Unidad 1.

# Plataformas de programación web en entorno servidor. Aplicaciones AMP.

Conceptos fundamentales:

* Ejecución de aplicaciones en entorno servidor, los componentes implicados.
* Diferencia entre páginas web estáticas.
* Ejecución de código en las páginas web.
* Principales tecnologías disponibles para el desarrollo de aplicaciones web.
* Creación de un entorno de programación PHP basado en Xampp e IDE de Apache Netbeans.
* Elementos básicos de programación en PHP (literales, constantes, variables, operadores, librería, programación de expresiones)

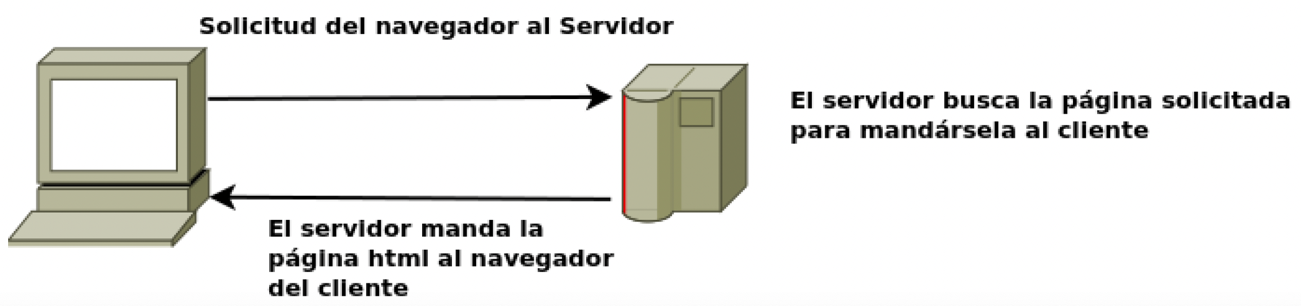
# 1.- Características de la programación web.

¿Seguro que ya sabes exactamente qué es una página web?

Cuando una página web se descarga a tu ordenador, su contenido define qué se debe mostrar en pantalla. Este contenido está programado en un lenguaje de marcado, formado por etiquetas HTML o XHTML. Las etiquetas indican el objetivo de cada una de las partes que la componen (encabezado, un párrafo, parte de una tabla, negrita, …).

Página bien estructurada, la información se mostrará según un estilo (hoja de estilos o CSS). La hoja de estilos se encuentra indicada en la página web y el navegador la descarga junto a ésta (tipo de letra, color, alineación, …)

Estos dos ficheros se descargan a tu ordenador desde un servidor web como respuesta a una petición.



Pasos:

1. Un ordenador solicita a un servidor web una página.
2. El servidor busca esa página.
3. Si el servidor encuentra esa página, la recupera.
4. Y por último se la envía al navegador para que muestre su contenido.

Comunicación cliente-servidor. El cliente es el que hace la petición e inicia la comunicación, y el servidor es el que recibe la petición y la atiende (navegador es el cliente web).

## 1.1.- Páginas web estáticas y dinámicas (I).

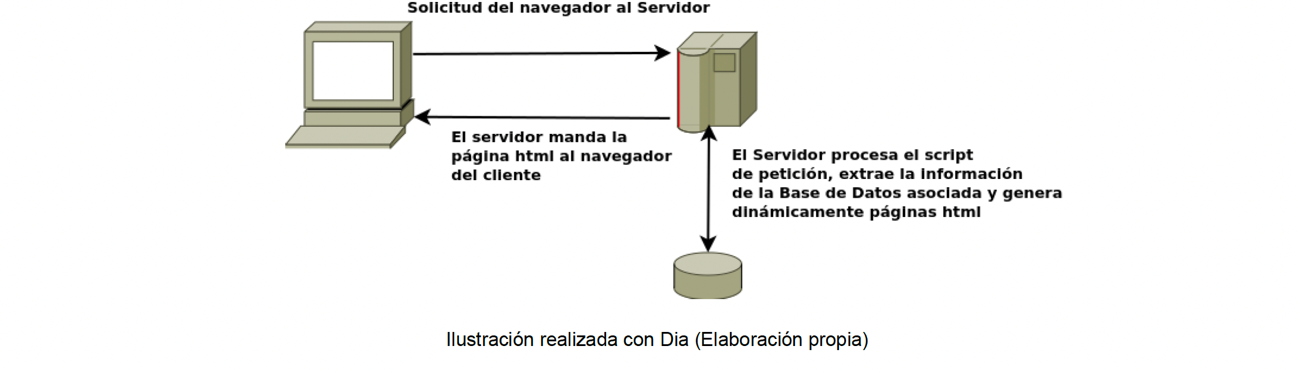
Páginas **web estáticas** se encuentran almacenadas en su forma definitiva, tal y como se crearon y su contenido no varía. Son útiles para mostrar una información concreta y mostrarán esa misma información cada vez que se carguen. La única forma en que pueden cambiar es si un programador la modifica y actualiza su contenido.

Páginas **web dinámicas**, su contenido cambia en función de diversas variables, según el tipo de navegador, el usuario con el que te has identificado o las acciones que has efectuado con anterioridad.

Webs dinámicas, dos tipos:

* Aquellas que incluyen código que ejecuta el navegador, normalmente en lenguaje JavaScript se incluye dentro del HTML y se descarga junto con la página. Este código puede incorporar múltiples funcionalidades que pueden ir desde mostrar animaciones, cambiar la apariencia y el contenido de la página. Este código puede incorporar múltiples funcionalidades, mostrar animaciones, cambiar la apariencia y el contenido de la página.
* Muchas páginas tienen extensiones como .php .asp .jsp .cgi .aspx, el HTML de estas páginas se forma como resultado de la ejecución de un programa y esa ejecución tiene lugar en el servidor web (aunque no necesariamente por ese mismo servidor)

Esquema página web dinámica:



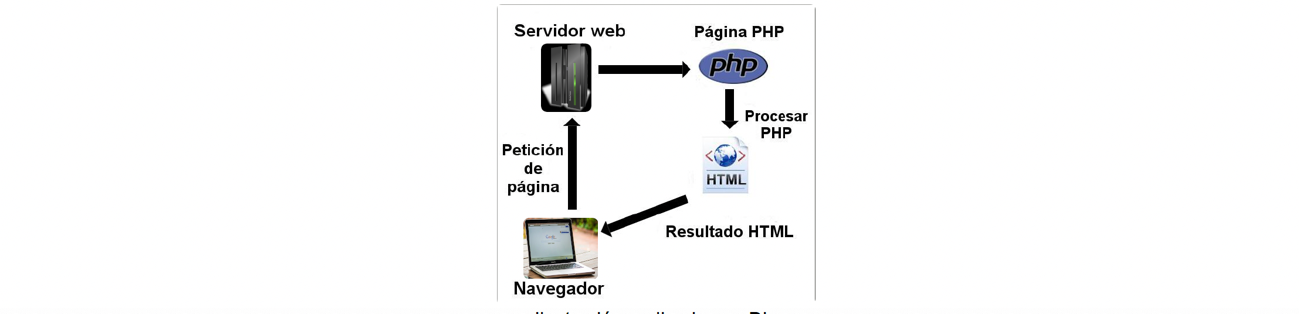
Pasos:

1. El cliente web (navegador) solicita a un servidor web una página web con extensión asp, php, jsp.
2. El servidor busca esa página y la recupera.
3. En el caso de que se trate de una página web dinámica, el servidor web contacta con el módulo responsable de ejecutar el código y se lo envía.
4. Como parte del proceso de ejecución puede ser necesario obtener información de algún repositorio o base de datos.
5. El resultado de la ejecución será una página en formato HTML.
6. El servidor web envía el resultado al navegador que lo proceso y muestra.

El navegador no envía esa misma página a todos los usuarios, sino que la genera de forma dinámica en función de quién sea el usuario que se conecte.

Página web PHP dinámica:

1. El cliente hace una petición al servidor.
2. El servidor tendrá un servidor web con un motor PHP.
3. El motor procesa los scripts de la página en cuestión y genera el código HTML.
4. El servidor manda la página HTML al cliente.



## Páginas web estáticas y dinámicas (II).

Las páginas web estáticas tienen también algunas ventajas:

* No es necesario saber programar para crear un sitio que utilice únicamente páginas web estáticas. Simplemente habría que conocer HTML, XHTML, CSS.
* Su contenido nunca varía y esto en algunos casos también puede suponer una ventaja (indexación de Google para la búsqueda de páginas web).
* Para que un servidor web pueda procesar una página web dinámica necesita ejecutar un programa. Esta ejecución la realiza un módulo concreto, que puede estar integrado en el servidor o ser independiente. Puede ser necesario consultar una base de datos como parte de la ejecución del programa.
* Estos recursos deben instalarse y mantenerse. Las páginas web estáticas sólo necesitan un servidor web que se comunique con tu navegador para enviártela.
* Las páginas web estáticas tienen limitaciones o desventajas; la actualización es de forma manual y para gran cantidad de contenido es prohibitivo en determinados sitios web.

## Aplicaciones web.

Las aplicaciones web emplean páginas web dinámicas para crear aplicaciones que se ejecuten en un servidor web y se muestren en un navegador. Aplicaciones web para múltiples tareas (cliente de correo una de las primeras).

Ventajas:

* No es necesario instalarlas en aquellos equipos en que se vayan a utilizar (solamente en el servidor).
* Es muy sencillo gestionarlas al estar instalada en un equipo (copias de seguridad, actualizarlas, corregir errores).
* Se pueden utilizar en todos aquellos sistemas que dispongan de un navegador web.
* Se puede utilizar desde cualquier lugar en el que dispongamos de conexión con el servidor.

Inconvenientes:

* La interfaz de usuario de las aplicaciones web es una página que se muestra en el navegador.
* Dependemos de una conexión con el servidor para poder utilizarlas.
* La información que se muestra en el navegador debe transmitirse desde el servidor (no recomendable para editores de vídeo).

Aplicaciones como Drupal, Joomla ofrecen dos partes bien diferenciadas y la mayoría de su contenido está almacenado en una base de datos:

* Una parte externa, que es el conjunto de páginas que ven la gran mayoría de usuarios que las usan. (usuarios externos)
* Una parte interna, que es otro conjunto de páginas dinámicas que utilizan las personas que producen el contenido y las que administran la aplicación web. (usuarios internos)

## 1.2.- Ejecución de código en el servidor y en el cliente.

Cuando tu navegador solicita a un servidor web una página, es posible que antes de enviártela haya tenido que ejecutar, por sí mismo o por delegación, algún programa para obtenerla. Ese programa es el que genera, en parte o en su totalidad, la página web que llega a tu equipo. El código se ejecuta en el entorno del servidor web.

También es posible que incluya algún programa o fragmentos de código que se deban ejecutar. Código en lenguaje JavaScript, se ejecuta en tu navegador (puede realizar diferentes acciones).

Estas dos tecnologías se complementan una con otra. Muchas de las aplicaciones web actuales utilizan estas dos tecnologías: la ejecución del código en el servidor y en el cliente.

Esta división es así porque el código que se ejecuta en el cliente web (en tu navegador) no tiene, o mejor dicho tradicionalmente no tenía, acceso a los datos que se almacenan en el servidor.

Sin embargo, desde hace unos años existe una técnica de desarrollo web conocida como **AJAX**, que nos posibilita realizar programas en los que el código JavaScript que se ejecuta en el navegador pueda comunicarse con un servidor de Internet para obtener información con la que, por ejemplo, modificar la página web actual.

## 2.- Tecnologías para programación web del lado del servidor.

Cuando programas una aplicación, utilizas un lenguaje de programación.

Los componentes principales con los que debes contar para ejecutar aplicaciones web en un servidor son los siguientes:

* Un servidor web para recibir las peticiones de los clientes web (navegador) y enviarles la página que solicitan. Servidores web como Apache, Nginx o IIS.
* El módulo encargado de ejecutar el código o programa y generar la página web resultante. Este módulo debe integrarse de alguna forma con el servidor web, y dependerá del lenguaje y tecnología que utilicemos para programar la aplicación web.
* Una aplicación de bbdd, que también será un servidor y que es normalmente utilizado. Mysql, MariaDb y SQLite.
* El lenguaje de programación que utilizarás para desarrollar las aplicaciones. PHP y JavaScript para NodeJS.

Es importante decidir cómo vas a organizar el código de la aplicación. Muchas de las arquitecturas que se usan en la programación de aplicaciones web te ayudan a estructurar el código de las aplicaciones en capas o niveles.

El motivo es que se puedan separar las funciones lógicas de la misma, de tal forma que sea posible ejecutar cada una en un servidor distinto (si es necesario).

En una aplicación podemos distinguir:

* Funciones de presentación, se encarga de dar formato a los datos para presentárselo al usuario final.
* Lógica, utiliza los datos para ejecutar un proceso y obtener un resultado.
* Persistencia, mantiene los datos almacenados de forma organizada.
* Acceso, que obtiene e introduce datos en el espacio de almacenamiento.

Aplicaciones de 3 capas:

* **Capa cliente**, es donde programarás todo lo relacionado con el interfaz de usuario (la parte visible de la aplicación con la que interactuará el usuario).
* **Capa intermedia** donde deberás programar la funcionalidad de tu aplicación.
* **Capa de acceso a datos,** que se encargará de almacenar la información de la aplicación en una base de datos y recuperarla cuando sea necesario.

## 2.1.- Arquitecturas y plataformas.

Qué arquitectura utilizar. Hoy en día:

* **JavaEE** (Enterprise Edition) antes J2EE. Lenguaje Java orientada a la programación de aplicaciones. Puede funcionar con distintos gestores de bases de datos, e incluye varias librerías y especificaciones para el desarrollo de aplicaciones de forma modular. (JSP, servlets, EJB)
* **AMP**. Son las siglas de Apache, MySQL y PHP/Perl/Python. Dependiendo del sistema operativo se utilizan las siglas LAMP, WAMP o MAMP. Se puede utilizar PostgreSQL en lugar de MySQL.

Todos los componentes de esta arquitectura son de código abierto (open source). Su gran ventaja es la gran comunidad que la soporta y la multitud de aplicaciones de código libre disponibles.

* **CGI/Perl: Perl,** un potente lenguaje de código libre creado originalmente para la administración de servidores y CGI un estándar para permitir al servidor web ejecutar programas genéricos, escritos en cualquier lenguaje (se puede utilizar C o Python) que devuelven páginas web como resultado de su ejecución. CGI es lento, Perl es un lenguaje muy potente con una amplia comunidad de usuarios y mucho código libre disponible.
* **ASP.Net** propuesta por Microsoft para el desarrollo de aplicaciones. Es la parte de la plataforma .Net destinada a la generación de páginas web dinámicas. Proviene de ASP.

El lenguaje de programación puede ser Visual Basic.Net o C#. La arquitectura utiliza el servidor web de Microsoft, IIS y puede obtener información de varios gestores de bbdd como, por ejemplo, Microsoft SQL Server.

Una de las mayores ventajas de la arquitectura .Net es que incluye todo lo necesario para el desarrollo y el despliegue de aplicaciones, pero de código propietario.

* **Node.js** Multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación ECMAScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google.

## 2.1.1.- Selección de una arquitectura de programación web.

Para tomar una decisión correcta, deberás considerar entre otros los siguientes puntos:

* ¿Qué tamaño tiene el proyecto?
* ¿Qué lenguajes de programación conozco? ¿Vale la pena el esfuerzo de aprender uno nuevo?
* ¿Voy a usar herramientas de código abierto o herramientas propietarias? ¿Cuál es el coste de utilizar soluciones comerciales?
* ¿Voy a programar la aplicación yo solo o formaré parte de un grupo de programadores?
* ¿Cuento con algún servidor web o gestor de base de datos disponible o puedo decidir libremente utilizar el que crea necesario?
* ¿Qué tipo de licencia voy a aplicar a la aplicación que desarrolle?

## 2.2.- Integración con el servidor web

La comunicación entre un cliente web o navegador y un servidor web se lleva a cabo gracias al protocolo HTTP. En el caso de las aplicaciones web, HTTP es el vínculo de unión entre el usuario y la aplicación en sí. Cualquier introducción de información que realice el usuario se transmite mediante una petición HTTP y el resultado que obtiene le llega por medio de una respuesta HTTP.

En el lado del servidor, estas peticiones son procesadas por el servidor web (también llamado servidor HTTP). Es por tanto el servidor web el encargado de decidir cómo procesar las peticiones que recibe.

La tecnología más antigua es CGI. CGI es un protocolo estándar que existe en muchas plataformas. Lo implementan la gran mayoría de servidores web. Define qué debe hacer el servidor web para delegar en un programa externo la generación de una página web. Esos programas externos se conocen como guiones CGI, independientemente del lenguaje en el que estén programados.

El principal problema de CGI es que cada vez que se ejecuta un guion CGI, el sistema operativo debe crear un nuevo proceso. Esto implica un mayor consumo de recursos y menor velocidad de ejecución. Existen algunas soluciones que aceleran la ejecución, como FastCGI y también otros métodos para ejecutar guiones en el entorno de un servidor web (módulo mod\_perl, mod\_php, mod\_python).

La arquitectura JavaEE ahora llamada **JakartaEE**, es más compleja. Dos opciones de ejecución:

* Servidores de aplicaciones, que implementan todas las tecnologías disponibles en JavaEE y contenedores de servlets, que soportan solo parte de la especificación.
* Existen varias implementaciones de servidores de aplicaciones JavaEE certificados. Las dos soluciones comerciales son IBM Websphere y BEA Weblogic. JBoss, Geronimo o Glassfish (código abierto).
* La mayoría de las ocasiones no es necesario utilizar un servidor de aplicaciones completo, especialmente si no utilizamos EJB en nuestras aplicaciones, sino que nos será suficiente un contenedor de servlest. Como Tomcat, la implementación por referencia de un contenedor de servlets y de código abierto.

La arquitectura ASP.Net utiliza el servidor IIS de Microsoft, que ya integra soporte en forma de módulos para manejar peticiones de páginas dinámicas ASP y ASP.Net. La utilidad de administración del servidor web incluye funciones de administración de aplicaciones web instaladas en el mismo.

## 2.3.- Lenguajes

Tres grandes grupos:

* **Lenguajes de guiones (scripting)**: los programas se ejecutan directamente a partir de su código fuente original. Se almacenan normalmente en un fichero de texto plano. Cuando el servidor web necesita ejecutar código programado en un lenguaje de guiones, le pasa la petición a un intérprete, que procesa las líneas del programa y genera como resultado una página web (ej: Perl, Python, PHP, ASP, ASP.Net).
* **Lenguajes compilados a código nativo**. Son aquellos en los que el código fuente se traduce a código binario, dependiente del procesador, antes de ser ejecutado. El servidor web almacena los programas en su modo binario, que ejecuta directamente cuando se le invoca. El método principal es CGI, utilizándolos podemos hacer que el servidor web ejecute código programado en cualquier lenguaje de propósito general como C.
* **Lenguajes compilados a código intermedio**. Son lenguajes en los que el código fuente original se traduce a un código intermedio, independiente del procesador, antes de ser ejecutado. Es la forma en la que se ejecutan por ejemplo las aplicaciones Java, y lo que hace que puedan ejecutarse en varias plataformas distintas.

Así operan los lenguajes de Java (Jakarta EE) y ASP.Net. En estas plataformas se utilizan un procedimiento de compilación JIT. Este término hace referencia a la forma en que se convierte el código intermedio a código binario para ser ejecutado por el procesador. Para acelerar la ejecución, el compilador puede traducir todo o parte del código intermedio a código nativo cuando se invoca a un programa. El código nativo obtenido suele almacenarse para ser utilizado de nuevo cuando sea necesario.

**Ventajas e inconvenientes:**

* Los lenguajes de guiones tienen la ventaja de que no es necesario traducir el código fuente original para ser ejecutados, lo que aumenta su portabilidad. Se puede realizar modificaciones al momento. Por el contrario, el proceso de interpretación ofrece un peor rendimiento que las otras alternativas.
* Los lenguajes compilados a código nativo son los de mayor velocidad de ejecución, pero tiene problemas en lo relativo a su integración con el servidor web. Son programas de propósito general que no están pensados para ejecutarse en el entorno de un servidor web. Por cada petición se debe ejecutar un nuevo proceso. Además, los programas no son portables entre distintas plataformas.
* Los lenguajes compilados a código intermedio ofrecen un equilibrio entre las dos opciones anteriores. Su rendimiento es muy bueno y pueden portarse entre distintas plataformas en las que exista una implementación de la arquitectura.

## 2.3.1.- Código embebido en el lenguaje de marcas.

Los guiones CGI son programas estándar, que se ejecutan por el sistema operativo, pero que generan como salida el código HTML de una página web. Por tanto, los guiones CGI deben contener, mezcladas dentro de su código, sentencias encargadas de generar la página web.

Principales formas de realizar páginas web dinámicas: **integrar las etiquetas HTML en el código de los programas.**

Un enfoque distinto consiste en **integrar el código del programa en medio de las etiquetas** HTML de la página web. De esta forma, el contenido que no varía de la página se puede introducir directamente en HTML, y el lenguaje de programación se utilizará para todo aquello que pueda variar de forma dinámica.

Esta metodología de programación es la que se emplea en los lenguajes de ASP, PHP y con las páginas JSP de Java EE.

Los **servlets de JavaEE** se diferencian de las páginas **JSP** en que los primeros son programas Java compilados y almacenados en el contenedor de servlets. Sin embargo, las páginas JSP contienen código Java embebido en lenguaje HTML y se almacenan de forma individual en el servidor web. La primera vez que se necesita una página JSP, se convierte a un servlet y éste se guarda para ser utilizado en posteriores llamadas a la misma página.

En la arquitectura **ASP.Net,** cada página se divide en dos ficheros: uno contiene las etiquetas HTML y otro el código en el lenguaje de programación utilizado. De esta forma se logra cierta independencia entre el aspecto de la aplicación y la gestión del contenido dinámico. A partir de esos ficheros se obtiene un código intermedio (MSIL en la terminología de la plataforma) que es lo que almacena el servidor.

## 2.3.2.- Programación web con Java.

Java es el lenguaje de programación más utilizado hoy en día. Es un lenguaje orientado a objetos, basado en la sintaxis de C y C++ y eliminando algunas características de éstos que daban lugar a errores de programación, como los punteros. Todo el código que escribas en Java debe pertenecer a una clase.

El código fuente se escribe en archivos con extensión .java. El compilador genera por cada clase un archivo .class. Para ejecutar una aplicación programada en Java necesitamos tener instalado un entorno de ejecución (JRE). Para crear aplicaciones en Java necesitamos el kit de desarrollo de Java (JDK), que incluye el compilador.

Como ya viste, existen básicamente dos tecnologías que te permiten programar páginas web dinámicas utilizando Java EE, ahora llamada Jakarta EE: servlets y JSP (páginas web que contienen instrucciones para añadir contenido de forma dinámica).

Aunque no es así en todos los casos, la mayoría de las implementaciones disponibles para JSP compilan cada página y generan un servlet a ejecutar. Este servlet a partir de la misma la primera vez que se va se almacena para ser usado en futuras peticiones.

Para realizar una página JSP, incluyendo el código en Java dentro de las etiquetas HTML utilizando los delimitadores **<% y %>**. O también utilizando el método **println** dentro de un servlet.

No hay nada que se pueda hacer con JSP que no pueda hacerse también con servlets. De hecho, las primeras se suelen convertir en servlets para ser ejecutadas.

El problema de utilizar servlets directamente es que, aunque son muy eficientes, son muy tediosos de programar puesto que hay que generar la salida en código HTML con gran cantidad de funciones como println. Este problema se resuelve fácilmente utilizando JSP, puesto que aprovecha la eficiencia del código Java, para generar el contenido dinámico, y la lógica de presentación se realiza con HTML normal.

De esta forma estas dos tecnologías se suelen combinar para crear aplicaciones web. Los servlets se encargan de procesar la información y obtener resultados, y las páginas JSP se encargan de la interfaz, incluyendo los resultados obtenidos por los servlets dentro de una página web.

## 3.- Entorno de programación PHP.

**Descripción de los elementos de un entorno de desarrollo PHP.**

Los programas necesarios son los siguientes:

* Un programa que interprete el código PHP.
* Una base de datos, que suele ser MySQL. Lo más habitual es que un desarrollo PHP use datos almacenados en una base de datos para construir el contenido dinámico de la página web.
* Un servidor web, normalmente Apache o Nginx para recibir la petición de un navegador y lanzar el proceso de ejecución de un script PHP. El proceso de ejecución consiste en interpretar el código PHP incrustado en el fichero con extensión .php que se solicita en la petición HTTP al servidor.
* Un IDE que apoye al programador en las tareas de desarrollo de los programas tanto en el momento de su escritura como en las pruebas y depuración de estos. Más adelante se listan los IDEs más utilizados en el desarrollo de aplicaciones PHP.

Los IDEs proporcionan a los programadores funcionalidades que les ayudan a desarrollar código PHP. Entre estas características se encuentran:

* Resaltado de texto. Muestra con distinto color o tipo de letra los diferentes elementos del lenguaje: sentencias, variables, comentarios, etc. También genera "indentado" automático para diferenciar de forma clara los distintos bloques de un programa.
* Completado automático. Detecta qué estás escribiendo y cuando es posible te muestra distintas opciones para completar el texto.
* Navegación en el código. Permite buscar de forma sencilla elementos dentro del texto, por ejemplo, definiciones de variables.
* Comprobación de errores al editar. Reconoce la sintaxis del lenguaje y revisa el código en busca de errores mientras lo escribes.
* Generación automática de código. Ciertas estructuras, como la que se utiliza para las clases, se repiten varias veces en un programa. La generación automática de código puede encargarse de crear la estructura básica, para que sólo tengas que rellenarla.
* Ejecución y depuración. Esta característica es una de las más útiles. El IDE se puede encargar de ejecutar un programa para poder probar su funcionamiento. Además, cuando algo no funciona, te permite depurarlo con herramientas como la ejecución paso a paso, el establecimiento de puntos de ruptura o la inspección del valor que almacenan las variables.
* Gestión de versiones. En conjunción con un sistema de control de versiones, el entorno de desarrollo te puede ayudar a guardar copias del estado del proyecto a lo largo del tiempo, para que si es necesario puedas revertir los cambios realizados.

IDEs más utilizados en la actualidad son:

* PHPStorm de JetBrains (requiere licencia, pero estudiantes y profesorado pueden solicitar una gratuita de un año).
* VisuaStudio Code de Microsoft (muy versátil, se le pueden instalar muchas extensiones para darle una gran funcionalidad en casi cualquier lenguaje de programación).
* SublimeText al igual que el anterior se le pueden instalar gran cantidad de extensiones, es software propietario, pero se puede usar para enseñanza.
* Clásicos como **NetBeans** y Eclipse. Ambos además ofrecen para la descarga versiones personalizadas del IDE que pueden ser usadas directamente para programar en un lenguaje determinado, sin necesidad de cambiar la configuración o instalar módulos.

**Entorno de desarrollo PHP 8.1 para el desarrollo del módulo.**

El material del curso ha sido diseñado para la versión de PHP 8.1 por lo que el uso de otras versiones de PHP podría provocar que algún ejercicio o programa no funcione correctamente.

Plataforma Windows 10:

* XAMPP versión 8.1.x
* Java JDK necesario para el funcionamiento de IDE Apache Netbeans. Versiones 11 y 17.
* Apache Netbeans entorno de desarrollo integrado, gratuito y de código abierto para el desarrollo de aplicaciones. Versión 13 o superior.

Las secciones de este manual cubren el proceso de instalación y configuración de estos paquetes SW.

## 3.1.- XAMPP

XAMPP es una distribución de Apache que incluye varios softwares libres. El nombre es un acrónimo compuesto por las iniciales de los programas que lo constituyen.

La instalación se llevará a cabo en el sistema operativo Windows 10.

Descargar el paquete de instalación y seguir los pasos de instalación.

## 3.2.- Apache Netbeans IDE.

Apache NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. Apache NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

A continuación, se presenta un tutorial para instalar Java JDK LTS 17.

Después instalaremos y configuraremos Apache Netbeans para ser utilizado como IDE de desarrollo de aplicaciones PHP.

**Configuración Apache NetBeans para PHP**

Apache Netbeans debe ser configurado para que se convierta en un IDE de desarrollo de PHP. Por un lado, se debe configurar la ruta al ejecutable que implementa la funcionalidad de intérprete PHP y por otra se debe instalar y configurar la librería **XDebug** utilizada para proporcionar la funcionalidad de depuración.

**Configuración ruta de intérprete PHP**

Desde la opción del menú principal de Tools/Herramientas seleccionamos la opción de Options/Opciones. La ventana de dialogo ofrece varias pestañas en su parte superior y nos dirigimos a la pestaña de PHP. En la ventana de configuración de PHP configuramos la entrada de PHP Interpreter con el ejecutable de PHP incluido en la distribución de XAMPP. El intérprete PHP está localizado dentro de la carpeta "php" de la distribución de xampp, C:\xampp\php. (Asegúrate que seleccionas el fichero php.exe y no php.ini)

**Configuración XDebug**

La función de depuración con NetBeans IDE en PHP se hace por medio de la librería XDebug. La instalación de NetBeans no incluye XDebug por defecto así que hay que instalarla aparte. Para descargar la versión correcta de la librería de depuración vamos a utilizar el "wizard" facilitado por la web de XDebug.

Una vez descargada la librería dinámica se renombra a php\_xdebug.dll y se copia en la carpeta C:\xampp\php\ext junto con el resto de las librerías de PHP instaladas con Xampp.

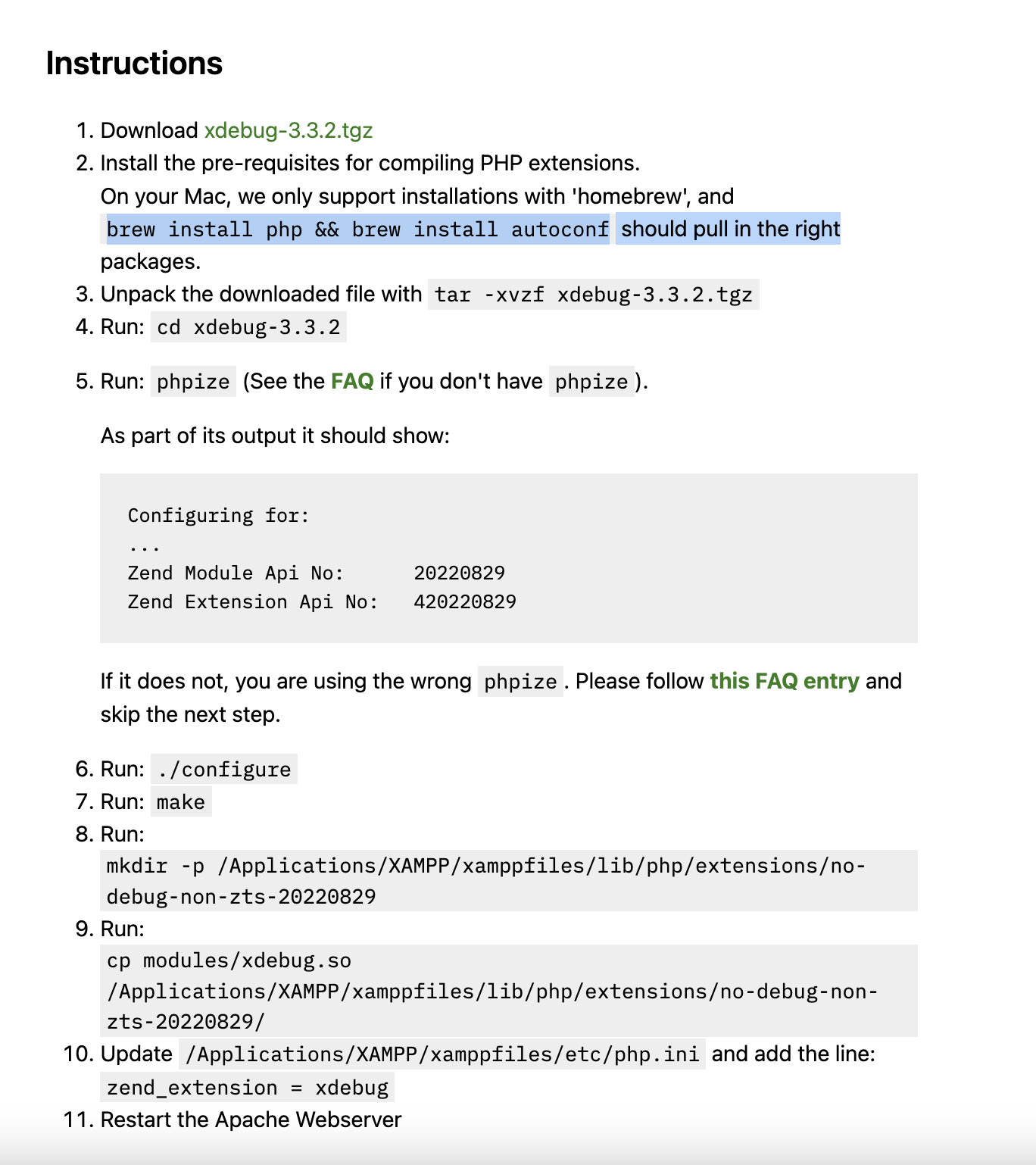
Ya solamente queda configurar el fichero php.ini situado en la carpeta php de la distribución de Xampp, C:\xampp\php. Se deben añadir los siguientes datos de configuración al final del fichero.

La ruta en el parámetro zend\_extension debe apuntar a la ubicación donde se copió la librería en el paso anterior.

Una vez realizados estos pasos podemos terminar de configurar la función de depuración en Apache NetBeans IDE accediendo a la ventana de configuración de PHP a la que accedemos desde el menú de Tools (Herramientas), seleccionando Options (Opciones).

Se recomienda la siguiente configuración de la sección de depuración (Debugging)

En Mac:



<https://www.youtube.com/watch?v=4RNmQjPJQo8>

## 4.- Conceptos básicos de programación web con PHP.

PHP es un lenguaje de guiones de propósito general, pero diseñado para el desarrollo de páginas web dinámicas utilizando código embebido dentro del lenguaje de marcas.

Similar a Java, C/C++, los delimitadores son **<?php ?>**

El código se ejecuta por un entorno de ejecución con el que se integra el servidor web (normalmente utilizando Apache con el módulo mod\_php). La configuración tanto del servidor web Apache, como de PHP se realiza por medio de ficheros de configuración (httpd.conf y php.ini)

Directivas **php.ini:**

* **Short\_open\_tags:** indica si se pueden utilizar en PHP los delimitadores cortos <? Y ?> (preferible no usarlos, problemas con XML). Por defecto en On.
* **Max\_execution\_time:** número máximo de segundos que podrá durar la ejecución de un script PHP (evita que el servidor se bloquee).
* **Display\_errors:** permite visualizar los errores que se produzcan en el código PHP. Para un entorno de producción a Off para un entorno de desarrollo a On. Por defecto en On.
* **Error\_reporting:** indica qué tipo de errores se mostrarán en el caso de que se produzcan.
* **File\_uploads:** indica si se pueden o no subir ficheros al servidor por HTTP.
* **Upload\_max\_filesize:** indica el límite máximo permitido para el tamaño de cada archivo.
* **Post\_max\_size:** complementa la directiva anterior, establece el tamaño máximo de un archivo subido por POST.

## 4.1.- Crear proyecto PHP “Hola Mundo”.

Se crea el primer proyecto con Apache Netbeans.

Seguir todos los pasos del Wizard para la configuración de nuestro proyecto en PHP.

Escribir el código para que pueda imprimir nuestro “Hola Mundo”.

Se puede ver las propiedades del proyecto y ejecutar el proyecto, icono verde o run.

Comprobar nuestro código mediante el depurador configurado previamente.

## 4.2.- Literales, constantes, variables y tipos de datos.

**Var\_dump()**: devuelve una cadena con el valor y el tipo del valor de lo que pasamos a la función como parámetro de entrada. Esta función nos va a permitir visualizar los valores manipulados en nuestro programa durante su ejecución.

**Literales**

Los literales en PHP se utilizan para representa valores concretos de tipo entero, decimal, cadena de caracteres y booleanos.

**Los literales enteros** pueden expresarse mediante notación decimal (base 10), hexadecimal (base 16), octal (base 8) o binaria (base 2), opcionalmente precedidos por un signo (- o +).

**Los literales en "coma flotante"** o float representan los valores decimales. Estos valores pueden expresarse mediante números decimales o notación científica.

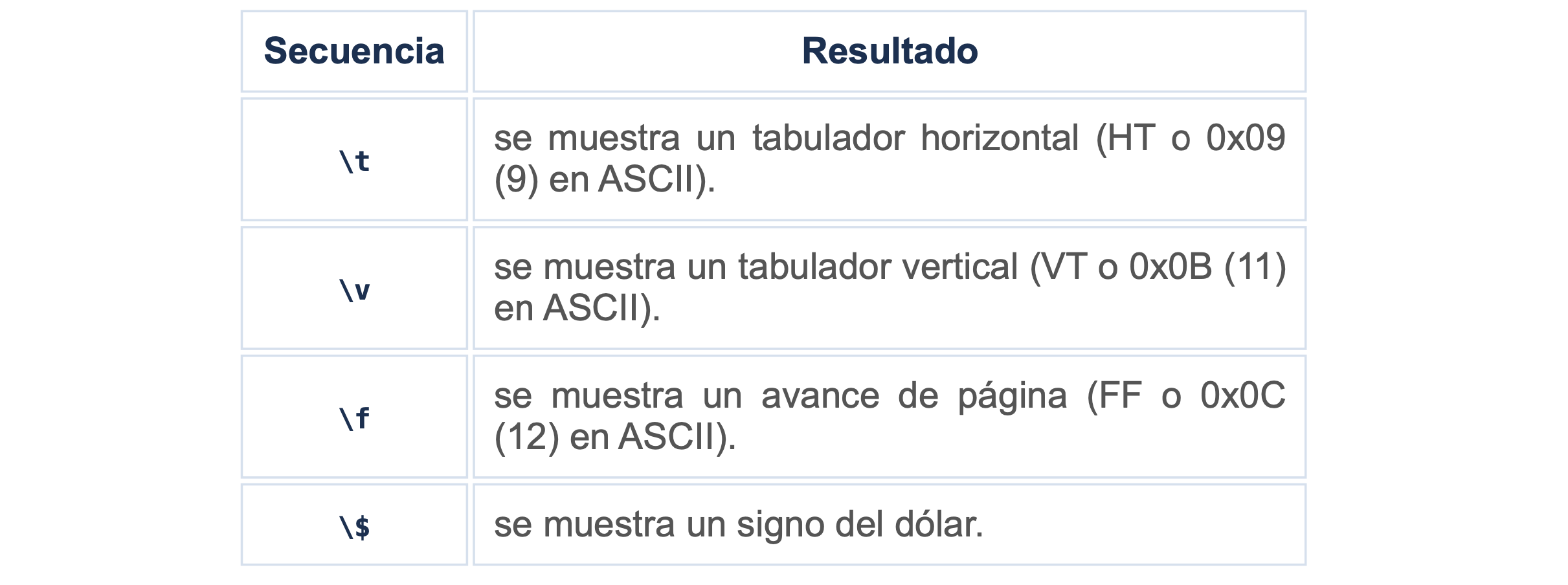
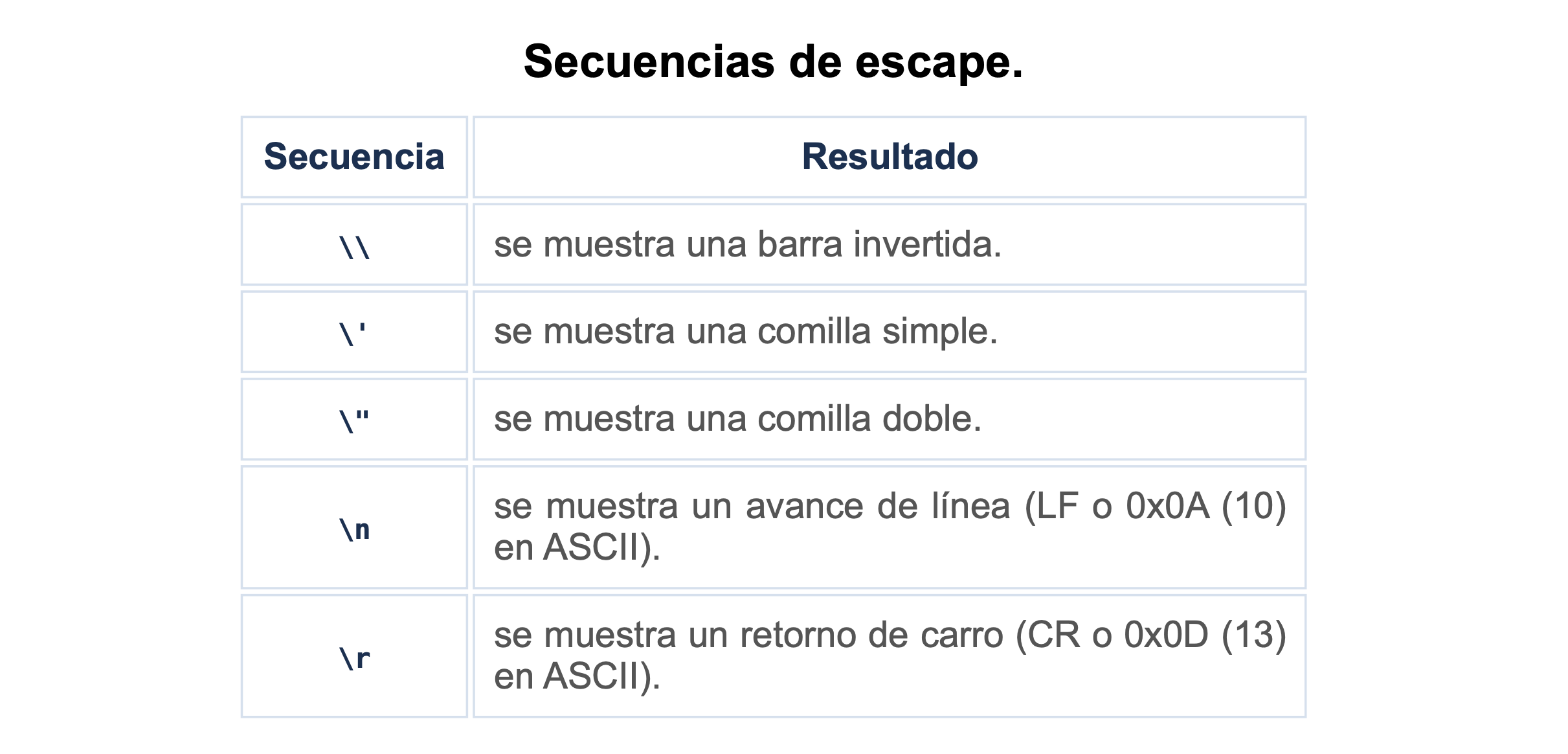
**Los literales cadena** de caracteres pueden crearse utilizando comilla simple, comilla doble y también con sintaxis HEREDOC y NOWDOC. La diferencia entre la comilla simple y la comilla doble radica en cómo se procesan los caracteres de la cadena antes de devolver su valor.

Cuando se usan comillas simples, sólo se realizan dos sustituciones dentro de la cadena:

1. Cuando se encuentra la secuencia de caracteres \', se muestra en la salida una comilla simple. Si no se hace así la comilla cerraría la comilla inicial y se produciría un error sintáctico.

2. Cuando se encuentra la secuencia \\, se muestra en la salida una barra invertida.

Sin embargo, las cadenas creadas con comillas dobles pueden incorporar los siguientes caracteres especiales:



**Los literales booleanos** son true y false y representan los valores booleanos Verdadero o true y Falso o False.

**Constantes**

Una constante es un identificador o un nombre que asociamos a un valor simple. Una vez definida la asociación entre el identificador y el valor no puede ser modificada.

**Variables**

Las variables en PHP siempre deben comenzar **por el signo $.** Los nombres de las variables deben comenzar por **letras o por el carácter \_,** y pueden contener también números. Sin embargo, al contrario que en muchos otros lenguajes, en PHP no es necesario declarar una variable ni especificarle un tipo (entero, cadena, …) concreto. Para empezar a usar una variable, simplemente asignarle un valor utilizando el operador =.

Dependiendo del valor que se le asigne, a la variable se le aplica un tipo de datos, que puede cambiar si cambia su contenido. Esto es, el tipo de la variable se decide en función del contexto en que se emplee.

**Tipos de datos**

Los tipos de datos simples en PHP son:

* **booleano** (boolean). Sus posibles valores son true y false. Además, cualquier número entero se considera como true, salvo el 0 que es false.
* **entero** (integer). Cualquier número sin decimales. Se pueden representar en formato decimal, octal (comenzando por un 0), o hexadecimal (comenzando por 0x).
* **real** (float). Cualquier número con decimales. Se pueden representar también en notación científica.
* **cadena** (string). Conjuntos de caracteres delimitados por comillas simples o dobles.
* **null**. Es un tipo de datos especial, que se usa para indicar que la variable no tiene valor.

Por otra parte, tenemos unas funciones para comprobar el tipo de dato como:

* **is\_bool():** Comprueba si una variable es de tipo booleano.
* **is\_float():** Comprueba si el tipo de una variable es float.
* **is\_numeric():** Comprueba si una variable es un número o un string numérico.
* **is\_string():** Comprueba si una variable es de tipo string.
* **is\_array():** Comprueba si una variable es un array.
* **is\_object():** Comprueba si una variable es un objeto.

Todas devuelven true o false.

También es posible forzar el cambio de tipo de un valor dado a otro tipo distinto. Este proceso se denomina **casting** y se puede realizar prefijando el nuevo tipo entre paréntesis antes del valor a modificar.

Los siguientes forzados de tipos están permitidos:

* **(int),** (integer) - forzado a integer
* **(bool),** (boolean) - forzado a boolean
* **(float),** (double), (real) - forzado a float
* **(string)** - forzado a string
* **(array)** - forzado a array
* **(object)** - forzado a object
* **(unset)** - forzado a NULL (PHP 5)

## 4.3.- Operadores y funciones de librería.

Las expresiones se suelen utilizar para obtener y almacenar valores calculados a partir de unos datos de entrada y para escribir condiciones lógicas en sentencias de control condicionales e iterativas.

Las expresiones se componen normalmente de uno o más operadores que se aplican a otros valores (o expresiones) llamados operandos.

**Operadores**

* **Operadores aritméticos:** cambio de signo -, suma +, resta -, multiplicación \*, división / y módulo %.
* **Operadores de asignación**. Además del operador de asignación=, existen operadores con los que realizar operaciones y asignaciones en un único paso (+=, -=, …). Entre éstos se incluyen operadores de pre y post incremento y decremento, ++ y --. Estos operadores incrementan o decrementan el valor del operando al que se aplican. Si se utilizan junto a una expresión de asignación, modifican el operando antes o después de la asignación en función de su posición (antes o después) con respecto al operando.
* **Operadores de comparación.** Además de los que nos podemos encontrar en otros lenguajes (>, >=, …), en PHP tenemos dos operadores para comprobar igualdad (==,===) y tres para comprobar diferencia (<>, != y !==). Los operadores <> y != son equivalentes. Comparan los valores de los operandos. El operador === devuelve verdadero (true) sólo si los operandos son del mismo tipo y además tienen el mismo valor. El operador !== devuelve verdadero (true) si los valores de los operandos son distintos o bien si éstos no son del mismo tipo.
* **Operadores lógicos o booleanos**. Tratan a los operandos como variables booleanas (true o false). Existen operadores para realizar un Y lógico (operadores and o &&), O lógico (operadores or o ||), No lógico (operador !) y O lógico exclusivo (operador xor).
* **Operadores bit a bit:** Tratan a los operandos como secuencias de bits. Pueden buscar bits que coinciden o no en ambos datos (&, |, ^ y ~), desplazar los bits del operando hacia la izquierda (<<) o hacia la derecha (>>).
* **Operador de concatenación de cadenas.** Este operador (.) recibe dos cadenas y devuelve la cadena resultado de su concatenación.
* **Operador ternario:** Recibe como operandos tres expresiones, una expresión booleana que se evalúa y si el resultado es verdadero devuelve el resultado de evaluar la segunda expresión, si el resultado es falso devuelve el resultado de evaluar la tercera expresión.
* **Fusión de null:** Este operador recibe un número variable de parámetros y devuelve el primer operando de izquierda a derecha que exista y no sea null. Devuelve null si no hay valores distintos de null. Es muy útil para establecer valores por defecto.
* **Expresión match:** La expresión match ramifica la evaluación basada en una comprobación de identidad de un valor. Una expresión match tiene una expresión de sujeto que se compara con múltiples alternativas. Se evaluará a un valor muy parecido al de las expresiones ternarias. La comparación es una comprobación de identidad (===) en lugar de una comprobación de igualdad débil (==).

Los operadores siempre quieren poder aplicarse sin generar errores que puedan entorpecer la generación de las páginas dinámicas. Hay casos en los que nuestro programa intenta aplicar operadores a valores que no concuerdan en tipo a lo que espera el operador.



**Funciones de librería**

El lenguaje PHP pone a disposición del programador una serie de librerías que facilitan la tarea de programar. Algunas de estas librerías contienen un repertorio de funciones que transforman datos en otros valores resultado.

**Funciones matemáticas**

A continuación, se presenta un conjunto de funciones numéricas de uso frecuente:

* **round**(int|float $num, int $precision = 0, int $mode = PHP\_ROUND\_HALF\_UP): float : redondea un número al entero más próximo.
* **floor**(int|float $num): float : redondea un número al entero inferior (es decir, devuelve la parte entera).
* **ceil**(int|float $num): float : redondea un número al entero superior.
* **pow**(mixed $num, mixed $exponent): int|float|object : calcula la potencia de un número.
* **sqrt**(float $num): float : calcula la raíz cuadrada de un número.
* **max**(mixed $value, mixed ...$values): mixed : devuelve el máximo de los valores de entrada.
* **min**(mixed $value, mixed ...$values): mixed : devuelve el mínimo de los valores de entrada.
* **mt\_rand**(int $min, int $max): int : genera un numero aleatorio entre los valores min y max.

También es útil conocer cómo formatear un número para que se imprima de la manera adecuada. Para ello se utiliza la siguiente función:

**number\_format(float $num, int $decimals = 0, ?string $decimal\_separator = ".", ? string $thousands\_separator = "," ): string**

La función requiere dos o cuatro argumentos:

* el primer argumento es el número a formatear.
* el segundo argumento es el número de decimales a mostrar (el número se redondea o trunca dependiendo de la longitud del número, compárese el segundo y el cuarto ejemplo de los ejemplos anteriores).
* el tercer argumento es el carácter a utilizar como separador de la parte entera de la decimal.
* el cuarto argumento es el carácter a utilizar como separador de miles.

La función devuelve el número formateado. Si sólo se utilizan dos argumentos, se utiliza el punto como separador de parte entera y decimal y la coma como separador de miles (notación inglesa).

**Funciones de cadenas de caracteres**

Del mismo modo, PHP proporciona un gran número de funciones de cadenas de caracteres muy útiles para el programador. A continuación, se presentan las funciones de uso más frecuente:

* **ltrim**(string $string, string $characters = " \n\r\t\v\x00"): string : Elimina los caracteres en blanco que se encuentren al principio de la cadena. Lo mismo hace rtrim por la derecha.
* **str\_contains**(string $haystack, string $needle): bool : Determina si una cadena contiene otra cadena dada.
* **str\_repeat**(string $string, int $times): string : Repite una cadena dada.
* **str\_replace**( array|string $search, array|string $replace, string|array $subject, int &$count = null ): string|array : Reemplaza una cadena por otra dada.
* **strcmp**(string $string1, string $string2): int : Compara dos cadenas de entrada.
* **strpos**(string $haystack, string $needle, int $offset = 0): int|false : Encuentra la posición de la primera ocurrencia de una cadena en otra dada.
* **strlen**(string $string): int : Devuelve la longitud de una cadena.
* **strtolower**(string $string): string : Convierte una cadena a minúsculas.
* **strupper** convierte la cadena a mayúsculas.
* **substr**(string $string, int $offset, ?int $length = null): string : Devuelve una parte de la cadena

## 4.4.- Evaluación de expresiones.

Las expresiones permiten crear fórmulas que, cuando se evalúan, transforman los datos de entrada en un resultado que puede ser mostrado en la página web dinámica o utilizado por las sentencias de control de nuestro algoritmo.

Se pueden crear expresiones con distintos tipos de elementos, pero lo más habitual es que contengan literales, variables y operadores de distintos tipos.

La asignación en PHP se realiza por medio del operador asignación que conlleva el efecto secundario de almacenar en la variable de la izquierda el valor obtenido en la derecha. En este caso el operador tiene asociatividad derecha, lo que significa que si tengo varias asignaciones en la expresión se ejecutan de derecha a izquierda.

El proceso de evaluación también puede hacer uso de un recurso llamado cortocircuito **(short-circuit)** por el que un operador no necesite conocer el valor de todos sus operandos para devolver un resultado.

**El operador ternario** (?:) funciona de manera parecida ya que dependiendo del resultado de la evaluación de la expresión booleana se evaluará solamente la segunda o la tercera expresión.

Este operador tiene una variante que consiste en omitir el tercer operando de tal manera que si el resultado de evaluar la expresión resulta un valor que corresponde a verdadero el operador devuelve dicho valor.

El operador de **fusión de nulo** (??) el comportamiento es parecido al operador ternario y solo se evalúan operandos hasta llegar al primero no nulo empezando por la izquierda.

La expresión **match** es otro tipo de operador que puede sustituir a la sentencia de control switch ya que funciona de manera similar. Dado un valor de entrada se proporcionan un número de expresiones asociadas a valores que se comparan con el primer valor. Si alguna de las comparaciones tiene éxito se devuelve el valor resultante de la evaluación de la expresión asociada. Es importante resaltar que la comparación de valores se realiza con el operador (===).

**Fin**.