Curso POO PHP Funcións

Funcións

En PHP temos a nosa disposición gran cantidade de funcións. Destas, moitas están incluídas no núcleo de PHP e pódense usar directamente. Outras moitas encóntranse dispoñibles en forma de extensións, e pódense incorporar á linguaxe cando se necesitan.

Coa distribución de PHP inclúense varias extensións. Para poder usar as funcións dunha extensión, tes que asegurarte de activala mediante o uso dunha directiva extension no ficheiro php.ini. Moitas outras extensións non se inclúen con PHP e antes de poder utilizalas tes que descargalas.

Para obter extensións para a linguaxe PHP podes utilizar **PECL**. PECL é un repositorio de extensións para PHP. Xunto con PHP inclúese un comando pecl que podes utilizar para instalar extensións de forma sinxela:

```
# pecl install nome_extensión
```

Para facer unha chamada a unha función, abonda con poñer o seu nome e unhas parénteses.

```
<?php
encabezado();
?>
```

Para crear as túas propias funcións, deberás usar a palabra **function**. En PHP non é necesario que definas unha función antes de utilizala, agás cando está condicionalmente definida (por exemplo, a definición da función realízase dentro dun bloque **if**). Cando unha función está definida dunha forma condicional as súas definicións deben ser procesadas antes de ser chamadas. Polo tanto, a definición da función debe estar antes de calquera chamada.

```
<?php
function encabezado() {
  echo "<head>"
  echo "<title>Aprendendo PHP</title>"
}
?>
```

Valor de retorno e argumentos

As funcións poden devolver un valor usando a sentenza **return**. Cando nunha función se encontra unha sentenza return, remata o seu procesamento e devolve o valor que se indica.

```
<?php
function encabezado() {
  return "<head>"."<title>Aprendendo PHP</title>";
}
?>
```

Tamén é posible facer que unha función devolva unha referencia a unha variable, e non o seu valor. Isto faise empregando o operador **&** tanto na definición da función (e non na sentencia return), como na chamada á función.

```
<?php
function &acumula($valor) {
  global $acumulador;
  $acumulador += $valor;
  return $acumulador;
}
?>

// $b pasa a ser unha referencia á variable global acumulador
$b =& acumula(3);
```

Os argumentos indícanse na definición da función como unha lista de variables separada por comas. Non se indica o tipo de cada argumento, ao igual que non se indica se a función vai devolver ou non un valor (se unha función non ten unha sentenza return, devolve **null** ao finalizar o seu procesamento).

```
<?php
function suma($a, $b) {
```

```
  return $a + $b;
}
?>
```

Ao definir a función, podes indicar valores por defecto para os argumentos, de forma que cando fagas unha chamada á función podes non indicar o valor dun argumento; neste caso tómase o valor por defecto indicado.

```
<?php
function suma($a, $b=0) {
  return $a + $b;
}
?>
```

Por defecto en PHP os argumentos pásanse por valor, isto é, créase unha nova variable de xeito que calquera cambio que se faga sobre o seu valor dentro da función non afectará ao valor da variable que se pasa como parámetro. Se queres que isto aconteza debes definir o parámetro para que o seu valor se pase por referencia, engadindo o símbolo **&** antes do seu nome.

```
<?php
function suma(&$a, $b) {
  $a += $b;
}
?>
```

Tamén é posible definir unha función que reciba un número indefinido de argumentos. Faise empregando as funcións **func_num_args**, **func_get_arg**, e **func_get_args**.

```
<?php
function suma() {
  $num_sumandos = func_num_args();
  for($resultado = 0;$num_sumandos>0;$num_sumandos--;) {
    $resultado += func_get_arg($num_sumandos-1);
  }
}
?>
```

Variables estáticas

As variables locais a unha función perden o seu valor cando esta remata. Se queres manter o valor dunha variable entre as distintas chamadas a unha función, debes empregar variables estáticas.

```
<?php
function acumulador($valor) {
  static $total = 0;
  $total += $valor;
}
?>
```

As variables estáticas inicialízanse soamente a primeira vez que se executa a función.

--Víctor Lourido 14:43 25 jun 2013 (CEST)