Declaración de líneas de código:

Estructuras de Control  
Estos incluyen si, por mientras, switch, etc. Este es un ejemplo if, ya que es el más complicado de ellos:

<? Php

if (( condición1 ) || ( condition2 )) {

accion1 ;

} elseif (( Condition3 ) && ( condition4 )) {

acción2 ;

} else {

DefaultAction ;

}

?>

Las sentencias de control deben tener un espacio entre la palabra clave de control y paréntesis de apertura, para distinguirlas de las llamadas a funciones.

Se le recomienda utilizar siempre llaves, incluso en situaciones en que son técnicamente opcional. Tenerlos aumenta la legibilidad y disminuye la probabilidad de errores de lógica que se introdujo cuando se añaden nuevas líneas.

Para sentencias switch:

<? Php

interruptor ( condición ) {

caso 1 :

accion1 ;

break; caso

2 :

acción2 ;

break; por defecto:

DefaultAction ;

break;

}

?>

Llamadas de función

Funciones deben ser llamados sin espacios entre el nombre de la función, el paréntesis de apertura y el primer parámetro; espacios entre las comas y cada parámetro, y no hay espacio entre el último parámetro, el paréntesis de cierre, y el punto y coma. He aquí un ejemplo:

<? Php

$ var = foo ( $ bar , baz $ , $ quux );

?>

Como se muestra arriba, debe haber un espacio a ambos lados de un signo igual se utiliza para asignar el valor de retorno de una función a una variable. En el caso de un bloque de asignaciones relacionadas, más espacio se puede insertar para promover la legibilidad:

<? Php

$ a corto = foo ( $ bar );

$ long\_variable = foo ( $ baz );

?>

Para apoyar la legibilidad, los parámetros en las llamadas posteriores a la misma función / método se pueden alinear por nombre de parámetro:

<?php $this

-> callSomeFunction ( 'param1' , 'second' , true );

$this -> callSomeFunction ( 'parameter2' , 'third' , false );

$this -> callSomeFunction ( '3' , 'verrrrrrylong' , true );

?>

Definición de Clases

Declaraciones de clase tienen su llave de apertura de una nueva línea:

<? Php

clase foo\_bar

{

// ... código va aquí

}

?>

Definiciones de función

Las declaraciones de funciones siguen el " estilo K & R ":

<? Php

función fooFunction ( $ arg1 , arg2 $ = '' )

{

si ( condición ) {

declaración ;

}

retorno $ val ;

}

?>

Los argumentos con valores por defecto van al final de la lista de argumentos. Siempre tratar de devolver un valor significativo a partir de una función de si uno es apropiado. He aquí un ejemplo un poco más largo:

<? Php

función conectar (y $ dsn , $ persistente = false )

{

if ( is\_array ( $ dsn )) {

$ dsninfo = & $ dsn ;

} else {

$ dsninfo = DB :: parseDSN ( $ dsn );

} if ( !

$ dsninfo ||! $ dsninfo [ 'phptype' ]) {

volver $ this -> raiseError ();

} devolver

cierto ;

}

?>

Arrays

Las asignaciones en arrays pueden estar alineados. Al dividir las definiciones de matriz en varias líneas, el último valor puede tener también una coma final. Esta es la sintaxis PHP válida y ayuda a mantener diffs código mínimo:

<? Php $ some\_array

= array (

'foo' => 'bar' ,

'correo no deseado' => 'jamón' ,

);

?>