

ÉRICA PALOMINO

TEMA 2

# INTRODUCCIÓN A PHP

**[erica.palomino@escuelaartegranada.com](mailto:erica.palomino@escuelaartegranada.com)**

ESCUELAARTEGRANADA

# ¿QUÉ ES PHP?

**PHP** es un acrónimo recursivo. Significa "**PHP Hypertext Preprocessor**".

Es un lenguaje de programación de propósito general. De hecho, puedes programar con él cualquier aplicación de escritorio.

Sin embargo, y por cosas del azar, se empezó a usar para desarrollo web al comienzo de la web 2.0, y hoy en día se utiliza casi exclusivamente para ese propósito.

# ¿QUÉ ES PHP?

Cuando se usa en desarrollo web, PHP **aparece embebido** dentro de documentos **HTML**.

Igual que sucede con Javascript, pocos proyectos nuevos se desarrollan con PHP clásico. Lo normal es usar un framework que funciona por encima de PHP.

Por supuesto, cualquier desarrollador/a web debe conocer tanto PHP como el funcionamiento de los frameworks que corren sobre PHP.



# **INSTALACIÓN DE PHP**

# HERRAMIENTAS

Vamos a utilizar el servidor Xampp Server.

- **X**: funciona en cualquier SO.
- **A**: utiliza un servidor Apache.
- **M**: utiliza MariaDB como SGDB, en versiones antiguas se utilizaba MySQL.
- **P**: PHP como lenguaje de programación.
- **P**: también admite uso de Perl como lenguaje.

Es una herramienta de software gratuito y de código abierto.

Se puede descargar en <https://www.apachefriends.org/es/index.html>



**INICIAMOS  
XAMPP**

# HERRAMIENTAS

Los documentos que queramos mostrar deberán de estar guardados en la dirección: **Disco:/xampp/htdocs/**

En el panel de control tenemos un botón para acceder a los archivos del software.

**OJO:** PHP va incrustado dentro de etiquetas HTML y, aunque seguimos el mismo formato, los archivos tienen que tener **SIEMPRE** extensión ***.php***, **NO *.html***

# HERRAMIENTAS

## ¿Cómo funciona esto de HTML y PHP?

Cuando el servidor web encuentre un archivo con extensión .html, lo enviará al cliente sin mirar ni siquiera lo que hay en su interior.

En cambio, cuando el servidor web encuentre un archivo con extensión .php, lo abrirá y buscará las etiquetas **<?php ... ?>**, y ejecutará el código que haya dentro antes de enviar el resultado al cliente.



# HERRAMIENTAS

## ¿Cómo funciona esto de HTML y PHP?

Como veréis, PHP tiene una sintaxis muy parecida a JAVA, ofreciendo las mismas características básicas como las estructuras de control, funciones, etc.

Al igual que JAVA, **cada sentencia de código termina en punto y coma (;).**

De igual forma, los comentarios se especifican con **//** y los bloques de comentarios con **/\* \*/**



# **VAMOS A IMPLEMENTAR EL PRIMER PROGRAMA DE PHP**

# **EJERCICIO:**

**USANDO PHP PARA IMPRIMIR EL TEXTO, CREA  
UNA PÁGINA QUE MUESTRE EL MENSAJE  
"HOLA, SOY " Y VUESTRO NOMBRE. EL TEXTO  
TENDRÁ QUE ESTAR ALINEADO A LA  
IZQUIERDA, SER DE COLOR AZUL Y TENER  
TAMAÑO DE LETRA 12**



# **PHP: VARIABLES**

# VARIABLES

## IMPORTANTE: LAS VARIABLES EN PHP SON TIPADAS DÉBILES

Esto quiere decir que, a la hora de declararlas no hay que especificar el tipo de dato que van a almacenar. También implica que **una misma variable puede almacenar distintos tipos de datos en periodo de ejecucion.**

Esto tiene sus ventajas (menos duplicidad, aprovechamiento de memoria) y desventajas (tenemos que llevar un control exhaustivo de los datos que almacenan nuestras variables en todo momento).

# VARIABLES

Al no ser tipadas, tampoco hace falta declarar las variables previo a su uso. Éstas se crean en el primer momento en el que se usan.

Para especificar que lo que estoy usando es una variable, hay que utilizar el **símbolo \$ seguido de una letra o ( \_ )**.

**IMPORTANTE PHP distingue mayúsculas de minúsculas.**

Para asignar un dato, al igual que en JAVA, se usa el **operador =**.

# VARIABLES - CADENAS DE CARACTERES

Se pueden especificar con **comillas simples (')** o **dobles (")**.

**Con ':** No se evalúa nada de lo que vaya introducido en la cadena, tal cual está escrito, se muestra.

**Con ":** Permite introducir variables para que sean evaluadas al mostrar la cadena.



**VAMOS A  
MODIFICAR EL  
EJERCICIO  
ANTERIOR**





# **PHP: OPERADORES**

# OPERADORES

En PHP tenemos los mismo operadores básicos que en JAVA (+, -, /, %) junto con sus variantes de operación + asignación (+=, -=, etc).

También existe el operador de **concatenación (.)** que concatena varias variables en un solo string.

También tenemos **pre y post - incremento y decremento (++\$a, \$a--, etc).**

# OPERADORES

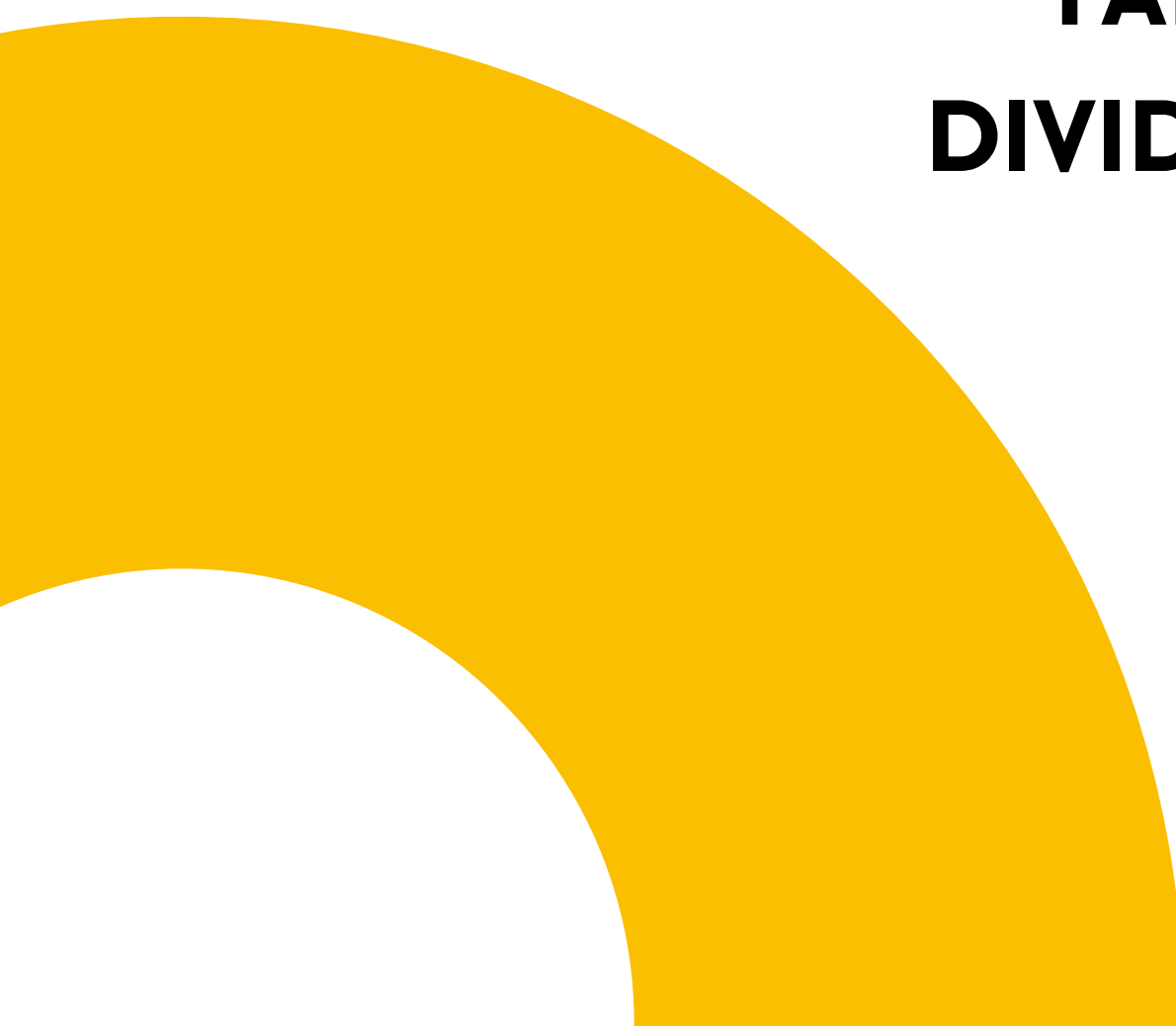

También tenemos los operadores lógicos que ya conocemos de JAVA (`==`, `!=`, `<`, `>`, `<=`) y (`&&`, `||`, `!`).

Aunque también tenemos disponibles algunos más raro que vale la pena destacar:

**?? (Coalescencia nulo)** : Asigna a una variable un valor u otro dependiendo de si está definida o no.

Ej: `$user = $nomUsu ?? "No iniciado";`

**<=> (Nave espacial)**: Devuelve -1 si `$a < $b`, 0 si `$a == $b` y 1 si `$a > $b`.



**EJERCICIO:**  
**CREA UNA PÁGINA QUE CONTENGA UNA**  
**TABLA QUE MUESTRE EL DIVISOR, EL**  
**DIVIDENDO, EL COCIENTE Y EL RESTO DE**  
**VARIAS DIVISIONES.**



# **PHP: FUNCIONES DE VARIABLES**

# FUNCIONES DE VARIABLES

**GetType(variable)** : devuelve el tipo de la variable.

**Settype (variable, tipo)**: asigna a la variable el tipo.

**isset(variable)**: true si la variable ha sido inicializada con un valor.

**Unset(variable)**: destruye la variable.

**Empty(variable)**: true si la variable tiene valor.

# FUNCIONES DE VARIABLES

**is\_int(variable):** true si la variable es int.

**is\_integer, is\_long, is\_float, is\_numeric, is\_real, is\_double, is\_bool, is\_array, is\_string, is\_object.**

**intval(var):** convierte el valor de la variable a entero.

**floatval(), strval().**

# PHP: ARRAYS



# ARRAYS POSICIONALES

Los arrays en PHP son colecciones de variables del mismo o de distinto tipo identificadas por un índice.

La declaración de un array se hace como si cada posición fuese una variable independiente.

Al igual que en JAVA, el acceso a cada posición se hace indicando dicha **posición entre [ ]**.

Si queremos acceder a la última posición del array solo hay que dejar los **[ ] en blanco**.

## **EJERCICIO:**

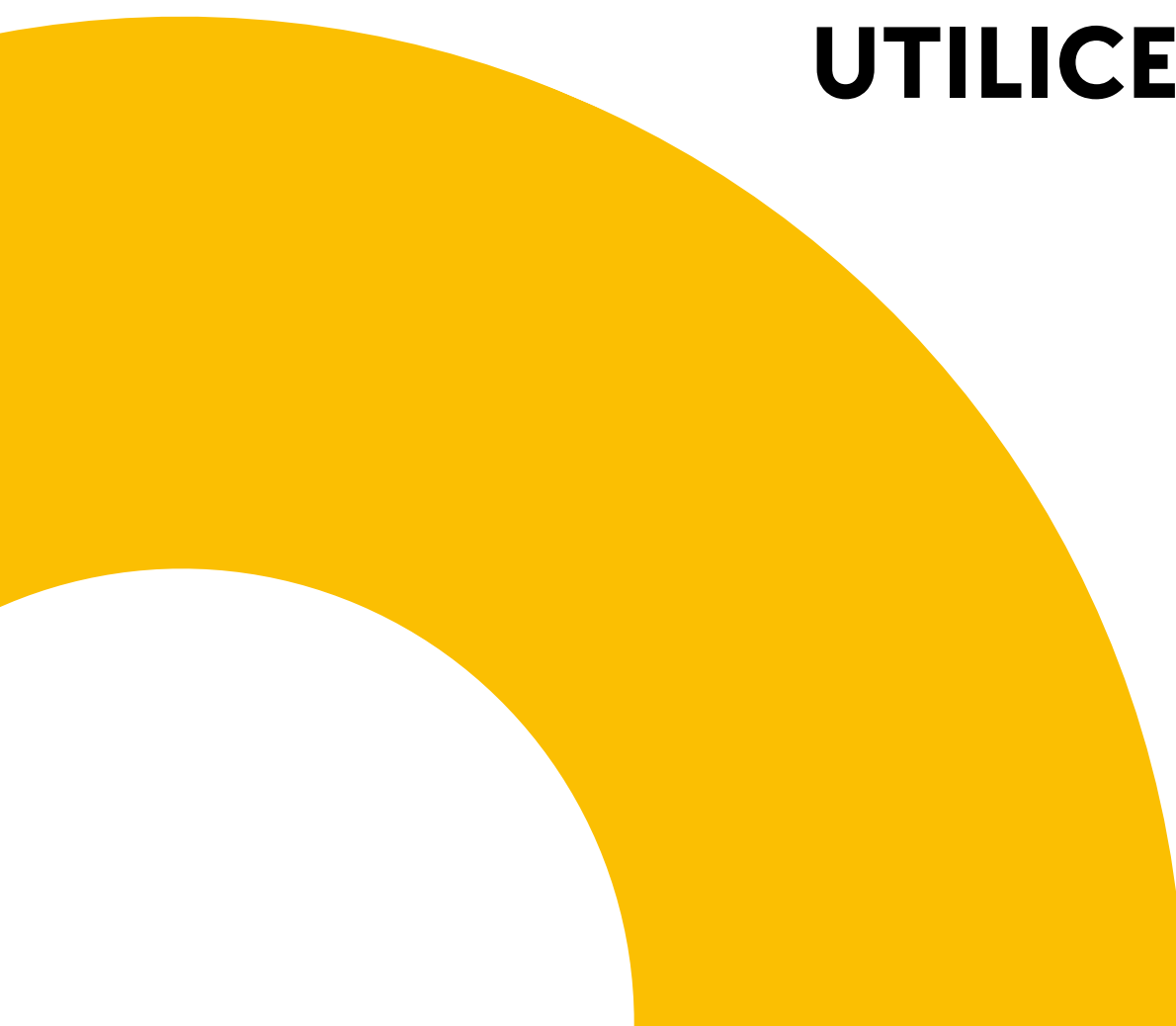

**CREA UNA PÁGINA QUE CONTENGA UNA  
TABLA QUE MUESTRE EL NOMBRE, EL APELLIDO  
1, EL APELLIDO 2, LA EDAD, LA ALTURA (M) Y EL  
PESO, ESTANDO ESTOS DATOS  
ALMACENADOS EN UN ARRAY.**

# ARRAYS ASOCIATIVOS

Lo índices de un array no tienen por qué ser números enteros:  
**pueden ser Strings.**

En este caso, el acceso a las distintas posiciones se hace de indicando entre **[]** **el nombre de la posición** a la que se quiere acceder.

**Ej: \$datosUsu["NOM"] = 'Manolo';  
\$datosUsu["EDAD"] = 20;**



**EJERCICIO:**  
**MODIFICA EL EJERCICIO ANTERIOR PARA QUE**  
**UTILICE UN ARRAY ASOCIATIVO EN LUGAR DE**  
**UNO POSICIONAL.**

# ARRAYS MÉTODOS Y ATRIBUTOS

- **count(\$a)**: devuelve el número de elementos del array \$a.
- **in\_array("valor", \$a)**: busca el elemento "valor" en el array \$a. Devuelve true o false.
- **unset(\$a[4])**: elimina un elemento (el 4, en este ejemplo) del array \$a.
- **next(\$a)**: devuelve el siguiente elemento de un array (el primero, si es la primera vez que se invoca).
- **prev(\$a)**: devuelve el elemento anterior de un array (el último si es la primera vez que se invoca).

# ARRAYS MÉTODOS Y ATRIBUTOS

- **array\_push(\$a, \$elemento)**: añade el \$elemento al final del array \$a.
- **\$elemento = array\_pop(\$a)**: elimina el último elemento del array \$a (y lo asigna a la variable \$elemento).
- **sort(\$a) y asort(\$a)**: ordena el array \$a. sort() se utiliza con arrays convencionales y asort() con arrays asociativos.



# **PHP: CONSTANTES**

# CONSTANTES

Las constantes, por convenio, suelen nombrarse en **MAYÚSCULAS y no se utiliza \$**. El propio PHP tiene muchas constantes predefinidas (todas en mayúsculas) de ámbito global.

Si queremos crear una constante utilizaremos la función **define(const, valor)**.

Si queremos comprobar si una constante está definida, utilizaremos **defined(const)**.



# CONSTANTES GLOBALES

**PHP\_VERSION.** Cadena que representa la versión del intérprete de PHP en uso.

**PHP\_OS.** Cadena con el nombre del sistema operativo en el que se está ejecutando el intérprete de PHP

**TRUE**

**FALSE**

**E\_ERROR.** Información sobre errores distintos a los interpretación del cual no es posible recuperarse.

**E\_PARSE.** Informa que el intérprete encontró una sintaxis inválida en el archivo de comandos. Finaliza la ejecución.

# CONSTANTES GLOBALES

**E\_NOTICE.** Informa que se produjo algo incorrecto que puede provenir o no de un error. A ejecución continúa.

**E\_WARNING.** Denota un error que no impide que continúe la ejecución.

**E\_ALL.** Conjunto con todos los errores que se han producido.

Algunas de estas constantes predefinidas empiezan y terminan por un doble subrayado, como `_LINE_` o `_FILE_`.

Estas constantes se llaman **constantes mágicas** y nos van a resultar muy útiles más adelante