Curso POO PHP Espazos de nomes

Sumario

Espazos de nomes

A partires da versión 5.3 de PHP, introduciuse na linguaxe a capacidade de empregar espazos de nomes. É unha característica similar a que podemos atopar noutras linguaxes como Java ou C#. Os espazos de nomes permiten agrupar baixo un nome un conxunto de elementos (clases, interfaces, funcións e constantes mediante **const**, non define), de xeito que non se produzan conflictos entre os seus nomes e outros nomes iguais presentes noutros espazos de nomes.

Declaración dos espazos de nomes

Os espazos de nomes decláranse empregando a palabra **namespace**. Faise antes de calquera outro código (excepto sentenzas declare), e existen dúas sintaxes posibles:

- Rematando a sentenza nun punto e coma (**;**).

```
namespace Platega;

class xxx {
   ...
}
...
```

- Pechando entre chaves o código pertencente ao espazo de nomes.

```
namespace Platega {
   class xxx {
      ...
   }
   ...
}
```

Recoméndase empregar a primeira forma. A segunda é máis lexible cando temos dous ou máis espazos de nomes nun mesmo ficheiro, aínda que é aconsellable ter un único espazo de nomes por ficheiro.

Tamén é posible ter un mesmo espazo de nomes espallado en varios ficheiros. Deste xeito poderiamos, por exemplo, definir cada clase dun espazo de nomes no seu propio ficheiro.

No caso das librarías complexas de código, aconséllase empregar xerarquías de espazos de nomes. Faise separando cada rama do espazo de nomes da seguinte por unha barra invertida (\).

```
namespace Platega\Informatica\PHP;

class xxx {
   ...
}
...
```

Utilización dos espazos de nomes

Todos os obxecto que se atopan fora dun espazo de nomes definido mediante un namespace, forman parte do espazo de nomes global. Dependendo do espazo de nomes ao que pertenza o código que estea a executarse, poderemos empergar unha ou outra sintaxe para acceder a elementos no mesmo ou noutro espazo de nomes. Por exemplo, no seguinte código:

```
namespace Platega {
```

```
    const OLA = "Ola Platega.";
        ...
}
namespace Platega\PHP {
    const OLA = "Ola Platega PHP.";
        ...
}
```

- Se o código que estamos a executar se atopa **no espazo de nomes global**, teremos que empregar nomes completamente cualificados (rutas absolutas) para acceder ás constantes "OLA":

```
// Amosa "Ola Platega."
echo \Platega\OLA;

// Amosa "Ola Platega PHP."
echo \Platega\PHP\OLA;
```

  Como o espazo de nomes global atópase na raíz (\), tamén podemos emprear nomes parcialmente cualificados (rutas relativas) para acceder ás contantes anteriores:

```
// Amosa "Ola Platega."
echo Platega\OLA;

// Amosa "Ola Platega PHP."
echo Platega\PHP\OLA;
```

- Se o código que estamos a executar se atopa **no espazo de nomes PLATEGA**, podemos emprear nomes non cualificados, totalmente cualificados ou parcialmente cualificados do seguinte xeito:

```
// Nome totalmente cualificado. Amosa "Ola Platega."
echo \Platega\OLA;
// Nome non cualificado. Tamén amosa "Ola Platega."
echo OLA;

// Nome totalmente cualificado. Amosa "Ola Platega PHP."
echo \Platega\PHP\OLA;
// Nome parcialmente cualificado. Amosa "Ola Platega PHP."
echo PHP\OLA;
```

  Neste caso, cando empregamos nomes non cualificados PHP busca os elementos primeiro no espazo de nomes actual e despois no global. Para acceder a elementos do espazo de nomes global nos que ocurra unha colisión de nomes, teremos que empregar unha barra invertida.

```
// Nome non cualificado. Accede á constante do espazo de nomes global PHP_VERSION
echo PHP_VERSION;
// Se no espazo de nomes actual houbera outra constante con ese mesmo nome
//  teriamos que empregar un nome completamente cualificado para acceder á do espazo de nomes global
echo \PHP_VERSION;
```

Operador use

Podemos empregar o operador **use**, o mesmo que se usa nos trazos, para simplificar as referencias aos elementos dun espazo de nomes. Por exemplo, partindo dos seguintes espazos de nomes:

```
<?php
namespace Platega {
    const OLA = "Ola Platega.";
    class Proba {
        function __construct() {echo "Proba en Platega." . "<br />";}
    }
}
namespace Platega\PHP {
    const OLA = "Ola Platega PHP.";
    class Proba {
        function __construct() {echo "Proba en Platega PHP." . "<br />";}
    }
}
?>
```

Se queremos empregar elementos pertencentes ao espazo de nomes "Platega\PHP", o normal é facer:

```
<?php
$proba_platega_php = new \Platega\PHP\Proba();
?>
```

Cando traballamos con moitos elementos dun mesmo espazo de nomes, podemos porles un alias para acortar as rutas. Esta é a utilidade de **use**:

```
<?php
use Platega\PHP as P;
$proba_platega_php = new P\Proba();
?>
```

Se omitimos a segunda parte do *use* (o *as*), o alias a empregar é o último nome na xerarquía.

```
<?php
use Platega\PHP ;
$proba_platega_php = new PHP\Proba();
?>
```

O operador *use* tamén admite alias para os nomes das clases.

```
<?php
use Platega\PHP\Proba ;
$proba_platega_php = new Proba();
?>
```

Pero se dende dentro do espazo de nomes "Platega" facemos:

```
<?php
namespace Platega;
use Platega\PHP\Proba;
$proba = new Proba();
```

O nome da clase "Proba" pode facer referencia á do espazo de nomes actual (\Platega) ou á clase "Proba" da que fixemos o alias, a do espazo de nomes "\Platega\PHP". Cando isto sucede, PHP prioriza os alias sobre os elementos da clase actual. Isto é, o código anterior amosa a cadea "Proba en Platega PHP.".

Nestes casos podemos empregar a palabra clave **namespace** para facer referencia aos elementos do espazo de nomes actual.

```
<?php
namespace Platega;
use Platega\PHP\Proba;
$proba1 = new Proba();           // Amosa "Proba en Platega PHP."
$proba2 = new namespace\Proba(); // Amosa "Proba en Platega."
```

--Víctor Lourido 12:33 23 jul 2013 (CEST)