Advanced Weaving

Advanced Weaving

Painter's Edit Weave dialog lets you you create your own patterns, using Painter's own weaving language.

The Edit Weave dialog displays the controls for drafting a weave on Painter's 8-harness, 8-treadle loom. Painter lets you create woven fabrics of virtually any description. You can create fabrics for wall papers, carpets, clothes and furniture.

In this section, actual weaving vernacular is used so that weavers can become familiar with these techniques more easily. For those of you who are not familiar with weaving, read on and you will learn more about this ancient and venerable art.

To access the Edit Weave dialog:

* Click the menu icon in the Weave section of the Art Materials palette and choose **Edit Weave**.

 The Edit Weave dialog appears.
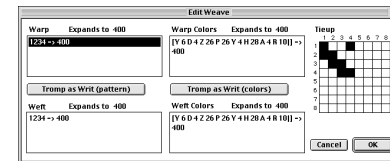
The Edit Weave dialog.

About Weaving

A weaving consists of vertical warp fibers and horizontal weft fibers that are interconnected on a loom to form a fabric.

Each warp thread is connected to exactly one harness in the loom. The harnesses are connected in a pattern to treadles. Pressing a treadle raises a set of harnesses, lifting the associated warp threads. A weft thread is then passed horizontally through the loom. The particular set of harnesses that are raised determines which warp threads are in front of the weft thread and which are behind.

For the next weft thread, a different treadleis pressed, raising a different set of harnesses and, hence, a different set of warp threads.

The Tieup

The tieup is an 8 by 8 array that describes which harnesses (on the left) are connected to which treadles (across the top). A black cell indicates that the harness is connected for the corresponding treadle.


The Tieup section of the Edit Weave dialog.

A white cell means the harness is not connected. During weaving, a black cell places the warp thread in front. A white cell places the weft thread in front. Only the thread that is in front is visible at that point in the weaving.

You can change the tieup by clicking on the cells, which flip between black and white.

The Warp and Weft Expressions

The warp and weft expressions (in the panels to the left of the draft design window) use the information in the tieup to extend and evolve the weaving into larger and more complex patterns.
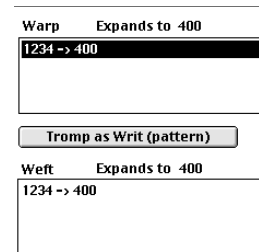
The Warp and Weft panels.

In terms of traditional weaving, the warp expression is the threading sequence, and the weft expression is the treadling sequence.

Warp

The threading sequence describes which warp threads are connected to which harnesses.

For example, a warp sequence of 234123 means to...

Connect warp thread #1 to harness #2

Connect thread #2 to harness #3

Connect thread #3 to harness #4

Connect thread #4 to harness #1

Connect thread #5 to harness #2

Connect thread #6 to harness #3.

The sequence repeats for subsequent threads.

Connect thread #7 to harness #2

Connect thread #8 to harness #3

Connect thread #9 to harness #4

continuing....

Notice that this threading sequence has a length of 6. This means that the pattern runs for 6 threads before repeating.

Weft

The treadling sequence determines which treadle is pressed (and therefore which harnesses are raised) for each successive weft thread.

For example, a weft sequence of 654 means to...

Press treadle #6 for weft thread #1.

Press treadle #5 for weft thread #2

Press treadle #4 for weft thread #3

The sequence repeats for subsequent threads.

Press treadle #6 for weft thread #4.

Press treadle #5 for weft thread #5

Press treadle #4 for weft thread #6

repeating....

Changing the Warp and Weft

You can change the threading and treadling sequences by editing the warp and weft expressions in the weaving draft design window. To simplify the process, Painter allows you to work in a pattern expression language.

For example, 1223334444333221 can be expressed in the pattern language as:

(1-4-1)[](1-4-1).

The following section describes the weaving pattern language in detail.

Working with pattern representation symbols can be far easier than the number jumble of the expanded sequence.

As you create a warp or weft sequence, the "Expands to n" message above the panel changes to give the length n (number of digits) of the expression. This describes the number of threads in the sequence.

When Painter cannot resolve the expression, the expansion length is given as "!" (error).

The Tromp as Writ button copies the warp expression to the weft expression.

Changing Thread Colors

Both the warp and the weft threads may be colored. You may use one color for all the warp threads and another for all the weft threads, or you may set up a sequence of colors for each.

The pattern language is recognized for thread color sequences. The colors must be identified alphabetically, so you are limited to 26 colors of thread in the weaving.

In the warp and weft color panels, create expressions that describes the color sequences you want. Use any letters you like; you can change the associated color later.

For information on editing the colors in the color set, refer to "Editing Weave Colors" in the Painter 6 User Guide.

To open the color set for the weave:

• Click the menu icon in the Weave section of the Art Materials palette and choose **Get Color Set**. Painter displays the color set.

To see the letters associated with each color:

• Open the Color Sets section of the Art Materials Palette and enable the Display Text option in the bottom left corner.

For example: [B 8 D 4] means to "Use color B for 8 threads, use color D for 4 threads."

For example: ABC[]248 expands to [A2 B4 C8]
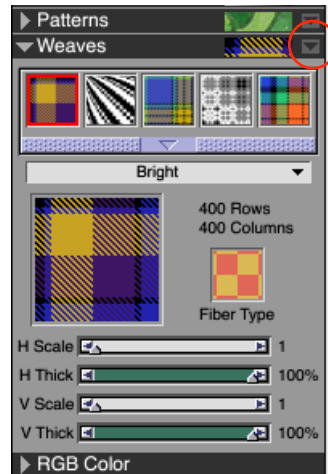
The menu icon in the Weave section of the Art Materials palette.

Getting into the Weaving Pattern Language

The Domain

The Painter Weaving Pattern Language is designed around a group of values, called the domain of the pattern. For the warp or weft sequences, the domain is the numbers 1 through 8, which correspond to the harnesses and treadles in the tieup.

Patterns wrap around on their domain so the number one higher than 8 is actually 1, and the number one lower than 1 is actually 8. This is called modular arithmetic. Some examples are found in the sequences: ...5678123... or ...3218765....

Expression Operators

The following table describes each operator in the pattern language.

In all examples, the operands used for the operators are numeric sequences. However, the operands may be other patterns created from other operators. In this case, use parentheses to distinguish operands.

"Expansion" refers to the actual warp or weft numeric sequence created when the expression is translated from the shorthand pattern language.

"Length" describes the number of elements in the expanded sequence. For example, (1-4-1)[](1-4-1) expands to 1223334444333221, which has a length of 16.

BLOCK

Operator Usage: <pattern> block <count> Short form: []

The left operand is a sequence of any length. The right operand is a pattern of digit characters for specifying block lengths from 0 to 9.

If either operand is shorter than the other, it is extended (via repeating) to exactly the length of the other operand.

Then, with equal length operands, each element of the left operand is repeated individually by the value of the corresponding digit in the right operand.

The length of the expansion is the sum of the digits in the (possibly lengthened) right operand.

Individual elements of the right operand (lengths) can be replaced by {number} for counts from 1 to 127.

Examples: 1234 [] 2 Expansion: 11223344

34512 [] 23 Expansion: 334445511122

12345678 [] 87654321 Expansion: 111111112222222333333444445555666778

12 [] 123494321 Expansion: 122111222111111112222111221

123 [] 1{12}3 Expansion: 1222222222222333

Block Expressions

Block expressions may be used in an interleaved format. This is particularly useful for color sequences like those used in the Scottish tartan plaids. For example, one of the Drummond color sequences is expressed as follows:

[G 8 R 2 G 2 R 56 G 16 B 2 G 2 B 2 G 36 R 2 G 2 R 8|].

REPEAT

Operator Usage: <pattern> repeat <count> Short form: rep, ★

The left operand is a pattern of any length. The right operand is an integer count. The expansion is the <pattern> repeated <count> times.

Example: 1234567 repeat 3 Expansion: 123456712345671234567

EXTEND

Operator Usage: <pattern> extend <count> Short form: ext, ->

The left operand is a pattern of any length. The right operand is an integer that is interpreted as a length. The left operand is reshaped to be the length as specified by the right operand. If the left operand is shorter that the desired length, it is repeated out to the right length. If the left operand is shorter, it is merely trimmed.

Examples: 123 -> 8 Expansion: 12312312

123456787654321 -> 10 Expansion: 1234567876

CONCAT

Operator Usage: <pattern> concat <pattern> Short form: ,

The left and right operands are both patterns of any length. The result is the left operand concatenated with the right operand. Note that when a pattern is spelled out (e.g. 1234), the concat operation is implicit between each of its members (i.e. 1234 is equivalent to 1,2,3,4). The concat operator is used only when needed—for example, to separate parenthetical operands.

Examples: (1-4|),(4-1|) Expansion: 123432432123

                           1-8,(1-8 [] 2) Expansion: 12345678112233445566778

INTERLEAVE

Operator Usage: <pattern> interleave <pattern> Short form: int, ~

                           The left and right operands are both patterns of any length. If either operand is shorter than
                           the other, it is extended (via repeating) to exactly the length of the other operand. Then,
                           the operands are interleaved Interleaving alternates the elements of the left operand with
                           the elements of the right (like lacing your fingers together). The expansion contains the first
                           element of the left, the first element of the right, the second element of the left, the second
                           element of the right, and so on.

Examples: 123 interleave 76 Expansion: 172637

                           12345678 ~ 4321 Expansion: 1423324154637281

                           123456787654321 ~ 121221222 Expansion: 112231425261728272615241322211

UPTO

Operator Usage: <pattern> upto <pattern> Short form: <, -

                           The "-" may be used as an upto operator only when the last element of the left operand is
                           strictly less than the first element of the right operand.

                           The left and right operands are both patterns of any length.

                           The expansion is the left operand concatenated with the sequence between the last element
                           of the left operand and the first element of the right operand, concatenated with the rest of
                           the right operand.

                           Upto wraps around on the domain. For example, 7 upto 2 expands to 7812.

The upto operator may be followed immediately (with no intervening characters) by any number of tick marks ('). A tick mark indicates to cycle through the domain before running up to the right operand. The number of tick marks sets the number of cycles.

Examples:

	1–8 Expansion:	12345678
	1–"5 Expansion:	123456781234567812345
	123–765 Expansion:	123456765
	2<1 Expansion:	23456781
	13 upto" 5812 Expansion:	13456781234567812345812

DOWNTO

Operator Usage: \<pattern\> downto \<pattern\> Short form: \>, –

The "–" may be used as an downto operator only when the last element of the left operand is strictly greater than the first element of the right operand.

The left and right operands are both patterns of any length.

The expansion is the left operand concatenated with the descending sequence between the last element of the left operand and the first element of the right operand, concatenated with the rest of the right operand.

Downto wraps around on the domain. For example, 2 downto 7 expands to 2187.

The downto operator may be followed immediately (with no intervening characters) by any number of tick marks ('). A tick mark indicates to cycle through the domain before running down to the right operand. The number of tick marks sets the number of cycles.

Examples:	8-1 Expansion:	87654321
	1-8-1 Expansion:	123456787654321
	5-"1 Expansion:	543218765432187654321
	567-321 Expansion:	567654321
	2>1 Expansion:	21
	13 downto" 5812 Expansion:	132187654321876543218765812

TEMPLATE

Operator Usage: <pattern> template <pattern> Short form: temp, :

The right and left operands are patterns of any length.

Template allows texturing (sub-articulation) of a master pattern (the left operand) with a texture pattern (the right operand).

The first element of the template is taken as the root. All other elements in the template sequence are considered with respect to their difference from the root. For example, if 342 is the template pattern, the first element (3) is the root (r). The second element (4) is r+1, and the third element (2) is r-1. The template is then r, r+1, r-1.

This template is then applied to each element in the left operand, with the element's value replacing the root.

Remember that the values wrap around on the domain.

Examples:	1-8 temp 121 Expansion:	121232343454565676787818
	1-8 : 12121 Expansion:	12121232323434345454565656767676787878 1818

12321 temp 878 Expansion:	878181212181878
12321 : 3 Expansion:	34543

PALINDROME

Operator Usage: <pattern> palindrome Short form: pal, |

This operator only takes one operand, the left operand. That operand can be a pattern of any length. The resultant expansion is the left operand concatenated with itself backwards. However, the center element is not repeated, and the last element of the result is dropped automatically. This is essential when the palindromed sequence is repeated and helps to remove flaws from a weaving draft.

A palindrome word is the same, spelled backwards or forwards, like "madam."

Examples:	1-8	Expansion:	12345678765432
	1223334444555	Expansion:	12233344445555544443332

PERMUTE

Operator Usage: <pattern> permute <permutation> Short form: perm

The left operand is a pattern of any length. The right operand is a permutation vector.

The left operand is extended to a length that is an integer multiple of the right operand's length.

Then the left operand is permuted in chunks of the right operand's length. The organization of the permutation pattern is used to reorganize the elements in each chunk of the pattern.

For example, a permutation of 312 puts "the third element first, the first element second and the second element third."

Examples: 123456787654321 permute 312 Expansion: 312645778465132

                             1-8 perm 1324 Expansion: 13245768

PBOX

Operator Usage: <pattern> pbox <permutation> Short form: none

                             The left operand is a pattern of any length. The right operand is a permutation vector. Its
                             length will be adjusted, if necessary, (by trimming or repeating) to equal the length of the
                             left operand.

                             Then, the organization of the permutation pattern is used to reorganize the elements of the
                             pattern.

                             Pbox maintains the length of the left operand pattern.

Examples: 123456787654321 pbox 21436587 Expansion: 214365878563412

                             1-'8 pbox 4-18-5 Expansion: 4321876543218765

UPDOWN

Operator Usage: <pattern> updown <pattern> Short form: <>

                             The left and right operands are both patterns of any length. If either operand is shorter than
                             the other, it is extended (via repeating) to exactly the length of the other operand.

                             Then, UPDOWN generates alternating ascending and descending runs. An odd number of
                             runs is always generated. The expansion contains the ascending run from the first element
                             of the left operand to the first element of the right operand, followed by the descending run
                             from there to the second element of the first operand, etc. The operation is complete with
                             the ascending run from the last element of the left operand to the last element of the right
                             operand.

The updown operator may be followed immediately (with no intervening characters) by any number of tick marks ('). A tick mark indicates to cycle through the domain before completing the up and down runs. The number of tick marks sets the number of cycles.

Examples: 1234 updown 5678 Expansion: 123454323456543456765445678

12 <>' 567 Expansion: 12345678123454321876543234567812345654321876543212345678 1234567

DOWNUP

Operator Usage: <pattern> downup <pattern> Short form: ><

The left and right operands are both patterns of any length. If either operand is shorter than the other, it is extended (via repeating) to exactly the length of the other operand.

Then, DOWNUP generates alternating descending and ascending runs. An odd number of runs is always generated. The expansion contains the descending run from the first element of the left operand to the first element of the right operand, followed by the ascending run from there to the second element of the first operand, etc. The operation is complete with the descending run from the last element of the left operand to the last element of the right operand.

The downup operator may be followed immediately (with no intervening characters) by any number of tick marks ('). A tick mark indicates to cycle through the domain before completing the up and down runs. The number of tick marks sets the number of cycles.

Examples: 1234 downup 5678 Expansion: 187656781218767812321878123432187812343218

12 ><' 567 Expansion: 18765432187656781234567812187654321 876781234567818765432187

ROTATE

Operator Usage: \<pattern\> rotate \<number\> Short Form: rot, #

The left operand is a pattern of any length. The right operand is an integer (which may be negative) that is interpreted as a length. Rotate takes the set of elements specified by \<number\> from the start of the pattern and moves them to the end. A negative number takes a set of elements from the end of the pattern and moves them to the beginning.

Examples: 12345678 rotate 3 Expansion: 45678123
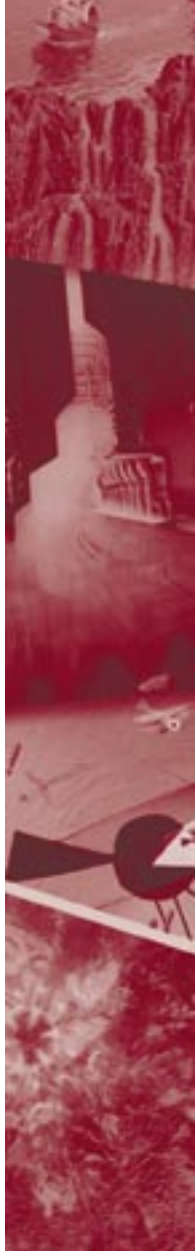
1-4-1 rotate –3 Expansion: 3211234

REVERSE

Operator Usage: \<pattern\> reverse Short form: rev, `

This operator only takes one operand, the left operand. That operand may be a pattern of any length. The resulting expansion is the left operand written backwards.

Examples: 1-8 reverse Expansion: 87654321

1223334444555\` Expansion: 5554444333221

Precedence Ordering

When several operators are used in an expression, they are evaluated in order of precedence. That is, some operations are performed before others. The precedence levels for the pattern language are shown in the following:

upto, downto, concat

downup, updown

repeat, extend, template, interleave,

permute, pbox

palindrome

When two operators have the same precedence, left to right ordering is followed.

Examples:

1-4-2 is interpreted as:	((1 upto 4) downto 2)
1234[]4321| is interpreted as:	((1234 block 4321) palindrome)
1-4 block 4-1 is interpreted as:	((1 upto 4) block (4 upto 1))

| (1-4|),(4-1|)| is interpreted as: | ((((1 upto 4) palindrome) concat ((4 upto 1) palindrome)) palindrome) |

Where precedence does the wrong thing, or where it is unclear, use parentheses to bring out the intended order of evaluation.