Tutorial Completo de PHP: Bases de Datos, Formularios y Conexiones

1. Introducción a PHP

¿Qué es PHP?

- Lenguaje de programación del lado del servidor
- Especialmente diseñado para desarrollo web
- Permite crear páginas web dinámicas
- Código se ejecuta en el servidor antes de enviar al navegador

2. Sintaxis Básica de PHP

```
<?php
// Comentario de una línea
/* Comentario
   de múltiples líneas */

// Variables
$nombre = "Juan";
$edad = 25;
$altura = 1.75;

// Imprimir en pantalla
echo "Hola, mi nombre es $nombre";
?>
```

3. Conexión a Base de Datos con MySQLi

Método Orientado a Objetos

```
<?php
$servername = "localhost";
$username = "tu_usuario";
$password = "tu_contraseña";
$database = "mi_basedatos";

// Crear conexión
$conexion = new mysqli($servername, $username, $password, $database);

// Verificar conexión
if ($conexion->connect_error) {
    die("Error de conexión: " . $conexion->connect_error);
}

// Realizar consulta
$consulta = $conexion->query("SELECT * FROM usuarios");

// Recorrer resultados
while ($fila = $consulta->fetch_assoc()) {
    echo $fila['nombre'];
}

// Cerrar conexión
$conexion->close();
?>
```

Método con PDO (Más Seguro)

```php
<?php
try {
    $conexion = new PDO("mysql:host=localhost;dbname=mi_basedatos", "usuario", "contraseña");
    $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // Consulta preparada
    $stmt = $conexion->prepare("SELECT * FROM usuarios WHERE id = :id");
    $stmt->execute(['id' => $usuario_id]);
    $resultado = $stmt->fetchAll();
} catch(PDOException $e) {
    echo "Error: " . $e->getMessage();
}
?>
```

4. Capturar Datos de Formularios HTML

Formulario HTML

```html
<form action="procesar.php" method="POST">
    <input type="text" name="nombre">
    <input type="email" name="correo">
    <input type="submit" value="Enviar">
</form>
```

Procesar Datos en PHP

```php
<?php
// Método POST
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $nombre = $_POST['nombre'];
    $correo = $_POST['correo'];

    // Validar datos
    if (!empty($nombre) && filter_var($correo, FILTER_VALIDATE_EMAIL)) {
        // Guardar en base de datos
        $stmt = $conexion->prepare("INSERT INTO usuarios (nombre, correo) VALUES (?, ?)");
        $stmt->bind_param("ss", $nombre, $correo);
        $stmt->execute();
    }
}
?>
```

5. Seguridad y Buenas Prácticas

Prevenir Inyección SQL

- Usar siempre consultas preparadas
- Validar y sanear entradas de usuario
- Usar prepared statements

Gestión de Errores

```php
<?php
try {
    // Código propenso a errores
} catch (Exception $e) {
    // Manejo de errores
    error_log($e->getMessage());
    echo "Ocurrió un error. Intente nuevamente.";
}
?>
```

6. Consejos Adicionales

- Usar variables de entorno para credenciales
- Implementar CSRF tokens
- Sanitizar siempre entradas de usuario
- Usar frameworks como Laravel para proyectos grandes

7. Recursos de Aprendizaje

- Documentación oficial de PHP: php.net
- Tutoriales en línea gratuitos
- Cursos en plataformas como Udemy