Espacios de nombres en PHP (II). Agrupando Namespaces.

En el artículo anterior nos introdujimos en el uso de los espacios de nombres, o namepaces (en muchos textos aparecen, abreviadamente, como **ns**). En este artículo vamos a ver como usar dos espacios de nombres diferentes en un mismo script.
Antes de entrar en materia debo advertirte que lo que vamos a aprender en este artículo es una práctica que, aún siendo sintácticamente legal en PHP, se encuentra sumamente desaconsejada, en orden a tener un código limpio y reutilizable. En realidad, no existe ninguna razón clara para usar lo que vamos a ver aquí, y sí hay muchas para no usarlo. La más obvia es que nuestro código será más limpio y reutilizable cuanto más encapsulado esté. En efecto, si yo necesito emplear los elementos que se encuentran bajo un determinado espacio de nombres, no hay ninguna razón por la que deba cargar un script que incluya otros espacios de nombres cuyos elementos no voy a necesitar.
A menudo, la diferencia entre encapsulación y sobrefragmentación del código es una línea muy delgada, pero el criterio que acabo de exponerte la define bastante bien.
Y ahora, si las prácticas que vamos a ver están desaconsejadas ¿por qué hablar de ellas? Con toda franqueza, no quería escribir sobre este punto, pero he visto varios códigos en Internet que emplean estas técnicas y, aunque nosotros, como buenos programadores, no las emplearemos nunca, debemos conocerlas (por si alguna vez nos toca refactorizar un script mal concebido.

MÚLTIPLES NAMESPACES

La forma de declarar más de un espacio de nombres en el mismo script es agrupando los elementos de cada uno bajo llaves ({ y }) y, dentro de estas, los correspondientes elementos. Lo vemos en el listado s1.php:

```
<?php
 namespace ns1{
  const CONSTANTE = 'Valor de constante en ns1'.'<br>';
  function funcionDePrueba(){
   echo "Esta en la función del script en ns1".'<br>';
  }
  class ClaseDePrueba{
   public static function metodoDeClase(){
    echo "Esta es una función de clase en ns1".'<br>';
   }
  }
 }

 namespace ns2{
  const CONSTANTE = 'Valor de constante en ns2'.'<br>';
  function funcionDePrueba(){
   echo "Esta en la función del script en ns2".'<br>';
  }
  class ClaseDePrueba{
   public static function metodoDeClase(){
    echo "Esta es una función de clase en ns2".'<br>';
   }
  }
```

```
 }
?>
```

Si te acuerdas, en el artículo anterior te decía que la línea que declara el espacio de nombres debe ser la primera del script, y también que había una excepción a esta regla. Pues bien, esta es la excepción: cuando agrupas los elementos que pertenecen a un espacio de nombres bajo llaves, cómo ves en el ejemplo.

Ahora tenemos el script index.php, que, en esta ocasión, ya no incluye dos scripts, sino, tan solo, el que acabamos de ver. Por lo demás, el listado es el mismo que en el artículo anterior:

```
<?php
 include 's1.php';

 const CONSTANTE = 'Valor de constante en script principal'.'<br>';

 function funcionDePrueba(){
  echo "Esta en la función del script en script principal".'<br>';
 }

 class ClaseDePrueba{
  public static function metodoDeClase(){
   echo "Esta es una función de clase en script principal".'<br>';
  }
 }

 echo CONSTANTE;
 funcionDePrueba();
 ClaseDePrueba::metodoDeClase();

 echo ns1CONSTANTE;
 ns1funcionDePrueba();
 ns1ClaseDePrueba::metodoDeClase();

 echo ns2CONSTANTE;
 ns2funcionDePrueba();
 ns2ClaseDePrueba::metodoDeClase();
?>
```

Si lo pruebas, verás que funciona igual que en el caso anterior (puedes descargarte los scripts en este enlace). No obstante, como hemos comentado, no es la forma adecuada de usar espacios de nombres.