1.1. Introducción a PHP y sintaxis básica

LENGUAJES DE PROGRAMACIÓN NIVEL 4. UNIDAD 1

PHP - LENGUAJE DE PROGRAMACION PARA EL BACKEND

Financiado por la Unión Europea

Plan de Recuperación, Transformación y Resiliencia

Contenido

Introducción

PHP, que significa "PHP: Hypertext Preprocessor", es un lenguaje de programación de servidor ampliamente utilizado y diseñado principalmente para el desarrollo web. Se utiliza para crear páginas web dinámicas e interactivas.

A diferencia del HTML que se ejecuta en el cliente (navegador del usuario), PHP se ejecuta en el servidor, generando el HTML que luego se envía al cliente. Esto permite a los desarrolladores crear contenidos que pueden cambiar dependiendo de la interacción del usuario, datos almacenados en el servidor o cualquier otra condición que el servidor pueda procesar.

Lenguajes de programación de servidor

Un lenguaje de programación de servidor, también conocido como lenguaje de programación del lado del servidor o backend, es un tipo de lenguaje utilizado para desarrollar programas que se ejecutan en el servidor, en lugar de en la máquina del usuario (el cliente). Estos lenguajes son una parte esencial de la programación web y son responsables de gestionar la lógica de la aplicación, la interacción con las bases de datos, la manipulación de datos y la ejecución de las operaciones necesarias para generar el contenido web que los usuarios finales solicitan y ven.

Características de los lenguajes de programación de servidor:

- Ejecución en el Servidor:
  A diferencia de los lenguajes del lado del cliente que se ejecutan en el navegador del usuario (como JavaScript), los lenguajes del lado del servidor se ejecutan en un entorno de servidor. Esto significa que el procesamiento y la generación de las respuestas se realizan en un servidor remoto, y el resultado (generalmente HTML, CSS y JavaScript) se envía al navegador del cliente para su visualización.
- Interacción con Bases de Datos:
  Los lenguajes del lado del servidor están diseñados para interactuar eficientemente con bases de datos, recuperando y almacenando datos que son fundamentales para la funcionalidad de las aplicaciones web.

- Control de la Lógica de la Aplicación:
  La lógica empresarial, como las reglas de negocio, las validaciones y las transacciones, se maneja en el servidor, proporcionando una capa de seguridad y control que no estaría disponible si se realizara en el cliente.
- Manejo de Solicitudes HTTP:
  Estos lenguajes manejan las solicitudes HTTP que vienen del cliente, procesan esas solicitudes y formulan respuestas adecuadas.
- Generación de Contenido Dinámico:
  A diferencia del contenido estático que es el mismo para todos los usuarios, los lenguajes del lado del servidor pueden generar contenido dinámico personalizado para cada usuario, como páginas de perfil o dashboards con información actualizada.

Ejemplos de lenguajes de programación de servidor:

Algunos de estos programas los veremos a lo largo de este nivel:

- PHP: Es uno de los lenguajes de programación de servidor más conocidos y ampliamente utilizados. PHP es especialmente popular para el desarrollo de aplicaciones web debido a su facilidad de uso y su integración con el sistema de gestión de bases de datos MySQL.
- Python: Con frameworks como Django y Flask, Python se ha convertido en un lenguaje muy popular para el desarrollo web debido a su legibilidad y eficiencia.
- Ruby: Gracias a frameworks como Ruby on Rails, Ruby es conocido por su enfoque en la convención sobre la configuración, lo que permite un desarrollo rápido y eficiente.
- JavaScript (Node.js): Aunque JavaScript es tradicionalmente un lenguaje del lado del cliente, Node.js ha permitido que JavaScript se ejecute en el servidor, ofreciendo un entorno de ejecución rápido y escalable.
- Java: Utilizado en una gran cantidad de grandes empresas y aplicaciones corporativas, Java es conocido por su robustez, portabilidad y extensa biblioteca de clases que facilita la creación de aplicaciones web complejas y de gran escala.
- C# con .NET: C# es un lenguaje de programación potente y de tipado fuerte utilizado junto con el framework .NET de Microsoft para

construir aplicaciones web robustas y de alto rendimiento en la plataforma ASP.NET.

En resumen, los lenguajes de programación de servidor son fundamentales para el desarrollo de aplicaciones web porque realizan tareas esenciales que no pueden llevarse a cabo en el lado del cliente por razones de seguridad, acceso a recursos del servidor, o simplemente por la necesidad de realizar procesamientos que serían demasiado pesados o inseguros de realizar en un navegador. Al elegir un lenguaje de programación de servidor, los desarrolladores consideran factores como la escalabilidad, la seguridad, la facilidad de uso, el soporte comunitario y la eficiencia para satisfacer las necesidades específicas de cada proyecto web.

Un poco de historia sobre PHP y su evolución

PHP fue creado por Rasmus Lerdorf en 1994 inicialmente como una simple colección de scripts para rastrear visitas a su currículum vitae en línea. Desde entonces, se ha desarrollado y evolucionado en un poderoso lenguaje de programación con un rico ecosistema de frameworks y una gran comunidad de desarrolladores.

A lo largo de sus versiones, PHP ha añadido muchas características nuevas, mejoras de rendimiento y capacidades de orientación a objetos, lo que lo ha mantenido relevante en la rápida evolución del desarrollo web.

Ventajas y usos comunes de PHP en la web actual

PHP tiene varias ventajas que lo han mantenido popular entre los desarrolladores:

- Facilidad de Uso: PHP es conocido por su sintaxis simple y legible, lo que lo hace accesible para principiantes, mientras que su profunda funcionalidad satisface las necesidades de los programadores avanzados.
- Flexibilidad: PHP puede ser incrustado directamente en el código HTML y se integra bien con varios sistemas de gestión de bases de datos.
- Costo-efectividad: PHP es de código abierto y se ejecuta en diversas plataformas como Windows, Linux y macOS, lo que reduce los costos de desarrollo.

- Amplio Soporte: Existe una amplia cantidad de documentación, frameworks y bibliotecas disponibles que facilitan el desarrollo de aplicaciones robustas y seguras.

En la web de hoy, PHP se utiliza en una variedad de sitios, desde blogs y sistemas de gestión de contenidos hasta aplicaciones de comercio electrónico y portales de trabajo. Grandes plataformas como WordPress, Drupal y Joomla están construidas en PHP, lo que demuestra su adaptabilidad y durabilidad en el desarrollo web.

PHP sigue siendo una herramienta esencial en el kit de desarrollo web, ofreciendo una solución completa para el servidor que puede manejar formularios, sesiones, cookies y mucho más, manteniendo su posición como uno de los lenguajes de programación de servidor más utilizados en la creación y mantenimiento de sitios web dinámicos y ricos en datos.

Entorno de desarrollo

Antes de sumergirnos en el emocionante mundo del desarrollo web con PHP, es esencial contar con un espacio de trabajo adecuado.

Este módulo es esencial porque establece las bases que permitirán probar y ejecutar scripts de PHP. En este sentido, cuentas con dos opciones:

- Por un lado, configurar un entorno local que es una solución que requiere más esfuerzo, pero es la solución más profesional si quieres profundizar en PHP
- Por otro, utilizar soluciones en línea que pueden ser una alternativa conveniente para los principiantes que aún se están familiarizando con el lenguaje.

Ambas opciones tienen su lugar: una es ideal para aprender y hacer pruebas rápidas, y la otra es indispensable para un planteamiento de conocimiento más profundo y profesional de la herramienta.

Alternativas en línea

Para quienes están dando sus primeros pasos en el mundo del desarrollo web con PHP, configurar un entorno de desarrollo local puede parecer una tarea desalentadora. Afortunadamente, existen alternativas en línea que ofrecen una manera simple y rápida de comenzar a codificar en PHP sin la necesidad de configuraciones complejas.

Plataformas en línea para practicar PHP

- PHP Fiddle: Es un entorno en línea donde puedes escribir y ejecutar pequeños fragmentos de código PHP. Es ideal para experimentar con pedazos de código y compartirlos con otros.
  https://php-fiddle.com/
- Repl.it: Esta plataforma proporciona un entorno de desarrollo integrado (IDE) en el navegador que es perfecto para principiantes. Puedes escribir, ejecutar y probar tus scripts PHP sin instalar nada en tu computadora. Además, Repl.it facilita la colaboración y el compartir código, lo cual es excelente para el aprendizaje en el aula o en grupos de estudio.
  https://replit.com/
- Paiza.IO: Esta herramienta ofrece un entorno sencillo y efectivo para escribir y ejecutar código PHP. Es ideal para realizar pruebas rápidas y ejercicios de codificación.
  https://paiza.io/es

Ventajas de las plataformas en línea

- Accesibilidad: No se requiere instalación, lo que significa que puedes comenzar a programar desde cualquier computadora con acceso a internet.
- Facilidad de Uso: Las interfaces suelen ser limpias y fáciles de navegar, lo que permite a los principiantes concentrarse en aprender PHP sin distracciones.
- Inmediatez: Puedes ver los resultados de tu código de inmediato, lo que proporciona una gratificación instantánea y un ciclo de aprendizaje más rápido.
- Compartir y Colaborar: Muchas plataformas permiten compartir tu código fácilmente, lo que es útil para obtener ayuda y trabajar en proyectos de grupo.

Consideraciones importantes

Aunque las plataformas en línea son excelentes para comenzar y son una opción conveniente para la práctica y el aprendizaje rápido, tienen sus limitaciones. Para proyectos más grandes, desarrollo colaborativo a gran escala, y para una experiencia de desarrollo más realista y profesional, es beneficioso aprender a configurar y utilizar un entorno de desarrollo local.

Además, las habilidades adquiridas al configurar un servidor local son transferibles y valiosas en el mercado laboral.

Configuración de un entorno de desarrollo

Al sumergirnos en el aprendizaje de PHP, un lenguaje de programación de servidor es esencial contar con un entorno que simule el servidor web donde nuestras aplicaciones cobrarán vida. Mientras que las soluciones en línea ofrecen una manera rápida y conveniente de experimentar con PHP, configurar un entorno de desarrollo local es un paso crucial para cualquier aspirante a desarrollador web. Esto se debe a que PHP, al ser un lenguaje de servidor, necesita un servidor web para interpretar y ejecutar el código que escribimos.

Configurar este entorno en tu propio equipo te brinda una experiencia más cercana a la realidad del desarrollo profesional, donde tendrás control total sobre tu espacio de trabajo y podrás experimentar con todas las características del lenguaje.

¿Qué es un servidor local?

Un servidor local se refiere a un entorno de servidor que se configura y ejecuta en una computadora personal o en una red interna, en lugar de en un servidor remoto o en la nube accesible a través de Internet. En el contexto del desarrollo web, un servidor local actúa como un entorno de pruebas que simula el funcionamiento de un servidor web real, permitiendo a los desarrolladores escribir, probar y depurar sus aplicaciones web en su propia máquina antes de desplegarlas en un entorno de producción en línea.

¿Por qué configurar un entorno local?

- Simulación de un Servidor Real: Un entorno local de PHP te permite probar tus scripts y aplicaciones en un contexto que imita un servidor de hosting real, preparándote para la publicación y el despliegue de aplicaciones web en el futuro.
- Desarrollo Offline: Puedes seguir desarrollando y probando tus aplicaciones incluso sin conexión a internet, lo cual es vital para garantizar la productividad en todo momento.

- Aprendizaje Profundo: Al configurar tu entorno, ganarás conocimientos valiosos sobre cómo funciona PHP junto con otros componentes del servidor, como bases de datos y servidores web.
- Personalización y Control: Tendrás la libertad de personalizar tu entorno según tus necesidades específicas, optimizando tu flujo de trabajo y el rendimiento de tus aplicaciones.

Pasos para la configuración local

El proceso se compone de los siguientes pasos:

1. Elección del software de servidor local

Al iniciar en el desarrollo con PHP, una de las primeras decisiones que debes tomar es qué software de servidor local utilizarás. Este software es esencial porque emula un servidor web en tu propia máquina, permitiéndote desarrollar y probar aplicaciones PHP sin necesidad de un servidor web real. Aquí hay algunos puntos a considerar:

Opciones populares

- XAMPP: Uno de los paquetes de servidor local más populares, XAMPP es una excelente opción para principiantes debido a su fácil instalación y configuración. Incluye Apache (el servidor web), MariaDB (la base de datos), PHP y Perl. Es multiplataforma, lo que significa que funciona en Windows, Linux y macOS. Esta es la opción que te recomiendo.
- WAMP: Específico para usuarios de Windows, WAMP (Windows, Apache, MySQL, PHP) es otra opción conveniente para configurar un entorno de desarrollo PHP. Proporciona una interfaz de usuario simple para gestionar el servidor y los servicios.
- MAMP: Originalmente diseñado para macOS (Mac, Apache, MySQL, PHP), MAMP ahora también está disponible para Windows. Ofrece una instalación sencilla y es una buena opción para los usuarios de Mac que buscan una solución fácil para configurar un entorno local de PHP.
- LAMP: Para usuarios de Linux, LAMP (Linux, Apache, MySQL, PHP) es la opción estándar. Aunque la instalación y configuración pueden ser un poco más técnicas en comparación con las soluciones basadas en Windows o macOS, LAMP ofrece un entorno de desarrollo potente y flexible.

Consideraciones para la elección

- Compatibilidad del Sistema Operativo: Asegúrate de que el paquete de software que elijas sea compatible con tu sistema operativo.
- Facilidad de Uso: Si eres principiante, es posible que prefieras una solución que ofrezca una configuración más guiada o interfaces gráficas de usuario (GUI) para gestionar tus servicios.
- Comunidad y Soporte: Considera la popularidad del paquete y la disponibilidad de tutoriales, foros y recursos en línea que puedan ayudarte si te encuentras con problemas.
- Requisitos del Proyecto: Aunque la mayoría de los paquetes ofrecen características similares, algunos pueden incluir herramientas adicionales o configuraciones específicas que podrían ser beneficiosas para tus proyectos.

2. Instalación del software elegido

Una vez que hayas elegido el software de servidor local que mejor se adapte a tus necesidades, el siguiente paso es descargarlo del sitio web oficial y seguir el proceso de instalación. Este proceso generalmente implica ejecutar el instalador y seguir las instrucciones en pantalla, que te guiarán a través de la configuración de los componentes básicos como Apache, MySQL/MariaDB y PHP.

Es importante realizar la instalación con cuidado, prestando atención a las opciones de configuración, especialmente si estás personalizando la instalación para adaptarla a tus necesidades específicas de desarrollo.

3. Configuración

Una vez instalado tu servidor local, hay varios ajustes que puedes realizar para optimizar tu entorno de desarrollo PHP:

Verificación de componentes

- Inicio de Servicios: Asegúrate de que los servicios necesarios, como Apache (el servidor web) y MySQL/MariaDB (el sistema de gestión de bases de datos), se inician automáticamente al arrancar tu servidor local. Esto suele hacerse desde el panel de control del paquete de software que hayas elegido.

- Prueba de PHP: Para verificar que PHP esté funcionando correctamente, crea un archivo de prueba llamado info.php en tu directorio htdocs (o el equivalente en tu paquete). Dentro de este archivo, escribe el siguiente código PHP:

```
<?php phpinfo(); ?>
```

[1]

Navega a http://localhost/info.php en tu navegador. Si ves una página que muestra información sobre tu configuración de PHP, significa que PHP está funcionando correctamente.

4. Creación de proyectos:

La organización y ubicación correcta de tus archivos de proyecto son esenciales para el desarrollo eficiente y la correcta ejecución de tus aplicaciones web en PHP.

Aquí te explico cómo proceder una vez que tienes tu servidor local configurado.

Creación y organización de proyectos PHP

- Ubicación de los Archivos del Proyecto
  - Directorio Raíz: En la mayoría de los entornos de servidor local como XAMPP, WAMP o MAMP, hay un directorio designado como la "raíz del documento" donde debes ubicar tus archivos de proyecto para que puedan ser servidos por el servidor web Apache. Comúnmente, este directorio se llama htdocs en XAMPP y MAMP, y www en WAMP.
  - Creación de Directorios de Proyectos: Dentro del directorio htdocs, es una buena práctica crear un nuevo directorio para cada proyecto individual. Esto mantiene tus proyectos organizados y separados, lo cual es especialmente útil si trabajas en múltiples proyectos. Por ejemplo, si tu proyecto se llama "MiPrimerSitio", crearías un directorio dentro de htdocs con ese nombre.
- Estructura de Archivos del Proyecto

[1] Accesibilidad:
<?php phpinfo(); ?>

- Dentro del directorio de tu proyecto, puedes comenzar a crear la estructura de archivos y directorios según las necesidades de tu aplicación. Una estructura básica podría incluir:
  - index.php: Este suele ser el archivo principal y el punto de entrada de tu aplicación web. Aquí es donde comienza la lógica de tu aplicación y se generan las respuestas a las solicitudes del usuario.
  - Directorios de Organización: Según la complejidad de tu proyecto, puedes tener directorios separados para tus scripts PHP, archivos de estilo (css), scripts de JavaScript (js), imágenes (img), y otros recursos. Mantener una estructura organizada te ayuda a gestionar mejor tu código a medida que tu proyecto crece.

- Acceso a tus Proyectos
  - URL Local: Para ver tu proyecto en un navegador, debes usar una URL que apunte a tu servidor local y al directorio específico de tu proyecto. Si tu directorio de proyecto se llama "MiPrimerSitio", accederías a él a través de http://localhost/MiPrimerSitio en tu navegador.
  - Desarrollo y Pruebas: Trabajar en este entorno te permite desarrollar y probar tu aplicación en un entorno que simula un servidor web real. Puedes hacer cambios en tus archivos PHP, recargar tu navegador y ver los resultados de inmediato.

Buenas prácticas

- Nomenclatura Consistente: Usa nombres de archivos y directorios que sean claros y descriptivos. Evita usar espacios y caracteres especiales en los nombres.
- Separación de Preocupaciones: Mantén separados tu código PHP, HTML, CSS y JavaScript tanto como sea posible. Esto mejora la legibilidad y mantenibilidad de tu código.
- Permisos de Archivos: Asegúrate de que los archivos y directorios en tu proyecto tengan los permisos adecuados para que Apache pueda leerlos y servirlos sin problemas.

Organizar correctamente tus archivos de proyecto y entender cómo ubicarlos en el directorio adecuado del servidor local es fundamental para el desarrollo web en PHP. Establecer una estructura clara desde el principio facilita la gestión de tu código y te ayuda a evitar problemas cuando tu

aplicación crece en complejidad. Con estas bases, estás listo para sumergirte en el desarrollo de aplicaciones PHP y experimentar con todo lo que el lenguaje tiene para ofrecer.

5. Pruebas:

Finalmente, te mostraré cómo iniciar y detener los servicios del servidor, cómo acceder a tus proyectos PHP a través del navegador y cómo interpretar los mensajes de error para resolver problemas comunes.

Iniciar y detener los servicios del servidor

- Panel de Control: La mayoría de los paquetes de servidor local como XAMPP, WAMP o MAMP vienen con un panel de control. Desde aquí, puedes iniciar y detener fácilmente servicios individuales como Apache (el servidor web) y MySQL (la base de datos). Simplemente haz clic en 'Start' para iniciar un servicio y 'Stop' para detenerlo.
- Comprobación del Estado: Asegúrate de que Apache y MySQL estén en ejecución antes de intentar acceder a tus proyectos. En el panel de control, deberías ver indicadores o mensajes que muestran el estado actual de cada servicio.

Acceder a tus proyectos PHP

- URL Local: Para ver tu proyecto en acción, abre un navegador web y escribe la URL correspondiente a tu proyecto local. Por lo general, esta URL sigue el formato http://localhost/nombre_del_proyecto, donde nombre_del_proyecto es el nombre del directorio de tu proyecto dentro de htdocs o el equivalente.
- Visualización de Cambios: Cada vez que realices cambios en tus archivos PHP, simplemente guarda los cambios y recarga la página en tu navegador para ver los efectos de tus modificaciones.

Interpretación de mensajes de error

- Errores Comunes: Es común encontrarse con errores al desarrollar en PHP. Estos pueden incluir errores de sintaxis, problemas de conexión a la base de datos, errores en las rutas de archivos, entre otros.
- Mensajes de Error de PHP: PHP proporciona mensajes de error bastante descriptivos que te ayudarán a identificar la línea y el archivo donde ocurrió el error. Asegúrate de que la configuración de display_errors esté activada en tu archivo php.ini para ver los errores directamente en tu navegador.

- Logs de Error: Además de mostrar errores en el navegador, también puedes revisar los logs de error de Apache y PHP para obtener más detalles sobre los problemas. La ubicación de estos logs puede variar, pero generalmente se encuentran dentro del directorio de instalación de tu servidor local.
- Depuración: Herramientas como Xdebug pueden facilitar enormemente la depuración de tus scripts PHP al permitirte establecer puntos de interrupción, inspeccionar variables y seguir el flujo de ejecución de tu código.

Realizar pruebas efectivas en tu entorno de desarrollo local es crucial para el desarrollo web en PHP. Aprender a iniciar y detener los servicios del servidor, acceder a tus proyectos y manejar los mensajes de error te equipará con las habilidades necesarias para desarrollar, probar y depurar tus aplicaciones de manera eficiente. Estas prácticas no solo te ayudarán a construir aplicaciones más robustas y confiables, sino que también te prepararán para manejar desafíos más complejos a medida que avanzas en tu carrera de desarrollo web.

Configuración del espacio de trabajo y herramientas necesarias

Una vez que hayas instalado el software de servidor local como XAMPP, WAMP, MAMP o LAMP, el siguiente paso es configurar tu entorno de desarrollo. Esto incluye preparar tu espacio de trabajo y asegurarte de que tienes todas las herramientas necesarias para comenzar a programar en PHP.

Preparación del espacio de trabajo

- Directorio de Proyectos: Los paquetes de software como XAMPP y WAMP tienen un directorio predeterminado donde debes guardar tus proyectos PHP. Por ejemplo, en XAMPP, este directorio suele ser htdocs dentro del directorio de instalación de XAMPP. Aquí es donde crearás tus carpetas de proyectos y archivos PHP.
- Pruebas Locales: Para acceder a tus proyectos en un navegador, normalmente navegarás a localhost/nombre_del_proyecto. "localhost" es el nombre de host que representa tu máquina local, y nombre_del_proyecto es el nombre de la carpeta donde guardaste tu script PHP en el directorio htdocs.

Herramientas de desarrollo

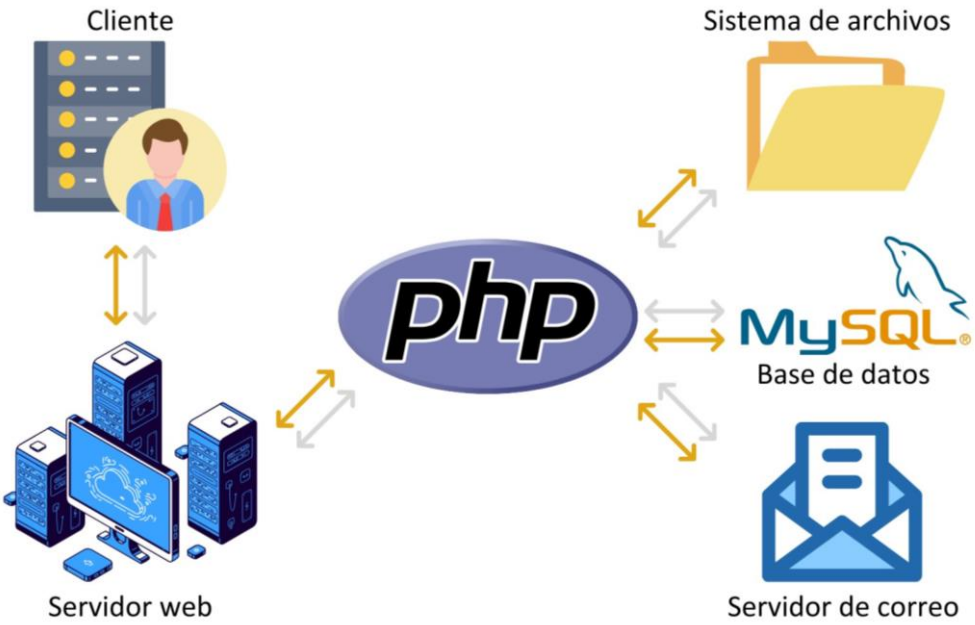
- Editor de Código: Una herramienta esencial es un buen editor de código o un Entorno de Desarrollo Integrado (IDE) que te ayude a escribir tu código eficientemente. Visual Studio Code, Sublime Text y PHPStorm son opciones populares, cada una con sus propias características, como resaltado de sintaxis, autocompletado de código y depuración.
- Consola/Terminal: Familiarizarte con la línea de comandos o terminal de tu sistema operativo es útil para ejecutar scripts PHP, gestionar bases de datos con MySQL, y realizar tareas de mantenimiento.
- Navegador Web: Es fundamental para probar y visualizar tus aplicaciones web. Herramientas de desarrollo integradas en navegadores como Chrome o Firefox pueden ayudarte a depurar tu código HTML, CSS y JavaScript.

Configuración adicional

- Configuración de PHP y Apache: Es posible que necesites ajustar la configuración de PHP y Apache para adaptarla a tus necesidades de desarrollo. Esto se hace editando archivos de configuración como php.ini para PHP y httpd.conf para Apache. Cambios comunes incluyen aumentar el límite de memoria, activar extensiones PHP y configurar la versión de PHP.
- Base de Datos: Si tu proyecto requiere una base de datos, deberás configurar MySQL o MariaDB que vienen incluidos en tu paquete. Esto incluye la creación de bases de datos y usuarios, y la gestión de permisos. Herramientas como phpMyAdmin, que a menudo se incluyen en estos paquetes, facilitan la gestión de tus bases de datos a través de una interfaz web.

Configurar adecuadamente tu entorno de trabajo y familiarizarte con las herramientas necesarias son pasos fundamentales para un flujo de desarrollo eficiente en PHP. Al tomar el tiempo para ajustar tu espacio de trabajo a tus preferencias personales y a las necesidades de tu proyecto, te aseguras de tener una base sólida sobre la cual construir aplicaciones web complejas y funcionales. Con un entorno bien configurado, estás listo para sumergirte en el desarrollo de proyectos en PHP, experimentar con el lenguaje y empezar a dar vida a tus ideas para la web.

Sintaxis básica de PHP



Adentrarse en el mundo de PHP comienza con entender su sintaxis básica, el conjunto de reglas que define cómo escribir instrucciones que el intérprete de PHP puede entender y ejecutar. Esta comprensión es esencial no solo para escribir código que funcione sino también para mantenerlo legible, mantenible y libre de errores.

A medida que avanzas en PHP y en otros módulos de este curso, encontrarás que una base sólida en la sintaxis básica te facilitará el aprendizaje de conceptos más avanzados y la resolución de problemas complejos

Estructura de un archivo PHP

Un archivo PHP típicamente contiene código PHP mezclado con HTML. El código PHP se encierra en etiquetas especiales que le indican al servidor que debe procesarlo como tal.

Un ejemplo simple de un archivo PHP podría ser una página web que usa PHP para mostrar la fecha y hora actuales. El archivo podría tener tanto HTML estándar como bloques de código PHP insertados.

```
<!DOCTYPE html>
<html>
<head>
    <title>Mi Primera Página PHP</title>
</head>
<body>

<h1>Bienvenido a mi página</h1>
<p>La fecha y hora actual es: <?php echo date('Y-m-d H:i:s'); ?></p>

</body>
</html>
```

2

Etiquetas PHP para abrir y cerrar código

Las etiquetas más comunes para delimitar el código PHP son <?php para abrir y ?> para cerrar. Todo el código PHP debe estar entre estas etiquetas.

```
<?php
// Tu código PHP aquí
?>
```

3

[2] Accesibilidad:

```
<!DOCTYPE html>
<html>
<head>
    <title>Mi Primera Página PHP</title>
</head>
<body>

<h1>Bienvenido a mi página</h1>
<p>La fecha y hora actual es: <?php echo dato("Y-m-d:i:s"); ?></p>

</body>
</html>
```

[3] Accesibilidad:

```
<?php
// Tu código PHP aquí
?>
```

Algunas consideraciones:

- Etiquetas Cortas: En algunos entornos, las etiquetas cortas <? y ?> también pueden usarse, pero no son recomendables porque pueden estar deshabilitadas en la configuración del servidor y no son tan claras como las etiquetas estándar.
- Etiqueta de Cierre Opcional: En archivos que contienen solo código PHP, la etiqueta de cierre ?> es opcional y a menudo se omite para evitar la inserción accidental de espacios en blanco después de la etiqueta, lo que podría causar errores.

Comentarios en PHP

Los comentarios son cruciales para documentar tu código. Permiten explicar la funcionalidad y las decisiones de diseño, facilitando tanto el trabajo en equipo como tu propia comprensión cuando revisas el código más adelante.

Tipos de comentarios:

- Comentarios de una sola línea: Se pueden hacer con // o #.

```
// Este es un comentario de una sola línea
# Este también es un comentario de una sola línea
```

4

[4] Accesibilidad
// Este es un comentario de una sola línea
Este también es un comentario de una sola línea

- Comentarios de varias líneas o comentarios de bloque: Se delimitan con /* para abrir y */ para cerrar.

```
/*
Este es un comentario
de varias líneas
*/
```

[5]

Dominar la sintaxis básica de PHP es el primer paso esencial en tu camino como desarrollador PHP. A través de una comprensión clara de la estructura de los archivos PHP, el uso adecuado de las etiquetas y la importancia de los comentarios, estarás bien preparado para escribir código limpio, eficiente y mantenible. Estos fundamentos te servirán de base mientras exploras conceptos más avanzados y comienzas a construir aplicaciones web más complejas.

[5] Accesibilidad:
/*
Este es un comentario
de varias líneas
*/

Primeros pasos en PHP

Tras familiarizarnos con la sintaxis básica de PHP, es hora de poner en práctica ese conocimiento y dar nuestros primeros pasos escribiendo código PHP funcional.

En este módulo, aprenderemos cómo PHP interactúa con el usuario a través de la salida de texto, la importancia de organizar y reutilizar el código mediante include y require, y cómo PHP se integra de manera fluida con HTML para crear páginas web dinámicas. Estos conceptos son piedras angulares en el desarrollo web con PHP, permitiéndote crear aplicaciones web interactivas y eficientes.

Salida de texto en pantalla

- Echo y Print:
  PHP ofrece varias maneras de enviar salida al navegador. Las construcciones echo y print son las más comunes para mostrar texto. Aunque son muy similares, echo es ligeramente más rápido y puede aceptar múltiples parámetros, mientras que print solo acepta uno y siempre devuelve 1, lo que significa que puede usarse en expresiones.

```php
<?php
echo "¡Hola, mundo!";
print "Bienvenido a PHP";
```

[6]

- HTML Integrado:
  PHP puede incrustarse directamente en HTML, lo que permite insertar dinámicamente contenido o datos en una página web.

```php
<p>La hora actual es: <?php echo date('H:i:s'); ?></p>
```

[7]

Incluir archivos PHP

Mantener tu código DRY (Don't Repeat Yourself - No te repitas) es una práctica fundamental en la programación. PHP permite incluir archivos mediante las instrucciones include y require, lo que facilita la reutilización de código como encabezados, pies de página, o bloques de funciones comunes en múltiples páginas.

[6] Accesibilidad
<?php
echo “¡Hola, mundo!”;
print “Bienvenido a PHP”;

[7] Accesibilidad
La hora actual es: <?php echo date('H:i:s'); ?>

La diferencia principal entre include y require radica en cómo manejan los errores:

- include producirá una advertencia si el archivo no se encuentra, pero el script continuará ejecutándose.
- require, por otro lado, producirá un error fatal y detendrá la ejecución del script.

```php
<?php
include 'header.php';
require 'functions.php';
```

8

Diferenciación entre código PHP y HTML

Uno de los aspectos más poderosos de PHP es su capacidad para integrarse dentro de HTML. Puedes abrir y cerrar bloques PHP tantas veces como sea necesario dentro de un archivo .php, lo que te permite insertar o calcular dinámicamente contenido HTML.

```php
<ul>
<?php for ($i = 1; $i <= 5; $i++): ?>
    <li>Elemento <?php echo $i; ?></li>
<?php endfor; ?>
</ul>
```

9

[8] Accesibilidad
<?php
include 'header.php';
require 'functions.php'

[9] Accesibilidad

<?php for ($i = 1; $i <= 5; $i++): ?>
Elemento <?php echo $i; ?></1i>
<?php endfor; ?>

Buenas prácticas iniciales

A medida que te embarcas en el aprendizaje de PHP y comienzas a escribir tus primeros scripts, es crucial adoptar buenas prácticas desde el inicio. Estas prácticas no solo te ayudarán a evitar errores comunes, sino que también asegurarán que tu código sea fácil de leer, entender y mantener, tanto para ti como para otros desarrolladores que puedan trabajar contigo.

En este módulo, exploraremos algunas de las mejores prácticas iniciales en PHP, que formarán una base sólida para tu desarrollo como programador PHP.

Convenciones de nomenclatura

- Variables y Funciones: Elige nombres claros y descriptivos para tus variables y funciones. Para las variables, usa nombres que reflejen el valor que almacenan; para las funciones, usa verbos que indiquen la acción que realizan.
- Estilos de Nomenclatura: En PHP, es común usar camelCase para nombrar variables y funciones (miVariable, calcularTotal). Para nombres de clases, se utiliza PascalCase (MiClase). Aunque PHP no es sensible al estilo de nomenclatura, adoptar un estilo consistente mejora la legibilidad.

Evitar errores comunes

- Inicialización de Variables: Asegúrate de inicializar tus variables antes de usarlas. Esto te ayuda a evitar errores de "variable no definida" y asegura que tu código se comporte como esperas.
- Uso de Comillas: PHP permite el uso de comillas simples (') y dobles ("), pero su uso puede afectar cómo se interpretan las variables y los caracteres de escape dentro de las cadenas. Entender la diferencia y usarlas consistentemente puede ayudarte a evitar errores sutiles.
- Validación de Datos: Al trabajar con datos de entrada del usuario, siempre valida y limpia estos datos para proteger tu aplicación de inyecciones de código malicioso y otros problemas de seguridad.

Legibilidad y mantenibilidad del código

- Comentarios: Usa comentarios para explicar el "por qué" detrás de bloques de código complejos o decisiones de diseño no obvias. No obstante, evita comentar lo obvio; tu código debe ser lo suficientemente claro para explicar el "qué" por sí mismo.
- Consistencia: Mantén una consistencia en todo tu código: estilo de nomenclatura, indentación, uso de espacios en blanco, etc. Considera usar herramientas de formateo de código o seguir una guía de estilo PHP.
- Refactorización: A medida que aprendas más y tu proyecto crezca, revisa tu código anterior buscando oportunidades para refactorizarlo. Esto puede incluir simplificar la lógica, eliminar código duplicado o dividir bloques grandes de código en funciones más pequeñas y manejables.

Adoptar buenas prácticas desde el principio de tu viaje de aprendizaje en PHP no solo te ayudará a escribir código más limpio y eficiente, sino que también te preparará para proyectos más grandes y colaboración con otros desarrolladores. Recuerda que el objetivo final es escribir código que no solo funcione, sino que también sea fácil de leer, entender y mantener.

Conclusión

PHP, desde su creación, ha jugado un papel crucial en el mundo del desarrollo web, especialmente en el backend. Su flexibilidad, facilidad de uso y amplia adopción lo han consolidado como un pilar en la creación de sitios web dinámicos y aplicaciones web. A través de este módulo, hemos establecido las bases de PHP, desde la sintaxis básica hasta las buenas prácticas iniciales, sentando así los cimientos para tu viaje de desarrollo web.

Entender profundamente los fundamentos de PHP es esencial antes de adentrarse en aspectos más avanzados y complejos del desarrollo web. Una base sólida te permite escribir código más eficiente, solucionar problemas de manera efectiva y adaptarte a nuevas tecnologías y frameworks que utilizan PHP. A medida que avanzas, descubrirás que muchos principios aprendidos aquí se aplican no solo dentro de PHP sino también en otros lenguajes y tecnologías.

El aprendizaje de PHP, como cualquier otro lenguaje de programación, es una jornada continua. Te animo a seguir explorando, experimentando y construyendo con PHP. Aprovecha los recursos disponibles, participa en proyectos reales, contribuye a la comunidad y, lo más importante, no temas cometer errores, ya que son parte esencial del proceso de aprendizaje.

Este módulo marca el comienzo de tu aventura en el desarrollo backend con PHP. Esperamos que te haya proporcionado una comprensión clara de los aspectos básicos y te haya inspirado a seguir profundizando en tus conocimientos de PHP. Recuerda, cada línea de código que escribes te acerca un paso más a convertirte en el desarrollador que aspiras ser. Sigue curioso, sigue aprendiendo y, sobre todo, disfruta del viaje.

Nos vemos en el siguiente modulo.

Recursos adicionales

A medida que avanzas en tu viaje de aprendizaje de PHP, es invaluable contar con una variedad de recursos que puedan ofrecerte diferentes perspectivas, ejemplos y desafíos. Los recursos siguientes han sido cuidadosamente seleccionados para complementar lo que has aprendido en este curso y para ayudarte a profundizar en áreas específicas de interés.

Documentación oficial de PHP

- PHP.net: La documentación oficial de PHP es el recurso definitivo para desarrolladores de todos los niveles. Ofrece una guía exhaustiva de todas las funciones de PHP, desde las más básicas hasta las más avanzadas, incluyendo ejemplos de código y notas de usuarios que pueden proporcionar insights adicionales.
  https://www.php.net/

Recursos educativos para principiantes

- W3Schools PHP Tutorial: W3Schools ofrece tutoriales introductorios que son fáciles de seguir y cubren una amplia gama de temas en PHP, perfectos para principiantes.
  https://www.w3schools.com/php/
- Codecademy's PHP Course: Codecademy ofrece un curso interactivo de PHP que te guía a través de los conceptos básicos con ejercicios prácticos, ideal para consolidar lo aprendido.
  https://www.codecademy.com/catalog/language/php4
- PHP: The Right Way: PHP: The Right Way es una guía de buenas prácticas popular que ofrece una visión general de PHP, incluyendo recomendaciones sobre estilos de codificación, prácticas recomendadas y enlaces a recursos de calidad.
  https://phptherightway.com/

Comunidades y foros

- Reddit r/PHP: El subreddit r/PHP es un buen lugar para mantenerse al día con las noticias de PHP, hacer preguntas y participar en discusiones con otros desarrolladores PHP.
  https://www.reddit.com/r/PHP/?rdt=45690
- PHP Freaks: PHP Freaks es un foro donde puedes encontrar respuestas a tus preguntas, participar en discusiones y conectarte con otros desarrolladores PHP.
  http://www.phpfreaks.com/