PROGRAMACIÓN WEB EN ENTORNO SERVIDOR UD-6

FUNCIONES

ÍNDICE

[INTRODUCCIÓN 3](#_Toc146125486)

[STRING Y ARRAYS 3](#_Toc146125487)

[ARRAYS CON CLAVE 4](#_Toc146125488)

[ARRAYS MULTIDIMENSIONALES 4](#_Toc146125489)

[RECORRIDO CON FOREACH 4](#_Toc146125490)

[OPERACIONES DE COMPARACIÓN 4](#_Toc146125491)

# DEFINICIÓN

Cuando utilizamos en múltiples ocasiones un mismo fragmento de código debemos usar funciones (functions). Entre sus ventajas veremos que nos ayuda a que los ficheros tengan un menor tamaño y sea más fácil de mantener.

Es recomendable crear las funciones al principio del código para no tener problemas en las llamadas.

Al no declararse los tipos de datos, los parámetros de las funciones no tienen tipo ni se indica el tipo de dato que devuelven. El paso de parámetros se realiza por valor, es decir, se realiza una copia de la variable. Los parámetros pueden tener valores por defecto, en tal caso irán al final.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

# DECLARACIÓN DE TIPOS EXPLÍCITA

Dentro de una función se pueden especificar los tipos de los parámetros, tanto de entrada como de salida.



Podemos obligar a que el tipado sea esplícito poniendo al principio del archivo declare(strict\_types=1); No puede ir encima ni la etiqueta <html>, solo puede ir <?php.

Si justo antes del tipo se pone el modificador ?, significará que puede tomar dicho tipo o el valor null.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Para comprobar y establecer el tipo de datos de una variable se puede usar la función , que devuelve una cadena de texto, que puede ser array, boolean, double, integer, object, string, null, resource o unknown type.

También podemos comprobar si la variable es de un tipo concreto utilizando una de las siguientes funciones: is\_array(), is\_bool(), is\_float(),is\_integer(), is\_null(), is\_numeric(), is\_object(), is\_resource(), is\_scalar() e is\_string(). Devuelven true si la variable es del tipo indicado.

Análogamente, para establecer el tipo de una variable utilizamos la función settype pasándole como parámetros la variable a convertir, y una de las siguientes cadenas: boolean, integer, float, string, array, object o null. La función settype devuelve true si la conversión se realizó correctamente, o false en caso contrario.

Texto

Descripción generada automáticamente

# PASO DE PARÁMETROS POR REFERENCIA

Si queremos pasar un parámetro por referencia, en la declaración de la función, indicaremos los parámetros mediante el operador & para indicar la dirección de memoria de la variable. Se puede hacer tanto con parámetros tipados como sin tipar.

Texto

Descripción generada automáticamente

En estos casos es menos frecuente devolver un valor pero se puede hacer

# PARÁMETROS

Texto

Descripción generada automáticamente

# FUNCIONES CON PARÁMETROS VARIABLES

Podemos tener funciones donde en la declaración no indiquemos la cantidad de datos de entrada.

se puede utilizar el operador ... (variadics) el cual "disfraza" los parámetros como un array.

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

# ARGUMENTOS CON NOMBRE

Podemos pasar los argumentos con el nombre (además de por posición, como hemos hecho hasta ahora). Los argumentos con nombre se pasan poniendo el nombre como prefijo del parámetros separado por dos puntos: $resultado = funcion( arg1 : valor1, arg2 : valor2);

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

# FUNCIONES APUNTAS POR UNA VARIABLE

A una variable se le asigna el nombre de una función, de tal forma que si ponemos la variable con paréntesis, entenderemos que se está haciendo una llamada a la función que tenga el nombre que tiene la variable asignado.

Podemos asignar directamente la función a la variable. En este caso la función no tendría nombre y solo podría ser invocada desde la variable.

Texto

Descripción generada automáticamente

# BIBLIOTECAS Y LIBRERÍAS

Podemos agrupar un conjunto de funciones en un archivo, para permitir su reutilización.

Para importarlas tendremos que usar:

* include(archivo); / include\_once(archivo);
* require(archivo); / require\_once(archivo);

Si no encuentra el archivo, require lanza un error fatal, include lo ignora Las funciones \_once sólo se cargan una vez, si ya ha sido incluida previamente, no lo vuelve a hacer, evitando bucles. Su uso no es muy recomendable, ya que tarda más en hacer la importación.

Es recomendable poner las importaciones al principio del código.

Muchos programadores utilizan la doble extensión .inc.php para aquellos ficheros en lenguaje PHP cuyo destino es ser incluidos dentro de otros, y nunca han de ejecutarse por sí mismos.

El lenguaje ofrece un abanico de funciones ya definidas, agrupadas por su funcionalidad: <https://www.php.net/manual/es/funcref.php>

Mediante el uso de la instrucción include también podemos separar fragmentos de código PHP/HTML que queramos reutilizar en nuestros sitios web y crear un sistema muy sencillo de plantillas, donde, por ejemplo, incluiremos el encabezado base en un fichero.

Existen muchas librerías ya creadas que se pueden incorporar a nuestros proyectos. Se pueden consultar en el manual de php en la dirección <https://www.php.net/manual/es/funcref.php>