Funciones

Funciones

Definición

Generalmente las funciones se crean para reutilizar código y es una de las herramientas más potentes que tiene un lenguaje de programación. En algunos lenguajes de programación se les conoce como procedimientos, rutinas, subrutinas…

- Nos ayuda a mejorar la legibilidad y simplificación del código
- Reduce el tiempo de depuración.
- Una función se ejecuta al ser invocada o “llamada”.
- La definición de las funciones se puede hacer sin problema en cualquier punto del script de PHP, dentro de una clase, dentro de otra función.
- Y es accesible desde cualquier punto del script de PHP.

Funciones II

Declaración

- Para declarar una función debemos hacerlo mediante la palabra reservada ***"function"*** antes del nombre de la función.
- El nombre de la función sigue la misma regla que una variable, debe comenzar por una letra o guion bajo.
- Tras el nombre debemos indicar los argumentos que puede recibir la función desde su llamada. Estos argumentos están introducidos todos dentro de un paréntesis tras el nombre. Una función puede no recibir ningún argumento, por lo que el paréntesis se dejará vacío.
- Y además podemos definir el tipo de dato que se espera que devuelva la función. Si no devuelve ningún dato se considera vacía y es el formato que tiene PHP por defecto.

Funciones III

Declaración II

- Tras indicar el tipo de dato devuelto tendremos dentro de unas llaves el código que se ejecutará cuando se llame la función.
- La devolución se realiza mediante la palabra reservada ***"return"***.

```php
function funcion_vacia():void
{
    echo "Hola mundo";
}
```

```php
function funcion_retorno_sin_argumentos():DateTime
{
    return new DateTime( datetime: 'now');
}
```

```php
function funcion_argumentos(int $arg1, float $arg2):void
{
    echo "Se han recibido los argumentos:".$arg1."+".$arg2;
}
```

```php
function funcion_retorno(int $arg):int
{
    return $arg**$arg;
}
```

Uso

Invocación o llamada

Se pueden invocar o llamar desde cualquier punto del código de PHP o autoinvocarse (por ejemplo, constructores).

Se utilizar de la misma forma que una variable.

- Podemos asignar el valor retornado a otra variable.
- Podemos invocarla dentro de otra función.
- Realizando cálculos.
- Etc…

Parámetros / Argumentos

Es importante diferenciar entre:

- Parámetros → identificadores de la definición de la función.
- Argumentos → valores reales recibidos por la función.

También existe la posibilidad de que tengamos parámetros por defecto, garantizando que, si se omite algún argumento, el valor sea el indicado. Esto se haría asignando un valor en la introducción de los parámetros en la declaración de la función.

```php
function funcion_param_defec(int $arg=3)
{
    echo "Se ha recibido el argumento: ".$arg;
}
```

Si no se recibe el valor del argumento, y no existe un parámetro por defecto PHP genera un error por falta de datos.

Parámetros / Argumentos II

Array de argumentos

Es posible que en determinados momentos definamos una función sin argumentos predefinidos. Podemos pasarle argumentos igualmente y recogerlos si fuera necesario.

```php
<?php

foo(1,2,3);
1 usage
function foo()
{
    //Nos devuelve el número de argumentos recibidos
    $numargs = func_num_args();
    echo "Número de argumentos recibidos: $numargs <br>";
    if($numargs>1){
        //recogemos un array con todos los argumentos
        $arg = func_get_args();
        foreach ($arg as $key => $value){
            echo "El argumento recibido en la posición: $key es $value<br>";
        }
    }

}
```

Número de argumentos recibidos: 3
El argumento recibido en la posición: 0 es 1
El argumento recibido en la posición: 1 es 2
El argumento recibido en la posición: 2 es 3

Accesibilidad

Variables

Las variables son accesibles dentro del ámbito que se crean.

- Por si solas las funciones representan un ámbito propio
- Por lo tanto, las variables declaradas en su interior no son accesibles desde el exterior de la función.
- De forma que la información que se quiera trabajar dentro de la función debe pasarse por parámetros.

Accesibilidad

Variables globales

Si no trabajamos con orientación a objetos para poder acceder a las variables globales debemos usar la variable superglobal ➔ ***$GLOBALS*** o la palabra reservada ***global.***

```php
$a = 10;
$b = 5;
$c = 0;

1 usage
function multiplica()
{
    $GLOBALS['c'] = $GLOBALS['a'] * $GLOBALS['b'];
}
multiplica();
echo $c."<br>";

1 usage
function suma()
{
    //En este caso estamos definiendo que las variables a, b y c son las globales
    global $a ,$b, $c;
    $c = $a / $b;
}
suma();
echo $c;
```

```
50
2
```

Funciones Nativas

Las funciones nativas, son aquellas que vienen predefinidas por el propio lenguaje de programación y que nos permiten cubrir una serie de necesidades o rutinas habituales en la propia codificación.

Evitando inventar cada uno un método diferente y reutilizar el código de forma eficiente y eficaz.

https://www.php.net/manual/es/indexes.functions.php

Funciones Nativas II

Funciones con arrays

Existen una serie de funciones que nos permiten trabajar con arrays de una forma más rápida y eficiente.

https://www.php.net/manual/en/ref.array.php

Tenemos funciones que nos permiten extraer, eliminar o añadir valores del array. Algunos ejemplos:

- array_slice() ➔ extrae a un nuevo array un número de posiciones.
- array_shift() ➔ Elimina del array la primera posición.
- array_pop() ➔ Elimina del array la última posición.
- array_pad() ➔ Añade un número “n” de posiciones con el valor indicado.
- array_push() ➔ Añade el valor indicado en la última posición.

Funciones Nativas III

Funciones matemáticas

Existen una serie de funciones que nos permiten trabajar con funciones y operaciones matemáticas de una forma más rápida y eficiente.

https://www.php.net/manual/es/ref.math.php

Algunos ejemplos:

- abs — Valor absoluto
- acos — Arco coseno
- acosh — Arco coseno hiperbólico
- asin — Arco seno
- asinh — Arco seno hiperbólico
- atan2 — Arco tangente de dos variables
- atan — Arco tangente
- atanh — Arco tangente hiperbólica
- base_convert — Convertir un número entre bases arbitrarias
- bindec — Binario a decimal
- ceil — Redondear fracciones hacia arriba
- cos — Coseno
- cosh — Coseno hiperbólico
- decbin — Decimal a binario
- dechex — Decimal a hexadecimal
- decoct — Decimal a octal
- deg2rad — Convierte el número en grados a su equivalente en radianes
- exp — Calcula la exponencial de e

Funciones Nativas IV

Funciones de cadenas de caracteres (strings)

Existen una serie de funciones que nos permiten trabajar con cadenas de caracteres de una forma más rápida y eficiente.

https://www.php.net/manual/es/ref.strings.php

Algunos ejemplos:

- addcslashes — Escapa una cadena al estilo de C
- addslashes — Quote string with slashes
- bin2hex — Convierte datos binarios en su representación hexadecimal
- chop — Alias de rtrim
- chr — Devuelve un caracter específico
- chunk_split — Divide una cadena en trozos más pequeños
- convert_cyr_string — Convierte de un juego de caracteres cirílico a otro juego de caracteres cirílico
- convert_uudecode — Descodifica una cadena codificada mediante uuencode
- convert_uuencode — Codificar mediante uuencode una cadena
- count_chars — Devuelve información sobre los caracteres usados en una cadena
- crc32 — Calcula el polinomio crc32 de una cadena
- crypt — Hash de cadenas de un sólo sentido
- echo — Muestra una o más cadenas
- explode — Divide un string en varios string
- fprintf — Escribir una cadena con formato a una secuencia
- get_html_translation_table — Devuelve la tabla de traducción utilizada por htmlspecialchars y htmlentities

Referencias

Webgrafía

Todos los ejemplos se han sacado de la documentación oficial de PHP

- https://www.php.net/

Bibliografía

Existen muchos libros de PHP, pero se recomienda:

Lopez, A. (2023). *Learning PHP 7*. Packt Publishing.