

Лабораторна робота №7
Тема. Проектування *Web*-документів. Оформлення - CSS.

Мета. Оформлення *Web*-сторінки за допомогою таблиці каскадних стилів (CSS).
 Селектори. Правила. Директиви.

Селектори.



Завдання до лабораторної роботи №7

Розробити файл(и) з таблицею каскадних стилів для сторінок із попередніх лабораторних робіт. При необхідності сторінки переробити таким чином щоб на сторінці було декілька рівнів вкладеності блокових тегів.

При розробці таблиц каскадних стилів студент може використовувати будь-які селектори та правила CSS, але для кожного студента, згідно варіанту, задається набір обов'язкових селекторів, правил та їх атрибутів і значень, що повинні бути присутні.

Варіанти задань

Варіант	обов'язкові селектори, правила та значення
1	Селектори, псевдокласи та псевдоелементи: Селектори - ідентифікаторів, Селектори - тегів; Селектори - дітей (дочірний), Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr^=value] ; Псевдокласи::first-child, Псевдокласи::last-of-type, Псевдокласи::enabled, Псевдокласи::active , Псевдокласи::lang, Псевдокласи::last-child, Псевдокласи::disabled; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки: margin-right:<розмір>, margin-right:<процент> , margin-top:<розмір>, margin-top:<процент> , margin:<розмір>, margin:<процент> ; padding-left:<розмір>, padding-top:<розмір>, padding-right:<розмір>; border-width:thin , border-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-right-color:<колір>, border-top-color:<колір>, border-color:<колір>; border-style:none, border-style: groove, border-style: hidden, border-style: dotted, border-bottom-style:none, border-bottom-style: groove, border-bottom-style: hidden, border-bottom-style: dotted, border-right-style:none, border-right-style: groove, border-right-style: hidden, border-right-style: dotted; Позиціонування елементів:display: inline, display: inline-table, display: table-cell, display: table-column-group, display: compact, display: block; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:width, max-width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height:

	normal, line-height: <розмір>; vertical-align: baseline, vertical-align: sub, vertical-align: bottom, vertical-align: <розмір>; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline, outline-width; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: <процент> {1,2};
2	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - універсальні; Селектори - контекстний(спадкоємців), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи::first-of-type, Псевдокласи: :focus, Псевдокласи::active , Псевдокласи::checked, Псевдокласи::last-of-type, Псевдокласи::invalid; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі і рамки:margin-left:<розмір>, margin-left:auto, margin-bottom:<розмір>, margin-bottom:auto, margin-right:<розмір>, margin-right:auto; padding-left:<розмір>, padding:<розмір>, padding-bottom:<розмір>; border-left-width:thin , border-left-width: medium , border-right-width:thin , border-right-width: medium , border-bottom-width:thin , border-bottom-width: medium ; border-right-color:<колір>, border-color:<колір>, border- top-color:<колір>; border-style:none, border-style: groove, border-style: dotted, border-style: outset, border- top-style:none, border- top-style: groove, border- top-style: dotted, border- top-style: outset, border-right-style:none, border-right-style: groove, border-right-style: dotted, border-right-style: outset; Позиціонування елементів:display: inline, display: run-in, display: table-row, display: table-cell, display: table, display: table-column; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, max-width; max-height: <розмір>, max-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: top, vertical-align: text-bottom, vertical-align: text-top; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-style, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: right;
3	Селектори, псевдокласи та псевдоелементи:Селектори - універсальні, Селектори - класів; Селектори - контекстний(спадкоємців), Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr\$=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи::not(selector), Псевдокласи::lang(language), Псевдокласи::invalid, Псевдокласи: :visited, Псевдокласи::last-of-type, Псевдокласи::last-child; Псевдоелементи ::first-line, Псевдоелементи ::selection; !important; Границі, заповнювачі і рамки:margin-right:<розмір>, margin-right:auto, margin-left:<розмір>, margin-left:auto, margin-bottom:<розмір>, margin-bottom:auto; padding-right:<розмір>, padding-left:<розмір>, padding:<розмір>; border-width:thin , border-width: <розмір>, border-left-width:thin , border-left-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>; border-left-color:<колір>, border- top-color:<колір>, border-color:<колір>; border-right-style:none, border-right-style: dotted, border-right-style: solid, border-right-style: ridge, border-left-style:none, border-left-style: dotted, border-left-style: solid, border-left-style: ridge, border- top-style:none, border- top-style: dotted, border- top-style: solid, border- top-style: ridge; Позиціонування елементів:display: inline, display: table-caption, display: inline-table, display: table-cell, display: table-footer-group, display: table-row; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, max-width; min-height: <розмір>, min-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: top, vertical-align: super, vertical-align: <розмір>; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-style; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: center, background-position: bottom;
4	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - тегів; Селектори - контекстний(спадкоємців), Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr =value] ; Псевдокласи::first-child, Псевдокласи::link , Псевдокласи::disabled, Псевдокласи::first-of-type, Псевдокласи::in-range, Псевдокласи: :focus, Псевдокласи::hover; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі і рамки:margin-top:<розмір>, margin- top:<процент> , margin:<розмір>, margin:<процент> , margin-right:<розмір>, margin-right:<процент> ; padding:<розмір>, padding- top:<розмір>, padding-

	<p>bottom:<розмір>; border-left-width:thin , border-left-width: thick , border-right-width:thin , border-right-width: thick , border-top-width:thin , border-top-width: thick ; border-bottom-color:<колір>, border-left-color:<колір>, border-right-color:<колір>; border-bottom-style:none, border-bottom-style: double, border-bottom-style: groove, border-bottom-style: outset, border-top-style:none, border-top-style: double, border-top-style: groove, border-top-style: outset, border-style:none, border-style: double, border-style: groove, border-style: outset; Позиціонування елементів:display: inline, display: table-column, display: table-header-group, display: table-row-group, display: block, display: list-item; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; min-height: <розмір>, min-height:, max-height: <розмір>, max-height:none; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: <процент>, vertical-align: text-top, vertical-align: super; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: right, background-position: <процент> {1,2};</p>
5	<p>Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - класів; Селектори - контекстний(спадкоємців), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи::enabled, Псевдокласи::last-child, Псевдокласи::last-of-type, Псевдокласи::lang(language), Псевдокласи::lang, Псевдокласи::empty; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі і рамки:margin-right:<розмір>, margin-right:auto, margin-bottom:<розмір>, margin-bottom:auto, margin-left:<розмір>, margin-left:auto; padding-right:<розмір>, padding-bottom:<розмір>, padding-left:<розмір>; border-width:thin , border-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-bottom-color:<колір>, border-right-color:<колір>, border-left-color:<колір>; border-right-style:none, border-right-style: groove, border-right-style: dashed, border-right-style: ridge, border-top-style:none, border-top-style: groove, border-top-style: dashed, border-top-style: ridge, border-style:none, border-style: groove, border-style: dashed, border-style: ridge; Позиціонування елементів:display: inline, display: table-footer-group, display: table-header-group, display: compact, display: run-in, display: inline-table; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, width; min-height: <розмір>, min-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: sub, vertical-align: text-bottom, vertical-align: bottom; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: right, background-position: <процент> {1,2};</p>
6	<p>Селектори, псевдокласи та псевдоелементи:Селектори - універсальні, Селектори - тегів; Селектори - контекстний(спадкоємців), Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи::last-child, Псевдокласи::enabled, Псевдокласи::disabled, Псевдокласи::link , Псевдокласи::not(selector), Псевдокласи::last-of-type; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі і рамки:margin-left:<розмір>, margin-left:auto, margin-right:<розмір>, margin-right:auto, margin:<розмір>, margin:auto; padding-left:<розмір>, padding-right:<розмір>, padding-bottom:<розмір>; border-bottom-width:thin , border-bottom-width: <розмір>, border-top-width:thin , border-top-width: <розмір>, border-width:thin , border-width: <розмір>; border-left-color:<колір>, border-top-color:<колір>, border-right-color:<колір>; border-left-style:none, border-left-style: hidden, border-left-style: solid, border-left-style: dotted, border-style:none, border-style: hidden, border-style: solid, border-style: dotted, border-right-style:none, border-right-style: hidden, border-right-style: solid, border-right-style: dotted; Позиціонування елементів:display: inline, display: table, display: table-column, display: table-caption, display: table-header-group, display: table-row-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:width, max-width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: sub, vertical-align: top, vertical-align: text-top; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір></p>

	{1,2}, background-position: center, background-position: bottom;
7	<p>Селектори, псевдокласи та псевдоелементи: Селектори - ідентифікаторів, Селектори - тергів; Селектори - дітей (дочірний), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~value] , Селектори - атрибутів:[attr =value] ; Псевдокласи::first-child, Псевдокласи::enabled, Псевдокласи: :focus, Псевдокласи::checked, Псевдокласи::link , Псевдокласи: :visited, Псевдокласи::first-of-type; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі і рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin-right:<розмір>, margin-right:<процент> , margin:<розмір>, margin:<процент> ; padding-bottom:<розмір>, padding-top:<розмір>, padding:<розмір>; border-top-width:thin , border-top-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-width:thin , border-width: <розмір>; border-color:<колір>, border-bottom-color:<колір>, border-right-color:<колір>; border-top-style:none, border-top-style: inset, border-top-style: dashed, border-top-style: hidden, border-left-style:none, border-left-style: inset, border-left-style: dashed, border-left-style: hidden, border-right-style:none, border-right-style: inset, border-right-style: dashed, border-right-style: hidden; Позиціонування елементів:display: inline, display: table-footer-group, display: table, display: none, display: table-column, display: run-in; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: sub, vertical-align: text-top, vertical-align: super; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline, outline-style; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: center;</p>
8	<p>Селектори, псевдокласи та псевдоелементи:Селектори - класів, Селектори - тергів; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr =value] , Селектори - атрибутів:[attr^=value] ; Псевдокласи::first-child, Псевдокласи::in-range, Псевдокласи::empty, Псевдокласи::first-of-type, Псевдокласи::active , Псевдокласи::invalid, Псевдокласи::last-of-type; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі і рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin-left:<розмір>, margin-left:<процент> , margin:<розмір>, margin:<процент> ; padding:<розмір>, padding-left:<розмір>, padding-bottom:<розмір>; border-right-width:thin , border-right-width: thick , border-bottom-width:thin , border-bottom-width: thick , border-top-width:thin , border-top-width: thick ; border-left-color:<колір>, border-bottom-color:<колір>, border-right-color:<колір>; border-right-style:none, border-right-style: ridge, border-right-style: inset, border-right-style: double, border-left-style:none, border-left-style: ridge, border-left-style: inset, border-left-style: double, border-top-style:none, border-top-style: ridge, border-top-style: inset, border-top-style: double; Позиціонування елементів:display: inline, display: table-row, display: compact, display: block, display: table-column-group, display: table-header-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; height: <розмір>, height: <процент>, max-height: <розмір>, max-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: bottom, vertical-align: super, vertical-align: text-top; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: bottom;</p>
9	<p>Селектори, псевдокласи та псевдоелементи:Селектори - універсальні, Селектори - класів; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::active , Псевдокласи::last-child, Псевдокласи::empty, Псевдокласи::link , Псевдокласи::not(selector), Псевдокласи::checked; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі і рамки:margin:<розмір>, margin:auto, margin-bottom:<розмір>, margin-bottom:auto, margin-top:<розмір>, margin-top:auto; padding-right:<розмір>, padding-top:<розмір>, padding-left:<розмір>; border-left-width:thin , border-left-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-width:thin , border-width: <розмір>; border-color:<колір>, border-bottom-color:<колір>,</p>

	border-top-color:<колір>; border-right-style:none, border-right-style: double, border-right-style: dashed, border-right-style: hidden, border-top-style:none, border-top-style: double, border-top-style: dashed, border-top-style: hidden, border-left-style:none, border-left-style: double, border-left-style: dashed, border-left-style: hidden; Позиціонування елементів:display: inline, display: compact, display: table-footer-group, display: table-cell, display: inline-table, display: marker; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; max-height: <розмір>, max-height:none, min-height: <розмір>, min-height:; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: middle, vertical-align: <розмір>, vertical-align: text-bottom; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline, outline-style; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: top;
10	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - класів; Селектори - контекстний(спадкоємців), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::lang(language), Псевдокласи: :visited, Псевдокласи::link , Псевдокласи::enabled, Псевдокласи::last-of-type, Псевдокласи::disabled; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі і рамки:margin-right:<розмір>, margin-right:<процент> , margin:<розмір>, margin:<процент> , margin-bottom:<розмір>, margin-bottom:<процент> ; padding-top:<розмір>, padding-right:<розмір>, padding-left:<розмір>; border-width:thin , border-width: <розмір>, border-top-width:thin , border-top-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-right-color:<колір>, border-left-color:<колір>, border-bottom-color:<колір>; border-right-style:none, border-right-style: groove, border-right-style: outset, border-right-style: double, border-left-style:none, border-left-style: groove, border-left-style: outset, border-left-style: double, border-top-style:none, border-top-style: groove, border-top-style: outset, border-top-style: double; Позиціонування елементів:display: inline, display: list-item, display: table-column, display: table, display: table-caption, display: table-cell; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, min-width; height: <розмір>, height: auto, max-height: <розмір>, max-height:none; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: super, vertical-align: sub, vertical-align: <розмір>; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-style, outline-width; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: <процент> {1,2}, background-position: top;
11	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - класів; Селектори - сусідів, Селектори - контекстний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~=value] , Селектори - атрибутів:[attr*=value] ; Псевдокласи::first-child, Псевдокласи::last-child, Псевдокласи::enabled, Псевдокласи::invalid, Псевдокласи: :visited, Псевдокласи::last-of-type, Псевдокласи::empty; Псевдоелементи ::first-line, Псевдоелементи ::selection; !important; Границі, заповнювачі і рамки:margin-left:<розмір>, margin-left:<процент> , margin-top:<розмір>, margin-top:<процент> , margin-right:<розмір>, margin-right:<процент> ; padding:<розмір>, padding-top:<розмір>, padding-left:<розмір>; border-right-width:thin , border-right-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>, border-width:thin , border-width: <розмір>; border-top-color:<колір>, border-bottom-color:<колір>, border-left-color:<колір>; border-style:none, border-style: dotted, border-style: dashed, border-style: groove, border-left-style:none, border-left-style: dotted, border-left-style: dashed, border-left-style: groove, border-top-style:none, border-top-style: dotted, border-top-style: dashed, border-top-style: groove; Позиціонування елементів:display: inline, display: none, display: table-caption, display: table-column, display: table-footer-group, display: run-in; position: static, position: fixed; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; height: <розмір>, height: auto, max-height: <розмір>, max-height:none; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: middle, vertical-align: top, vertical-align: text-bottom; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: top;

12	<p>Селектори, псевдокласи та псевдоелементи: Селектори - універсальні, Селектори - класів; Селектори - сусідів, Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr^=value] ; Псевдокласи::first-child, Псевдокласи::last-of-type, Псевдокласи::invalid, Псевдокласи::focus, Псевдокласи::active , Псевдокласи::in-range, Псевдокласи::checked; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі и рамки:margin:<розмір>, margin:<процент> , margin-top:<розмір>, margin-top:<процент> , margin-right:<розмір>, margin-right:<процент> ; padding-right:<розмір>, padding-top:<розмір>, padding-left:<розмір>; border-right-width:thin , border-right-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-color:<колір>, border-bottom-color:<колір>, border-top-color:<колір>; border-right-style:none, border-right-style: double, border-right-style: dotted, border-right-style: inset, border-bottom-style:none, border-bottom-style: double, border-bottom-style: dotted, border-bottom-style: inset, border-top-style:none, border-top-style: double, border-top-style: dotted, border-top-style: inset; Позиціонування елементів:display: inline, display: inline-table, display: table-footer-group, display: compact, display: table, display: table-caption; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:min-width, max-width; max-height: <розмір>, max-height:none, height: <розмір>, height: auto; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: middle, vertical-align: <процент>, vertical-align: sub; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-style, outline-width; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: <процент> {1,2}, background-position: left;</p>
13	<p>Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - універсальні; Селектори - дітей (дочірний), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr =value] ; Псевдокласи::first-child, Псевдокласи: :visited, Псевдокласи::first-of-type, Псевдокласи::lang, Псевдокласи::lang(language), Псевдокласи::link , Псевдокласи::last-of-type; Псевдоелементи ::first-line, Псевдоелементи ::selection; !important; Границі, заповнювачі и рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin:<розмір>, margin:<процент> , margin-right:<розмір>, margin-right:<процент> ; padding:<розмір>, padding-top:<розмір>, padding-bottom:<розмір>; border-bottom-width:thin , border-bottom-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-top-color:<колір>, border-top-color:<колір>, border-right-color:<колір>; border-bottom-style:none, border-bottom-style: groove, border-bottom-style: hidden, border-bottom-style: ridge, border-top-style:none, border-top-style: groove, border-top-style: hidden, border-top-style: ridge, border-left-style:none, border-left-style: groove, border-left-style: hidden, border-left-style: ridge; Позиціонування елементів:display: inline, display: inline-table, display: none, display: marker, display: compact, display: table-column; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; max-height: <розмір>, max-height:none, min-height: <розмір>, min-height:; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: bottom, vertical-align: <розмір>, vertical-align: text-bottom; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-style; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: <процент> {1,2}, background-position: center;</p>
14	<p>Селектори, псевдокласи та псевдоелементи:Селектори - класів, Селектори - ідентифікаторів; Селектори - дітей (дочірний), Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr\$=value] , Селектори - атрибутів:[attr^=value] ; Псевдокласи::first-child, Псевдокласи: :visited, Псевдокласи::last-child, Псевдокласи::first-of-type, Псевдокласи: :focus, Псевдокласи::active , Псевдокласи::link ; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі и рамки:margin-right:<розмір>, margin-right:auto, margin-bottom:<розмір>, margin-bottom:auto, margin-top:<розмір>, margin-top:auto; padding-top:<розмір>, padding-right:<розмір>, padding-left:<розмір>; border-bottom-width:thin , border-bottom-width: <розмір>, border-top-width: <розмір>, border-top-width: <розмір>, border-top-color:<колір>, border-color:<колір>, border-left-color:<колір>; border-style:none, border-style: ridge, border-style: double, border-style: inset, border-top-style:none, border-top-style: ridge, border-top-style: double, border-top-style: inset, border-left-style:none, border-left-style: ridge, border-left-style: double, border-left-</p>

	style: inset; Позиціювання елементів:display: inline, display: table-row, display: table-column-group, display: run-in, display: none, display: table-cell; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:width, min-width; min-height: <розмір>, min-height:, max-height: <розмір>, max-height:none; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <процент>, vertical-align: sub, vertical-align: text-top; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline-color; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: top;
15	Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - класів; Селектори - дітей (дочірний), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::active , Псевдокласи: :focus, Псевдокласи::last-of-type, Псевдокласи::first-of-type, Псевдокласи::invalid, Псевдокласи::lang(language); Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі і рамки:margin- top:<розмір>, margin- top:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-bottom:<розмір>, margin-bottom:<процент> ; padding- top:<розмір>, padding-right:<розмір>, padding-left:<розмір>; border- top-width:thin , border- top-width: <розмір>, border-width:thin , border-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-left-color:<колір>, border-bottom-color:<колір>, border-right-color:<колір>; border-left-style:none, border-left-style: hidden, border-left-style: outset, border-left-style: ridge, border-right-style:none, border-right-style: hidden, border-right-style: outset, border-right-style: ridge, border-bottom-style:none, border-bottom-style: hidden, border-bottom-style: outset, border-bottom-style: ridge; Позиціювання елементів:display: inline, display: table-caption, display: list-item, display: none, display: table-column-group, display: run-in; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, min-width; max-height: <розмір>, max-height:none, height: <розмір>, height: auto; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: super, vertical-align: <розмір>, vertical-align: bottom; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-style, outline-width; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: right;
16	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - універсальні; Селектори - сусідів, Селектори - контекстний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr\$=value] ; Псевдокласи::first-child, Псевдокласи::disabled, Псевдокласи::invalid, Псевдокласи::lang(language), Псевдокласи::empty, Псевдокласи::not(selector), Псевдокласи: :focus; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі і рамки:margin-right:<розмір>, margin-right:auto, margin:<розмір>, margin:auto, margin-left:<розмір>, margin-left:auto; padding- top:<розмір>, padding:<розмір>, padding-right:<розмір>; border-right-width:thin , border-right-width: <розмір>, border-left-width:thin , border-left-width: <розмір>, border- top-width:thin , border- top-width: <розмір>; border- top-color:<колір>, border-bottom-color:<колір>, border-color:<колір>; border-bottom-style:none, border-bottom-style: double, border-bottom-style: solid, border-bottom-style: dotted, border-style:none, border-style: double, border-style: solid, border-style: dotted, border- top-style:none, border- top-style: double, border- top-style: solid, border- top-style: dotted; Позиціювання елементів:display: inline, display: list-item, display: none, display: table-cell, display: run-in, display: table-column-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:width, min-width; min-height: <розмір>, min-height:, max-height: <розмір>, max-height:none; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: text-bottom, vertical-align: text-top, vertical-align: <процент>; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: center;
17	Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - класів; Селектори - контекстний(спадкоємців), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr\$=value] ; Псевдокласи::first-child, Псевдокласи::lang(language), Псевдокласи::first-of-type, Псевдокласи::invalid, Псевдокласи::link , Псевдокласи::last-child, Псевдокласи::in-range; Псевдоелементи ::first-

	<p>line, Псевдоелементи ::selection; !important; Границі, заповнювачі и рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-left:<розмір>, margin-left:<процент> ; padding-bottom:<розмір>, padding-top:<розмір>, padding-left:<розмір>; border-right-width:thin , border-right-width: thick , border-top-width:thin , border- top-width: thick , border-bottom-width:thin , border-bottom-width: thick ; border-color:<колір>, border-right-color:<колір>, border-left-color:<колір>; border-bottom-style:none, border-bottom-style: outset, border-bottom-style: dashed, border-bottom-style: solid, border-left-style:none, border-left-style: outset, border-left-style: dashed, border-left-style: solid, border-style:none, border-style: outset, border-style: dashed, border-style: solid; Позиціонування елементів:display: inline, display: table, display: marker, display: table-column-group, display: none, display: table-row-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; min-height: <розмір>, min-height:, max-height: <розмір>, max-height:none; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <розмір>, vertical-align: text-top, vertical-align: super; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline, outline-color; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: top, background-position: <процент> {1,2};</p>
18	<p>Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - ідентифікаторів; Селектори - дітей (дочірний), Селектори - контекстний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr^=value] , Селектори - атрибутів:[attr*=value] ; Псевдокласи::first-child, Псевдокласи::enabled, Псевдокласи::empty, Псевдокласи::last-of-type, Псевдокласи::lang(language), Псевдокласи::lang, Псевдокласи::active ; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки:margin:<розмір>, margin:<процент> , margin-left:<розмір>, margin-left:<процент> , margin- top:<розмір>, margin- top:<процент> ; padding:<розмір>, padding-left:<розмір>, padding-bottom:<розмір>; border-width:thin , border-width: <розмір>, border- top-width:thin , border- top-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>; border-right-color:<колір>, border-left-color:<колір>, border- top-color:<колір>; border-bottom-style:none, border-bottom-style: double, border-bottom-style: hidden, border-bottom-style: groove, border-right-style:none, border-right-style: double, border-right-style: hidden, border-right-style: groove, border- top-style:none, border- top-style: double, border- top-style: hidden, border- top-style: groove; Позиціонування елементів:display: inline, display: none, display: table-cell, display: inline-table, display: block, display: table-row-group; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; min-height: <розмір>, min-height: <процент>, max-height: <розмір>, max-height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: middle, vertical-align: bottom, vertical-align: text-top; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline, outline-width; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: top;</p>
19	<p>Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - класів; Селектори - контекстний(спадкоємців), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr\$=value] ; Псевдокласи::first-child, Псевдокласи::hover, Псевдокласи::checked, Псевдокласи::disabled, Псевдокласи::first-of-type, Псевдокласи::invalid, Псевдокласи::last-child; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки:margin-left:<розмір>, margin-left:<процент> , margin-right:<розмір>, margin-right:<процент> , margin:<розмір>, margin:<процент> ; padding-left:<розмір>, padding-bottom:<розмір>, padding- top:<розмір>; border-left-width:thin , border-left-width: <розмір>, border-width:thin , border-width: <розмір>, border-right-width:thin , border-right-width: <розмір>; border-right-color:<колір>, border- top-color:<колір>, border-color:<колір>; border- top-style:none, border- top-style: double, border- top-style: outset, border- top-style: dashed, border-style:none, border-style: double, border-style: outset, border-style: dashed, border-left-style:none, border-left-style: double, border-left-style: outset, border-left-style: dashed; Позиціонування елементів:display: inline, display: list-item, display: table-column-group, display: table-caption, display: compact, display: marker; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, max-width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline,</p>

	vertical-align: bottom, vertical-align: super, vertical-align: <процент>; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline, outline-color; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: bottom;
20	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - тегів; Селектори - сусідів, Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи::active , Псевдокласи::in-range, Псевдокласи::hover, Псевдокласи::lang, Псевдокласи::first-of-type, Псевдокласи:::visited; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки:margin-left:<розмір>, margin-left:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-bottom:<розмір>, margin-bottom:<процент> ; padding-top:<розмір>, padding-left:<розмір>, padding:<розмір>; border-right-width:thin , border-right-width: <розмір>, border-top-width:thin , border-top-width: <розмір>, border-bottom-width:thin , border-bottom-width: <розмір>; border-right-color:<колір>, border-top-color:<колір>, border-left-color:<колір>; border-right-style:none, border-right-style: ridge, border-right-style: groove, border-right-style: solid, border-left-style:none, border-left-style: ridge, border-left-style: groove, border-left-style: solid, border-bottom-style:none, border-bottom-style: ridge, border-bottom-style: groove, border-bottom-style: solid; Позиціонування елементів:display: inline, display: list-item, display: none, display: marker, display: table-caption, display: table-footer-group; position: static, position: fixed; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, min-width; height: <розмір>, height: auto, min-height: <розмір>, min-height:; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: <процент>, vertical-align: bottom, vertical-align: super; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-style, outline-width; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: right, background-position: top;
21	Селектори, псевдокласи та псевдоелементи:Селектори - класів, Селектори - ідентифікаторів; Селектори - сусідів, Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::disabled, Псевдокласи::not(selector), Псевдокласи::active , Псевдокласи::checked, Псевдокласи::link , Псевдокласи::empty; Псевдоелементи ::first-line, Псевдоелементи ::after; !important; Границі, заповнювачі и рамки:margin-left:<розмір>, margin-left:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding-left:<розмір>, padding:<розмір>, padding-top:<розмір>; border-left-width:thin , border-left-width: <розмір>, border-top-width:thin , border-top-width: <розмір>, border-width:thin , border-width: <розмір>; border-color:<колір>, border-top-color:<колір>, border-bottom-color:<колір>; border-left-style:none, border-left-style: double, border-left-style: groove, border-left-style: outset, border-top-style:none, border-top-style: double, border-top-style: groove, border-top-style: outset, border-right-style:none, border-right-style: double, border-right-style: groove, border-right-style: outset; Позиціонування елементів:display: inline, display: table, display: table-caption, display: table-row, display: run-in, display: table-footer-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <розмір>; vertical-align: baseline, vertical-align: bottom, vertical-align: text-top, vertical-align: <розмір>; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-style, outline; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: center;
22	Селектори, псевдокласи та псевдоелементи:Селектори - класів, Селектори - тегів; Селектори - дітей (дочірний), Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::hover, Псевдокласи::not(selector), Псевдокласи::checked, Псевдокласи::empty, Псевдокласи::last-child, Псевдокласи::active ; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі и рамки:margin-bottom:<розмір>, margin-bottom:auto, margin-left:<розмір>, margin-left:auto, margin-right:<розмір>, margin-right:auto; padding-right:<розмір>, padding-left:<розмір>, padding-bottom:<розмір>; border-width:thin , border-width: medium , border-bottom-width:thin

	, border-bottom-width: medium , border-right-width:thin , border-right-width: medium ; border-right-color:<колір>, border-color:<колір>, border-left-color:<колір>; border-right-style:none, border-right-style: dashed, border-right-style: outset, border-right-style: inset, border-style:none, border-style: dashed, border-style: outset, border-style: inset, border-left-style:none, border-left-style: dashed, border-left-style: outset, border-left-style: inset; Позиціонування елементів:display: inline, display: table-header-group, display: table-row, display: compact, display: inline-table, display: table-column; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, width; min-height: <розмір>, min-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <розмір>, vertical-align: sub, vertical-align: top; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-style; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: left, background-position: right;
23	Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - класів; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr =value] ; Псевдокласи::first-child, Псевдокласи: :visited, Псевдокласи::active , Псевдокласи::in-range, Псевдокласи::enabled, Псевдокласи::not(selector), Псевдокласи::empty; Псевдоелементи ::first-line, Псевдоелементи ::selection; !important; Границі, заповнювачі и рамки:margin-top:<розмір>, margin-top:auto, margin-left:<розмір>, margin-left:auto, margin-bottom:<розмір>, margin-bottom:auto; padding-left:<розмір>, padding-top:<розмір>, padding-right:<розмір>; border-left-width:thin , border-left-width: medium , border-bottom-width:thin , border-bottom-width: medium , border-width:thin , border-width: medium ; border-right-color:<колір>, border-bottom-color:<колір>, border-left-color:<колір>; border-right-style:none, border-right-style: dashed, border-right-style: inset, border-right-style: hidden, border-style:none, border-style: dashed, border-style: inset, border-style: hidden, border-left-style:none, border-left-style: dashed, border-left-style: inset, border-left-style: hidden; Позиціонування елементів:display: inline, display: block, display: table-caption, display: run-in, display: compact, display: table-row; position: static, position: absolute; z-index: auto; float: left, float: right; clear: none, clear: right; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:max-width, width; min-height: <розмір>, min-height:, height: <розмір>, height: auto; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: bottom, vertical-align: <розмір>, vertical-align: text-top; overflow: visible, overflow: hidden; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-style, outline-width; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: right, background-position: bottom;
24	Селектори, псевдокласи та псевдоелементи:Селектори - ідентифікаторів, Селектори - тегів; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr ~=value] , Селектори - атрибутів:[attr= value] ; Псевдокласи::first-child, Псевдокласи::enabled, Псевдокласи::in-range, Псевдокласи::checked, Псевдокласи::hover, Псевдокласи::disabled, Псевдокласи::last-child; Псевдоелементи ::first-line, Псевдоелементи ::selection; !important; Границі, заповнювачі и рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding-bottom:<розмір>, padding-top:<розмір>, padding:<розмір>; border-left-width:thin , border-left-width: thick , border-top-width:thin , border-top-width: thick , border-right-width:thin , border-right-width: thick ; border-left-color:<колір>, border-bottom-color:<колір>, border-right-color:<колір>; border-left-style:none, border-left-style: dashed, border-left-style: double, border-left-style: inset, border-style:none, border-style: dashed, border-style: double, border-style: inset, border-bottom-style:none, border-bottom-style: dashed, border-bottom-style: double, border-bottom-style: inset; Позиціонування елементів:display: inline, display: table-row-group, display: table-column-group, display: table-header-group, display: table-footer-group, display: marker; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, width; max-height: <розмір>, max-height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <процент>, vertical-align: middle, vertical-align: sub; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline, outline-width; Колір фону:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position:

	<процент> {1,2}, background-position: right;
25	<p>Селектори, псевдокласи та псевдоелементи: Селектори - класів, Селектори - універсальні; Селектори - сусідів, Селектори - контекстний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr^=value] , Селектори - атрибутів:[attr\$=value] ; Псевдокласи::first-child, Псевдокласи::link , Псевдокласи::last-of-type, Псевдокласи::active , Псевдокласи::lang, Псевдокласи::lang(language), Псевдокласи::in-range; Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки:margin-left:<розмір>, margin-left:<процент> , margin:<розмір>, margin:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding:<розмір>, padding-left:<розмір>, padding-bottom:<розмір>; border-left-width:thin , border-left-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-top-width:thin , border-top-width: <розмір>; border-color:<колір>, border-left-color:<колір>, border-top-color:<колір>; border-right-style:none, border-right-style: inset, border-right-style: hidden, border-right-style: dotted, border-left-style:none, border-left-style: inset, border-left-style: hidden, border-left-style: dotted, border-bottom-style:none, border-bottom-style: inset, border-bottom-style: hidden, border-bottom-style: dotted; Позиціонування елементів:display: inline, display: list-item, display: table-caption, display: table-column-group, display: none, display: table-row-group; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів:width, max-width; min-height: <розмір>, min-height:, max-height: <розмір>, max-height:none; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <розмір>, vertical-align: text-bottom, vertical-align: <процент>; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline-color; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: top, background-position: left;</p>
26	<p>Селектори, псевдокласи та псевдоелементи: Селектори - універсальні, Селектори - тегів; Селектори - сусідів, Селектори - контекстний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr\$=value] , Селектори - атрибутів:[attr*=value] ; Псевдокласи::first-child, Псевдокласи::invalid, Псевдокласи::in-range, Псевдокласи::hover, Псевдокласи::enabled, Псевдокласи::link , Псевдокласи::not(selector); Псевдоелементи ::first-line, Псевдоелементи ::before; !important; Границі, заповнювачі и рамки:margin-bottom:<розмір>, margin-bottom:<процент> , margin-left:<розмір>, margin-left:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding-bottom:<розмір>, padding-top:<розмір>, padding-left:<розмір>; border-width:thin , border-width: thick , border-right-width:thin , border-right-width: thick , border-left-width:thin , border-left-width: thick ; border-bottom-color:<колір>, border-left-color:<колір>, border-color:<колір>; border-left-style:none, border-left-style: double, border-left-style: inset, border-left-style: groove, border-right-style:none, border-right-style: double, border-right-style: inset, border-right-style: groove, border-top-style:none, border-top-style: double, border-top-style: inset, border-top-style: groove; Позиціонування елементів:display: inline, display: table-header-group, display: table-cell, display: compact, display: none, display: table-column; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, width; height: <розмір>, height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: bottom, vertical-align: sub, vertical-align: super; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-color, outline; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: center;</p>
27	<p>Селектори, псевдокласи та псевдоелементи: Селектори - ідентифікаторів, Селектори - класів; Селектори - дітей (дочірний), Селектори - сусідів; Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr =value] , Селектори - атрибутів:[attr^=value] ; Псевдокласи::first-child, Псевдокласи::last-child, Псевдокласи::visited, Псевдокласи::empty, Псевдокласи::in-range, Псевдокласи::not(selector), Псевдокласи::active ; Псевдоелементи ::first-line, Псевдоелементи ::first-letter; !important; Границі, заповнювачі и рамки:margin-top:<розмір>, margin-top:auto, margin:<розмір>, margin:auto, margin-right:<розмір>, margin-right:auto; padding-top:<розмір>, padding-left:<розмір>, padding-right:<розмір>; border-left-width:thin , border-left-width: <розмір>, border-top-width:thin , border-top-width: <розмір>,</p>

	border-right-width:thin , border-right-width: <розмір>; border-color:<колір>, border-left-color:<колір>, border-bottom-color:<колір>; border-left-style:none, border-left-style: ridge, border-left-style: hidden, border-left-style: double, border-bottom-style:none, border-bottom-style: ridge, border-bottom-style: hidden, border-bottom-style: double, border-right-style:none, border-right-style: ridge, border-right-style: hidden, border-right-style: double; Позиціювання елементів:display: inline, display: table-cell, display: marker, display: table-column-group, display: table, display: table-row; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:min-width, width; max-height: <розмір>, max-height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <розмір>, vertical-align: <процент>, vertical-align: bottom; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline-color; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: bottom, background-position: right;
28	Селектори, псевдокласи та псевдоелементи:Селектори - універсальні, Селектори - тегів; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr*=value] , Селектори - атрибутів:[attr=value] ; Псевдокласи::first-child, Псевдокласи::link , Псевдокласи::checked, Псевдокласи: :focus, Псевдокласи::empty, Псевдокласи::last-of-type, Псевдокласи::lang; Псевдоелементи ::first-line, Псевдоелементи :::selection; !important; Границі, заповнювачі и рамки:margin-left:<розмір>, margin-left:<процент> , margin-right:<розмір>, margin-right:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding-left:<розмір>, padding-top:<розмір>, padding:<розмір>; border-width:thin , border-width: thick , border-bottom-width:thin , border-bottom-width: thick , border-top-width:thin , border-top-width: thick ; border-right-color:<колір>, border-left-color:<колір>, border-bottom-color:<колір>; border-style:none, border-style: dashed, border-style: hidden, border-style: ridge, border-left-style:none, border-left-style: dashed, border-left-style: hidden, border-left-style: ridge, border-bottom-style:none, border-bottom-style: dashed, border-bottom-style: hidden, border-bottom-style: ridge; Позиціювання елементів:display: inline, display: table, display: table-cell, display: none, display: run-in, display: marker; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; min-height: <розмір>, min-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: <процент>, vertical-align: bottom, vertical-align: text-bottom; overflow: visible, overflow: auto; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: collapse; outline-width, outline-style; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: <процент> {1,2}, background-position: left;
29	Селектори, псевдокласи та псевдоелементи:Селектори - тегів, Селектори - ідентифікаторів; Селектори - сусідів, Селектори - контексний(спадкоємців); Селектори - атрибутів:[attr] , Селектори - атрибутів:[attr=value] , Селектори - атрибутів:[attr ~=value] ; Псевдокласи::first-child, Псевдокласи: :visited, Псевдокласи::last-child, Псевдокласи: :focus, Псевдокласи::link , Псевдокласи::invalid, Псевдокласи::empty; Псевдоелементи ::first-line, Псевдоелементи :::selection; !important; Границі, заповнювачі и рамки:margin-right:<розмір>, margin-right:<процент> , margin:<розмір>, margin:<процент> , margin-top:<розмір>, margin-top:<процент> ; padding-bottom:<розмір>, padding:<розмір>, padding-right:<розмір>; border-top-width:thin , border-top-width: <розмір>, border-right-width:thin , border-right-width: <розмір>, border-left-width:thin , border-left-width: <розмір>; border-color:<колір>, border-top-color:<колір>, border-right-color:<колір>; border-top-style:none, border-top-style: solid, border-top-style: double, border-top-style: dotted, border-style:none, border-style: solid, border-style: double, border-style: dotted, border-right-style:none, border-right-style: solid, border-right-style: double, border-right-style: dotted; Позиціювання елементів:display: inline, display: table-column, display: table-header-group, display: block, display: table-caption, display: table-row; position: static, position: relative; z-index: auto; float: left, float: none; clear: none, clear: left; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів:max-width, width; max-height: <розмір>, max-height: <процент>, min-height: <розмір>, min-height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: sub, vertical-align: bottom, vertical-align: text-bottom; overflow: visible, overflow: scroll; clip: rect(<top>,<right>,<bottom>,<left>); visibility: visible, visibility: hidden; outline-width, outline; Колір фон:color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll;

	background-position: <розмір> {1,2}, background-position: top, background-position: bottom;
30	Селектори, псевдокласи та псевдоелементи: Селектори - тегів, Селектори - ідентифікаторів; Селектори - контекстний (спадкоємців), Селектори - дітей (дочірний); Селектори - атрибутів: [attr], Селектори - атрибутів: [attr^=value], Селектори - атрибутів: [attr*=value]; Псевдокласи: :first-child, Псевдокласи: :in-range, Псевдокласи: :last-child, Псевдокласи: :hover, Псевдокласи: :enabled, Псевдокласи: :lang, Псевдокласи: :checked; Псевдоелементи: :first-line, Псевдоелементи: :after; !important; Границі, заповнювачі і рамки: margin-left: <розмір>, margin-left: <процент>, margin-bottom: <розмір>, margin-bottom: <процент>, margin-right: <розмір>, margin-right: <процент>; padding: <розмір>, padding-right: <розмір>, padding-bottom: <розмір>; border-width: thin, border-width: thick, border-bottom-width: thin, border-bottom-width: thick, border-top-width: thin, border-top-width: thick; border-bottom-color: <колір>, border-color: <колір>, border-right-color: <колір>; border-style: none, border-style: ridge, border-style: dotted, border-style: groove, border-bottom-style: none, border-bottom-style: ridge, border-bottom-style: dotted, border-bottom-style: groove, border-top-style: none, border-top-style: ridge, border-top-style: dotted, border-top-style: groove; Позиціонування елементів: display: inline, display: table-cell, display: table-row, display: inline-table, display: list-item, display: table-column-group; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: embed; Візуалізація елементів: width, max-width; max-height: <розмір>, max-height: none, height: <розмір>, height: auto; line-height: normal, line-height: <число>; vertical-align: baseline, vertical-align: top, vertical-align: sub, vertical-align: <процент>; overflow: visible, overflow: hidden; clip: rect(<top>, <right>, <bottom>, <left>); visibility: visible, visibility: collapse; outline-style, outline; Колір фон: color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: right, background-position: top;
31	Селектори, псевдокласи та псевдоелементи: Селектори - ідентифікаторів, Селектори - універсальні; Селектори - сусідів, Селектори - дітей (дочірний); Селектори - атрибутів: [attr], Селектори - атрибутів: [attr=value], Селектори - атрибутів: [attr\$=value]; Псевдокласи: :first-child, Псевдокласи: :disabled, Псевдокласи: :last-child, Псевдокласи: :link, Псевдокласи: :focus, Псевдокласи: :not(selector), Псевдокласи: :first-of-type; Псевдоелементи: :first-line, Псевдоелементи: :after; !important; Границі, заповнювачі і рамки: margin: <розмір>, margin: <процент>, margin-top: <розмір>, margin-top: <процент>, margin-left: <розмір>, margin-left: <процент>; padding-right: <розмір>, padding-top: <розмір>, padding-left: <розмір>; border-left-width: thin, border-left-width: thick, border-width: thin, border-width: thick, border-right-width: thin, border-right-width: thick; border-top-color: <колір>, border-left-color: <колір>, border-right-color: <колір>; border-bottom-style: none, border-bottom-style: hidden, border-bottom-style: solid, border-bottom-style: inset, border-style: none, border-style: hidden, border-style: solid, border-style: inset, border-right-style: none, border-right-style: hidden, border-right-style: solid, border-right-style: inset; Позиціонування елементів: display: inline, display: table-column-group, display: table-header-group, display: table-cell, display: table-row-group, display: list-item; position: static, position: fixed; z-index: auto; float: left, float: none; clear: none, clear: both; direction: ltr; unicode-bidi: normal, unicode-bidi: bidi-override; Візуалізація елементів: max-width, min-width; max-height: <розмір>, max-height: <процент>, height: <розмір>, height: <процент>; line-height: normal, line-height: <процент>; vertical-align: baseline, vertical-align: middle, vertical-align: text-bottom, vertical-align: super; overflow: visible, overflow: auto; clip: rect(<top>, <right>, <bottom>, <left>); visibility: visible, visibility: hidden; outline-color, outline-style; Колір фон: color: <колір>; background-color: <колір>; background-image: <uri>; background-attachment: scroll; background-position: <розмір> {1,2}, background-position: center, background-position: <процент> {1,2};

Додаток.

CSS (Cascading Style Sheets)

У грудні 1996 р. W3C був стандартизований перший рівень *каскадних таблиць стилів* (CSS1 - Cascading Style Sheets), що визначив правила опису стилів візуального відображення елементів HTML-документів.

У травні 1998 р. їм була прийнята стандарт другого рівня таблиць стилів (CSS2), що істотно розширив можливості таблиць стилів. Основними особливостями CSS2 є наступні:

- CSS2 — це мова, що дозволяє приєднувати стилі до будь-яких структурованих документів. На сьогодні такими є HTML-документи й XML-додатки.
- CSS2 поширив поняття стилю відображення на друкувальні пристрої, синтезатори мови й інші пристрої відображення документів.
- CSS3 – підтримка модульності, для пристроїв з обмеженими ресурсами, (Робочий проект W3C, 23 травня 2001) ...

В основу даної частини нашого довідника покладений саме стандарт CSS2.

Мета створення каскадних таблиць стилів полягала в тому, щоб відокремити структуру документа (описану мовою HTML або підмножині XML) від правил його відображення на різних пристроях (задава таблицями CSS). Щоб пояснити це твердження, розглянемо наступний приклад HTML-документа:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<HTML>
<HEAD>
<TITLE>Моя домашня сторінка</TITLE>
<STYLE type="text/css">
<!--i сховати стилі CSS від старих оглядачів
  BODY { color: blue }
  H1 { color: red }
--i>
</STYLE>
</HEAD>
<BODY>
<H1>Моя домашня сторінка</H1>
<P>Ласкаво просимо!</P>
</BODY>
</HTML>
```

Цей документ містить елемент **STYLE**, утримуючий два правила CSS: перше з них визначає, що кольори відображення елемента **BODY** повинен бути червоним, а друге — що кольори відображення елемента **H1** повинен бути синім. Оскільки стиль відображення елемента **P** не заданий, він успадкує стиль свого батьківського елемента, у цьому випадку елемента **BODY**. У результаті даний HTML-документ буде відображатися так:

Моя домашня сторінка

Ласкаво просимо!

Тепер для зміни стилю відображення цього документа нам досить міняти тільки вміст елемента **STYLE**, не вносячи ніяких модифікацій в інший HTML-документ. Для більшої гнучкості ми можемо створити текстовий файл **STYLE.CSS** і перенести в нього опис стилів:

```
BODY { color: blue }
H1 { color: red }
```

HTML-документ після цього варто змінити в такий спосіб:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<HTML>
<HEAD>
<TITLE>Моя домашня сторінка</TITLE>
<LINK rel="stylesheet" href="style.css" type="text/css">
</HEAD>
<BODY>
<H1>Моя домашня сторінка</H1>
```

```
<P>Ласкаво просимо!</P>
</BODY>
</HTML>
```

Надалі для зміни стилів ми будемо змінювати тільки файл STYLE.CSS, не стосуючись головного HTML-документа.

Тепер представимо цей документ у вигляді *дерева* й уведемо відповідну термінологію, що буде нам необхідна надалі викладі:

Коренем цього дерева є елемент **HTML**, який має двох *синів* — **HEAD** і **BODY**. Елемент **HEAD** є батьком елемента **TITLE**, а елемент **BODY** — батьком елементів **H1** і **P** (два останніх елементи називаються *братами*, причому **H1** є *старшим братом*, а **P** — *молодшим*). Всі елементи дерева є *нащадками* кореня, а той є їхнім *предком*. У такий спосіб можна представити у вигляді дерева будь-який документ, до якого застосовні правила мови CSS.

2.1.2. Включення таблиць стилів в HTML-документ

Є три способи завдання стилів в HTML-документі. Перелічимо їх у порядку переваги.

2.1.2.1. Зовнішні таблиці стилів

Для підключення до документа *зовнішньої таблиці стилів* (тобто таблиці стилів, що зберігається в окремому файлі) варто помістити в заголовок документа елемент **LINK**, як показано вище, наприклад:

```
<LINK rel="stylesheet" href="style.css" type="text/css">
```

В елементі **LINK** можна додатково вказати [типи устроїв](#), на які поширюється дана таблиця стилів, наприклад:

```
<LINK rel="stylesheet" href="style.css" type="text/css" media="screen,
print">
```

Зовнішні таблиці стилів варто використати в тому випадку, коли кілька HTML-документів користуються єдиною таблицею стилів.

2.1.2.2. Внутрішні таблиці стилів

Для включення в документ *внутрішньої таблиці стилів* варто помістити в заголовок документа елемент **STYLE**. Приклад:

```
<HEAD>
  <STYLE type="text/css">
    <!--i
      H1 {border-width: 1; border: solid; text-align: center}
    -i>
  </STYLE>
</HEAD>
<BODY>
  <H1>Цей заголовок має зазначений вище стиль</H1>
</BODY>
```

Внутрішні таблиці стилів варто використати в тому випадку, коли дана таблиця стилів використовується тільки в даному HTML-документі.

2.1.2.3. Таблиці стилів елементів

Останнім способом завдання стилів є визначення таблиці стилів окремого елемента шляхом завдання його атрибута **style**. Наприклад, стиль елемента [H1](#) з попереднього приклада міг би бути заданий і так:

```
<H1 style="border-width: 1; border: solid; text-align: center">  
Цей заголовок має зазначений стиль</H1>
```

Загалом кажучи, подібної практики варто уникати; вона прийнятна тільки в тому випадку, коли ваш документ містить єдиний елемент із даним набором стилів.

2.1.3. Синтаксис CSS

2.1.3.1. Кодування символів

Таблиця стилів CSS — це послідовність символів Unicode. Ці символи можуть зберігатися в будь-якому припустимому кодуванні, при дотриманні наступних правил. Внутрішні таблиці стилів повинні мати те ж кодування символів, що й документ у цілому. Кодування зовнішніх таблиць стилів визначається оглядачем у наступному порядку:

- спочатку аналізується **charset** у поле [Content-Type](#) метаописателя HTML-документа;
- якщо його ні, то директива **@charset**;
- якщо її ні, то посилання на таблицю стилів у документі (наприклад, атрибут **charset** елемента **LINK**).

Для включення в таблицю стилів символів, відсутніх у даному кодуванні, варто використати формат "\xxxx", де xxxx — шестнадцатеричний код символу в Unicode. Наприклад, код "\410" задає прописну російську букву А.

2.1.3.2. Директива @charset

Директива **@charset** задає кодування символів таблиці стилів, наприклад:

```
@charset "ISO-8859-1";
```

Дана директива повинна бути першою в таблиці стилів, причому таблиця повинна містити не більше однієї директиви **@charset**. Список припустимих кодувань символів наведений у [Приложенні 7](#), але оглядач не зобов'язаний підтримувати всі перераховані там кодування.

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

2.1.3.3. Оператори, директиви й правила

Таблиця стилів складається з набору *операторів*. При цьому кожен оператор є або *директивою*, або *правилом*. Оператори можуть розділятися *пробілами*. В CSS пробілами вважаються тільки наступні символи Unicode: пробіл (код 32), табуляція (код 9), переклад рядка (код 10), повернення каретки (код 13) і переклад формату (код 12).

Директива (at-rule) починається із символу at (@) і відповідного ключового слова. CSS2 містить наступні директиви:

Таблиця 2.1. Директиви мови CSS	
@charset	Задає кодировку символів .
@font-face	Задає опису шрифтів.

@import	Включає іншу таблицю стилів у поточну.
@media	Задає имена устройств отображения .
@page	Задає властивості сторінки для печатки .

Приклади директив:

```
@import "subs.css";
@media print {
  BODY { font-size: 10pt }
}
```

Правило (rule) складається із *селектора* й *блоку декларацій*, укладеного у фігурні дужки, наприклад:

```
H1 {
  font-weight: bold;
  font-size: 12pt;
  font-family: Helvetica;
  font-variant: normal;
  font-style: normal
}
```

Тут **H1** — це селектор, за яким треба блок декларацій. Кожна *декларація*, як видно з даного приклада, складається з *назви властивості* (наприклад, **font-weight**), символу двокрапки (:) і *значення властивості* (наприклад, **bold**). Декларації повинні розділятися крапкою з коми (;).

Відзначимо, що відповідно до визначення CSS всі його елементи не залежать від регістра. У такий спосіб ми можемо писати **h1** замість **H1**, **Font-Weight** замість **font-weight** і т.д. Проте з метою однаковості в даному довіднику ми пишемо імена селекторів прописними буквами, а назви властивостей — рядковими. Значення властивостей, які в документі можуть залежати від регістра, визначаються мовою документа. Так, назви тегів HTML не залежать від регістра, а назви тегів XML — залежать.

Таблиці стилів можуть містити *коментарі*. Коментар починається із символів /* і закінчується символами */. Вкладені коментарі не допускаються. Приклад:

```
H1 {
  font-weight: bold; /* жирність шрифту */
  font-size: 12pt; /* розмір шрифту */
  font-family: Helvetica; /* назва шрифту */
  font-variant: normal; /* варіант шрифту */
  font-style: normal /* стиль шрифту */
}
```

2.1.4. Типи даних CSS

Значення властивостей CSS можуть бути наступних типів.

Цілі й дійсні числа

Деякі властивості можуть мати цілі (далі позначаються як <ціле>) або дійсні (далі позначаються як <число>) значення. Ці значення можуть записуватися тільки в десятковій нотації. Ціле число складається з однієї або більше десяткових цифр 0-9. Дійсне число або є цілим, або складається з нуля або більше десяткових цифр, за яких треба крапка (.) і одна або більше десяткових цифр. І перед цілими, і перед дійсними числами може стояти знак: плюс (+) або мінус (-). Приклади:

Цілі числа	Дійсні числа
101	101

-21	1. 234567
1234567	-.999

Розміри

Розміри (далі позначаються як <розмір>) указують на вертикальний або горизонтальний розміри чого-небудь. Розмір задається як [<число>](#), за яким треба *одиниця виміру*. Якщо розмір дорівнює 0, то одиницю виміру можна не вказувати.

Одиниці виміру підрозділяються на *абсолютні* й *відносні*. Абсолютні одиниці виміру задають точний фізичний розмір, а відносні — указують розмір щодо іншого розміру. Їхні описи зведені в наступні таблиці:

Таблиця 2.2. Абсолютні одиниці виміру	
in	дюйми (1 дюйм = 2.54 див = 25.4 мм = 72 крапки = 6 пів)
cm	сантиметри (1 див = 10 мм = 0.39 дюйма = 2.36 піки = 28.35 крапки)
mm	міліметри (1 мм = 0.1 див = 0.039 дюйма = 0.24 піки = 2.84 крапки)
pt	крапки (1 крапка = 1/12 піки = 1/72 дюйма = 0.035 див = 0.35 мм)
pc	піки (1 піка = 12 крапок = 1/6 дюйма = 0.423 див = 4.23 мм)
Таблиця 2.3. Відносні одиниці виміру	
em	розмір (font-size) відповідного шрифту
ex	висота малих літер (x-height) відповідного шрифту
px	пиксели (розмір залежить від пристрою відображення)

Абсолютні одиниці виміру застосовні тільки тоді, коли нам відомі точні фізичні розміри пристрою відображення (наприклад, екрана дисплея або сторінки принтера). Тому в більшості випадків використовуються відносні одиниці, призначення яких варто пояснити докладніше.

Одиниці **em** й **ex** ґрунтуються на розмірі шрифту того елемента, до якого ставиться дана декларація. При цьому **em** задає *розмір* шрифту, тобто розмір його найбільшої букви (звичайно це буква 'М', звідси абревіатура em), а **ex** — *висоту малих літер* шрифту (звичайно це висота букви 'x', звідси англійська назва x-height й абревіатура ex). З іншого боку, одиниця **px** заснована на *розмірі пикселя* пристрою відображення (звичайно це дисплей). Пиксель — це крапка дисплея і її розмір залежить як від фізичних розмірів екрана, так і від його дозволу: пиксель на екрані з дозволом 640x480 буде більше, ніж на екрані з дозволом 1280x1024. Приклади завдання розмірів:

```
H1 { margin: 0.5em }
H1 { text-indent: 1ex }
H3 { font-size: 12px }
H1 { margin: 0.5in }
H2 { line-height: 3cm }
H3 { word-spacing: 4mm }
H4 { font-size: 12pt }
H4 { font-size: 1pc }
```

Процентні значення

Процентні значення (далі позначаються як <відсоток>) указують на величину чого-небудь у відсотках від іншої величини. Вони задаються як [<число>](#), за яким треба символ відсотка (%), наприклад:

```
H1 { font-size: 120% }
```

Усюди, де CSS допускає використання процентних значень, в описі відповідної властивості чітко вказується, яка саме величина є основою для обчислення значення властивості у відсотках.

URI

Цей тип значень (далі позначуваний як <uri>) задає посилання на [уніфіцированные идентификаторы ресурсов](#). URI задається у формі **url(*URI*)**, наприклад:

```
BODY { background: url(http://mysite.com/bg.gif) }
```

Текст усередині **url()** можна укласти в апострофи або лапки — це нічого не змінить, наприклад:

```
BODY { background: url("http://mysite.com/bg.gif") }
```

Кольори

Колірні значення (далі позначаються як <кольори>) можуть задаватися або шестнадцатеричним числом із префіксом "#" виду "#rrggbb", що задає RGB-код кольори, або одним з 16-ти символічних імен, наведених у [Таблице П9.1](#). CSS допускає використання короткої форми RGB-кодів виду "#rgb"; при цьому коротка форма перетвориться в повну повторенням цифр, а не додаванням нулів. Іншими словами, код #FA1 відповідає повному коду #FFAA11. Крім того, RGB-код кольорів може бути заданий конструкцією **rgb(*r,g,b*)**, наприклад, що впливають декларації задають той самий червоні кольори:

```
EM { color: red }           /* назва кольорів */
EM { color: #f00 }          /* #rgb */
EM { color: #ff0000 }       /* #rrggbb */
EM { color: rgb(255,0,0) }   /* цілі в діапазоні 0 до 255 */
EM { color: rgb(100%,0%,0%) } /* дійсні в діапазоні від 0.0% до 100.0% */
*/
```

Примітка. Оглядачі Microscape додатково підтримують імена квітів, перераховані в [Таблице П9.2](#), але ми рекомендуємо використати замість них відповідні шестнадцатеричні значення.

Крім стандартних імен квітів CSS підтримує назви *системних квітів*, призначених для використання відповідно до графічного користувацького інтерфейсу операційних систем клієнтів. Ці назви і їхнє призначення наведені в наступній таблиці (значення цих квітів, прийняті за замовчуванням у системі Windows, зазначені в [Таблице П9.3](#)).

Таблиці 2.4. Системні кольори CSS			
activeborder	Кольори рамки активного вікна	inactivecaptiontext	Кольори тексту заголовка неактивного вікна
activecaption	Кольори тла	infobackground	Кольори тла підказок (tooltips)
appworkspace	Кольори тла многооконой програми	infotext	Кольори
background	Кольори тла системної підкладки (desktop)	menu	Кольори тла меню
buttonface	Кольори тривимірних кнопок	menutext	Кольори
buttonhighlight	Кольори виділених тривимірних кнопок	scrollbar	Кольори тла смуг прокручування
buttonshadow	Кольори тіні тривимірних кнопок	threeddarkshadow	
buttontext	Кольори	threedface	Кольори тривимірних елементів
captiontext	Кольори тексту заголовків	threedhighlight	Кольори виділених тривимірних елементів

graytext	Сірі кольори (для заборонених елементів)	threeilightshadow	Світла тінь тривимірних елементів
highlight	Кольори тла	threeedshadow	Тінь тривимірних елементів
highlighttext	Кольори тексту виділених елементів	window	Кольори тла вікон
inactiveborder	Кольори рамки неактивного вікна	windowframe	Кольори
inactivecaption	Кольори тла заголовка неактивного вікна	windowtext	Кольори

Рядка

Текстові рядки (далі позначаються як <рядок>) можуть бути укладені або в лапки, або в апострофи. При цьому якщо рядок укладений у лапки, для включення в неї символу лапок потрібно використати форму \" або \22. Аналогічно, якщо рядок укладений в апострофи, для включення в неї символу апострофа потрібно використати форму \' або \27. Для включення в рядок символу нового рядка використається форма \A (код перекладу рядка в Unicode). Приклади текстових рядків:

```
"це 'рядок'"
'це \'рядок\''
'це "рядок"'
"це \"рядок\\""
"це рядок,\A складається із двох рядків"
```

Лічильники

Лічильники (далі позначаються як <лічильник>) позначаються ідентифікаторами. Для добування значення лічильника використаються позначення **counter(ідентифікатор)** або **counter(ідентифікатор, стиль)**. За замовчуванням **стиль** дорівнює *decimal*. Для доступу до всіх лічильників з даним ім'ям використаються позначення **counters(ідентифікатор, рядок)** або **counters(ідентифікатор, рядок, стиль)**. Остання функція повертає значення всіх лічильників з даним ім'ям, що існують у цей момент, розділених текстом *рядок*.

Доступ до значення лічильників можливий тільки із властивості **content**. Подробиці див. у [п. 2.11.3](#).

Кутові величини

Кутові величини (далі позначаються як <кут>) використаються у звукових таблицях стилів для завдання просторових характеристик звуку. Кут задається як <число>, за яким треба *кутова одиниця виміру*. Якщо кут дорівнює 0, то одиницю виміру можна не вказувати. CSS підтримує наступні одиниці виміру кутів:

deg	градуси
grad	гради
rad	радіани

Ці одиниці співвідносяться в такий спосіб: 90 градусів = 100 градов = 1.570796326794897 радіан.
Приклади кутових величин:

```
H1 { azimuth: 45deg }
P { azimuth: -10deg }
H1 { elevation: 100grad }
P { elevation: 3.14rad }
```

Часи

Часи (далі позначаються як <час>) використовуються у звукових таблицях стилів для завдання тимчасових характеристик звуку. Час задається як <число>, за яким треба *одиниця виміру часу*. Якщо час дорівнює 0, то одиницю виміру можна не вказувати. CSS підтримує наступні одиниці виміру часу:

ms	миллисекунды (1 мс = 0.001 с)
s	секунди (1 с = 1000 мс)

Часи не можуть бути негативними. Приклади:

```
H1 { pause-before: 1s }
P { pause: 20ms }
```

Частоти

Частоти (далі позначаються як <частота>) використовуються у звукових таблицях стилів для завдання частотних характеристик звуку. Частота задається як <число>, за яким треба *одиниця виміру частоти*. Якщо частота дорівнює 0, то одиницю виміру можна не вказувати. CSS підтримує наступні одиниці виміру частот:

hz	герци (1 Гц = 0.001 кГц)
khz	кілогерци (1 кГц = 1000 Гц)

Частоти не можуть бути негативними. Приклади:

```
H1 { pitch: 200hz } /* басовий звук */
P { pitch: 6khz } /* тремтячий звук */
```

2.1.5. Типи пристроїв відображення

Для кожної властивості CSS визначає ті устройства отображения, до яких ця властивість застосовна. Із цією метою всі пристрої відображення класифіковані в CSS по наступних ознаках:

- візуальні, звукові й тактильні пристрої;
- безперервні й сторінкові пристрої;
- символні й графічні пристрої;
- інтерактивні й статичні пристрої;
- **all** — всі пристрої.

Відповідна розбивка пристроїв на групи виглядає так.

Таблиця 2.5. Типи пристроїв відображення				
Пристрій	Групи пристроїв			
	візуальне/ звукове/ тактильне	безперервне/ сторінкове	символьне/ графічне	інтерактивне/ статичне
aural	звукове	безперервне	—	і те, і інше
braille	тактильне	безперервне	символьне	і те, і інше
emboss	тактильне	сторінкове	символьне	і те, і інше
handheld	візуальне	і те, і інше	і те, і інше	і те, і інше
print	візуальне	сторінкове	графічне	статичне

projection	візуальне	сторінкове	графічне	статичне
screen	візуальне	безперервне	графічне	і те, і інше
tty	візуальне	безперервне	символьне	і те, і інше
tv	візуальне, звукове	і те, і інше	графічне	і те, і інше

2.1.6. Директива @media

Директива **@media** призначена для створення *таблиць стилів, що залежать від пристрою відображення*. Вона задає список пристроїв відображення, розділених комами, до яких застосовні правила, що втримуються в ній, ув'язнені у фігурні дужки. Якщо поточного пристрою відображення немає в списку, то вміст даної директиви повинне ігноруватися оглядачем. Приклади:

```
@media print {
  BODY { font-size: 10pt }
}
@media screen {
  BODY { font-size: 12pt }
}
@media screen, print {
  BODY { line-height: 1.2 }
}
```

Глава 2.2. Селектори, псевдокласи й псевдоелементи

2.2.1. Базові селектори

Як було сказано вище, кожне правило CSS складається із селектора й набору декларацій. Декларації визначають властивості відображення, а селектор указує, до яких саме елементів документа саме ці декларації повинні застосовуватися (такі елементи називаються *суб'єктами* цього селектора). Почнемо з формального визначення селекторів.

Базовий селектор

Універсальний селектор або селектор типу.

Універсальний селектор

Символ зірочка (*), що означає, що його суб'єктами є всі елементи документа. Якщо зірочка є не єдиний складовий селектора, то вона може опускатися.

Селектор типу

Збігається з ім'ям елемента в документі й указує, що його суб'єктами є всі елементи документа з даним ім'ям.

Простий селектор

Базовий селектор, за яким ідуть нуль або більше селекторів атрибутів, селекторів ідентифікаторів або псевдокласов (у будь-якому порядку).

Складений селектор

Утвориться із простих селекторів з'єднанням їх за допомогою спеціальних символів.

Як ми бачимо, основу цих визначень становлять *базові селектори*. Пояснимо їхнього визначення прикладами:

```
* { font-family: sans-serif }      /* Застосовується до всіх елементів
документа */
h1 { font-family: sans-serif }     /* Застосовується до всіх елементів
h1 */
```

Якщо кілька селекторів мають однакові декларації, то їх можна *згрупувати*, тобто об'єднати в одне правило, перелічивши селектори через кому. Наприклад, що впливає набір правил

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
```

еквівалентний одному правилу

```
h1, h2, h3 { font-family: sans-serif }
```

Підтримка: Повна відповідність стандарту (5.0+)

Не підтримує універсальні селектори (4.0+)

2.2.2. Селектори класів

Для HTML-документів CSS підтримує *селектори класів*, які засновані на використанні атрибута **class** і мають вигляд *element.class*. Нехай, наприклад, наша таблиця стилів містить правило

```
P.warning { font-style: italic }
```

Тоді наступний елемент HTML-документа

```
<P class="warning">Рекомендується вибирати назви класів,  
отражающие їх призначення</P>
```

буде відображатися курсивним шрифтом:

Рекомендується вибирати назви класів, що відбивають їхнє призначення.

Селектор класу може не містити назви елемента, наприклад

```
.warning { font-style: italic }
```

Таке правило ставиться до всіх елементів, що мають атрибут *class="warning"*.

Примітка. Селектор класу, не утримуючого імені елемента, варто розуміти як селектор **.class*, у якому універсальний селектор *** опущений.

CSS дозволяє задавати й підмножини значення атрибута **class**, наприклад правило

```
P.warning.red { font-style: italic }
```

буде застосовно до елементів з атрибутом *class="warning red blue"*, але не застосовно до елементів з атрибутом *class="warning blue"*.

Підтримка: Повна відповідність стандарту (5.0+)

Не підтримує множинні класи (4.0+)

2.2.3. Селектори ідентифікаторів

Документи можуть містити елементи з атрибутами ID, що задають унікальні ідентифікатори цих елементів. Для HTML-документів це атрибут **id**, у додатках XML назва відповідного атрибута визначається додатком. Відповідний *селектор ідентифікатора* в CSS має вигляд *element#id* або просто *#id*.

Примітка. Селектор ідентифікатора, не утримуючого імені елемента, варто розуміти як селектор **#id*, у якому універсальний селектор *** опущений.

У прикладі

```

<HEAD>
  <STYLE type="text/css">
    #p001 { letter-spacing: 0.3em }
  </STYLE>
</HEAD>
<BODY>
  <P id="p001">Розріджений текст</P>
</BODY>

```

правило стилю буде застосовано до елемента **P** з атрибутом *id="p001"*. З іншого боку, у наступному прикладі

```

<HEAD>
  <STYLE type="text/css">
    H1#p001 { letter-spacing: 0.3em }
  </STYLE>
</HEAD>
<BODY>
  <P id="p001">Широкий текст</P>
</BODY>

```

правило стилю застосовне тільки до елемента **H1** з атрибутом *id="p001"* і тому не буде застосовано до елемента **P** із цим атрибутом.

Підтримка: Повна відповідність стандарту (5.0+)

Повна відповідність стандарту (4.0+)

2.2.4. Селектори атрибутів

Тепер розглянемо *прості селектори*, які є уточненням базових селекторів. Вище ми вже познайомилися із двома типами простих селекторів, які застосовуються для завдання стилів HTML-документів: селектори класів і селектори ідентифікаторів. Тут ми познайомимося ще з однією групою селекторів, а саме із *селекторами атрибутів*. Ці селектори мають потужний і гнучкий синтаксис і призначені для роботи з будь-якими документами, які підтримуються мовою CSS. На жаль, вони поки не підтримуються Веб-обозревателями. Існують чотири види селекторів атрибутів:

[attr]

Застосовується до всіх елементів, що мають атрибут *attr*, незалежно від його значення.

[attr=value]

Застосовується до всіх елементів, чий атрибут *attr* має значення *value*.

[attr~=value]

Застосовується до всіх елементів, чий атрибут *attr* має складатися зі списку значень, розділених пробілами, і одне із цих значень дорівнює *value*.

[attr|=value]

Застосовується до всіх елементів, чий атрибут *attr* має значення, що складається з декількох "слів", розділених дефісом, причому перше із цих слів дорівнює *value* (спочатку призначалося для виділення коду основної мови з повного [кода мови](#)).

Значення атрибутів повинні бути ідентифікаторами або текстовими рядками. Чи залежать вони від регістра, визначається мовою документа. Приведемо приклади.

```

H1[title] { color: blue }          /* Застосовується до всіх елементів
H1, */                             /* имеющим атрибут title */

SPAN[class="example"] { color: blue } /* Застосовується до всіх
елементів SPAN, */                  /* имеющим атрибут class="example" */

SPAN[class~="example"] { color: blue } /* Еквівалентно селектору
SPAN.example */

```



```

*[lang="fr"] { display: none }           /* Застосовується до всіх
елементів, */
                                           /* имеющим атрибут lang="fr" */
*[lang|="en"] { color: red }           /* Застосовується до всіх елементів,
у яких */
                                           /* атрибут lang починається з "en"
(наприклад, */
                                           /* "en-us" або "us-gb") */

```

Підтримка: Не підтримуються

Не підтримуються

2.2.5. Селектори нащадків

Перейдемо тепер до розгляду *складених селекторів*, які утворюються із простих селекторів шляхом їхнього з'єднання в новий селектор. Найважливішими зі складених селекторів є *селектори нащадків*, тобто селектори тих елементів, які є нащадками іншого елемента в дереві документа. Розглянемо таке завдання: визначити стиль відображення всіх елементів **ЕМ**, які перебувають усередині елементів **Н1**. Для цього ми могли б написати наступний набір правил:

```

H1 { color: red }
ЕМ { color: red }
H1 ЕМ { color: blue }

```

Тут перші два правила визначають стилі відображення для всіх елементів **Н1** й **ЕМ** відповідно, а третє правило саме й вирішує наше завдання. Таким чином, для завдання селектора нащадка досить утворити селектор із селектора, що задає предка, і селектора, що задає нащадка, розділивши їхнім пробілом. Розглянемо ще один приклад:

```

DIV * ЕМ { color: blue }

```

Дане правило буде застосовуватися до всіх елементів **ЕМ**, які перебувають усередині будь-яких елементів (універсальний селектор), які укладені в елемент **DIV**.

Підтримка: Повна відповідність стандарту (5.0+)

Не підтримує універсальні селектори (4.0+)

2.2.6. Селектори дітей

CSS підтримує використання *селекторів дітей*, тобто селекторів тих елементів, які є дітьми іншого елемента в дереві документа. Селектори дітей утворюються шляхом з'єднання декількох селекторів символом ">". Наступне правило буде застосовуватися до всіх елементів **Р**, які є дітьми елемента **BODY** (іншими словами, воно застосовно до тих і тільки тим елементам **Р**, які перебувають усередині елемента **BODY**, і ніякого проміжного елемента між **BODY** й **Р** немає):

```

BODY > Р { text-indent: 3em }

```

У наступному прикладі сполучаються селектори нащадків і селектори дітей:

```

DIV OL>LI Р { line-height: 1.5 }

```

Суб'єктом цього правила буде елемент **Р**, що є нащадком елемента **LI**, що є дитиною елемента **OL**, що є нащадком елемента **DIV**. Зверніть увагу, що необов'язкові пробіли навколо символу ">" у цьому прикладі опущені.

Підтримка: Не підтримуються
Не підтримуються

2.2.7. Селектори сусідів

CSS підтримує використання *селекторів сусідів*, тобто селекторів тих елементів, які є братами в дереві документа й розташовані в документі безпосередньо один за одним. Селектори сусідів утворюються шляхом з'єднання декількох селекторів символом "+". Наступне правило зменшує відстань між елементами **H1** й **H2**, якщо **H2** безпосередньо треба в документі за **H1**:

```
H1 + H2 { margin-top: -5mm }
```

Приведемо аналогічний приклад, але в ньому елемент **H1** повинен ставитися мати атрибут `class="opener"`:

```
H1.opener + H2 { margin-top: -5mm }
```

Підтримка: Не підтримуються
Не підтримуються

2.2.8. Псевдоелементи й псевдокласы

Всі розглянуті дотепер селектори ґрунтувалися на положенні елемента в дереві документа. Однак, для деяких цілей відображення даних одного дерева недостатньо. Із цією метою в CSS були включені поняття *псевдоелементів* і *псевдокласов*, що забезпечують можливості відображення документа, не описувані в термінах дерева документа. Різниця між цими поняттями полягає в тому, що псевдокласы можуть уживатися в будь-якому місці селектора, а псевдоелементи тільки після суб'єкта селектора.

2.2.9. Псевдокласы

2.2.9.1. Псевдоклас `:first-child`

Псевдоклас **`:first-child`** відповідає елементу, що є *першою дитиною* даного елемента. У наступному прикладі суб'єктами селектора будуть ті елементи **P**, які є першою дитиною елемента **DIV**:

```
DIV > P:first-child { text-indent: 3em }
```

У результаті перший елемент **P** усередині **DIV** у наступному фрагменті буде виводитися з відступом на початку тексту

```
<P>Останній абзац перед приміткою</P>  
<DIV class="note">  
<P>Перший абзац усередині примітки</P>  
</DIV>
```

а такий же елемент **P** у цьому фрагменті відображається без відступу, тому що не є першою дитиною **DIV**:

```
<P>Останній абзац перед приміткою</P>  
<DIV class="note">  
<H2>Примітка</H2>  
<P>Перший абзац усередині примітки</P>  
</DIV>
```

Наступне правило вказує, що елемент **ЕМ**, що втримується в елементі **Р**, що є першою дитиною, повинен відображатися напівжирним шрифтом:

```
P:first-child ЕМ { font-weight: bold }
```

Наступні два селектори еквівалентні між собою:

```
* > A:first-child /* А є першою дитиною будь-якого елемента */  
A:first-child      /* Те ж саме */
```

Підтримка: Не підтримується

Не підтримується

2.2.9.2. Псевдокласы **:link** й **:visited**

Оглядачі часто по-різному відображають *уже посещения* й *ще не посещения гіперпосилання*. CSS забезпечує можливість завдання стилів їхнього відображення через псевдокласы **:link** (не посещения гіперпосилання) і **:visited** (посещения гіперпосилання). Який саме елемент задає анкери гіперпосилань, визначається мовою документа. У мові HTML анкери задаються елементом **A**, тому наступні декларації еквівалентні:

```
A:link { color: red }  
:link { color: red }
```

Якщо наступне гіперпосилання

```
<A class="external" href="http://www.out.com/">Зовнішнє посилання</A>
```

буде посещена користувачем, те правило

```
A.external:visited { color: blue }
```

викличе її відображенням блакитними кольорами.

Підтримка: Відповідність стандарту (5.0+)

Не підтримує **:visited** (4.0+)

2.2.9.3. Псевдокласы **:hover**, **:active** й **:focus**

Стиль відображення деяких елементів може динамічно змінюватися в результаті певних дій користувача. Для цього CSS містить три наступних псевдокласы:

- Псевдоклас **:hover** застосовується до відповідного елемента, коли користувач *навів курсор миші на цей елемент, але не активував його* щигликом миші.
- Псевдоклас **:active** застосовується до відповідного елемента, коли користувач *активував його* щигликом миші.
- Псевдоклас **:focus** застосовується до відповідного елемента, коли він *одержує фокус* (у результаті натискання певних клавіш).

Ці псевдокласы не є взаємно виключають; можливі ситуації, коли до елемента будуть одночасно застосовані правила відображення, задані декількома з них.

CSS не визначає, до яких саме елементів можуть застосовуватися зазначені псевдокласы. Однак, сучасні оглядачі підтримують їх тільки стосовно до елементів **A**, наприклад:

```
A:link { color: red } /* непосещенные посилання */  
A:visited { color: blue } /* посещения посилання */
```

```
A:hover { color: yellow } /* посилання під курсором миші */
A:active { color: lime } /* активні посилання */
```

Підтримка: Підтримує **:hover** й **:active** тільки стосовно до елемента **A** (5.0+)

Не підтримуються (4.0+)

2.2.9.4. Псевдоклас :lang

Псевдоклас **:lang(код мови)** указує на мову елемента. У мові HTML мова елемента задається атрибутом **lang** або відповідної [метаописателем](#); у мові XML — атрибутом **XML:LANG**. Наприклад, що впливають правила задають лапок для елементів, написаних на англійській і російській мовах відповідно:

```
HTML:lang(en) { quotes: '« ' ' »' }
HTML:lang(ru) { quotes: '»' '«' '\2039' '\203A' }
```

Підтримка: Не підтримується

Не підтримується

2.2.10. Псевдоэлементы

2.2.10.1. Псевдоэлемент :first-line

Псевдоэлемент **:first-line** застосуємо до будь-якого блокового елемента й задає стиль відображення *його першого рядка*. Наприклад, що впливає фрагмент HTML-документа

```
<STYLE>
P:first-line { text-transform: uppercase }
</STYLE>

<P>Це довгий абзац, що буде розбитий
оглядачем на кілька рядків.
При цьому перший рядок абзацу буде відображатися
прописними буквами, як це задано в
таблиці стилів</P>
```

буде відображатися так:

ЦЕ ДОВГИЙ АВЗАЦ, ЩО БУДЕ РОЗБИТИЙ ОГЛЯДАЧЕМ НА КІЛЬКА РЯДКІВ. ПРИ ЦЬОМУ перший рядок абзацу буде відображатися прописними буквами, як це задано в таблиці стилів.

До даного псевдоелемента застосовні тільки наступні властивості: [свойства шрифтов](#), [свойства цвета](#), [свойства фона](#), [word-spacing](#), [letter-spacing](#), [text-decoration](#), [vertical-align](#), [text-transform](#), [line-height](#), [text-shadow](#) і [clear](#).

Підтримка: Відповідність стандарту (5.5+)

Не підтримується (4.0+)

2.2.10.2. Псевдоэлемент :first-letter

Псевдоэлемент **:first-letter** застосуємо до будь-якого блокового елемента й задає стиль відображення *його першої букви*. Наприклад, що впливає фрагмент HTML-документа

```
<STYLE>
```

```
P:first-letter { font-size: 200%; font-weight: bold }  
</STYLE>
```

```
<P>Це абзац, перша буква якого  
буде виділена оглядачем</P>
```

буде відображатися так:

Це абзац, перша буква якого буде виділена оглядачем.

До даного псевдоелементу застосовні тільки наступні властивості: [свойства шрифтов](#), [свойства цвета](#), [свойства фона](#), [свойства границ](#), [свойства заполнителей](#), [свойства рамок](#), [text-decoration](#), [vertical-align](#) (тільки якщо [float](#) дорівнює *none*), [text-transform](#), [line-height](#), [float](#), [text-shadow](#) і [clear](#).

Підтримка: Відповідність стандарту (5.5+)

Не підтримується (4.0+)

2.2.10.3. Псевдоелементи **:before** й **:after**

Псевдоелементи **:before** й **:after** використовуються для *вставки автоматично генерируемого вмісту* відповідно *перед* або *після* зазначеного елемента. Докладно вони описані в [п. 2.11.2](#).

Підтримка: Не підтримуються

Не підтримуються

Глава 2.3. Значення властивостей, каскадність і спадкування

2.3.1. Обчислення значень властивостей

Після того, як оглядач проаналізував документ і побудував дерево документа, для кожного елемента дерева обчислюється значення кожного з його властивостей, застосовних до поточного [устроювання](#) [отображення](#). Остаточне значення властивості обчислюється в три етапи. Спочатку значення визначається зі специфікації (*специфіковане значення*), потім при необхідності перетворюється до абсолютного значення (*обчислене значення*), і, нарешті, перетворюється з урахуванням обмежень контексту (*фактичне значення*). Пояснимо кожний із цих етапів докладніше.

Специфіковане значення визначається за допомогою наступних механізмів, перерахованих у порядку переваги:

1. Якщо [каскад](#) повертає значення, то використовується воно.
2. У протилежному випадку, якщо властивість є [наслідуючим](#), те використовується відповідне значення батьківського елемента
3. У протилежному випадку, використовується початкове значення властивості (воно зазначено нижче у визначенні кожного із властивостей).

Специфіковані значення можуть бути як абсолютними (наприклад, *red* або *2mm*), так і відносними (наприклад, *auto* або *2em* або *10%*). Для абсолютних значень обчислене значення збігається зі специфікованим. З іншого боку, відносні значення повинні бути перетворені до абсолютного. Так, процентні величини перетворюються в числа шляхом множення на відповідне значення; розміри, задані в **em**, **ex** або **px**, множаться на розмір шрифту або пикселя; значення **auto** замінюється на величину, що обчислює по формулі, зазначеної у визначенні відповідної властивості й т.п.

Нарешті, оглядач перевіряє, чи припустиме обчислене значення в контексті даної властивості, і якщо ні, відповідно перетворює його. Наприклад, розмір у пикселях може бути тільки цілим, тому буде потрібно округлити отримане дійсне число до цілого. Результатом таких перетворень й є фактичне значення властивості, використовуване при відображенні елемента.

2.3.2. Спадкування

Деякі властивості успадковуються дітьми елемента в дереві документа. У визначенні кожної властивості вказується, є воно наслідуваним чи ні. Нехай, наприклад, елемент **H1** містить елемент **EM**:

```
<H1>Цей заголовок <EM>дуже важливий</EM>!</H1>
```

Якщо елементу **EM** не привласнений кольори, то він успадкує кольори свого батька, тобто елемента **H1**.

Із цієї причини для завдання стилю відображення елементів "за замовчуванням", досить задати стиль елемента **HTML** або **BODY**. Всі інші елементи є нащадками цих елементів, тому вони будуть успадковувати їхні властивості. При цьому важливо пам'ятати, що значення, задані у вигляді процентних величин, не успадковуються!

Багато властивостей мають як можливе значення *inherit*. Воно означає, що як значення властивості повинне використатися обчислене значення даної властивості батьківського елемента.

2.3.3. Директива @import

Директива **@import** дозволяє нам включати у свою таблицю стилів інші таблиці стилів. Вона повинна містити [URI](#) імпортованої таблиці стилів; наступні дві директиви еквівалентні й демонструють синтаксис даної директиви:

```
@import "mystyle.css";  
@import url(mystyle.css);
```

Директива **@import** може містити список назв [устроїв отображення](#), до яких повинна застосовуватися дана таблиця стилів, розділених комами, наприклад:

```
@import url("fineprint.css") print;  
@import url("bluish.css") projection, tv;
```

Якщо списку назв пристроїв ні, то передбачається, що він дорівнює *all*, тобто імпортована таблиця стилів застосовна до всіх пристроїв.

Директиви **@import** повинні розташовуватися в таблиці стилів перед першим правилом і не можуть перебувати усередині блоку; у протилежному випадку вони ігноруються оглядачем.

Підтримка: Підтримуються тільки пристрої *all*, *screen* й *print* (5.0+)

Не підтримується

2.3.4. Каскадность

2.3.4.1. Порядок каскадів

Таблиці стилів можуть мати три джерела походження: автор, користувач й оглядач.

- *Автор* задає таблицю стилів для свого документа, як було описано [вище](#).
- *Користувач* також може задати свою таблицю стилів для конкретного документа.
- *Оглядач* повинен застосувати до документа свою [таблицю стилей по умовчанию](#) перш, ніж застосовувати до нього всі інші таблиці стилів.

Каскадность мови CSS полягає в тому, що кожному правилу приписана певна вага; якщо до конкретного елемента застосовні кілька правил, то використовується те, що має найбільша вага. У результаті відбувається кумулятивне нагромадження властивостей елементів відповідно до правил спадкування, і, тим самим, утворюється каскад властивостей, що поширюється від предків до нащадків.

За замовчуванням, вага правил таблиці автора більше, ніж вага правил таблиці користувача (за винятком правил з атрибутом *!important*, для яких це співвідношення є зворотним). Вага правил таблиць автора й користувача більше, ніж вага правил таблиці оглядача. Загальний порядок визначення правила й властивості, які будуть застосовані до елемента, такий:

1. Вибираються всі декларації, які відповідають даному пристрою відображення; з них вибираються всі правила, чиї селектори відповідають даному елементу.
2. Декларації сортуються по вазі їхнього джерела походження, як описано вище.
3. Виробляється вторинне сортування по [специфічності селектора](#): більше специфічні селектори сильніше, ніж більше загальні.
4. І, нарешті, останнє сортування: якщо два правила мають однакову вагу й специфічність, то застосовується останнє з них. При цьому правила імпортованих таблиць розташовуються до всіх правил таблиці, що імпортує.

2.3.4.2. Атрибут *!important*

Для того, щоб правила користувацької таблиці стилів могли перекривати авторську, CSS містить атрибут *!important*. Правило користувацької таблиці стилів, що має такий атрибут, має більша вага, чим відповідне правило авторської таблиці стилів. Розглянемо наступний приклад:

```
/* З таблиці стилів користувача */
P { text-indent: 1em !important }
P { font-style: italic !important }
P { font-size: 18pt }

/* З таблиці стилів автора */
P { text-indent: 1.5em !important }
P { font: 12pt sans-serif !important }
P { font-size: 24pt }
```

Тут перше правило таблиці стилів користувача містить атрибут *!important*, тому воно важче, чим перше правило таблиці стилів автора. Друге правило користувача також більш важче, по тій же самій причині. Однак, третє правило користувача менш важче, чим друге правило автора. Точно також, третє правило автора менш важче, чим його друге правило.

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

2.3.4.3. Специфічність селектора

Специфічність селектора обчислюється в такий спосіб:

- a. підрахувати кількість атрибутів **id** у селекторі;
- b. підрахувати кількість атрибутів **class** у селекторі;
- c. підрахувати кількість імен елементів у селекторі (всі псевдоелементи ігноруються).

Тепер запишемо ці три числа підряд, щоб одержати число із трьох цифр (нам, можливо, прийде привести числа a, b й c до системи числення з більшою підставою, щоб кожне з них виражалось однією цифрою). Результатом і буде специфічність селектора (чим вона вище, тим селектор специфічней). Приведемо список прикладів селекторів, відсортованих по їхній специфічності:

```
#id1      {...} /* a=1 b=0 c=0 -i> специфичность = 100 */
UL UL LI.red {...} /* a=0 b=1 c=3 -i> специфичность = 013 */
LI.red    {...} /* a=0 b=1 c=1 -i> специфичность = 011 */
LI        {...} /* a=0 b=0 c=1 -i> специфичность = 001 */
```

Якщо перевести цей опис із формальної мови на звичайну, то можна сказати, що клас елементів є більше специфічним, чим просто елемент, а ідентифікатор елемента більше специфічний, чим клас.

2.4.1. Загальний формат відображення

У цій главі ми описуємо загальний формат візуального відображення елементів документа, використовуваний CSS. Для кожного елемента генерується *объемлющий* його *прямокутник*, будова якого показане на малюнку.

Як видно з малюнка, объемлющий прямокутник складається, крім *умісту* (content), із *заповнювача* (padding), *рамки* (border) і *границі* (margin). У свою чергу, кожний із цих шарів розпадається на чотири частини: ліву (left), праву (right), верхню (top) і нижню (bottom). На малюнку ці частини позначені відповідними скороченнями: "LM" - ліва границя (left margin), "TB" - верхня рамка (top border) і т.п.

Периметр кожної із чотирьох частин прямокутника називається її *краєм*, так що кожен объемлющий прямокутник містить чотири краї:

Край умісту або внутрішній край

Обмежує відображуваний уміст елемента.

Край заповнювача

Обмежує заповнювач прямокутника. Якщо ширина заповнювача дорівнює 0, то край заповнювача збігається із внутрішнім краєм елемента. Прямокутник, обмежений краєм заповнювача, інакше називається *блоком*, що вміщає, елемента.

Край рамки

Обмежує рамку прямокутника. Якщо ширина рамки дорівнює 0, то край рамки збігається із краєм заповнювача елемента.

Край границі або зовнішній край

Обмежує границю прямокутника. Якщо ширина границі дорівнює 0, то край границі збігається із краєм рамки елемента.

Кожен край, у свою чергу, розпадається на чотири частини: ліву, праву, верхню й нижню.

Розміри області вмісту в прямокутнику, або *ширина вмісту* й *висота вмісту*, залежать від декількох факторів: що саме є вмістом елемента (текст, таблиця або інші елементи), чи задані властивості елемента **width** і **height** і т.д. Докладно ці питання обговорюються в [гл. 2.6](#). *Ширина объемлющего прямокутника* дорівнює сумі ширин лівої й правої границі, лівої й правої рамки, лівого й правого заповнювача й ширини вмісту. Його *висота* дорівнює сумі висот верхньої й нижньої границі, верхньої й нижньої рамки, верхнього й нижнього заповнювача й висоти вмісту.

Тло різних областей объемлющего прямокутника визначається в такий спосіб:

- *Уміст*: властивість **background** елемента, що генерує.
- *Заповнювач*: властивість **background** елемента, що генерує.
- *Рамка*: властивості рамки елемента, що генерує.
- *Границя*: границі завжди прозорі.

Проілюструємо уведені поняття прикладом, у якому використані всі чотири області объемлющего прямокутника.



Цей прямокутник сгенерований наступним елементом:

```
<P style="background-color: yellow; color: blue; margin: 20px 25%; padding: 10px;
```



```
border: thick solid green">Текст абзацу</P>
```

2.4.2. Властивості границі

Властивості *границі* задають розміри границі об'ємлюючого прямокутника. Всі вони мають тип <розміру-границі>, що визначається в такий спосіб:

<розміру-границі> = [<размер>](#) | [<процент>](#) | *auto*

[<размер>](#)

Задає фіксований розмір.

[<процент>](#)

Обчислюється щодо *ширини* блоку, що вміщає, навіть для вертикальних границь.

auto

Див. [раздел 2.6.2](#).

Допускаються негативні розміри границь, але оглядачі можуть накладати свої обмеження.

2.4.2.1. Розмір верхньої границі: властивість **margin-top**

Синтаксис: `margin-top: <размер-границы> | inherit`

Начально: 0

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **margin-top** задає розмір *верхньої границі* об'ємлюючого прямокутника. Наприклад, що впливає правило забирає верхню границю документа:

```
BODY { margin-top: 0 }
```

2.4.2.2. Розмір нижньої границі: властивість **margin-bottom**

Синтаксис: `margin-bottom: <размер-границы> | inherit`

Начально: 0

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **margin-bottom** задає розмір *нижньої границі* об'ємлюючого прямокутника. Наприклад, що впливає правило змінює нижню границю документа:

```
BODY { margin-bottom: 3em }
```

2.4.2.3. Розмір лівої границі: властивість **margin-left**

Синтаксис: `margin-left: <размер-границы> | inherit`

Начально: 0

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **margin-left** задає розмір *лівої границі об'ємлюючого прямокутника*. Наприклад, що впливає правило задає відступ для елементів **BLOCKQUOTE**:

```
BLOCKQUOTE{margin-left: 25% }
```

2.4.2.4. Розмір правої границі: властивість **margin-right**

Синтаксис: `margin-right: <размер-границы> | inherit`

Начально: 0

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **margin-right** задає розмір *правої границі об'ємлюючого прямокутника*. Наприклад, що впливає правило задає ширину вузьких абзаців:

```
P.narrow { margin-right: 50% }
```

2.4.2.5. Розміри границі: властивість **margin**

Синтаксис: `margin: <размер-границы> {1,4} | inherit`

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **margin** є скороченням для властивостей **margin-top**, **margin-bottom**, **margin-left** і **margin-right**. Воно задає розмір *всіх границь об'ємлюючого прямокутника* одночасно.

Його значенням може бути від одного до чотирьох розмірів. Якщо задані всі розміри, то вони застосовуються до верхньої, правої, нижньої й лівої границі відповідно. Якщо задано тільки один розмір, то він застосовується до всіх границь. Якщо задані два або три значення, то відсутній розмір приймається рівним розміру протилежної границі. Приклади:

```
BODY { margin: 1em } /* всі границі рівні 1em */
BODY { margin: 1em 2em } /* top & bottom = 1em, right & left = 2em */
BODY { margin: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
BODY { margin: 1em 2em 3em 4em } /* top=1em, right=2em, bottom=3em, left=4em */
```

2.4.2.6. Злиття границь

У деяких ситуаціях CSS допускає *злиття границь* сусідніх об'ємлюючих прямокутників. Злиття границь означає, що сусідні границі поєднуються в одну границю нової ширини.

Горизонтальні границі не зливаються ніколи. Вертикальні границі можуть зливатися відповідно до наступних правил:

- Дві й більше сусідніх вертикальних границі блокових елементів зливаються й ширина нової границі дорівнює ширині найбільшої із границь, що зливаються.
- Границі між плаваючим елементом і будь-яким іншим елементом не зливаються.
- Границі між абсолютно й відносно позиціонованими елементами не зливаються.

2.4.3. Властивості заповнювача

Властивості заповнювача задають розміри заповнювача об'ємлющего прямокутника. Всі вони мають тип <розміру-заповнювача>, що визначається в такий спосіб:

<розміру-заповнювача> = [<размер>](#) | [<процент>](#)
[<размер>](#)
 Задає фіксований розмір.
[<процент>](#)
 Обчислюється щодо ширини блоку, що вміщає, навіть для вертикальних границь.

В відмінність від розмірів границь, розміри заповнювачів не можуть бути негативними.

2.4.3.1. Розмір верхнього заповнювача: властивість padding-top

Синтаксис: padding-top: [<размер-заполнителя>](#) | [inherit](#)
Начально: 0
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: щодо ширини блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **padding-top** задає розмір *верхнього заповнювача об'ємлющего прямокутника*. Приклад:

```
BLOCKQUOTE { padding-top: 0.5em }
```

2.4.3.2. Розмір нижнього заповнювача: властивість padding-bottom

Синтаксис: padding-bottom: [<размер-заполнителя>](#) | [inherit](#)
Начально: 0
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: щодо ширини блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **padding-bottom** задає розмір *нижнього заповнювача об'ємлющего прямокутника*. Приклад:

```
BLOCKQUOTE { padding-bottom: 0.5em }
```

2.4.3.3. Розмір лівого заповнювача: властивість padding-left

Синтаксис: padding-left: [<размер-заполнителя>](#) | [inherit](#)
Начально: 0
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: щодо ширини блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **padding-left** задає розмір *лівого заповнювача об'ємлющего прямокутника*. Приклад:

```
BLOCKQUOTE { padding-left: 20px }
```

2.4.3.4. Розмір правого заповнювача: властивість **padding-right**

Синтаксис: `padding-right: <размер-заполнителя> | inherit`

Начально: 0

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **padding-right** задає розмір *правого заповнювача об'ємлющего прямокутника*. Приклад:

```
BLOCKQUOTE { padding-right: 20px }
```

2.4.3.5. Розміри заповнювача: властивість **padding**

Синтаксис: `padding: <размер-заполнителя> {1,4} | inherit`

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **padding** є скороченням для властивостей **padding-top**, **padding-bottom**, **padding-left** і **padding-right**. Воно задає розмір *всіх заповнювачів об'ємлющего прямокутника* одночасно.

Його значенням може бути від одного до чотирьох розмірів. Якщо задані всі розміри, то вони застосовуються до верхнього, правого, нижнього й лівого заповнювача відповідно. Якщо задано тільки один розмір, то він застосовується до всіх заповнювачів. Якщо задані два або три значення, то відсутній розмір приймається рівним розміру протилежного заповнювача. Приклади:

```
BODY { padding: 1em } /* всі заповнювачі рівні 1em */
BODY { padding: 1em 2em } /* top & bottom = 1em, right & left = 2em */
BODY { padding: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
BODY { padding: 1em 2em 3em 4em } /* top=1em, right=2em, bottom=3em, left=4em */
```

2.4.4. Властивості рамки

Властивості рамки задають розмір, кольори й стиль рамки об'ємлющего прямокутника. Відповідно їх можна розбити на наступні групи:

- Розмір рамки: **border-top-width**, **border-bottom-width**, **border-left-width**, **border-right-width** і **border-width**.
- Кольори рамки: **border-top-color**, **border-bottom-color**, **border-left-color**, **border-right-color** і **border-color**.

- Стиль рамки: **border-top-style**, **border-bottom-style**, **border-left-style**, **border-right-style** і **border-style**.
- Скорочення: **border-top**, **border-bottom**, **border-left**, **border-right** і **border**.

2.4.5. Розміри рамки

Розміри рамки задають розміри рамки об'ємлюючого прямокутника. Всі вони мають тип <розміру-рамки>, що визначається в такий спосіб:

<розміру-рамки> = *thin* | *medium* | *thick* | <размер>
thin
 Тонка рамка (точний розмір залежить від оглядача).
medium
 Середня рамка (точний розмір залежить від оглядача).
thick
 Толста рамка (точний розмір залежить від оглядача).
<размер>
 Задає фіксований розмір, що не може бути негативним.

2.4.5.1. Розмір верхньої рамки: властивість **border-top-width**

Синтаксис: **border-top-width:** <размер-рамки> | inherit
Начально: *medium*
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: визуальные

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-top-width** задає розмір *верхньої рамки об'ємлюючого прямокутника*. Приклад:

```
H1 { border-top-width: thin }
```

2.4.5.2. Розмір нижньої рамки: властивість **border-bottom-width**

Синтаксис: **border-bottom-width:** <размер-рамки> | inherit
Начально: *medium*
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: визуальные

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-bottom-width** задає розмір *нижньої рамки об'ємлюючого прямокутника*. Приклад:

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

2.4.5.3. Розмір лівої рамки: властивість **border-left-width**

Синтаксис: **border-left-width:** <размер-рамки> | inherit
Начально: *medium*
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: визуальные

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-left-width** задає розмір *лівої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-left-width: 10px }
```

2.4.5.4. Розмір правої рамки: властивість **border-right-width**

Синтаксис: `border-right-width: <размер-рамки> | inherit`

Начально: `medium`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Відповідає стандарту (4.0+)

Властивість **border-right-width** задає розмір *правої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-right-width: 0 }
```

2.4.5.5. Розміри рамки: властивість **border-width**

Синтаксис: `border-width: <размер-рамки> {1,4} | inherit`

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-width** є скороченням для властивостей **border-top-width**, **border-bottom-width**, **border-left-width** і **border-right-width**. Воно задає розмір *всіх рамок* *объемлющего прямокутника* одночасно.

Його значенням може бути від одного до чотирьох розмірів. Якщо задані всі розміри, то вони застосовуються до верхньої, правої, нижньої й лівої рамки відповідно. Якщо задано тільки один розмір, то він застосовується до всіх рамок. Якщо задані два або три значення, то відсутній розмір приймається рівним розміру протилежної рамки. Приклади:

```
H1 { border-width: thin } /* всі рамки мають розмір thin */
H1 { border-width: thin thick } /* top & bottom = thin, right &
left = thick */
H1 { border-width: thin thick medium } /* thin, thick, medium, thick */
```

2.4.6. Кольори рамки

Кольори рамки задають кольори рамки *объемлющего прямокутника*. Всі вони мають тип [<цвет>](#). Еслт кольори рамки не заданий, то він повинен збігатися з кольорами елемента, що генерує, наприклад наступна декларація

```
P {
  color: black;
  border: thin solid;
}
```

буде генерувати чорну тонку суцільну рамку навколо абзаців.

2.4.6.1. Кольори верхньої рамки: властивість **border-top-color**

Синтаксис: `border-top-color: <цвет> | inherit`

Начально: кольори елемента, що генерує
Применимо: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-top-color** задає кольори *верхньої рамки об'ємлюючого прямокутника*. Приклад:

```
H1 { border-top-color: red }
```

2.4.6.2. Кольори нижньої рамки: властивість **border-bottom-color**

Синтаксис: `border-bottom-color: <цвет> | inherit`
Начально: кольори елемента, що генерує
Применимо: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-bottom-color** задає кольори *нижньої рамки об'ємлюючого прямокутника*. Приклад:

```
H1 { border-bottom-color: red }
```

2.4.6.3. Кольори лівої рамки: властивість **border-left-color**

Синтаксис: `border-left-color: <цвет> | inherit`
Начально: кольори елемента, що генерує
Применимо: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-left-color** задає кольори *лівої рамки об'ємлюючого прямокутника*. Приклад:

```
H1 { border-left-color: blue }
```

2.4.6.4. Кольори правої рамки: властивість **border-right-color**

Синтаксис: `border-right-color: <цвет> | inherit`
Начально: кольори елемента, що генерує
Применимо: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-right-color** задає кольори *правої рамки об'ємлюючого прямокутника*. Приклад:

```
H1 { border-right-color: blue }
```

2.4.6.5. Кольори рамки: властивість **border-color**

Синтаксис: `border-color: <цвет> {1,4} | transparent | inherit`

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-color** є скороченням для властивостей **border-top-color**, **border-bottom-color**, **border-left-color** і **border-right-color**. Воно задає кольори *всіх рамок об'ємлющего прямокутника* одночасно. Значення *transparent* означає, що рамка повинна бути прозорою (хоча вона може мати при цьому ненульову ширину).

Значенням цієї властивості може бути від одного до чотирьох квітів. Якщо задані всі кольори, то вони застосовуються до верхньої, правої, нижньої й лівої рамки відповідно. Якщо задано тільки один колір, то він застосовується до всіх рамок. Якщо задані два або три значення, то відсутні кольори приймається рівним кольорам протилежної рамки. Приклади:

```
H1 { border-color: red }           /* всі рамки мають кольори red */
H1 { border-color: red blue }      /* top & bottom = red, right & left =
blue */
H1 { border-color: red green blue } /* red, green, blue, green */
```

2.4.7. Стили рамки

Стили рамки задають стилі зображення рамки об'ємлющего прямокутника. Всі вони мають тип <стилю-рамки>, що визначається в такий спосіб:

<стилю-рамки> = `none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset`

none

Ні рамки (тягне присвоювання **border-width** значення 0).

hidden

Схована рамка. Те ж, що *none*, крім таблиць (див. [п. 2.10.4.1](#)).

dotted

Пунктирна рамка (виводиться крапками).

dashed

Штрихова рамка (виводиться короткими відрізками).

solid

Суцільна рамка (виводиться суцільною лінією).

double

Подвійна рамка (виводиться подвійною суцільною лінією).

groove

Рамка зображується у вигляді тривимірної виїмки.

ridge

Протилежність *groove*. Рамка зображується у вигляді тривимірного виступу.

inset

Рамка зображується у вигляді тривимірного урізання.

outset

Протилежність *inset*. Рамка зображується у вигляді тривимірного вирізки.

Підтримка: Стиль *hidden* відображається як *none* (4.0+)

Стиль *hidden* відображається як *outset*, *dotted* й *dashed* як *solid* (4.0+)

2.4.7.1. Стиль верхньої рамки: властивість `border-top-style`

Синтаксис: `border-top-style: <стиль-рамки> | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість `border-top-style` задає стиль *верхньої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-top-style: solid }
```

2.4.7.2. Стиль нижньої рамки: властивість `border-bottom-style`

Синтаксис: `border-bottom-style: <стиль-рамки> | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість `border-bottom-style` задає стиль *нижньої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-bottom-style: none }
```

2.4.7.3. Стиль лівої рамки: властивість `border-left-style`

Синтаксис: `border-left-style: <стиль-рамки> | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість `border-left-style` задає стиль *лівої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-left-style: dashed }
```

2.4.7.4. Стиль правої рамки: властивість `border-right-style`

Синтаксис: `border-right-style: <стиль-рамки> | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість `border-right-style` задає стиль *правої рамки* *объемлющего прямокутника*. Приклад:

```
H1 { border-right-style: double }
```

2.4.7.5. Стилi рамки: властивiсть border-style

Синтаксис: border-style: [<стиль-рамки>](#) {1,4} | [inherit](#)

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **border-style** є скороченням для властивостей **border-top-style**, **border-bottom-style**, **border-left-style** і **border-right-style**. Воно задає стиль *всіх рамок об'ємлюючого прямокутника* одночасно.

Значенням цієї властивості може бути від одного до чотирьох стилів. Якщо задані всі стилі, то вони застосовуються до верхньої, правої, нижньої й лівої рамки відповідно. Якщо задано тільки один стиль, то він застосовується до всіх рамок. Якщо задані два або три значення, то відсутній стиль приймається рівним стилю протилежної рамки. Приклади:

```
H1 { border-style: solid } /* всі рамки мають стиль solid */
H1 { border-style: solid none } /* top & bottom = solid, right & left = none */
H1 { border-style: solid dashed dotted } /* solid, dashed, dotted, dashed */
```

2.4.8. Скорочені властивості рамки

2.4.8.1. Властивості верхньої рамки: властивість border-top

Синтаксис: border-top: [[<размер-рамки>](#) | [<стиль-рамки>](#) | [<цвет>](#)] | [inherit](#)

Начально: див. індивідуальні властивості

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-top** задає розмір, стиль і кольори *верхньої рамки об'ємлюючого прямокутника*. Якщо якийсь із цих властивостей не задано, то приймається його початкове значення. Приклади:

```
H1 { border-top: thick solid red }
H1 { border-top: solid red } /* те ж, що medium solid red */
```

2.4.8.2. Властивості нижньої рамки: властивість border-bottom

Синтаксис: border-bottom: [[<размер-рамки>](#) | [<стиль-рамки>](#) | [<цвет>](#)] | [inherit](#)

Начально: див. індивідуальні властивості

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-bottom** задає розмір, стиль і кольори *нижньої рамки об'ємлюючого прямокутника*. Якщо якесь із цих властивостей не задано, то приймається його початкове значення. Приклади:

```
H1 { border-bottom: thick solid red }
H1 { border-bottom: solid red }      /* те ж, що medium solid red */
```

2.4.8.3. Властивості лівої рамки: властивість border-left

Синтаксис: border-left: [[<розмер-рамки>](#)] | [[<стиль-рамки>](#)] | [[<цвет>](#)] | [inherit](#)
Начально: див. індивідуальні властивості
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-left** задає розмір, стиль і кольори *лівої рамки об'ємлюючого прямокутника*. Якщо якесь із цих властивостей не задано, то приймається його початкове значення. Приклади:

```
H1 { border-left: thick solid red }
H1 { border-left: solid red }      /* те ж, що medium solid red */
```

2.4.8.4. Властивості правої рамки: властивість border-right

Синтаксис: border-right: [[<розмер-рамки>](#)] | [[<стиль-рамки>](#)] | [[<цвет>](#)] | [inherit](#)
Начально: див. індивідуальні властивості
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border-right** задає розмір, стиль і кольори *правої рамки об'ємлюючого прямокутника*. Якщо якесь із цих властивостей не задано, то приймається його початкове значення. Приклади:

```
H1 { border-right: thick solid red }
H1 { border-right: solid red }      /* те ж, що medium solid red */
```

2.4.8.5. Властивості всіх рамок: властивість border

Синтаксис: border: [[<розмер-рамки>](#)] | [[<стиль-рамки>](#)] | [[<цвет>](#)] | [inherit](#)
Начально: див. індивідуальні властивості
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **border** задає розмір, стиль і кольори *всіх рамок об'ємлюючого прямокутника* одночасно. Якщо якесь із цих властивостей не задано, то приймається його початкове значення. Приклади:

```
H1 { border: thick solid red }
H1 { border: solid red }          /* те ж, що medium solid red */
```

2.5.1. Загальні положення

У попередній главі ми розглянули генерацію об'ємлюючих прямокутників для окремих елементів. Тут ми вивчимо, як оглядач переглядає дерево елементів і відображає їх на візуальному носії, тобто те, як об'ємлюючі прямокутники розташовуються відносно один одного.

У процесі відображення документа для кожного елемента оглядач генерує нуль або більше об'ємлюючих прямокутників. Зовнішній вигляд і розташування цих прямокутників визначається наступними параметрами:

- *розмірами* прямокутника, які задаються властивостями границі, заповнювача й рамки;
- *типом* прямокутника, що задається властивістю **display**;
- *схемою позиціонування* елемента, що задається властивістю [position](#);
- взаєминами елементів у дереві документа;
- зовнішньою інформацією (наприклад, розміром вікна відображення, власними розмірами графічних образів і т.п.).

Далі ми розглянемо, як саме відбувається генерація об'ємлюючих прямокутників залежно від перерахованих параметрів.

2.5.2. Типи об'ємлюючих прямокутників: властивість display

Синтаксис: `display: inline | block | list-item | run-in | compact | marker | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | none | inherit`

Начально: `inline`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрій: [все](#)

Підтримка: Підтримує тільки `inline`, `block`, `list-item`, `none` (4.0+), `table-header-group` й `table-footer-group` (5.0+)

Підтримує тільки `inline`, `block`, `list-item` й `none` (4.0+)

Властивість **display** задає *тип об'ємлюючого прямокутника* для даного елемента. CSS підтримує наступні типи прямокутників:

<i>block</i>	<i>Блоковий елемент.</i> Відповідає блочным элементам HTML , т. е. відображається як окремий абзац. При його відображенні генерується <i>головний прямокутник блоку</i> , у якому розташовуються <i>об'ємлюючі</i> прямокутники нащадків даного елемента.
<i>inline</i>	<i>Текстовий елемент.</i> Відповідає текстовым элементам HTML , текстовим <i>ображається</i> як текстові рядки усередині поточного абзацу, точніше усередині головного прямокутника відповідного <i>блоку</i> .
<i>list-item</i>	<i>Елемент списку.</i> Відображається як блоковий елемент із додаванням до <i>нього</i> маркера елемента списку (докладніше див. п. 2.11.5).
<i>marker</i>	<i>Маркер.</i> Подобиці див. у п. 2.11.4 .
<i>run-in</i>	<i>Елемент, що приєднує.</i> Якщо наступний за даним елемент є <i>блоковим</i> , то даний елемент форматується як його перший текстовий елемент. У протилежному випадку

	відображається як звичайний блоковий елемент.
<i>compact</i>	<i>Компактний елемент.</i> Якщо наступний за даним елемент є блоковим , то даний елемент форматується як однорядковий текстовий елемент, і якщо він міститься на лівій або правій границі наступного блоку (границя задається властивістю direction блоку , що вміщає), то на ній він і відображається. У протилежному випадку відображається як звичайний блоковий елемент.
<i>none</i>	Елемент і всі його нащадки <i>ігноруються при відображенні</i> .
<i>table</i>	Блокова таблиця. Подробиці див. у п. 2.10.1.
<i>inline-table</i>	Текстова таблиця. Подробиці див. у п. 2.10.1.
<i>table-row-group</i>	Група рядків таблиці. Подробиці див. у п. 2.10.1.
<i>table-header-group</i>	Група надзаголовків таблиці. Подробиці див. у п. 2.10.1.
<i>table-footer-group</i>	Група підзаголовків таблиці. Подробиці див. у п. 2.10.1.
<i>table-row</i>	Рядок таблиці. Подробиці див. у п. 2.10.1.
<i>table-column-group</i>	Група стовпців таблиці. Подробиці див. у п. 2.10.1.
<i>table-column</i>	Стовпець таблиці. Подробиці див. у п. 2.10.1.
<i>table-cell</i>	Осередок таблиці. Подробиці див. у п. 2.10.1.
<i>table-caption</i>	Заголовок таблиці. Подробиці див. у п. 2.10.1.

Хоча початкове значення цієї властивості *inline*, таблиця стилів оглядача, прийнята за замовчуванням, може підміняти це значення для окремих елементів. Подробиці [див. у Рекомендованій таблиці стилів для HTML](#). Приклади завдання типів відображення для деяких елементів:

```
P { display: block } /* Блоковий елемент */
EM { display: inline } /* Текстовий елемент */
LI { display: list-item } /* Елемент списку */
IMG { display: none } /* Не відображати графічні образи */
```

2.5.3. Схема позиціонування: властивість position

Синтаксис: position: static | relative | absolute | fixed | [inherit](#)
Начально: static
Застосовно: до всіх елементів, крім генерируемого змісту
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальні](#)

Підтримка: Підтримує тільки static, relative й absolute (4.0+)

Підтримує тільки relative й absolute; реалізовано з помилками (4.0+)

Властивість **position** задає *схему позиціонування* даного елемента. CSS підтримує наступні схеми позиціонування:

<i>static</i>	<i>Статичний елемент.</i> Не має спеціального позиціонування, відображається у звичайному потоці елементів.
<i>relative</i>	Відносно позиціонуемый елемент. Для нього спочатку генерується объемлющий

	прямокутник у звичайному потоці елементів, а потім зміщується на величини, задані властивостями left і top.
<i>absolute</i>	Абсолютно позиціонується елемент. Позиція елемента обчислюється щодо позиції його найближчого позиціонованого предка (або щодо елемента BODY , якщо всі предки не позиціоновані) на підставі властивостей left і top . Крім позиції елемента можна задати і його розмір властивостями right і bottom .
<i>fixed</i>	Фіксований елемент. Цей елемент є абсолютно позиціонованим , але додатково зафіксований щодо якоїсь відправної крапки . Для непрерывных устройств отображения його позиція зафіксована щодо вікна оглядача, тобто він залишається нерухомим при прокручуванні вікна. Для страничных устройств його позиція фіксується щодо сторінки.

Абсолютно позиціоновані елементи випадають зі звичайного потоку відображення елементів документа. Їхня позиція не залежить від навколишніх об'єктів, а визначається щодо найближчого позиціонованого предка. Якщо задана позиція вже зайнята іншим елементом, то обидва елементи будуть відображатися в одному місці, перекриваючи один одного. Те, який з елементів перебуває вище, а який нижче, задається властивістю **z-index**. Абсолютно позиціоновані елементи не мають границь, але мають заповнювачі й рамки. Приклад абсолютно позиціонованого графічного образу усередині відносно позиціонованого елемента **DIV**:

```
<DIV style="position: relative; left: 0; top: 0; height: 50px">
  <P>Частина тексту буде схована малюнком, тому що він позиціонован
поверх тексту</P>
  <IMG src="images\leaf.gif" style="position: absolute; left: 300px; top:
0">
</DIV>
```

Цей приклад буде відображатися так (Netscape Navigator 4.x відображає цей приклад невірно!):

Частина тексту буде схована малюнком, тому що він позиціонован поверх тексту.

Відносно позиціоновані елементи відображаються у звичайному потоці документа, але із заданим зсувом. Наступний приклад показує, як можна створити відносним позиціонуванням текст у верхньому індексі:

```
<P>Приклад виводу тексту у верхньому індексі:
  <SPAN style="position: relative; top: -3px">xyz</SPAN>.</P>
```

Цей приклад буде відображатися так:

Приклад виводу тексту у верхньому індексі: xyz.

Елемент, якому треба за відносно позиціонованим елементом, відображається так, ніби цей елемент не був зміщений. Елемент, якому треба за абсолютно позиціонованим елементом, відображається в тій позиції, що займали би цей елемент, не будучи абсолютно позиціонованим.

2.5.4. Завдання позиції елемента

Якщо елемент не є статичним, то його позиція повинна бути задана явно. Для цього використовуються чотири властивості: **left, top, right і bottom**, які задають відносний зсув елемента. Якщо вони не задані, то положення й розмір об'ємлющего елемента прямокутника обчислюється автоматично. Ці властивості мають тип `<зсув>`, що визначається в такий спосіб:

```
<зсув> = <размер> | <процент> | auto
<размер>
  Задає фіксований зсув.
<процент>
```

Обчислюється щодо *висоти* або *ширини* блоку, що вміщає.
auto
Положення елемента за замовчуванням у звичайному потоці елементів.
Див. [2.6.2](#) і [2.6.5](#).

2.6.5.4.1. Верхня позиція: властивість **top**

Синтаксис: `top: <смещение> | inherit`
Начально: `auto`
Застосовно: до всіх позиціонованим елементам
Наслідувано: немає
Відсотки: Обчислюється щодо *висоти* блоку, що вміщає.
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **top** задає зсув позиціонованого елемента *щодо верхнього краю блоку, що вміщає*.
Приклад:

```
<IMG src="sample.gif" style="position: absolute; left: 300px; top: 0">
```

2.5.4.2. Ліва позиція: властивість **left**

Синтаксис: `left: <смещение> | inherit`
Начально: `auto`
Застосовно: до всіх позиціонованим елементам
Наслідувано: немає
Відсотки: Обчислюється щодо *ширини* блоку, що вміщає.
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **left** задає зсув позиціонованого елемента *щодо лівого краю блоку, що вміщає*. Приклад:

```
<IMG src="sample.gif" style="position: absolute; left: 300px; top: 0">
```

2.5.4.3. Нижня позиція: властивість **bottom**

Синтаксис: `bottom: <смещение> | inherit`
Начально: `auto`
Застосовно: до всіх позиціонованим елементам
Наслідувано: немає
Відсотки: Обчислюється щодо *висоти* блоку, що вміщає.
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **bottom** задає зсув позиціонованого елемента *щодо нижнього краю блоку, що вміщає*.
Приклад:

```
<IMG src="sample.gif" style="position: absolute; right: 20px; bottom: 50px">
```

2.5.4.4. Права позиція: властивість **right**

Синтаксис: `right: <смещение> | inherit`
Начально: `auto`
Застосовно: до всіх позиціонованим елементам
Наслідувано: немає
Відсотки: Обчислюється щодо *ширини* блоку, що вміщає.
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **right** задає зсув позиціонованого елемента щодо правого краю блоку, що вміщає.
Приклад:

```
<IMG src="sample.gif" style="position: absolute; right: 20px; bottom: 50px">
```

2.5.5. Порядок відображення елементів: властивість z-index

Синтаксис: `z-index: auto | <целое> | inherit`
Начально: `auto`
Застосовно: до позиціонованим елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Підтримується тільки для шарів (layers) (4.0+)

Як ми вже відзначали вище, позиціонування елементів часто приводить до того, що объемлющие прямокутники декількох елементів перекриваються, тобто при відображенні ці елементи накладаються один на одного. При цьому елементи утворюють *стек*, розташований по осі координат Z. Чим більше індекс елементу в стеці, тим вище він розташований, тобто краще бачимо користувачем. Елементи з однаковим індексом розташовуються за звичайним правилом наповнення стека: чим раніше елемент розташований у тексті документа, тим глибше він перебуває.

Властивість **z-index** дозволяє явно задати *індекс елемента* в цьому *стеці* й тим самим управляти порядком накладення елементів один на одного при відображенні. Воно може приймати наступні значення:

<целое>	Явно задає індекс елемента в стеці.
<code>auto</code>	Індекс елемента приймається рівним індексу його батька.

У наступному прикладі текст абзацу розташований у документі раніше графічного образу, але шляхом завдання властивості **z-index** ми поміщаємо його поверх малюнка.

```
<DIV style="position: relative; top: 0; left: 0; width: 100%; height: 60px">
  <P style="position: absolute; top: 10px; left: 100px; z-index: 2;
    background: purple; color: white">Текст</P>
  <IMG src="images/leaf.gif" style="position: absolute; top: 10px; left: 100px;
    z-index: 1">
</DIV>
```

Наведений фрагмент буде відображатися так (Netscape Navigator 4.x відображає цей приклад невірно!):

Текст

2.5.6. Плаваючі елементи: властивість float

Синтаксис: float: left | right | none | [inherit](#)

Начально: none

Застосовно: до всіх елементів, крім позиціонованих і генерируемого змісту

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту; реалізовано з помилками (4.0+)

Властивість **float** зі значенням *left* або *right* указує, що елемент є *плаваючим* і задає його вирівнювання вліво або вправо. За замовчуванням ця властивість має значення *none*, що означає, що елемент не є плаваючим.

Плаваючий елемент завжди вважається блоковим, тобто його властивість **display** ігнорується. Плаваючий елемент зміщується вліво або вправо доти, поки не зустріне рамку, заповнювач або границю іншого блокового елемента. Елементи, що випливають за плаваючим елементом, зрушуються щодо його позиції і як би обтікають його відповідно праворуч або ліворуч (див. також опис властивості **clear**). Так, використання плаваючих абзаців дозволяє виводити їхній пліч-о-пліч, наприклад (Netscape Navigator 4.x відображає цей приклад невірно!):

Це абзац, вирівняний уліво.

А це абзац, вирівняний вправо.

Для виводу цього приклада використаний наступний фрагмент:

```
<P style="float: left; width: 50%">Це абзац, вирівняний уліво.</P>  
<P style="text-align: right"> А це абзац, вирівняний вправо.</P>
```

Плаваючий елемент повинен мати явно задану ширину. Вона задається або властивістю **width**, або самим елементом, якщо це, приміром, графічний образ.

У наступному прикладі абзац складається з тексту й двох плаваючих графічних образів, вирівняних відповідно вліво й вправо:

```
<P>  
  <IMG src="images/left.gif" alt="Стрілка вліво" style="float: left">  
  <IMG src="images/right.gif" alt="Стрілка вправо" style="float: right">  
  Це текст абзацу, що буде розташований між плаваючими образами.  
</P>
```

Цей фрагмент буде відображатися так (Netscape Navigator 4.x відображає його невірно!):

Це текст абзацу, що буде розташований між плаваючими образами.

2.5.7. Керування обтіканням тексту: властивість clear

Синтаксис: clear: none | left | right | both | [inherit](#)

Начально: none

Застосовно: до блокових елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту; реалізовано з помилками (4.0+)

Властивість **clear** указує, як даний елемент *не може обтікати* попередній плаваючий елемент. Він може приймати наступні значення:

<i>none</i>	Дозволяє обтікання плаваючого елемента по обидва боки.
<i>left</i>	Забороняє обтікання плаваючого елемента ліворуч.
<i>right</i>	Забороняє обтікання плаваючого елемента праворуч.
<i>both</i>	Забороняє обтікання плаваючого елемента по обидва боки.

Заборона обтікання означає, що даний елемент повинен відображатися нижче попереднього плаваючого елемента. За замовчуванням ця властивість дорівнює *none*, тобто дозволяє обтікання плаваючого елемента по обидва боки. Модифікуємо приклад з попереднього розділу, щоб продемонструвати застосування цієї властивості:

```
<P>
  <IMG src="images/left.gif" alt="Стрілка вліво" style="float: left">
  <IMG src="images/right.gif" alt="Стрілка вправо" style="float: right">
  Це текст абзацу, що буде розташований між плаваючими образами.
  <SPAN style="clear: both">Цей текст буде виведений нижче іншого
фрагмента</SPAN>
</P>
```

Цей фрагмент буде відображатися так (Netscape Navigator 4.x відображає цей приклад невірно!):

Це текст абзацу, що буде розташований між плаваючими образами. Цей текст буде виведений нижче іншого фрагмента.

2.5.8. Напрямок виводу тексту

Всім символам у кодуванні Unicode приписане напрямок, для того, щоб текст відображався правильно. Так, латинські й російські букви виводяться ліворуч праворуч, а єврейські й арабські - праворуч ліворуч.

Unicode визначає двунаправлений алгоритм, що повинен застосовуватися щораз, коли документ містить символи, виведені праворуч ліворуч. Хоча звичайно цей алгоритм дає правильне зображення тексту, існують ситуації, коли напрямок виводу тексту доводиться задавати явно за допомогою властивостей **direction** або **unicode-bidi**.

2.5.8.1. Завдання напрямку виводу тексту: властивість **direction**

Синтаксис: `direction: ltr | rtl | inherit`

Начально: `ltr`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: визуальные

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

Властивість **direction** визначає *напрямок виводу тексту елемента*: ліворуч праворуч (*ltr*, прийнято за замовчуванням) або праворуч ліворуч (*rtl*). При цьому воно перебиває двунаправлений алгоритм

Unicode. Крім того, воно задає напрямок виводу стовпців таблиць, напрямок горизонтального переповнення (див. **overflow**) і положення останнього неповного рядка в блоці із властивістю **"text-align: justify"**.

Для того, щоб ця властивість впливала на відображення текстових елементів, значення властивості **unicode-bidi** повинне бути дорівнює *embed* або *override*.

2.5.8.2. Керування двунаправленим алгоритмом Unicode: властивість **unicode-bidi**

Синтаксис: `unicode-bidi: normal | embed | bidi-override | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

Властивість **unicode-bidi** задає рівень вкладення для двунаправленого алгоритму Unicode. Воно може приймати наступні значення:

<i>normal</i>	Елемент не відкриває нового рівня вкладення.
<i>embed</i>	Елемент відкриває новий рівень вкладення, що задається властивістю direction і двунаправленим алгоритмом.
<i>bidi-override</i>	Елемент відкриває новий рівень вкладення, напрямок виводу тексту задається тільки властивістю direction ; двунаправлений алгоритм ігнорується.

Двунаправлений алгоритм Unicode автоматично перевертає вкладені ланцюжки символів відповідно до властивим його напрямком виводу. У тих випадках, коли оглядач все-таки не може правильно відобразити складну структуру вкладених друг у друга фраз на мовах з різним напрямком виводу тексту, варто користуватися цією властивістю для управління відображення цими фразами.

Глава 2.6. Візуалізація елементів

2.6.1. Ширина вмісту: властивість **width**

Синтаксис: `width: <розмір> | <процент> | auto | inherit`

Начально: `auto`

Застосовно: до всіх елементів, крім текстових, рядків таблиць і груп рядків

Наслідувано: немає

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Відповідає стандарту (4.0+)

Властивість **width** задає *ширину вмісту* при генерації об'ємлющего прямокутника блокових елементів й елементів, зовнішніх стосовно CSS (тобто графічних образів, об'єктів і деяких елементів форм). Ширина не може бути негативною й задається одним з наступних способів:

[<розмір>](#)

Задає фіксований розмір.

[<процент>](#)

Обчислюється щодо ширини блоку, що вміщає.

`auto`
Див. [раздел 2.6.2.](#)

Приклад: наступне правило встановлює ширину абзацу, рівної 400 пікселів.

```
P { width: 400px }
```

2.6.2. Обчислення ширин і границь

Обчислені значення властивостей [width](#), [margin-left](#), [margin-right](#), [left](#) і [right](#) залежать від типу генерируемого прямокутника. Як правило, обчислені значення збігаються зі специфікованим (крім значення *auto*, що замінюється на обчислене значення за замовчуванням), але є й виключення. Розрізняються наступні типи елементів:

- блокові й текстові;
- зовнішні стосовно CSS та інші;
- плаваючі й немає;
- абсолютно позиціоновані й інші.

Зверніть увагу, що для відносно позиціонованих елементів значення перерахованих властивостей обчислюються так само, як для непозиціонованих.

Не зовнішні текстові елементи

Специфіковане значення *auto* властивостей [margin-left](#), [margin-right](#), [left](#) і [right](#) замінюється обчисленим значенням 0. Властивість [width](#) тут не застосовно, тому що ширина вмісту однозначно визначається самим вмістом елемента.

Зовнішні текстові елементи

Специфіковане значення *auto* властивостей [margin-left](#), [margin-right](#), [left](#) і [right](#) замінюється обчисленим значенням 0. Специфіковане значення *auto* властивості [width](#) замінюється обчисленим значенням ширини відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа).

Не зовнішні блокові елементи у звичайному потоці елементів

Специфіковане значення *auto* властивостей [left](#) і [right](#) замінюється обчисленим значенням 0. Інші властивості зв'язані між собою наступним співвідношенням:

$$\text{margin-left} + \text{border-left-width} + \text{padding-left} + \text{width} + \text{padding-right} + \text{border-right-width} + \text{margin-right} = \text{ширина блоку, що вміщає}$$

Поэтому, якщо ми явно задамо значення всіх цих властивостей, відмінні від *auto*, те одне зі значень буде змінено оглядачем відповідно до наведеного співвідношення (звичайно це [margin-left](#), якщо **direction** = *ltr*, і [margin-right](#), якщо **direction** = *rtl*). Якщо рівно одне значення задане як *auto*, то воно обчислюється з наведеного співвідношення. Якщо [width](#) дорівнює *auto*, те всі інші значення *auto* замінюються на 0, і ширина обчислюється з наведеного співвідношення. Якщо [margin-left](#) і [margin-right](#) рівні *auto*, те їхнє обчислене значення будуть рівні між собою.

Зовнішні блокові елементи у звичайному потоці елементів

Специфіковане значення *auto* властивостей [left](#) і [right](#) замінюється обчисленим значенням 0. Інші властивості зв'язані між собою наведеним вище співвідношенням.

Якщо [width](#) дорівнює *auto*, те воно замінюється обчисленим значенням ширини відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа). Якщо одна із границь дорівнює *auto*, то її значення обчислюється з наведеного співвідношення. Якщо [margin-left](#) і [margin-right](#) рівні *auto*, те їхнє обчислене значення будуть рівні між собою.

Не зовнішні плаваючі елементи

Специфіковане значення *auto* властивостей [width](#), [margin-left](#), [margin-right](#), [left](#) і [right](#) замінюється обчисленим значенням 0.

Зовнішні плаваючі елементи

Специфіковане значення *auto* властивостей [margin-left](#), [margin-right](#), [left](#) і [right](#) замінюється обчисленим значенням 0. Специфіковане значення *auto* властивості [width](#) замінюється обчисленим значенням ширини відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа).

Не зовнішні абсолютно позиціоновані елементи

Обчислені значення властивостей зв'язані між собою наступним співвідношенням:

$$\text{left} + \text{margin-left} + \text{border-left-width} + \text{padding-left} + \text{width} + \text{padding-right} + \text{border-right-width} + \text{margin-right} + \text{right} = \text{ширина блоку, що вміщає}$$

При цьому обчислення значень властивостей виробляється в наступному порядку:

1. Якщо **left** дорівнює *auto*, а **direction** дорівнює *ltr*, то *auto* замінюється на відстань від лівого краю блоку, що вміщає, до лівої границі гіпотетичного прямокутника, що був би першим об'ємлющим прямокутником даного елемента, якби він був статичним. Це значення негативно, якщо гіпотетичний прямокутник розташований ліворуч від об'ємлющего блоку.
2. Если **right** дорівнює *auto*, а **direction** дорівнює *rtl*, то *auto* замінюється на відстань від правого краю блоку, що вміщає, до правої границі гіпотетичного прямокутника, що був би першим об'ємлющим прямокутником даного елемента, якби він був статичним. Це значення негативно, якщо гіпотетичний прямокутник розташований праворуч від об'ємлющего блоку.
3. Якщо [width](#) дорівнює *auto*, то значення *auto* для **left** і **right** замінюються на 0.
4. Якщо **left**, **right** або [width](#) усе ще рівні *auto*, то значення *auto* для [margin-left](#) і [margin-right](#) замінюються на 0.
5. Якщо [margin-left](#) і [margin-right](#) усе ще рівні *auto*, то їхні значення обчислюються з наведеного вище співвідношення так, щоб вони були рівні між собою.
6. Якщо до цього моменту тільки одне значення дорівнює *auto*, то воно обчислюється з наведеного співвідношення.
7. Якщо виявилось, що було задано занадто багато явних значень, то перераховується на основі наведеного співвідношення значення **left** (якщо [direction](#) дорівнює *ltr*) або **right** (якщо [direction](#) дорівнює *rtl*).

Зовнішні абсолютно позиціоновані елементи

Обчислення виробляються так само, як у попередньому випадку, але з однією відмінністю. Якщо [width](#) дорівнює *auto*, то воно замінюється обчисленим значенням ширини відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа), а потім виробляються всі перераховані розрахунки.

2.6.3. Мінімальна й максимальна ширина: властивості `min-width` й `max-width`

Синтаксис: `min-width: <размер> | <процент> | inherit`
Начально: визначається оглядачем
Застосовно: до всіх елементів, крім текстових і табличних
Наслідувано: немає
Відсотки: щодо ширини блоку, що вміщає
Устройства: [визуальные](#)
Синтаксис: `max-width: <размер> | <процент> | none | inherit`
Начально: `none`
Застосовно: до всіх елементів, крім текстових і табличних
Наслідувано: немає
Відсотки: щодо ширини блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Не підтримуються
Не підтримуються

Ці властивості дозволяють нам обмежити ширину вмісту при генерації об'ємлющего прямокутника мінімально й максимально припустимими значеннями. Ширина не може бути негативною й задається одним з наступних способів:

<размер>
Задає фіксований розмір.
<процент>
Обчислюється щодо ширини блоку, що вміщає.
none
Необмежена ширина прямокутника.

Ці властивості застосовуються в такий спосіб. Спочатку обчислюється властивість **width**, як описано в [розділі 2.6.2](#). Потім перевіряється, чи попадає воно в діапазон від **min-width** до **max-width**, і, якщо ні, те обчислене значення **width** замінюється на відповідне гранично припустиме значення.

Приклад: наступне правило встановлює межі для ширини абзацу.

```
P { min-width: 20px; max-width: 400px }
```

2.6.4. Висота вмісту: властивість **height**

Синтаксис: `height: <размер> | <процент> | auto | inherit`
Начально: `auto`
Застосовно: до всіх елементів, крім текстових, стовпців таблиць і груп стовпців
Наслідувано: немає
Відсотки: щодо висоти блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)
Відповідає стандарту (4.0+)

Властивість **height** задає *висоту вмісту* при генерації об'ємлющего прямокутника блокових елементів й елементів, зовнішніх стосовно CSS (тобто графічних образів, об'єктів, аплетів і т.п.). Відзначимо, що висота текстових елементів задається властивістю **line-height**. Висота не може бути негативною й задається одним з наступних способів:

<размер>
Задає фіксований розмір.
<процент>
Обчислюється щодо висоти блоку, що вміщає. Якщо ця висота явно не задана, те значення інтерпретується як `auto`.
`auto`
Див. [раздел 2.6.5](#).

Приклад: наступне правило встановлює висоту абзацу, рівної 100 пікселів.

```
P { height: 100px }
```

Якщо висота якогось абзацу буде перевищувати 100 пікселів, то такий абзац відображається у відповідності зі значенням властивості **overflow**.

2.6.5. Обчислення висот і границь

Обчислені значення властивостей [height](#), [margin-top](#), [margin-bottom](#), [top](#) і [bottom](#) залежать від типу генерируемого прямокутника. Як правило, обчислене значення збігається зі специфікованим (крім значення *auto*, що замінюється на обчислене значення за замовчуванням), але є й виключення. Розрізняються наступні типи елементів:

- блокові й текстові;
- зовнішні стосовно CSS та інші;
- плаваючі й немає;
- абсолютно позиціоновані й інші.

Зверніть увагу, що для відносно позиціонованих елементів значення перерахованих властивостей обчислюються так само, як для непозиціонованих.

Не зовнішні текстові елементи

Специфіковане значення *auto* властивостей [margin-top](#), [margin-bottom](#), [top](#) і [bottom](#) замінюється обчисленим значенням 0. Властивість [height](#) тут не застосовно, але висоту такого елемента можна задати властивістю **line-height**.

Зовнішні елементи (текстові, блокові у звичайному потоці елементів і плаваючі)

Специфіковане значення *auto* властивостей [margin-top](#), [margin-bottom](#), [top](#) і [bottom](#) замінюється обчисленим значенням 0. Специфіковане значення *auto* властивості **height** замінюється обчисленим значенням висоти відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа).

Не зовнішні блокові елементи (у звичайному потоці елементів або плаваючі)

Специфіковане значення *auto* властивостей [margin-top](#), [margin-bottom](#), [top](#) і [bottom](#) замінюється обчисленим значенням 0. Якщо [height](#) дорівнює *auto*, те його обчислене значення залежить від того, чи має даний елемент блокових дітей. Якщо всі його діти є текстовими, то висота елемента обчислюється як відстань від верхньої лінії самого верхнього рядкового блоку до нижньої лінії самого нижнього рядкового блоку. Якщо ж елемент має блокових дітей, то його висота обчислюється як відстань від верхнього краю рамки самого верхнього блоку до нижнього краю рамки самого нижнього блоку. При цьому враховуються тільки ті діти, які перебувають у звичайному потоці елементів (іншими словами, що плавають й абсолютно позиціоновані елементи ігноруються, а відносно позиціоновані враховуються без свого зсуву).

Не зовнішні абсолютно позиціоновані елементи

Обчислені значення властивостей зв'язані між собою наступним співвідношенням:

$$\text{top} + \text{margin-top} + \text{border-top-width} + \text{padding-top} + \text{height} + \text{padding-bottom} + \text{border-bottom-width} + \text{margin-bottom} + \text{bottom} = \text{висота блоку, що вміщає}$$

При цьому обчислення значень властивостей виробляється в наступному порядку:

1. Якщо **top** дорівнює *auto*, те *auto* замінюється на відстань від верхнього краю блоку, що вміщає, до верхньої границі гіпотетичного прямокутника, що був би першим об'ємлющим прямокутником даного елемента, якби він був статичним. Це значення негативно, якщо гіпотетичний прямокутник розташований вище об'ємлющего блоку.
2. Если й **height**, і **bottom** рівні *auto*, те значення *auto* для **bottom** замінюється на 0.
3. Якщо **bottom** або [height](#) усе ще рівні *auto*, те значення *auto* для [margin-top](#) і [margin-bottom](#) замінюються на 0.
4. Якщо [margin-top](#) і [margin-bottom](#) усе ще рівні *auto*, те їхні значення обчислюються з наведеного вище співвідношення так, щоб вони були рівні між собою.
5. Якщо до цього моменту тільки одне значення дорівнює *auto*, то воно обчислюється з наведеного співвідношення.

6. Якщо виявилось, що було задано занадто багато явних значень, то перераховується на основі наведеного співвідношення значення **bottom**.

Зовнішні абсолютно позиціоновані елементи

Обчислення виробляються так само, як у попередньому випадку, але з однією відмінністю. Якщо [height](#) дорівнює *auto*, те воно замінюється обчисленням значення висоти відповідного об'єкта (яка задається або його власними розмірами, або атрибутом мови документа), а потім виробляються всі перераховані розрахунки.

2.6.6. Мінімальна й максимальна висота: властивості **min-height** й **max-height**

Синтаксис: `min-height: <размер> | <процент> | inherit`
Начально: визначається оглядачем
Застосовно: до всіх елементів, крім текстових і табличних
Наслідувано: немає
Відсотки: щодо висоти блоку, що вміщає
Устройства: [визуальные](#)
Синтаксис: `max-height: <размер> | <процент> | none | inherit`
Начально: *none*
Застосовно: до всіх елементів, крім текстових і табличних
Наслідувано: немає
Відсотки: щодо висоти блоку, що вміщає
Устройства: [визуальные](#)

Підтримка: Не підтримуються

Не підтримуються

Ці властивості дозволяють нам обмежити висоту вмісту при генерації об'ємлюючого прямокутника мінімально й максимально припустимими значеннями. Висота не може бути негативною й задається одним з наступних способів:

[<размер>](#)
Задає фіксований розмір.

[<процент>](#)
Обчислюється щодо висоти блоку, що вміщає. Якщо ця висота явно не задана,
те значення інтерпретується як *auto*.

none
Необмежена висота прямокутника.

Ці властивості застосовуються в такий спосіб. Спочатку обчислюється властивість **height**, як описано в [розділі 2.6.5](#). Потім перевіряється, чи попадає воно в діапазон від **min-height** до **max-height**, і, якщо ні, те обчислене значення **height** замінюється на відповідне гранично припустиме значення.

Приклад: наступне правило встановлює межі для висоти абзацу.

```
p { min-height: 200px; max-height: 1000px }
```

2.6.7. Відображення текстових елементів

При відображенні текстових елементів їхні прямокутники розташовуються горизонтально один за одним. При цьому їхнє взаємне розташування по вертикалі визначається властивістю **vertical-align**. Прямокутна область, що містить прямокутники, що утворюють рядок тексту, називається *рядковим блоком*. Ширина рядкового блоку визначається шириною блоку, що вміщає. Його висота обчислюється на основі висот всіх прямокутників даного рядка й може бути змінена властивістю **line-height**.

2.6.7.1. Висота тексту: властивість **line-height**

Синтаксис: `line-height: normal | <число> | <размер> | <процент> | inherit`
Начально: `normal`
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: щодо розміру шрифту елемента
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **line-height** задає висоту рядка тексту при генерації об'ємлюючого прямокутника. Якщо елемент є блоковим, то ця властивість задає мінімальну висоту вхідних у нього рядкових блоків. Якщо елемент є текстовим, то ця властивість задає точну висоту його рядкового блоку.

Висота не може бути негативною й задається одним з наступних способів:

`normal`

Висота рядкового блоку за замовчуванням, що оглядач визначає на основі розміру його шрифту.

`<число>`

Обчислене значення дорівнює даному числу, помноженому на розмір шрифту.

`<размер>`

Задає фіксований розмір.

`<процент>`

Обчислюється щодо розміру шрифту елемента.

Наступні три правила дадуть той самий результат:

```
DIV { font-size: 10pt; line-height: 1.2 }  
DIV { font-size: 10pt; line-height: 1.2em }  
DIV { font-size: 10pt; line-height: 120% }
```

Якщо текст елемента відображається декількома шрифтами, то вибирається найбільший з розмірів цих шрифтів.

2.6.7.2. Вертикальне вирівнювання: властивість **vertical-align**

Синтаксис: `vertical-align: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <размер> | <процент> | inherit`
Начально: `baseline`
Застосовно: до текстових елементів й осередків таблиць
Наслідувано: немає
Відсотки: щодо значення **line-height** елемента
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **vertical-align** задає положення по вертикалі об'ємлюючого прямокутника текстового елемента усередині утримуючого його рядкового блоку. Воно може приймати наступні значення:

<code>baseline</code>	Вирівнювання по базовій лінії рядкового блоку.
<code>sub</code>	Розташування в позиції нижніх індексів (але без зміни розміру шрифту).

<i>super</i>	Розташування в позиції верхніх індексів (але без зміни розміру шрифту).
<i>top</i>	Вирівнювання по верху рядкового блоку.
<i>text-top</i>	Вирівнювання по верхній лінії шрифту рядкового блоку.
<i>middle</i>	Вирівнювання по центрі рядкового блоку.
<i>bottom</i>	Вирівнювання по низі рядкового блоку.
<i>text-bottom</i>	Вирівнювання по нижній лінії шрифту рядкового блоку.
<u><размер></u>	Збільшити (позитивне значення) або зменшити (негативне значення) на дану величину .
<u><процент></u>	Збільшити (позитивне значення) або зменшити (негативне значення) на дану величину у відсотках від значення <u>line-height</u> .

Приклад:

```
EM.center { vertical-align: middle }
```

2.6.8. Переповнення й обрізка

У більшості випадків вміст блокового елемента розташовується усередині краю вмісту його об'ємлюючого прямокутника. Однак, у деяких випадках відбувається переповнення, тобто частина вмісту виявляється за межами цього прямокутника, наприклад:

- Рядок тексту не може бути розірвана, і рядковий блок стає ширше блокового прямокутника.
- Блоковий прямокутник занадто широкий для блоку, що вміщає.
- Висота елемента більше, ніж значення властивості **height** блоку, що вміщає.
- Елемент абсолютно позиціонований.
- Об'ємлюючий прямокутник має негативні границі.

Щораз, коли відбувається переповнення, значення властивості **overflow** указує як треба (і чи треба взагалі) обрізати вміст блоку; розмір і форма області обрізки при цьому задається властивістю **clip**.

2.6.8.1. Реакція на переповнення: властивість overflow

Синтаксис: `overflow: visible | hidden | scroll | auto | inherit`

Начально: `visible`

Застосовно: до блокових і зовнішніх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: визуальные

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **overflow** визначає *правила обрізки вмісту* елемента при переповненні. Воно може приймати наступні значення:

<i>visible</i>	Уміст не обрізається , тобто відображається за межами об'ємлюючого прямокутника.
<i>hidden</i>	Уміст обрізається; розмір і форма області обрізки при цьому задається властивістю clip .
<i>scroll</i>	Уміст обрізається , але оглядач повинен забезпечити механізм прокручування вмісту для того, щоб користувач міг одержати доступ до всього вмісту елемента. При відображенні на пристрої print й projection повинне друкуватися весь уміст елемента.
<i>auto</i>	Уміст обрізається , і при необхідності забезпечується прокручування вмісту.

Розглянемо наступний приклад.

```
<DIV style="width: 100px; height: 100px; border: thin solid red;
overflow: scroll">
  <BLOCKQUOTE style="width: 125px; height: 100px; margin-top: 50px;
margin-left: 50px; border: thin dashed black">
    <P>Якби будівельники будували будинки так само, як програмісти пишуть
    програми, перший дятел, що залетів, зруйнував би цивілізацію</P>
    <P style="text-align: right">Другий закон Вейнберга</P>
  </BLOCKQUOTE>
</DIV>
```

Тут елемент **BLOCKQUOTE** виходить за межі утримуючого його елемента **DIV**, тому цей фрагмент буде відображатися так (Netscape Navigator 4.x не підтримує обрізку й прокручування!):

Якби будівельники будували будинки так само, як програмісти пишуть програми, перший дятел, що залетів, зруйнував би цивілізацію.

Другий закон Вейнберга

2.6.8.2. Область обрізки: властивість **clip**

Синтаксис: `clip: rect(<top>,<right>,<bottom>,<left>) | auto | inherit`
Начально: `auto`
Застосовно: до блокових і зовнішніх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Підтримується тільки для абсолютно позиціонованих елементів (4.0+)

Підтримується тільки для шарів (layers) (4.0+)

Властивість **clip** задає *область обрізки вмісту* елемента при переповненні. Воно може приймати наступні значення:

`auto`
Область обрізки збігається з об'ємлющим прямокутником.
`rect(<top>,<right>,<bottom>,<left>)`
Область вирізки є прямокутником із заданими координатами, де `<top>`, `<right>`, `<bottom>` й `<left>` — це зсуви від відповідних сторін об'ємлющего прямокутника. Кожне із цих зсувів повинне бути дорівнює або `auto` (що означає нульовий зсув), або [<размер>](#). Негативні зсуви допускаються.

Приклад:

```
P { clip: rect(5px, -5px, 10px, 5px); }
```

2.6.9. Видимість елемента: властивість **visibility**

Синтаксис: `visibility: visible | hidden | collapse | inherit`
Начально: `inherit`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Підтримуються тільки значення `visible`, `hidden` й `inherit` (4.0+)

Підтримується тільки для шарів (`layers`) (4.0+)

Властивість **visibility** задає *видимість* елемента при відображенні. Воно може приймати наступні значення:

<code>visible</code>	Объемлющий прямокутник елемента бачимо.
<code>hidden</code>	Объемлющий прямокутник елемента не бачимо.
<code>collapse</code>	Те ж, що <code>hidden</code> . Застосовується тільки в таблицях, де має особливе призначення (див. п. 2.10.2).

Примітки:

- Невидимість елемента означає, що його объемлющий прямокутник стає прозорим, але продовжує займати своє місце в структурі відображення. Для того, щоб повністю видалити елемент, варто використати значення `none` властивості **display**.
- Ця властивість широко застосовується в сценаріях для створення динамічних ефектів.

2.6.10. Форма курсору: властивість `cursor`

Синтаксис: `cursor: [[<uri>,* [auto|crosshair|default|pointer|move|e-resize|ne-resize|nw-resize|n-resize|se-resize|sw-resize|s-resize|w-resize|text|wait|help]] |inherit`

Начально: `auto`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные интерактивные](#)

Підтримка: Замість `pointer` використовується `hand`, список `<uri>` не підтримується (4.0+)

Не підтримується

Властивість **cursor** задає *форму курсору миші* при наведенні її на даний елемент. Воно може приймати наступні значення:

<code>auto</code>	Форма курсору визначається оглядачем залежно від умісту елемента.
<code>crosshair</code>	Перехрестя.
<code>default</code>	Курсор операційної системи за замовчуванням. В Windows це стрілка.
<code>pointer</code>	Показчик на гіперпосилання. Звичайно це кисть руки.
<code>move</code>	Схрещені стрілки, що вказують на те, що щось повинне переміститися.
<code>*-resize</code>	Стрілка, що вказує на зрули границю, що; * задає напрямок зрушення (<i>n</i> = північ, <i>s</i> = південь, <i>w</i> = захід, <i>e</i> = схід і т.п.).
<code>text</code>	Указує на редагує текст, що. Звичайно має форму букви I .
<code>wait</code>	Піскові годинники, що вказують на те, що програма зайнята , і користувач повинен почекати.
<code>help</code>	Знак питання, що вказує, що для даного елемента можна одержати довідку.

<code><uri></code>	Задає список URI ресурсів, що містять курсори.
--	--

Якщо значення даного списку задано списком URI, то оглядач намагається спочатку завантажити перший курсор зі списку, при невдачі — другої й т.д. Рекомендується закінчувати такий список одним із ключових слів, щоб при неможливості завантажити курсор оглядач знав, яку з визначених форм курсорів йому використати, наприклад:

```
P { cursor: url("mycursor.cur"), url("second.csr"), text }
```

2.6.11. Динамічні контури

2.6.11.1. Загальний опис

CSS допускає створення навколо відображуваних елементів динамічних контурів. Це зручно для виділення таких об'єктів, як кнопки, активні поля форм або карти посилань. Контури відрізняються від рамок у двох відношеннях: по-перше, вони не займають екранного простору, по-друге, не зобов'язані мати прямокутну форму.

Відображення контуру виробляється поверх навколишніх елементів, безпосередньо від краю рамки даного елемента. Сценарії забезпечують можливість динамічного оконтурювання елементів. Крім того, для цього можна використати псевдокласи **:focus** і **:active**. На жаль, оглядачі поки не підтримують перерахованих тут властивостей.

2.6.11.2. Розмір контуру: властивість **outline-width**

Синтаксис: `outline-width: <розмір-рамки> | inherit`

Начально: `medium`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные интерактивные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **outline-width** задає *розмір динамічного контуру*. Воно приймає ті ж значення, що й **border-width**. Приклад:

```
A:active { outline-width: thick }
```

2.6.11.3. Стиль контуру: властивість **outline-style**

Синтаксис: `outline-style: <стиль-рамки> | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные интерактивные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **outline-style** задає *стиль динамічного контуру*. Воно приймає ті ж значення, що й **border-style**. Приклад:

```
A:focus { outline-style: double }
```

2.6.11.4. Кольори контуру: властивість **outline-color**

Синтаксис: `outline-color: <цвет> | invert | inherit`

Начально: `invert`

Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные интерактивные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **outline-color** задає *кольори динамічного контуру*. Його значеннями можуть бути будь-які кольори, а також *invert*, що означає, що кольори контуру повинен бути отриманий обігом квітів пікселів екрана. Приклад:

```
A:focus { outline-color: red }
```

2.6.11.5. Властивості контуру: властивість outline

Синтаксис: `outline: [<outline-width>|<outline-style>|<outline-color>]|inherit`
Начально: див. індивідуальні властивості
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные интерактивные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **outline** задає *розмір, стиль і кольори контуру* одночасно. Якщо якесь із цих властивостей не задано, то приймається його початкове значення. Приклад:

```
A:active { outline: thick solid red }
```

Глава 2.7. Кольори й фон

2.7.1. Кольори тексту: властивість color

Синтаксис: `color: <цвет> | inherit`
Начально: залежить від оглядача
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **color** задає *кольори тексту* вмісту елемента при відображенні. Приклади:

```
H1 { color: blue }  
H2 { color: #000080 }  
H3 { color: #0c0 }
```

Примітка. У запобіганні конфлікту з користувацькою таблицею стилів, ця властивість варто задавати разом із властивістю **background**.

2.7.2. Завдання тла

Тло елемента може задаватися або кольорами (**background-color**), або графічним образом (**background-image**, **background-repeat**, **background-attachment** і **background-position**). Можна також користуватися скороченою властивістю **background**. Нагадаємо, що тло елемента поширюється на область умісту елемента і його заповнювача.

Тло кореня дерева документів є тлом для відображення документа в цілому, але для HTML-документів рекомендується задавати загальне тло у властивостях елемента **BODY**, а не **HTML**.

2.7.2.1. Кольори тла: властивість **background-color**

Синтаксис: `background-color: <цвет> | transparent | inherit`
Начально: `transparent`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **background-color** задає *кольори тла* елемента при відображенні. Значення *transparent*, прийняте за замовчуванням, відповідає прозорому тлу. Приклади:

```
BODY { background-color: white }
H1 { background-color: #0000F0 }
H2 { background-color: #0cc }
```

Примітка. У запобіганні конфлікту з користувальницькою таблицею стилів, при завданні цієї властивості варто задавати також властивість **background-image** (звичайно зі значенням *none*).

2.7.2.2. Графічний образ тла: властивість **background-image**

Синтаксис: `background-image: <uri> | none | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **background-image** задає *графічний образ тла* елемента при відображенні. Значення *none*, прийняте за замовчуванням, означає відсутність фонового образу. Приклади:

```
BODY { background-image: url("marble.gif") }
P { background-image: none }
```

Примітка. При завданні цієї властивості варто задавати також властивість **background-color** на випадок неможливості завантаження графічного образу. Якщо задані обидві ці властивості, то кольори тла буде видний у прозорих областях образу.

2.7.2.3. Повтор фонових образів: властивість **background-repeat**

Синтаксис: `background-repeat: repeat | repeat-x | repeat-y | no-repeat | inherit`
Начально: `repeat`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Якщо задано графічний образ тла, то властивість **background-repeat** задає *повтор фонового образа* при відображенні. Його можливі значення:

<i>repeat</i>	Образ повторюється по горизонталі й вертикалі.
<i>repeat-x</i>	Образ повторюється тільки по горизонталі.
<i>repeat-y</i>	Образ повторюється тільки по вертикалі.
<i>no-repeat</i>	Образ не повторюється: відображається тільки одна його копія.

Приклад:

```
BODY {  
  background: white url("image.gif");  
  background-repeat: repeat-y;  
  background-position: center;  
}
```

2.7.2.4. Прокручування фонового образа: властивість background-attachment

Синтаксис: background-attachment: *scroll* | *fixed* | *inherit*

Начально: *scroll*

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Якщо задано графічний образ тла, то властивість **background-attachment** задає *прокручування фонового образа* при відображенні. Його можливі значення:

<i>scroll</i>	Образ прокручується разом з документом.
<i>fixed</i>	Образ зафіксований і не прокручується.

Приклад:

```
BODY {  
  background: white url("image.gif");  
  background-repeat: repeat-y;  
  background-attachment: fixed;  
}
```

2.7.2.5. Позиція фонового образа: властивість background-position

Синтаксис: background-position: [[<размер> | <процент>]{1,2} |
[[*top*|*center*|*bottom*] | |
[[*left*|*center*|*right*]]] | *inherit*

Начально: 0% 0%

Застосовно: до блокових і зовнішніх елементів

Наслідувано: немає

Відсотки: щодо розмірів об'ємлющего прямокутника

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Якщо задано графічний образ тла, то властивість **background-position** задає *позицію фонового образу* при відображенні. Його можливі значення:

<u><размер></u> <u><размер></u>	Якщо ця пара значень дорівнює "2cm 3cm", то лівий верхній кут образу міститься на 2 див правее й 3 див нижче лівого верхнього угола області заповнювача.
<u><процент></u> <u><процент></u>	Якщо ця пара значень дорівнює "0% 0%", то лівий верхній кут образу сполучається з лівим верхнім кутом області заповнювача. Якщо ця пара значень дорівнює "100% 100%", то правий нижній кут образу сполучається із правим нижнім кутом області заповнювача. Якщо ця пара значень дорівнює "10% 80%", то лівий верхній кут образу міститься на 10% правее й 80% нижче лівого верхнього угола області заповнювача.

Якщо задано тільки одне значення, то воно вважається позицією по горизонталі, а позиція по вертикалі приймається рівної 50%. Якщо дані два значення, то позиція по горизонталі коштує першої. Дозволяються комбінації розмірів і відсотків, наприклад "2cm 30%". Негативні позиції також припустимі. Крім того, значення цієї властивості може задаватися наступними комбінаціями ключових слів (і тільки ними):

<i>top left = left top</i>	Те ж, що "0% 0%".
<i>top = top center = center top</i>	Те ж, що "50% 0%".
<i>right top = top right</i>	Те ж, що "100% 0%".
<i>left = left center = center left</i>	Те ж, що "0% 50%".
<i>center = center center</i>	Те ж, що "50% 50%".
<i>right = right center = center right</i>	Те ж, що "100% 50%".
<i>bottom left = left bottom</i>	Те ж, що "0% 100%".
<i>bottom = bottom center = center bottom</i>	Те ж, що "50% 100%".
<i>bottom right = right bottom</i>	Те ж, що "100% 100%".

Приклад:

```
BODY {  
  background-image: url("banner.jpg");  
  background-position: top center;  
}
```

Якщо фоновий образ зафіксований (див. властивість **background-attachment**), те його зсуви обчислюються щодо вікна оглядача, а не області заповнення елемента. У наступному прикладі фоновий образ буде розміщений у правому нижньому куті вікна оглядача.

```
BODY {  
  background-image: url("logo.png");  
  background-attachment: fixed;  
  background-position: 100% 100%;  
  background-repeat: no-repeat;  
}
```

2.7.2.6. Властивості тла: властивість background

Синтаксис: background: [<background-color>] | [<background-image>] |

[<background-repeat>](#) || [<background-attachment>](#) ||
[<background-position>](#)] | *inherit*

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: застосовні тільки в [<background-position>](#)

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **background** є скороченням для властивостей [background-color](#), [background-image](#), [background-repeat](#), [background-attachment](#) і [background-position](#). Воно дозволяє задати всі властивості тла одночасно. При цьому спочатку всім властивостям тла привласнюються їхні початкові значення, а потім змінюються значення тих властивостей, які явно задані в даній властивості.

У першому правилі наступного приклада задається тільки кольори тла, у другому зазначені всі властивості тла:

```
BODY { background: blue }  
P { background: url("bkgrd.gif") gray 50% repeat fixed }
```

Примітка. У запобіганні конфлікту з користувацькою таблицею стилів, ця властивість варто задавати разом із властивістю **color**

Глава 2.8. Шрифти

2.8.1. Введення

Уже в стандарті CSS1 була закладена можливість використання різних шрифтів для відображення тексту документа. При цьому передбачалося, що необхідні шрифти встановлені на комп'ютері-клієнті; якщо ж їх ні, те замість них оглядач повинен був використати власні шрифти, близькі по характеристиках до що вимагаються.

Стандарт CSS2 вніс істотні зміни в цю концепцію для того, щоб автори мали більшу волю у виборі шрифтів, а оглядачі — у своїх діях, коли заданий автором шрифт безпосередньо не доступний. В основі нової концепції лежить поняття *шрифтів, що завантажують*, тобто шрифтів, відсутніх на комп'ютері-клієнті, але доступних для завантаження з Мережі. На додаток до цьому CSS2 передбачає наявність бази даних про шрифти, що містить їхній різноманітні характеристики й необхідності, що *дозволяє в міру*, синтезувати відсутні шрифти на основі доступних оглядачеві шрифтів.

Нижче викладені всі способи роботи зі шрифтами, надавані CSS. При цьому властивості, що управляють відображенням шрифтів й успадковані з CSS1, описані в [розділі 2.8.2](#), а нові методи доступу до завантажують шрифтам, що, що з'явилися в CSS2, описуються в [розділі 2.8.3](#).

2.8.2. Завдання властивостей шрифтів

2.8.2.1. Сімейство шрифтів: властивість font-family

Синтаксис: `font-family: [[<имя-семейства>|<родовое-имя>],] *`
`[<имя-семейства>|<родовое-имя>] | inherit`

Начально: залежить від оглядача

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Властивість **font-family** задає *список імен сімейств шрифтів* для відображення вмісту елемента. Цей список складається з імен сімейств шрифтів, розділених комами. Імена сімейств розташовуються в порядку переваги. Наприклад, що впливає властивість

```
font-family: Verdana, Arial, sans-serif;
```

варто розуміти так: "використати шрифт Verdana; якщо його ні, те використати шрифт Arial; якщо його ні, те використати родовий шрифт sans-serif". Такий список необхідний, оскільки ми заздалегідь не знаємо, які саме шрифти встановлені на комп'ютерах наших користувачів.

Ім'я сімейства шрифтів може бути задано двома способами:

<імені-сімейства>

Задає назва сімейства шрифтів. Якщо ця назва містить пробіли, то воно повинне бути укладене в лапки або апострофи.

<родов-ім'я>

Одне з наступних імен: *serif, sans-serif, cursive, fantasy* й *monospace*.

Див. [раздел 2.8.2.2](#).

Для досягнення найбільшої сумісності рекомендується задавати родові ім'я шрифту останнім у списку. У цьому випадку, якщо оглядач не знайде на комп'ютері-клієнті жодного із заданих шрифтів, він використає свій родовий шрифт із заданим ім'ям.

2.8.2.2. Родові імена шрифтів

Родові імена шрифтів були розроблені на той найгірший випадок, коли на комп'ютері-клієнті не встановлений жоден зі шрифтів, заданих автором. У цьому випадку оглядач використає родовий шрифт, накреслення якого нагадує авторський шрифт. Існує п'ять імен родових шрифтів:

serif (шрифт із зарубками)

Це пропорційні шрифти із зарубками на буквах. Типові приклади: Times New Roman, Bodoni, Garamond.

sans-serif (шрифт без зарубок)

Це пропорційні шрифти без зарубок на буквах. Типові приклади: Arial, Verdana, Helvetica, Tahoma.

cursive (каліграфічний шрифт)

Це шрифти, стилізовані під рукописний текст, звичайно з типовими для нього з'єднаннями між буквами. Типовий приклад: Zapf-Chancery.

FANTASY (декоративний шрифт)

Це шрифти декоративного характеру. Типовий приклад: Western.

monospace (телетайпний шрифт)

Це шрифти фіксованої ширини, що нагадують шрифт друкарської машинки. Типові приклади: Courier New, Prestige, Everson Mono.

Те, який саме конкретний шрифт використається як реалізація родового, залежить від оглядача.

2.8.2.3. Стиль шрифту: властивість font-style

Синтаксис: `font-style: normal | italic | oblique | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту; `oblique` відображається як `italic` (4.0+)

Відповідає стандарту; `oblique` не підтримується (4.0+)

Властивість **font-style** задає *стиль шрифту* для відображення вмісту елемента. Воно може мати наступні значення:

<i>normal</i>	Звичайний шрифт
<i>italic</i>	Курсивний шрифт
<i>oblique</i>	Похилий шрифт

Наприклад, що впливає фрагмент

```
<P>
  <SPAN style="font-style: normal">Звичайний шрифт. </SPAN>
  <SPAN style="font-style: italic">Курсивний шрифт. </SPAN>
  <SPAN style="font-style: oblique">Похилий шрифт. </SPAN>
</P>
```

буде відображатися так:

Звичайний шрифт. Курсивний шрифт. Похилий шрифт.

2.8.2.4. Варіант шрифту: властивість font-variant

Синтаксис: `font-variant: normal | small-caps | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **font-variant** задає *варіант шрифту* для відображення вмісту елемента. Воно може приймати наступні значення:

<i>normal</i>	Звичайні букви
<i>small-caps</i>	Малі прописні букви

Наприклад, що впливає фрагмент

```
<P style="font-variant: small-caps">Малої прописної букви</P>
```

буде відображатися так:

МАЛІ ПРОПИСНІ БУКВИ.

Про перетворення тексту в прописні букви див. також опис властивості **text-transform**.

2.8.2.5. Жирність шрифту: властивість font-weight

Синтаксис: `font-weight: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 |`

`500 | 600 | 700 | 800 | 900 | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: так, успадковується обчислене значення

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **font-weight** задає *жирність шрифту* для відображення вмісту елемента. Воно може приймати наступні значення:

<i>100 - 900</i>	Задає жирність шрифту (чим більше значення, тим вище жирність шрифту).
<i>normal</i>	Те ж, що <i>400</i> .
<i>bold</i>	Те ж, що <i>700</i> .
<i>bolder</i>	Указує, що шрифт повинен бути більше жирним, чим батьківський шрифт.
<i>lighter</i>	Указує, що шрифт повинен бути менш жирним, чим батьківський шрифт.

Те, як саме числові значення відповідають реальній жирності шрифту, залежить від оглядача. Наприклад, що впливає фрагмент

```
<P>
  <SPAN style="font-weight: 100">100 </SPAN>
  <SPAN style="font-weight: 200">200 </SPAN>
  <SPAN style="font-weight: 300">300 </SPAN>
  <SPAN style="font-weight: 400">400 </SPAN>
  <SPAN style="font-weight: 500">500 </SPAN>
  <SPAN style="font-weight: 600">600 </SPAN>
  <SPAN style="font-weight: 700">700 </SPAN>
  <SPAN style="font-weight: 800">800 </SPAN>
  <SPAN style="font-weight: 900">900 </SPAN>
</P>
```

може відображатися так:

100 200 300 400 500 **600 700 800 900**

2.8.2.6. Включення шрифту: властивість font-stretch

Синтаксис: `font-stretch: normal | wider | narrower | ultra-condensed | extra-condensed |`

`condensed | semi-condensed | semi-expanded |`

`expanded |`

`extra-expanded | ultra-expanded | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **font-stretch** задає *выключку шрифту*, тобто інтервал між символами при відображенні вмісту елемента. Воно може мати наступні значення (у порядку збільшення выключки):

<i>ultra-condensed</i>	Сверхуплотненный шрифт
<i>extra-condensed</i>	Сильно ущільнений шрифт
<i>condensed</i>	Ущільнений шрифт
<i>semi-condensed</i>	Напівущільнений шрифт
<i>normal</i>	Звичайний шрифт
<i>semi-expanded</i>	Напіврозріджений шрифт
<i>expanded</i>	Розріджений шрифт
<i>extra-expanded</i>	Сильно розріджений шрифт
<i>ultra-expanded</i>	Сверхразрезанный шрифт

Крім того, є можливість задавати відносну розрядку шрифту:

<i>wider</i>	Указує, що шрифт повинен бути більше розрідженим, чим батьківський шрифт.
<i>narrower</i>	Указує, що шрифт повинен бути більше ущільненим, чим батьківський шрифт.

Те, як саме ці значення відповідають реальної выключке шрифту, залежить від оглядача. Приклад:

```
<P><SPAN style="font-stretch: expanded">Розрядка моя </SPAN>(Л. И.  
Брежнев) .</P>
```

2.8.2.7. Розмір шрифту: властивість **font-size**

Синтаксис: `font-size: xx-small | x-small | small | medium | large | x-large |`
`xx-large | larger | smaller | <размер> | <процент>`
`| inherit`

Начально: `medium`

Застосовно: до всіх елементів

Наслідувано: так, успадковується обчислене значення

Відсотки: щодо розміру батьківського шрифту

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **font-size** задає *розмір шрифту* для відображення вмісту елемента. Воно може задаватися декількома способами:

`xx-small | x-small | small | medium | large | x-large | xx-large`
Задає абсолютний розмір шрифту (від найменшого до найбільшого).
`smaller | larger`
Задає розмір шрифту щодо батьківського шрифту (менше або більше).
[<размер>](#)
Задає фіксований розмір. Негативні розміри неприпустимі.

<процент>

Обчислюється щодо розміру батьківського шрифту.

Фактичне значення цієї властивості може відрізнятися від обчисленого по двох причинах: якщо шрифт не має заданого розміру або під впливом властивості **font-size-adjust**.

CSS2 рекомендує, щоб послідовні значення абсолютних розмірів шрифту відрізнялися для екрана в 1.2 рази, однак це залежить від оглядача. Приклад відображення семи абсолютних розмірів шрифтів:

xx-small
x-small
small
medium
large
x-large
xx-large

2.8.2.8. Аспект шрифту: властивість font-size-adjust

Синтаксис: `font-size-adjust: none | <число> | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Властивість **font-size-adjust** задає *аспект шрифту*. Пояснимо його призначення докладніше.

У шрифтах, що містять і рядкові, і прописні букви, суб'єктивний зовнішній вигляд символу (тобто те, наскільки він здається перекрученим людському оку) залежить від співвідношення значення властивості **font-size** і величини **x-height**; у дійсності, що визначає їхнє відношення, що називається *аспектом шрифту* (aspect value) і дорівнює $font-size / x-height$. Чим менше аспект шрифту, тим скоріше шрифт стає перекрученим при зменшенні його розмірів. Тому просте завдання розміру шрифту може приводити до неприємного для ока відображенню його символів.

Допустимо тепер, що властивість **font-family** містить список з декількох назв шрифтів. Ясно, що розміри шрифту для відображення даного елемента підбиралися автором, виходячи з того, що він буде відображатися першим шрифтом у списку. Якщо ж цей шрифт відсутній на комп'ютері-клієнті, то оглядач використає другий шрифт зі списку. Але аспект цього шрифту має інше значення, чим аспект першого й, у підсумку, зовнішній вигляд відображуваного тексту може бути перекручений. Для того, щоб зменшити можливе перекручування, необхідно перерахувати розмір символів для другого шрифту з урахуванням зміни аспекту. Але для цього оглядач повинен знати аспект першого шрифту, що на його комп'ютері саме й відсутній! Саме для того, щоб повідомити йому цю інформацію, і була уведена дана властивість, що може мати наступні значення:

`none`

Не коректувати розмір шрифту.

<число>

Задає аспект шрифту, як описано вище.

Розглянемо наступний приклад:

```
P {font-family: Verdana, sans-serif; font-size: 14px; font-size-adjust: 0.58 }
```

Шрифт Verdana має аспект 0.58, що й зазначено в значенні властивості **font-size-adjust**. Якщо цей шрифт недоступний, то оглядач використає родовий шрифт *sans-serif*. Допустимо, що аспект цього шрифту

дорівнює 0.46. Тоді оглядач перерахує розмір шрифту *sans-serif* по формулі $14\text{px} * (0.58 / 0.46) = 17.65\text{px}$ і буде відображати абзаци шрифтом з розміром 18px (з урахуванням округлення).

2.8.2.9. Властивості шрифту: властивість **font**

Синтаксис: `font: [[<font-style>|<font-variant>|<font-weight>]?<font-size>`

`[/<line-height>]?<font-family>]|caption|icon|menu|message-box|small-caption|status-bar|inherit`

Начально: не визначено для сокрашений

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: застосовні тільки в `<font-size>` і `<line-height>`

Пристрій: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **font** є скороченням для властивостей [font-style](#), [font-variant](#), [font-weight](#), [font-size](#), [font-family](#) і [line-height](#). Воно дозволяє задати всі властивості шрифту одночасно. При цьому спочатку всім властивостям шрифту (включаючи **font-stretch** і **font-size-adjust**, хоча значення цих властивостей даною властивістю задані бути не можуть) привласнюються їхні початкові значення, а потім змінюються значення тих властивостей, які явно задані в даній властивості. Приведемо кілька прикладів:

```
P { font: 12pt/14pt sans-serif }
P { font: 80% sans-serif }
P { font: x-large/110% "new century schoolbook", serif }
P { font: bold italic large Georgia, serif; font-size-adjust: 1.16 }
P { font: normal small-caps 120%/120% fantasy }
P { font: oblique 12pt Helvetica, serif; font-stretch: condensed }
```

Ця властивість дозволяє також указати, що необхідно використати один із системних шрифтів; у цьому випадку задається тільки назва системного шрифту, і ніякі інші характеристики шрифту вказувати більше не можна. Існують наступні назви системних шрифтів:

<i>caption</i>	Шрифт тексту керуючих елементів (кнопок, міток і т.д.)
<i>icon</i>	Шрифт міток іконок
<i>menu</i>	Шрифт пунктів меню
<i>message-box</i>	Шрифт діалогових вікон
<i>small-caption</i>	Шрифт тексту малих керуючих елементів
<i>status-bar</i>	Шрифт статусного рядка

Приклад відображення системних шрифтів:

caption
icon
menu
message-box
small-caption
status-bar

2.8.3. Підбор і завантаження шрифтів

2.8.3.1. Підбір шрифту

У цьому розділі повністю описаний механізм CSS по роботі зі шрифтами, заснований на їхньому підборі, синтезі на основі бази даних або завантаженні з Веб-узлов. Існують чотири різновиди реалізації цього механізму:

Підбор шрифту по його імені

У цьому випадку оглядач використовує шрифт, установлений на комп'ютері-клієнті й имеючий те ж ім'я, що й запитаний шрифт (при цьому й гарнітура, і метричні характеристики шрифту можуть виявитися відмінними від авторських, якщо автор і користувач завантажували шрифти з різних джерел). Таким чином, єдиною основою для підбора шрифту є його ім'я. Це єдиний метод, що використовується в CSS1 й який описаний у попередньому розділі.

Підбор шрифту по його характеристиках

У цьому випадку оглядач використовує шрифт, установлений на комп'ютері-клієнті й зовнішній вид, що має той же, що й запитаний шрифт (при цьому метричні характеристики шрифту можуть виявитися відмінними від авторських). Основою для підбора шрифту в цьому випадку служать такі його характеристики, як тип шрифту (текстова або символний), наявність або відсутність зарубок, жирність, висота рядкових і прописних букв, нахил символів і т.п.

Синтез шрифту

У цьому випадку оглядач створює шрифт, що не тільки має той же зовнішній вигляд, що й запитаний шрифт, але й відповідний йому по метричних характеристиках. Для такого синтезу шрифту необхідна більше точна інформація, чим у схемах підбора шрифту. Зокрема, оглядачеві необхідно знати точні ширини всіх символів і правила відображення кодів Unicode у відповідні зображення.

Завантаження шрифту

Оглядач може, нарешті, завантажити шуканий шрифт із заданого Веб-узла аналогічно тому, як при відображенні документа завантажуються графічні образи, Java-аплети й інші об'єкти. Це, однак, може приводити до істотної затримки при завантаженні документа.

Із чотирьох перерахованих схем у сучасних оглядачах реалізовані дві: перша й остання. Перша схема була описана в попередньому розділі, а остання, пов'язана із завантаженням шрифтів з Мережі, описується тут.

2.8.3.2. Директива @font-face

Директива **@font-face** дозволяє нам включати у свою таблицю стилів *опису шрифтів*. Вона складається з набору *дескрипторів шрифту* і їхніх значень і має такий вигляд:

```
@font-face {  
  дескриптор: значення;  
  ...  
  дескриптор: значення;  
}
```

Дескриптори шрифтів можна розбити на три групи:

1. дескриптори, що забезпечують зв'язок між описом шрифту і його використанням у властивостях CSS (вони мають ті ж назви, що й відповідні властивості);
2. дескриптор, що задає URI, на якому перебувають дані шрифту для завантаження;
3. дескриптори, що задають подальші характеристики шрифту і які забезпечують зв'язок опису шрифту з його даними.

Приклад директиви **@font-face**:

```
@font-face {  
  font-family: comic; // ім'я шрифту для CSS  
  src: url(http://www.fonts.com/comic.eot); // адреса файлу для  
  завантаження  
}
```

Підтримка: Підтримуються тільки дескриптори `font-family` й `src` (5.0+)

Не підтримується

2.8.3.3. Дескриптори властивостей шрифту

Ця група дескрипторів має ті назви, що й відповідні властивості CSS і можуть мати одне значення або кілька значень, розділених комами. Якщо дескриптор не зазначений у директиві **@font-face**, те приймається його початкове значення.

Синтаксис: `font-family: [<имя-семейства>|<родовое-имя>]`
`[, [<имя-семейства>|<родовое-имя>]]*`
Начально: залежить від оглядача
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

Цей дескриптор задає *список імен сімейств шрифтів* і приймає ті ж значення, що й властивість **font-family**.

Синтаксис: `font-style: all | [normal | italic | oblique][, [normal | italic | oblique]]*`
Начально: `all`
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *стиль шрифту* й може приймати ті ж значення, що й властивість **font-style**. Єдина відмінність полягає в тому, що його значенням може бути список стилів або значення *all* (всі стилі).

Синтаксис: `font-variant: [normal | small-caps][, [normal | small-caps]]*`
Начально: `normal`
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *варіант шрифту* й може приймати ті ж значення, що й властивість **font-variant**. Єдина відмінність полягає в тому, що його значенням може бути список варіантів.

Синтаксис: `font-weight: all | [normal | bold | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900][, [normal | bold | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900]]*`
Начально: `all`
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *жирність шрифту* й може приймати ті ж значення, що й властивість **font-weight**, з наступними відмінностями:

- відносні значення (*bolder* й *lighter*) не допускаються;
- значенням може бути список жирностей;
- значенням може бути *all* (усе жирності).

Синтаксис: `font-stretch: all | [normal | ultra-condensed | extra-condensed | condensed | semi-condensed | semi-expanded | expanded | extra-expanded | ultra-expanded][, [normal | ultra-condensed | extra-condensed | condensed | semi-condensed | semi-expanded | expanded | extra-expanded | ultra-expanded]]*`

Начально: `normal`

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *выключку шрифту* й може приймати ті ж значення, що й властивість **font-stretch**, з наступними відмінностями:

- відносні значення (*wider* й *narrower*) не допускаються;
- значенням може бути список выключек;
- значенням може бути *all* (всі выключки).

Синтаксис: `font-size: all | <размер> | [, <размер>]*`

Начально: `all`

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *розмір шрифту* й може приймати ті ж значення, що й властивість **font-size**, з наступними відмінностями:

- допускаються тільки абсолютні розміри шрифтів;
- значенням може бути список розмірів;
- значенням може бути *all* (всі розміри).

2.8.3.4. Дескриптор місця розташування шрифту

Синтаксис: `src: [<uri> [format(<строка> [, <строка>]*)] | <спецификация-шрифта>]`

`[, <uri> [format(<строка> [, <строка>]*)] | <спецификация-шрифта>]*`

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Підтримується у вигляді `src: url(URI)`, де *URI* указує на шрифт

формату Embedded OpenType (5.0+)

Не підтримується

Цей дескриптор задає *місце розташування шрифту*. Його значенням є список посилань на файли даних шрифтів. Кожна з посилань являє собою або URI зовнішнього шрифту, що завантажує з Веб-узла, або специфікацію локального шрифту, встановленого на комп'ютері-клієнті. Список містить посилання в порядку їхніх пріоритетів, тобто спочатку оглядач намагається завантажити шрифт із першого посилання, потім, якщо це не вдалося, із другої й т.д.

URI зовнішнього шрифту може супроводжуватися підказкою *format(...)*, що містить інформацію про формат файлу шрифту, для того, щоб оглядач міг ігнорувати ті шрифти, формат яких він не підтримує. Специфікація CSS перераховує наступні типові формати шрифтів:

Таблиця 2.6. Формати шрифтів, що завантажують		
Рядок	Формат шрифту	Розширення файлу
"embedded-opentype"	Embedded OpenType	.eot, .ote
"intellifont"	Intellifont	
"openType"	OpenType , включаючи TrueType Open	.ttf
"speedo"	Speedo	
"truedoc-pfr"	TrueDoc™ Portable Font Resource	.pfr
"truetype"	TrueType	.ttf
"truetype-gx"	TrueType с розширеннями GX	
"type-1"	PostScript™ Type 1	.pfb, .pfa

Приклади посилань на завантажують шрифти, що:

```
src: url(http://www.fonts.com/comic.eot);
src: url("http://site/magda-extra.pfr") format("truedoc-pfr");
```

Посилання на локальний шрифт задаються *специфікацією шрифту* виду *local(ім'я)*, де ім'я - це повна назва шрифту в операційній системі (наприклад, в Windows ім'я повинне збігатися з назвою шрифту в системному реєстрі). Приклад посилання на локальний шрифт:

```
src: local("BT Century 751 No. 2 Semi Bold Italic");
```

Приклад повного синтаксису:

```
src: local("T-26 Typeka Mix"), url("http://cgi-bin/bar?stuff")
    format("type-1", "opentype");
```

Останній приклад містить два посилання. Спочатку оглядач спробує завантажити локальний шрифт "T-26 Турека Міх", а потім, якщо він не знайдений, буде завантажувати шрифт із вузла "http://cgi-bin/bar?stuff". У цьому випадку це сценарій сервера, що може генерувати шуканий шрифт у двох форматах: Type 1 й OpenType.

2.8.3.5. Дескриптор діапазону символів

Синтаксис: `unicode-range: [<діапазон>] [, <діапазон>] *`

Начально: U+0-7FFFFFFF

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *діапазон символів шрифту*. Його значенням є список діапазонів символів Unicode, які є в даному шрифті. (Див. [перечень стандартных подмножеств Unicode](#).)

<Діапазон> задається шестнадцатеричним числом, що відповідає коду символу в Unicode, із префіксом "U+". При цьому знак питання "?" замість шестнадцатеричної цифри означає, що вона приймає будь-яке значення (0 - F), наприклад:

```
unicode-range: U+20A7; // один символ (символ іспанської песети)
unicode-range: U+215?; // діапазон від 2150 до 215F (символи простих дробів)
unicode-range: U+00??; // діапазон від 0000 до 00FF (базова латиниця)
unicode-range: U+4??; // діапазон від 0400 до 04FF (кирилиця)
```

Для завдання діапазону використовується також пара чисел, розділених дефісом "-", наприклад:

```
unicode-range: U+4E00-9FFF; // діапазон від 4E00 до 9FFF (уніфіковані ієрогліфи)
```

У загальному випадку, значення даного дескриптора є списком таких діапазонів, що охоплюють всі символи, що втримуються в даному шрифті, наприклад:

```
unicode-range: U+370-3FF, U+1F??; // сучасні грецькі букви (370-3FF) і
                                     // давньогрецькі політонічні букви
                                     (1F00-1FFF)
unicode-range: U+3000-303F, U+3100-312F, U+32??, U+33??, U+4E00-9FFF,
U+F900-FAFF,
                                     U+FE30-FE4F; // повний китайський шрифт
```

2.8.3.6. Дескриптор координатної системи

Синтаксис: units-per-em: [<число>](#)

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *координатну систему метрики шрифту*. Його значенням є кількість координатних одиниць осторонь квадрата, що містить найбільший символ шрифту (звичайно це буква 'M', звідси назва *em-квадрат*). Це значення звичайно визначається форматом шрифту. Типові приклади його значень: 250 (Intellifont), 1000 (Type 1) і 2048 (TrueType, TrueType GX й OpenType).

2.8.3.7. Дескриптори для підбора шрифтів

Ця група необов'язкових дескрипторів призначена для підбора шрифту по його характеристиках.

Синтаксис: panose-1: [<целое>](#) {10}

Начально: 0 0 0 0 0 0 0 0 0 0

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Це дескриптор *чисел Panose-1*. Його значенням є десять цілих чисел, розділених пробілами. Panose-1 — це стандартний механізм класифікації й підбора шрифтів TrueType, придатний для всіх шрифтів, що мають рядкові й прописні букви (латиниця, кирилиця, грецьке лист), але не дозволяє класифікувати інші системи листа (наприклад, однорегистрові писемності, такі, як вірменське, арабське або івритське лист, і

ієрогліфічні, такі, як китайське, корейське або японський лист). Подобиці див. в

<http://www.fonts.com/hp/panose/greybook> .

Синтаксис: stemv: <число>

Начально: не визначено

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *ширину вертикального штриха* (vertical stem width) символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: stemh: <число>

Начально: не визначено

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *ширину горизонтального штриха* (horizontal stem width) символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: slope: <число>

Начально: 0

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *кут нахилу* (vertical stroke angle) у градусах символів шрифту від вертикалі проти вартоний стрілки (тому якщо символи нахилені вправо, те він негативний).

Синтаксис: cap-height: <число>

Начально: не визначено

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *висоту прописних букв* шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: x-height: <число>

Начально: не визначено

Пристрою: визуальные

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *висоту малих літер* шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: ascent: [<число>](#)

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *максимальну висоту символів* шрифту без обліку діакритичних знаків (maximum unaccented height). Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: descent: [<число>](#)

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *максимальну глибину символів* шрифту без обліку діакритичних знаків (maximum unaccented depth). Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

2.8.3.8. Дескриптори для синтезу шрифтів

Ця група необов'язкових дескрипторів призначена для синтезу шрифту й задає його додаткові метричні характеристики.

Синтаксис: widths: [[<діапазон>](#)]? [[<число>](#)] + [, [[<діапазон>](#)]? [[<число>](#)] +]

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *ширини символів шрифту*. Його значенням є список діапазонів символів, за кожним з яких треба одне або більше значення ширини. Якщо цей дескриптор присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**. Якщо діапазон опущений, то він передбачається рівним U+0-7FFFFFFF, тобто поширюється на всі символи шрифту. Якщо кількість заданих ширин менше, ніж символів у діапазоні, то остання із ширин застосовується до всіх символів, чия ширина не зазначена; якщо ж ширин більше, ніж символів, те зайві ширини ігноруються. Пояснимо сказане на прикладах:

widths: U+4E00-4E1F 1736 1874 1692;

widths: U+1A?? 1490, U+215? 1473 1838 1927 1684 1356 1792
1815 1848 1870 1492 1715 1745 1584 1992 1978 1770;

У першому прикладі зазначений діапазон з 32 символів, від 4E00 до 4E1F. Перший символ (з кодом 4E00) буде мати ширину 1736, другий символ — ширину 1874, третій — ширину 1692. Оскільки ширин задано менше, ніж символів, остання ширина (1692) буде привласнена всім іншим символам. У другому прикладі задається єдина ширина для 256 символів у діапазоні від 1A00 до 1AFF, а потім явно вказуються ширини ще для 16 символів.

Синтаксис: bbox: [<число>](#), [<число>](#), [<число>](#), [<число>](#)

Начально: не визначено
Пристрою: [визуальные](#)

Підтримка: Не підтримується
Не підтримується

Цей дескриптор визначає *максимальний об'ємлюючий прямокутник* (maximal bounding box) шрифту. Його значенням є список із чотирьох чисел, які задають відповідно абсцису лівого нижнього кута, ординату лівого нижнього кута, абсцису верхнього правого кута й ординату верхнього правого кута об'ємлюючого прямокутника для всіх символів шрифту.

Синтаксис: definition-src: [<uri>](#)
Начально: не визначено
Пристрою: [визуальные](#)

Підтримка: Не підтримується
Не підтримується

Цей дескриптор указує, що інші дескриптори шрифту втримуються не в таблиці стилів, а в окремому *ресурсі опису шрифту*, що зберігається в зазначеному URI. Його використання дозволяє заощадити обсяг даних, переданих по мережі, якщо багато таблиць стилів звертаються до тим самим шрифтів.

2.8.3.9. Дескриптори для вирівнювання шрифтів

Ця група необов'язкових дескрипторів призначена для вирівнювання шрифту щодо інших шрифтів.

Синтаксис: baseline: [<число>](#)
Начально: 0
Пристрою: [визуальные](#)

Підтримка: Не підтримується
Не підтримується

Цей дескриптор задає *нижню базову лінію* символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: centerline: [<число>](#)
Начально: не визначено
Пристрою: [визуальные](#)

Підтримка: Не підтримується
Не підтримується

Цей дескриптор задає *центральну базову лінію* символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: mathline: [<число>](#)
Начально: не визначено
Пристрою: [визуальные](#)

Підтримка: Не підтримується
Не підтримується

Цей дескриптор задає *математичну базову лінію* символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Синтаксис: `topline: <число>`

Начально: не визначено

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Цей дескриптор задає *верхню базову лінію* символів шрифту. Якщо він присутній в описі шрифту, то повинен бути присутнім і дескриптор **units-per-em**.

Глава 2.9. Текст

У цій главі наведений опис властивостей, що визначають різні параметри відображення тексту.

2.9.1. Новий рядок: властивість **text-indent**

Синтаксис: `text-indent: <размер> | <процент> | inherit`

Начально: 0

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: щодо ширини блоку, що вміщає

Устройства: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту (4.0+)

Властивість **text-indent** задає *відступ першого рядка* при відображенні блокових елементів. Відступ може бути негативним і задається одним з наступних способів:

[<размер>](#)

Задає фіксований розмір.

[<процент>](#)

Обчислюється щодо ширини блоку, що вміщає.

Приклад: наступне правило вказує, що абзаци повинні починатися з нового рядка, рівної 3em:

```
P { text-indent: 3em }
```

2.9.2. Вирівнювання тексту: властивість **text-align**

Синтаксис: `text-align: left | right | center | justify | <строка> | inherit`

Начально: залежить від оглядача й напрямку виводу тексту

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Підтримуються тільки left, right, center й justify (4.0+)

Підтримуються тільки left, right й center (4.0+)

Властивість **text-align** задає *вирівнювання тексту* при відображенні блокових елементів. Воно може приймати наступні значення:

<i>left</i>	Вирівнювання по лівому краї .
<i>right</i>	Вирівнювання по правому краї .
<i>center</i>	Вирівнювання по центрі .
<i>justify</i>	Вирівнювання по лівому краї з выключкой по правому краї .
<u><строка></u>	Задає рядок, по якій будуть вирівнюватися осередку в стовпці таблиці. Це значення застосовне тільки до осередків таблиці. Для інших елементів повинне трактуватися як <i>left</i> або <i>right</i> залежно від значення властивості direction (<i>ltr</i> або <i>rtl</i> відповідно).

Приклад: наступне правило вказує, що всі елементи, що містяться в елементі **DIV** з атрибутом *class="center"*, повинні вирівнюватися по центрі вікна оглядача (це забезпечується тим, що дана властивість є наслідуванням):

```
DIV.center { text-align: center }
```

2.9.3. Прикраса тексту: властивість text-decoration

Синтаксис: `text-decoration: none | [underline || overline || line-through || blink] | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту; `blink` не підтримується (4.0+)

Відповідає стандарту; `overline` не підтримується (4.0+)

Властивість **text-decoration** задає *прикраса тексту* при відображенні елементів. Воно може приймати наступні значення:

<i>none</i>	Звичайний текст.	приклад
<i>underline</i>	Підкреслений текст.	<u>приклад</u>
<i>overline</i>	Надчеркнутый текст.	приклад
<i>line-through</i>	Перекреслений текст.	приклад
<i>blink</i>	Миготливий текст.	приклад

Ця властивість не наслідувана, але всі нащадки блокового елемента будуть мати його прикраси тексту. Кольори цих прикрас буде тим же, що в блокового елемента, навіть якщо нащадки мають інше значення властивості **color**.

Приклад: наступне правило вказує, що гіперпосилання повинні підкреслюватися:

```
A:link { text-decoration: underline }
```

2.9.4. Перетворення тексту: властивість text-transform

Синтаксис: `text-transform: none | capitalize | uppercase | lowercase | inherit`

Начально: `none`
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Відповідає стандарту; реалізовано з помилками (4.0+)

Властивість **text-transform** задає *перетворення тексту* при відображенні елементів. Воно може приймати наступні значення (Netscape Navigator 4.x відображає приклади невірно!):

<code>none</code>	Без перетворення.	Це приклад.
<code>capitalize</code>	Робити першу букву кожного слова прописний .	Це приклад.
<code>uppercase</code>	Виводити текст прописними буквами .	ЦЕ ПРИКЛАД.
<code>lowercase</code>	Виводити текст малими літерами.	Це приклад.

Фактичне перетворення символів залежить від мови, на якому написаний текст. Приклад: наступне правило вказує, що заголовок **H1** повинен виводитися прописними буквами:

```
H1 { text-transform: uppercase }
```

2.9.5. Завдання тіні: властивість text-shadow

Синтаксис: `text-shadow: none | [<цвет>|<размер><размер><размер>?,] * [<цвет>|<размер><размер><размер>?] | inherit`
Начально: `none`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **text-shadow** задає *тінь тексту* при відображенні елементів. Його значенням є список ефектів, розділених комами. Ці ефекти повинні застосовуватися до тексту в тім порядку, як вони задані, і можуть перекриватися при відображенні один з одним (але не із самим текстом!).

Кожен ефект задається обов'язковим зсувом від тексту й необов'язковими кольорами тіні й радіусом плями. Зсув тіні щодо тексту задається двома значеннями типу [<размер>](#), перше з яких указує горизонтальний зсув тіні вправо від тексту, а друге — вертикальний зсув тіні вниз від тексту. Ці значення можуть бути негативними, що означає зсув тіні вліво або нагору щодо тексту відповідно.

Після зсуву може бути зазначений необов'язковий радіус плями. Точний алгоритм обчислення плями не заданий і залежить від оглядача.

Крім того, перед зазначеними розмірами або після них може бути заданий необов'язковий колір тіні. Якщо він не заданий, то використається значення властивості **color**.

Тіні можуть застосовуватися до псевдоелементам [:first-letter](#) і [:first-line](#).

Приклади:

```
H1 { text-shadow: 0.2em 0.3em }
H2 { text-shadow: 3px 3px 5px red }
H3 { text-shadow: 3px 3px red, yellow -3px 3px 2px, 3px -3px }
```

У першому прикладі задана тінь зі зсувом 0.2em правее тексту й 0.3em нижче його. У другому прикладі задані зсуви по 3px правее й нижче тексти, радіус плями 5px і червоний кольори тіні. Останній приклад містить список із трьох ефектів.

2.9.6. Інтервал між буквами: властивість letter-spacing

Синтаксис: `letter-spacing: normal | <размер> | inherit`
Начально: `normal`
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **letter-spacing** задає *інтервал між буквами* при відображенні тексту. Його значення задається одним з наступних способів:

`normal`

Стандартний інтервал для поточного шрифту.

`<размер>`

Задає інтервал на додаток до стандартного інтервалу. Це значення може бути негативним.

Приклад: наступне правило збільшує інтервал між символами в заголовку **H1** на 0.5em:

```
H1 { letter-spacing: 0.5em }
```

2.9.7. Інтервал між словами: властивість word-spacing

Синтаксис: `word-spacing: normal | <размер> | inherit`
Начально: `normal`
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **word-spacing** задає *інтервал між словами* при відображенні тексту. Його значення задається одним з наступних способів:

`normal`

Стандартний інтервал для поточного шрифту.

`<размер>`

Задає інтервал на додаток до стандартного інтервалу. Це значення може бути негативним.

Алгоритм вибору інтервалів між словами залежить від оглядача й від наявності виключки тексту по правому краю (див. властивість **text-align**).

Приклад: наступне правило збільшує інтервал між словами в заголовку **H1** на 1em:

```
H1 { word-spacing: 1em }
```

2.9.8. Обробка пробілів: властивість white-space

Синтаксис: `white-space: normal | pre | nowrap | inherit`

Начально: `normal`

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Підтримуються тільки значення `normal` й `pre` (4.0+)

Властивість **white-space** задає правила *обробки пробілів* при відображенні блокового елемента. Воно може приймати наступні значення:

<i>normal</i>	Пробіли й розриви рядків відображаються обычным образом .
<i>pre</i>	Пробіли й розриви рядків відображаються як у форматированном тексте .
<i>nowrap</i>	Пробіли відображаються обычным образом , але розриви рядків заборонені.

Приклад: наступні правила відповідають правилам HTML по відображенню елементів **P** й **PRE**:

```
P {white-space: normal }  
PRE { white-space: pre }
```

Глава 2.10. Таблиці

2.10.1. Загальний опис

Таблиці забезпечують можливість розташування багатомірних даних по рядках і стовпцях. Структура й зміст таблиці описуються мовою документа, а CSS дозволяє описати правила її відображення — або візуального, або звукового.

Візуальне відображення таблиць складається в завданні правил відображення заголовків й осередків таблиці, їхнього вирівнювання відносно один одного, зображення рамок навколо них і т.д. При звуковому відображенні таблиці задаються правила того, як повинні вимовлятися заголовки й уміст осередків.

Будова таблиць в CSS засновано на [строении таблиц в HTML](#) і успадковує всі їхні складові. Кожна таблиця може мати *заголовок*, що містить її короткий опис. Крім того, вона може мати *наздаголовок* і *підзаголовок*. Тіло таблиці складається з *рядків*, у свою чергу, що складаються з *осередків*. *Стовпці* таблиць явно не описуються, а визначаються будовою рядків: перший осередок кожного рядка ставиться до першого стовпця, друга до другого й т.д. Рядки й стовпці можуть поєднуватися в *групи*, які можуть мати свої особливості відображення.

CSS не вимагає, щоб мова документа містив у собі елементи для кожної з перерахованих складових таблиць. Наприклад, XML-додатка не містять визначених табличних структур. Проте, автори можуть відобразити елементи на таблиці за допомогою властивості **display**. Для присвоєння табличної семантики довільним елементам мови документа використовуються наступні значення цієї властивості:

Значення	Елемент HTML	Опис
----------	--------------	------

<i>table</i>	TABLE	Блокова таблиця. Елемент повинен відображатися як блоковий .
<i>inline-table</i>	TABLE	Текстова таблиця. Елемент повинен відображатися як текстовий .
<i>table-row</i>	TR	Елемент повинен відображатися як рядок таблиці.
<i>table-row-group</i>	TBODY	Елемент описує групу рядків таблиці.
<i>table-header-group</i>	THEAD	Елемент повинен відображатися як група надзаголовків таблиці. Відображається перед всіма іншими групами рядків таблиці. При виводі на принтер може використатися як надзаголовок при печатці кожної сторінки таблиці.
<i>table-footer-group</i>	TFOOT	Елемент повинен відображатися як група підзаголовків таблиці. Відображається після всіх інших груп рядків таблиці. При виводі на принтер може використатися як підзаголовок при печатці кожної сторінки таблиці.
<i>table-column-group</i>	COLGROUP	Елемент описує групу стовпців таблиці. Не відображається, але може впливати на відображення стовпців даної групи.
<i>table-column</i>	COL	Елемент описує стовпець таблиці. Не відображається, але може впливати на відображення осередків даного стовпця .
<i>table-cell</i>	TD, TH	Елемент повинен відображатися як осередок таблиці.
<i>table-caption</i>	CAPTION	Елемент повинен відображатися як заголовок таблиці.

[Таблиця стилей для HTML](#) за замовчуванням задає наступні властивості елементів:

```
TABLE    { display: table }
TR       { display: table-row }
THEAD    { display: table-header-group }
TBODY    { display: table-row-group }
TFOOT    { display: table-footer-group }
COL       { display: table-column }
COLGROUP { display: table-column-group }
TD, TH   { display: table-cell }
CAPTION  { display: table-caption }
```

Підтримка: Підтримуються тільки `table-header-group` й `table-footer-group` (5.0+)

Не підтримуються

2.10.2. Селектори стовпців

Осередку таблиць належать одночасно до двох контекстів: рядкам і стовпцям. При цьому в тексті документа осередки є нащадками рядків, але не стовпців. Проте, завдання певних властивостей стовпців або їхніх груп впливає на відображення осередків. Такими властивостями є:

border

Властивості рамок застосовуються до стовпців тільки тоді, коли таблиця має [слившіися рамки](#). У цьому випадку оглядач вибирає рамку для кожної границі осередку з урахуванням рамок стовпців, як описано в [п. 2.10.4.1](#).

background

Властивості тла стовпця застосовуються до його осередків тільки в тому випадку, коли й рядок, і сам осередок мають прозоре тло.

width

Ця властивість задає мінімальну ширину стовпця.

visibility

Якщо ця властивість для стовпця має значення *collapse*, то осередку цього стовпця не відображаються, а ті осередки, які поширюються на кілька стовпців, обрізаються. Крім того, ширина таблиці зменшується на ширину даного стовпця. Подробиці див. нижче в описі [динамических эффектов](#). Інші значення цієї властивості ігноруються.

Приведемо два приклади:

```
TABLE { border-style: hidden }
COL { border-style: none solid }

TABLE { table-layout: fixed }
COL.total { width: 5em }
```

У першому прикладі показано, як реалізувати мовою CSS значення *cols* атрибута **rules** таблиць HTML (виділення рамкою тільки границь стовпців). Другий приклад показує, як задати фіксовану ширину стовпця.

2.10.3. Візуальне відображення таблиць

2.10.3.1. Розташування заголовка таблиці: властивість **caption-side**

Синтаксис: `caption-side: top | bottom | left | right | inherit`
Начально: `top`
Застосовно: до елементів типу `table-caption`
Наслідувано: да
Відсотки: не використовуються
Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **caption-side** визначає розташування заголовка таблиці щодо тіла таблиці при її відображенні. Воно може приймати наступні значення:

<i>top</i>	Заголовок відображається над тілом таблиці.
<i>bottom</i>	Заголовок відображається під тілом таблиці.
<i>left</i>	Заголовок відображається ліворуч від тіла таблиці.
<i>right</i>	Заголовок відображається праворуч від тіла таблиці.

Відображення заголовка над або під тілом таблиці виробляється так само, як відображення двох послідовно розташованих блокових елементів. При відображенні заголовка ліворуч або праворуч від тіла таблиці варто явно задавати його ширину властивістю **width**, т. к. не ясно, що в цьому випадку означає значення *auto*, прийняте за замовчуванням. Горизонтальне вирівнювання заголовка задається властивістю **text-align**, а вертикальне — властивістю **vertical-align**, яке повинне мати значення *top*, *middle* або *bottom*.

У наступному прикладі заголовок таблиці розміщується ліворуч від її, сама таблиця центрується (завданням лівій і правій границям значення *auto*), і весь прямокутник, об'ємлющий заголовок і тіло таблиці, зрушується вліво на ширину заголовка.

```
BODY {
  margin-left: 10em
}
TABLE {
  margin-left: auto;
  margin-right: auto
}
```

```
CAPTION {
  caption-side: left;
  margin-left: -10em;
  width: 10em;
  text-align: right;
  vertical-align: bottom
}
```

2.10.3.2. Завдання ширини стовпців: властивість **table-layout**

Синтаксис: `table-layout: auto | fixed | inherit`

Начально: `auto`

Застосовно: до елементів типу `table` й `inline-table`

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

Властивість **table-layout** визначає спосіб обчислення ширини стовпців таблиці при її відображенні. Воно може приймати наступні значення:

<i>auto</i>	Автоматичне обчислення ширини стовпців .
<i>fixed</i>	Фіксована ширина стовпців .

Завдання *фіксованої ширини стовпців* означає наступне. Оглядач аналізує перший рядок таблиці, і на основі цього аналізу обчислює ширини всіх стовпців (наступні рядки не враховуються, тому цей алгоритм забезпечує більше швидке відображення таблиці, чим автоматичний розрахунок ширин стовпців). У цьому випадку ширина таблиці повинна бути задана явно властивістю **width**, інакше виробляється автоматичний розрахунок ширин. Обчислення фіксованої ширини стовпців виробляється так:

1. Якщо для стовпця задана властивість **width** і його значення відмінно від *auto*, те воно задає ширину стовпця.
2. У протилежному випадку, якщо для осередку в першому рядку таблиці задана властивість **width** і його значення відмінно від *auto*, те воно задає ширину стовпця.
3. Між стовпцями, що залишилися, нарівно ділиться простір, що залишилося, таблиці по горизонталі.

Якщо при відображенні наступних рядків таблиці виявиться, що вміст якого-небудь осередку не міститься в стовпець, то воно може бути обрізано. Обрезкой вмісту осередків управляє властивість **overflow**.

Автоматичне обчислення ширини стовпців займає більше часу, тому що вимагає перегляду всієї таблиці, але дає більше акуратні результати. Воно засновано на тім, що для кожного стовпця проглядаються всі його осередки, і ширина стовпця визначається шириною найбільшої частини осередку, що не може бути розірвана переходом на новий рядок. Якщо при цьому для стовпця задана властивість **width** і його значення відмінно від *auto*, те воно задає мінімальну ширину стовпця. Завдання цього значення у відсотках означає відсоток від ширини всієї таблиці.

У наступному прикладі ширина стовпців таблиці фіксована й дорівнює 100px, 300px й 200px відповідно:

```
<TABLE style="table-layout:fixed; width: 600px">
<COL style="width: 100px"><COL style="width: 300px"><COL style="width:
200px">
...
</TABLE>
```


2.10.3.3. Динамічні ефекти при відображенні таблиць

Властивість [visibility](#) може приймати значення *collapse* для рядків, груп рядків, стовпців і груп стовпців. Це значення приводить до того, що рядок або стовець таблиці повністю зникають із екрана, а простір, що вони займали, стає доступним для відображення іншого змісту. Однак, придушення відображення строк або стовпців не викликає перерахунку висот і ширин осередків таблиці. Завдяки цьому стає можливим динамічно видаляти рядки й стовпці таблиці, не змінюючи в іншому її зовнішнього вигляду.

2.10.4. Відображення рамок таблиць

2.10.4.1. Завдання типу рамок: властивість border-collapse

Синтаксис: `border-collapse: collapse | separate | inherit`

Начально: `collapse`

Застосовно: до елементів типу `table` й `inline-table`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (5.0+)

Не підтримується

Властивість **border-collapse** задає *тип рамок таблиці*. CSS підтримує для таблиць два типи рамок: *рамки, що злилися (collapse)* і *роздільні рамки (separate)*. Роздільні рамки відповідають стандарту HTML; рамки, що злилися, відрізняються тим, що сусідні рамки зливаються в єдину рамку. Розходження між ними наочно демонструється наступним прикладом (Netscape Navigator відображає його невірно!):

рамки, що злилися

Осередок	Осередок	Осередок
Осередок	Осередок	Осередок
Осередок	Осередок	Осередок

Роздільні рамки

Осередок	Осередок	Осередок
Осередок	Осередок	Осередок
Осередок	Осередок	Осередок

Тип рамки таблиці визначає й те, які властивості рамки застосовні до даної таблиці, і те, як саме відображається той або інший стиль рамки.

У таблиці з роздільними рамками кожен осередок має власну рамку, тому завдання рамок для рядків, стовпців, груп рядків і груп стовпців ігнорується оглядачем. До таблиці з роздільними рамками застосовні властивості **border-spacing** і **empty-cells**; для таблиць із рамками, що злилися, вони ігноруються.

У таблиці зі слившимися рамками можна задавати рамки як для осередків, так і для рядків, стовпців, груп рядків і груп стовпців. Якщо рамки окремих складових суперечать один одному, то використовується наступний алгоритм дозволу конфліктів:

1. Рамки зі значенням **border-style**, рівним *hidden*, мають пріоритет над іншими конфліктуючими рамками. Будь-яка рамка цього стилю придушує всі рамки в даному місці таблиці.
2. Рамки зі стилем *none* мають найменший пріоритет. Рамка не буде відображатися тільки тоді, коли всі елементи, розташовані в даному місці, мають цей стиль рамки.
3. В інших випадках більше широкі рамки мають пріоритет перед більше вузькими. Якщо ж кілька рамок мають одне значення **border-width**, те стилі рамок у порядку убуння пріоритетів такі: *double, solid, dashed, dotted, ridge, outset, groove, inset*.
4. Якщо рамки відрізняються тільки квітами, то порядок убуння пріоритетів такий: рамка осередку, рамка рядка, рамка групи рядків, рамка стовпця, рамка групи стовпців, рамка таблиці.

Відображення деяких [стилей рамок](#) для таблиць також залежить від типу рамки й трохи відмінно від звичайного, а саме:

hidden

Схована рамка. Те ж, що *none*, але для рамок, що злилися, придушує всі інші рамки.

inset

Для роздільних рамок зображується у вигляді тривимірного урізання; для рамок, що злилися, збігається з *groove*.

outset

Для роздільних рамок зображується у вигляді тривимірної вирізки; для рамок, що злилися, збігається з *ridge*.

2.10.4.2. Відстань між рамками: властивість **border-spacing**

Синтаксис: `border-spacing: <размер> <размер> ? | inherit`

Начально: `0`

Застосовно: до елементів типу `table` й `inline-table`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **border-spacing** задає відстань між сусідніми рамками таблиці в тому випадку, коли таблиця має роздільні рамки. Якщо зазначено тільки один розмір, то він задає відстань між рамками й по вертикалі, і по горизонталі. Якщо зазначені два розміри, то перший задає відстань між рамками по горизонталі, а другий — по вертикалі. Відстані не можуть бути негативними. Приклад:

```
TABLE { border: outset 10pt; border-collapse: separate; border-spacing: 15pt }
```

2.10.4.3. Рамки навколо порожніх осередків: властивість **empty-cells**

Синтаксис: `empty-cells: show | hide | inherit`

Начально: `show`

Застосовно: до елементів типу `table-cell`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **empty-cells** управляє відображенням рамок навколо порожніх осередків у тому випадку, коли таблиця має роздільні рамки. Осередок таблиці вважається порожнім, якщо не містить нічого, крім пробілів ("`\20`"), табуляцій ("`\09`"), перекладів рядка ("`\0D`") і повернень каретки ("`\0A`"). Крім того, порожніми вважаються осередки, у яких значення властивості **visibility** дорівнює *hidden*.

Якщо дана властивість має значення *show*, то рамки навколо порожніх осередків відображаються звичайним образом. Якщо воно має значення *hide*, то рамки навколо порожніх осередків не відображаються; більше того, якщо весь рядок таблиці складається з порожніх осередків, то вона поводить себе так, ніби мала властивість **display** зі значенням *none*. Приклад:

```
TABLE { border-collapse: separate; empty-cells: show }
```

2.10.5. Звукове відображення таблиць: властивість **speak-header**

Синтаксис: `speak-header: once | always | inherit`

Начально: `once`

Застосовно: до елементів, що містять заголовки таблиць

Наслідувано: да
Відсотки: не використовуються
Пристрою: [звуків](#)

Підтримка: Не підтримується
Не підтримується

Якщо таблиця вимовляється синтезатором мови, то взаємини між осередками даних й осередками заголовків виражаються інакше, чим при візуальному відображенні таблиці. Властивість **speak-header** дозволяє нам указати, у який момент повинні вимовлятися заголовки. Воно може приймати наступні значення:

once
Заголовок вимовляється один раз, перед серією осередків.
always
Заголовок вимовляється перед кожним осередком, до якої він ставиться.

Спосіб завдання заголовків залежить від мови документа. Зокрема, HTML містить три атрибути елементів **TD** і **TH** (**headers**, **scope** і **axis**), які дозволяють зв'язати з кожним осередком таблиці відповідний заголовок.

Приклад:

```
TH.totals { speak-header: always }
```

Глава 2.11. Генерація вмісту, нумерація й списки

2.11.1. Поняття генерації вмісту

У деяких випадках CSS дозволяє відображати зміст, якого немає в дерева документа. Типовим прикладом є нумерований список: автор документа не нумерує його пункти, а жадає від оглядача, щоб той генерував номери пунктів автоматично. У цій главі зібрані всі можливості CSS по генерації вмісту, а саме:

- генерація вмісту за допомогою псевдоелементів **:before** й **:after**;
- автоматична нумерація з використанням лічильників;
- створення маркерів і відображення списків.

2.11.2. Генерація вмісту

2.11.2.1. Завдання вмісту: властивість **content**

Синтаксис: `content: [<строка> | <uri> | <счетчик> | attr(X) | open-quote | close-quote | no-open-quote | no-close-quote]+ | inherit`
Начально: порожній рядок
Застосовно: до псевдоелементам **:before** й **:after**
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [все](#)

Підтримка: Не підтримується
Не підтримується

Властивість **content** використовується разом із псевдоелементами **:before** й **:after** для генерації вмісту. Воно задає *текст вмісту для вставки* й може приймати наступні значення:

[<строка>](#)
Уставляється даний текст.

[<uri>](#)

Уставляється вміст ресурсу з даним URI.

[<счетчик>](#)

Уставляється значення зазначеного [счетчика](#) або лічильників.

attr(X)

Уставляється значення атрибута X суб'єкта селектора.

open-quote

Уставляються відкриваючі лапки (див. властивість **quotes**).

close-quote

Уставляються закриваючі лапки (див. властивість **quotes**).

no-open-quote

Не уставляється нічого, але лічильник вкладення лапок збільшується на 1.

no-close-quote

Не уставляється нічого, але лічильник вкладення лапок зменшується на 1.

Місце вставки тексту визначається псевдоелементом у селекторі даного елемента — **:before** означає вставку перед елементом, а **:after** після нього. Спосіб вставки (текстовий, блоковий або маркер) задається значенням властивості **display**. Наприклад, що впливає декларація задає вивід тексту "Кінець документа" у центрі останнього рядка екрана.

```
BODY:after {  
  content: "Кінець документа";  
  display: block;  
  margin-top: 2em;  
  text-align: center;  
}
```

2.11.2.2. Лапки: властивість **quotes**

Синтаксис: **quotes:** [[<строка><строка>](#)]+|*none*|[inherit](#)

Начально: залежить від оглядача

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **quotes** задає *символи лапок* і може приймати наступні значення:

none

Значення *open-quote* й *close-quote* властивості **content** не генерують символів лапок.

[[<строка><строка>](#)]+

Задає пари символів лапок для значень *open-quote* й *close-quote*

властивості [content](#).

Кожна пара відповідає черговому рівню вкладення лапок.

Як лапки звичайно використовуються наступні символи Unicode:

Код (16-ний)	Вид	Опис
0022	"	подвійні лапки
0027	'	одинарні лапки (апостроф)
2039	<	відкриваючі одинарні кутові лапки
203A	>	закриваючі одинарні кутові лапки
00AB	«	відкриваючі подвійні кутові лапки

00BB	»	закриваючі подвійні кутові лапки
2018	‘	відкриваючі одинарні лапки
2019	’	закриваючі одинарні лапки
201A	,	закриваючі нижні одинарні лапки
201C	“	відкриваючі подвійні лапки
201D	”	закриваючі подвійні лапки
201E	„	закриваючі нижні подвійні лапки

У наступному прикладі заданий зовнішній вигляд лапок і зазначено, що вони повинні уставлятися на початку й кінці елемента **Q**.

```
Q { quotes: "\201C" "\201D" "\2018" "\2019" }
Q:before ( content: open-quote )
Q:after ( content: close-quote )
```

2.11.3. Автоматична нумерація

2.11.3.1. Лічильники

Для автоматичної нумерації в CSS використовуються [сметчики](#). Лічильник — це ідентифікатор, якому привласнюються цілі значення за допомогою властивостей **counter-reset** і **counter-increment**. Поточне значення лічильника може використатися при генерації вмісту властивістю **content**.

За замовчуванням, лічильник позначається як *counter(name)* і його значенням є десяткові числа. Разом з тим ми можемо задавати стиль відображення значень лічильника у форматі *counter(name,style)*, де *style* може приймати будь-які значення властивості **list-style-type**. Відзначимо, що стиль *none* допустимо: *counter(name,none)* генерує порожній рядок. Приклади:

```
H1:before { content: counter(chapter, upper-latin) ". " }
H2:before { content: counter(section, upper-roman) " - " }
BLOCKQUOTE:after { content: " [" counter(bq, hebrew) "]" }
DIV.note:before { content: counter(notecnt, disc) " " }
P:before { content: counter(pcount, none) }
```

Елементи зі значенням *none* властивості **display** не змінюють значення лічильників. З іншого боку, елементи зі значенням *hidden* властивості **visibility** змінюють їхні значення, як звичайні елементи.

Підтримка: Не підтримуються

Не підтримуються

2.11.3.2. Початкове значення лічильника: властивість counter-reset

Синтаксис: `counter-reset: [<ідентифікатор> <целое> ?]+ | none | inherit`
Начально: *none*
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [все](#)

Підтримка: Не підтримується

Не підтримується

Властивість **counter-reset** задає *початкові значення лічильників*. Його значенням є список імен лічильників, за кожним з яких може впливати необов'язкове ціле число; якщо таке число зазначене, то воно стає значенням лічильника, якщо ні, те лічильнику привласнюється значення 0.

Якщо елемент *i* задає початкове значення лічильника, *i* використовує його значення у властивості [content](#), те значення використовується *після* присвоювання.

Дана властивість треба правилам каскадності. Тому наступні декларації

```
H1 { counter-reset: chapter 1 }
H1 { counter-reset: section -1 }
```

зададуть тільки значення лічильника section. Для завдання початкових значень обох лічильників їх потрібно вказати в одному правилі:

```
H1 { counter-reset: chapter 1 section -1 }
```

2.11.3.3. Зміна значення лічильника: властивість counter-increment

Синтаксис: counter-increment: [<ідентифікатор> [<целое>](#) ?]⁺ | none | [inherit](#)

Начально: none

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [все](#)

Підтримка: Не підтримується

Не підтримується

Властивість **counter-increment** *змінює значення лічильників*. Його значенням є список імен лічильників, за кожним з яких може впливати необов'язкове ціле число; якщо таке число зазначене, то воно додається до поточного значення лічильника, якщо ні, те значення лічильника збільшується на 1. Ціле число може бути й нулем, і негативним.

Якщо використано лічильник, якому не привласнене початкове значення, то воно приймається рівним нулю. Якщо елемент *i* скидає, *i* змінює значення лічильника, то воно спочатку скидається, а потім змінюється. Якщо елемент *i* змінює значення лічильника, *i* використовує його значення у властивості [content](#), те значення використовується *після* зміни значення.

Наступний приклад показує, як нумерувати глави текстом виду "Розділ 1. ", а розділи — текстом виду "1.1 " і т.д.

```
H1:before {
  counter-increment: chapter; /* Збільшити chapter на 1 */
  counter-reset: section; /* Обнулить section */
  content: "Глава " counter(chapter) ". ";
}
H2:before {
  counter-increment: section; /* Збільшити section на 1 */
  content: counter(chapter) "." counter(section) " ";
}
```

2.11.4. Маркери

2.11.4.1. Створення маркерів

Більшість блокових елементів в CSS генерують один об'ємлюючий прямокутник. Однак, CSS містить два механізми, що змушують елемент генерувати два прямокутники: головний прямокутник для вмісту елемента й *прямокутник маркера*, що може розташовуватися як усередині, так і поза головним прямокутником. На відміну від генерируемого вмісту, прямокутник маркера не впливає на позицію

головного прямокутника. Більше загальний із цих механізмів називаються *маркери* й описаний у цьому розділі. Другий, більше обмежений по можливостях, але частіше використовуваний, називаються *списки* й описаний у наступному розділі.

Маркери створюються присвоюванням значення *marker* властивості **display** у псевдоелементах **:before** й **:after**. Прямокутник маркера форматується як однорядковий прямокутник, розташований поза головним прямокутником елемента. Він створюється тільки в тому випадку, коли властивість **content** псевдоелемента дійсно генерує зміст. Прямокутник маркера має рамку й заповнювач, але не має границь.

Для псевдоелемента **:before** базова лінія тексту прямокутника маркера вертикально вирівнюється по базовій лінії тексту першого рядка вмісту головного прямокутника, а для псевдоелемента **:after** — по базовій лінії тексту останнього рядка. Якщо головний прямокутник не містить тексту, то вирівнювання виробляється відповідно по верхньому й нижньому зовнішньому краї головного прямокутника.

Висота прямокутника маркера задається властивістю **line-height**. Вертикальне вирівнювання маркера усередині його прямокутника задається властивістю **vertical-align**.

Якщо властивість **width** має значення *auto*, те ширина прямокутника маркера визначається його вмістом, в інших випадках — значенням **width**. Якщо при цьому ширина вмісту більше ширини прямокутника, то обрізка визначається значенням властивості **overflow**. Прямокутник маркера може накладатися на головний прямокутник. Якщо ширина вмісту менше ширини прямокутника, то горизонтальне вирівнювання вмісту задається властивістю **text-align**.

Властивість **marker-offset** задає зсув прямокутника маркера по горизонталі щодо головного прямокутника, точніше, відстань між найближчими краями їхніх рамок.

Якщо властивість **display** має значення *marker* для вмісту, сгенерованного елементом, у якого значення **display** дорівнює *list-item*, те прямокутник маркера, сгенерований для **:before**, заміщає звичайний маркер елемента списку.

Наприклад, що впливає HTML-документ

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<HTML>
<HEAD>
  <TITLE>Створення списку з маркерами</TITLE>
  <STYLE type="text/css">
    LI:before {
      display: marker;
      counter-increment: mycounter;
      content: counter(mycounter, lower-roman) ". ";
    }
  </STYLE>
</HEAD>
<BODY>
  <OL>
    <LI>Перший елемент списку</LI>
    <LI>Другий елемент списку</LI>
    <LI>Третій елемент списку</LI>
  </OL>
</BODY>
</HTML>
```

буде відображатися так:

- i. Перший елемент списку.
- ii. Другий елемент списку.
- iii. Третій елемент списку.

Підтримка: Не підтримуються

Не підтримуються

2.11.4.2. Позиція маркера: властивість **marker-offset**

Синтаксис: `marker-offset: <размер> | auto | inherit`

Начально: `auto`

Застосовно: до елементів типу `display: marker`

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **marker-offset** задає зсув *прямокутника маркера* по горизонталі щодо головного прямокутника, точніше, відстань між найближчими краями їхніх рамок. Воно може бути задане явно ([<размер>](#)) або залишено на розсуд оглядача (*auto*). Розмір може бути негативним. Приклад:

```
LI:before { marker-offset: 3em }
```

2.11.5. Списки

2.11.5.1. Введення в списки

Елементи списків створюються присвоюванням значення *list-item* властивості [display](#). Як й у більше загальному випадку маркерів, вони відображаються у вигляді двох прямокутників: головного прямокутника для вмісту елемента й необов'язкового прямокутника маркера списку. Властивості списків дозволяють задати тип цього маркера і його положення щодо головного прямокутника. Відзначимо, що при відображенні списків тло елемента поширюється тільки на головний прямокутник; прямокутник маркера завжди прозорий.

2.11.5.2. Тип маркера списку: властивість **list-style-type**

Синтаксис: `list-style-type: disc | circle | square | decimal | decimal-leading-zero |`

`lower-roman | upper-roman | lower-greek | lower-alpha |`

`lower-latin | upper-alpha | upper-latin | hebrew |`

`armenian | georgian | cjk-ideographic | hiragana | katakana |`

`hiragana-iroha | katakana-iroha | none |`

[inherit](#)

Начально: `disc`

Застосовно: до елементів типу `display: list-item`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Підтримуються тільки `disc`, `circle`, `square`, `none`, `decimal`, `lower-roman`, `upper-roman`, `lower-alpha`, `upper-alpha` (4.0+)

Підтримуються тільки `disc`, `circle`, `square`, `none`, `decimal`, `lower-roman`, `upper-roman`, `lower-alpha`, `upper-alpha` (4.0+)

Властивість **list-style-type** задає *тип маркера списку*. Воно може приймати наступні значення:

<i>disc</i>	Чорний кружок (залежить від оглядача)
<i>circle</i>	Світлий кружок (залежить від оглядача)
<i>square</i>	Чорний квадратик (залежить від оглядача)
<i>decimal</i>	Десяткові числа (1, 2, 3, ...)
<i>decimal-leading-zero</i>	Десяткові числа із провідним нулем (01, 02, 03, ... 98, 99)
<i>lower-roman</i>	Рядкові римські числа (i, ii, iii, iv, v, ...)
<i>upper-roman</i>	Прописні римські числа (I, II, III, IV, V, ...)
<i>lower-greek</i>	Рядкові грецькі букви (α , β , γ , ..., ω)
<i>lower-alpha</i>	Рядкові латинські букви (a, b, c, ..., z)
<i>lower-latin</i>	Рядкові латинські букви (a, b, c, ..., z)
<i>upper-alpha</i>	Прописні латинські букви (A, B, C, ..., Z)
<i>upper-latin</i>	Прописні латинські букви (A, B, C, ..., Z)
<i>hebrew</i>	Традиційна ивритская нумерація
<i>armenian</i>	Традиційна вірменська нумерація
<i>georgian</i>	Традиційна грузинська нумерація (an, ban, gan, ..., he, tan, in, in-an, ...)
<i>cjk-ideographic</i>	Далекосхідна ідеографічна нумерація
<i>hiragana</i>	Японська нумерація абеткою хирогана (a, i, u, e, o, ka, ki, ...)
<i>katakana</i>	Японська нумерація абеткою катакана (A, I, U, E, O, KA, KI, ...)
<i>hiragana-iroha</i>	Японська нумерація хирогана-ироха (i, ro, ha, ni, ho, he, to, ...)
<i>katakana-iroha</i>	Японська нумерація катакана-ироха (I, RO, HA, NI, HO, HE, TO, ...)

Специфікація CSS не визначає, які символи повинні використатися як маркери списку, коли заданий алфавіт буде вичерпаний. Наприклад, 27-й елемент списку типу *lower-alpha* не визначений. Тому алфавітні типи маркерів повинні використатися тільки для коротких списків заздалегідь відомого розміру. Якщо оглядач не підтримує заданий тип маркера, то він повинен замінятися на *decimal*.

Ета властивість застосовується до списку тільки тоді, коли значенням властивості **list-style-image** є *none* або зазначений у ньому графічний образ не може бути завантажений.

Наступний приклад нумерує елементи нумерованого списку рядковими римськими числівниками:

```
OL { list-style-type: lower-roman }
```

2.11.5.3. Графічний маркер списку: властивість **list-style-image**

Синтаксис: `list-style-image: <uri> | none | inherit`

Начально: *none*

Застосовно: до елементів типу `display: list-item`

Наслідувано: да

Відсотки: не використовуються

Пристрою: визуальні

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **list-style-image** задає *графічний маркер списку* вказівкою на URI відповідного графічного образу. Якщо воно дорівнює *none*, то тип маркера списку задається властивістю **list-style-type**.

Наступний приклад поміщає в початок кожного елемента маркірованого списку графічний образ "ellipse.gif":

```
UL { list-style-image: url("http://mylibrary.com/ellipse.gif") }
```

2.11.5.4. Позиція маркера списку: властивість **list-style-position**

Синтаксис: `list-style-position: inside | outside | inherit`

Начально: `outside`

Застосовно: до елементів типу `display: list-item`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **list-style-position** задає *положення маркера списку* щодо елемента списку. Воно може приймати наступні значення:

inside

Прямокутник маркера стає першим текстовим блоком головного прямокутника.

outside

Прямокутник маркера розміщується поза головним прямокутником.

Наприклад, що впливає фрагмент HTML-документа

```
<UL style="list-style-type: square; list-style-position: inside; width: 40%">
  <LI>Перший елемент списку розташований таким от образом</LI>
  <LI>Другий елемент списку розташовується так же.</LI>
</UL>
```

буде відображатися так (Netscape Navigator 4.x відображає цей приклад невірно!):

- Перший елемент списку розташований у такий от спосіб.
- Другий елемент списку розташовується так само.

2.11.5.5. Завдання властивостей списку: властивість **list-style**

Синтаксис: `list-style: [<list-style-type> | <list-style-image> | <list-style-position>] | inherit`

Начально: не визначено для скорочень

Застосовно: до елементів типу `display: list-item`

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные](#)

Підтримка: Відповідає стандарту (4.0+)

Не підтримується

Властивість **list-style** є скороченням для властивостей **list-style-type**, **list-style-image** і **list-style-position**. Воно дозволяє задати всі властивості списку одночасно. З його допомогою стиль маркірованого списку з попереднього приклада міг би бути заданий так:

```
<UL style="list-style: square inside; width: 40%">
```

2.12.1. Загальний опис

Сторінкові пристрої виводу (наприклад, папір у принтері або сторінки, виведені на екран дисплея) відрізняються від безперервних пристроїв тим, що документ розбивається на окремі сторінки. Для роботи з такими пристроями в CSS уведена *модель сторінки*, що дозволяє авторам задавати розміри сторінки і її границь, а також управляти переходом на нову сторінку. Модель сторінки дозволяє вказати, як елементи документа повинні розташовуватися в *прямокутнику сторінки*, що має фіксовані розміри. При цьому розміри прямокутника не обов'язково збігаються з розмірами реальної сторінки, на яку буде зроблене остаточне відображення документа. CSS не визначає, як саме оглядач буде перетворювати прямокутник сторінки в реальні сторінки, але дозволяє задати їхній розмір й орієнтацію.

2.12.2. Директива @page

Директива **@page** призначена для завдання властивостей сторінок. Вона має такий вигляд:

```
@page [селектора-сторінки] {  
    правило;  
    ...  
    правило;  
}
```

Селектори-сторінки — це або [ім'я сторінки](#), або один з наступних псевдокласов:

:first	Перша сторінка документа
:left	Ліва сторінка (для двосторонньої печатки)
:right	Права сторінка (для двосторонньої печатки)

Правила, зазначені в директиві **@page** із селектором *:left* або *:right*, перекривають правила директиви **@page** без селекторів. Правила, зазначені в директиві **@page** із селектором *:first*, перекривають правила директиви **@page** із селектором *:left* або *:right* (це залежить від напрямку виводу тексту). Нарешті, правила, зазначені в директиві **@page** з ім'ям сторінки, перекривають правила інших директив **@page**.

Підтримка: Синтаксично відповідає стандарту, але підтримується тільки

шаблонами печатки MSHTML (5.5+)

Не підтримується

2.12.2.1. Завдання границь сторінки

Для завдання границь прямокутника сторінки використовуються властивості **margin-left**, **margin-right**, **margin-top**, **margin-bottom** і **margin**. Приклад:

```
@page { margin: 2cm } /* Всі границі рівні 2 див */
```

2.12.2.2. Розмір сторінки: властивість size

Синтаксис: `size: <размер>{1,2} | auto | portrait | landscape | inherit`

Начально: `auto`

Застосовно: до сторінок

Наслідувано: не визначено

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **size** задає *розмір й орієнтацію прямокутника сторінки*. Розміри сторінки можуть бути або абсолютними, або відносними. Абсолютні розміри задаються одним або двома значеннями типу [<размер>](#). Якщо задано одне значення, то воно визначає розмір сторінки й по вертикалі, і по горизонталі. Якщо задані два значення, то перше визначає ширину сторінки, а друге — її висоту.

Відносні розміри задаються одним з наступних ключових слів:

<i>auto</i>	Прямокутник сторінки збігається з реальним аркушем по розмірі й орієнтації.
<i>portrait</i>	Прямокутник сторінки збігається з реальним аркушем по розмірі ; орієнтація книжкова.
<i>landscape</i>	Прямокутник сторінки збігається з реальним аркушем по розмірі ; орієнтація альбомна.

Приклади:

```
@page { size: 8.5in 11in }  
@page { size: portrait; margin: 10% }
```

2.12.2.3. Позначки: властивість **marks**

Синтаксис: `marks: [crop || cross] | none | inherit`

Начально: `none`

Застосовно: до сторінок

Наслідувано: не визначено

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

У високій пресі часто використовуються спеціальні позначки на полях сторінки. Властивість **marks** дозволяє задати *тип позначок на сторінці*. Воно може приймати наступні значення:

<i>none</i>	Не друкувати позначки.
<i>crop</i>	Позначки, що вказують місце обріза сторінки.
<i>cross</i>	Позначки, використовувані для вирівнювання сторінок.

Приклад:

```
@page { marks: crop cross } /* Друкувати обидва типи позначок */
```

2.12.3. Імена сторінок: властивість **page**

Синтаксис: `page: <ідентифікатор> | auto`

Начально: `auto`

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **page** дозволяє привласнити елементу *ім'я сторінки* для того, щоб використати властивості директиви **@page**, селектором якої є дане ім'я. Наприклад, що впливають правила вказують, що всі таблиці документа необхідно друкувати в альбомній орієнтації:

```
@page rotated { size: landscape }  
TABLE { page: rotated; ... }
```

За замовчуванням ця властивість має значення *auto*, тому до елементів, що не мають імен сторінок, застосовуються правила директиви **@page** без іменного селектора.

2.12.4. Керування розривами сторінок

2.12.4.1. Розрив сторінки перед елементом: властивість **page-break-before**

Синтаксис: `page-break-before: auto | always | avoid | left | right | inherit`

Начально: `auto`

Застосовно: до блокових елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Підтримуються `auto`, `always` й `""` замість `avoid` (4.0+)

Не підтримується

Властивість **page-break-before** управляє *розривом сторінки перед печаткою* даного елемента. Воно може приймати наступні значення:

<i>auto</i>	Розрив сторінки виробляється звичайним образом , після заповнення сторінки.
<i>always</i>	Завжди починати печатка даного елемента з нової сторінки.
<i>avoid</i>	Ніколи не робити розриву сторінки перед даним елементом.
<i>left</i>	Один або два розриви сторінки, щоб даний елемент завжди друкувався на лівій сторінці.
<i>right</i>	Один або два розриви сторінки, щоб даний елемент завжди друкувався на правій сторінці.

Приклад:

```
H1 { page-break-before: always }
```

2.12.4.2. Розрив сторінки після елемента: властивість **page-break-after**

Синтаксис: `page-break-after: auto | always | avoid | left | right | inherit`

Начально: `auto`

Застосовно: до блокових елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Підтримуються `auto`, `always` й `""` замість `avoid` (4.0+)

Не підтримується

Властивість **page-break-after** управляє *розривом сторінки після печатки* даного елемента. Воно може приймати наступні значення:

<i>auto</i>	Розрив сторінки виробляється звичайним образом , після заповнення сторінки.
-------------	---

<i>always</i>	Завжди починати печатка наступного елемента з нової сторінки.
<i>avoid</i>	Ніколи не робити розриву сторінки після даного елемента.
<i>left</i>	Один або два розриви сторінки, щоб наступний елемент завжди друкувався на лівій сторінці.
<i>right</i>	Один або два розриви сторінки, щоб наступний елемент завжди друкувався на правій сторінці.

Приклад:

```
H1 { page-break-after: avoid }
```

2.12.4.3. Розрив сторінки усередині елемента: властивість **page-break-inside**

Синтаксис: `page-break-inside: auto | avoid | inherit`

Начально: `auto`

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **page-break-inside** управляє розривами сторінки при печатці даного елемента. Воно може приймати наступні значення:

<i>auto</i>	Розрив сторінки виробляється звичайним образом , після заповнення сторінки.
<i>avoid</i>	Ніколи не робити розриву сторінки усередині даного елемента.

Приклад:

```
H1 { page-break-after: avoid }
```

2.12.4.4. Кількість рядків наприкінці сторінки: властивість **orphans**

Синтаксис: `orphans: <целое> | inherit`

Начально: `2`

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **orphans** задає мінімальна кількість рядків абзацу, що повинне залишитися наприкінці сторінки при печатці даного елемента. Приклад:

```
P { orphans: 3 }
```

2.12.4.5. Кількість рядків на початку сторінки: властивість **widows**

Синтаксис: `widows: <целое> | inherit`

Начально: `2`

Застосовно: до блокових елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [визуальные страничные](#)

Підтримка: Не підтримується

Не підтримується

Властивість **widows** задає мінімальна кількість рядків абзацу, що повинне залишитися на початку сторінки при печатці даного елемента. Приклад:

```
P { widows: 3 }
```

Глава 2.13. Звукові таблиці стилів

2.13.1. Загальний опис

Крім візуального відображення документів, CSS підтримує їхнє *звукове відображення*, засноване на синтезації мови й звукових сигналів. Звукові таблиці стилів можуть бути як доповненням до візуальних, так й їхньою альтернативою.

Звукове відображення документа відбувається в тривимірному фізичному просторі (звуковому оточенні) і в часі (ми можемо вказати, які звуки повинні передувати, впливати або звучати одночасно з іншими звуками). Крім того, CSS дозволяє авторам управляти якістю синтезованої мови (типом голосу, частотою, модуляцією й т.п.).

2.13.2. Гучність: властивість **volume**

Синтаксис: `volume: <число> | <процент> | silent | x-soft | soft | medium | loud | x-loud | inherit`

Начально: `medium`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: щодо наслідуваного значення

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **volume** задає *гучність звуку* (точніше, розмір медіани звукової хвилі). Воно задається одним з наступних способів:

[<число>](#)

Задає гучність звуку числом у діапазоні від 0 до 100. При цьому 0 відповідає мінімально можливій гучності, а 100 — максимально прийнятної гучності.

[<процент>](#)

Обчислюється щодо наслідуваного значення, а потім обрізається по діапазоні від 0 до 100.

`silent`

Відключити звук. Це не те ж саме, що гучність 0!

`x-soft`

Те ж, що гучність 0.

`soft`

Те ж, що гучність 25.

`medium`

Те ж, що гучність 50.

`loud`

Те ж, що гучність 75.

`x-loud`

Те ж, що гучність 100.

Фактична гучність звуку визначається оглядачем. Приклад: наступне правило задає неголосне звучання всього документа:

```
BODY { volume: soft }
```

2.13.3. Завдання пауз

CSS дозволяє задати розміри пауз перед проголошенням елемента й після нього. Вони мають тип <розміру-паузи>, що визначається в такий спосіб:

<пауза^--паузи-розміру-паузи> = [<время>](#) | [<процент>](#)
[<время>](#)

Задає абсолютний розмір паузи в секундах або мілісекундах.

[<процент>](#)

Заснований на величині, зворотної значенню властивості **speech-rate**.

Якщо, наприклад,

це значення дорівнює 120 слів у хвилину (тобто одне слово займає 500 мс), те

пауза в 20% дорівнює $20\% * 500 \text{ мс} = 100 \text{ мс}$.

Для того, щоб таблиця стилів була стійка до зміни швидкості мови, рекомендується задавати розміри пауз у відсотках.

2.13.3.1. Пауза перед елементом: властивість **pause-before**

Синтаксис: `pause-before:` [<размер-паузы>](#) | [inherit](#)

Начально: залежить від оглядача

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: див. вище

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **pause-before** задає *паузу перед проголошенням елемента*. Пауза уставляється між умістом елемента й будь-яким умістом властивості **cue-before**. Приклад:

```
H1 { pause-before: 20ms }
```

2.13.3.2. Пауза після елемента: властивість **pause-after**

Синтаксис: `pause-after:` [<размер-паузы>](#) | [inherit](#)

Начально: залежить від оглядача

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: див. вище

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **pause-after** задає *паузу після проголошення елемента*. Пауза уставляється між умістом елемента й будь-яким умістом властивості **cue-after**. Приклад:

```
H1 { pause-after: 20ms }
```

2.13.3.3. Паузи до й після: властивість **pause**

Синтаксис: `pause:` [<размер-паузы>](#){1,2} | [inherit](#)

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: див. вище

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **pause** є скороченням для властивостей **pause-before** і **pause-after**. Якщо зазначені два значення, то перше з них задає паузу перед проголошенням елемента, а друге — паузу після його проголошення. Якщо зазначено тільки одне значення, то воно задає значення обох пауз. Приклади:

```
H1 { pause: 20ms } /* pause-before: 20ms; pause-after: 20ms */
H1 { pause: 30ms 40ms } /* pause-before: 30ms; pause-after: 40ms */
```

2.13.4. Завдання звукових сигналів

CSS дозволяє виводити *звукові сигнали* перед проголошенням елемента й після нього. Вони мають тип `<сигнал^–сигнал-звук-сигнал>`, що визначається в такий спосіб:

`<сигнал^–сигнал-звук-сигнал> = <uri> | none`

[<uri>](#)

Задає URI ресурсу, що містить звуковий файл. Якщо ресурс не є аудіофайлом, те він повинен ігноруватися.

`none`

Не виводити звукового сигналу.

2.13.4.1. Сигнал перед елементом: властивість **cue-before**

Синтаксис: `cue-before: <звуковой-сигнал> | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **cue-before** задає звуковий сигнал перед проголошенням елемента. Приклад:

```
H1 { cue-before: url("bell.wav") }
```

2.13.4.2. Сигнал після елемента: властивість **cue-after**

Синтаксис: `cue-after: <звуковой-сигнал> | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: немає

Відсотки: не використовуються

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **cue-after** задає звуковий сигнал після проголошення елемента. Приклад:

```
H1 { cue-after: url("dong.aiff") }
```

2.13.4.3. Сигнали до й після: властивість **cue**

Синтаксис: `cue: <звуковой-сигнал>{1,2} | inherit`

Начально: не визначено для скорочень

Застосовно: до всіх елементів

Наслідувано: немає
Відсотки: не використовуються
Пристрою: [звуків](#)

Підтримка: Не підтримується
Не підтримується

Властивість **cue** є скороченням для властивостей **cue-before** і **cue-after**. Якщо зазначені два значення, то перше з них задає сигнал перед проголошенням елемента, а друге — сигнал після його проголошення. Якщо зазначено тільки одне значення, то воно задає значення обох звукових сигналів. Наприклад, що впливають два правила еквівалентні:

```
H1 { cue: url(pop.au) }  
H1 { cue-before: url(pop.au); cue-after: url(pop.au) }
```

2.13.5. Фоновий звук: властивість **play-during**

Синтаксис: `play-during: <uri> mix? repeat? | auto | none | inherit`
Начально: `auto`
Застосовно: до всіх елементів
Наслідувано: немає
Відсотки: не використовуються
Пристрою: [звуків](#)

Підтримка: Не підтримується
Не підтримується

Властивість **play-during** задає *фоновий звук*, що повинен звучати під час проголошення вмісту елемента. Воно задається одним з наступних способів:

<uri>

Задає URI ресурсу, що містить звуковий файл. Якщо ресурс не є аудіофайлом, то він повинен ігноруватися.

mix

Якщо задано це ключове слово, то фоновий звук елемента повинен змішуватися з фоновим звуком його батька. Якщо його ні, то фоновий звук елемента заміняє фоновий звук його батька.

repeat

Якщо задано це ключове слово, то фоновий звук елемента повторюється доти, поки не закінчиться проголошення елемента. У протилежному випадку фоновий звук звучить один раз.

auto

Продовжує звучати фоновий звук батьківського елемента.

none

На час проголошення даного елемента фоновий звук відключається.

Приклади:

```
P.sad { play-during: url("violin.wav") repeat }  
SPAN.quite { play-during: none }
```

2.13.6. Просторові характеристики звуку

Просторове звучання забезпечує природний спосіб декількох голосів з різних крапок простору, як у реальному житті. Відтворення такого звучання забезпечується різними аудіосистемами: від звичайних

стереонаушників до домашніх театрів. Для керування такими ефектами CSS дозволяє задавати два кути повороту звуку: по горизонталі (**azimuth**) і по вертикалі (**elevation**).

2.13.6.1. Горизонтальний кут звуку: властивість **azimuth**

Синтаксис: `azimuth: <угол> | [[left-side | far-left | left | center-left | center | center-right | right | far-right | right-side] | behind] | leftwards | rightwards | inherit`

Начально: `center`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **azimuth** задає горизонтальний кут звучання. Воно задається одним з наступних способів:

<угол>

Задає кут у діапазоні від -360deg до 360deg. При цьому 0deg відповідає звучанню із центра, 90deg – повороту праворуч, 180deg – повороту назад, а 270deg – повороту ліворуч.

`left-side`

Те ж, що 270deg. Разом з `behind`, 270deg.

`far-left`

Те ж, що 300deg. Разом з `behind`, 240deg.

`left`

Те ж, що 320deg. Разом з `behind`, 220deg.

`center-left`

Те ж, що 340deg. Разом з `behind`, 200deg.

`center`

Те ж, що 0deg. Разом з `behind`, 180deg.

`center-right`

Те ж, що 20deg. Разом з `behind`, 160deg.

`right`

Те ж, що 40deg. Разом з `behind`, 140deg.

`far-right`

Те ж, що 60deg. Разом з `behind`, 120deg.

`right-side`

Те ж, що 90deg. Разом з `behind`, 90deg.

`leftwards`

Поворот проти вартовий стрілки на 20deg (тобто вирахування 20deg).

`rightwards`

Поворот по годинній стрілці на 20deg (тобто додавання 20deg).

Приклади:

```
H1 { azimuth: 30deg }
TD.a { azimuth: far-right } /* 60deg */
#id12 { azimuth: behind far-right } /* 120deg */
P.comment { azimuth: behind } /* 180deg */
```

2.13.6.2. Вертикальний кут звуку: властивість **elevation**

Синтаксис: `elevation: <угол> | below | level | above | higher | lower | inherit`

Начально: `level`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **elevation** задає *вертикальний кут звучання*. Воно задається одним з наступних способів:

<угол>

Задає кут у діапазоні від -90deg до 90deg. При цьому 0deg відповідає звучанню на рівні слухача, 90deg – звучанню прямо над його головою, а -90deg – звучанню прямо в нього під ногами.

below

Те ж, що -90deg.

level

Те ж, що 0deg.

above

Те ж, що 90deg.

higher

Збільшує поточний кут на 10deg.

lower

Зменшує поточний кут на 10deg.

Приклади:

```
H1 { elevation: above }
TR.a { elevation: 60deg }
TR.b { elevation: 30deg }
TR.c { elevation: level }
```

2.13.7. Характеристики мови

2.13.7.1. Швидкість мови: властивість **speech-rate**

Синтаксис: `speech-rate: <число> | x-slow | slow | medium | fast | x-fast | faster | slower | inherit`

Начально: *medium*

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **speech-rate** задає *швидкість мови*. Воно задається одним з наступних способів:

<число>

Явно задає кількість слів, вимовних у хвилину.

x-slow

80 слів у хвилину.

slow

120 слів у хвилину.

medium

180 – 200 слів у хвилину.

fast

300 слів у хвилину.

x-fast

500 слів у хвилину.

faster
Збільшує поточну швидкість мови на 40 слів у хвилину.
slower
Зменшує поточну швидкість мови на 40 слів у хвилину.

Приклад:

```
H1 { speech-rate: slow }
```

2.13.7.2. Голос: властивість **voice-family**

Синтаксис: `voice-family: [[<имя-голоса>|<родовое-имя>],]*`
`[<имя-голоса>|<родовое-имя>]|inherit`
Начально: залежить від оглядача
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **voice-family** задає список імен голосів для проголошення вмісту елемента. Цей список складається з імен голосів, розділених комами. Імена розташовуються в порядку переваги. Наприклад, що впливає властивість

```
voice-family: romeo, male;
```

варто розуміти так: "використати голос romeo; якщо його ні, те використати родовий голос male". Такий список необхідний, оскільки ми заздалегідь не знаємо, які саме голоси підтримуються синтезаторами мови наших користувачів.

Ім'я голосу може бути задано двома способами:

<голос^-голоси-імені-голосу>
Задає назва конкретного голосу. Якщо ця назва містить пробіли, то воно повинне бути укладене в лапки або апострофи.
<ім'я^-ім'я-родов-ім'я>
Одне з наступних визначених імен голосів: *male* (чоловічий), *female* (жіночий)
i child (дитячий).

Для досягнення найбільшої сумісності рекомендується задавати родові ім'я останнім у списку. У цьому випадку, якщо оглядач не знайде на комп'ютері-клієнті заданих голосів, він використає свій родовий голос із заданим ім'ям.

2.13.7.3. Частота звуку: властивість **pitch**

Синтаксис: `pitch: <частота> | x-low | low | medium | high | x-high |`
`inherit`
Начально: *medium*
Застосовно: до всіх елементів
Наслідувано: да
Відсотки: не використовуються
Пристрою: [звукowe](#)

Підтримка: Не підтримується

Не підтримується

Властивість **pitch** задає *середню частоту звуку*. Відзначимо, що середня частота звуку залежить від типу голосу. Наприклад, середня частота чоловічих голосів 120 Гц, а жіночих — 210 Гц. Частота задається одним з наступних способів:

<частота>

Явно задає частоту звуку в герцах.

x-low, low, medium, high, x-high

Указують відносну частоту звуку в порядку її збільшення. Оглядач повинен підібрати відповідну частоту залежно від типу голосу.

Приклад:

```
H1 { pitch: high }
```

2.13.7.4. Варіація частоти: властивість **pitch-range**

Синтаксис: `pitch-range: <число> | inherit`

Начально: 50

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **pitch-range** задає *варіацію середньої частоти звуку*. Його значенням є число в діапазоні від 0 до 100. При цьому значення 0 відповідає абсолютно монотонному голосу, значення 50 — звичайному голосу, а значення більше 50 створюють анимированное голосу. Приклад:

```
P { pitch-range: 30 }
```

2.13.7.5. Наголос: властивість **stress**

Синтаксис: `stress: <число> | inherit`

Начально: 50

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **stress** задає *висоту піків в інтонації голосу*. Його зміст істотно залежить від мови, на якому вимовляється текст документа. У нетонових мовах, таких, як російський або англійський, воно дозволяє вказати, яка варіація частоти звуку в тих мовних фрагментах, які перебувають під наголосом.

Значенням цієї властивості є число в діапазоні від 0 до 100, причому значення 50 повинне відповідати звичайному способу інтонирования наголосів для даної мови. Приклад:

```
P.stressed { stress: 70 }
```

2.13.7.6. Тембр: властивість **richness**

Синтаксис: `richness: <число> | inherit`

Начально: 50

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **richness** задає *тембр голосу*. Його значенням є число в діапазоні від 0 до 100. Чим більше це значення, тим богаче тембр голосу, тим краще він наповнює приміщення. Приклад:

```
H1 { richness: 75 }
```

2.13.8. Характеристики вимови

2.13.8.1. Спосіб вимови: властивість **speak**

Синтаксис: `speak: none | normal | spell-out | inherit`

Начально: `normal`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуків](#)

Підтримка: Не підтримується

Не підтримується

Властивість **speak** задає *спосіб вимови*. Воно може приймати наступні значення:

<i>none</i>	Елемент не вимовляється. Однак, його нащадки можуть перевизначити значення цієї властивості й тоді будуть вимовлятися.
<i>normal</i>	Звичайна вимова відповідно до норм мови .
<i>spell-out</i>	Вимова по буквах. Корисно для аббревіатур й акронімів.

Примітка. Елемент із властивістю **volume: silent** не вимовляється, але час, що вимагається йому для проголошення, однаково йому виділяється, включаючи паузи перед ним і після нього. Елемент із властивістю **speak: none** не вимовляється, і час на його проголошення не виділяється.

Пример. Наступне правило задає вимова аббревіатур по буквах:

```
ABBR { speak: spell-out }
```

2.13.8.2. Вимова пунктуації: властивість **speak-punctuation**

Синтаксис: `speak-punctuation: code | none | inherit`

Начально: `none`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуків](#)

Підтримка: Не підтримується

Не підтримується

Властивість **speak-punctuation** задає *спосіб вимови знаків пунктуації* (ком, двокрапок і т.п.). За замовчуванням, його значення дорівнює *none*, тобто назви знаків пунктуації не вимовляються, а їхня наявність виділяється паузами. Якщо ж значення цієї властивості *code*, то назви знаків пунктуації вимовляються в процесі синтезації мови. Приклад:

```
P { speak-punctuation: code }
```

2.13.8.3. Вимова числівників: властивість **speak-numeral**

Синтаксис: `speak-numeral: digit | continuous | inherit`

Начально: `continuous`

Застосовно: до всіх елементів

Наслідувано: да

Відсотки: не використовуються

Пристрою: [звуковые](#)

Підтримка: Не підтримується

Не підтримується

Властивість **speak-numeral** задає спосіб вимови числівників. За замовчуванням, його значення дорівнює *continuous*, тобто число "123" вимовляється "Сто двадцять три". Якщо ж значення цієї властивості *digit*, то це число вимовляється по цифрах, тобто "Один два три". Приклад:

```
TD { speak-numeral: digit }
```

2.13.8.4. Вимова заголовків таблиць: властивість **speak-header**

Опис цієї властивості наведено в [разделе, посвященном таблицам](#).