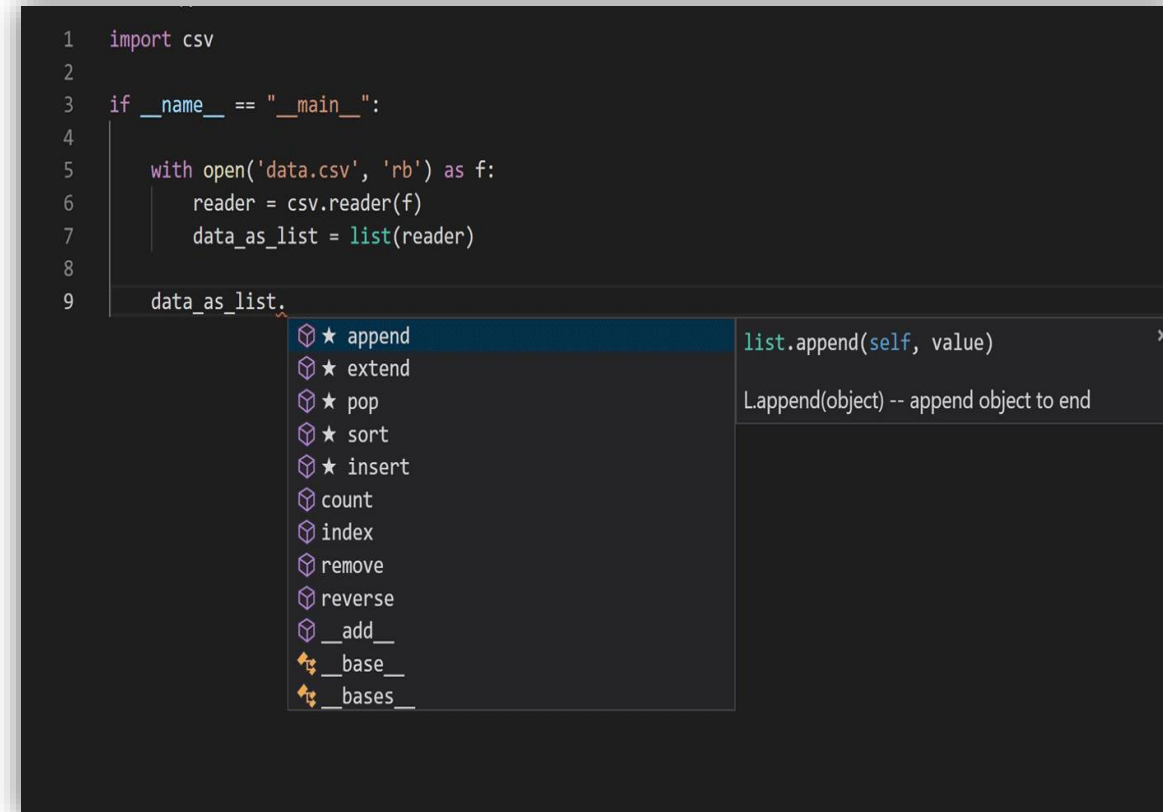
- Antes de começar a programar, é possível personalizar sua experiência no VSCode.
- Vá para as “Configurações” clicando em “File” e selecionando “Preferences” > “Settings”. É possível personalizar diversos aspectos do editor, como fonte, tema e atalhos de teclado.
- Para alterar o idioma pressione "Ctrl+Shift+P" para exibir a "Paleta de Comandos" e comece a digitar "display" para filtrar e exibir o comando "Configurar o Idioma de Exibição".
- Salvamento automático clique em “File” e selecionando “Auto Save”.
- Identação de Código: Shift+Alt+F
- Tema: Omni / Dracula



# Extensões Básicas para VSCode

- Vscod icons - é uma extensão para o Visual Studio Code que fornece um conjunto de ícones mais coloridos e detalhados para arquivos e pastas. Isso ajuda a tornar a navegação no editor mais visual e intuitiva.
- Output Colorizer - extensão é bastante útil para diferenciar com cores diferentes os pares de colchetes no código, tornando melhor a visualização dos vários colchetes encadeados que podem surgir.
- Visual Studio IntelliCode - uma extensão que utiliza a inteligência artificial e o contexto do seu código para criar, e te oferecer autocompletes referentes ao código que está escrevendo e acelerando o desenvolvimento do seu projeto.
- Extensão Git Lens - é uma extensão popular que sobrecarrega os recursos do Git incorporados ao VS Code.

```
1 import csv
2
3 if __name__ == "__main__":
4
5     with open('data.csv', 'rb') as f:
6         reader = csv.reader(f)
7         data_as_list = list(reader)
8
9     data_as_list.
```



The screenshot shows a code editor with a Python script. The script imports the 'csv' module and defines a main function that opens a file named 'data.csv' in read-binary mode, creates a CSV reader, and converts it to a list. The cursor is positioned at the end of the 'data\_as\_list' variable, and a dropdown menu is visible, suggesting various methods and attributes for lists, such as 'append', 'extend', 'pop', 'sort', 'insert', 'count', 'index', 'remove', 'reverse', '\_\_add\_\_', '\_\_base\_\_', and '\_\_bases\_\_'.



# Extensões HTML/CSS/JS no VSCode

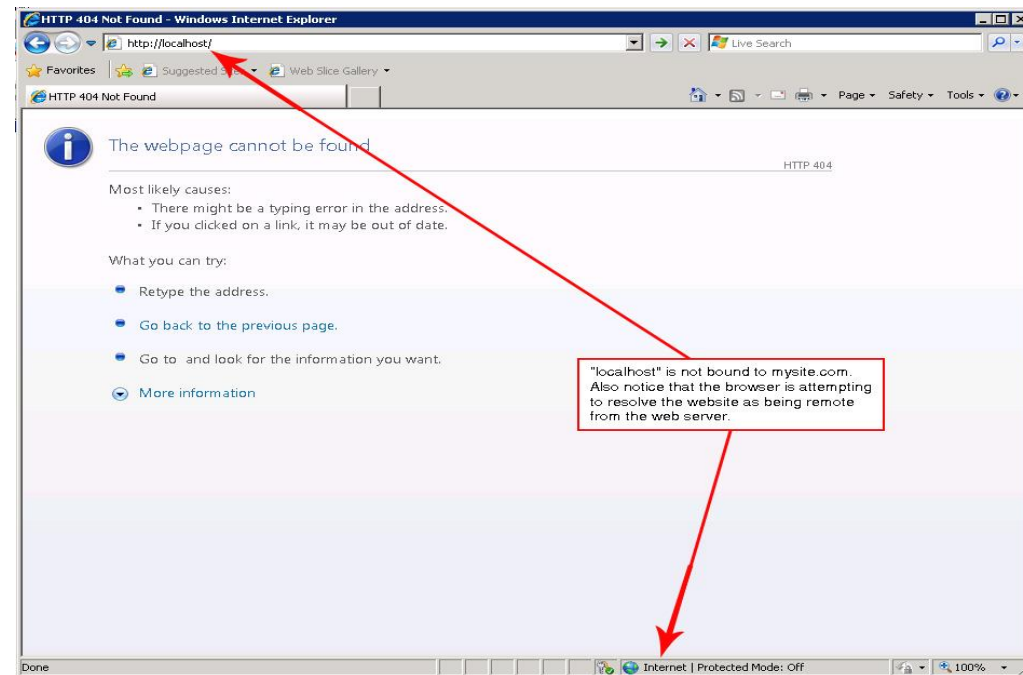
- Live Server: esse plugin cria um localhost da aplicação e toda alteração que é salva no projeto, é atualizada automaticamente, evitando dar refresh na página a cada modificação no código feita. Assim que instalado, é só clicar no botão Go Live e o localhost será aberto no seu navegador padrão.
- Color Highlight: extensão para identificar quando você escreve o código de uma cor e a exibe no editor. Assim fica fácil de saber de qual cor se trata cada código.
- Auto Rename Tag: essa extensão renomeia automaticamente as tags dentro do seu arquivo HTML.
- Image Preview: a extensão Image Preview, como o nome sugere, permite visualizar uma prévia da imagem à qual você está fazendo referência.
- Prettier: é uma extensão muito útil para formatação dos nossos arquivos. Ela ajusta a indentação do código facilmente.
- Emmet: permite que os usuários escrevam código de forma mais rápida e eficiente, expandindo abreviações em estruturas completas de código.

# Extensões PHP no VSCode

- PHP IntelliSense. O IntelliSense é uma ferramenta muito útil, pois ajuda a evitar erros ao digitar, economizando tempo e esforço. Sem ele, teríamos que digitar cada palavra manualmente, o que seria demorado e poderia levar a erros de digitação.
- PHP Debug. Esta excelente ferramenta permite que você comece a depurar seu código e identificar onde estão os erros.

# Localhost

- Na computação, o termo localhost se refere à localização do sistema que está sendo usado.
- É um dispositivo loopback ao qual é atribuído o endereço IP 127.0.0.1 no IPv4, ou::1 no IPv6, e pode ser usado por aplicações TCP/IP para testarem a comunicação consigo mesmas.





# Introdução ao PHP

---



# Introdução ao PHP

- PHP (um acrônimo recursivo para "PHP: Hypertext Preprocessor", originalmente Personal Home Page) é uma linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações presentes e atuantes no lado do servidor, capazes de gerar conteúdo dinâmico na World Wide Web.
- PHP é uma linguagem de scripts open source de uso geral, muito utilizada e especialmente adequada para o desenvolvimento web.
- Essa linguagem permite que desenvolvedores escrevam páginas geradas dinamicamente de forma rápida.
- PHP (um acrônimo recursivo para "PHP: Hypertext Preprocessor", originalmente Personal Home Page) é uma linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações presentes e atuantes no lado do servidor, capazes de gerar conteúdo dinâmico na World Wide Web



# Principais características

- A linguagem PHP é uma linguagem de programação de domínio específico, ou seja, seu escopo se estende a um campo de atuação que é o desenvolvimento web. Seu propósito principal é de implementar soluções web velozes, simples e eficientes.
- Velocidade e robustez.
- Orientação a objetos.
- Portabilidade - independência de plataforma - escreva uma vez, rode em qualquer lugar.
- Tipagem dinâmica.
- Sintaxe similar a C/C++ e o Perl.
- Open-source.
- Server-side (O cliente manda o pedido e o servidor responde em página HTML)

# Exemplo de uso do PHP

- O arquivo index foi salvo com a extensão .php para mostrarmos ao nosso interpretador que há um código PHP a ser interpretado.
- Além disso, no exemplo usamos a função echo para escrever na tela uma mensagem.

```
<!DOCTYPE html>
<html>
  <body>
    <h1>Estamos aprendendo PHP!</h1>
    <?php
      echo "Vamos prosseguir aprendendo PHP";
    ?>
  </body>
</html>
```

# Como comentar o código no PHP

- Para comentarmos o nosso código PHP usamos duas barras ou # para comentários de uma linha, e para comentários de múltiplas linhas usamos /\* \*/, o mesmo usado em CSS.

```
<?php
    echo "Oi, Eu serei visto na sua tela";
    // Eu não! Sou apenas um comentário.

    echo "Oi, Eu também serei visto por você";
    # Já eu não serei!

    echo "E eu aqui novamente na sua tela, rs";
    /* Eu não aparecerei na sua tela novamente
    pois sou um comentário */
?>
```

# Constantes no PHP

- O valor de uma constante jamais poderá ser alterado enquanto estiver sendo executada e para defini-la utilizamos a função `define()`.
- Utilizando a função `define()` definimos que a constante com o nome de PHP, terá como valor: Linguagem Open – Source.

```
<?php  
define("PHP", "Linguagem Open - Source");  
echo PHP; // Linguagem Open - Source  
?>
```

# Variáveis no PHP

- Para criarmos uma variável basta utilizar o sinal de cifrão. Uma variável pode armazenar textos e números. Além disso, a linguagem PHP é case sensitive, então A é diferente de a.
- No exemplo criamos uma variável (\$name) e declaramos a ela uma string, sendo assim precisamos colocá-la entre aspas. Já a outra variável (\$age) é declarada como inteiro, então não é necessário o uso de aspas. Ao usarmos echo nas variáveis, o resultado impresso é o conteúdo dessa variável.

```
<?php
$name = "Guilherme";
$age = 20;

echo $name; // Guilherme
echo "<br>";
echo $age; // 20
?>
```

# Nomeação de variáveis

- Não inicie o nome de uma variável com números;
- Não utilize espaços em brancos;
- Não utilize caracteres especiais, somente underline;
- Crie variáveis com nomes que ajudarão a identificar melhor a mesma;
- Evite utilizar letras maiúsculas.



# Tipos de variáveis

- Booleanos: Este é o tipo mais simples, pois só pode expressar apenas dois valores: TRUE (1) ou FALSE (0, null ou uma string vazia);
- Integer: é um número inteiro, podendo ser negativo ou positivo;
- Float : também chamado de double ou números reais representados com um ponto para separar os dígitos do valor inteiro dos dígitos do valor das casas decimais.
- Strings: é uma palavra ou frase entre aspas simples ou duplas, assim como também pode ser binário, como o conteúdo de um arquivo MP3 ou JPG.

# Tipos de variáveis

- Note que quando declaramos no echo "Olá, \$a, o PHP interpretou o conteúdo da \$a, pois está entre aspas duplas. E quando usamos a mesma forma, só que entre aspas simples (echo 'Olá, \$a'), não temos o mesmo resultado.
- Então quando queremos que o PHP interprete o valor de nossa variável dentro de uma string é necessário o uso de aspas duplas. Fique atento!

```
<?php
$a = "mundo!";
echo "Olá, $a"; // Olá, mundo!
echo 'Olá, $a'; // Olá, $a
?>
```

# Concatenar Strings

- Usar um ponto para concatenar strings

```
<?php  
echo "Olá," . " mundo!";  
//Olá, mundo!  
?>
```

# Operadores matemáticos

- Os operadores matemáticos disponíveis em PHP são:
- Adição: +
- Subtração: -
- Multiplicação: \*
- Divisão: /
- Módulo: %

```
<?php
$a = 3;
$b = 3;
$c = $a * $b; // resultado é 9
$d = $a + $b; // resultado é 6
$e = $c - $d; // resultado é 3
?>
```

# Operadores de Atribuição no PHP

- Utilizamos os operadores de atribuição para definir variáveis e seus valores, além de usá-los juntamente com os operadores matemáticos.

```
<?php
$a = 1; // A variável $a é igual a 1
$a += 2; // Somamos 2 ao valor da $a;
echo $a;
?>
```

```
<?php
$a -= 2; // Subtraímos 2 ao valor da variável $a;
$a *= 2; // Multiplicamos o valor da variável $a por 2;
$a /= 2; // Dividimos o valor da variável $a por 2.
?>
```

# Incrementar ou decrementar

- Podemos também incrementar ou decrementar variáveis utilizando os operadores de incrementação.

```
<?php
$a = 1;
echo ++$a; // Incrementamos 1 e retornamos o valor
echo $a++; // Retornamos o valor e incrementamos 1
echo --$a; // Decrementamos 1 e retornamos o valor
echo $a--; // Retornamos o valor e decrementamos 1
?>
```

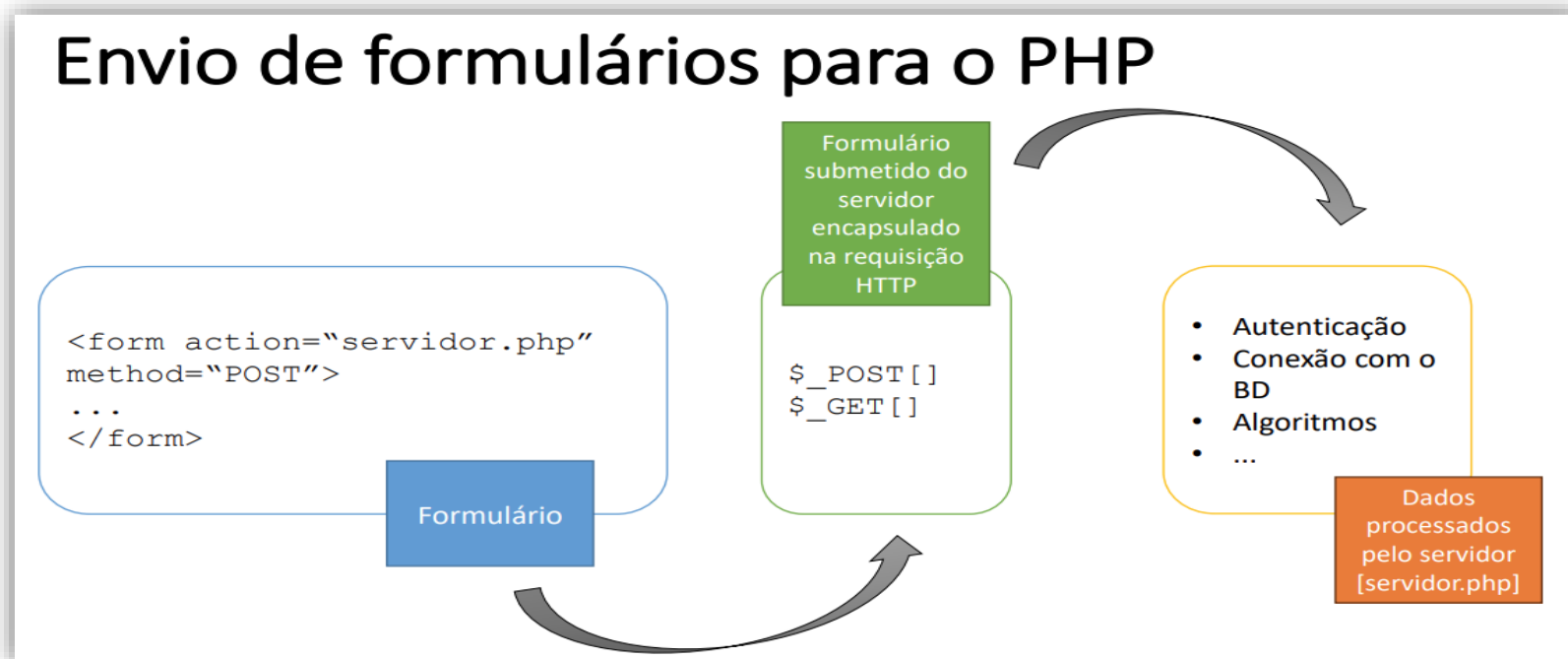


# Entrada de dados

- Para lidar com a entrada de dados em PHP, você pode usar vários métodos, como formulários HTML, variáveis superglobais (\$\_GET, \$\_POST).
- Variáveis superglobais são variáveis nativas que estão sempre disponíveis em todos escopos.
- \$\_GET : array associativo de variáveis passadas para o script atual via o método HTTP GET.
- \$\_POST : array associativo de variáveis passados para o script atual via método HTTP POST quando utilizado “application/x-www-form-urlencoded” ou “multipart/formdata” como valor do cabeçalho HTTP Content-Type na requisição.

# Envio de formulários

- Utilizamos a opção da action da tag form do HTML <form action="processa.php">.
- Existem dois métodos de passagem de parâmetros: GET e POST
- GET: <form action="processa.php" method="GET">
- POST: <form action="processa.php" method="POST">



# Propriedades do Formulário

- Primeiro: antes para poder enviar as informações, seu formulário deve conter um botão "submit".

```
1 | <input type=submit value="Texto do Botão">
```

- Segundo: todos os campos que serão tratados no script PHP devem conter o parâmetro "NAME", caso contrário, os dados não serão passados para o script PHP.

```
1 | <input type=text name=nome_do_campo>
```

- Como as informações chegam para o script PHP:

```
1 | <form action="script.php" method="post">
2 | Campo 1: <input type=text name=campo1><br>
3 | Campo 2: <input type=text name=campo2><br>
4 | <input type=submit value="OK">
5 | </form>
```

```
1 | <?php
2 | echo "O valor de CAMPO 1 é: " . $_POST["campo1"];
3 | echo "<br>O valor de CAMPO 2 é: " . $_POST["campo2"];
4 | ?>
```

# GET vs POST

GET	POST
Valores visíveis na URL	Valores não visíveis na URL (envio no corpo da mensagem)
Tem limitação de caracteres, dependendo do navegador (geralmente 255 caracteres)	Não tem limite de caracteres
Mais rápido que POST por causa da simplicidade	Mais lento que o GET por causa do tempo de encapsulamento
Suporta apenas strings	Suporta diferentes tipos de dados (strings, números, binário, etc.)
Fica no histórico do navegador	Não fica no histórico do navegador
Pode ser colocado nos favoritos (bookmark) por causa da URL	Não pode ser colocado nos favoritos (bookmark)

# Método GET

- O método GET é usado quando se quer obter dados de uma determinada origem ou recurso específico. Portanto, só devem ser usados para recuperar dados, e sua query string é enviada e exibida no endereço URL.
- Exemplo: `http://localhost/cadastra_usuario.php?nome=Fulano&idade=19`
- Caractere `?`: Representa o início de uma cadeia de parâmetros,
- Símbolo `&`: Identifica o início de uma nova variável.
- Caractere `=` : Separa as variáveis de seus respectivos valores.

```
<form action="recebe_dados.php">
    <p>Digite seu nome: <input type="text" name="nome" size="30"></p>
    <p>Digite seu e-mail: <input type="text" name="idade" size="3"></p>
    <p><input type="submit" value="Enviar!" name="enviar"></p>
</form>
```

`http://www.seusite.com.br/recebe_dados.php?nome=Joaquim&idade=20`

- `?` – representa o início da cadeia de variáveis
- `&` – identifica o início de uma nova variável
- `=` – separa as variáveis dos seus respectivos valores

# Criar Link com GET

- Para criar um link com o método GET, você pode usar a tag `<a>` do HTML. O método GET é o padrão para links, então você só precisa especificar a URL e os parâmetros que deseja passar.
- Exemplo: `<a href="https://example.com/page?param1=valor1&param2=valor2">Clique aqui</a>`
  - `https://example.com/page` é a URL de destino.
  - `param1=valor1` e `param2=valor2` são os parâmetros que serão enviados via GET.



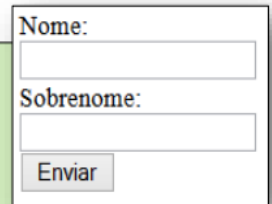


# Formulário GET

- Para criar um formulário que envie dados usando o método GET, você pode usar a tag <form> do HTML e definir o atributo method como "GET".

```
<form action="https://example.com/page" method="GET">
  <label for="param1">Parâmetro 1:</label>
  <input type="text" id="param1" name="param1">
  <br>
  <label for="param2">Parâmetro 2:</label>
  <input type="text" id="param2" name="param2">
  <br>
  <input type="submit" value="Enviar">
</form>
```

```
<form action= "recebedados.php" method= "get" >
  <label for= "nome"> Nome: <br/>
  <input type= "text" name="nome"><br/>
  <label for= "sobrenome"> Sobrenome: <br/>
  <input type= "text" name="sobrenome"><br/>
  <input type= "submit" value="Enviar">
</form>
```



- Neste exemplo:
  - action="https://example.com/page" define a URL para onde os dados do formulário serão enviados.
  - method="GET" especifica que os dados serão enviados usando o método GET.
  - Os campos de entrada (<input>) têm atributos name que correspondem aos nomes dos parâmetros que serão enviados.
  - Quando o usuário preencher o formulário e clicar em "Enviar", os dados serão anexados à URL como parâmetros de consulta.

# Recebendo dados via método GET

- Para recuperar os valores enviados através do método GET, basta utilizar o array global `$_GET`, indexado pelo nome do parâmetro:

```
<?php  
  
echo $_GET["nome"];  
echo $_GET["sobrenome"];
```

```
<?php  
  
echo "Olá".$_GET["nome"]." ".$_GET["sobrenome"]."!";
```

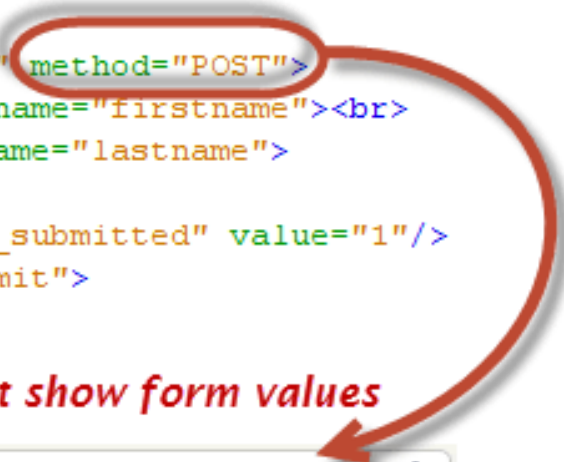
# Método POST

- O método POST é utilizado para enviar dados para o servidor, para atualizar ou criar um novo recurso.
- Diferentemente do método GET, o método POST não expõe as informações no endereço URL.

## FORM SUBMISSION POST METHOD

```
<form action="registration_form.php" method="POST">  
  First name: <input type="text" name="firstname"><br>  
  Last name: <input type="text" name="lastname">  
  <br>  
  <input type="hidden" name="form_submitted" value="1"/>  
  <input type="submit" value="Submit">  
</form>
```

*Submission URL does not show form values*



localhost/tuttis/registration\_form.php

# Formulário POST

- No caso do método POST, é necessário indicar explicitamente o método do envio.

```
<form action= "recebedados.php" method= "post">  
  <label for= "nome"> Nome: <br/>  
  <input type= "text" name="nome"><br/>  
  <label for= "sobrenome"> Sobrenome: <br/>  
  <input type= "text" name="sobrenome"><br/>  
  <input type= "submit" value="Enviar">  
</form>
```

Nome:

Sobrenome:

Enviar

# Recebendo dados via POST

- Caso os dados tenham sido enviados pelo método POST, deve-se utilizar o array global `$_POST` do PHP, indexado pelo nome do campo:
- Quando os dados são recebidos via método POST, eles vão ocultos. Não aparecem explicitamente para o usuário.

```
<?php  
  
echo "Olá".$_POST["nome"]." ".$_POST["sobrenome"]."!";
```

# Exercícios

- Crie o formulário e realize o processamento em PHP conforme figura abaixo:

The screenshot shows a Mozilla Firefox browser window with the title 'Formulário de pedido'. The address bar shows the URL 'http://localhost/minicurso/formul'. The page content includes a table with two columns: 'Item' and 'Quantidade'. There are three rows of input fields for 'Bola de volei:', 'Bola de futebol:', and 'Bola de handebol:'. Below the table is a button labeled 'Enviar pedido'.

Item	Quantidade
Bola de volei:	<input type="text"/>
Bola de futebol:	<input type="text"/>
Bola de handebol:	<input type="text"/>

The screenshot shows a Mozilla Firefox browser window with the title 'Mozilla Firefox'. The address bar shows the URL 'http://localhost/minicurso/pedid'. The page content displays a confirmation message: 'Seu pedido foi processado com sucesso.' followed by a list of items and their total cost: '10 bolas de volei lhe custaram R\$ 250', '10 bolas de futebol lhe custaram R\$ 100', '10 bolas de handebol lhe custaram R\$ 200', and 'Sua compra lhe custou R\$ 550'.

**Seu pedido foi processado com sucesso.**

10 bolas de volei lhe custaram R\$ 250  
10 bolas de futebol lhe custaram R\$ 100  
10 bolas de handebol lhe custaram R\$ 200  
Sua compra lhe custou R\$ 550



# Exercícios

- Faça um programa em PHP que peça um número e então mostre a mensagem: “O número informado foi.”

Informe um número

Enviar

# Campos Hidden

- Os campos hidden são usados para passar informações que não podem ser alteradas pelo usuário que estará inserindo informações no formulário. Por exemplo: você tem um site com sistema de login e o usuário quer alterar as informações de login dele.
- O script que irá manipular esse formulário, precisa saber o ID do usuário para poder alterar as informações no banco de dados, então esse ID é um campo hidden.

```
1 <form action="hidden.php" method="post">
2 <input type=hidden name=escondido value="valor do escondido">
3 <input type=hidden name=id value="111">
4 <input type=submit>
5 </form>
```

```
1 <?php
2 echo "Campo Hidden: " . $_POST["escondido"];
3 echo "<br>Oi, seu ID é: " . $_POST["id"];
4 ?>
```

# Campos Text, Textarea, Email, Number

- Os campos text e textarea são os tipos mais simples, onde há somente um possível valor por campo.

```
1 <form action="texts.php" method="post">
2 Nome: <input type="text" name="nome"><br>
3 Email: <input type="text" name="email"><br><br>
4 Mensagem: <textarea name="mensagem" cols=8 rows=3></textarea><br>
5 <input type="submit">
6 </form>
```

```
1 <?php
2 echo "Olá " . $_POST["nome"] . " (email: " . $_POST["email"] . ")<br><br>";
3 echo "Sua mensagem: " . $_POST["mensagem"];
4 ?>
```

# Campos Radio

- Campos Radio permitem um relacionamento de um para muitos entre identificador e valor, ou seja, eles têm múltiplos possíveis valores, mas somente um pode ser pré-exibido ou selecionado. Por exemplo: você tem um sistema de "quiz". Cada pergunta possui 5 possíveis respostas. Cada resposta é um radio, onde os 5 radios dessa pergunta possuem o mesmo identificador, mas cada com valores diferentes.

```
1 <form action="radio.php" method="post">
2 <B>Qual seu sistema operacional?</B><br>
3 <input type="radio" name="sistema" value="Windows 98"> Win 98
4 <input type="radio" name="sistema" value="Windows XP"> Win XP
5 <input type="radio" name="sistema" value="Linux"> Linux
6 <input type="radio" name="sistema" value="Mac"> Mac
7 <br><br>
8 <B>Qual a marca de seu monitor?</B><br>
9 <input type="radio" name="monitor" value="Samsung"> Samsung
10 <input type="radio" name="monitor" value="LG"> LG
11 <input type="radio" name="monitor" value="Desconhecido"> Desconhecido
12 <br><br>
13 <input type="submit">
14 </form>
```

```
1 <?php
2 echo "Seu sistema operacional é: " . $_POST["sistema"];
3 echo "<br>Seu monitor é: " . $_POST["monitor"];
4 ?>
```

# Campos Checkbox

- O tipo Checkbox tem somente um possível valor por entrada: on value (marcado) ou no value (desmarcado). No script você deve fazer a verificação para saber se o campo foi marcado ou não.
- Se é possível também utilizar grupos de checkbox com o mesmo nome. Para você deve adicionar "[]" no final do nome, para o PHP interpretar como array, veja o código exemplo.

```
1 <form action="checkbox.php" method="post">
2 <B>Escolha os numeros de sua preferência:</B><br>
3 <input type="checkbox" name="numeros[]" value=10> 10<br>
4 <input type="checkbox" name="numeros[]" value=100> 100<br>
5 <input type="checkbox" name="numeros[]" value=1000> 1000<br>
6 <input type="checkbox" name="numeros[]" value=10000> 10000<br>
7 <input type="checkbox" name="numeros[]" value=90> 90<br>
8 <input type="checkbox" name="numeros[]" value=50> 50<br>
9 <input type="checkbox" name="numeros[]" value=30> 30<br>
10 <input type="checkbox" name="numeros[]" value=15> 15<br><BR>
11 <input type="checkbox" name="news" value=1> <B>Receber
12 Newsletter?</B><br><BR>
13 <input type="submit">
14 </form>
```

```
// Faz loop pelo array dos numeros
foreach($_POST["numeros"] as $numero)
{
    echo "- " . $numero . "<BR>";
}
```

# Campos Select

- Os campos select permitem tratar uma variedade de opções, onde o usuário pode selecionar apenas uma opção ou múltiplas opções. Quando você permite múltiplas seleções, deve adicionar "[]" no final do nome, para o PHP interpretar como array.

```
1 <form action="select.php" method="post">
2 <B>Qual seu processador?</B><br>
3 <select name=processador>
4 <option value=Pentium>Pentium</option>
5 <option value=AMD>AMD</option>
6 <option value=Celeron>Celeron</option>
7 </select><BR><BR>
```

```
1 <?php
2 echo "Seu processador é: " . $_POST["processador"] . "<BR>";
3
```

```
<select name= "cidade">
```

```
<option value= "Nova Cruz" > Nova Cruz </option>
```

```
<option value= "Serrinha" > Serrinha </option>
```

```
<option value= "Montanhas" > Montanhas </option>
```

```
<option value= "Brejinho" > Brejinho </option>
```

```
<option value= "Monte Alegre" > Monte Alegre </option>
```

```
<option value= "Natal" > Natal </option>
```

```
</select>
```

```
...
```

# Função htmlspecialchars

- A função htmlspecialchars em PHP é usada para converter caracteres especiais em entidades HTML. Isso é importante para evitar ataques de Cross-Site Scripting (XSS), onde um atacante pode injetar código malicioso em páginas web.
- Como Funciona
  - Quando você usa htmlspecialchars, caracteres como <, >, & e " são convertidos em suas respectivas entidades HTML (&lt;, &gt;, &amp;, &quot;). Isso garante que qualquer entrada do usuário seja tratada como texto e não como código HTML ou JavaScript.

- Exemplo:

```
$entrada_usuario = "<script>alert('Ataque XSS!');</script>";  
$entrada_segura = htmlspecialchars($entrada_usuario);  
  
echo $entrada_segura; // Saída: &lt;script&gt;alert('Ataque XSS!');&lt;/script&gt;
```

- Quando Usar: sempre que exibir dados fornecidos pelo usuário em uma página web. Isso inclui formulários, comentários, mensagens e qualquer outra entrada que possa ser exibida no navegador.

# Array

- Para declarar um array em PHP utilizamos o construtor de linguagem `array()` , para o qual podemos passar por parâmetro os valores que desejamos armazenar, separados por vírgula.

```
1 | $array = array(1, 2, 3);
```

- Após declarar a variável que contém os valores, podemos acessá-los utilizando sua posição, como mostra o exemplo a seguir:

```
1 | echo $array[0];  
2 | echo $array[1];  
3 | echo $array[2];
```

- Também podemos sobrescrever o valor presente em uma posição específica do array utilizando a chave a ele associada.

```
1 | $array["chave2"] = 2;
```



# Array

- Por meio de uma estrutura de repetição, como o foreach, podemos percorrer os dados em um array.

```
<?php
$days = array("monday", "tuesday", "friday", "sunday");

foreach ($days as $value) {
    echo "$value <br>";
}
?>
```

# Array

- Unset: para remover determinado elemento do array:  
`unset($arr[5]);`
- Para remover o array da memória:  
`unset($arr);`
- Count: retorna a quantidade de elementos de um array:  
`count($arr);`
- Explode: transforma string em array  
`$arr = array();`  
`$arr = explode("/", "20/01/2001");`
- Implode: transforma array em string.  
`$arr = array("Flavio", "Alexandre", "Micheletti");`  
`$nomeCompleto = implode("-", $arr);`  
`var_dump($nomeCompleto)`

```
1 <?php
2
3 $fruits = ['apple', 'banana', 'orange'];
4
5 echo count($fruit);
6
7 // output is 3
8
9 // i mostly use the count() function to check if rows from db query are empty
10
11 // Example:
12
13 if(count($rows) > 0) {
14
15 // Do some thing rows array is not empty
16
17 }
```

# Operadores Relacionais

- Esses são usados para comparar valores ou expressões, retornando um valor booleano (true ou false):
- Igual: ==
- Diferente: != ou <>
- Menor que: <
- Maior que: >
- Menor ou igual: <=
- Maior ou igual: >=

# Operadores Lógicos

- Existem também os operadores lógicos para a criação de testes condicionais:
- \$a and \$b: enquanto A e B forem verdadeiros;
- \$a or \$b: enquanto A ou B forem verdadeiros;
- !: Negação

# Estrutura de Decisão if/else

- A condição é avaliada para que, caso algo seja verdadeiro, faça isto, senão, faça aquilo.
- Criamos a variável `$idade` que guarda um inteiro. Em seguida utilizamos *IF* para verificar se `$idade` é menor que 18, e caso seja será impresso: Você não pode entra aqui! Depois criamos um *ELSE*, que é o contrário da primeira condição.

```
<?php
    $idade = 17;

    if($idade < 18) {
        echo 'Você não pode entrar aqui!';
    } else {
        echo 'Seja bem - vindo';
    }
?>
```

# Estruturas de Decisão (SWITCH)

- Para não manter um código cheio de ELSEIF's, o mais indicado é usar o **SWITCH**, que permite criarmos infinitas condições de forma organizada.

```
<?php
$nome = 'Fulano';

switch($nome) {
    case 'Fulano':
        echo 'E ai Fulano!';
        break;

    case 'Sicrano':
        echo 'E ai Sicrano!';
        break;

    case 'Beltrano':
        echo 'E ai Beltrano!';
        break;

    default:
        echo 'Qual é o seu nome?';
        break;
}

// Resultado é: E ai Fulano!
?>
```

# Laços de Repetição (for)

- A primeira é executada ao início do loop, a segunda é a condição (enquanto ela for verdadeira, o loop continuará), e a terceira é executada ao fim de cada repetição.

```
<?php
for($a = 1; $a <= 10; $a++){
    $cubo = $a * $a * $a;
    echo "O cubo de $a é $cubo<br />";
}
?>
```

# Laços de Repetição (foreach)

- O **foreach** faz o mesmo que as demais estruturas já apresentadas, porém, com ela podemos trabalhar com arrays.
- No exemplo é criado um array e depois usamos o foreach para ir nesse array e repetir tudo o que conter nele. A sintaxe do foreach é mostrada entre parênteses onde colocamos o nome de nossa variável, e com o termo *as* alteramos o nome dela para \$can.
- Depois, basta dar um echo que tudo que conter em nosso array será mostrado.

```
<?php
$ead = array('Aqui na DevMedia ', 'você se torna um ', 'desenvolvedor PHP');

foreach($ead as $can){
    echo "$can";
}

//Aqui na DevMedia você se torna um desenvolvedor PHP
?>
```



# Funções

- Funções podem ser definidas blocos de código com um objetivo específico, identificados por um nome através do qual pode ser referenciado a partir de várias partes do código.
- Essa é uma das principais técnicas utilizadas para garantir a reutilização de código, tornando a programação mais prática e o código mais “limpo” e organizado.

# Funções para tratamento de Strings

**1. strlen()** – Retorna o número de caracteres de uma string

```
<?php
```

```
$str = "abcdef";
```

```
// Retorna 6
```

```
echo strlen($str);
```

```
$str = ' ab cd ';
```

```
// Retorna 7
```

```
echo strlen($str);
```

```
?>
```

```
<?php
```

```
/* Validar se um campo de formulário tem o  
número
```

```
necessário de caracteres */
```

```
$nome=$_POST['comentario'];
```

```
if (strlen($nome) > 200) {
```

```
echo "O comentário deve ter no máximo 200  
caracteres.";
```

```
}
```

```
?>
```

# Funções para tratamento de Strings

2. **strtoupper()** – Converte uma string para maiúsculas.

```
<?php  
$str = "eu sou uma string";  
  
// Retorna "EU SOU UMA STRING"  
echo strtoupper($str);  
?>
```

# Funções para tratamento de Strings

**3. strtolower()** – Converte uma string para minúsculas.

```
<?php  
$str = "EU SOU UMA STRING";  
  
// Retorna "eu sou uma string"  
echo strtolower($str);  
?>
```

# Funções para tratamento de Strings

**4. urlencode()** – Retorna a string, convertida para o formato urlencode.

```
<?php  
$frase = "Eu preciso ser passado por GET";  
// Retorna "Eu+preciso+ser+passado+por+GET";  
echo urlencode($frase)  
?>
```

**5. trim()** – Retira espaços e linhas em branco do início e do final da string fornecida.

```
<?php  
// Retorna "faael"  
echo trim(" faael \n \n ");  
?>
```

# Funções para tratamento de Data

1) Exibindo a data no formato 16/02/2004.

```
<?
//FUNÇÃO DATE()
echo $data = date("d/m/Y");
?>
```

2) Exibindo a data e a hora no formato 14/02/2004 21:04:02

```
<?
//FUNÇÃO DATE()
echo $data = date("d/m/Y H:i:s ");
?>
```

3) Exibindo a data por extenso Sat, 14 de Feb de 2004.

```
<?
//FUNÇÃO DATE()
echo $data = date("D, d de M de Y");
?>
```

# Funções Matemáticas

- // Arredonda pi 3.1416... para baixo

```
$round = round(M_PI);
```

```
print $round; // imprime 3
```

- // Desta vez, arredonda pi em 4 casas decimais

```
$round_decimal = round(M_PI, 4);
```

```
print $round_decimal; // imprime 3.1416
```

- // imprime um numero entre 0 e 32767

```
print rand();
```

- // imprime um numero entre 1 e 10

```
print rand(1,10);
```

- pow ( number \$base , number \$exp )

- sqrt ( float \$arg )

# Funções de sessão

- Session\_start inicia a sessão  
`session_start();`  
`$_SESSION['usuario'] = 'Thiago';`
- Deleta uma variável da sessão  
`unset($_SESSION['usuario']);`  
`session_destroy();` // Destrói toda sessão  
`header('location:index.php');`
- Testa variável de sessão  
`if((!isset($_SESSION['login']) == true) and (!isset($_SESSION['senha']) == true))`  
`{`  
`unset($_SESSION['login']);`  
`unset($_SESSION['senha']);`  
`header('location:index.php');`  
`}`



# Funções de include

- A declaração `include_once` inclui e avalia o arquivo informado durante a execução do script.

```
<?php
```

```
include_once "a.php"; // this will include a.php
```

```
include_once "A.php"; // this will include a.php again! (PHP 4 only)
```

```
?>
```

# Outras funções

- `isset` — Informa se a variável foi iniciada
- // Será interpretado como TRUE imprimindo o texto.
- ```
if (isset($var)) {  
    echo "Essa variável existe."  
}
```

# Bons Estudos!

