



Desarrollo web en entorno servidor

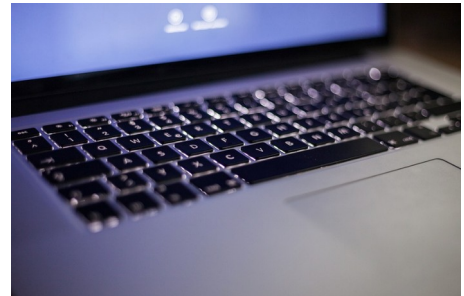
Unidad 1: Selección de arquitecturas y herramientas de programación

ÍNDICE

INTRODUCCIÓN.....	3
OBJETIVOS / CAPACIDADES.....	5
PROYECTO DE LA UNIDAD.....	6
1. MODELOS DE PROGRAMACIÓN EN ENTORNOS CLIENTE/SERVIDOR. CARACTERÍSTICAS. PROTOCOLO HTTP.....	9
2. MECANISMOS DE EJECUCIÓN DE CÓDIGO EN UN SERVIDOR WEB	11
3. GENERACIÓN DINÁMICA DE PÁGINAS WEB. VENTAJAS. CGI (COMMON GETAWAY INTERFACE).....	13
Cuestionario.....	15
4. LENGUAJES DE PROGRAMACIÓN EN ENTORNO SERVIDOR. TIPOS. CARACTERÍSTICAS.....	16
5. INTEGRACIÓN CON LOS LENGUAJES DE MARCAS.....	18
6. TECNOLOGÍAS ASOCIADAS. TIPOS DE SERVIDORES WEB. INSTALACIÓN Y CONFIGURACIÓN DE MÓDULOS PARA LA INTERPRETACIÓN DE LOS LENGUAJES DE SCRIPT DE SERVIDOR. 20	
Cuestionario.....	22
7. SERVIDORES DE APLICACIONES. FUNCIONALIDADES.....	23
8. INTEGRACIÓN CON LOS SERVIDORES WEB Y FUNCIONALIDAD DE LOS SERVIDORES DE APLICACIONES.....	25
9. HERRAMIENTAS DE PROGRAMACIÓN Y DEPURACIÓN. TIPOS...27	
10. EDITORES Y COMPILADORES.....	29
Cuestionario.....	32
RESUMEN.....	33
RECURSOS PARA AMPLIAR.....	34
BIBLIOGRAFÍA.....	35
GLOSARIO.....	36

INTRODUCCIÓN

El éxito de la web tuvo lugar gracias al **protocolo HTTP y al lenguaje HTML**. El protocolo HTTP es una implementación fácil y sencilla de un sistema de comunicaciones que permite enviar cualquier tipo de fichero por la red.



La web era un conjunto de páginas estáticas, documentos simples de información que podían consultarse o descargarse. Poco a poco fue evolucionando hasta permitir el desarrollo de páginas dinámicas que podían crearse en función de la petición enviada. Este método se conoce como **CGI (Common Gateway Interface)**. Los CGI definen una comunicación entre el cliente y el servidor mediante HTTP en la que los clientes pueden pedir información a los programas ejecutados en el servidor. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa. El resultado final de la ejecución son objetos MIME.

Pero los CGI tienen un punto débil: cada vez que se recibe una petición, el servidor web inicia un proceso para ejecutar el programa CGI. Además, si la web tiene muchos accesos al CGI, esto supone grandes problemas de rendimiento.

Por eso se empezaron a desarrollar **alternativas al CGI**. Las soluciones consisten en dotar al servidor de una interpretación de algún **lenguaje de programación** (PHP, JSP, etc.) que permita incluir el código en las páginas de manera que el servidor siga lo que está ejecutando, reduciendo así el tiempo de respuesta.

A partir de este momento se ha producido una explosión del número de arquitecturas y lenguajes de programación que nos permiten desarrollar aplicaciones web. Se presentan los lenguajes de programación integrados en el servidor que permiten

interpretar instrucciones incrustadas en las páginas HTML y un sistema de ejecución de programas más enlatado con el servidor que no presenta los problemas de rendimiento de los CGI.

Esta primera unidad servirá para tener una **perspectiva general de todos estos conceptos** y entender la **evolución de las tecnologías**, sobre todo las del lado del servidor, hasta nuestros días.

OBJETIVOS / CAPACIDADES

En esta unidad de aprendizaje, las capacidades que más se van a trabajar son:

- ✓ Seleccionar lenguajes, objetos y herramientas, interpretando las especificaciones para desarrollar aplicaciones web con acceso a bases de datos.
- ✓ Utilizar herramientas y lenguajes específicos, cumpliendo las especificaciones, para desarrollar e integrar componentes software en el entorno del servidor web.
- ✓ Emplear herramientas específicas, integrando la funcionalidad entre aplicaciones, para desarrollar servicios empleables en aplicaciones web.
- ✓ Desarrollar la creatividad y el espíritu de innovación para responder a los retos que se presentan en los procesos y organización de trabajo y de la vida personal.
- ✓ Tomar decisiones de forma fundamentada analizando las variables implicadas, integrando saberes de distinto ámbito y aceptando los riesgos y la posibilidad de equivocación en las mismas, para afrontar y resolver distintas situaciones, problemas o contingencias.



PROYECTO DE LA UNIDAD

Antes de empezar a trabajar el contenido, te presentamos la **actividad** que está relacionada con esta unidad de aprendizaje. Se trata de un **caso práctico** basado en una **situación real** con la que te puedes encontrar en tu puesto de trabajo. Con esta actividad se evaluará la puesta en práctica de los **criterios de evaluación** vinculados al resultado de aprendizaje que se trabaja en esta unidad. Para realizarla deberás hacer lo siguiente: lee el enunciado que te presentamos a continuación, dirígete al área general del módulo profesional, concretamente a la actividad de evaluación que se encuentra dentro de esta unidad, allí encontrarás todos los detalles sobre fecha y forma de entrega, objetivos... A lo largo de la unidad irás adquiriendo los conocimientos necesarios para ir elaborando este proyecto.

Enunciado:

Instalación de XAMPP y ejecución de scripts

Como hemos visto en la unidad 1 el desarrollador, una vez elegido el entorno de desarrollo y todos sus componentes necesarios, se verá obligado a tomar una última decisión para abordar sus proyectos: generar contenido dinámico mediante CGI o utilizar un lenguaje de script específico para el desarrollo web como PHP.

En esta actividad de evaluación crearemos una pequeña página web simulando la cabecera de un periódico y generando dinámicamente la fecha, primero con CGI usando Perl y después con PHP.

Genera el **archivo fecha.cgi** y colócalo en la carpeta pública de tu servidor web con el siguiente código:

Código: colocación en carpeta pública del archivo .cgi

```
#!"C:\xampp\perl\bin\perl.exe"
```

```

print "Content-type: text/html\n\n";

$datestring = localtime();

print "<html>\n";

print " <head>\n";

print "    <title>Periódico script CGI en Perl</title>\n";

print "    <meta charset='utf-8'>\n";

print "    </head>\n";

print "    <body>\n";

print "        <h1>El periódico de DRED</h1>\n";

print "        </p>$datestring</p>\n";

print "        <hr/>\n";

print "        </p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Vivamus nec fringilla risus.</p>\n";

print "    </body>\n";

print "</html>\n";

```

A continuación, crea el **archivo fecha.php** y colócalo en la carpeta pública de tu servidor web con el siguiente código:

Código: colocación en carpeta pública del archivo .php

```

<html>

<head>

    <title>Periódico script CGI en Perl</title>

    <meta charset='utf-8'>

</head>

<body>

```

```
<h1>El periódico de DRED</h1>

</p><?php echo date('l jS \of F Y h:i:s A'); ?></p>

<hr/>

</p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus
nec fringilla risus.</p>

</body>

</html>
```

Ejecuta ambos scripts. Analiza las semejanzas y diferencias en función del resultado obtenido, pero también del modo de creación.

1. MODELOS DE PROGRAMACIÓN EN ENTORNOS CLIENTE/SERVIDOR. CARACTERÍSTICAS. PROTOCOLO HTTP



DESTACADO

La **arquitectura cliente-servidor** es un modelo de diseño de software donde las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones y el servidor le da respuestas.

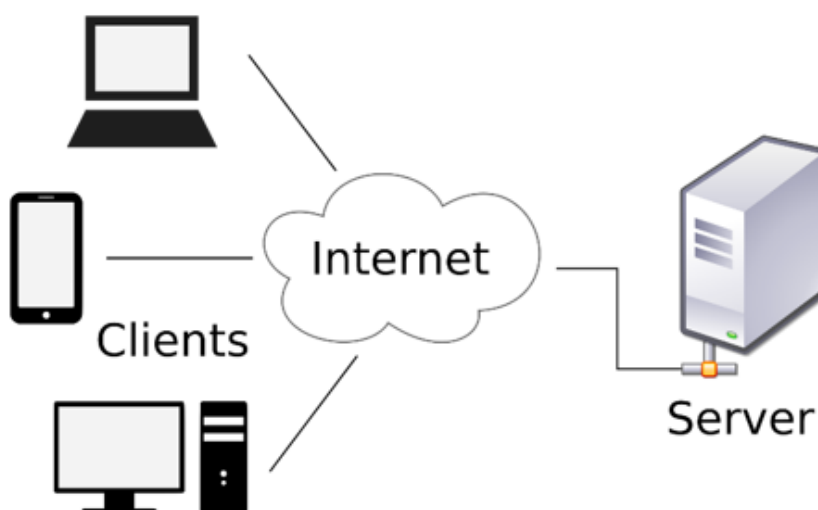


Ilustración donde múltiples clientes hacen peticiones a un servidor web a través de internet. Fuente: https://en.wikipedia.org/wiki/Client%E2%80%93server_model. Esta imagen se reproduce acogiéndose al derecho de cita o reseña (art. 32 LPI), y está excluida de la licencia por defecto de estos materiales.

A continuación, hablaremos de la **arquitectura cliente-servidor** y en los servicios World Wide Web (WWW):

- ➔ Hay muchas implementaciones en el mundo de la informática donde se utiliza la **arquitectura cliente-servidor**:
- ✓ Correo electrónico.
- ✓ Servicios de impresión.
- ✓ World Wide Web.

Normalmente asociamos la arquitectura cliente-servidor a un mecanismo con el propósito de ser desplegado en red. No es necesario que así sea, pero ciertamente es la implementación más usual.



Aunque la **arquitectura cliente-servidor** es la más utilizada no es la única que existe. Convive con otras arquitecturas como la de punto a punto (P2P), o la arquitectura de cliente-cola-cliente.

- Hace un momento hemos enumerado diferentes implementaciones de la arquitectura cliente-servidor. Ahora vamos a centrarnos en servicios **World Wide Web (WWW)** que es lo que nos atañe.



El funcionamiento de la arquitectura cliente-servidor para WWW es el siguiente: un servidor web es una pieza de software que responde a las peticiones de los navegadores y entrega la página al cliente a través de internet.

Cuando se llama a una página web por la dirección – la **URL (Uniform Resource Locator)**, por ejemplo, <https://dred.es/> –, la comunicación entre el cliente y el servidor es posible gracias a tres protocolos:

- ✓ **TCP (Transmission Control Protocol, protocolo de control de transmisión)**: es el responsable de hacer que el mensaje llegue al destino sin errores.
- ✓ **IP (Internet Protocol)**: es el responsable de hacer que el mensaje encuentre el camino hasta el servidor.
- ✓ **HTTP (Hypertext Transfer Protocol, protocolo de transferencia de hipertexto)**: es el protocolo que ha indicado el usuario a la hora de pedir el recurso al servidor. La primera parte de un recurso URL corresponde al protocolo que utilizarán cliente y servidor para intercambiar datos.

Estos protocolos forman parte de la especificación OSI y TCP/IP.

2. MECANISMOS DE EJECUCIÓN DE CÓDIGO EN UN SERVIDOR WEB

Los **mecanismos** que componen un servidor web son los siguientes:

- Un **servidor web** para recibir las peticiones de los clientes web (normalmente navegadores) y enviarles la página que solicitan (ya sean de tipo estático o dinámico). El servidor web debe conocer el procedimiento a seguir para generar la página web: qué módulo se encargará de la ejecución del código y cómo se debe comunicar con él.
- El **módulo encargado de ejecutar el código** o programa y generar la página web resultante. Este módulo va asociado con el lenguaje de programación que se utilizará para desarrollar las aplicaciones.
- Una **aplicación de base de datos**, que normalmente también será un servidor. Este módulo no es estrictamente necesario, pero en la práctica se utiliza en todas las aplicaciones web que utilizan grandes cantidades de datos para almacenarlos.
- Todo aquello que da **soporte al servidor**: hardware y sistema operativo que lo gobierna.



Componentes necesarios para la ejecución de código en un servidor web.



Además de los componentes a utilizar, también es importante decidir cómo se va a **organizar el código de la aplicación**. Muchas de las arquitecturas que se usan en la programación de aplicaciones web ayudan a estructurar el código de las aplicaciones en **capas o niveles**.

3. GENERACIÓN DINÁMICA DE PÁGINAS WEB. VENTAJAS. CGI (COMMON GETAWAY INTERFACE)

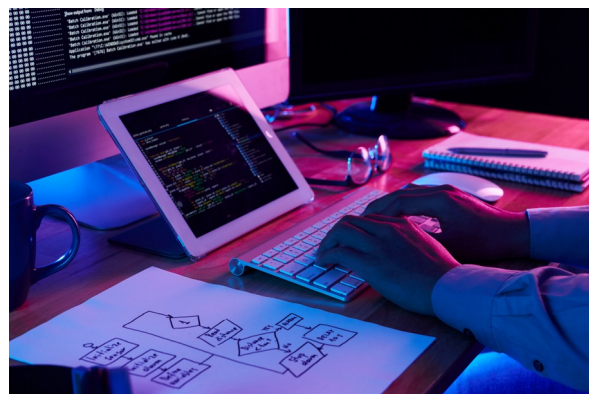


DESTACADO

Una **página web estática** está compuesta por archivos HTML individuales por cada página que son pre-generados y presentados al usuario a través del navegador de la misma forma.

Si se quisiera crear una página web estática, no es necesario utilizar entornos de desarrollos complejos, solo se necesita un editor de texto como el **bloc de notas** y saber de **CSS y HTML**. El lenguaje de marcas HTML (HyperText Markup Language) permite crear la estructura de un documento web y el lenguaje de diseño CSS (Cascading Style Sheets) define y aplica los estilos.

- ➔ Una **página web dinámica** tiene elementos que cambian continuamente, son interactivos y funcionales, en lugar de ser simplemente informativos. Por supuesto, eso requiere utilizar más que solo código HTML y CSS.



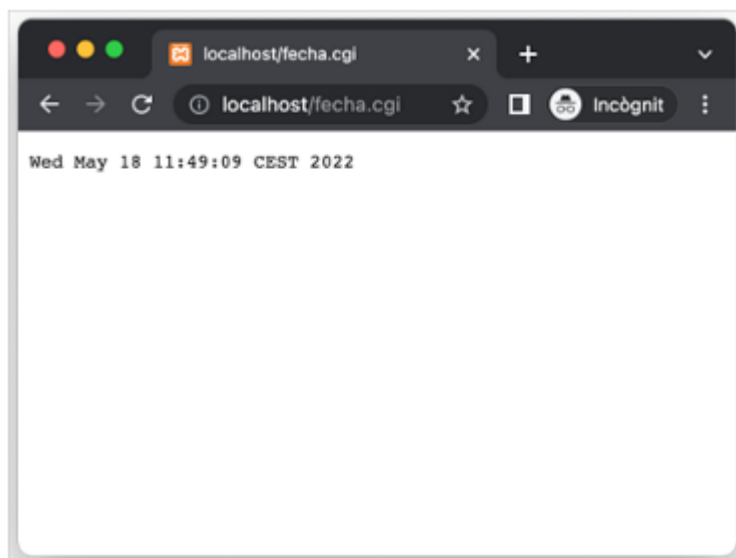
Uno de los mecanismos para la creación de página web dinámicas es la tecnología CGI (Common Gateway Interface).

- ➔ Las **aplicaciones CGI** fueron una de las primeras prácticas de crear contenido dinámico para las páginas web. En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo. Este programa puede estar escrito en cualquier lenguaje que soporte el servidor, normalmente se suelen usar lenguajes de script. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional.

- En el siguiente ejemplo **crearemos un CGI** que nos mostrará la fecha por pantalla. Cada vez que un cliente “pregunte” la fecha al servidor mostrará algo distinto. He aquí un contenido dinámico con CGI.

```
#!/bin/bash  
echo Content-type: text/plain  
echo  
/bin/date
```

El resultado de la ejecución del código CGI es:



Vídeo: creación y uso del CGI



Visualiza el siguiente vídeo en el que podrás ver cómo se crea y se usa el CGI mostrado en el ejemplo que encuentras en este mismo apartado.

Cuestionario



Lee el enunciado e indica la opción correcta:

El lenguaje de marcas HTML (HyperText Markup Language):

- a. Define y aplica los estilos de una página web.
- b. Permite crear la estructura de un documento web.
- c. Aporta la lógica de la aplicación.



Lee el enunciado e indica la opción correcta:

Las páginas web dinámicas:

- a. Se pueden crear con HTML.
- b. CGI es una opción para crear páginas web dinámicas.
- c. Sólo se pueden crear con CGI.



Lee el enunciado e indica la opción correcta:

La arquitectura cliente servidor:

- a. Solo sirve para aplicaciones WWW.
- b. Es un mecanismo donde el cliente hace peticiones y el servidor las responde.
- c. Sólo puede haber un cliente y un servidor.

4. LENGUAJES DE PROGRAMACIÓN EN ENTORNO SERVIDOR. TIPOS. CARACTERÍSTICAS



RECUERDA

En el punto anterior hemos visto el primer mecanismo para brindar a los servidores web la posibilidad de generar páginas con contenido dinámico.

La tecnología CGI fue muy usada, pero presentaba algunos **inconvenientes**:

→ **Seguridad**: en el script visto en el punto anterior hemos sacado la fecha del sistema operativo, por lo tanto, ese script tiene la posibilidad de operar con el sistema. Si el script tuviese código malicioso estaríamos frente una amenaza seria.



→ **Rendimiento**: ya que cada ejecución del script CGI era un proceso distinto para el sistema, cuando este tenía que ejecutar muchos procesos debido a mucha demanda, provocaba cuellos de botella en cuanto a rendimiento se refiere.

Para resolver estos problemas, se buscó desarrollar una tecnología que permitiera ejecutar, en un único proceso del servidor, todas las **peticiones de ejecución de código** sin importar la cantidad de clientes que se conectaban concurrentemente.



DESTACADO

*Así surgieron los denominados **servlets**, basados en la tecnología Java de Sun Microsystems, y los filtros ISAPI de Microsoft. Estos permitían ejecutar código en un único proceso externo que gestionaba todas las peticiones hacia el servidor web, consiguiendo así un punto más de seguridad y mejor rendimiento.*

A partir de aquí a la figura del programador web se escindió de la figura de informático general. Para este nuevo perfil profesional fueron surgiendo lenguajes de programación del lado del servidor más específicos para su tarea:

- **Lenguajes interpretados**, como por ejemplo las páginas ASP o PHP.
- **Lenguajes precompilados**, como en las páginas JSP o ASP.NET.

Actividad de aprendizaje 1: instalar los mecanismos de ejecución de código web

Hoy es tu primer día de trabajo en tu nueva empresa de programación. Tu primera tarea es preparar el equipo con el que vas a trabajar y para ello se te pide que instales el stack de desarrollo XAMPP.

Descarga la [versión más reciente de XAMPP](#) para tu sistema operativo y comparte el proceso de instalación en la wiki.

Seguro que mientras has buscado información de XAMPP habrás encontrado otros stacks parecidos: LAMP, WAMP, MAMP, etc. ¿Podrías explicar en la wiki las similitudes y diferencias de uno de ellos? Escoge uno del que aún no haya documentado nada tus compañeros y compañeras.

5. INTEGRACIÓN CON LOS LENGUAJES DE MARCAS



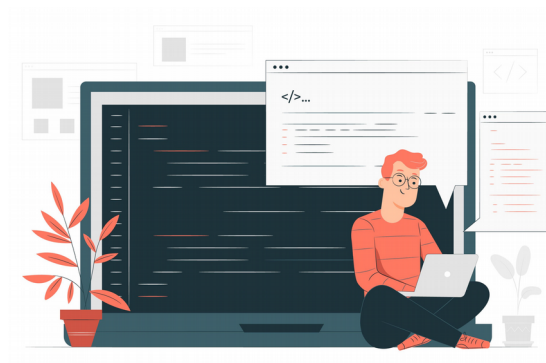
DESTACADO

Un **lenguaje de marcas** es una forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación.

El lenguaje de marcas por excelencia del desarrollo web es **HTML**. Con él se construyen páginas web y de hecho es la única estructura que entiende el cliente, que es el navegador web.

Entonces, ¿cómo se **integran** los lenguajes de programación, como **PHP**, dentro de una estructura **HTML** para generar contenido dinámico?

- ➔ La página HTML debe estar almacenada en el servidor o en su defecto el servidor debe ser capaz de generar de forma dinámica un documento HTML. De cualquiera de las dos maneras, al final, el **servidor web mandará el archivo HTML al cliente** para que éste lo consuma.



Si el contenido HTML se genera de forma dinámica el lenguaje de script se integra como una etiqueta más dentro del lenguaje de marcas.

- ➔ Fijémonos en el siguiente **ejemplo**. Tenemos una simple página web donde le añadimos contenido dinámico con PHP. En este caso mostramos la fecha del momento que se ejecute el código. En la línea 8 de la imagen se aprecia la apertura de la etiqueta `<?php`, después el código para mostrar la fecha, y finalmente el cierre de la etiqueta `?>`

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Ejemplo PHP embebido</title>
5   </head>
6   <body>
7     <h1>La fecha de hoy es:</h1>
8     <p><?php echo date('d/m/Y') ?></p>
9   </body>
10 </html>

```

→ **¿Qué se verá cuando se ejecute el script?** Pues desde el lado del cliente se verán todas las etiquetas, y lo que hay en la línea 8 ya no sería código, sino la fecha que produce la ejecución del código, por ejemplo:

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Ejemplo PHP embebido</title>
5   </head>
6   <body>
7     <h1>La fecha de hoy es:</h1>
8     <p>19/05/2022</p>
9   </body>
10 </html>

```



RECUERDA

No debe confundirse lenguaje de **marcas** con lenguaje de **programación**. Un lenguaje de marcas describe la estructura de unos datos y su relación. Un lenguaje de programación tiene, entre otros elementos, operadores que modifican el comportamiento del código que se escribe con ellos.

6. TECNOLOGÍAS ASOCIADAS. TIPOS DE SERVIDORES WEB. INSTALACIÓN Y CONFIGURACIÓN DE MÓDULOS PARA LA INTERPRETACIÓN DE LOS LENGUAJES DE SCRIPT DE SERVIDOR

Ya hemos visto que un **servidor es un ordenador que recibe peticiones** de un cliente y le sirve las respuestas. Este ordenador está gobernado por un sistema operativo, y en el caso de ser un servidor web, necesita software capaz de atender las peticiones que se mandan a través del protocolo HTTP.

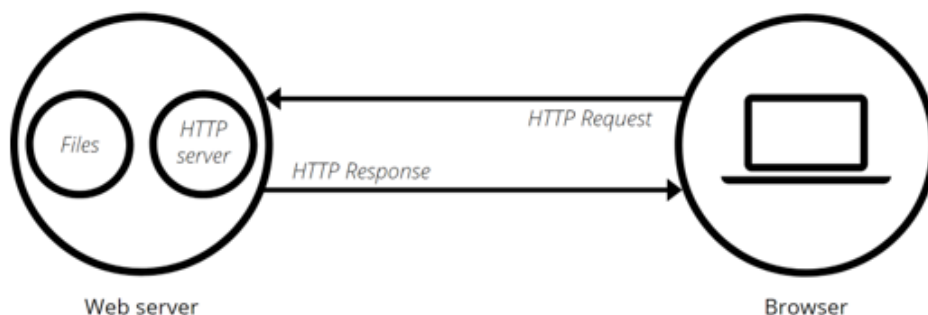


Ilustración del mecanismo request/response Fuente: <https://www.cloudcenterandalucia.es/blog/que-es-un-servidor-web-funcionamiento-y-tipos/>. Esta imagen se reproduce acogiendo al derecho de cita o reseña (art. 32 LPI), y está excluida de la licencia por defecto de estos materiales.

Hoy en día existen multitud de **servidores web**, pero quizás los más populares son:

- Apache.
- NGINX.
- IIS.

➔ En el proceso de creación de aplicaciones web es muy común que los desarrolladores tengan en su propio ordenador instalado un **entorno de desarrollo web**.

En el mercado ya hay entornos de desarrollo (que muy a menudo los mal llamamos "servidores web") que realmente unen el servidor web, que suele ser Apache, con el contenedor web, que suele ser PHP, para que se puedan instalar en la máquina de desarrollo.

- ➔ Uno de estos entornos de desarrollo es **XAMPP**. Vamos a utilizar éste ya que es multiplataforma, es decir, podrás utilizarlo tanto con Windows, como Linux como macOS.

XAMPP tiene como servidor web Apache, además de MariaDB como SGBD (sistema gestor de base de datos), y permite la ejecución de PHP y Perl para generar contenido dinámico.

Vídeo: XAMPP y PHP



Visualiza el siguiente vídeo en el que verás cómo instalar XAMPP, así como también ejecutar un script en PHP.

Actividad de aprendizaje 2: escoger un editor de texto o IDE

Siguiendo la actividad 1, busca entre los diferentes editores de texto o IDEs aquel que te parezca más atractivo usar. Recuerda que cualquiera te será útil, pero es importante que te lo sientas cómodo.

Comparte tu elección en el foro colaborativo. Explica y discute con tus compañeros por qué crees que el que has elegido será el mejor entorno de programación para tu trabajo.

Cuestionario



Lee el enunciado e indica la opción correcta:

Las siglas CGI significan:

- a. Custom Graphic Interface.
- b. Common General Information.
- c. Common Gateway Interface.



Lee el enunciado e indica la opción correcta:

Describe la estructura de unos datos y su relación:

- a. Lenguaje de programación.
- b. Lenguaje de marcas.
- c. Algoritmo.



Lee el enunciado e indica la opción correcta:

Apache, NGINX y IIS son:

- a. Servidores de aplicaciones.
- b. Servidores web.
- c. Lenguajes de programación.

7. SERVIDORES DE APLICACIONES. FUNCIONALIDADES



DESTACADO

*La **funcionalidad de un servidor de aplicaciones** es la de ejecutar los comandos solicitados por el servidor web para obtener los datos de las bases de datos. Un servidor de aplicaciones actúa como un conjunto o grupo de componentes a los que acceden los desarrolladores de software a través de una API (Interfaz de programa de aplicación).*

Un servidor web acepta y cumple las peticiones de los clientes de contenido estático (es decir, páginas HTML, archivos, imágenes y vídeos) de un sitio web. Los servidores web sólo gestionan solicitudes y respuestas HTTP.

Un servidor de aplicaciones expone la lógica empresarial a los clientes, que genera contenido dinámico. Es un marco de software que **transforma los datos para proporcionar la funcionalidad** especializada que ofrece una empresa, servicio o aplicación. Los servidores de aplicaciones mejoran las partes interactivas de un sitio web que pueden aparecer de forma diferente según el contexto de la solicitud.

A continuación, veremos las **diferencias** entre **servidor web** y **servidor de aplicaciones**:

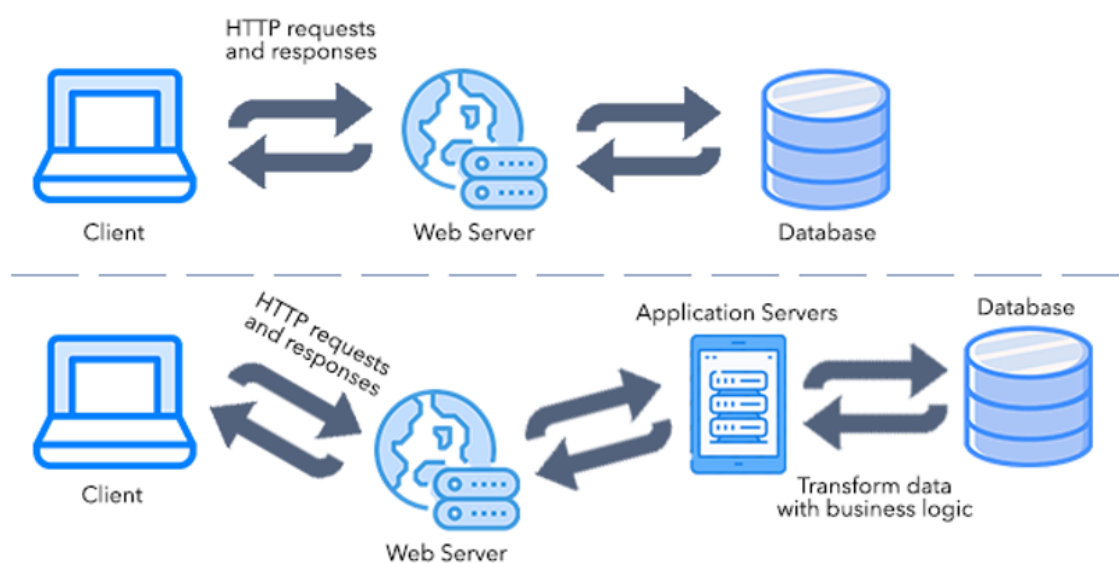
→ Servidor web:

- ✓ Entrega contenido estático.
- ✓ El contenido se entrega utilizando únicamente el protocolo HTTP.
- ✓ Sirve solo aplicaciones basadas en web.
- ✓ Sin soporte para subprocessos múltiples.
- ✓ Facilita el tráfico web que no requiere muchos recursos.

→ Servidor de aplicaciones:

- ✓ Ofrece contenido dinámico.

- ✓ Proporciona lógica empresarial a los programas utilizando varios protocolos (incluido HTTP).
- ✓ Puede servir aplicaciones web y empresariales.
- ✓ Utiliza subprocesos múltiples para admitir múltiples solicitudes en paralelo.
- ✓ Facilita procesos de ejecución más largos que requieren muchos recursos.



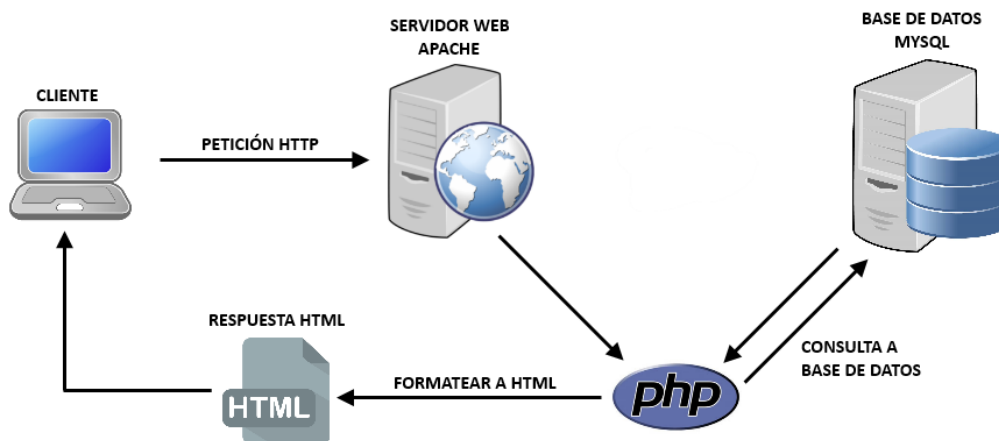
Diferencias entre el workflow sobre un servidor web y un servidor de aplicaciones..
Fuente: https://achievement.com.sg/view_blog.php?bgid=49. Esta imagen se reproduce acogiendo al derecho de cita o reseña (art. 32 LPI), y está excluida de la licencia por defecto de estos materiales.

8. INTEGRACIÓN CON LOS SERVIDORES WEB Y FUNCIONALIDAD DE LOS SERVIDORES DE APLICACIONES

Un cliente realiza una petición al servidor. Existen muchos tipos de peticiones, aunque las más comunes son las de tipo **GET y POST**.

- Las **peticiones GET**, solicitan al servidor algún tipo de información (página HTML, fichero XML, fotos, vídeos, etc.).
- Las **peticiones POST**, envían datos al servidor, como por ejemplo cuando rellenamos un formulario en un ecommerce.

El **ciclo de una petición** se compone de los siguientes elementos:



Ciclo de una petición HTTP. Fuente: <http://diymakers.es/raspberry-pi-como-servidor-web/>. Esta imagen se reproduce acogándose al derecho de cita o reseña (art. 32 LPI), y está excluida de la licencia por defecto de estos materiales.

- ➔ **1. Petición:** el cliente hace la petición desde su navegador web.
- ➔ **2. Resolución DNS:** para abrir cualquier página web debemos conocer su IP (código numérico que identifica al servidor). Para obtenerla, se ejecuta una consulta a un servidor DNS, con acceso a la agenda completa de direcciones en Internet.

- ➔ **3. Resolución DNS a IP:** una vez que el sistema operativo ha resuelto esa consulta DNS, utiliza el protocolo TCP/IP para comunicar con el servidor remoto que almacena la página que pretendemos cargar.
- ➔ **4. Preparación de la petición:** en esta fase el servidor web identifica a qué sitio corresponde la petición recibida, observando su URL. Aquí entra en juego el servidor de aplicaciones que genera el contenido dinámico.
- ➔ **5. Presentación:** finalmente, el servidor web devuelve al navegador una respuesta que identifica el tipo de datos enviados, facilitando así su funcionamiento y mostrando en pantalla la información que hemos solicitado.

Todos estos procesos se desarrollan en unos **pocos segundos**. Como las operaciones se ejecutan a nivel interno, pasan **inadvertidas** para el usuario.

- ➔ Aunque también es posible **ejecutar código en lenguaje PHP** utilizando CGI, los intérpretes PHP no se suelen utilizar con este método. Existen módulos que podemos instalar en el servidor web (Apache, NGINX, etc.) para que ejecute páginas web dinámicas.
- ➔ El **módulo mod_php** es la forma más habitual de encontrar PHP implementado en un servidor web, pero también existen alternativas como FastCGI o PHP-FPM.
- ➔ La **arquitectura Java EE** es más compleja. Para poder ejecutar aplicaciones Java EE en un servidor básicamente tenemos dos opciones: servidores de aplicaciones, que implementan todas las tecnologías disponibles en Java EE, y contenedores de servlets, que soportan solo parte de la especificación.

9. HERRAMIENTAS DE PROGRAMACIÓN Y DEPURACIÓN. TIPOS

Debemos tener en cuenta que el proceso de desarrollo de una aplicación web no suele producirse en el mismo equipo en el que finalmente se despliegue y ejecute. Habitualmente se separan estos procesos etiquetándolos en: **fase de desarrollo** y **fase de producción**.



*Como se ha mencionado anteriormente, los desarrolladores acostumbran a instalar un entorno de desarrollo que consta de servidor web, lenguaje de programación y servidor de base de datos. A menudo estos elementos se encuentran en un mismo **stack**, siendo **XAMPP** uno de los más usados.*

Con esto, ya tenemos la parte que se encarga de la **lógica**. Pero, para el ciclo completo faltan más **elementos** que enumeramos a continuación:

- ➔ **Navegador web**: que actúa como cliente o user-agent.
- ➔ **Editores de código**: ya sean editores de texto o IDEs.
- ➔ **Herramientas de tratamiento de imágenes**: es algo opcional pero también forma parte de las herramientas que debe conocer un desarrollador web.
- ➔ **Herramientas para la creación y administración de bases de datos**: a parte del motor de base de datos herramientas como phpMyAdmin o MySQL Workbench suelen ayudar a un desarrollo más cómodo.

Cuando ejecutamos el código que hemos programado, a menudo nos encontramos **errores que rompen la lógica** que habíamos imaginado. La mayoría de estos errores se evitan siguiendo buenas prácticas de programación, pero a pesar de estas siempre se "cuelan" **bugs (errores)**. *De ahí viene el verbo debug o depurar.*

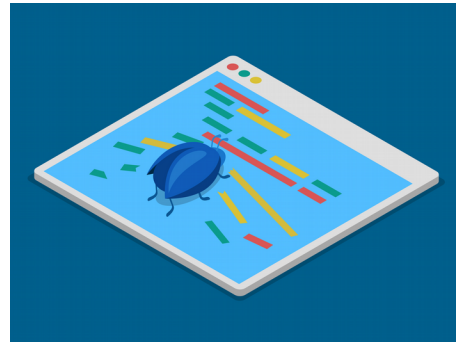
Para **depurar un código** podemos utilizar las siguientes técnicas:

➔ **Debugging:** consiste en observar valores de variables y detener temporalmente la aplicación con breakpoints.

➔ **Logs:** hace un vaciado de cómo las variables van cambiando y es más fácil rastrear la información.

➔ **Historial:** agiliza la forma de monitorear y observar los comportamientos de nuestro software. Comparando valores y agrupando información.

➔ **Reportes:** consiste en observar anomalías, acelerar el tiempo de respuesta, prevenir ataques o fallas.



Una de las herramientas que nos permiten **depurar el código en PHP es Xdebug**. Sencillamente es un módulo que trabaja con PHP y nos permite:

- Depuración de pasos.
- Mejoras en el informe de errores de PHP.
- Rastreo.
- Herramientas de perfilado.
- Análisis de cobertura de código.

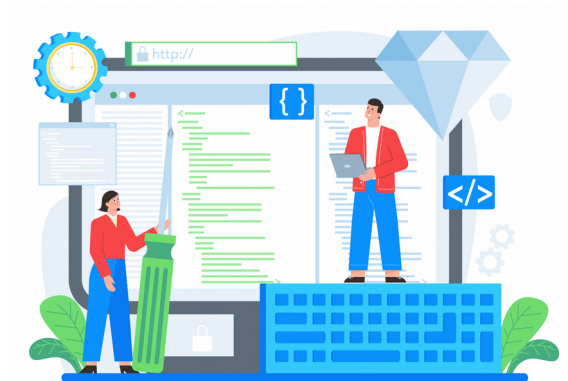
10. EDITORES Y COMPILADORES

A la hora de ponerse a escribir código, los desarrolladores pueden escoger un simple editor de textos o un robusto **IDE (Integrated Development Environment)**.

Veamos las **diferencias**:

→ Editor de textos:

- ✓ Software ligero con ayudas para escribir código (resaltado de sintaxis, autocompletado, etc).
- ✓ Soporta múltiples lenguajes y tecnologías.
- ✓ Enfocado en archivos (no acostumbran a tener el concepto de proyecto).
- ✓ Se pueden agregar plugins para sumar funcionalidades.



→ IDE:

- ✓ Integra un editor con las herramientas que necesita un desarrollador (debugger, compilador, etc.).
- ✓ Se especializa en un lenguaje o tecnología (C, Java, PHP, etc.).
- ✓ Enfocado a proyectos completos. Autogenera los scaffoldings necesarios.
- ✓ Trae herramientas integradas y configurables (por ejemplo XCode lleva un simulador de iOS).

A pesar que los desarrolladores escriben los programas con editores de texto y sentencias de alto nivel, las computadoras no pueden **comprender** directamente **estos lenguajes**, ya que solo pueden entender directamente los **lenguajes de máquina**.

Por lo tanto, los lenguajes de alto nivel como **PHP**, **JSP** o **ASP** deben traducirse a lenguajes de máquina antes de que las computadoras puedan ejecutar los programas que se han escrito. Hay **dos métodos de traducción**, uno es la **compilación** y el otro es la **interpretación**.

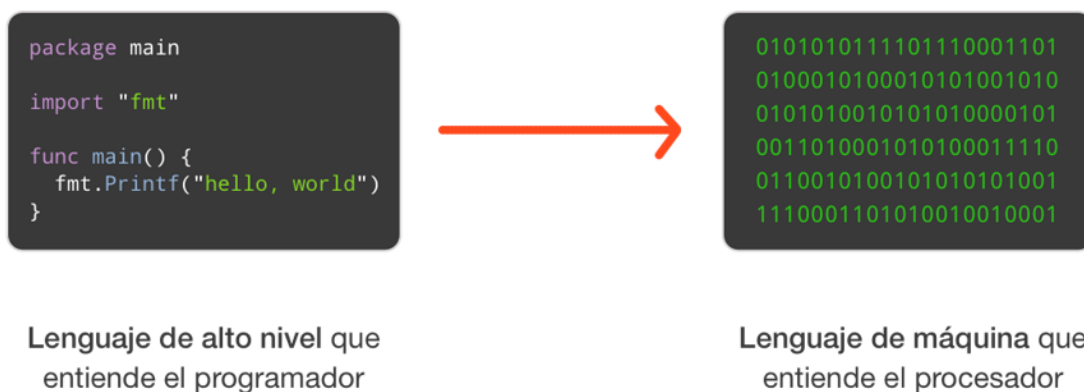


Imagen que ilustra la conversión de código de alto nivel a lenguaje máquina. Fuente: <https://blog.makeitreal.camp/lenguajes-compilados-e-interpretados/>. Esta imagen se reproduce acogiéndose al derecho de cita o reseña (art. 32 LPI), y está excluida de la licencia por defecto de estos materiales.

Tanto compiladores como interpretadores son programas que convierten el código que se escribe a lenguaje de máquina, que son las **instrucciones** que entiende el computador (el procesador para ser más exactos) en **código binario** (unos y ceros).



Un lenguaje de servidor interpretado es PHP.
Un lenguaje de servidor compilado son los servlets de Java.

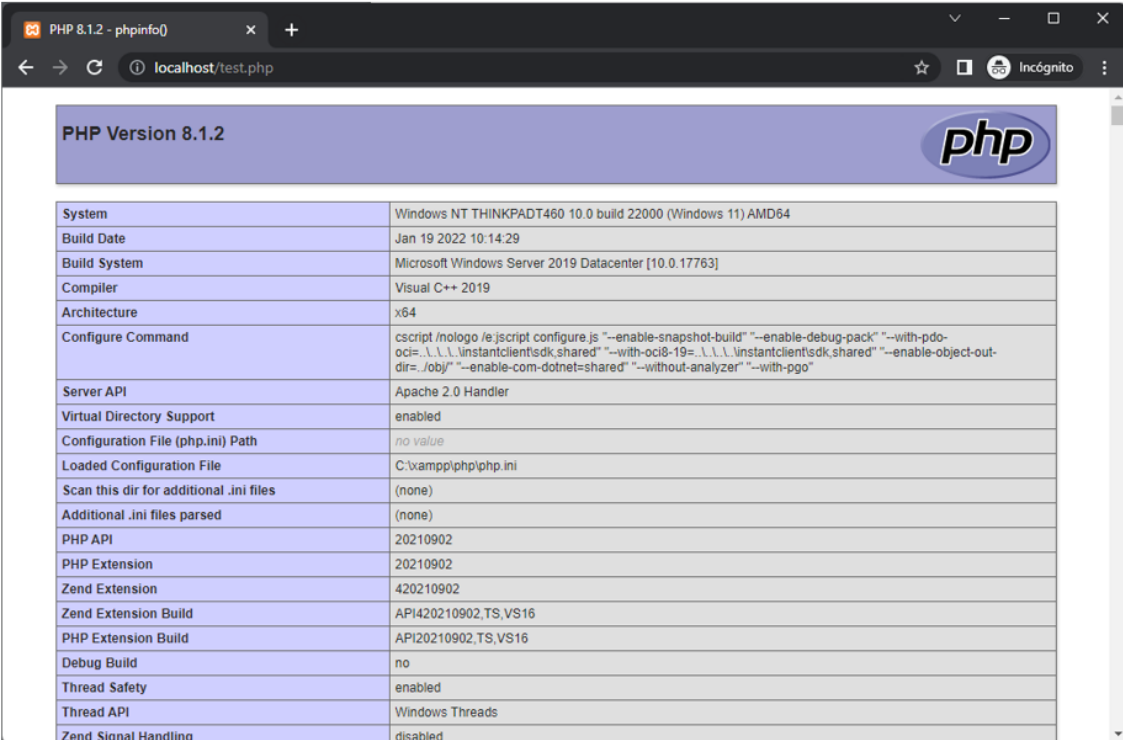
Actividad de aprendizaje 3: mi primer script en PHP

Dirígete a la carpeta pública de tu servidor web, crea un nuevo documento llamado test.php y pega el siguiente código:

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

A continuación, arranca tu servidor web. Después abre tu navegador y escribe en la barra de direcciones *http://localhost/test.php*.

Debería verse algo parecido a:



PHP Version 8.1.2	
System	Windows NT THINKPADT460 10.0 build 22000 (Windows 11) AMD64
Build Date	Jan 19 2022 10:14:29
Build System	Microsoft Windows Server 2019 Datacenter [10.0.17763]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cmdscript /nologo ie.js script configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=\\.\.\.\instantclient\sdk,shared" "--with-oci8-19=\\.\.\.\instantclient\sdk,shared" "--enable-object-out-dir=.\obj" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	no value
Loaded Configuration File	C:\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,TS,VS16
PHP Extension Build	API20210902,TS,VS16
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads
Zend Signal Handling	disabled

Crea con tus compañeros una base de conocimiento sobre la instalación de XAMPP aprovechando la wiki, donde se expliquen los errores más comunes, la diferencia de instalación según el sistema operativo, etc.

Cuestionario



Lee el enunciado e indica la opción correcta:

El lenguaje PHP:

- a. Es un lenguaje de marcas.
- b. Es un lenguaje compilado.
- c. Es un lenguaje interpretado.



Lee el enunciado e indica la opción correcta:

Los servidores web:

- a. Sirven contenido dinámico.
- b. Sirven contenido estático.
- c. También pueden gestionar el correo electrónico.



Lee el enunciado e indica la opción correcta:

El elemento que actúa como user-agent es:

- a. El editor de código.
- b. La base de datos.
- c. El navegador web.

RESUMEN

En esta unidad hemos visto de forma muy genérica y sin entrar en peculiaridades, todos aquellos **elementos que rodean el desarrollo de aplicaciones web**. Ciertamente nos hemos centrado en la parte del servidor, pero sin olvidar que el ciclo completo empieza cuando un cliente hace una petición a dicho servidor.

También hemos visto la **evolución de la generación dinámica de páginas web**, primero con CGI y más recientemente con lenguajes de script como PHP. Paralelamente hemos dibujado la diferencia entre servidores web y servidores de aplicaciones.

Finalmente nos hemos centrado en todas aquellas **herramientas que rodean el desarrollo web**: editores, IDEs, depuradores de código, etc. No hemos ahondado, pero ya tenemos las herramientas listas para empezar con la segunda unidad didáctica.

RECURSOS PARA AMPLIAR



PÁGINAS WEB

- Tutorial sobre cómo instalar y ejecutar Xdebug: <https://platzi.com/tutoriales/1462-php-avanzado/7795-como-depurar-aplicaciones-en-php/> [Consulta noviembre 2022]
- Tutorial sobre HTML de la plataforma w3schools: <https://www.w3schools.com/html/default.asp> [Consulta noviembre 2022].
- Tutorial sobre PHP de la plataforma w3schools: <https://www.w3schools.com/php/default.asp> [Consulta noviembre 2022]
- Vídeo: 5 mejores editores web: <https://www.youtube.com/watch?v=AJnhqf5IRC4> [Consulta noviembre 2022]



BIBLIOGRAFÍA



PÁGINAS WEB

- Blog de apachefriends.org: <https://www.apachefriends.org/blog.html> [Consulta noviembre 2022].
- Herramienta X-Debug: <https://xdebug.org/> [Consulta noviembre 2022].
- PSR - PHP Standard Recommendations: <https://www.php-fig.org/> [Consulta noviembre 2022].
- Referencia oficial de PHP: <https://www.php.net/> [Consulta noviembre 2022].



GLOSARIO

- **CGI:** Common Gateway Interface, interfaz de entrada común. Es una tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web.
- **CSS:** Cascading Style Sheets, hojas de estilo en cascada.
- **HTML:** HyperText Markup Language, lenguaje de etiquetas de hipertexto.
- **HTTP:** Hypertext Transfer Protocol, protocolo de transferencia de hipertexto. Es el protocolo de comunicación que permite las transferencias de información a través de archivos (XML, HTML...) en la World Wide Web.
- **IDE:** Integrated Development Environment, entorno de desarrollo integrado.
- **IP:** Internet Protocol, protocolo de internet. Su función principal es el uso bidireccional en origen o destino de comunicación para transmitir datos mediante un protocolo no orientado a conexión.
- **OSI:** Open System Interconnection, sistema de interconexión abierto. Es un modelo de referencia para los protocolos de la red (no es una arquitectura de red), creado en el año 1980 por la Organización Internacional de Normalización (ISO).
- **TCP:** Transmission Control Protocol, protocolo de control de transmisión. Es un protocolo fundamental de Internet que está ubicado en la capa de transporte del modelo OSI.
- **URL:** Uniform Resource Locator, localizador de recursos uniforme. Es la dirección donde localizar recursos en la World Wide Web.

