**TEMA 1: Desarrollo Web en Entorno Servidor**

**1. Funcionamiento de internet**

Pasos del funcionamiento de internet:

1. El usuario abre un navegador y escribe una URL
2. El navegador descompone la URL en varias partes: Protocolo, servidor, puerto, ruta y recurso. El servidor será la dirección de la página (Ej. [www.google.es](http://www.google.es))
3. Otro protocolo convertirá la petición en una IP
4. El servidor recogerá la petición de cliente, la analizará y devolverá una respuesta
5. Dicha respuesta se presentará en el navegador

**1.1. Protocolos**

Un protocolo es un sistema de reglas que permite a dos entidades comunicarse entre si mediante la transmisión de información.

Define por la sintaxis, semántica y sincronización estándar, así como la posible recuperación de errores.

Estos protocolos pueden implementarse por software y/o hardware.

**1.2. Protocolo HTTP**

El HTTP (protocolo de transferencia de hipertexto) es el lenguaje común entre clientes y servidores, dando lugar a la web moderna.

Es un cliente del tipo cliente-servidor que permite al navegador comunicarse con un servidor de aplicaciones y devolver como resultado una página web.

Se trata de un protocolo muy inseguro, ya que cualquiera puede visualizar y capturar los mensajes que estamos transmitiendo. Para transacciones que requieran más seguridad se utiliza HTTPS

**1.3. Cliente-servidor**

Cliente: Realizan consultas, visitan páginas y solicitar información. Hoy en día no se limitan solo a los PCs. También pueden ser portátiles, tablet, móvil, smartwatch, coche…

Servidor: Reciben las peticiones de los clientes y analiza las variables para dar una respuesta: Quien manda la petición, desde donde, con qué navegador… La respuesta suele presentarse en formato web.

**2. PHP**

PHP (Hypertext Preprocessor) es un lenguaje de scripting de propósito general y código abierto pensado para el desarrollo web y que puede embeberse en páginas HTML.

Su sintaxis es similar a C, Java y Perl. Su función es crear rápidamente webs dinámicas. Fue creado en 1995 por Rasmus Lerdorf.

**2.1 Versiones de PHP**

PHP se caracteriza por la estabilidad de sus versiones. Además, siempre intenta ser retrocompatible con las anteriores, lo que ha frenado su desarrollo. Hasta la versión 4, la estructura del código PHP era muy caótica.

A partir de la versión 5 incorpora orientación a objetos, surgiendo frameworks que toman como referencia PHP. La versión 6 no se llegó a publicar por problemas con Unicode, saltando directamente a la 7.

**2.2. Frameworks**

Un framework permite configurar determinados problemas ya desarrollados, como trabajo con BBDD, Seguridad, Usuarios…

**3. El entorno de trabajo**

* Control de versiones: Permiten mantener el código guardado, realizar un seguimiento de versiones y planificar las ramas de trabajo
* Gestión de equipos: Permite la planificación de tareas, gestión de recursos y gestión del tiempo.

**4. Navegador**

Un navegador web permite al usuario mostrar contenido de aplicaciones web alojadas en internet mediante el protocolo HTTP.

Nos permitirá visualizar el resultado final de nuestro código programado. Es imprescindible, pero no es nuestro objetivo como en Desarrollo de Cliente.

**5. IDE de desarrollo**

Algunos de los IDEs más interesantes al ser ligeros, flexibles y hackeables son Sublime y Atom.io

**5.1 Tipos de instalaciones**

Sistemas en la nube:

* Platform as a Service (Paas): Nos permite disfrutar de recursos virtualizados en la nube como disco duro, RAM, procesador… Sobre esa base instalaremos un sistema operativo, un servidor (Apache) y un gestor de BBDD (MySQL)
* Software as a Service (SaaS): No solo nos permite disfrutar de recursos virtualizados, sino que ya contiene un software instalado. Con esto no tendremos que preocuparnos de la instalación y mantenimiento del sistema.

Instalaciones locales: Instalando software empaquetado como XAMPP, tendremos todos los recursos necesarios para el desarrollo en local.

Ventajas:

* Fácil de instalar y configurar.
* Control total del sistema.
* Fácil de mantener y realizar backups.

Desventajas:

* No apto para trabajo colaborativo.
* No apto para trabajar en producción.

**5.2 XAMPP**

Un archivo PHP es una página construida en HTML que contiene scripts en PHP. En XAMPP se guardarán en la carpeta htdocs.

**5.3 Editores de código**

Sublime Text, Notepad++, Atom, Adobe Brackets…

**6. Git y GitHub**

Hoy en día es crucial poder compartir código, dividir tareas, trabajar de forma distribuida y mantenerlo. De eso se encargan los controles de versiones como Git.

Git puede descargarse desde su web oficial en Windows o utilizando el comando apt-get en Ubuntu.

GitHub y BitBucket son algunas de las opciones que tenemos para alojar un entorno colaborativo en la nube.

Para unir nuestro trabajo en local a la nube:

* En Windows hacemos clic derecho sobre una carpeta y elegimos “Git Bash Here”
* Inicializamos el repositorio mediante *git init*
* Añadimos los ficheros mediante *git add -A*
* Añadimos el contenido mediante *git commit -m “mensaje”*
* Creamos el repositorio en GitHub si no está creado ya
* Copiamos el repositorio mediante *git remote add repositorio*
* Añadimos el repositorio mediante *git remote -v*
* Subimos el repositorio mediante *git push repositorio master*
* Para eliminar el repositorio remoto usamos *git remote rm repositorio*
* Para descargar el repositorio usamos *git clone* [*https://github.com/user/repositorio*](https://github.com/user/repositorio)

**7. Programando en PHP**

Todo el código php se escribe dentro de la etiqueta *<?php ?>*

*echo* permite enviar al navegador código HTML.

**7.1 Comentarios**

Los comentarios en PHP pueden introducirse con //, # y /\* \*/

**7.2 Variables y constantes**

Para declarar una variable en PHP basta con poner su nombre precedido del signo $

Para declarar una constante, se utiliza la función define, cuyos parámetros serán el nombre de la variable en formato cadena y el valor de la misma.

**7.3 Operadores**

Aritméticos:

* + Suma
* - Resta
* \* Multiplicación
* / División

Lógicos:

* &&, and: y
* ||, or: ó
* !: Distinto
* xor: Exclusivo

Asignación:

* = Asigna
* += Suma y asigna
* -= Resta y asigna
* \*= Multiplicación y asigna
* /= División y asigna

Comparación:

* < Menor
* <= Menor o igual
* == Igual
* > Mayor
* >= Mayor o igual
* != Distinto
* === Estrictamente igual
* !== Estrictamente distinto

Incremento:

* ++$x Incremento en la línea
* $x++ Incremento tras la línea
* --$x Decremento en la línea
* $x-- Decremento tras la línea

Concatenación:

* .
* .=

**7.4 Sentencias de control**

La sentencia if permite ejecutar código mediante la evaluación de una condición. Puede usarse en conjunto con else if y else.

Se puede usar el shorthand del if mediante ?

La sentencia switch permite ejecutar código evaluando una serie de casos. Tras cada caso debe utilizarse break. Para un caso no contemplado puede utilizarse default.

**7.5 Sentencias repetitivas**

Ejecutan un mismo bloque de código según el caso:

* while: Lo ejecuta mientras se cumpla una condición.
* do/while: Lo ejecuta hasta que deje de cumplirse una condición.
* for: Lo ejecuta un número determinado de veces.
* foreach: Forma alternativa de realizar un for.

Las sentencias break y continue se utilizan para detener o continuar un bucle. Su uso se considera mala práctica.

**7.6 Arrays**

Los arrays permiten almacenar, recorrer y editar un conjunto de datos.

Un array asociativo es aquel que devuelve valores según el valor de una clave, no por su índice.

Un array multidimensional es aquel que a su vez contiene otros arrays.

Funciones:

* count: Devuelve el tamaño de un Array
* asort: Ordenar el array alfabéticamente por sus valores
* shuffle: Ordena el array de forma aleatoria
* array\_merge: Devuelve dos arrays en uno.

Los arrays se recorren a través de los bucles for y foreach. Para un array asociativo será necesario utilizar foreach.

También pueden visualizarse usando var\_dump o print\_r. La diferencia es que esta última devuelve un valor de retorno si se le pasa true en su segundo argumento, mientras que var\_dump no lo hace.

var\_export devuelve un código PHP válido