Curso POO PHP Utilización de obxectos

Sumario

Utilización de obxectos

Para poder instanciar un obxecto dunha clase, esta debeu declararse previamente. Se temos cada clase no seu propio arquivo, deberemos facer un include ou require por cada clase que imos a empregar no script.

```
require_once("Produto.php");
require_once("Vendas.php");
require_once("Clientes.php");
...
```

Nota

En PHP5 é posible definir unha función de nome **__autoload** que faga ese traballo por nos. Cando no script intentemos instanciar un obxecto de unha clase non recoñecida, executarase a función __autoload pasándolle o nome da clase. Na función __autoload teremos poñer o código necesario para incluir a clase.

Por exemplo, o seguinte código inclúe un arquivo co mesmo nome da clase que se está a empregar:

```
function __autoload($nome_clase) {
    require_once $nome_clase.".php";
}
```

O problema coa función __autoload é que soamente permite definir unha función de busca da clase. Para poder definir máis de unha función de carga automática de clases, podemos empregar no seu lugar a función **spl_autoload_register**. O seu funcionamento é semellante ao de __autoload, pero debemos definir a función de carga con calquera nome, e pasarlle ese nome como parámetro. spl-autoload-register leva un rexistro de todas as funcións de carga automática que se lle pasan e executa as funcións de unha en unha para atopar a clase que se lle indica.

```
function _blog_autoload($nome_clase) {
    require_once $nome_clase.".php";
}
spl_autoload_register('_blog_autoload');
```

Unha vez declarada a clase, xa sexa de xeito automático ou manual, podemos usar a palabra **new** para instanciar obxectos da seguinte forma:

```
$p = new Produto();
```

Sobre os obxectos podes utilizar o operador **instanceof** para comprobar se é ou non unha instancia dunha clase determinada.

```
if ($p instanceof Produto) {
?
}
```

Para acceder aos atributos e aos métodos públicos do obxecto utilízase o operador frecha "**->**" (fíxate que só se pon o símbolo $ diante do nome do obxecto):

```
$p->nome = 'Samsung Galaxy S';
$p->get_codigo();
```

Para acceder ás constantes dunha clase, non é necesario que exista ningún obxecto da clase. Débese utilizar o nome da clase e o operador "**::**", chamado **operador de resolución de ámbito**.

```
echo Produto::EURO;
```

Cando dende un obxecto se invoca un método da clase, a este pásaselle sempre unha referencia ao obxecto que fixo a chamada. Esta referencia almacénase na variable **$this**. Utilízase, por exemplo, para ter acceso aos atributos e métodos privados do obxecto (que só son accesibles dende os métodos da clase).

Copia de obxectos e referencias

Unha característica da POO que debes ter moi en conta é que sucede cos obxectos cando os pasas a unha función, ou simplemente cando executas un código como o seguinte:

```
$p = new Produto();
$p->nome = 'Samsung Galaxy S';
$a = $p;
```

En PHP4, a última liña do código anterior crea un novo obxecto cos mesmos valores do orixinal, da mesma forma, que se copia calquera outro tipo de variable. Se despois de facer a copia se modifica, por exemplo, o atributo 'nome' dun dos obxectos, o outro obxecto non se vería modificado.

Non obstante, **en PHP5 este comportamento varía**. O código anterior simplemente crearía un novo identificador do mesmo obxecto, o mesmo que sucede cando creamos unha referencia a unha variable empregando o operador "**&**". Aínda que haxa dous ou máis identificadores do mesmo obxecto, en realidade todos se refiren á única copia que se almacena deste.

Polo tanto, a partir de PHP5 non podes copiar un obxecto utilizando o operador "**=**". Se necesitas copiar un obxecto, debes utilizar a palabra **clone**. Ao utilizar clone sobre un obxecto existente, crea unha copia de todos os atributos deste nun novo obxecto.

Comparación de obxectos

Ás veces tes dous obxectos e queres saber a súa relación exacta. Para comparar dous obxectos, en PHP5 podes utilizar os operadores "**==**" e "**===**" (en PHP4 as comparacións de obxectos funcionan de xeito diferente).

Se utilizas o operador de comparación **==**, comparas os valores dos atributos dos obxectos. Polo tanto dous obxectos serán iguais se son instancias da mesma clase e, ademais, os seus atributos teñen os mesmos valores.

Non obstante, se utilizas o operador **===**, o resultado da comparación será true só cando as dúas variables sexan referencias ao mesmo obxecto.

Funcións de utilidade para clases e obxectos

Ademais, en PHP5 inclúense unha serie de funcións útiles para o desenvolvemento de aplicacións utilizando POO.

Función	Significado
get_class	Devolve o nome da clase do obxecto
class_exists	Devolve true se a clase está definida ou false no caso contrario
get_declared_classes	Devolve un array cos nomes das clases definidas
class_alias	Crea un alcume para unha clase
get_class_methods	Devolve un array cos nomes dos métodos dunha clase que son accesibles dende onde se fai a chamada
method_exists	Devolve true se existe o método no obxecto ou a clase que se indica, ou false no caso contrario, independentemente de se é accesible ou non
get_class_vars	Devolve un array cos nomes dos atributos dunha clase que son accesibles dende onde se fai a chamada
get_object_vars	Devolve un array cos nomes dos métodos dun obxecto que son accesibles dende onde se fai a chamada
property_exists	Devolve true se existe o atributo no obxecto ou a clase que se indica, ou false no caso contrario, independentemente de se é accesible ou non

Implicación de tipos

A partir da versión 5, PHP incorpora a posibilidade de especificar a clase á que deben pertencer os obxectos que se pasen como parámetros ás funcións. Esta característica chámase **implicación de tipos**, e tamén pode empregarse con outros tipos como interfaces, arrays (dende PHP 5.1) e tipos "callable".

Chega con indicar o tipo antes do parámetro:

```
public function vendeProduto(Produto $p) {
?
```

```
}
```

Se cando se realiza a chamada, o parámetro non é do tipo axeitado, prodúcese un erro que se poderá capturar.

--Víctor Lourido 20:14 27 jun 2013 (CEST)