5. Control Structures

The general style rules for control structures are as follows:

•There MUST be one space after the control structure keyword

•There MUST NOT be a space after the opening parenthesis

•There MUST NOT be a space before the closing parenthesis

•There MUST be one space between the closing parenthesis and the opening brace

•The structure body MUST be indented once

•The body MUST be on the next line after the opening brace

•The closing brace MUST be on the next line after the body

<?php

if ($expr1) {

// if body

} elseif ($expr2) {

// elseif body

} else {

// else body;

}

?>

The keyword elseif SHOULD be used instead of else if so that all control keywords look like single words.

Expressions in parentheses MAY be split across multiple lines, where each subsequent line is indented at least once. When doing so, the first condition MUST be on the next line. The closing parenthesis and opening brace MUST be placed together on their own line with one space between them. Boolean operators between conditions MUST always be at the beginning or at the end of the line, not a mix of both.

<?php

if (

$expr1

&& $expr2

) {

// if body

} elseif (

$expr3

&& $expr4

) {

// elseif body

}

?>

5.2 switch, case

A switch structure looks like the following. Note the placement of parentheses, spaces, and braces. The case statement MUST be indented once from switch, and the break keyword (or other terminating keywords) MUST be indented at the same level as the case body. There MUST be a comment such as // no break when fall-through is intentional in a non-empty case body.

<?php

switch ($expr) {

case 0:

echo 'First case, with a break';

break;

case 1:

echo 'Second case, which falls through';

// no break

case 2:

case 3:

case 4:

echo 'Third case, return instead of break';

return;

default:

echo 'Default case';

break;

}

?>

Expressions in parentheses MAY be split across multiple lines, where each subsequent line is indented at least once. When doing so, the first condition MUST be on the next line. The closing parenthesis and opening brace MUST be placed together on their own line with one space between them. Boolean operators between conditions MUST always be at the beginning or at the end of the line, not a mix of both.

<?php

switch (

$expr1

&& $expr2

) {

// structure body

}

?>

5.3 while, do while

A while statement looks like the following. Note the placement of parentheses, spaces, and braces.

<?php

while ($expr) {

// structure body

}

?>

Expressions in parentheses MAY be split across multiple lines, where each subsequent line is indented at least once. When doing so, the first condition MUST be on the next line. The closing parenthesis and opening brace MUST be placed together on their own line with one space between them. Boolean operators between conditions MUST always be at the beginning or at the end of the line, not a mix of both.

<?php

while (

$expr1

&& $expr2

) {

// structure body

}

Similarly, a do while statement looks like the following. Note the placement of parentheses, spaces, and braces.

<?php

do {

// structure body;

} while ($expr);

?>

Expressions in parentheses MAY be split across multiple lines, where each subsequent line is indented at least once. When doing so, the first condition MUST be on the next line. Boolean operators between conditions MUST always be at the beginning or at the end of the line, not a mix of both.

<?php

do {

// structure body;

} while (

$expr1

&& $expr2

);

?>

5.4 for

A for statement looks like the following. Note the placement of parentheses, spaces, and braces.

<?php

for ($i = 0; $i < 10; $i++) {

// for body

}

?>

Expressions in parentheses MAY be split across multiple lines, where each subsequent line is indented at least once. When doing so, the first expression MUST be on the next line. The closing parenthesis and opening brace MUST be placed together on their own line with one space between them.

<?php

for (

$i = 0;

$i < 10;

$i++

) {

// for body

}

?>

5.5 foreach

A foreach statement looks like the following. Note the placement of parentheses, spaces, and braces.

<?php

foreach ($iterable as $key => $value) {

// foreach body

}

?>

5.6 try, catch, finally

A try-catch-finally block looks like the following. Note the placement of parentheses, spaces, and braces.

<?php

try {

// try body

} catch (FirstThrowableType $e) {

// catch body

} catch (OtherThrowableType | AnotherThrowableType $e) {

// catch body

} finally {

// finally body

}

?>

6. Operators

Style rules for operators are grouped by arity (the number of operands they take).

When space is permitted around an operator, multiple spaces MAY be used for readability purposes.

All operators not described here are left undefined.

6.1. Unary operators

The increment/decrement operators MUST NOT have any space between the operator and operand.

$i++;

++$j;

Type casting operators MUST NOT have any space within the parentheses:

$intValue = (int) $input;

6.2. Binary operators

All binary arithmetic, comparison, assignment, bitwise, logical, string, and type operators MUST be preceded and followed by at least one space:

if ($a === $b) {

$foo = $bar ?? $a ?? $b;

} elseif ($a > $b) {

$foo = $a + $b \* $c;

}

The conditional operator, also known simply as the ternary operator, MUST be preceded and followed by at least one space around both the ? and : characters:

$variable = $foo ? 'foo' : 'bar';

When the middle operand of the conditional operator is omitted, the operator MUST follow the same style rules as other binary comparison operators:

$variable = $foo ?: 'bar';