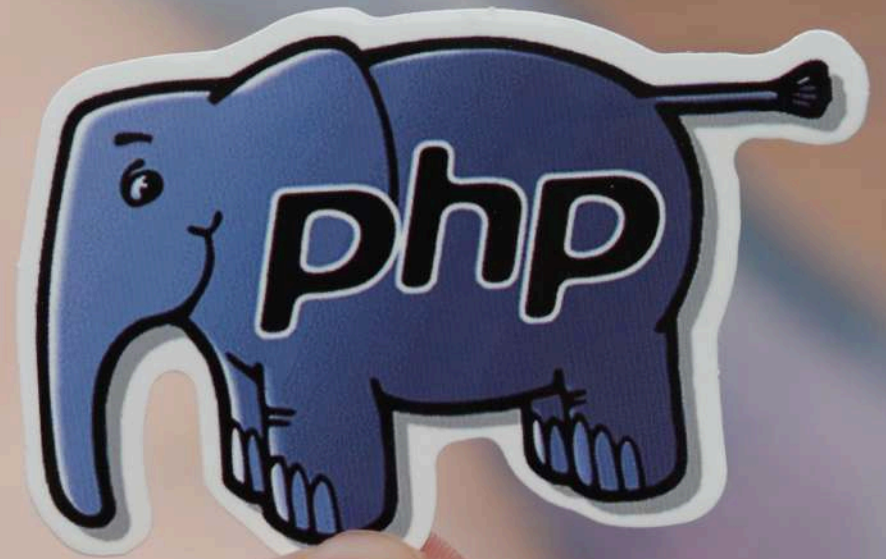


Bloque B

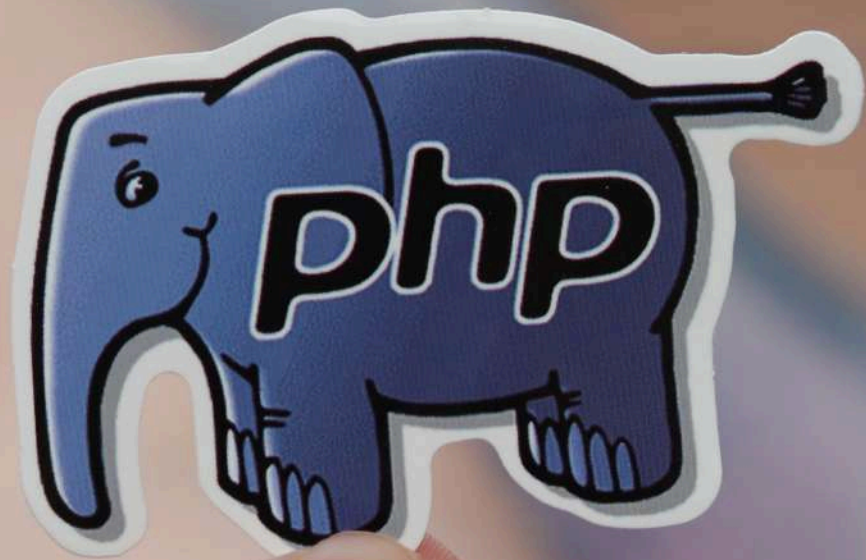
Páginas Web Dinámicas

Introducción



Contenidos

1. Introducción
2. Peticiones y respuestas HTTP
3. Esquemas de codificación
4. Herramientas integradas en el intérprete PHP
5. Arrays superglobales
6. Mensajes de error
7. Ajustes y opciones para el intérprete PHP
8. Contenidos del bloque



Introducción

Introducción

En este nuevo bloque aprenderemos cómo utilizar PHP para crear páginas web dinámicas. Estas son páginas donde el contenido que los usuarios ven puede cambiar sin que un programador altere manualmente el archivo.

Introducción

En el bloque A se introdujo la sintaxis del lenguaje PHP. Se mostró cómo:

- Variables y arrays almacenan datos
- Los operadores crean un único valor a partir de múltiples piezas de información
- Las condiciones y los bucles determinan cuándo se ejecuta el código
- Las funciones y clases agrupan sentencias relacionadas

Introducción

En este bloque, aprenderemos a aplicar estos conceptos básicos para crear páginas web dinámicas.

En un nivel básico, un ordenador es una máquina que está programada para:

- Aceptar datos conocidos como **entrada** (*input*)
- **Procesar** esos datos y realizar tareas con ellos
- Crear **salidas** (*output*) que los usuarios puedan ver u oír.
- También pueden **almacenar** datos para utilizarlos más adelante.

Introducción

Las páginas PHP que aprenderás a escribir en este bloque son como programas básicos; pueden aceptar información de un navegador web, procesar esos datos, y luego utilizarlos para generar una página HTML que se adapte a un visitante individual.

Aprenderás a:

- Utilizar un conjunto de funciones y clases que forman parte de PHP
- Recoger y procesar los datos enviados por los navegadores.
- Trabajar con imágenes y otros archivos que los usuarios pueden cargar
- Almacenar datos sobre los visitantes del sitio web utilizando cookies y sesiones.
- Tratar los errores y solucionar los problemas de tu código

Introducción

Para adentrarnos en este bloque, necesitamos entender cómo el intérprete PHP maneja las peticiones y cómo responde a ellas.

Para gestionar las peticiones de páginas, los servidores siguen reglas establecidas en protocolos y esquemas de codificación.

Introducción

Peticiones y respuestas HTTP

El **Protocolo de Transferencia de Hipertexto (HTTP)** es un conjunto de normas que controlan la comunicación entre navegadores y servidores. Por eso las URL de los sitios web empiezan por `http://` o `https://`. HTTP especifica qué datos

- Los navegadores envían a un servidor cuando solicitan un archivo.
- Los servidores envían al navegador cuando responden con el archivo solicitado.

Introducción

Esquemas de codificación

Los ordenadores representan texto, imágenes y audio utilizando datos binarios, que están formados por una serie de Os y 1s.

Los **esquemas de codificación** son reglas que utilizan los ordenadores para traducir lo que ves y oyes en Os y 1s que el ordenador procesa. Si no sabes cómo decirle a PHP qué esquema de codificación utilizar, puede que no procese o muestre los datos correctamente.

Introducción

El intérprete PHP viene con varias herramientas que te ayudan a construir páginas web dinámicas:

Arrays, funciones, clases

El intérprete PHP viene con conjuntos de:

- **Arrays superglobales:** son arrays que crea cada vez que se solicita un archivo.
- **Funciones incorporadas:** que realizan tareas que los programadores necesitan realizar a menudo.
- **Clases incorporadas:** para crear objetos que representan cosas con las que los programadores necesitan tratar a menudo.

Introducción

Mensajes de error

El intérprete PHP crea mensajes de error cuando encuentra un problema; aprender a leer estos mensajes te ayudará a solucionar problemas en tu código.

Introducción

Ajustes

Como muchos otros tipos de software, tanto el intérprete PHP como el servidor web tienen configuraciones que puedes controlar. Veremos como cambiar la configuración de ambos utilizando archivos de texto.



Peticiones y respuestas HTTP

Peticiones y respuestas HTTP

El Protocolo de Transferencia de Hipertexto (HTTP) es un conjunto de reglas que especifican

- cómo los navegadores deben **solicitar** páginas y
- cómo los servidores deben dar formato a su **respuesta**.

Esto ayuda a comprender qué datos se envían en cada paso.

Peticiones y respuestas HTTP

Cuando un navegador web **solicita** una página PHP, la barra de direcciones del navegador muestra una URL que especifica cómo el navegador puede encontrar esa página. Cada URL tiene:

- Un **protocolo** (para páginas web es HTTP o HTTPS)
- Un **host** (el servidor al que enviar la petición)
- Una **ruta** que identifica el archivo solicitado
- Una **cadena de consulta** opcional con datos adicionales que la página podría necesitar.

Peticiones y respuestas HTTP



The diagram illustrates the components of the URL `http://eg.link/year.php?year=2021`. Brackets above the URL categorize it into four parts: PROTOCOL, HOST, PATH, and QUERY STRING. The QUERY STRING is further broken down into NAME and VALUE.

| PROTOCOL | HOST | PATH | QUERY STRING |
|----------|---------|-----------|--------------|
| http:// | eg.link | /year.php | year=2021 |
| | | | NAME VALUE |

Peticiones y respuestas HTTP

Cuando se añade una cadena de consulta al final de una URL, cada dato que envía es como una variable; tiene un:

- **Nombre** que describe los datos que se envían. El nombre es el mismo cada vez que se utiliza la URL.
- **Valor** del dato. El valor puede cambiar cada vez que se solicita la página.

Peticiones y respuestas HTTP

Cuando un navegador solicita una página web, también envía **cabeceras de solicitud HTTP** al servidor. No se muestran en la ventana principal del navegador (como la URL), pero pueden verse en las herramientas para desarrolladores que vienen con la mayoría de los navegadores (ver imagen a continuación).

Las cabeceras contienen datos que el servidor puede encontrar útiles, y también son similares a una variable; tienen un:

- **Nombre** que describe qué datos se están enviando. El nombre es el mismo cada vez que se utiliza una URL.
- **Valor** del dato.

Peticiones y respuestas HTTP

Las cabeceras en la captura de pantalla que puedes ver a continuación, muestra:

- El idioma que habla el visitante (inglés de EE.UU.). En sitios multilingües, esto podría utilizarse para seleccionar el idioma correcto para un visitante.
- Que la solicitud del usuario procede de otra página web, junto con la URL de esa página.
- El tipo de navegador es Chrome en un Mac con OSX. Esto podría utilizarse para determinar si se debe enviar a un visitante a una versión de escritorio/móvil del sitio.

Peticiones y respuestas HTTP

| | × | Headers | Preview | Response | Initiator | Timing |
|------------------------|---|------------------|--|----------|-----------|--------|
| | ▼ | Request Headers | View source | | | |
| LANGUAGE → | | Accept-Language: | en-GB,en-US;q=0.9,en;q=0.8 | | | |
| PAGE REQUEST IS FROM → | | Referer: | http://localhost:8888/phpbook/section_b/intro/index.php | | | |
| BROWSER → | | User-Agent: | Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/93.0.4577.63 Safari/537.36 | | | |

Peticiones y respuestas HTTP

Cuando el servidor web recibe una petición de una página PHP, **responde** a esa petición de la siguiente manera:

- Encontrando el archivo PHP solicitado en la URL
- Hacer que el intérprete PHP procese cualquier código PHP que contenga el archivo PHP
- Enviando una página HTML al navegador que solicitó la página.

Peticiones y respuestas HTTP

Cuando el servidor envía la página HTML de vuelta al navegador, también envía las **cabeceras de respuesta HTTP** al navegador, que contienen datos que el navegador puede necesitar saber sobre el archivo que está devolviendo.

Al igual que las cabeceras de solicitud (*request header*), cada cabecera de respuesta (*response header*) tiene un nombre y un valor (como una variable), que puede visualizarse en las herramientas de desarrollo del navegador.

Peticiones y respuestas HTTP

En la siguiente captura de pantalla, el servidor está enviando cabeceras de respuesta HTTP que indican al navegador el:

- Tipo de medio del archivo y el esquema de codificación utilizado (para ayudar a asegurar que muestra el archivo correctamente).
- Fecha y hora de envío del archivo
- Tipo de servidor web utilizado para enviar el archivo

Peticiones y respuestas HTTP

Las cabeceras de respuesta pueden actualizarse utilizando:

- La configuración del intérprete PHP
- Una función integrada llamada `header()`

Cuando un navegador recibe el archivo HTML, se muestra de la misma manera que mostraría cualquier página HTML.

Peticiones y respuestas HTTP



Peticiones y respuestas HTTP

El servidor también devuelve dos datos para indicar si la solicitud se ha realizado correctamente o no:

- Un **código de estado** de tres dígitos para que lo interprete el software
- Una **frase explicativa** que los usuarios puedan leer.

Si la solicitud se realiza correctamente, el código de estado es *200* y la frase explicativa es *OK*.

Si un servidor no puede encontrar un archivo, el código de estado es *404* y la frase explicativa es *Not Found*.

Peticiones y respuestas HTTP

Al navegar por Internet, es posible que haya visto una pantalla como ésta, que indica que no se ha encontrado una página.

Not Found

The requested URL `/code/section_b/c5/test.php` was not found on this server.

Peticiones y respuestas HTTP

La siguiente tabla muestra los códigos de estado y las frases de motivo más comunes. Códigos como *301* (movido permanentemente) y *404* (no encontrado) ayudan a los motores de búsqueda a indexar sitios cuando encuentran enlaces a páginas que han sido borradas o movidas a una nueva URL.

| STATUS CODE | REASON PHRASE |
|-------------|-----------------------|
| 200 | OK |
| 301 | Moved permanently |
| 307 | Temporary redirect |
| 403 | Forbidden |
| 404 | Not found |
| 500 | Internal server error |

Cómo se envían los datos utilizando HTTP GET y POST

HTTP especifica dos formas en que los navegadores pueden enviar datos al servidor:

- HTTP GET pone los datos en la cadena de consulta al final de la URL.
- HTTP POST añade los datos a las cabeceras HTTP.

Cómo se envían los datos utilizando HTTP GET y POST

Cuando se envían datos a una página web a través de **HTTP GET**, el navegador coloca los datos en una cadena de consulta y la añade al final de la URL de la página. Un signo de interrogación separa la URL de la página de la cadena de consulta.

Una cadena de consulta puede contener varios pares nombre/valor. Un signo igual separa cada nombre de su valor. Para enviar más de un par nombre/valor, se separa cada uno con un ampersand (&).

Cómo se envían los datos utilizando HTTP GET y POST

QUERY STRING

`http://eg.link/hotel.php?location=Paris&year=2021`

NAME VALUE NAME VALUE

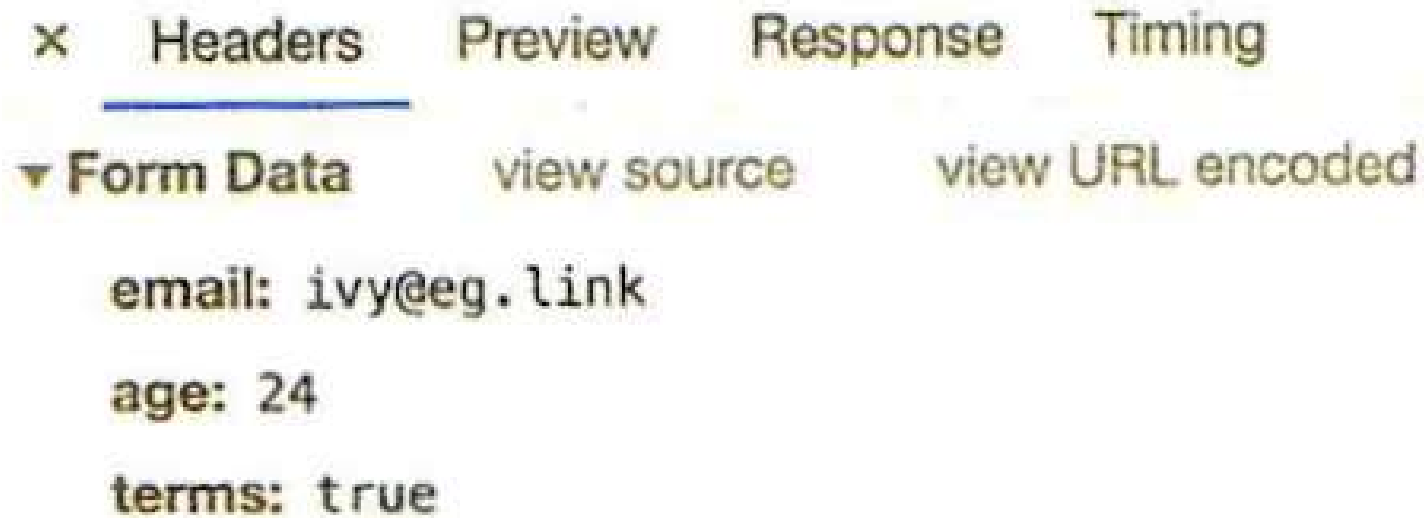
The diagram illustrates the structure of a query string in an HTTP GET request. The URL `http://eg.link/hotel.php?location=Paris&year=2021` is shown. A bracket above the query string part (`?location=Paris&year=2021`) is labeled "QUERY STRING". Below the query string, four segments are identified with brackets and labels: "location" is labeled "NAME", "Paris" is labeled "VALUE", "&year" is labeled "NAME", and "2021" is labeled "VALUE".

Cómo se envían los datos utilizando HTTP GET y POST

Cuando se envían datos a través de **HTTP POST**, el navegador añade pares nombre/valor adicionales a las cabeceras de la petición HTTP. El navegador puede enviar varios pares nombre/valor al servidor con cada solicitud.

Las cabeceras no se muestran en la ventana principal del navegador, pero puedes verlas en las herramientas para desarrolladores que vienen con la mayoría de los navegadores. A continuación, puedes ver tres cabeceras y sus valores correspondientes.

Cómo se envían los datos utilizando HTTP GET y POST



The screenshot shows the 'Form Data' tab of a web browser's developer tools. The tab is selected and underlined. Below the tab, the form data is displayed as key-value pairs: 'email: ivy@eg.link', 'age: 24', and 'terms: true'. The 'Form Data' tab is preceded by a dropdown arrow, and there are links for 'view source' and 'view URL encoded' to the right.

| x | Headers | Preview | Response | Timing |
|--------------------|-----------|-------------|------------------|--------|
| ▼ | Form Data | view source | view URL encoded | |
| email: ivy@eg.link | | | | |
| age: 24 | | | | |
| terms: true | | | | |

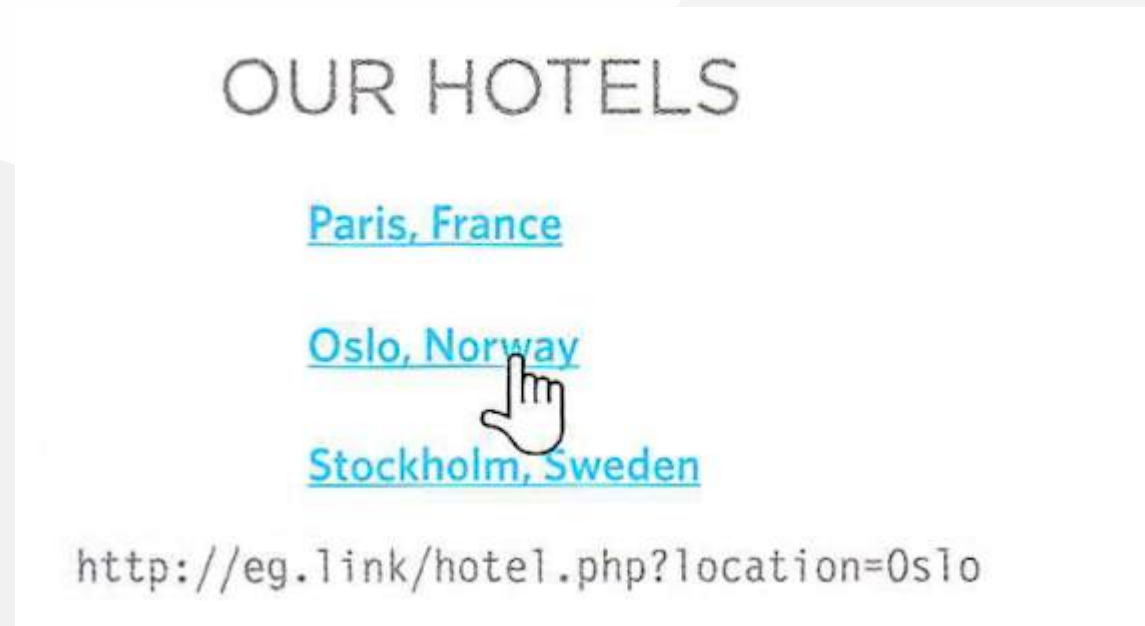
Cómo se envían datos desde enlaces y formularios

HTML utiliza **enlaces** y **formularios** para enviar datos adicionales al servidor al mismo tiempo que se solicita una página.

Cómo se envían datos desde enlaces y formularios

Un **enlace** puede utilizar una cadena de consulta para enviar datos adicionales al servidor.

Los datos de la cadena de consulta suelen indicar al servidor que obtenga información específica y la muestre en la página que devuelve.



Cómo se envían datos desde enlaces y formularios

Normalmente, HTTP GET se utiliza cuando un navegador quiere obtener información del servidor, y esa información sería la misma para todos los visitantes del sitio. Por ejemplo, cuando:

- Hacen clic en enlaces para mostrar información específica
- Introducen un término de búsqueda en un formulario

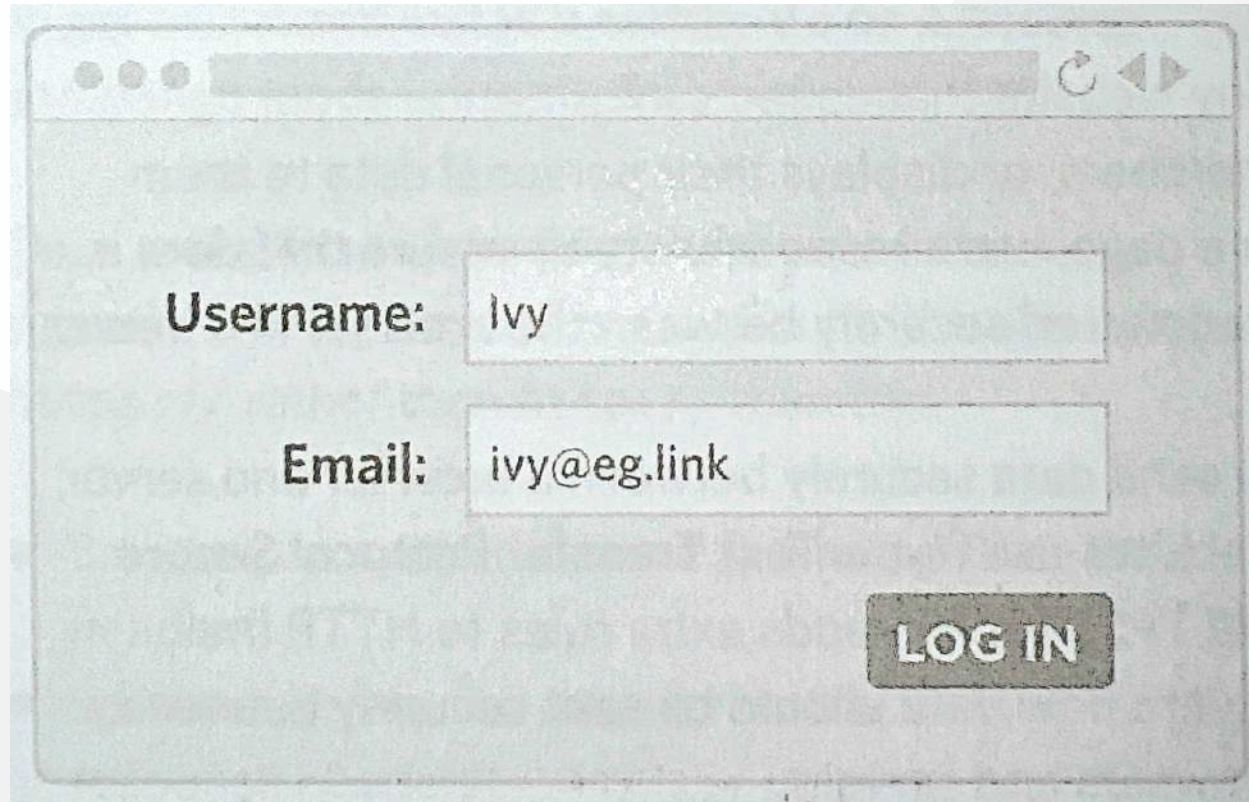
Los programadores a veces se refieren a este tipo de solicitud como una **interacción segura** porque el usuario no es responsable de la tarea que realiza (por ejemplo, no está aceptando los términos y condiciones o comprando un producto).

Cómo se envían datos desde enlaces y formularios

Los **formularios** tienen entradas que permiten a los usuarios introducir texto o números, seleccionar una de una lista de opciones o marcar una casilla.

Los datos del formulario pueden añadirse a la cadena de consulta o enviarse en las cabeceras HTTP.

Cómo se envían datos desde enlaces y formularios

A screenshot of a web browser window showing a login form. The form has two input fields: one for 'Username' with the value 'ivy' and one for 'Email' with the value 'ivy@eg.link'. Below the fields is a 'LOG IN' button. The browser window has a standard address bar and navigation buttons.

Username: ivy

Email: ivy@eg.link

LOG IN

Cómo se envían datos desde enlaces y formularios

Normalmente, HTTP POST se utiliza cuando un usuario está enviando (o publicando) información al servidor que le identifica o que se utiliza para actualizar los datos almacenados sobre él en el servidor. Por ejemplo, cuando:

- Acceden a su cuenta personal
- Compran un producto
- Se suscriben a un servicio
- Aceptan los términos y condiciones

En estas situaciones, el usuario puede ser responsable de sus acciones porque tiene que rellenar el formulario y luego enviarlo.

Protegiendo datos enviados hacia o desde un servidor

Cuando se envían datos confidenciales entre un navegador y un servidor, deben cifrarse.

- La **encriptación** codifica los datos para que no puedan ser leídos.
- El **descifrado** devuelve los datos a un formato que puede leerse.

Protegiendo datos enviados hacia o desde un servidor

Cuando los datos se envían a través de Internet, pueden viajar por distintas redes, pasando por muchos routers y servidores para llegar a su destino. Durante este viaje, personas no autorizadas podrían acceder a los datos e intentar leerlos.

Protegiendo datos enviados hacia o desde un servidor

Cualquier sitio web que recopile información sobre sus miembros, o les muestre sus datos personales en una página, tiene la responsabilidad de garantizar que los datos se transfieren de forma segura entre el navegador y el servidor.

Para enviar datos de forma segura entre el navegador y el servidor, los sitios web utilizan el **Protocolo Seguro de Transferencia de Hipertexto (HTTPS)**. HTTPS añade reglas adicionales a HTTP que rigen cómo deben enviarse los datos de forma segura entre navegadores y servidores.

Protegiendo datos enviados hacia o desde un servidor

Para enviar datos de forma segura a través de Internet, es necesario cifrarlos. Esto implica alterar los datos para que, incluso si fueran interceptados durante su viaje, la gente no pudiera leerlos.

Los mensajes se cifran sustituyendo los caracteres originales por un conjunto diferente de caracteres. Para ello se utiliza un conjunto de reglas conocido como **cifrado**.

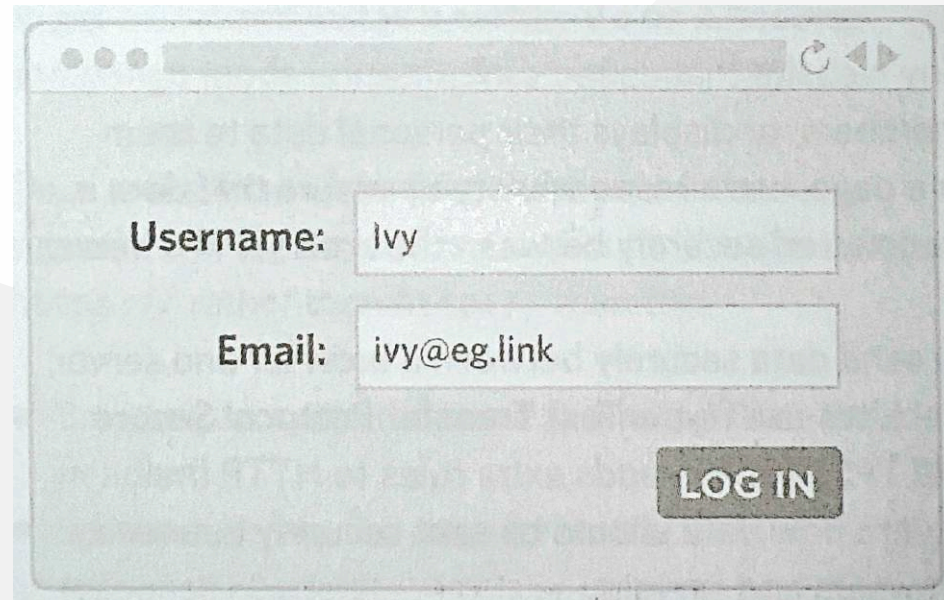
Protegiendo datos enviados hacia o desde un servidor

A continuación, el destinatario del mensaje debe descriptarlo para que vuelva a ser legible. Para descifrar el mensaje, el destinatario necesita saber cómo se cifró.

Los datos necesarios para descifrar el mensaje se conocen como **clave** porque "desbloquean" el mensaje.

Protegiendo datos enviados hacia o desde un servidor

1. Cuando el usuario envía un formulario, el navegador encripta los datos.



A screenshot of a web browser window displaying a login form. The form has two input fields: 'Username:' with the value 'Ivy' and 'Email:' with the value 'ivy@eg.link'. Below the fields is a 'LOG IN' button. The browser window has a standard address bar and navigation buttons.

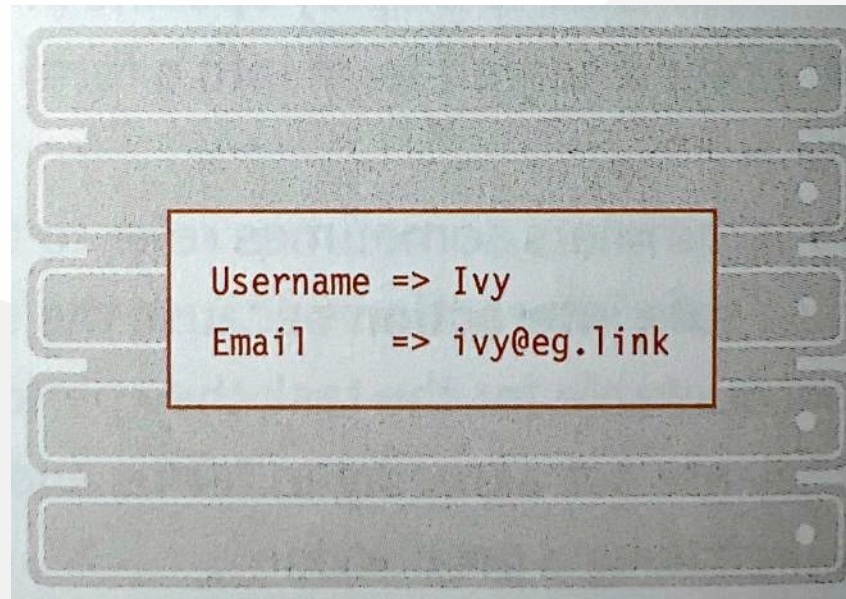
Protegiendo datos enviados hacia o desde un servidor

2. Durante el tránsito, los datos cifrados no pueden leerse.

```
NKFAyGCNYKdbNCDTA+XIwR698oP  
pAdN1ghyUmRPtkE8y2evzf8LEMe  
r0Q89N6XJN2AFt919bAr+qk/qSv  
C6b/dRAbb6NqIYXqc6s0IZta/VZ  
1UwJTUJH0Io6Qj68+paMgZX/6wX  
X0f2VWLxxBM7XwU7ufVZ53VLQA+  
mz/wA4jbAFevz8y2f8dbNCBW2wA
```

Protegiendo datos enviados hacia o desde un servidor

3. El servidor utiliza una clave para descifrar los datos.



Protegiendo datos enviados hacia o desde un servidor

Para cifrar y descifrar los datos enviados entre navegadores y servidores que utilizan HTTPS, hay que instalar un **certificado** en el servidor web.

Este indica al navegador cómo cifrar la información que envía a ese servidor.

Protegiendo datos enviados hacia o desde un servidor

Para obtener un certificado para instalar en un servidor web, debes seguir estos tres pasos:

1. Crear una **solicitud de firma de certificado** (**certificate signing request** o **CSR**).
Estos son generados por el servidor web en el que vive el sitio. Parecen una serie de caracteres aleatorios.

Protegiendo datos enviados hacia o desde un servidor

2. Adquirir un certificado de una empresa llamada **autoridad de certificación** (**certificate authority** o **CA**). Nos pedirán una solicitud de firma de certificado e información sobre el sitio web y su propietario. Las CA, habitualmente, cobran una tarifa anual por el certificado.
 - [Let's Encrypt](#) que ofrece hasta el día de hoy certificados gratuitos.
 - [DigiCert](#) (que también posee [GeoTrust](#), [Thawte](#) and [RapidSSL](#))
 - [Globalsign](#)
 - [Sectigo](#)
 - [Entrust](#)

Protegiendo datos enviados hacia o desde un servidor

3. Instalar el certificado (que es un archivo de texto) en el servidor en el que se ejecuta el sitio.

NOTA: Los certificados no siempre se emiten inmediatamente, por lo que deben obtenerse antes de que el sitio vaya a entrar en funcionamiento.

Protegiendo datos enviados hacia o desde un servidor

Cuando desarrolles un sitio localmente utilizando MAMP o XAMPP, puedes configurar el servidor web para que funcione utilizando HTTPS sin necesidad de adquirir un certificado. Puedes ver instrucciones sobre cómo hacerlo en el [Anexo I](#).

Para obtener una CSR e instalar un certificado en los servidores de una empresa de alojamiento (hosting), deberemos consultar sus archivos de soporte.

Una vez que se ha instalado un certificado en el servidor web, si un navegador solicita una URL para el sitio utilizando `https://` en lugar de `http://` entonces el:

- El navegador encripta la petición y las cabeceras de petición HTTP.
- El servidor encripta la página que devuelve y las cabeceras de respuesta HTTP.

Cuando se utiliza `https://`, los navegadores suelen mostrar un icono de candado en la barra de direcciones.

Protegiendo datos enviados hacia o desde un servidor

Históricamente, HTTPS utiliza dos protocolos diferentes (conjunto de reglas) para añadir cifrado a las peticiones y respuestas enviadas mediante HTTP:

- **Secure Socket Layer (SSL)**
- **Transport Layer Security (TLS)**

A menudo se utilizan los términos SSL y TLS indistintamente, pero técnicamente son diferentes.

TLS es como una versión más reciente de SSL, por tanto TLS es el protocolo que deberías utilizar en tu sitio web.



Esquemas de codificación

Esquemas de codificación

Los ordenadores representan texto, imágenes y audio utilizando **datos binarios**, que se componen de una serie de Os y 1s.

Los **esquemas de codificación** traducen lo que ves o escuchas en Os y 1s que el ordenador procesa.

Esquemas de codificación

Comprender la función de los esquemas de codificación es importante porque, si un ordenador utiliza un esquema de codificación incorrecto para traducir entre los datos binarios y el texto, las imágenes y el audio que ves y oyes, los datos no se mostrarán/reproducirán correctamente.

Esquemas de codificación

Todos los datos que procesa y almacena un ordenador se representan mediante **bits** (**dígitos binarios**). Un bit es un 0 o un 1. Por lo tanto, todo en tu ordenador (las letras que tecleas, las imágenes que ves, el audio que oyes) se representa en Os y 1s.

A continuación, puedes ver el equivalente binario de cada letra de la palabra HELLO:

| | | | | |
|----------|----------|----------|----------|----------|
| H | E | L | L | O |
| 01001000 | 01000101 | 00101100 | 00101100 | 01001111 |

Como muestra, incluso los datos más sencillos requieren muchos bits. Una serie de 8 bits se denomina **byte**.

Esquemas de codificación

Los esquemas de codificación son reglas que utiliza un ordenador para traducir entre texto, imágenes y audio y los datos binarios (combinaciones de 0s y 1s) que el ordenador procesa y almacena.

- Al escribir texto, cargar imágenes o grabar audio, se utiliza un esquema de codificación para traducir ese contenido en 0s y 1s.
- Cuando el ordenador te muestra texto e imágenes o reproduce un archivo de audio, utiliza un esquema de codificación para traducir los 0s y 1s en cosas que puedas ver u oír.

Esquemas de codificación

Los **esquemas de codificación de imágenes** especifican cómo representar imágenes utilizando bits. Las imágenes de ordenador están formadas por cuadrados llamados píxeles.

A continuación, puedes ver cómo un icono básico de un corazón en blanco y negro puede representarse utilizando 0s y 1s. Cada cuadrado blanco está representado por un 0 y cada cuadrado negro está representado por un 1.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Esquemas de codificación

Para recrear imágenes en color, el ordenador debe saber de qué color es cada píxel, lo que requiere más datos. Los distintos formatos de imagen (como GIF, JPEG, PNG y WebP) utilizan diferentes esquemas de codificación para representar el color de cada píxel mediante Os y 1s.

Para manipular la imagen, un ordenador puede cambiar los datos que almacena para cada píxel. Por ejemplo, se puede utilizar un filtro para oscurecer o aclarar cada píxel de la imagen, o se puede recortar una imagen eliminando píxeles de los bordes.

Esquemas de codificación

Los **esquemas de codificación de caracteres** especifican cómo representar texto utilizando bits, y algunos esquemas de codificación de caracteres admiten más caracteres que otros. Cuantos más caracteres admita un sistema de codificación, más bytes de datos necesitará para manejarlos.

Cuando crees un sitio web, para poder atender a un público internacional, debes utilizar un esquema de codificación de caracteres que incluya los caracteres utilizados por los idiomas de quienes visiten tu sitio.

Esquemas de codificación

ASCII

ASCII es uno de los primeros sistemas de codificación de caracteres que utiliza 7 bits de datos para representar cada carácter.

Un inconveniente de ASCII es que sólo hay 128 combinaciones posibles de 0 y 1 utilizando 7 dígitos binarios, por lo que no hay combinaciones suficientes para admitir todos los caracteres que se utilizan en todos los idiomas. De hecho, ASCII sólo admite 95 caracteres de texto.

Esquemas de codificación

ISO 8859-1

ISO 8859-1 utiliza 8 bits (1 byte) de datos para representar cada carácter. El bit extra de datos significa que hay suficientes combinaciones de 0 y 1 para representar los mismos caracteres que ASCII, así como los caracteres acentuados que utilizan las lenguas de Europa Occidental. Pero no admite idiomas que utilizan juegos de caracteres diferentes, como el chino, el japonés o el ruso.

Esquemas de codificación

UTF-8

UTF-8 representa todos los caracteres de todos los idiomas, por lo que es el mejor esquema de codificación de caracteres que se puede utilizar al crear sitios web.

Para ser compatible con estos idiomas, UTF-8 necesita hasta cuatro bytes de datos para representar cada carácter (cuatro conjuntos de 8 combinaciones de 0 y 1). Los caracteres representados utilizando más de un byte de datos se conocen como caracteres multibyte. Por ejemplo, a continuación puedes ver los equivalentes binarios de tres símbolos monetarios diferentes:

Esquemas de codificación

UTF-8

| SYMBOL | BINARY | BYTES |
|--------|----------------------------|-------|
| \$ | 00100100 | 1 |
| £ | 11000010 10100011 | 2 |
| € | 11100010 10000010 10101100 | 3 |

Esquemas de codificación

Es importante entender cómo funcionan los esquemas de codificación de caracteres porque:

- Deben especificarse en varios lugares
- Afectan a los caracteres que se pueden utilizar.
- Determinan qué funciones integradas pueden utilizarse.

Esquemas de codificación

Funciones integradas

Algunas de las funciones integradas del intérprete de PHP tienen un parámetro que se utiliza para especificar el esquema de codificación de caracteres que se utiliza.

También hay algunas funciones incorporadas que trabajan con un esquema de codificación de caracteres específico.

Esquemas de codificación

Funciones integradas

Por ejemplo, PHP tiene una función que cuenta el número de caracteres en una cadena. Esta función fue añadida a PHP cuando el esquema de codificación de caracteres por defecto era ISO 8859-1, y la función trabajaba contando el número de bytes que utiliza una cadena (porque cada carácter utiliza 1 byte de datos).

Cuando PHP comenzó a soportar UTF-8, esta función dio resultados inexactos debido a que cada carácter podía utilizar más de un byte. Por lo tanto, se agregó una nueva función incorporada que podía contar caracteres multibyte.

Esquemas de codificación

Configuración de PHP

Cuando el intérprete PHP crea una página para enviarla de vuelta al navegador, le indica al navegador el esquema de codificación de caracteres que ha utilizado para que pueda mostrar los datos correctamente.

El intérprete PHP tiene configuraciones que especifican el esquema de codificación utilizado para crear las páginas HTML que envía de vuelta y, si el esquema de codificación no está configurado correctamente, el navegador puede mostrar un símbolo de interrogación (?) para caracteres que no entiende (o puede que no los muestre en absoluto).

Esquemas de codificación

Editores de código

Debido a que los archivos PHP en sí son texto, los editores de código usualmente te permiten especificar el esquema de codificación de caracteres que debe utilizar para guardar los archivos PHP. En el [anexo II](#) se muestra cómo establecer el esquema de codificación de caracteres en algunos editores de código populares.

En el bloque C del módulo, también verás que las bases de datos necesitan conocer el esquema de codificación que utiliza un sitio web.



Herramientas integradas en el intérprete PHP

Herramientas integradas en el intérprete PHP

En este apartado, conocerás herramientas que están integradas en el intérprete PHP para ayudarte a crear páginas web dinámicas.

Herramientas integradas en el intérprete PHP

Hemos visto que el intérprete PHP es una pieza de software que se ejecuta en un servidor web.

Cuando abres un software en un ordenador de sobremesa o portátil (como un procesador de textos o un software de edición de imágenes), hay una interfaz gráfica de usuario (GUI). Las barras de herramientas y las opciones de menú de la GUI se utilizan para realizar las tareas para las que fue diseñado el software, y los resultados se muestran en pantalla.

Herramientas integradas en el intérprete PHP

El intérprete PHP no tiene una GUI. En su lugar, el intérprete PHP tiene un conjunto de arrays, funciones y clases incorporadas que el código PHP en tus archivos PHP puede utilizar para realizar tareas que comúnmente se necesitan cuando se procesan datos y se crean páginas HTML para enviar al navegador.

El intérprete PHP también utiliza archivos de texto para permitir el control de opciones y preferencias, y registrar cualquier error que pueda ocurrir.

Herramientas integradas en el intérprete PHP

Arrays superglobales

Cada vez que un navegador solicita una página PHP, el intérprete PHP crea un conjunto de arrays llamados **arrays superglobales** que contienen datos que pueden ser accedidos y utilizados por el código PHP en esa página.

Los arrays superglobales son todos arrays asociativos, por lo que necesita saber:

- Cómo se llaman los arrays
- Las claves en cada uno de los arrays
- Qué almacena cada una de esas claves

Herramientas integradas en el intérprete PHP

Arrays superglobales

Una vez que el intérprete de PHP ha creado la página HTML y la ha enviado de vuelta al navegador, los datos en estos arrays se olvidan porque los datos sólo se aplican a esa petición individual de la página.

La próxima vez que el archivo se ejecute, podrá acceder a un conjunto de arrays superglobales con datos relacionados con esa petición específica.

Herramientas integradas en el intérprete PHP

Funciones integradas

Las **funciones integradas** pueden compararse con los comandos que se encuentran en los menús de los programas que tienen una GUI. Por ejemplo, una función busca caracteres en una cadena y los reemplaza por otros, como la función de buscar y reemplazar de un procesador de textos. En lugar de utilizar un elemento de menú en la GUI, se llama a la función en el código PHP.

Herramientas integradas en el intérprete PHP

Funciones integradas

En la unidad 3, vimos como crear una definición de función y llamar a la función. Se llama a las funciones integradas de la misma manera pero no se incluye una definición de función en la página porque está incorporada en el intérprete PHP.

Para utilizar las funciones integradas, necesitas conocer el:

- Nombre de la función
- Parámetro(s) que requiere
- Valor que devolverá (o lo que muestra en la página)

Herramientas integradas en el intérprete PHP

Clases integradas

Las **clases integradas** se utilizan para crear objetos que representan cosas con las que los programadores tienen que tratar a menudo. Por ejemplo, la clase `DateTime` se utiliza para crear objetos que representan fechas y horas. Dicha clase tiene propiedades y métodos que permiten trabajar con las fechas y horas que representa un objeto creado con ella.

Herramientas integradas en el intérprete PHP

Clases integradas

En la unidad 4, vimos cómo escribir definiciones de clase y utilizarlas para crear objetos. No se incluye una definición de clase en una página cuando se crea un objeto utilizando una clase integrada, porque está incorporada en el intérprete PHP.

Para utilizar las clases integradas, es necesario saber como crear un objeto utilizando esa clase, y que:

- Propiedades tiene la clase
- Métodos tiene la clase
- Parámetros tiene cada método
- Valor devuelve cada método

Herramientas integradas en el intérprete PHP

Mensajes de error

Si el intérprete PHP encuentra errores en el código que está intentando ejecutar, generará un mensaje de error.

Cuando se está desarrollando un sitio, los mensajes de error deben mostrarse en la página que el intérprete de PHP devuelve al navegador. Esto permite a los desarrolladores ver inmediatamente cualquier error que se produzca al intentar ejecutar la página.

Herramientas integradas en el intérprete PHP

Mensajes de error

Cuando un sitio se pone en marcha, los errores deben ocultarse a los visitantes. Como los desarrolladores no pueden ver los errores en tiempo real, los mensajes de error se guardan en un archivo de texto llamado **archivo de registro (log file)** que se almacena en el servidor. Los desarrolladores comprueban entonces los archivos de registro para ver si se ha producido algún error que se haya pasado por alto durante el desarrollo del sitio.

Herramientas integradas en el intérprete PHP

Ajustes

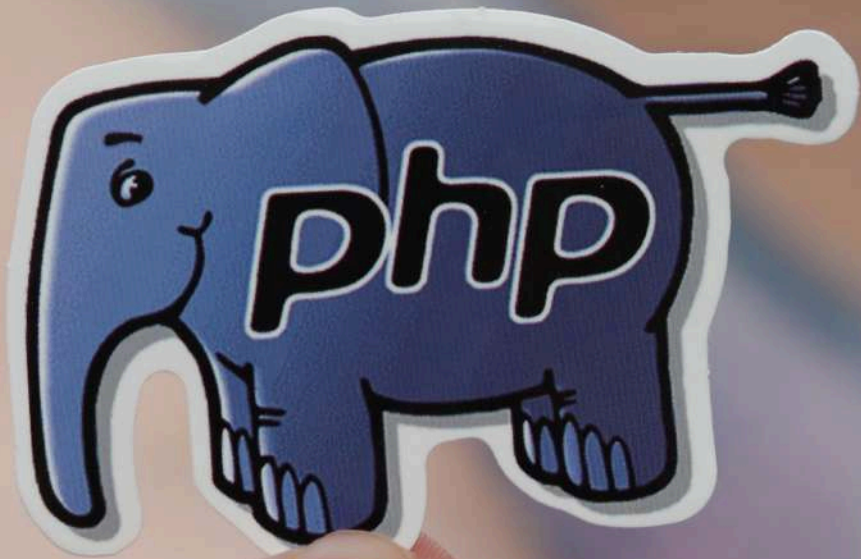
El software de escritorio suele tener opciones de menú que abren ventanas que permiten a los usuarios controlar las preferencias u opciones sobre el funcionamiento del software. Por ejemplo, la configuración de un procesador de textos puede permitir al usuario seleccionar el tamaño del papel o el idioma por defecto en el que está redactado un documento.

Herramientas integradas en el intérprete PHP

Ajustes

Debido a que el intérprete PHP y el servidor web no tienen una GUI, utilizan archivos de texto para controlar una serie de ajustes. Por ejemplo, hay configuraciones que controlan si el intérprete PHP debe mostrar los mensajes de error en la pantalla o guardarlos en un archivo de registro, y dónde debe vivir el archivo de registro de errores en el servidor.

Estos archivos de texto pueden ser editados utilizando el mismo editor de código que utilizamos para crear las páginas PHP.



Arrays superglobales

Arrays superglobales

El array superglobal `$_SERVER` es un ejemplo de uno de los arrays superglobales que el intérprete PHP crea cada vez que se solicita una página. Cada array superglobal tiene datos que el código PHP de la página puede utilizar.

Todos los arrays superglobales son arrays asociativos. Necesitas saber el nombre del array, las claves que tiene cada array y los datos que contienen.

A continuación, puedes ver lo que almacena el array superglobal `$_SERVER`. Contiene datos sobre el:

- Navegador (que se envía en las cabeceras HTTP)
- Tipo de petición HTTP (GET o POST)
- URL solicitada
- Ubicación del archivo en el servidor

Arrays superglobales

| KEY | PURPOSE |
|---|---|
| <code>\$_SERVER['REMOTE_ADDR']</code> | Browser's IP address |
| <code>\$_SERVER['HTTP_USER_AGENT']</code> | Type of browser used to request the page |
| <code>\$_SERVER['HTTP_REFERER']</code> | If the visitor came to this page via a link, a browser can send the URL of the page that linked to this one. Not all browsers send this data. |
| <code>\$_SERVER['REQUEST_METHOD']</code> | The type of HTTP request: GET or POST |
| <code>\$_SERVER['HTTPS']</code> | Only added to array if page accessed using HTTPS; if so, its value is true |
| <code>\$_SERVER['HTTP_HOST']</code> | The host name (could be a domain name, IP address, or localhost) |
| <code>\$_SERVER['REQUEST_URI']</code> | The URI (<i>after</i> the host name) used to request this page |
| <code>\$_SERVER['QUERY_STRING']</code> | Any data in the query string |
| <code>\$_SERVER['SCRIPT_NAME']</code> | Path from the document root folder to the file currently being executed |
| <code>\$_SERVER['SCRIPT_FILENAME']</code> | Path from the filesystem root to the file currently being executed |
| <code>\$_SERVER['DOCUMENT_ROOT']</code> | Path from the filesystem root to document root of the file being executed |

Arrays superglobales

Los datos que almacenan los arrays superglobales se recuperan de la misma forma que se accede a cualquier array asociativo.

Para almacenar la dirección IP del navegador del usuario en una variable llamada `$ip` se utilizaría lo siguiente:



```
$ip = $_SERVER['REMOTE_ADDR'];
```

The diagram illustrates the structure of the `$_SERVER` superglobal array. A bracket above the entire expression `$_SERVER['REMOTE_ADDR'];` is labeled "SUPERGLOBAL ARRAY". Another bracket below the string `'REMOTE_ADDR'` is labeled "KEY".

Datos en el array superglobal `$_SERVER`

Cada elemento del array superglobal `$_SERVER` almacena una información diferente sobre la petición o el fichero que se ha solicitado.

En el siguiente ejemplo, se escriben los valores individuales almacenados en el array superglobal `$_SERVER` seleccionando los datos en las diferentes claves del array.

Datos en el array superglobal \$_SERVER

```
<table>
  <tr><th colspan="2" class="title">Data About Browser Sent in HTTP Headers  </th></tr>
  <tr><th>Browser's IP address  </th><td><?= $_SERVER['REMOTE_ADDR'] ?>  </td></tr>
  <tr><th>Type of browser  </th><td><?= $_SERVER['HTTP_USER_AGENT'] ?></td></tr>
  <tr><th colspan="2" class="title">HTTP Request  </th></tr>
  <tr><th>Host name  </th><td><?= $_SERVER['HTTP_HOST'] ?>  </td></tr>
  <tr><th>URI after host name  </th><td><?= $_SERVER['REQUEST_URI'] ?>  </td></tr>
  <tr><th>Query string  </th><td><?= $_SERVER['QUERY_STRING'] ?>  </td></tr>
  <tr><th>HTTP request method  </th><td><?= $_SERVER['REQUEST_METHOD'] ?>  </td></tr>
  <tr><th colspan="2" class="title">Location of the File Being Executed  </th></tr>
  <tr><th>Document root  </th><td><?= $_SERVER['DOCUMENT_ROOT'] ?>  </td></tr>
  <tr><th>Path from document root  </th><td><?= $_SERVER['SCRIPT_NAME'] ?>  </td></tr>
  <tr><th>Absolute path  </th><td><?= $_SERVER['SCRIPT_FILENAME'] ?></td></tr>
</table>
```

Datos en el array superglobal \$_SERVER

DATA ABOUT BROWSER SENT IN HTTP HEADERS

BROWSER'S IP ADDRESS ::1
TYPE OF BROWSER Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/17.5 Safari/605.1.15

HTTP REQUEST

HOST NAME localhost
URI AFTER HOST NAME /phpbook/section_b/intro/server-superglobal.php

QUERY STRING

HTTP REQUEST METHOD GET

LOCATION OF THE FILE BEING EXECUTED

DOCUMENT ROOT /Applications/XAMPP/xamppfiles/htdocs
PATH FROM DOCUMENT ROOT /phpbook/section_b/intro/server-superglobal.php
ABSOLUTE PATH /Applications/XAMPP/xamppfiles/htdocs/phpbook/section_b/intro/server-superglobal.php

Mostrando datos con funciones integradas

La función `var_dump()` es un ejemplo de una de las funciones incorporadas de PHP. Se utiliza cuando se desarrollan sitios para comprobar qué valor(es) contiene una variable y qué tipo de datos son.

Mostrando datos con funciones integradas

Para utilizar (o llamar) a una función incorporada, sólo necesita conocer su nombre, sus parámetros y qué valores devolverá o mostrará en la página.

`var_dump()` tiene un parámetro: el nombre de una variable. No devuelve un valor; muestra los valores almacenados en la variable en la página HTML que se crea.

```
var_dump($variable);
```


Mostrando datos con funciones integradas

Si la variable contiene un valor **escalar** (una cadena, un número o un entero), muestra el tipo de datos y el valor.

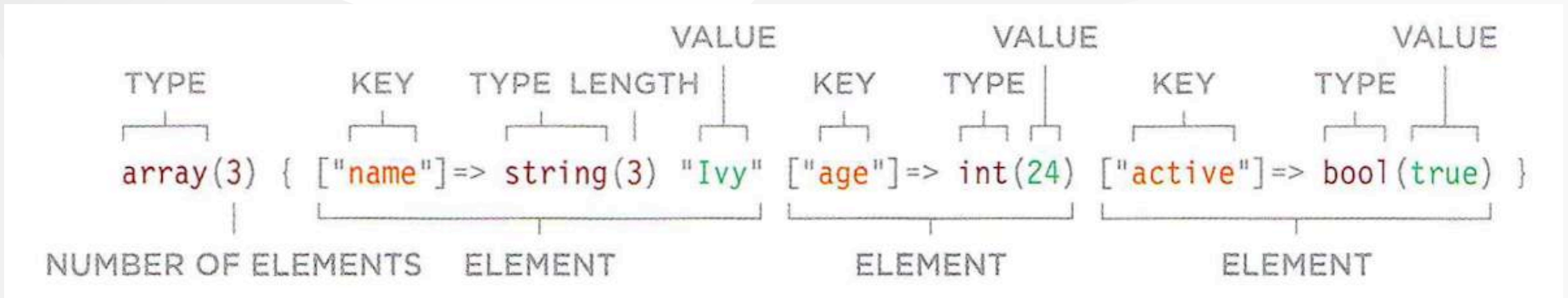
Si el valor es una **cadena**, el número de caracteres de la cadena se muestra entre paréntesis después del tipo de datos.

| DATA TYPE | LENGTH | VALUE |
|-----------|--------|-------|
| string(3) | | "Ivy" |

Mostrando datos con funciones integradas

Si la variable contiene un **array**, muestra la palabra array y el número de elementos del array entre paréntesis.

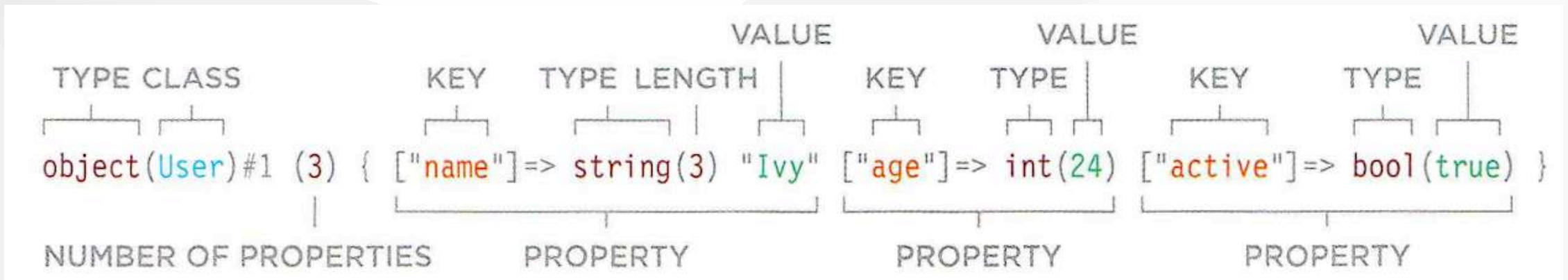
A continuación, en un par de llaves, muestra la clave, el tipo de datos del valor y el valor de cada elemento.



Mostrando datos con funciones integradas

Si la variable contiene un **objeto**, muestra la palabra objeto, el nombre de la clase y el número de propiedades.

Para cada propiedad, muestra el nombre, el tipo de datos del valor y el valor. (Los métodos no se muestran).



Ejemplo: mostrando el contenido de una variable

Este ejemplo crea variables para almacenar un valor escalar, un array y un objeto, y utiliza `var_dump()` para mostrar lo que contiene cada variable.

1. Una variable llamada `$username` almacena el nombre de un miembro de un sitio web como una cadena.
2. En una variable llamada `$user_array` se almacena un array. El array almacena el nombre del miembro, su edad y si está activo.

Ejemplo: mostrando el contenido de una variable

3. Se crea una clase llamada `User` para que actúe como plantilla de los objetos que representan a los miembros de un sitio. Tiene tres propiedades: el nombre del usuario, su edad y si está activo o no.
4. Se crea un objeto utilizando la clase `User` y se almacena en una variable llamada `$user_object`.
5. Los valores almacenados en las variables se muestran utilizando la función `var_dump()`.

Ejemplo: mostrando el contenido de una variable

```
<?php
① $username    = 'Ivy';

[
  $user_array = [
    'name' => 'Ivy',
    ② 'age'   => 24,
    'active' => true,
  ];

  class User
  {
    public $name;
    public $age;
    public $active;
    ③ public function __construct($name, $age, $active) {
      $this->name    = $name;
      $this->age     = $age;
      $this->active  = $active;
    }
  }

  ④ $user_object = new User('Ivy', 24, true);
  ?>

...
⑤ [
  <p>Scalar:    <?php var_dump($username); ?></p>
  <p>Array:     <?php var_dump($user_array); ?></p>
  <p>Object:    <?php var_dump($user_object); ?></p>
]
```

Ejemplo: mostrando el contenido de una variable

```
Scalar: string(3) "Ivy"
```

```
Array: array(3) { ["name"]=> string(3) "Ivy" ["age"]=> int(24) ["active"]=> bool(true) }
```

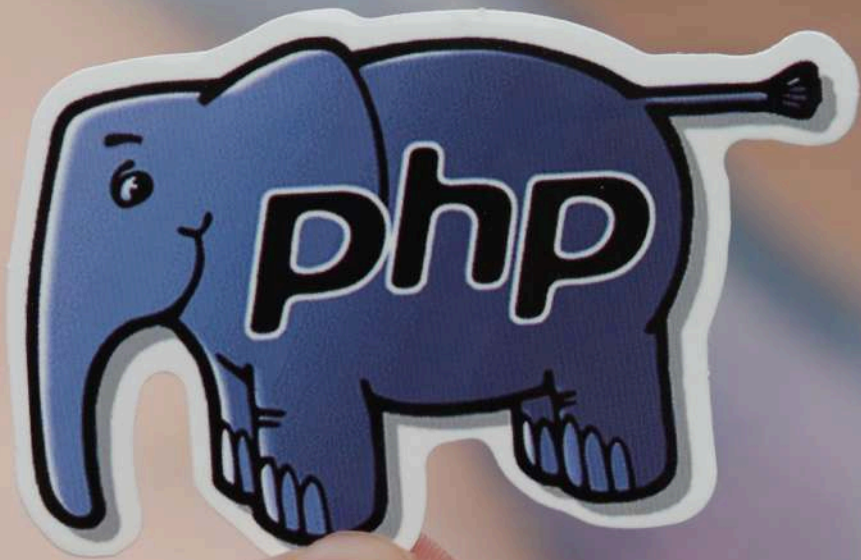
```
Object: object(User)#1 (3) { ["name"]=> string(3) "Ivy" ["age"]=> int(24) ["active"]=> bool(true) }
```

NOTA: Si añades etiquetas HTML `<pre>` alrededor de cada bloque PHP, los datos que se muestren se distribuirán en líneas separadas, lo que facilitará su lectura.

```
<pre>
<?php var_dump($username) ?>
</pre>
```


Ejemplo: mostrando el contenido de una variable

Adapta el ejemplo anterior utilizando las etiquetas `<pre>` para imprimir de forma mas visual la información de las variables.



Mensajes de error

Mensajes de error

Cuando hay un problema con el código PHP, el intérprete PHP genera mensajes de error que Si el intérprete PHP encuentra un problema con el código que está ejecutando, genera un mensaje de error. Hay dos maneras de ver los mensajes; pueden ser:

- Mostrados en el HTML que se envía al navegador
- Guardados en un archivo de texto conocido como **registro de errores (error log)**.

Mensajes de error

Cada mensaje de error contiene cuatro datos que nos ayudarán a localizar el problema para poder solucionarlo:

- El nivel de error (o gravedad del error; los niveles se describen en la tabla siguiente).
- Una descripción del error
- El archivo que contiene el error
- El número de línea donde se encontró el error.

| ERROR LEVEL | ERROR DESCRIPTION | PHP FILE | LINE NUMBER |
|-------------|-----------------------|-------------|-------------|
| Error: | description goes here | in test.php | on line 21 |

Mensajes de error

| LEVEL | DESCRIPTION |
|------------|--|
| PARSE | Errors in the syntax of the PHP code prevent the PHP interpreter from trying to run the page at all |
| FATAL | An error in the PHP code that stops any further code (after the error) from running |
| WARNING | Something that will <i>probably</i> cause a problem, but the interpreter tries to run the rest of the page |
| NOTICE | Something that <i>could</i> indicate an problem, but the interpreter tries to run the rest of the page |
| DEPRECATED | PHP code that is likely to be removed from future versions of PHP |
| STRICT | PHP code that could be written in a better way and will be more future-proof |

Mensajes de error

Cuando se desarrolla un sitio, los mensajes de error deben mostrarse en las páginas HTML que el intérprete de PHP devuelve al navegador para que el desarrollador pueda verlos inmediatamente.

Cuando un sitio está activo, deben guardarse en un registro de errores (un archivo de texto en el servidor); no deben mostrarse a los visitantes. Aprenderás cómo cambiar esta configuración más adelante.

Mensajes de error

El código de este bloque muestra los errores en la página HTML. Cuando intentes hacer los ejercicios, no te desanimes si ves mensajes de error.

Los mensajes te ayudarán a descubrir dónde hay problemas con tu código y cómo solucionarlos. En la unidad 10 profundizaremos más sobre el manejo de errores.

Ejemplos de mensajes de error

Los mensajes de error pueden parecer crípticos al principio, pero la información que contienen te ayuda a averiguar qué falla en tu código.

1. La página intenta escribir una variable que no ha sido creada. Una advertencia dice que hay una variable *Undefined* `$name` en `error1.php` en la línea 2. Debido a que el nivel de error es una *advertencia*, el intérprete continúa ejecutándose.

Hasta PHP 7.4, este error creaba un aviso, no una advertencia.

2. La página escribe el texto "welcome to our site" (que puedes ver en la última línea, después del mensaje de error).

Ejemplos de mensajes de error

```
<?php  
① echo $name;  
② echo ' welcome to our site.';  
?>
```

Warning: Undefined variable \$name in
/Users/Jon/Sites/localhost/phpbook/section_b/intro/error1.php on line 2
welcome to our site.

Ejemplos de mensajes de error

3. La sentencia echo escribiría la palabra Hello pero la siguiente línea crea un error de interpretación que impide que el intérprete PHP ejecute el código de la página.

El error es causado por la falta del símbolo `$` al comienzo de la variable `username` .

Ejemplos de mensajes de error

```
<?php
③ { echo 'Hello ';
    username = 'Ivy';
?>
```

Parse error: syntax error, unexpected token "=" in
/Users/Jon/Sites/localhost/phpbook/section_b/intro/error2.php on line 3



Ajustes y opciones para el interprete PHP

Ajustes y opciones para el interprete PHP

Los ajustes y preferencias del software de un ordenador de sobremesa suelen controlarse a través de un menú en la interfaz de usuario. La configuración del intérprete PHP y del servidor web Apache se controla mediante archivos de texto.

El servidor web Apache y el intérprete PHP tienen configuraciones que controlan cosas como la codificación de caracteres por defecto utilizada, si mostrar a los usuarios mensajes de error si hay un problema, y cuánta memoria puede consumir una página web individual.

Los archivos que se utilizan para controlar estos ajustes se pueden editar en nuestro editor de código.

Ajustes y opciones para el interprete PHP

php.ini

Un archivo de texto llamado `php.ini` controla la configuración por defecto del intérprete PHP. Los ajustes en este archivo pueden ser cambiados, pero no deben ser eliminados. Una vez realizados los cambios, el servidor web debe reiniciarse para que los cambios surtan efecto.

Para encontrar la ubicación de nuestro archivo `php.ini`, PHP tiene una función incorporada llamada `phpinfo()`. Esta función muestra la configuración del intérprete PHP en una tabla HTML (ver imagen a continuación). La ruta al archivo `php.ini` está en la primera tabla junto al título **Loaded configuration file** (Archivo de configuración cargado).

Ajustes y opciones para el interprete PHP

php.ini

Algunas compañías de alojamiento (*hostings*) no nos proporcionan acceso al archivo `php.ini` porque normalmente controla cómo se ejecutan todos los archivos PHP en ese servidor web (y por lo tanto afectaría a otros sitios en el mismo servidor). Si no se tiene acceso a `php.ini`, se puede utilizar `.htaccess` para controlar muchos de los mismos ajustes.

Si un archivo `.php` necesita una configuración diferente a la de los otros archivos PHP en el servidor web, se puede utilizar una función incorporada en PHP llamada `ini_set()` para anular (o modificar en tiempo de ejecución) algunas de las configuraciones del archivo `php.ini`.

Ajustes y opciones para el interprete PHP

httpd.conf

Un archivo de texto llamado httpd.conf controla la configuración por defecto del servidor web Apache. Varios parámetros se solapan con los de php.ini, y Apache debe reiniciarse para que los cambios surtan efecto. Los proveedores de alojamiento no siempre dan acceso a httpd.conf a sus clientes, ya que normalmente controla la configuración de todo el servidor web.

Ajustes y opciones para el interprete PHP

.htaccess

Apache permite a los usuarios añadir archivos llamados `.htaccess` a cualquier carpeta de la raíz de documentos (*document root*) de un sitio. Las reglas de un archivo `.htaccess` sólo se aplican a los archivos del mismo directorio que el archivo `.htaccess` y a cualquiera de sus carpetas hijas, y anulan la configuración de `httpd.conf` y `php.ini`.

Los cambios en los archivos `.htaccess` surten efecto en cuanto se guarda el archivo. Pero sólo debe utilizar `.htaccess` si no puede utilizar `httpd.conf` o `php.ini` para realizar la tarea, ya que es más lento que cambiar la configuración predeterminada.

Ajustes y opciones para el interprete PHP

.htaccess

La mayoría de las empresas de hosting nos permiten crear archivos `.htaccess`, pero pueden restringir la configuración que puede utilizar (como el tamaño máximo de los archivos subidos).

Los sistemas operativos tratan los archivos `.htaccess` como archivos ocultos, por lo que es posible que tengamos que indicar a nuestro explorador de archivos o programa FTP que los muestre.

Cómo ver la configuración del intérprete PHP

Como la mayoría del software, el intérprete PHP tiene configuraciones (o preferencias) que controlan su funcionamiento.

La función incorporada `phpinfo()` muestra tablas que indican los ajustes que se pueden configurar y sus valores actuales.

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


Cómo ver la configuración del intérprete PHP

| Core | | |
|----------------------|-----------------|-----------------|
| PHP Version | 8.2.4 | |
| Directive | Local Value | Master Value |
| allow_url_fopen | On | On |
| allow_url_include | Off | Off |
| arg_separator.input | & | & |
| arg_separator.output | & | & |
| auto_append_file | <i>no value</i> | <i>no value</i> |
| auto_globals_jit | On | On |
| auto_prepend_file | <i>no value</i> | <i>no value</i> |
| browscap | <i>no value</i> | <i>no value</i> |
| default_charset | UTF-8 | UTF-8 |
| default_mimetype | text/html | text/html |
| disable_classes | <i>no value</i> | <i>no value</i> |

Cómo ver la configuración del intérprete PHP

La función `phpinfo()` crea un largo conjunto de tablas que muestran los ajustes para el intérprete PHP y sus valores por defecto. Estos ajustes afectan a todos los ficheros PHP que ejecuta el intérprete PHP.

Los archivos de texto descritos en el apartado anterior (`php.ini` , `httpd.conf` y `.htaccess`) pueden ser utilizados para cambiar estas configuraciones.

La siguiente tabla describe algunas de las configuraciones más populares del intérprete PHP. Se encuentran bajo el apartado *Core* a menos que se indique lo contrario.

Cómo ver la configuración del intérprete PHP

| SETTING | DESCRIPTION |
|----------------------------------|--|
| <code>default_charset</code> | Default character encoding. (This should be set to UTF-8.) |
| <code>display_errors</code> | Turn on/off errors on in HTML page. On for development. Off when site goes live. |
| <code>log_errors</code> | Turn on/off saving of errors to a log file. On when the site goes live. |
| <code>error_log</code> | Path to the log file that errors can be written to when the site goes live. |
| <code>error_reporting</code> | What errors should be recorded. (E_ALL is the setting to show all errors.) |
| <code>upload_max_filesize</code> | Maximum size of a single file a browser can upload to the server. |
| <code>max_execution_time</code> | Maximum number of seconds a script can run before the PHP interpreter stops it. |
| <code>date.timezone</code> | Default timezone used by the server. Shown under the heading Date. |

Cambiando ajustes del interprete: php.ini

El archivo `php.ini` es un archivo largo ya que contiene todos los ajustes para el intérprete PHP. También tiene un montón de comentarios para explicar la configuración. Cualquier cosa después de un punto y coma es un comentario.

Las configuraciones se controlan utilizando **directivas** que son como variables. Sólo edita los valores de las directivas (no borres ninguna). Para encontrar la configuración que quieres controlar, abre el archivo y busca esa configuración.

La lista completa de directivas se encuentra en: <https://www.php.net/manual/en/ini.list.php>

Cambiando ajustes del interprete: php.ini

Cada directiva comienza en una nueva línea y se compone de:

- Una opción a modificar
- Un operador de asignación
- El valor que debe tener

Cuando el valor es:

- Cadena: colócalo entre comillas
- Número: no utilices comillas
- Booleano: no utilices comillas



The diagram shows two lines of PHP configuration code. The first line is `date.timezone = "Europe/Rome"` and the second line is `display_errors = On`. Brackets are drawn under each line. The bracket under the first line is labeled "OPTION" and the bracket under the second line is labeled "VALUE".

```
date.timezone = "Europe/Rome"  
display_errors = On  
└──┬──┘      └──┬──┘  
    OPTION      VALUE
```

Cambiando ajustes del interprete: php.ini

```
; A selection of the values that can be changed in the php.ini file with comments
default_charset      = "UTF-8"          ; Default character set used
display_errors       = On               ; Whether or not to show errors on screen
log_errors           = On               ; Write errors to a log file
error_reporting       = E_ALL           ; Show all errors
upload_max_filesize  = 32M              ; Max size of a file that can be uploaded
post_max_size        = 32M             ; Max amount of data sent via HTTP POST
max_execution_time   = 30               ; Max execution time of each script, in seconds
memory_limit         = 128M            ; Max amount of memory a script may consume
date.timezone        = "Europe/Rome"   ; Default timezone
```


Cambiando ajustes del servidor: .htaccess

Los archivos `.htaccess` pueden añadirse a cualquier directorio de un servidor web Apache. Anulará la configuración del intérprete PHP para los archivos de esa carpeta. También afecta a las carpetas hijas y a los archivos que contienen.

Cambiando ajustes del servidor: .htaccess

Un archivo `.htaccess` sólo necesita contener los ajustes que desea anular en `php.ini`. Las configuraciones utilizan los mismos nombres y valores que las de `php.ini` excepto que están precedidas por:

- `php_flag` si el valor de la directiva es un booleano (representando un ajuste que puede estar activado o desactivado)
- `php_value` si hay más de dos opciones (por ejemplo, números, ubicaciones o esquemas de codificación)

Los comentarios comienzan con el símbolo `#` y deben aparecer en una nueva línea (no en la misma línea que la directiva).

Cambiando ajustes del servidor: .htaccess

```
php_value date.timezone "Europe/London"  
php_flag  display_errors On
```

TYPE OPTION VALUE

Cambiando ajustes del servidor: .htaccess

El siguiente archivo `.htaccess` garantizará que se muestren errores en las páginas HTML en muchos de los ejemplos que estudiaremos en este bloque.

El archivo `.htaccess` también puede controlar opciones del servidor web Apache que el archivo `php.ini` no puede.

Cambiando ajustes del servidor: .htaccess

```
# sample .htaccess file used in code examples (options described in php.ini example)
php_value  default_charset      "UTF-8"
php_flag   display_errors      On
php_flag   log_errors           Off
php_value  error_reporting      -1
php_value  upload_max_filesize  32M
php_value  post_max_size        32M
php_value  max_execution_time   30
php_value  memory_limit         128M
php_value  date.timezone        "Europe/London"
```



Contenidos del bloque

Contenidos del bloque

Este bloque se compone de las siguientes unidades didácticas:

5. Funciones integradas

Cada una de las funciones integradas de PHP realiza una tarea específica que los programadores a menudo necesitan lograr cuando trabajan con datos. Las funciones integradas se introducen primero porque se utilizan en cada unidad del resto del módulo.

Contenidos del bloque

6. Obteniendo datos de los navegadores

Esta unidad muestra cómo el intérprete PHP puede acceder a los datos enviados desde un navegador, comprobar que los datos que la página necesita han sido proporcionados, y comprobar que están en el formato correcto. También aprenderás cómo asegurarte de que los datos suministrados por los visitantes son seguros para ser mostrados en una página.

Contenidos del bloque

7. Imágenes y archivos

Si permites a los usuarios enviar imágenes u otros archivos a un sitio web, necesitas saber cómo el intérprete PHP maneja esos archivos. Esta unidad también muestra cómo realizar tareas como cambiar el tamaño de las imágenes y crear versiones más pequeñas de las imágenes conocidas como miniaturas.

Contenidos del bloque

8. Fechas y horas

Las fechas y horas se escriben de muchas maneras diferentes, por lo que necesita saber cómo PHP puede ayudarle a formatear fechas y horas de manera consistente. También aprenderemos a realizar tareas comunes como representar un intervalo de tiempo y manejar eventos recurrentes.

Contenidos del bloque

9. Cookies y sesiones

Esta unidad muestra cómo los archivos de texto llamados *cookies* pueden ser guardados en el navegador de un usuario para almacenar información sobre ese visitante. También muestra cómo se pueden utilizar las sesiones para almacenar información en el servidor web durante un corto periodo de tiempo (como una única visita a un sitio).

Contenidos del bloque

10. Gestión de errores

Todos cometemos errores cuando escribimos código, lo que resulta en que el intérprete PHP cree mensajes de error. Esta unidad te muestra cómo leer estos mensajes de error, así como técnicas que nos ayudarán a encontrar y resolver errores en tu código.

Anexo I: Creación e instalación de un certificado en XAMPP

Los certificados pueden utilizarse para cifrar los datos que se envían entre navegadores y servidores. Esto ayuda a garantizar que los datos permanezcan seguros (y no puedan ser interceptados mientras se mueven entre ambos).

Cuando desarrolles sitios localmente (en su propio ordenador), querrás asegurarse de que el entorno es lo más parecido posible al del servidor web, e instalar un certificado ayudará a reflejar los servidores web que los utilizan.

Anexo I: Creación e instalación de un certificado en XAMPP

Para obtener un certificado para un sitio web, se acude a una Autoridad de Certificación. Cuando trabajes localmente, tendrás que crear el certificado tu mismo, y luego decirle a Apache cómo utilizarlo.

Este anexo describe cómo:

- Crear un certificado
- Configurar XAMPP para que Apache pueda utilizarlo

Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado

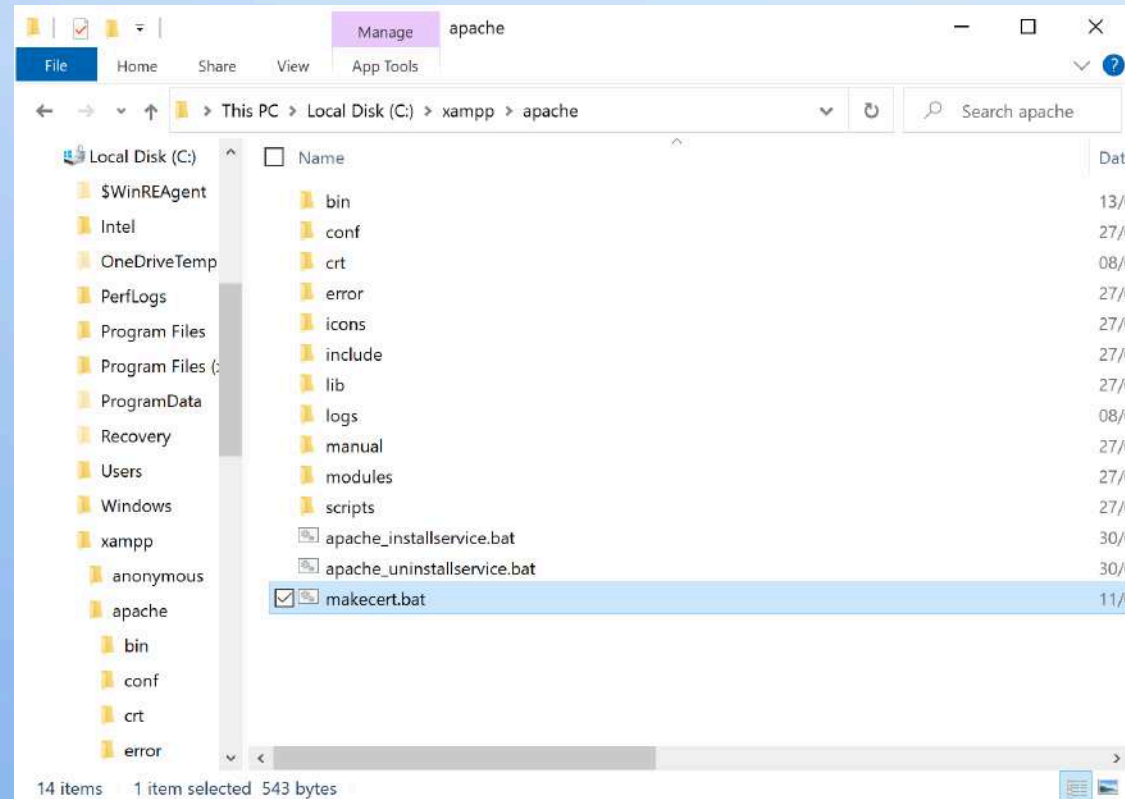
El primer paso es crear el certificado que el servidor puede utilizar. Navega a la carpeta donde XAMPP fue instalado, y mira dentro de la carpeta apache. Típicamente la ruta a esta carpeta es:

```
C:/xampp/apache
```

En esa carpeta, deberías ver un archivo llamado `makecert.bat`.

Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado



Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado

Haz doble clic en el archivo `makecert.bat` y se abrirá una ventana de línea de comandos que te hará algunas preguntas.

La primera pregunta es para una contraseña para la clave de su certificado. La llaman **frase de paso (passphrase)**. Debes recordarla porque te la volverá a preguntar antes de terminar el proceso.

Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado

A continuación te pregunta por tu

- Nombre del país (código de dos letras)
- Estado o provincia (nombre completo)
- Nombre de la localidad (por ejemplo, ciudad)
- Nombre de la organización (por ejemplo, empresa)
- Nombre del servidor (nombre común - suele ser *localhost*)
- Dirección de correo electrónico

Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado

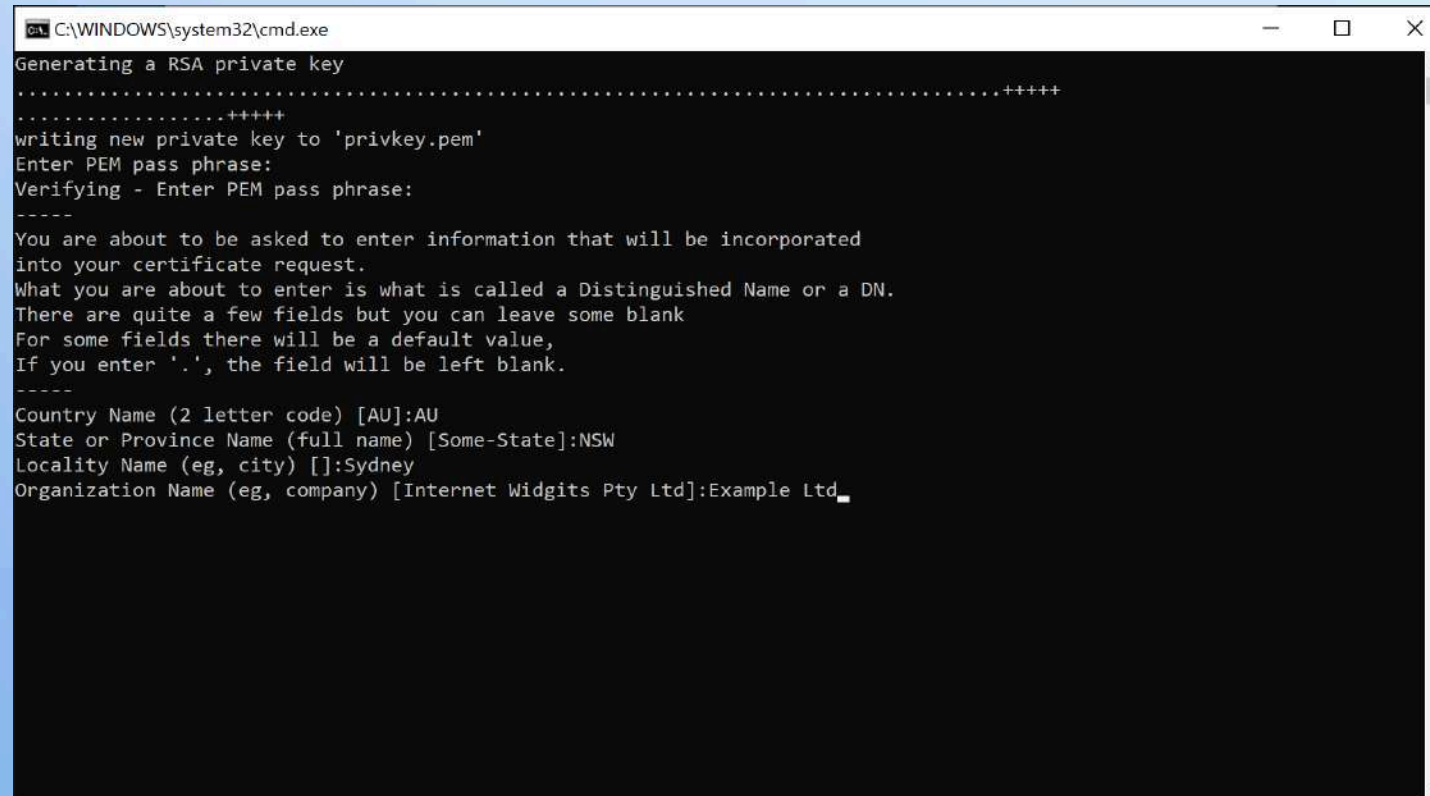
Es posible que le pida una contraseña de comprobación (que suelen utilizar las autoridades de certificación para confirmar su identidad) y, a continuación, te pedirá la frase de contraseña que hayas introducido en primer lugar.

Cuando hayas completado esto, se habrán añadido dos archivos a la carpeta apache:

```
C:\xampp\apache\conf\ssl.crt  
C:\xampp\apache\conf\ssl.key
```

Anexo I: Creación e instalación de un certificado en XAMPP

Creando un certificado autofirmado



```
C:\WINDOWS\system32\cmd.exe
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:Sydney
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Example Ltd_
```

Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

A continuación, tienes que configurar Apache para que utilice el nuevo certificado que acabas de crear.

Abre el Explorador de Windows y ve al siguiente directorio:

```
C:/xampp/apache/conf/extra/
```

Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

A continuación, busca el archivo llamado `httpd-vhosts.conf` y ábrelo en tu editor de código favorito.

Añade la entrada que se muestra a continuación, asegurándote de que la ruta a tu carpeta `htdocs` es correcta.

Luego reinicia Apache utilizando el Panel de Control de XAMPP.

Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

```
<VirtualHost *:443>
    DocumentRoot "C:/xampp/htdocs"
    ServerName localhost
    SSLEngine on
    SSLCertificateFile "conf/ssl.crt/server.crt"
    SSLCertificateKeyFile "conf/ssl.key/server.key"
    ErrorLog "logs/localhost-error.log"
    CustomLog "logs/localhost-access.log" combined
    <Directory "C:/xampp/htdocs">
        Require all granted
    </Directory>
</VirtualHost>
```

Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

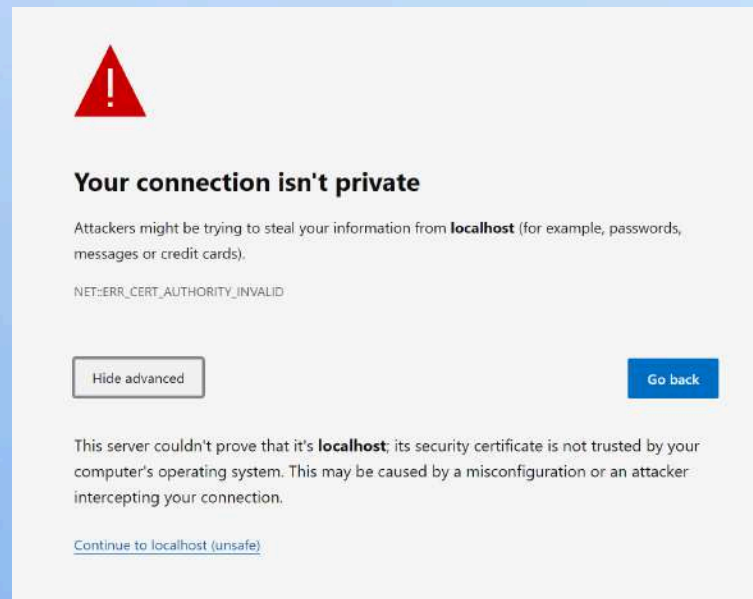
Ahora, deberías poder utilizar `https://localhost/` para solicitar archivos utilizando SSL localmente.

Debido a que has creado el certificado tú mismo, y el navegador no puede verificarlo con una Autoridad de Certificación, el navegador probablemente te advertirá de que el sitio no es seguro.

Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

Tendrás que decirle que estás de acuerdo en continuar con localhost (en la mayoría de los navegadores, esto requiere hacer clic en una opción de configuración avanzada).



Anexo I: Creación e instalación de un certificado en XAMPP

Configurando Apache para usar el certificado

A continuación, podrás seguir utilizando el sitio con el SSL instalado.

Si utilizas un número de puerto en tu URL, es posible que tengas que actualizarlo a 443, en lugar de 80.

Anexo II: Cómo establecer la codificación de caracteres en los editores de código

Al crear sitios web, los archivos de código deben guardarse utilizando la codificación de caracteres UTF-8 para garantizar que todos los caracteres se muestren como se desea.

UTF-8 es la codificación de caracteres por defecto utilizada en los editores de código e IDEs (Entornos de Desarrollo Integrado) que se muestran a continuación, pero se pueden utilizar las siguientes instrucciones para comprobar que esta configuración no ha sido modificada.

Anexo II: Cómo establecer la codificación de caracteres en los editores de código

Sublime Text

Para abrir el archivo de texto que contiene la configuración de Sublime Text, dirígete a:

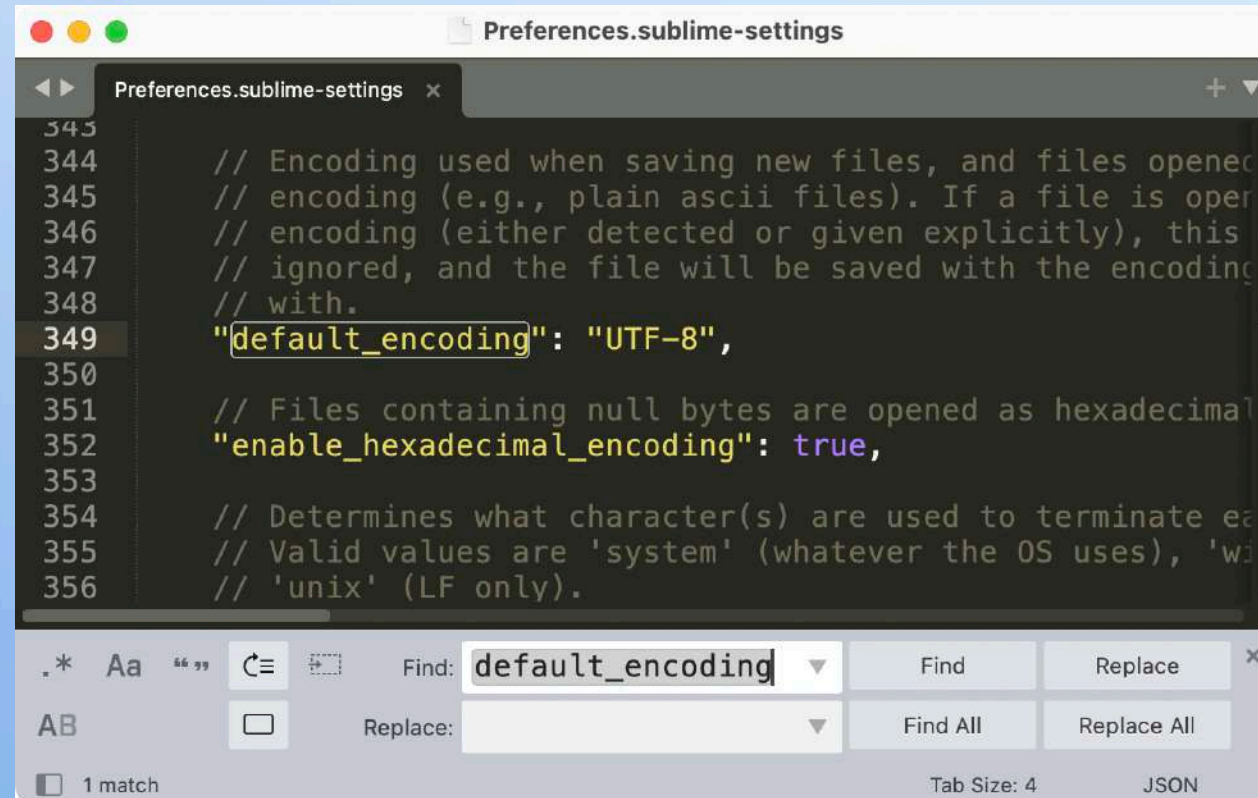
- MacOS: Sublime Text > Preferencias > Configuración
- Preferencias > Configuración

Busca las palabras: `default_encoding` .

El valor debe ser `"default_encoding": "UTF-8",`

Anexo II: Cómo establecer la codificación de caracteres en los editores de código

Sublime Text



Anexo II: Cómo establecer la codificación de caracteres en los editores de código

VS Code

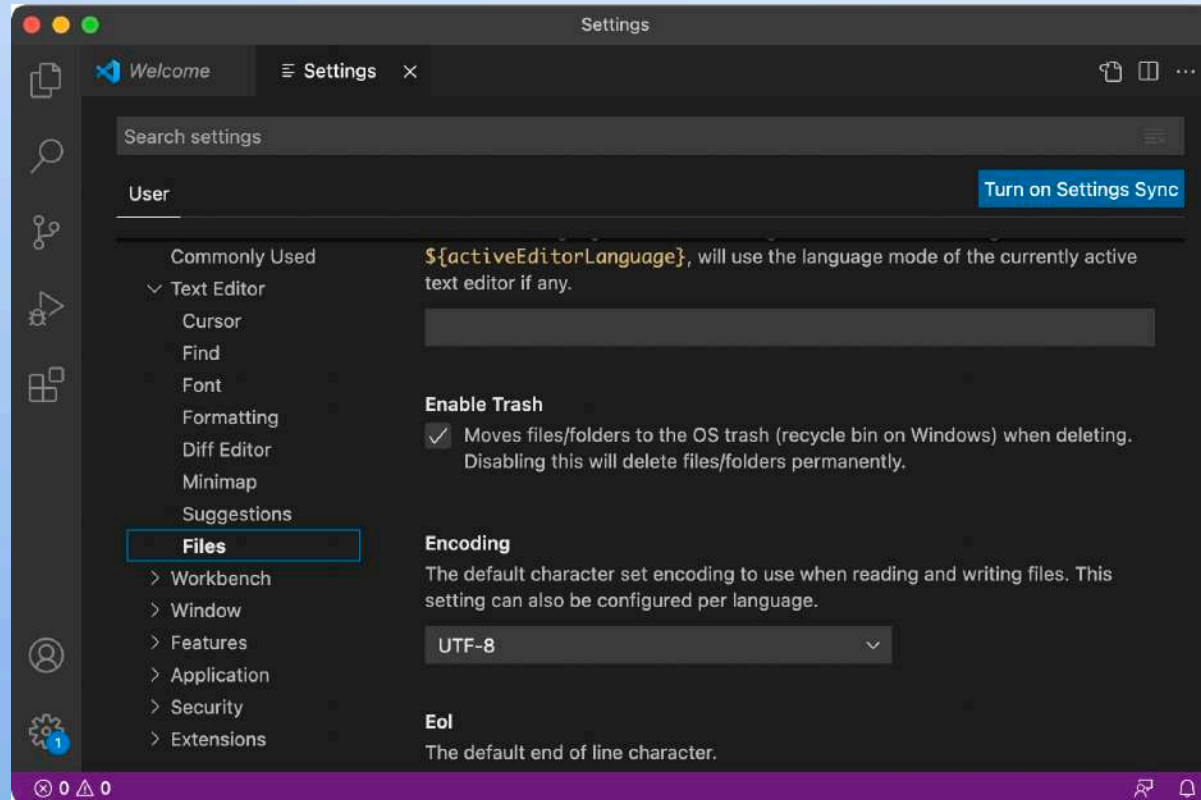
Para abrir la ventana de Configuración de VS Code, ve a:

- Código > Preferencias > Configuración
- Archivo > Preferencias > Configuración

Luego selecciona *Editor de texto > Archivos*. La codificación por defecto debe ser UTF-8 .

Anexo II: Cómo establecer la codificación de caracteres en los editores de código

VS Code



Anexo II: Cómo establecer la codificación de caracteres en los editores de código

PhpStorm

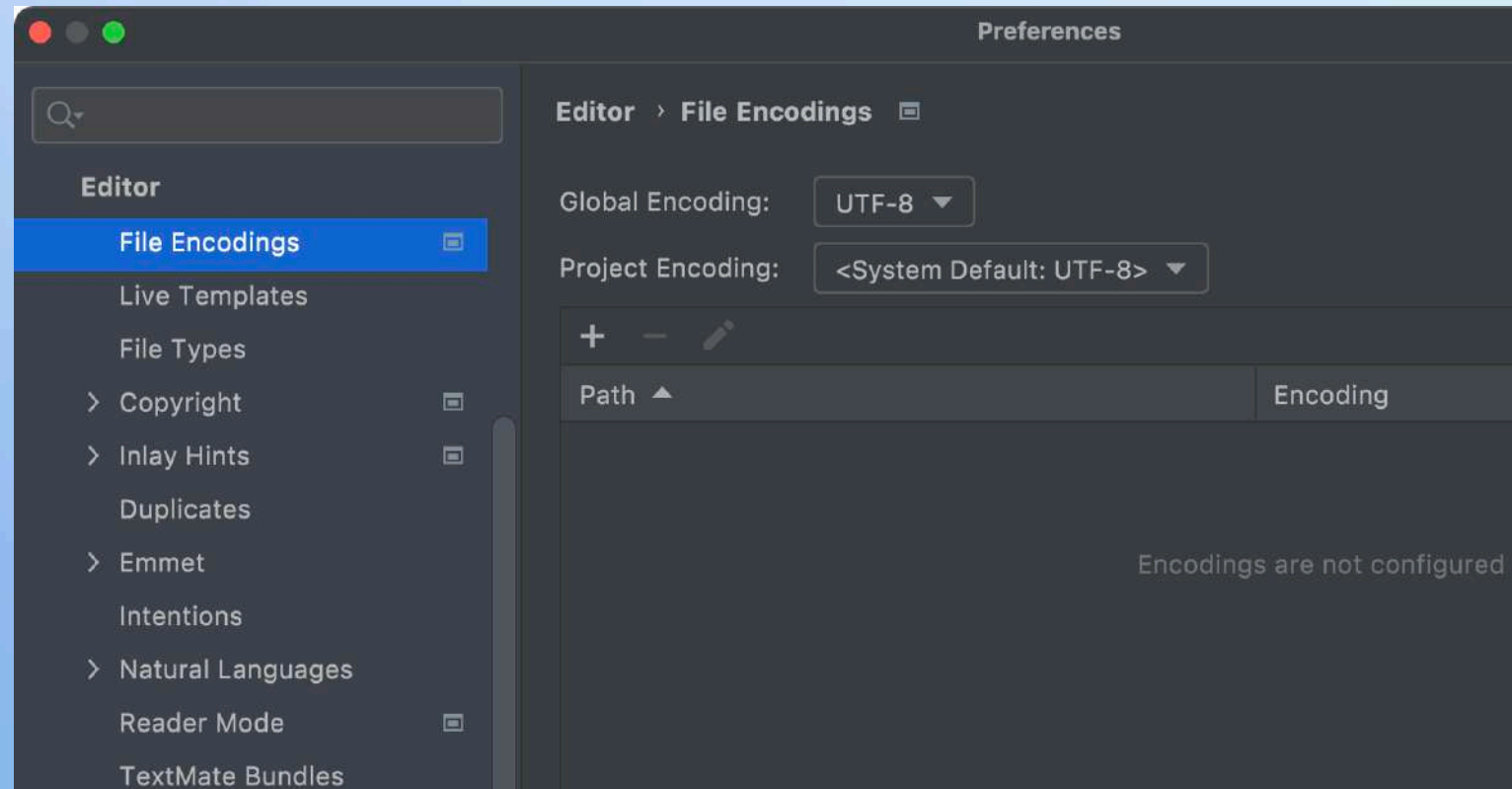
Para abrir la ventana de Preferencias de PHP Storm, ve a:

- PhpStorm > Preferencias
- Archivo > Configuración

Selecciona *Editor > Codificación de Archivos*. Hay cajas de selección desplegables para *Global* y *Project encodings*. Ambas deben estar en `UTF-8`.

Anexo II: Cómo establecer la codificación de caracteres en los editores de código

PhpStorm



Anexo III: Diferentes tipos de directorio raíz

Cuando se habla de un sistema de archivos, el directorio raíz del sistema de archivos es la carpeta superior del sistema de archivos.

Al crear un sitio web, hay otros dos directorios raíz para ese sitio, la raíz de la aplicación y la raíz del documento.

Anexo III: Diferentes tipos de directorio raíz

Raíz del sistema de archivos

La carpeta raíz del sistema de archivos suele referirse a la carpeta superior de tu ordenador.

- En Windows, se suele representar por una letra de unidad (seguida de dos puntos), y luego una barra invertida para representar la carpeta raíz de esa unidad, como `C:\`
- En Mac o Linux, suele representarse con una barra inclinada `/`.

Anexo III: Diferentes tipos de directorio raíz

Raíz de la aplicación

La carpeta raíz de la aplicación contiene todos los archivos que necesita un sitio web cuando se traslada a un nuevo servidor.

Puede contener archivos de clases, definiciones de funciones, includes y otros archivos que están por encima de la raíz del documento. El servidor web puede acceder a estos archivos, pero no se pueden abrir a través de una URL.

Anexo III: Diferentes tipos de directorio raíz

Raíz de documentos

La carpeta raíz del documento contiene todos los archivos que se pueden solicitar a través de una URL, incluyendo CSS, JavaScript y archivos de imagen.

Esta carpeta debe corresponder al nombre de dominio, por lo que si tu carpeta raíz de documentos contiene una carpeta llamada *img* que contiene una imagen llamada *test.jpg* y el dominio es *example.org* debería poder verse en:

```
http://example.org/img/test.jpg .
```

Bloque B

Páginas Web Dinámicas

Introducción

