8 Box model

Contents

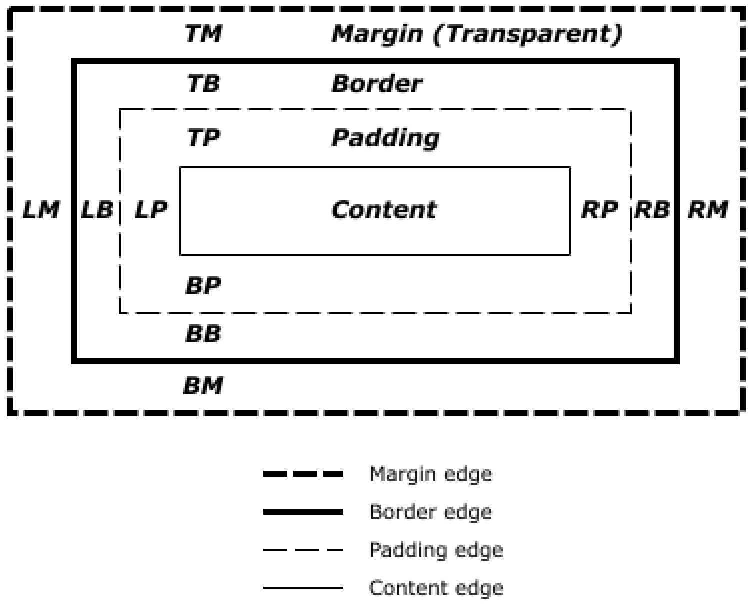
Note: Several sections of this specification have been updated by other specifications. Please, see "Cascading Style Sheets (CSS) — The Official Definition" in the latest *CSS Snapshot* for a list of specifications and the sections they replace.

The CSS Working Group is also developing CSS level 2 revision 2 (CSS 2.2).

The CSS box model describes the rectangular boxes that are generated for elements in the document tree and laid out according to the visual formatting model.

8.1 Box dimensions

Each box has a *content area* (e.g., text, an image, etc.) and optional surrounding *padding*, *border*, and *margin* areas; the size of each area is specified by properties defined below. The following diagram shows how these areas relate and the terminology used to refer to pieces of margin, border, and padding:



The margin, border, and padding can be broken down into top, right, bottom, and left segments (e.g., in the diagram, "LM" for left margin, "RP" for right padding, "TB" for top border, etc.).

The perimeter of each of the four areas (content, padding, border, and margin) is called an "edge", so each box has four edges:

content edge or inner edge

The content edge surrounds the rectangle given by the width and height of the box, which often depend on the element's rendered content. The four content edges define the box's *content box*.

padding edge

The padding edge surrounds the box padding. If the padding has 0 width, the padding edge is the same as the content edge. The four padding edges define the box's *padding box*.

border edge

The border edge surrounds the box's border. If the border has 0 width, the border edge is the same as the padding edge. The four border edges define the box's *border box*.

margin edge or outer edge

The margin edge surrounds the box margin. If the margin has 0 width, the margin edge is the same as the border edge. The four margin edges define the box's *margin box*.

Each edge may be broken down into a top, right, bottom, and left edge.

The dimensions of the content area of a box — the *content width* and *content height* — depend on several factors: whether the element generating the box has the 'width' or 'height' property set, whether the box contains text or other boxes, whether the box is a table, etc. Box widths and heights are discussed in the chapter on visual formatting model details.

The background style of the content, padding, and border areas of a box is specified by the 'background' property of the generating element. Margin backgrounds are always transparent.

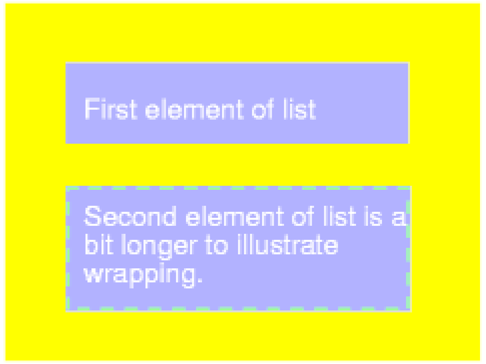
8.2 Example of margins, padding, and borders

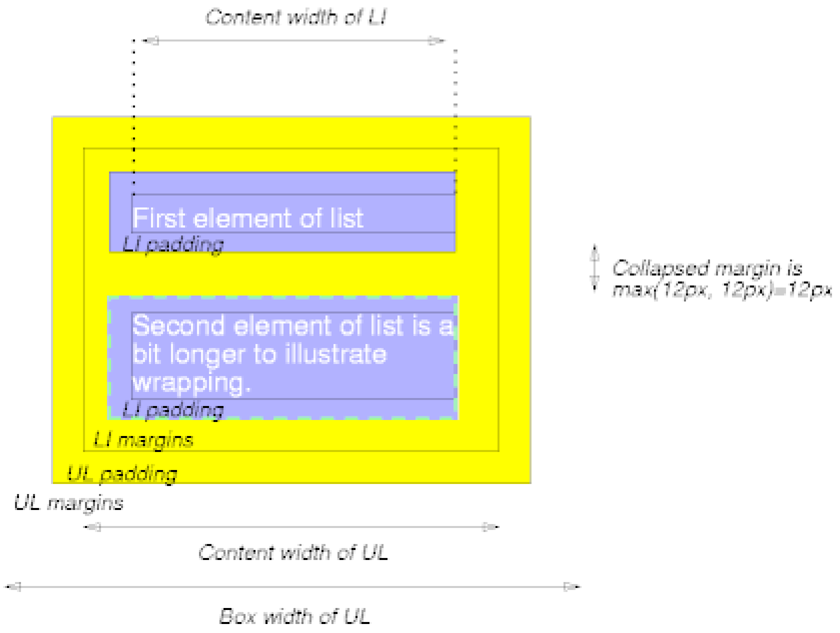
This example illustrates how margins, padding, and borders interact. The example HTML document:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
  <HEAD>
    <TITLE>Examples of margins, padding, and borders</TITLE>
    <STYLE type="text/css">
      UL {
        background: yellow;
        margin: 12px 12px 12px 12px;
        padding: 3px 3px 3px 3px;
                                     /* No borders set */
      }
      LI {
        color: white;                /* text color is white */
        background: blue;            /* Content, padding will be blue */
        margin: 12px 12px 12px 12px;
        padding: 12px 0px 12px 12px; /* Note 0px padding right */
        list-style: none             /* no glyphs before a list item */
                                     /* No borders set */
      }
      LI.withborder {
        border-style: dashed;
        border-width: medium;        /* sets border width on all sides */
        border-color: lime;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <UL>
      <LI>First element of list
      <LI class="withborder">Second element of list is
           a bit longer to illustrate wrapping.
    </UL>
  </BODY>
</HTML>
```

results in a document tree with (among other relationships) a UL element that has two LI children.

The first of the following diagrams illustrates what this example would produce. The second illustrates the relationship between the margins, padding, and borders of the UL elements and those of its children LI elements. (Image is not to scale.)


First element of list
Second element of list is a bit longer to illustrate wrapping.


Content width of LI
First element of list
LI padding
Second element of list is a bit longer to illustrate wrapping.
LI padding
LI margins
UL padding
UL margins
Content width of UL
Box width of UL
Collapsed margin is max(12px, 12px)=12px

Note that:

- The content width for each LI box is calculated top-down; the containing block for each LI box is established by the UL element.
- The margin box height of each LI box depends on its content height, plus top and bottom padding, borders, and margins. Note that vertical margins between the LI boxes collapse.
- The right padding of the LI boxes has been set to zero width (the 'padding' property). The effect is apparent in the second illustration.
- The margins of the LI boxes are transparent — margins are always transparent — so the background color (yellow) of the UL padding and content areas shines through them.
- The second LI element specifies a dashed border (the 'border-style' property).

8.3 Margin properties: 'margin-top', 'margin-right', 'margin-bottom', 'margin-left', and 'margin'

Margin properties specify the width of the margin area of a box. The 'margin' shorthand property sets the margin for all four sides while the other margin properties only set their respective side. These properties apply to all elements, but vertical margins will not have any effect on non-replaced inline elements.

The properties defined in this section refer to the **<margin-width>** value type, which may take one of the following values:

<length>
: Specifies a fixed width.

<percentage>
: The percentage is calculated with respect to the *width* of the generated box's containing block. Note that this is true for 'margin-top' and 'margin-bottom' as well. If the containing block's width depends on this element, then the resulting layout is undefined in CSS 2.1.

auto
: See the section on calculating widths and margins for behavior.

Negative values for margin properties are allowed, but there may be implementation-specific limits.

'margin-top', 'margin-bottom'

Value:	<margin-width> | inherit
Initial:	0
Applies to:	all elements except elements with table display types other than table-caption, table and inline-table
Inherited:	no
Percentages:	refer to width of containing block
Media:	visual
Computed value:	the percentage as specified or the absolute length

These properties have no effect on non-replaced inline elements.

'margin-right', 'margin-left'

Value:	<margin-width> | inherit
Initial:	0
Applies to:	all elements except elements with table display types other than table-caption, table and inline-table
Inherited:	no
Percentages:	refer to width of containing block
Media:	visual
Computed value:	the percentage as specified or the absolute length

These properties set the top, right, bottom, and left margin of a box.

```
h1 { margin-top: 2em }
```

'margin'

Value:	<margin-width>{1,4} | inherit
Initial:	see individual properties
Applies to:	all elements except elements with table display types other than table-caption, table and inline-table

Inherited:	no
Percentages:	refer to width of containing block
Media:	visual
Computed value:	see individual properties

The 'margin' property is a shorthand property for setting 'margin-top', 'margin-right', 'margin-bottom', and 'margin-left' at the same place in the style sheet.

If there is only one component value, it applies to all sides. If there are two values, the top and bottom margins are set to the first value and the right and left margins are set to the second. If there are three values, the top is set to the first value, the left and right are set to the second, and the bottom is set to the third. If there are four values, they apply to the top, right, bottom, and left, respectively.

```
body { margin: 2em }         /* all margins set to 2em */
body { margin: 1em 2em }     /* top & bottom = 1em, right & left = 2em */
body { margin: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
```

The last rule of the example above is equivalent to the example below:

```
body {
  margin-top: 1em;
  margin-right: 2em;
  margin-bottom: 3em;
  margin-left: 2em;        /* copied from opposite side (right) */
}
```

8.3.1 Collapsing margins

In CSS, the adjoining margins of two or more boxes (which might or might not be siblings) can combine to form a single margin. Margins that combine this way are said to *collapse*, and the resulting combined margin is called a *collapsed margin*.

Adjoining vertical margins collapse, except:

- Margins of the root element's box do not collapse.
- If the top and bottom margins of an element with clearance are adjoining, its margins collapse with the adjoining margins of following siblings but that resulting margin does not collapse with the bottom margin of the parent block.

Horizontal margins never collapse.

Two margins are *adjoining* if and only if:

- both belong to in-flow block-level boxes that participate in the same block formatting context
- no line boxes, no clearance, no padding and no border separate them (Note that certain zero-height line boxes (see 9.4.2) are ignored for this purpose.)
- both belong to vertically-adjacent box edges, i.e. form one of the following pairs:
  - top margin of a box and top margin of its first in-flow child
  - bottom margin of box and top margin of its next in-flow following sibling
  - bottom margin of a last in-flow child and bottom margin of its parent if the parent has 'auto' computed height
  - top and bottom margins of a box that does not establish a new block formatting context and that has zero computed 'min-height', zero or 'auto' computed 'height', and no in-flow children

A collapsed margin is considered adjoining to another margin if any of its component margins is adjoining to that margin.

Note. Adjoining margins can be generated by elements that are not related as siblings or ancestors.

Note the above rules imply that:

- Margins between a floated box and any other box do not collapse (not even between a float and its in-flow children).
- Margins of elements that establish new block formatting contexts (such as floats and elements with 'overflow' other than 'visible') do not collapse with their in-flow children.
- Margins of absolutely positioned boxes do not collapse (not even with their in-flow children).
- Margins of inline-block boxes do not collapse (not even with their in-flow children).
- The bottom margin of an in-flow block-level element always collapses with the top margin of its next in-flow block-level sibling, unless that sibling has clearance.
- The top margin of an in-flow block element collapses with its first in-flow block-level child's top margin if the element has no top border, no top padding, and the child has no clearance.
- The bottom margin of an in-flow block box with a 'height' of 'auto' and a 'min-height' of zero collapses with its last in-flow block-level child's bottom margin if the box has no bottom padding and no bottom border and the child's bottom

margin does not collapse with a top margin that has clearance.
- A box's own margins collapse if the 'min-height' property is zero, and it has neither top or bottom borders nor top or bottom padding, and it has a 'height' of either 0 or 'auto', and it does not contain a line box, and all of its in-flow children's margins (if any) collapse.

When two or more margins collapse, the resulting margin width is the maximum of the collapsing margins' widths. In the case of negative margins, the maximum of the absolute values of the negative adjoining margins is deducted from the maximum of the positive adjoining margins. If there are no positive margins, the maximum of the absolute values of the adjoining margins is deducted from zero.

If the top and bottom margins of a box are adjoining, then it is possible for margins to *collapse through* it. In this case, the position of the element depends on its relationship with the other elements whose margins are being collapsed.

- If the element's margins are collapsed with its parent's top margin, the top border edge of the box is defined to be the same as the parent's.
- Otherwise, either the element's parent is not taking part in the margin collapsing, or only the parent's bottom margin is involved. The position of the element's top border edge is the same as it would have been if the element had a non-zero bottom border.

Note that the positions of elements that have been collapsed through have no effect on the positions of the other elements with whose margins they are being collapsed; the top border edge position is only required for laying out descendants of these elements.

8.4 Padding properties: 'padding-top', 'padding-right', 'padding-bottom', 'padding-left', and 'padding'

The padding properties specify the width of the padding area of a box. The 'padding' shorthand property sets the padding for all four sides while the other padding properties only set their respective side.

The properties defined in this section refer to the **<padding-width>** value type, which may take one of the following values:

<length>
: Specifies a fixed width.

<percentage>
: The percentage is calculated with respect to the *width* of the generated box's containing block, even for 'padding-top' and 'padding-bottom'. If the containing block's width depends on this element, then the resulting layout is undefined in CSS 2.1.

Unlike margin properties, values for padding values cannot be negative. Like margin properties, percentage values for padding properties refer to the width of the generated box's containing block.

'padding-top', 'padding-right', 'padding-bottom', 'padding-left'

Value:	<padding-width> | inherit
Initial:	0
Applies to:	all elements except table-row-group, table-header-group, table-footer-group, table-row, table-column-group and table-column
Inherited:	no
Percentages:	refer to width of containing block
Media:	visual
Computed value:	the percentage as specified or the absolute length

These properties set the top, right, bottom, and left padding of a box.

```
blockquote { padding-top: 0.3em }
```

'padding'

Value:	<padding-width>{1,4} | inherit
Initial:	see individual properties
Applies to:	all elements except table-row-group, table-header-group, table-footer-group, table-row, table-column-group and table-column
Inherited:	no

Percentages:	refer to width of containing block
Media:	visual
Computed value:	see individual properties

The 'padding' property is a shorthand property for setting 'padding-top', 'padding-right', 'padding-bottom', and 'padding-left' at the same place in the style sheet.

If there is only one component value, it applies to all sides. If there are two values, the top and bottom paddings are set to the first value and the right and left paddings are set to the second. If there are three values, the top is set to the first value, the left and right are set to the second, and the bottom is set to the third. If there are four values, they apply to the top, right, bottom, and left, respectively.

The surface color or image of the padding area is specified via the 'background' property:

```
h1 {
  background: white;
  padding: 1em 2em;
}
```

The example above specifies a '1em' vertical padding ('padding-top' and 'padding-bottom') and a '2em' horizontal padding ('padding-right' and 'padding-left'). The 'em' unit is relative to the element's font size: '1em' is equal to the size of the font in use.

8.5 Border properties

The border properties specify the width, color, and style of the border area of a box. These properties apply to all elements.

Note. *Notably for HTML, user agents may render borders for certain user interface elements (e.g., buttons, menus, etc.) differently than for "ordinary" elements.*

8.5.1 Border width: 'border-top-width', 'border-right-width', 'border-bottom-width', 'border-left-width', and 'border-width'

The border width properties specify the width of the border area. The properties defined in this section refer to the **<border-width>** value type, which may take one of the following values:

thin
: A thin border.

medium
: A medium border.

thick
: A thick border.

<length>
: The border's thickness has an explicit value. Explicit border widths cannot be negative.

The interpretation of the first three values depends on the user agent. The following relationships must hold, however:
'thin' <='medium' <= 'thick'.
Furthermore, these widths must be constant throughout a document.

'border-top-width', 'border-right-width', 'border-bottom-width', 'border-left-width'

Value:	<border-width> | inherit
Initial:	medium
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	absolute length; '0' if the border style is 'none' or 'hidden'

These properties set the width of the top, right, bottom, and left border of a box.

'border-width'

Value:	<border-width>{1,4} | inherit
Initial:	see individual properties

Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	see individual properties

This property is a shorthand property for setting 'border-top-width', 'border-right-width', 'border-bottom-width', and 'border-left-width' at the same place in the style sheet.

If there is only one component value, it applies to all sides. If there are two values, the top and bottom borders are set to the first value and the right and left are set to the second. If there are three values, the top is set to the first value, the left and right are set to the second, and the bottom is set to the third. If there are four values, they apply to the top, right, bottom, and left, respectively.

In the examples below, the comments indicate the resulting widths of the top, right, bottom, and left borders:

```
h1 { border-width: thin }                 /* thin thin thin thin */
h1 { border-width: thin thick }           /* thin thick thin thick */
h1 { border-width: thin thick medium }    /* thin thick medium thick */
```

8.5.2 Border color: 'border-top-color', 'border-right-color', 'border-bottom-color', 'border-left-color', and 'border-color'

The border color properties specify the color of a box's border.

'border-top-color', 'border-right-color', 'border-bottom-color', 'border-left-color'

Value:	<color> | transparent | inherit
Initial:	the value of the 'color' property
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual

Computed value:	when taken from the 'color' property, the computed value of 'color'; otherwise, as specified

'border-color'

Value:	[<color> | transparent]{1,4} | inherit
Initial:	see individual properties
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	see individual properties

The 'border-color' property sets the color of the four borders. Values have the following meanings:

<color>
Specifies a color value.

transparent
The border is transparent (though it may have width).

The 'border-color' property can have from one to four component values, and the values are set on the different sides as for 'border-width'.

If an element's border color is not specified with a border property, user agents must use the value of the element's 'color' property as the computed value for the border color.

In this example, the border will be a solid black line.

```
p {
  color: black;
  background: white;
  border: solid;
}
```

8.5.3 Border style: 'border-top-style', 'border-right-style', 'border-bottom-style', 'border-left-style', and 'border-style'

The border style properties specify the line style of a box's border (solid, double, dashed, etc.). The properties defined in this section refer to the **<border-style>** value type, which may take one of the following values:

none
: No border; the computed border width is zero.

hidden
: Same as 'none', except in terms of border conflict resolution for table elements.

dotted
: The border is a series of dots.

dashed
: The border is a series of short line segments.

solid
: The border is a single line segment.

double
: The border is two solid lines. The sum of the two lines and the space between them equals the value of 'border-width'.

groove
: The border looks as though it were carved into the canvas.

ridge
: The opposite of 'groove': the border looks as though it were coming out of the canvas.

inset
: The border makes the box look as though it were embedded in the canvas.

outset
: The opposite of 'inset': the border makes the box look as though it were coming out of the canvas.

All borders are drawn on top of the box's background. The color of borders drawn for values of 'groove', 'ridge', 'inset', and 'outset' depends on the element's border color properties, but UAs may choose their own algorithm to calculate the actual colors used. For instance, if the 'border-color' has the value 'silver', then a UA could use a gradient of colors from white to dark gray to indicate a sloping border.

'border-top-style', 'border-right-style', 'border-bottom-style', 'border-left-style'

Value:	<border-style> | inherit
Initial:	none
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	as specified

'border-style'

Value:	<border-style>{1,4} | inherit
Initial:	see individual properties
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	see individual properties

The 'border-style' property sets the style of the four borders. It can have from one to four component values, and the values are set on the different sides as for 'border-width' above.

```
#xy34 { border-style: solid dotted }
```

In the above example, the horizontal borders will be 'solid' and the vertical borders will be 'dotted'.

Since the initial value of the border styles is 'none', no borders will be visible unless the border style is set.

8.5.4 Border shorthand properties: 'border-top', 'border-right', 'border-bottom', 'border-left', and 'border'

'border-top', 'border-right', 'border-bottom', 'border-left'

Value:	[<border-width> || <border-style> || <'border-top-color'>] | inherit
Initial:	see individual properties
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	see individual properties

This is a shorthand property for setting the width, style, and color of the top, right, bottom, and left border of a box.

```
h1 { border-bottom: thick solid red }
```

The above rule will set the width, style, and color of the border **below** the H1 element. Omitted values are set to their initial values. Since the following rule does not specify a border color, the border will have the color specified by the 'color' property:

```
H1 { border-bottom: thick solid }
```

'border'

Value:	[<border-width> || <border-style> || <'border-top-color'>] | inherit
Initial:	see individual properties
Applies to:	all elements
Inherited:	no
Percentages:	N/A
Media:	visual
Computed value:	see individual properties

The 'border' property is a shorthand property for setting the same width, color, and style for all four borders of a box. Unlike the shorthand 'margin' and 'padding' properties, the 'border' property cannot set different values on the four borders. To do so, one or more of the other border properties must be used.

For example, the first rule below is equivalent to the set of four rules shown after it:

```
p { border: solid red }
p {
  border-top: solid red;
  border-right: solid red;
  border-bottom: solid red;
  border-left: solid red
}
```

Since, to some extent, the properties have overlapping functionality, the order in which the rules are specified is important.

Consider this example:

```
blockquote {
  border: solid red;
  border-left: double;
  color: black;
}
```

In the above example, the color of the left border is black, while the other borders are red. This is due to 'border-left' setting the width, style, and color. Since the color value is not given by the 'border-left' property, it will be taken from the 'color' property. The fact that the 'color' property is set after the 'border-left' property is not relevant.

8.6 The box model for inline elements in bidirectional context

For each line box, UAs must take the inline boxes generated for each element and render the margins, borders and padding in visual order (not logical order).

When the element's 'direction' property is 'ltr', the left-most generated box of the first line box in which the element appears has the left margin, left border and left padding, and the right-most generated box of the last line box in which the element appears has the right padding, right border and right margin.

When the element's 'direction' property is 'rtl', the right-most generated box of the first line box in which the element appears has the right padding, right border and right margin, and the left-most generated box of the last line box in which the element appears has the left margin, left border and left padding.