UNIPAC
Barbacena

Centro Universitário Presidente Antônio Carlos
Programação para Internet

PHP - Formulários - MySQL

Felipe Roncalli de Paula Carneiro
felipecarneiro@unipac.br

O que vamos aprender nessa aula

- PHP - Formulários;
- POST;
- Validando Dados;
- Conexão com Banco
- Exercícios

Importância do PHP na programação Web

O PHP (Hypertext Preprocessor) é uma linguagem de programação amplamente utilizada na programação para web e desempenha um papel fundamental na construção de aplicativos e sites dinâmicos. A importância do PHP na programação para web pode ser destacada por várias razões:

Importância do PHP na programação Web

Facilidade de Aprendizado: O PHP é conhecido por ser uma linguagem de programação relativamente fácil de aprender e de usar, tornando-a acessível para desenvolvedores iniciantes.

Ampla Compatibilidade: PHP é compatível com a maioria dos servidores web, como Apache, Nginx, IIS, entre outros, tornando-o uma escolha popular para o desenvolvimento web.

Importância do PHP na programação Web

Código Aberto: O PHP é uma linguagem de código aberto, o que significa que é gratuito e amplamente suportado pela comunidade de desenvolvedores. Isso resulta em uma grande quantidade de bibliotecas, frameworks e recursos disponíveis.

Integração com Bancos de Dados: PHP é frequentemente usado em conjunto com bancos de dados, como MySQL, PostgreSQL e outros. Ele oferece facilidade na conexão e manipulação de dados, o que é essencial para aplicativos web que precisam armazenar e recuperar informações.

Importância do PHP na programação Web

Ampla Comunidade e Documentação: Existem muitos recursos de aprendizado, fóruns de suporte e comunidades de desenvolvedores dedicadas ao PHP. Isso facilita a obtenção de ajuda e orientação sempre que necessário.

Desenvolvimento Rápido: PHP é uma linguagem interpretada, o que permite que os desenvolvedores vejam resultados imediatos à medida que escrevem e atualizam seu código. Isso agiliza o desenvolvimento de aplicativos web.

Importância do PHP na programação Web

Flexibilidade e Versatilidade: O PHP é flexível o suficiente para lidar com uma ampla variedade de tarefas, desde a criação de sites simples até o desenvolvimento de sistemas web complexos e aplicativos corporativos.

Suporte a APIs e Integração com Serviços Web: PHP é frequentemente usado para criar APIs (Application Programming Interfaces) que permitem a comunicação entre aplicativos web e serviços externos, tornando-o uma escolha sólida para aplicativos baseados em microserviços e serviços web.

Importância do PHP na programação Web

Framework PHP: Existem inúmeros frameworks PHP populares, como Laravel, Symfony, CodeIgniter e Zend, que simplificam o desenvolvimento e seguem boas práticas de programação. Isso acelera o processo de desenvolvimento e melhora a manutenção do código.

Formulário HTML

```html
<form action="processar_formulario.php" method="POST">
    <!-- Campo de Texto para Nome -->
    <label for="nome">Nome:</label>
    <input type="text" id="nome" name="nome" required>
    <br><br>

    <!-- Campo de Texto para E-mail -->
    <label for="email">E-mail:</label>
    <input type="email" id="email" name="email" required>
    <br><br>

    <!-- Área de Texto para Comentários -->
    <label for="comentario">Comentários:</label>
    <textarea id="comentario" name="comentario" rows="4" cols="50"></textarea>
    <br><br>

    <!-- Lista Suspensa para Gênero -->
    <label for="genero">Gênero:</label>
    <select id="genero" name="genero">
        <option value="masculino">Masculino</option>
        <option value="feminino">Feminino</option>
        <option value="outro">Outro</option>
    </select>
    <br><br>

    <!-- Botão de Envio -->
    <input type="submit" value="Enviar">
</form>
```

O que é o método HTTP POST

O método HTTP POST é um dos principais métodos de transferência de dados na arquitetura da web. Ele é usado para enviar dados do cliente para o servidor. Quando um navegador envia uma solicitação POST a um servidor web, os dados são enviados no corpo da solicitação, em vez de anexados à URL (como no método GET). Isso permite o envio de dados sensíveis e/ou grandes, como os dados de um formulário HTML.

Principais Características do Método POST

Envio no Corpo da Solicitação: Os dados do formulário são enviados no corpo da solicitação HTTP, tornando-os menos visíveis na URL do navegador. Isso é importante para a privacidade e segurança dos dados sensíveis.

Uso de Formulários HTML: O método POST é comumente usado em formulários HTML. Quando um usuário preenche um formulário e o envia, os dados do formulário são enviados ao servidor usando o método POST.

Principais Características do Método POST

Segurança: Embora o POST seja mais seguro que o GET para o envio de informações confidenciais, ele não é totalmente seguro por si só. Medidas adicionais, como criptografia SSL/TLS, são frequentemente usadas para proteger ainda mais os dados durante a transmissão.

Capacidade para Dados Complexos: O POST não possui limitações de tamanho de dados na teoria, embora servidores e navegadores possam impor limites práticos. Isso o torna adequado para o envio de grandes volumes de dados, como upload de arquivos.

Recebendo dados com POST

Receber e exibir os dados enviados por um formulário usando $_POST em PHP é uma tarefa relativamente simples. Aqui está um exemplo de como fazê-lo:

```php
<?php
    // Verifica se o formulário foi submetido
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        // Recebe os dados do formulário usando $_POST
        $nome = $_POST["nome"];
        $email = $_POST["email"];

        // Exibe os dados na página
        echo "<p>Nome: " . htmlspecialchars($nome) . "</p>"; // E
        echo "<p>E-mail: " . htmlspecialchars($email) . "</p>"; /
    } else {
        // Se o formulário não foi submetido, exibe uma mensagem
        echo "<p>O formulário não foi submetido.</p>";
    }
    ?>
```

Validação de Dados

A validação dos dados do formulário é uma parte fundamental do processamento de formulários em PHP para garantir a integridade e a consistência dos dados que você recebe. Aqui estão alguns passos para validar os dados do formulário, incluindo a verificação de campos obrigatórios e tipos de dados:

```php
<?php
if (empty($_POST["nome"])) {
    $erros[] = "O campo Nome é obrigatório.";
}
if (empty($_POST["email"])) {
    $erros[] = "O campo E-mail é obrigatório.";
}
?>
```

Validação de Dados

Exibindo erros: Após realizar as verificações de validação, você deve verificar se há erros e, se houver, exibi-los para o usuário. Você pode fazer isso de várias maneiras, mas uma abordagem comum é exibir uma lista de erros acima ou abaixo do formulário.

```php
<?php
if (!empty($erros)) {
    echo "<ul>";
    foreach ($erros as $erro) {
        echo "<li>$erro</li>";
    }
    echo "</ul>";
}
?>
```

PHP + Banco de Dados

Configurando os detalhes de conexão: Primeiro, você precisa definir os detalhes de conexão, como o nome do servidor, nome de usuário, senha e nome do banco de dados. Substitua os valores apropriados nas variáveis a seguir:

```
$servidor = "seu_servidor_mysql";
$usuario = "seu_usuario_mysql";
$senha = "sua_senha_mysql";
$banco = "seu_banco_de_dados";
```

PHP + Banco de Dados

Executando consultas SQL: Com a conexão estabelecida, você pode executar consultas SQL para interagir com o banco de dados. Aqui está um exemplo de como você pode realizar uma consulta de seleção e exibir os resultados:

```php
$sql = "SELECT nome, email FROM usuarios";
$resultado = mysqli_query($conexao, $sql);

if (mysqli_num_rows($resultado) > 0) {
    while ($row = mysqli_fetch_assoc($resultado)) {
        echo "Nome: " . $row["nome"] . " - E-mail: " . $row["email"] . "<br>";
    }
} else {
    echo "Nenhum resultado encontrado.";
}
```

PHP + Banco de Dados

Estabelecendo a conexão: Use a função mysqli_connect() para criar uma conexão com o banco de dados. Você pode adicionar tratamento de erro para lidar com falhas na conexão.

```php
$conexao = mysqli_connect($servidor, $usuario, $senha, $banco);

// Verifica se a conexão foi bem-sucedida
if (!$conexao) {
    die("Falha na conexão: " . mysqli_connect_error());
}
```

PHP + Banco de Dados

Fechando a conexão: É importante fechar a conexão com o banco de dados quando você terminar de usá-la para liberar recursos. Você pode fazer isso usando a função mysqli_close().

```php
mysqli_close($conexao);
```

PHP + SQL

O comando **mysqli_prepare** é uma função em PHP que faz parte da extensão MySQLi (MySQL Improved), usada para interagir com bancos de dados MySQL de forma segura e eficiente. A função mysqli_prepare é usada para criar uma instrução SQL preparada que pode ser posteriormente executada. Instruções preparadas são úteis para evitar ataques de injeção de SQL e para melhorar o desempenho ao executar consultas SQL múltiplas com parâmetros variáveis.

PHP + SQL

```php
// Insira os dados no banco de dados
$sql = "INSERT INTO cadastro (nome, email) VALUES (?, ?)";
$stmt = mysqli_prepare($conexao, $sql);
```

PHP + SQL

A função **mysqli_stmt_bind_param** faz parte da extensão MySQLi em PHP e é usada para vincular parâmetros a uma consulta preparada. Ela é usada principalmente para fornecer valores aos espaços reservados em uma instrução SQL preparada. A função mysqli_stmt_bind_param ajuda a garantir que os valores sejam tratados de forma segura e evita possíveis ataques de injeção de SQL.

PHP + SQL

```
mysqli_stmt_bind_param($stmt, "ss", $nome, $email);
```

$stmt (obrigatório): É o objeto de consulta preparada criado anteriormente com a função mysqli_prepare. É o alvo da vinculação dos parâmetros.

$types (obrigatório): É uma string que especifica os tipos de dados dos parâmetros a serem vinculados na consulta preparada

$nome, $email,..., $var: São as variáveis que contêm os valores que você deseja vincular aos espaços reservados na consulta preparada

PHP + SQL

```
mysqli_stmt_execute($stmt)
```

A função mysqli_stmt_execute é usada em PHP para executar uma consulta preparada criada com a extensão MySQLi (MySQL Improved). As consultas preparadas são úteis para executar consultas SQL de forma segura e eficiente, especialmente quando você precisa executar a mesma consulta com diferentes valores de parâmetros. A função mysqli_stmt_execute permite que você execute a consulta preparada com os valores vinculados aos espaços reservados

PHP + SQL

```
mysqli_stmt_close($stmt)
```

A função mysqli_stmt_close é utilizada em PHP para fechar uma consulta preparada (prepared statement) criada com a extensão MySQLi. O objetivo de fechar a consulta preparada é liberar os recursos associados a ela e liberar a memória utilizada. É uma boa prática fechar as consultas preparadas após concluí-las ou quando elas não forem mais necessárias

PHP + SQL

```php
$sql = "INSERT INTO cadastro (nome, email) VALUES (?, ?)";
$stmt = mysqli_prepare($conexao, $sql);

if ($stmt) {
    mysqli_stmt_bind_param($stmt, "ss", $nome, $email);

    if (mysqli_stmt_execute($stmt)) {
        echo "Usuário cadastrado com sucesso!";
    } else {
        echo "Erro na inserção de dados: " . mysqli_error($conexao);
    }

    mysqli_stmt_close($stmt);
} else {
    echo "Erro na preparação da consulta: " . mysqli_error($conexao);
}

// Feche a conexão com o banco de dados
mysqli_close($conexao);
```

Exercícios

Implementar o exercício da aula anterior. Construir os CRUDs necessários. Utilizar os conhecimentos adquiridos em aulas passadas para uma experiência do usuário mais interessante.

Dúvidas??