Entorns de Desenvolupament

15. Funcions: arguments

Unió Europea
Fons social europeu
L'FSE inverteix en el teu futur

IES DR. LLUÍS SIMARRO

15. Funcions: arguments

Qualsevol informació pot ser passada a les funcions mitjançant la llista d'arguments o paràmetres, una llista d'expressions delimitades per comes. Els arguments són **avaluats d'esquerra a dreta**.

PHP admet el **pas d'arguments per valor** (el predeterminat), el **pas per referència**, i valors d'**arguments predeterminats**.

Pas d'arguments per valor

Per defecte, els **arguments** de les funcions són **passats per valor** (si el valor de l'argument dins de la funció canvia, aquest no canvia fora de la funció). "Per valor" vol dir que el canvi d'una dada d'un paràmetre no actualitza la dada de la variable que es va passar a la funció. Per exemple, quan invoquem una funció passant una variable com a paràmetre, tot i que canviem el valor del paràmetre dins de la funció, la variable original no es veu afectada per aquest canvi.

Exemple 1:

```
<?php
function per_valor($parametre1) {
   //modifiquem el valor de la variable de l'argument definit a la funció
   $parametre1 = "hola";
   echo $parametre1."<br/>"; //Imprimeix "hola"
}

$variable = "açò no canvia";
per_valor($variable);
echo $variable."<br/>"; //Imprimeix "açò no canvia"
?>
```

Aquest codi té per resultat:

hola

açò no canvia

Pas d'arguments per referència

Per permetre a una funció **modificar els seus arguments**, aquests han de **passar-se per referència**. Per fer que un argument a una funció siga sempre passat per referència cal anteposar al nom de l'argument el **símbol '&'** en la definició de la funció:

Exemple 2:

```
<?php //Pas de paràmetres d'una funció per referència
      function afegir_alguna_cosa (&$cadena) {
          $cadena = $cadena.' I alguna cosa més.';
      }
      $cad ='Això és una cadena,';
      afegir_alguna_cosa($cad);
      //el valor de la variable $cad ha sigut modificat en la funció
      echo $cad; // Imprimeix 'Això és una cadena, I alguna cosa més.'
?>
```

Valors d'arguments predeterminats

Una funció pot definir **valors predeterminats per als seus arguments**, tenint en compte que el valor per defecte ha de ser una expressió constant, no (per exemple) una variable, un membre d'una classe o una crida a una funció:

Exemple 3:

```php
<?php //Ús de paràmetres predeterminats en funcions
      function fer_cafe ($tipus = "capucino") {
            return "Fer una tassa de cafè $tipus. <br/>";
      }
      echo fer_cafe();
      echo fer_cafe(null);
      echo fer_cafe("espresso");
?>
```

El resultat de l'exemple seria:

Fer una tassa de cafè capucino.
Fer una tassa de cafè.
Fer una tassa de cafè espresso.

Es pot definir el valor per defecte de diversos arguments en una mateixa funció, per exemple, si tinguérem una funció amb la següent interfície:

```php
function funcio1($parametre1 = "David", $parametre2 = 3)
```

Per a la definició de funció anterior, `$parametre1` té com a valor per defecte `"david"`, mentre que `$parametre2` té `3` com a valor per defecte.

Si cridem a la funció sense indicar valors als paràmetres, aquests prendran els valors assignats per defecte:

```php
funcio1() // $parametre1 val "david" i $parametre2 val 3
```

Si cridem a la funció indicant un valor, aquest serà tingut en compte per al primer paràmetre:

```php
funcio1("hola") // $parametre1 val "hola" i $parametre2 val 3
```

S'ha de tindre en compte també que, en el cas que es vulga utilitzar paràmetres amb **valors per defecte, s'han de declarar obligatòriament al final en la llista de paràmetres de la interfície** (capçalera) de la funció. Els paràmetres que no tinguen valors per defecte es declararan a l'inici.

PHP també permet l'ús d'arrays i del tipus especial NULL com a valors predeterminats.

Cal observar que quan s'utilitzen **arguments predeterminats**, qualsevol d'ells hauria d'estar **a la dreta dels arguments no predeterminats**; si no, les coses no funcionaran com s'esperava. Analitzem el següent fragment de codi:

Exemple 4:

```
<?php // Ús incorrecte d'arguments predeterminats en una funció
      function fer_iogurt($tipus = "ensucrat", $sabor) {
            return "Fer un bol de iogurt $tipus de $sabor. <br/>";
      }
echo fer_iogurt("plàtan"); // No funcionarà com s'esperava
?>
```

El resultat de l'exemple seria:

Warning: Missing argument 2 for fer_iogurt()...

Ara, comparem l'exemple de dalt amb aquest:

Exemple 5:

```
<?php // Ús correcte d'arguments predeterminats en una funció
      function fer_iogurt($sabor, $tipus = "ensucrat") {
            return "Fer un bol de iogurt $tipus de $sabor.<br/>";
      }

      echo fer_iogurt("plàtan"); // Funciona com s'esperava
?>
```

El resultat de l'exemple seria:

Fer un bol de iogurt ensucrat de plàtan.

Declaracions de tipus

La declaració de tipus també es coneix com a '**Determinació de tipus**'. Les declaracions de tipus permeten a les funcions requerir que els paràmetres siguen de cert tipus durant una crida. Si el valor donat és d'un tipus incorrecte, es generarà un error. Aquesta funcionalitat, per a la majoria dels tipus, és **vàlida a partir de php 7**.

Per especificar una declaració de tipus, s'ha d'anteposar el nom del tipus al nom del paràmetre. Es pot fer que una declaració accepte valors `null` si el valor predeterminat del paràmetre s'estableix a `null`.

Exemple 6:

```
<?php
	function suma_enters(int $a, int $b) {
	 return $a + $b;
	}
	echo "Suma: ".sumaEnters(3,7)."<br/>"; //crida 1
	echo "Suma: ".sumaEnters("Tres",5)."<br/>"; //crida 2
?>
```

El resultat de les crides anteriors seria:
Suma: 10
Fatal error: Uncaught TypeError: sumaEnters(): Argument #1 ($a) must be of type int, string given

Valor de retorn

També a partir de la versió PHP 7, es pot indicar el tipus de dades de retorn de les funcions. Es fica a continuació de la capçalera de la funció, precedit pels dos punts (:). Els tipus que accepta PHP com a retorn de funcions són els següents:

- int
- float
- bool
- string
- array
- interfaces
- callable

A més, per indicar que PHP no faça conversions entre tipus per adequar els valors de les variables, s'ha d'establir el mode de treball **strict_types**.

Per **exemple**:

```
<?php
   declare(strict_types = 1); // els tipus esperats són els declarats
   function mostraInt(int $valor): int {
      return $valor;
   }
   echo "Valor: ".mostraInt(5);
?>
```

Resultat: 5

Si intentarem retornar un valor que no fora `int`, estant activat el mode `strict_types`, donaria un error.

Per **exemple**:

```
<?php
   declare(strict_types = 1); // els tipus esperats són els declarats
   function mostraInt2(int $valor): int {
      return $valor + 5.0;
   }
   echo "Resultat: ".mostraInt2(5);
?>
```

L'execució d'aquest codi mostrarà un missatge del tipus:

Fatal error: Uncaught TypeError: mostraInt2(): Return value must be of type int, float returned ...

Si comentarem la declaració `strict_types`, aquest error no es produiria i el que es mostraria en pantalla seria:

Resultat: 10