1 Curso POO PHP Xerar saída

1.1 Xerar saída

Existen varias formas incluír contido na páxina web a partir do resultado da execución de código PHP. A forma máis sinxela é usando **echo**, que amosa como saída o texto dos parámetros que recibe.

Outra posibilidade é **print**, que funciona de forma similar. A diferenza máis importante entre print e echo, é que print só pode recibir un parámetro e devolve sempre 1.

Tanto **print** como **echo** non son realmente funcións, senón elementos propios da linguaxe, polo que non é obrigatorio que poñas paréntese cando as utilices.

printf é outra opción para xerar unha saída dende PHP. Pode recibir varios parámetros, o primeiro dos cales é sempre unha cadea de texto que indica o formato que se ha de aplicar. Esa cadea debe conter un especificador de conversión por cada un dos demais parámetros que se lle pasen á función, e na mesma orde en que figuran na lista de parámetros.

Cada especificador de conversión vai precedido do caracter **%** e componse das seguintes partes:

- **signo** (opcional). Indica se se pon signo aos número negativos (por defecto) ou tamén aos positivos (indícase cun signo +).
- **recheo** (opcional). Indica que carácter se usará para axustar o tamaño dunha cadea. As opcións son o carácter 0 ou o carácter espazo (por defecto úsase o espazo).
- **aliñación** (opcional). Indica que tipo de aliñación se usará para xerar a saída: xustificación dereita (por defecto) ou esquerda (indícase co carácter -).
- **ancho** (opcional). Indica o mínimo número de carácteres de saída para un parámetro dado.
- **precisión** (opcional). Indica o número de díxitos decimais que se mostrarán para un número real. Escríbese como un díxito precedido por un punto.
- **tipo** (obrigatorio). Indica como se debe tratar o valor do parámetro correspondente.

Na seguinte táboa podes ver unha lista con todos os especificadores de tipo.

Especificador	Significado
b	o argumento trátase como un enteiro e preséntase como un número binario
c	o argumento trátase como un enteiro e preséntase como o carácter con ese valor ASCII
d	o argumento trátase como un enteiro e preséntase como un número decimal
u	o argumento trátase como un enteiro e preséntase como un número decimal sen signo
o	o argumento trátase como un enteiro e preséntase como un número octal
x	o argumento trátase como un enteiro e preséntase como un número hexadecimal (con minúsculas)
X	o argumento trátase como un enteiro e preséntase como un número hexadecimal (con maiúsculas)
f	o argumento trátase como un dobre e preséntase como un número de coma flotante (tendo en conta a localidade)
F	o argumento trátase como un dobre e preséntase como un número de coma flotante (sen ter en conta a localidade)
e	o argumento preséntase en notación científica, utilizando a e minúscula (por exemplo, 1.2e+3)
E	o argumento preséntase en notación científica, utilizando a e maiúscula(por exemplo, 1.2E+3)
g	emprégase a forma máis curta entre %f y %e
G	emprégase a forma máis curta entre %f y %E
s	o argumento trátase como unha cadea e preséntase como tal
%	amósase o carácter %. Non necesita argumento

Existe unha función similar a printf pero en vez de xerar unha saída coa cadea obtida, permite gardala nunha variable: **sprintf**.

Outras funcións que empregan para depurar mentres estamos programando son **print_r** e **var_dump**. Ambas amosan de xeito intelixente o contido do que se lles pasa como parámetro.

```
<?php
$cor = [
    "vermello"  => "#FF0000",
    "verde" => "#00FF00",
    "azul"   => "#0000FF"
```

```
];
var_dump($cor);
?>
```

O código anterior amosa en pantalla:

```
array(3) { ["vermello"]=> string(7) "#FF0000" ["verde"]=> string(7) "#00FF00" ["azul"]=> string(7) "#0000FF" }
```

--Víctor Lourido 14:42 25 jun 2013 (CEST)