**Самовчитель CSS**

F

### 

### Авторскі права

Самовчитель CSS (далі твір) поширюється на умовах такої ліцензії Creative Commons:

«Зазначення авторства - Некомерційне використання - Без похідних» (Attribution-NonCommercial-NoDerivs 2.5 Generic (CC BY-NC-ND 2.5)).

Ви зможете вільно

* копіювати, поширювати і передавати твір.

На наступних умовах

* обов'язково вказувати автора твору (Влад Мержевіч).
*  не можна використовувати твір для заробітку.
*  забороняється змінювати твір або створювати інші з його допомогою.

### Позначення

Для зручності і наочного подання матеріалу використовується поділ за кольорами різних елементів.

**<strong>** — тег.

align — атрибут тега, стильова властивість, ключове слово або виділення.

right — значення.

layer — ім'я класу або ідентифікатора.

File > Open — пункт меню зазначеної програми.

— клавіша на клавіатурі.



Tab

 Цей символ покликаний привернути увагу до деякого зауваження в тексті.

##### Браузери

У таблиці з браузерами зустрічаються такі зображення:

 — властивість повністю підтримується браузером з усіма допустимими значеннями;

 — властивість браузером не сприймається і ігнорується;

 — властивість підтримується лише частково, наприклад, не всі допустимі значення діють або властивість застосовується не до всіх елементів, які вказані в специфікації;

 — властивість розуміється браузером, але при його роботі можлива поява різних помилок. У примітці зазвичай вказується, якого роду помилки виявляються в браузері.

##### Приклади

Як правило, приклади містять список браузерів, в яких перевірявся код. Для скорочення місця браузери позначаються двома буквами з номером версії.

IE — Internet Explorer. Cr — Google Chrome. Op — Opera.

Sa — Apple Safari. Fx — Mozilla Firefox.

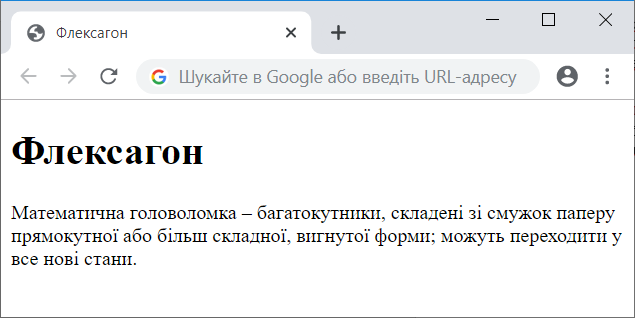
Працездатність прикладу в браузері позначається кольором, що збігається з кольорами в таблиці з браузерами.

HTML задає основну структуру веб-сторінки, а також вказує, які елементи на ній присутні. Саме оформлення веб-сторінки, положення та вид елементів покладено на стилі або CSS (Cascading Style Sheets, каскадні таблиці стилів). Коли говорять про верстку веб-сторінок, мається на увазі синергія HTML і CSS. Що таке синергія? Сам HTML не представляє окремого інтересу, в силу своєї простоти і обмеженості. Також і CSS не грає окремої ролі, оскільки прив'язується до певних елементів коду і задає їх оформлення. Тому працюючи разом в одній зв'язці, вони перетворюють скромну сторінку в той документ, який придумав і намалював дизайнер. Таке взаємне посилення властивостей, сумуючий ефект і є синергією.

Будь-яка веб-сторінка це, по суті, комбінація HTML-коду і CSS-коду. Без основних знань цих технологій не вийде грамотно зверстати жоден документ. Тому перша глава присвячена основам CSS і його застосуванню на практиці. Якщо ви вважаєте, що це вже вам відомо, можете пропустити цю главу і перейти до наступної.

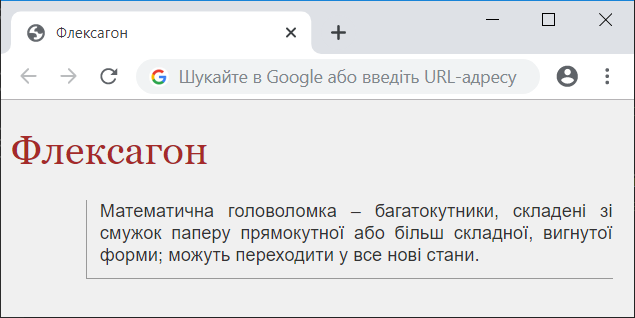
##### Що таке стилі?

Стилі являють собою набір параметрів, керуючих видом і станом елементів веб-сторінки. Щоб стало зрозуміло, про що йде мова, подивимося на рис. 1.1.



*Рис. 1.1. Веб-сторінка, створена тільки на HTML*

Це звичайна веб-сторінка, оформлена без жодних надмірностей. Той же самий документ, але вже з додаванням стилів набуває зовсім інший вигляд (рис. 1.2).



*Рис. 1.2. Веб-сторінка, створена на HTML і CSS*

Зміна разюча, тому заглянемо в код, щоб зрозуміти, в чому ж різниця (приклад 1.1).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.1. Вихідний код документа | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html xmlns="http://www.w3.org/1999/xhtml">  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Флексагон</title>  <link rel="stylesheet" href="style.css" type="text/css" />  </head>  <body>  <h1>Флексагон</h1>  <p>Математична головоломка – багатокутники, складені зі смужок паперу прямокутної або більш складної, вигнутої форми; можуть переходити у все нові стани.</p>  </body>  </html> | | | | | | | | | |

Сам код HTML ніяких змін не зазнав і єдине додавання - це рядок <link rel="stylesheet" href="style.css" type="text/css"/>. Він посилається на зовнішній файл з описом стилів під ім'ям style.css. Вміст цього файлу показано в прикладі 1.2.

Приклад 1.2. Вміст стильового файлу style.css

body {

font-family: Arial, Verdana, sans-serif; /\* Сімейство шрифтів \*/ font-size: 11pt; /\* Розмір основного шрифту в пунктах \*/ background-color: #f0f0f0; /\* Колір фону веб-сторінки \*/

color: #333; /\* Колір основного тексту \*/

}

h1 {

color: #a52a2a; /\* Колір заголовку \*/

font-size: 24pt; /\* Розмір шрифту в пунктах \*/

font-family: Georgia, Times, serif; /\* Сімейство шрифтів \*/ font-weight: normal; /\* Нормальне зображення тексту \*/

}

p {

text-align: justify; /\* Вирівнювання по ширині \*/ margin-left: 60px; /\* Лівий відступ в пікселях \*/ margin-right: 10px; /\* Відступ справа в пікселях\*/

border-left: 1px solid #999; /\* Параметри лінії зліва \*/ border-bottom: 1px solid #999; /\* Параметри лінії знизу \*/ padding-left: 10px; /\* Відступ від лінії зліва до тексту \*/ padding-bottom: 10px; /\* Відступ від лінії знизу до тексту\*/

}

У файлі style.css якраз і описані всі параметри оформлення таких тегів як **<body>**, **<h1>** і **<p>**. Зауважте, що самі теги в коді HTML пишуться як зазвичай.

Оскільки на файл зі стилем можна посилатися з будь-якого веб-документа, це призводить у результаті до скорочення обсягу повторюваних даних. А завдяки поділу коду та оформлення підвищується гнучкість управління видом документа і швидкість роботи над сайтом.

##### Типи стилів

Розрізняють декілька типів стилів, які можуть спільно застосовуватися до одного документу. Це стиль браузера, стиль автора і стиль користувача.

###### Стиль браузера

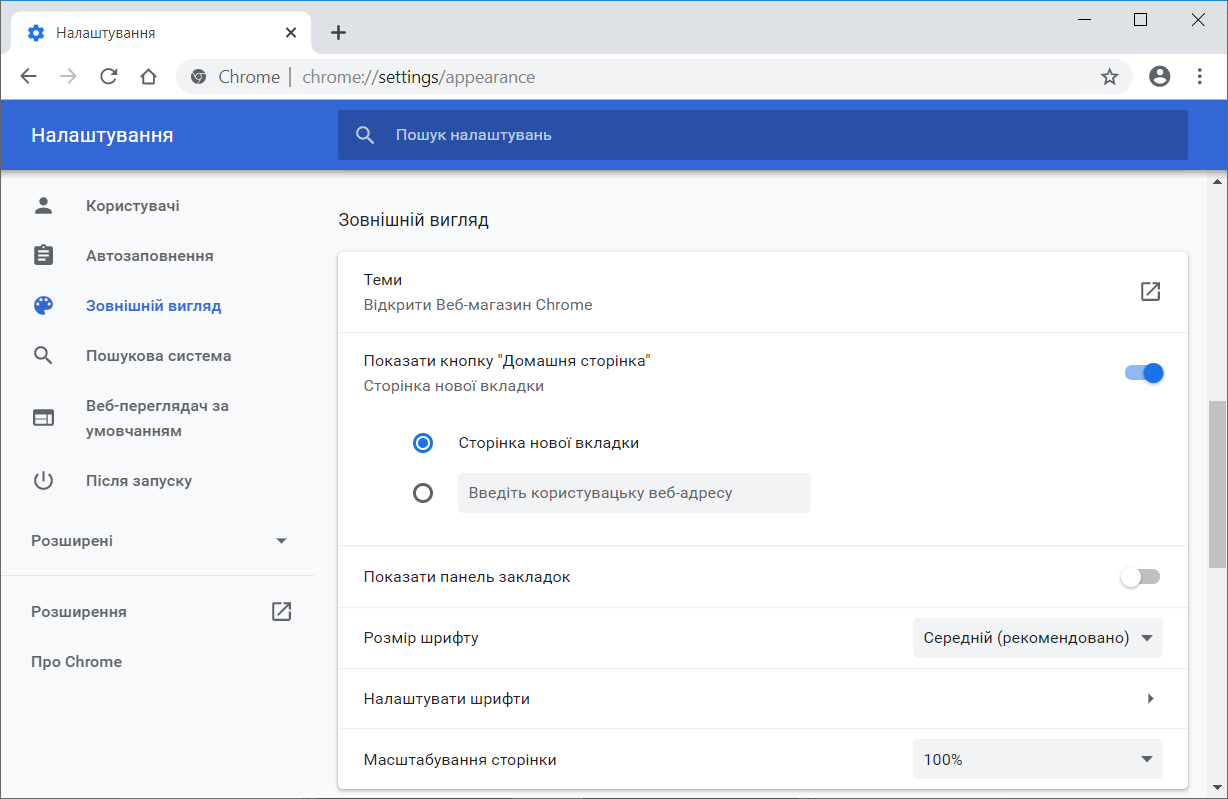
Оформлення, яке за замовчуванням застосовується до елементів веб-сторінки браузером. Це оформлення можна побачити в разі «голого» HTML, коли в документ не додається ніяких стилів. Наприклад, заголовок сторінки, що формується тегом **<H1>**, в більшості браузерів виводиться шрифтом із зарубками розміром 24 пункти.

###### Стиль автора

Стиль, який додає до документа його розробник. У прикладі 1.1 показаний один з можливих способів підключення авторського стилю.

###### Стиль користувача

Це стиль, який може включити користувач сайту через настройки браузера. Такий стиль має більш високий пріоритет і перевизначає вихідне оформлення документа. У браузері Google Chrome підключення стилю користувача робиться через меню Налаштування > Зовнішній вигляд > Показати кнопку «Домашня сторінка», як показано на рис. 1.3.



*Рис. 1.3. Підключення стилю користувача в браузері Google Chrome*

У браузері Opera аналогічну дію відбувається через команду Інструменти > Налаштування > Вкладка «Розширені» > Вміст > Кнопка «Параметри стилі».

Зазначені типи стилів можуть спокійно існувати один з одним, якщо вони не намагаються змінити вид одного елемента. У разі виникнення суперечності спочатку має пріоритет стиль користувача, потім стиль автора і останнім йде стиль браузера.

### Способи додавання стилів на сторінку

Для додавання стилів на веб-сторінку існує кілька способів, які розрізняються своїми можливостями і призначенням. Далі розглянемо їх докладніше.

##### Пов’язані стилі

При використанні пов'язаних стилів опис селекторів та їх значень розташовується в окремому файлі, як правило, з розширенням css, а для зв'язування документа з цим файлом застосовується тег **<link>**. Даний тег поміщається в контейнер **<head>**, як показано в прикладі 1.3.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.3. Підключенння пов’язаних стилів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Стилі</title>  <link rel="stylesheet" type="text/css" href="style/mysite.css" />  <link rel="stylesheet" type="text/css" [href="http://www.htmlbook.ru/main.css"](http://www.htmlbook.ru/main.css) />  </head>  <body>  <h1>Заголовок</h1>  <p>Текст</p>  </body>  </html> | | | | | | | | | |

Значення атрибутів тега **<link>** — rel і type залишаються незмінними незалежно від коду, як у цьому прикладі. Значення href задає шлях до CSS-файлу, він може бути заданий як відносно, так і абсолютно. Зауважте, що таким чином можна підключати таблицю стилів, яка знаходиться на іншому сайті.

Вміст файлу mysite.css підключається за допомогою тега **<link>** наведено в прикладі 1.4.

Приклад 1.4. Файл зі стилем

H1 {

color: #000080; font-size: 2em;

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

text-align: center; /\* Вирівнювання по центру \*/

}

P {

padding-left: 20px;

}

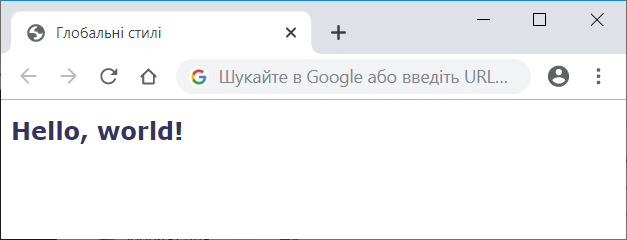
Як видно з цього прикладу, файл зі стилем не зберігає ніяких даних, крім синтаксису CSS. У свою чергу і HTML-документ містить лише посилання на файл зі стилем, тобто таким способом в повній мірі реалізується принцип поділу коду та оформлення сайту. Тому використання пов'язаних стилів є найбільш універсальним і зручним методом додавання стилю на сайт. Адже стилі зберігаються в одному файлі, а в HTML-документах вказується тільки посилання на нього.

##### Глобальні стилі

При використанні глобальних стилів властивості CSS описуються в самому документі і розташовуються в заголовку веб-сторінки. За своєю гнучкістю і можливостями цей спосіб додавання стилю поступається попередньому, але також дозволяє зберігати стилі в одному місці, в даному випадку прямо на сторінці за допомогою контейнера **<style>**, як показано в прикладі 1.5.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.5. Використання глобального стилю | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Глобальні стилі</title>  <style type="text/css"> H1 {  font-size: 1.2em;  font-family: Verdana, Arial, Helvetica, sans-serif; color: #333366;  }  </style>  </head>  <body>  <h1>Hello, world!</h1>  </body>  </html> | | | | | | | | | |

В даному прикладі визначений стиль тега **<h1>**, який потім можна повсюдно використовувати на даній веб-сторінці (рис. 1.5).



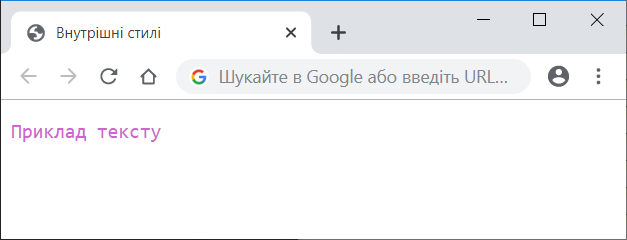
*Рис. 1.5. Вигляд заголовку, оформленого за допомогою стилів*

##### Внутрішні стилі

Внутрішній або вбудований стиль є по суті розширенням для одиночного тега використаного на поточній веб-сторінці. Для визначення стилю використовується атрибут style, а його значенням виступає набір стильових правил (приклад 1.6).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.6. Використання внутрішнього стилю | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Внутрішні стилі</title>  </head>  <body>  <p style="font-size: 120%; font-family: monospace; color: #cd66cc">Приклад тексту</p>  </body>  </html> | | | | | | | | | |

В даному прикладі стиль тега **<p>** задається за допомогою атрибута style, в якому через крапку з комою перераховуються стильові властивості (рис. 1.6).



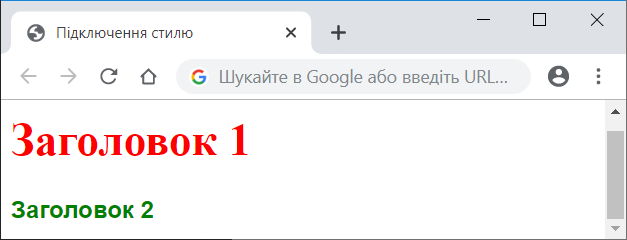
*Рис. 1.6. Використання внутрішніх стилів для зміни вигляду тексту*

 Внутрішні стилі рекомендується застосовувати на сайті обмежено або взагалі відмовитися від їх використання. Справа в тому, що додавання таких стилів збільшує загальний обсяг файлів, що веде до підвищення часу їх завантаження в браузері, і ускладнює редагування документів для розробників.

Всі описані методи використання CSS можуть застосовуватися як самостійно, так і в поєднанні один з одним. В цьому випадку необхідно пам'ятати про їх ієрархії. Першим завжди застосовується внутрішній стиль, потім глобальний стиль і в останню чергу пов'язаний стиль. У прикладі 1.7 застосовується відразу два методу додавання стилю в документ.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.7. Поєднання різних методів підключення стилю | XHTML 1.0 | CS  S 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Підключення стилю</title>  <style type="text/css"> H1 {  font-size: 1.2em;  font-family: Arial, Helvetica, sans-serif; color: green;  }  </style>  </head>  <body>  <h1 style="font-size: 36px; font-family: Times, serif; color: red">Заголовок 1</h1>  <h1>Заголовок 2</h1>  </body>  </html> | | | | | | | | | |

В даному прикладі перший заголовок задається червоним кольором розміром 36 пікселів за допомогою внутрішнього стилю, а наступний - зеленим кольором через таблицю глобальних стилів (рис. 1.7).



*Рис. 1.7. Результат застосування стилів*

##### Імпорт CSS

У поточну стильову таблицю можна імпортувати вміст CSS-файлу за допомогою команди @import. Цей метод допускається використовувати спільно зі зв'язаними або глобальними стилями, але ніяк не з вбудованими стилями. Загальний синтаксис наступний.

@import url("ім’я файлу") типи носіїв; @import "ім’я файлу" типи носіїв;

Після ключового слова @import вказується шлях до стильового файлу одним з двох наведених способів - за допомогою url або без нього. У прикладі 1.8 показано, як можна імпортувати стиль із зовнішнього файлу в таблицю глобальних стилів.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.8. Імпорт CSS в глобальну таблицю стилів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Імпорт</title>  <style type="text/css">  @import url("style/mysite.css"); H2 {  font-size: 1.2em;  font-family: Arial, Helvetica, sans-serif; color: green;  }  </style>  </head>  <body>  <h1>Заголовок 1</h1>  <h2>Заголовок 2</h2>  </body>  </html> | | | | | | | | | |

В даному прикладі показано підключення файлу mysite.css, який розташований в папці style.

Аналогічно відбувається імпорт і в файлі зі стилем, який потім підключається до документа (приклад 1.9).

Приклад 1.9. Імпорт в таблиці пов'язаних стилів

@import "/style/print.css"; @import "/style/palm.css"; BODY {

font-family: Arial, Verdana, Helvetica, sans-serif; font-size: 90%;

background: white; color: black;

}

В даному прикладі показано вміст файлу mysite.css, який додається до потрібних документів способом, показаним в прикладі 1.3, а саме за допомогою тега **<link>**.

### Типи носіїв

Широкий розвиток різних платформ і пристроїв змушує розробників робити під них спеціальні версії сайтів, що є досить трудомістким і проблематичним. Разом з тим, часи і потреби змінюються, і створення сайту для різних пристроїв є неминучим і необхідною ланкою його розвитку. З урахуванням цього в CSS введено поняття типу носія, коли стиль застосовується тільки для певного пристрою. У табл. 1.1 перераховані деякі типи носіїв.

*Табл. 1.1. Типи носіїв та їх опис*

|  |  |
| --- | --- |
| Тип | Опис |
| all | Всі типи. Це значення використовується за умовчанням. |
| aural | Мовні синтезатори, а також програми для відтворення тексту вголос. Сюди, наприклад, можна віднести мовні браузери. |
| braille | Пристрої, засновані на системі Брайля, які призначені для сліпих людей. |
| handheld | Портативні комп'ютери і аналогічні їм апарати. |
| print | Друкуючі пристрої на кшталт принтера. |
| projection | Проектор. |
| screen | Екран монітора. |
| tv | Телевізор. |

У CSS, щоб вказати типи носіїв застосовуються команди @media і @import, за допомогою яких можна визначити стиль для елементів залежно від того, виводиться документ на екран або на принтер.

 Ключові слова @media і @import відносяться до ет-правил. Така назва походить від назви символу @ - «at», з якого і починаються ці ключові слова. В Інтернеті для позначення символу @ застосовується усталений термін «равлик». Тільки ось використовувати вираз «равликове правило» язик не повертається.

При імпортуванні стилю через команду @import тип носія вказується після адреси файлу. При цьому допускається задавати відразу декілька типів, згадуючи їх через кому, як показано в прикладі 1.10.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.10. Імпорт стильового файлу | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Імпорт стилю</title>  <style type="text/css">  @import "style/main.css" screen; /\* Стиль для виводу результату на монітор \*/ @import "style/print.css" print, handheld; /\* Стиль для друку і смартфону \*/  </style>  </head>  <body>  <p>...</p>  </body>  </html> | | | | | | | | | |

В даному прикладі імпортується два файли – main.css призначений для зміни виду документа при його перегляді на екрані монітора, і print.css – при друкові сторінки і відображенні на смартфоні.

 Браузер Internet Explorer до сьомої версії включно не підтримує типи носіїв при імпорті стильового файлу. Більше того, при додаванні типу носія стильовий файл взагалі не завантажується.

Команда @media дозволяє вказати тип носія для глобальних або пов'язаних стилів і в загальному випадку має такий синтаксис.

@media тип носія 1 {

Опис стилю для типу носія 1

}

@media тип носія 2 {

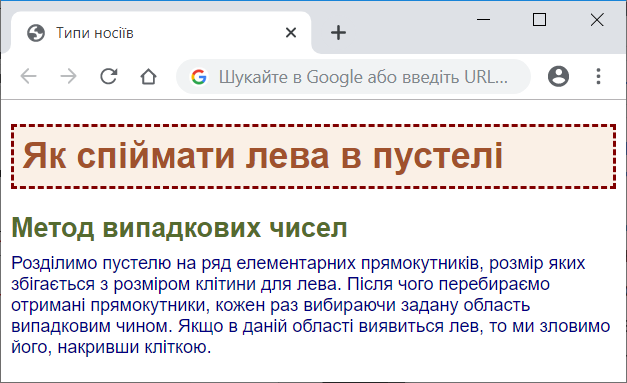
Опис стилю для типу носія 2

}

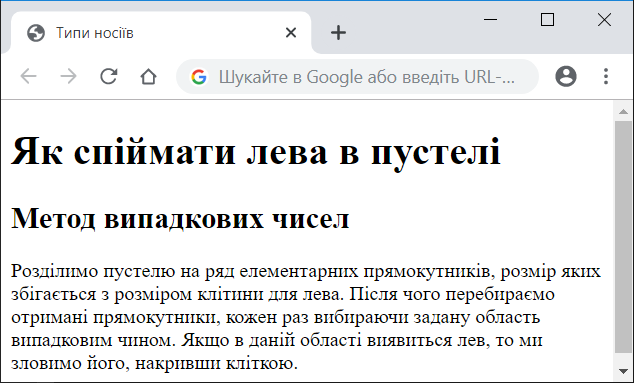
Після ключового слова @media йде один або кілька типів носія, перерахованих в табл. 1.1, якщо їх більше одного, то вони поділяються між собою комою. Після чого йдуть обов'язкові фігурні дужки, всередині яких йде звичайне опис стильових правил. У прикладі 1.11 показано, як задати різний стиль для друку та відображення на моніторі.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.11. Стилі для різних типів носіїв | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Типи носіїв</title>  <style type="text/css">  @media screen { /\* Стиль для відображення в браузері \*/ BODY {  font-family: Arial, Verdana, sans-serif; /\* Рубаний шрифт \*/ font-size: 0.9em; /\* Размір шрифту \*/  color: #000080; /\* Колір тексту \*/  }  H1 {  background: #faf0e6; /\* Колір тексту \*/  border: 2px dashed maroon; /\* Рамка навколо заголовку\*/ color: #a0522d; /\* Колір тексту \*/  padding: 7px; /\* Поля навколо тексту \*/  }  H2 {  color: #556b2f; /\* Колір тексту\*/ margin: 0; /\* Без відступів \*/  }  P {  margin-top: 0.5em; /\* Відступ зверху \*/  }  }  @media print { /\* Стиль для друку \*/ BODY {  font-family: Times, 'Times New Roman', serif; /\* Шрифт с засічками \*/  }  H1, H2, P {  color: black; /\* Черний колір тексту \*/  }  }  </style>  </head>  <body>  <h1> Як спіймати лева в пустелі </h1>  <H2> Метод випадкових чисел </h2>  <P> Розділимо пустелю на ряд елементарних прямокутників, розмір яких збігається з розміром клітини для лева. Після чого перебираємо отримані прямокутники, кожен раз вибираючи задану область випадковим чином. Якщо в даній області виявиться лев,  то ми зловимо його, накривши кліткою.</p>  </body>  </html> | | | | | | | | | |

В даному прикладі вводиться два стилі – один для зміни виду елементів при їх звичайному відображенні в браузері, а другий – при виведенні сторінки на друк. При цьому вигляд документа для різних носіїв може сильно відрізнятися між собою, наприклад, як це показано на рис. 1.8 і рис. 1.9.



*Рис. 1.8. Сторінка для відображення у вікні браузера*



*Рис. 1.9. Сторінка, призначена для друку*

Переглянути документ, у якого CSS встановлений як тип print можна, якщо роздрукувати певну сторінку. Або піти на хитрість і тимчасово замінити print на screen, щоб відобразити підсумок у браузері. Саме так було отримано рис. 1.9.

Команда @media застосовується в основному для формування одного стильового файлу, який розбитий на блоки по типу пристроїв. Іноді ж має сенс створити кілька різних CSS-файлів - один для друку, інший для відображення в браузері - і підключати їх до документа в міру необхідності. У подібному випадку слід скористатися тегом **<link>** з атрибутом media, значенням якого виступають усі ті ж типи, перераховані в табл. 1.1.

У прикладі 1.12 показано, як створювати посилання на CSS-файли, які призначені для різних типів носіїв.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1. 12. Підключення стилів для різних носіїв | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Різні носії</title>  <link media="print, handheld" rel="stylesheet" href="style/print.css" type="text/css" />  <link media="screen" rel="stylesheet" href="style/main.css" type="text/css" />  </head>  <body>  <p>...</p>  </body>  </html> | | | | | | | | | |

В даному прикладі використовуються дві таблиці пов'язаних стилів, одна для відображення в браузері, а друга - для друку документа і його перегляду на смартфоні. Хоча на сторінку завантажуються одночасно два різні стилі, застосовуються вони тільки для певних пристроїв.

Аналогічно можна використовувати і глобальні стилі, додаючи параметр media до тегу **<style>** (приклад 1.13).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.13. Стиль для смартфона | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Різні носії</title>  <style type="text/css" media="handheld"> BODY {  width: 320px; /\* Ширина сторінки \*/  }  </style>  </head>  <body>  <p>Розділимо пустелю на ряд елементарних прямокутників, розмір яких збігається з розміром клітини для лева. Після чого перебираємо отримані прямокутники, кожен раз вибираючи задану область випадковим чином. Якщо в даній області виявиться лев,  то ми зловимо його, накривши кліткою.</p>  </body>  </html> | | | | | | | | | |

В даному прикладі ширина для пристроїв типу handheld обмежена розміром 320 пікселів.

### Базовий синтаксис CSS

Як вже було раніше зазначено, стильові правила записуються в своєму форматі, відмінному від HTML. Основним поняттям виступає селектор - це деякий ім'я стилю, для якого додаються параметри форматування. Як селектора виступають теги, класи і ідентифікатори. Загальний спосіб запису має наступний вигляд.

селектор властивість значення

body { background: #ffc910; }

Спочатку пишеться ім'я селектора, наприклад, **TABLE**, це означає, що всі стильові правила будуть застосовуватися до тега **<table>**, потім йдуть фігурні дужки, в яких записується стильова властивість, а його значення вказується після двокрапки. Стильові властивості поділяються між собою крапкою з комою, вкінці цей символ можна опустити.

CSS не чутливий до регістру, перенесення рядків, прогалин і символів табуляції, тому форма запису залежить від бажання розробника. Так, в прикладі 1.14 показані два різновиди оформлення селекторів і їх правил.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.14. Використання стилів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Заголовки</title>  <style type="text/css">  H1 { color: #a6780a; font-weight: normal; } H2 {  color: olive;  border-bottom: 2px solid black;  }  </style>  </head>  <body>  <h1>Заголовок 1</h1>  <h2>Заголовок 2</h2>  </body>  </html> | | | | | | | | | |

В даному прикладі властивості селектора **H1** записані в один рядок, а для селектора **H2** кожна властивість знаходиться на окремому рядку. У другому випадку легше шукати потрібні властивості і редагувати їх за потребою, але при цьому незначно зростає обсяг даних за рахунок активного використання прогалин і переносів рядків. Тому в будь-якому випадку спосіб оформлення стильових правил залежить від розробника.

##### Правила застосування стилів

Далі наведені деякі правила, які необхідно знати при описі стилю.

###### Форма запису

Для селектора допускається додавати стильову властивість і її значення окремо, як це показано в прикладі 1.15.

Приклад 1.15. Розширена форма запису

td { background: olive; } td { color: white; }

td { border: 1px solid black; }

Однак такий запис не дуже зручний. Доводиться повторювати кілька разів один і той же селектор, та й легко заплутатися в їх кількості. Тому пишіть всі властивості для кожного селектора разом. Зазначений набір записів в такому випадку отримає наступний вигляд (приклад 1.16).

Приклад 1.16. Компактна форма запису

td {

background: olive; color: white;

border: 1px solid black;

}

Ця форма запису більш наочна і зручна у використанні.

Має пріоритет значення, вказане в коді нижче

Якщо для селектора спочатку задається властивість з одним значенням, а потім та ж властивість, але вже з іншим значенням, то застосовуватися буде те значення, яке в коді встановлено нижче (приклад 1.17).

Приклад 1.17. Різні значення у однієї властивості

P { color: green; } P { color: red; }

В даному прикладі для селектора **P** колір тексту спочатку задається зеленим, а потім червоним. Оскільки значення red розташоване нижче, то воно в підсумку і буде застосовуватися до тексту.

Насправді такого запису краще взагалі уникати і видаляти повторювані значення. Але подібне може статися, не відкрито, наприклад, в разі підключення різних стильових файлів, в яких містяться однакові селектори.

###### Значення

У кожної властивості може бути тільки відповідне її функції значення. Наприклад, для color, який встановлює колір тексту, як значення неприпустимо використовувати числа.

##### Коментарі

Коментарі потрібні, щоб робити пояснення з приводу використання тої чи іншої стильової властивості, виділяти розділи або писати свої нотатки. Коментарі дозволяють легко згадувати логіку і структуру селекторів, і підвищують розбірливість коду. Разом з тим, додавання тексту збільшує обсяг документів, що негативно позначається на часі їх завантаження. Тому коментарі зазвичай застосовують в налагоджувальних або навчальних цілях, а при публікуванні сайту в мережу їх видаляють.

Щоб позначити, що текст є коментарем, застосовують наступну конструкцію / \* ... \* / (приклад 1.18).

Приклад 1.18. Коментарі в CSS-файлі

/\*

Стиль для сайту htmlbook.ru Зроблено для ознайомлювальних цілей

\*/

div {

width: 200px; /\* Ширина контенту \*/

margin: 10px; /\* Поля навколо елемента \*/

float: left; /\* Обтікання по правому краю \*/

}

Як випливає з даного прикладу, коментарі можна додавати в будь-яке місце CSS-документа, а також писати текст коментаря в кілька рядків. Вкладені коментарі неприпустимі.

### Значення стильових властивостей

Все різноманіття значень стильових властивостей може бути зведене до певного типу: рядок, число, відсотки, розмір, колір, адреса або ключове слово.

##### Рядки

Будь-які рядки необхідно брати в подвійні або одинарні лапки. Якщо всередині рядка потрібно залишити одну або кілька лапок, то можна комбінувати типи лапок або додати перед лапками слеш (приклад 1.19).

Приклад 1.19. Допустимі рядки

'Готель "Турист"' "Готель 'Турист'" "Готель \"Турист\""

В даному прикладі в першому рядку застосовуються одинарні лапки, а слово «Турист» взято в подвійні лапки. В другому рядку все з точністю до навпаки, в третьому же рядку використовуються тільки подвійні лапки, але внутрішні екрановані за допомогою слеша.

##### Числа

Значенням може виступати ціле число, що містить цифри від 0 до 9 і десяткова дробу, в якій ціла і десяткова частина розділяються крапкою (приклад 1.20).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.20. Числа в якості значень | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Числа</title>  <style type="text/css"> P {  font-weight: 600; /\* Жирне накреслення \*/ line-height: 1.2; /\* Міжрядковий інтервал \*/  }  </style>  </head>  <body>  <p>Приклад текста</p>  </body>  </html> | | | | | | | | | |

Якщо в десятковому дробі ціла частина дорівнює нулю, то її дозволяється не писати. Запис .7 і 0.7 рівнозначний.

##### Відсотки

Відсотковий запис зазвичай застосовується в тих випадках, коли треба змінити значення щодо батьківського елементу або коли розміри залежать від зовнішніх умов. Так, ширина таблиці 100% означає, що вона буде підлаштовуватися під розміри вікна браузера і змінюватися разом з шириною вікна (приклад 1.21).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.21. Відсотковий запис | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Ширина в процентах</title>  <style type="text/css"> TABLE {  width: 100%; /\* Ширина таблиці у відсотках \*/ background: #f0f0f0; /\* Колір фону \*/  }  </style>  </head>  <body>  <table>  <tr><td>Вміст таблиці </td></tr>  </table>  </body>  </html> | | | | | | | | | |

Відсотки не обов'язково повинні бути цілим числом, допускається використовувати десяткові дроби, на зразок значення 56.8%, але не завжди.

##### Розміри

Для завдання розмірів різних елементів, в CSS використовуються абсолютні і відносні одиниці виміру.

Абсолютні одиниці не залежать від пристрою виведення, а відносні одиниці визначають розмір елемента щодо значення іншого розміру.

###### Відносні одиниці

Відносні одиниці зазвичай використовують для роботи з текстом, або коли треба обчислити відсоткове співвідношення між елементами. У табл. 1.2 перераховані основні відносні одиниці.

*Табл. 1.2. Відносні одиниці вимірювання*

|  |  |
| --- | --- |
| Одиниця | Опис |
| em | Розмір шрифту поточного елементу |
| ex | Висота символу x |
| px | Піксель |
| % | Відсоток |

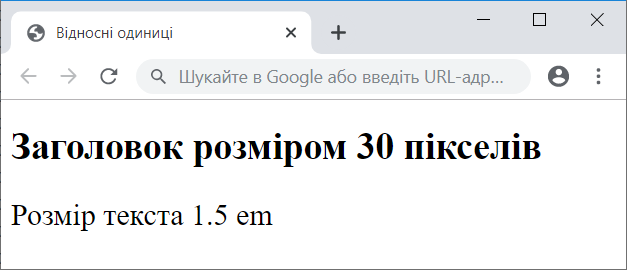
Одиниця em це змінне значення, яке залежить від розміру шрифту поточного елементу (розмір встановлюється через стильову властивість font-size). В кожному браузері закладений розмір тексту, який застосовується в тому випадку, коли цей розмір явно не заданий. Тому спочатку 1em дорівнює розміру шрифту, заданого в браузері за замовчуванням або розміром шрифту батьківського елементу. Відсотковий запис ідентичний em, в тому сенсі, що значення 1em і 100% рівні.

Одиниця ex визначається як висота символу «x» в нижньому регістрі. На ex поширюються ті ж правила, що і для em, а саме, він прив'язаний до розміру шрифту, заданого в браузері за замовчуванням, або до розміру шрифту батьківського елементу.

Піксель це елементарна точка, яка відображається монітором або іншим подібним пристроєм, наприклад, смартфоном. Розмір пікселя залежить від дозволу пристрою і його технічних характеристик. У прикладі 1.22 показано застосування пікселів і em для завдання розміру шрифту.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.22. Використання відносних одиниць | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Відносні одиниці</title>  <style type="text/css"> H1 { font-size: 30px; } P { font-size: 1.5em; }  </style>  </head>  <body>  <h1>Заголовок розміром 30 пікселів</h1>  <p>Розмір текста 1.5 em</p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний нижче (рис. 1.10).



###### *Рис. 1.10. Розмір тексту при різних одиниця*

###### 

###### Абсолютні одиниці

Абсолютні одиниці застосовуються рідше, ніж відносні і зазвичай при роботі з текстом. У табл. 1.3 перераховані основні абсолютні одиниці.

*Табл. 1.3. Абсолютні одиниці вимірювання*



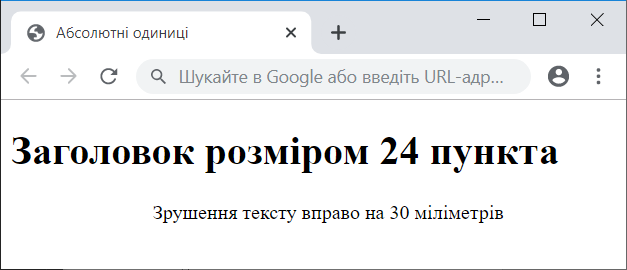
|  |  |
| --- | --- |
| Одиниця | Опис |
| In | Дюйм (1 дюйм дорівнює 2,54 см) |
| Cm | Сантиметр |
| Mm | Міліметр |
| Pt | Пункт (1 пункт дорівнює 1/72 дюйма) |
| Pc | Піка (1 піку дорівнює 12 пунктам) |



Найбільш, мабуть, найпоширенішою одиницею є пункт, який використовується для вказівки розміру шрифту. Хоча ми звикли вимірювати все в міліметрах і подібних одиницях, пункт, мабуть, єдина величина з не метричної системи вимірювання, яка використовується у нас повсюдно. І все завдяки текстовим редакторам і видавничим системам. У прикладі 1.23 показано використання пунктів і міліметрів.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.23. Використання абсолютних одиниць | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Абсолютні одиниці</title>  <style type="text/css"> H1 { font-size: 24pt; } P { margin-left: 30mm; }  </style>  </head>  <body>  <h1>Заголовок розміром 24 пункта</h1>  <p>Зрушення тексту вправо на 30 міліметрів</p>  </body>  </html> | | | | | | | | | |

Результат використання абсолютних одиниць вимірювання показаний нижче (рис. 1.11).



*Рис. 1.11. Розмір тексту при різних одиницях*

При установці розмірів обов'язково вказуйте одиниці виміру, наприклад width: 30px. В іншому випадку браузер не зможе показати бажаний результат, оскільки не розуміє, який розмір вам потрібно. Одиниці не додаються тільки при нульовому значенні (margin: 0).

##### Колір

Колір в стилях можна задавати різними способами: по шістнадцятирічним значенням, за назвою, в форматі RGB, RGBA, HSL, HSLA.

###### За шістнадцятковим значенням

Для задання кольорів використовуються числа в шістнадцятковому коді. Шістнадцяткова система, на відміну від десяткової системи, базується, як випливає з її назви, на числі 16. Цифри будуть наступні: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Цифри від 10 до 15 замінені латинськими буквами. Числа більше 15 в шістнадцятковій системі утворюються об'єднанням двох чисел в одне. Наприклад, числу 255 в десятковій системі відповідає число FF в шістнадцятковій системі. Щоб не виникало плутанини у визначенні системи числення, перед шістнадцятковим числом ставлять символ грат #, наприклад # 666999. Кожен з трьох кольорів - червоний, зелений і синій - може набувати значень від 00 до FF. Таким чином, позначення кольору розбивається на три складові #rrggbb, де перші два символи відзначають червоний компонент кольору, два середніх - зелену, а два останніх - синю. Допускається використовувати скорочену форму виду #rgb, де кожен символ слід подвоювати. Так, запис # fe0 слід розцінювати як # ffee00

###### За назвою

Браузери підтримують деякі кольори по їх назві. У табл. 1.4 наведені назви, шістнадцятковий код, значення в форматі RGB, HSL і опис.

*Табл. 1.4. Назви кольорів*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ім’я | Колір | Код | RGB | HSL | Опис |
| white |  | #ffffff или #fff | rgb(255,255,255) | hsl(0,0%,100%) | Білий |
| silver |  | #c0c0c0 | rgb(192,192,192) | hsl(0,0%,75%) | Сірий |
| gray |  | #808080 | rgb(128,128,128) | hsl(0,0%,50%) | Темно-сірий |
| black |  | #000000 или #000 | rgb(0,0,0) | hsl(0,0%,0%) | Чорний |
| maroon |  | #800000 | rgb(128,0,0) | hsl(0,100%,25%) | Темно-червоний |
| red |  | #ff0000 или #f00 | rgb(255,0,0) | hsl(0,100%,50%) | Червоний |
| orange |  | #ffa500 | rgb(255,165,0) | hsl(38.8,100%,50%) | Помаранчевий |
| yellow |  | #ffff00 или #ff0 | rgb(255,255,0) | hsl(60,100%,50%) | Жовтий |
| olive |  | #808000 | rgb(128,128,0) | hsl(60,100%,25%) | Оливковий |
| lime |  | #00ff00 или #0f0 | rgb(0,255,0) | hsl(120,100%,50%) | Світло-зелений |
| green |  | #008000 | rgb(0,128,0) | hsl(120,100%,25%) | Зелений |
| aqua |  | #00ffff или #0ff | rgb(0,255,255) | hsl(180,100%,50%) | Блакитний |
| blue |  | #0000ff или #00f | rgb(0,0,255) | hsl(240,100%,50%) | Синій |
| navy |  | #000080 | rgb(0,0,128) | hsl(240,100%,25%) | Темно-синій |
| teal |  | #008080 | rgb(0,128,128) | hsl(180,100%,25%) | Синьо-зелений |
| fuchsia |  | #ff00ff или #f0f | rgb(255,0,255) | hsl(300,100%,50%) | Рожевий |
| purple |  | #800080 | rgb(128,0,128) | hsl(300,100%,25%) | Фіолетовий |

###### За допомогою RGB

Можна визначити колір, використовуючи значення червоної, зеленої та синьої складової в десятковому численні. Кожна з трьох складових кольору приймає значення від 0 до 255. Також допустимо ставити колір в відсотковому відношенні, при цьому 100% буде відповідати числу 255. Спочатку вказується ключове слово rgb, а потім в дужках, через кому вказуються компоненти кольору, наприклад rgb (255, 128, 128) або rgb (100%, 50%, 50%).

###### RGBA

Формат RGBA схожий по синтаксису на RGB, але включає в себе альфа-канал, що задає прозорість елемента. Значення 0 відповідає повній прозорості, 1 - непрозорості, а проміжне значення на зразок 0.5 - напівпрозорості.

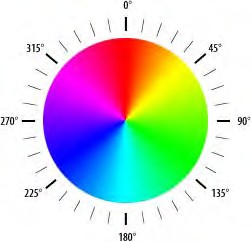
RGBA доданий в CSS3, тому валідацію CSS-коду треба проводити саме за цією версією. Слід зазначити, що стандарт CSS3 ще знаходиться в розробці і деякі можливості в ньому можуть помінятися. Наприклад, колір в форматі RGB доданий до властивості background-color проходить валідацію, а доданий до властивості background вже немає. При цьому браузери цілком коректно розуміють колір для того і іншого властивості. У табл. 1.4 наведено список браузерів, які підтримують RGBA; в браузерах, де цей формат не працює, значення кольору буде ігноруватися.

*Табл. 1.4. Підтримка браузерами формату RGBA*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Internet Explorer | | | | Chrome | | | | Opera | | | | Safari | | | Firefox | | | |
| 6.0 | 7.0 | 8.0 | 9.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.2 | 9.6 | 10 | 11 | 3.1 | 4.0 | 5.0 | 2.0 | 3.0 | 3.6 | 4.0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

###### HSL

Назва формату HSL утворено від сполучення перших букв Hue (відтінок), Saturate (насиченість) і Lightness (світлини). Відтінок це значення кольору на колірному колі (рис. 1.12) і задається в градусах. 0 ° відповідає червоному кольору, 120 ° - зеленому, а 240 ° - синього. Значення відтінку може змінюватися від 0 до 359.



*Рис. 1.12. Кольоровий круг*

Насиченістю називається інтенсивність кольору, вимірюється у відсотках від 0% до 100%. Значення 0% означає відсутність кольору і відтінок сірого, 100% максимальне значення насиченості.

Прозорість задає, наскільки колір яскравий і вказується у відсотках від 0% до 100%. Малі значення роблять колір темніше, а високі світліше, крайні значення 0% і 100% відповідають чорному і білому кольору.

Приклади використання HSL наведені в табл. 1.4. За винятком Internet Explorer все сучасні браузери коректно працюють з цим форматом (табл. 1.5).

*Табл. 1.5. Підтримка браузерами формату HSL*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Internet Explorer | | | | Chrome | | | | Opera | | | | Safari | | | Firefox | | | |
| 6.0 | 7.0 | 8.0 | 9.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.2 | 9.5 | 10 | 11 | 3.1 | 4.0 | 5.0 | 2.0 | 3.0 | 3.6 | 4.0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

###### HSLA

Формат HSLA схожий по синтаксису на HSL, але включає в себе альфа-канал, що задає прозорість елемента. Значення 0 відповідає повній прозорості, 1 - непрозорості, а проміжне значення на зразок 0.5 - напівпрозорості. У порівнянні з HSL підтримка браузерами трохи інша (табл. 1.6).

*Табл. 1.6. Підтримка браузерами формату HSLA*

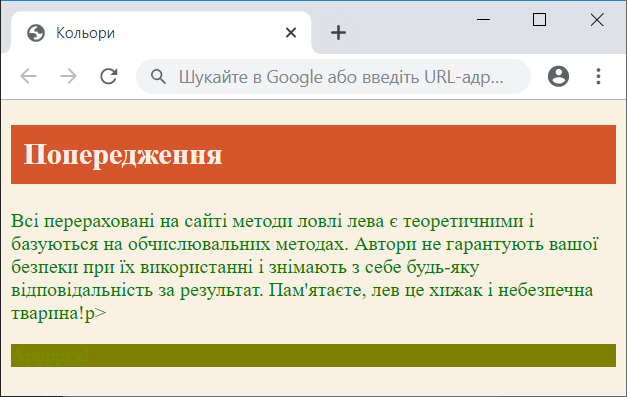
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Internet Explorer | | | | Chrome | | | | Opera | | | | Safari | | | Firefox | | | |
| 6.0 | 7.0 | 8.0 | 9.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.2 | 9.6 | 10 | 11 | 3.1 | 4.0 | 5.0 | 2.0 | 3.0 | 3.6 | 4.0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Значення кольору в форматах RGBA, HSL і HSLA додані в CSS3, тому при використанні цих форматів перевіряйте код на валідність з урахуванням версії.

В прикладі 1.24 представлені різні способи завдання кольорів елементів веб-сторінок.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.24. Подання кольорів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Кольори</title>  <style type="text/css"> BODY {  background-color: #F9F2E3;  }  H2 {  background-color: rgb(214,86,43); color: rgba(255,255,255,.9); padding: 10px;  }  P {  color: green;  } DIV {  background-color: hsl(60,100%,25%); color: hsla(120,100%,50%,0.1);  }  </style>  </head>  <body>  <h2> Попередження </h2>  <p> Всі перераховані на сайті методи ловлі лева є теоретичними і базуються на обчислювальних методах. Автори не гарантують вашої безпеки при їх використанні і знімають з себе будь-яку відповідальність за результат. Пам'ятаєте, лев це хижак і  небезпечна тварина!p>  <div>Арррргх!</div>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.13.



*Рис. 1.13. Кольори на веб-сторінці*

##### Адреса

Адреси застосовуються для вказівки шляху до файлу, наприклад, для установки фонової картинки на сторінці. Для цього застосовується ключове слово url(), всередині дужок пишеться відносний або абсолютний адресу файлу. При цьому адреси можна задавати в необов'язкових одинарних або подвійних лапках (приклад 1.25).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.25. Адреса графічного файлу | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title> Додавання фону </title>  <style type="text/css"> BODY {  background: [url('http://webimg.ru/images/156\_1.png')](http://webimg.ru/images/156_1.png%27)) no-repeat;  } DIV {  background: url(images/warning.png) no-repeat; padding-left: 140px;  }  </style>  </head>  <body>  <div> Увага, запитувана сторінка не знайдена!</div>  </body>  </html> | | | | | | | | | |

В даному прикладі в селекторі BODY використовується абсолютна адреса до графічного файлу, а в селекторі DIV - відносна.

##### Ключові слова

Як значення активно застосовуються ключові слова, які визначають бажаний результат дії стильових властивостей. Ключові слова пишуться без лапок.

Правильно: P { text-align: right; } Невірно: P { text-align: "right"; }

##### inherit

Ключове слово, яке повідомляє, що необхідно наслідувати значення властивості у батьківського елементу. Природно, результат буде помітний лише в тому випадку, якщо у батьків вказана властивість встановлено. Браузер Internet Explorer до версії 7.0 включно не підтримує значення inherit.

##### initial

Значення initial застосовується для установки початкового значення властивості. Може стати в нагоді в декількох випадках, наприклад, відновити значення властивостей, заданих браузером за замовчуванням або задати початкове значення властивості, змінене в результаті успадкування. Ключове слово initial додано в CSS3 і поки погано підтримується браузерами (табл. 1.7).

*Табл. 1.7. Підтримка браузерами значення initial*

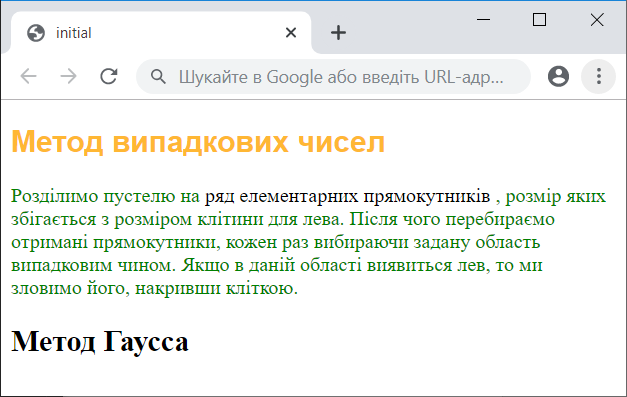
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Браузер | Internet Explorer | Chrome | Opera | Safari | Firefox |
| Версія | — | 2.0+ | — | 2.0+ | 1.0+ |
| Значення | — | Initial | — | initial | -moz-initial |

Значення -moz-initial є нестандартним, тому його застосування призведе до невалідность коду CSS.

В прикладі 1.26 показані деякі аспекти застосування initial.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.26. Використання initial | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>initial</title>  <style type="text/css"> H2 {  color: #ffb734;  font-family: Arial, sans-serif;  }  P {  color: green;  }  .initial { color: initial;  color: -moz-initial; font-family: initial;  font-family: -moz-initial;  }  </style>  </head>  <body>  <h2>Метод випадкових чисел </h2>  <p> Розділимо пустелю на <span class = "initial"> ряд елементарних прямокутників </span>, розмір яких збігається з розміром клітини для лева. Після чого перебираємо отримані прямокутники, кожен раз вибираючи задану область випадковим чином.  Якщо в даній області виявиться лев, то ми зловимо його, накривши кліткою.</p>  <h2 class="initial">Метод Гаусса</h2>  </body>  </html> | | | | | | | | | |

В даному прикладі змінюється колір тексту і шрифт заголовку. Колір тексту всередині тега <span> не задається, але він успадковує значення кольору свого батька - тега <p>. За допомогою класу initial колір фрагмента тексту встановлюється вихідним, за замовчуванням він чорний. Аналогічно йде справа і з заголовком, через стилі задається його колір і шрифт. Щоб відновити стиль за замовчуванням, до тегу <h1> додається клас initial, в якому використовується значення initial (рис. 1.14).



*Рис. 1.14. Результат виконання initial*

### Селектори тегів

В якості селектора може виступати будь-який тег HTML для якого визначаються правила форматування, такі як: колір, фон, розмір і т.д. Правила задаються в наступному виді.

Тег { властивість1: значення; властивість 2: значення; ... }

Спочатку вказується ім'я тега, оформлення якого буде перевизначити, великими або малими символами не має значення. Усередині фігурних дужок пишеться стильова властивість, а після двокрапки - його значення. Набір властивостей поділяється між собою крапкою з комою і може розташовуватися як в один рядок, так і в кілька (приклад 1.27).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.27. Зміна стилю тега <p> | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селекторы тегов</title>  <style type="text/css"> P {  text-align: justify; /\* Вирівнювання по ширині \*/ color: green; /\* Зелений колір текста \*/  }  </style>  </head>  <body>  <p> Більш ефективним способом лову лева в пустелі  є метод золотого перетину. При його використанні пустеля ділиться на дві нерівні частини, розмір яких підпорядковується правилу золотого перетину.</p>  </body>  </html> | | | | | | | | | |

В даному прикладі змінюється колір тексту і вирівнювання тексту абзацу. Стиль буде застосовуватися тільки до тексту, який розташовується усередині контейнера <p>.

Слід розуміти, що хоча стиль можна застосувати до будь-якого тегу, результат буде помітний лише для тегів, які безпосередньо відображаються в контейнері **<body>**.

### 

### Класи

Класи застосовують, коли необхідно визначити стиль для індивідуального елемента веб-сторінки або задати різні стилі для одного тега. При використанні спільно з тегами синтаксис для класів буде наступний.

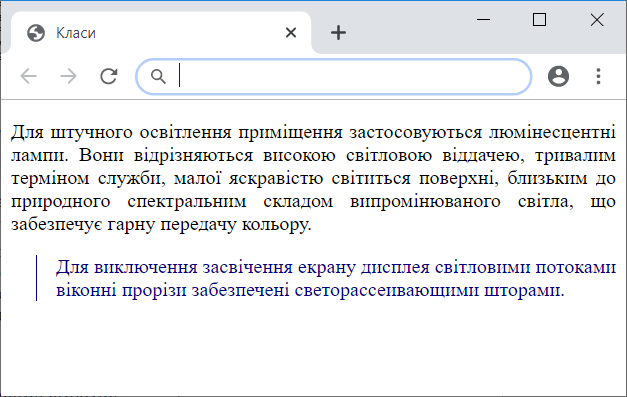
Тег.Ім’я класа { властивість: значення; властивості 2: значення; ... }

Усередині стилю спочатку пишеться бажаний тег, а потім, через точку для користувача ім'я класу. Імена класів повинні починатися з латинського символу і можуть містити в собі символ дефіса (-) і підкреслення (\_).

Використання укаїнських букв в іменах класів неприпустимо. Щоб вказати в коді HTML, що тег використовується з певним класом, до тегу додається атрибут class значенням якого виступає ім'я класу (приклад 1.28).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.28. Використання класів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Класи</title>  <style type="text/css">  P { /\* Звичайний абзац \*/  text-align: justify; /\* Вирівнювання тексту по ширині \*/  }  P.cite { /\* Абзац з класом cite \*/ color: navy; /\* Колір текста \*/  margin-left: 20px; /\* Відступ зліва \*/  border-left: 1px solid navy; /\* Кордон зліва від тексту \*/ padding-left: 15px; /\* Відстань від лінії до тексту \*/  }  </style>  </head>  <body>  <p> Для штучного освітлення приміщення застосовуються люмінесцентні лампи. Вони відрізняються високою світловою віддачею, тривалим терміном служби, малої яскравістю світиться поверхні, близьким до природного спектральним складом випромінюваного світла, що забезпечує гарну передачу кольору. </P>  <P class = "cite"> Для виключення засвічення екрану дисплея світловими потоками віконні прорізи забезпечені светорассеивающими шторами.  </p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.15.



*Рис. 1.15. Вигляд тексту, оформленого за допомогою стильових класів*

Перший абзац вирівняний по ширині з текстом чорного кольору (цей колір задається браузером за замовчуванням), а наступний, до якого застосовано клас з ім'ям cite — відображається синім кольором і з лінією зліва.

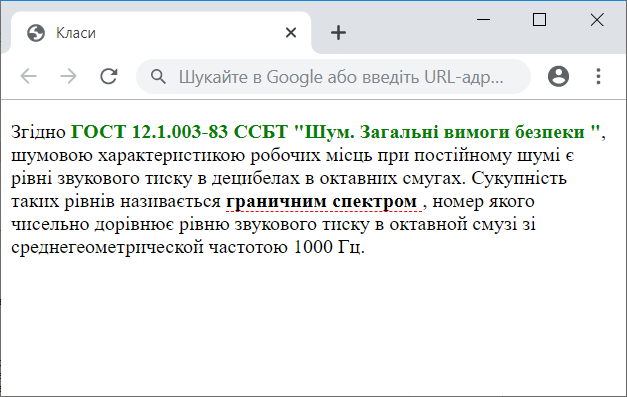
Можна, також, використовувати класи і без вказівки тега. Синтаксис в цьому випадку буде наступний.

.Ім’я класу {властивість: значення; властивості 2: значення; ... }

При такому записі, клас можна застосовувати до будь-якого тегу (приклад 1.29).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.29. Використання класів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Класи</title>  <style type="text/css">  .gost {  color: green; /\* Колір текста \*/  font-weight: bold; /\* Жирне накреслення \*/  }  .term {  border-bottom: 1px dashed red; /\* Підкреслення під текстом \*/  }  </style>  </head>  <body>  <p>Згідно <span class="gost">ГОСТ 12.1.003-83 ССБТ &quot;Шум. Загальні вимоги безпеки &quot;</span>, шумовою характеристикою робочих місць при постійному шумі є рівні звукового тиску в децибелах в октавних смугах. Сукупність таких рівнів називається  <b class="term"> граничним спектром </b>, номер якого чисельно дорівнює рівню звукового тиску в октавной смузі зі среднегеометрической частотою 1000&nbsp;Гц.</p>  </body>  </html> | | | | | | | | | |

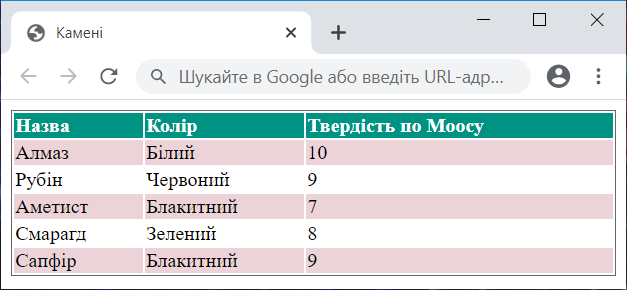
Результат застосування класів по тегам <span> і <b> показаний на рис. 1.16.



*Мал. 1.16. Вигляд тегів, оформлених за допомогою класів*

Класи зручно використовувати, коли потрібно застосувати стиль до різних елементів веб-сторінки: клітинок таблиці, посиланням, абзаців та ін. В прикладі 1.30 показано зміна кольору фону рядків таблиці для створення «зебри».

Результат даного прикладу показаний на рис. 1.17. У прикладі клас з ім'ям odd використовується для зміни кольору фону рядка таблиці. За рахунок того, що цей клас додається не до всіх тегів <tr> і виходить чергування різних кольорів.



*Рис. 1.17. Результат застосування класів*

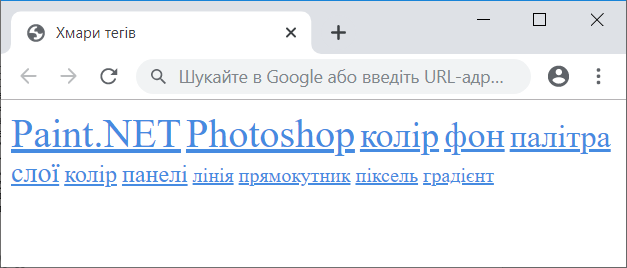
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.30. Використання класів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Камені</title>  <style type="text/css"> table.jewel {width: 100%; /\* Ширина таблиці \*/  border: 1px solid #666; /\* Рамка навколо таблиці \*/}  th {background: #009383; /\*Колір фона \*/ color: #fff; /\* Колір текста \*/  text-align: left; /\* Вирівнювання по лівому краю \*/}  tr.odd {background: #ebd3d7; /\* Колір фона \*/}  </style>  </head>  <body>  <table class="jewel">  <tr>  <th>Назва</th><th>Колір</th><th>Твердість по Моосу</th>  </tr>  <tr class="odd">  <td>Алмаз</td><td>Білий</td><td>10</td>  </tr>  <tr>  <td>Рубін</td><td>Червоний</td><td>9</td>  </tr>  <tr class="odd">  <td>Аметист</td><td>Блакитний</td><td>7</td>  </tr>  <tr>  <td>Смарагд</td><td>Зелений</td><td>8</td>  </tr>  <tr class="odd">  <td>Сапфір</td><td>Блакитний</td><td>9</td>  </tr>  </table>  </body>  </html> | | | | | | | | | |

##### Одночасне використання різних класів

До будь-якого тегу можна одночасно додати кілька класів, перераховуючи їх в значенні атрибута class через пробіл, така форма називається мультикласи. В цьому випадку до елементу застосовується стиль, описаний в правилах для кожного класу. Оскільки при додаванні декількох класів вони можуть містити однакові стильові властивості, але з різними значеннями, то береться значення у класу, який описаний в коді нижче. У прикладі 1.31 показано використання різних класів для створення хмар тегів.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.31. Поєднання різних класів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| !DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Хмари тегів </title>  <style type="text/css">  .level1 { font-size: 1em; }  .level2 { font-size: 1.2em; }  .level3 { font-size: 1.4em; }  .level4 { font-size: 1.6em; }  .level5 { font-size: 1.8em; }  .level6 { font-size: 2em; } A.tag {color: #468be1; /\*Колір силок \*/}  </style>  </head>  <body>  <div>  <a href="/term/2" class="tag level6">Paint.NET</a>  <a href="/term/69" class="tag level6">Photoshop</a>  <a href="/term/3" class="tag level5">колір</a>  <a href="/term/95" class="tag level5">фон</a>  <a href="/term/11" class="tag level4">палітра</a>  <a href="/term/43" class="tag level3">слої</a>  <a href="/term/97" class="tag level2">колір</a>  <a href="/term/44" class="tag level2">панелі</a>  <a href="/term/16" class="tag level1">лінія</a>  <a href="/term/33" class="tag level1">прямокутник</a>  <a href="/term/14" class="tag level1">піксель</a>  <a href="/term/27" class="tag level1">градієнт</a>  </div>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.18.



*Рис. 1.18. Хмара тегів*

В стилях також допускається використовувати запис вигляду .layer1 .layer2, де layer1 і layer2 являють собою імена класів. Стиль застосовується тільки для елементів, у яких одночасно задані класи layer1 і layer2 (приклад 1.32).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.32. Використання мультикласів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Мультікласи</title>  <style type="text/css">  .layer1 { color: red; }  .layer2 { color: blue; }  .layer1.layer2 {color: green; }  </style>  </head>  <body>  <p class="layer1">Текст червогого кольору</p>  <p class="layer2">Текст синього кольору</p>  <p class="layer1 layer2">Текст зеленого кольору</p>  </body>  </html> | | | | | | | | | |

Браузер IE 6 некоректно працює з мультикласами і розуміє запис .a .b як .b, тобто сприймає тільки ім'я останнього класу, що призводить до помилок в роботі даного прикладу. Текст з класом layer2 буде відображатися зеленим кольором, а не синім, як повинно бути.

### Ідентифікатори

Ідентифікатор (званий також «ID селектор») визначає унікальне ім'я елемента, яке використовується для зміни його стилю і звернення до нього через скрипти.

Синтаксис застосування ідентифікатора наступний.

#Ім'я ідентифікатора {властивість: значення; властивості: значення; ...}

При описі ідентифікатора спочатку вказується символ решітки (#), потім йде ім'я ідентифікатора. Воно повинно починатися з латинського символу і може містити в собі символ дефіса (-) і підкреслення (\_). Використання українських букв в іменах ідентифікатора неприпустиме. На відміну від класів ідентифікатори повинні бути унікальні, іншими словами, зустрічатися в коді документа лише один раз.

Звернення до ідентифікатора відбувається аналогічно класам, але як атрибут тега використовується id, значенням якого виступає ім'я ідентифікатора (приклад 1.33). Символ решітки при цьому вже не вказується.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.33. Використання ідентифікатора | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Ідентифікатори</title>  <style type="text/css"> #help {  position: absolute; /\* Абсолютне позиціонування \*/ right: 20px; /\*Положення від правого краю\*/  top: 50px; /\*Положення від верхнього краю\*/ width: 225px; /\* Ширина блока \*/  padding: 5px; /\*Поля навколо тексту \*/ background: #f0f0f0; /\* Колір фона \*/ display: none; /\* Приховати \*/  }  </style>  </head>  <body>  <p><a href="#" onclick="document.getElementById('help').style.display='block'">Довідка</a></p>  <div id="help">  Ця довідка допомагає в разі, коли ви знаходитесь в усвідомленні того факту, що абсолютно не розумієте, хто і як вам може допомогти. Саме в цей момент ми і підказуємо, що допомогти вам ніхто не зможе.  </div>  </body>  </html> | | | | | | | | | |

В даному прикладі визначається стиль тега <div> через ідентифікатор з ім'ям help. Початково цей шар приховується від перегляду і робиться видимим при натисканні на посилання, що викликає невеликий скрипт через подію onclick.

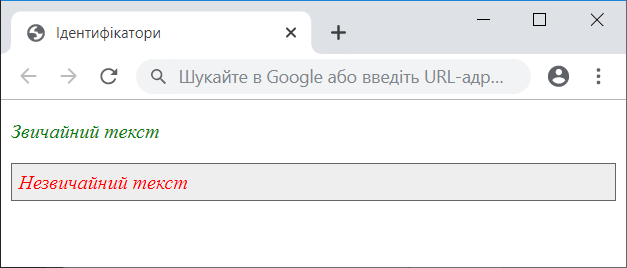
Як і при використанні класів, ідентифікатори можна застосовувати до конкретного тегу. Синтаксис при цьому буде наступний.

Тег# Ім'я ідентифікатора {властивість: значення; властивості: значення; ... }

Спочатку вказується ім'я тега, потім без пробілів символ решітки та назва ідентифікатора. У прикладі 1.34 показано використання ідентифікатора стосовно тегу **<p>**.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.34. Застосування ідентифікатора спільно з тегом | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Ідентифікатори</title>  <style type="text/css"> P {  color: green; /\* Зелений колір текста \*/  font-style: italic; /\* Курсив накреслення текста \*/  }  P#opa {  color: red; /\* Червоний колір текста \*/  border: 1px solid #666; /\* Параметри рамкі \*/ background: #eee; /\* Колір фона \*/  padding: 5px; /\* Поля вокруг текста \*/  }  </style>  <script type="text/javascript"> function newText() {  document.getElementById("opa").innerHTML = 'Незвичайний текст';  }  </script>  </head>  <body onload="newText()">  <p>Звичайний текст</p>  <p id="opa"></p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.19.



*Рис. 1.19. Вигляд веб-сторінки*

В даному прикладі вводиться ідентифікатор з ім’ям opa, який застосовується до тегу **<p>**. У функції ідентифікатора входить зміна стилю абзацу та висновок в нього тексту через скрипт. Саме це завдання покладено на метод innerHTML.

Контекстні селектори

При створенні веб-сторінки часто доводиться вкладати одні теги всередину інших. Щоб стилі для цих тегів використовувалися коректно, допоможуть селектори, які працюють тільки в певному контексті. Наприклад, задати стиль для тега **<b>** тільки коли він розташовується всередині контейнера **<p>**. Таким чином можна одночасно встановити стиль для окремого тега, а також для тега, який знаходиться всередині іншого.  
  
Контекстний селектор складається з простих селекторів розділених пропуском. Так, для селектора тега синтаксис буде наступний.

Тег1 Тег2 {...}

В цьому випадку стиль буде застосовуватися до Тегу2 коли він розміщується всередині Тега1, як показано нижче.

<Тег1>

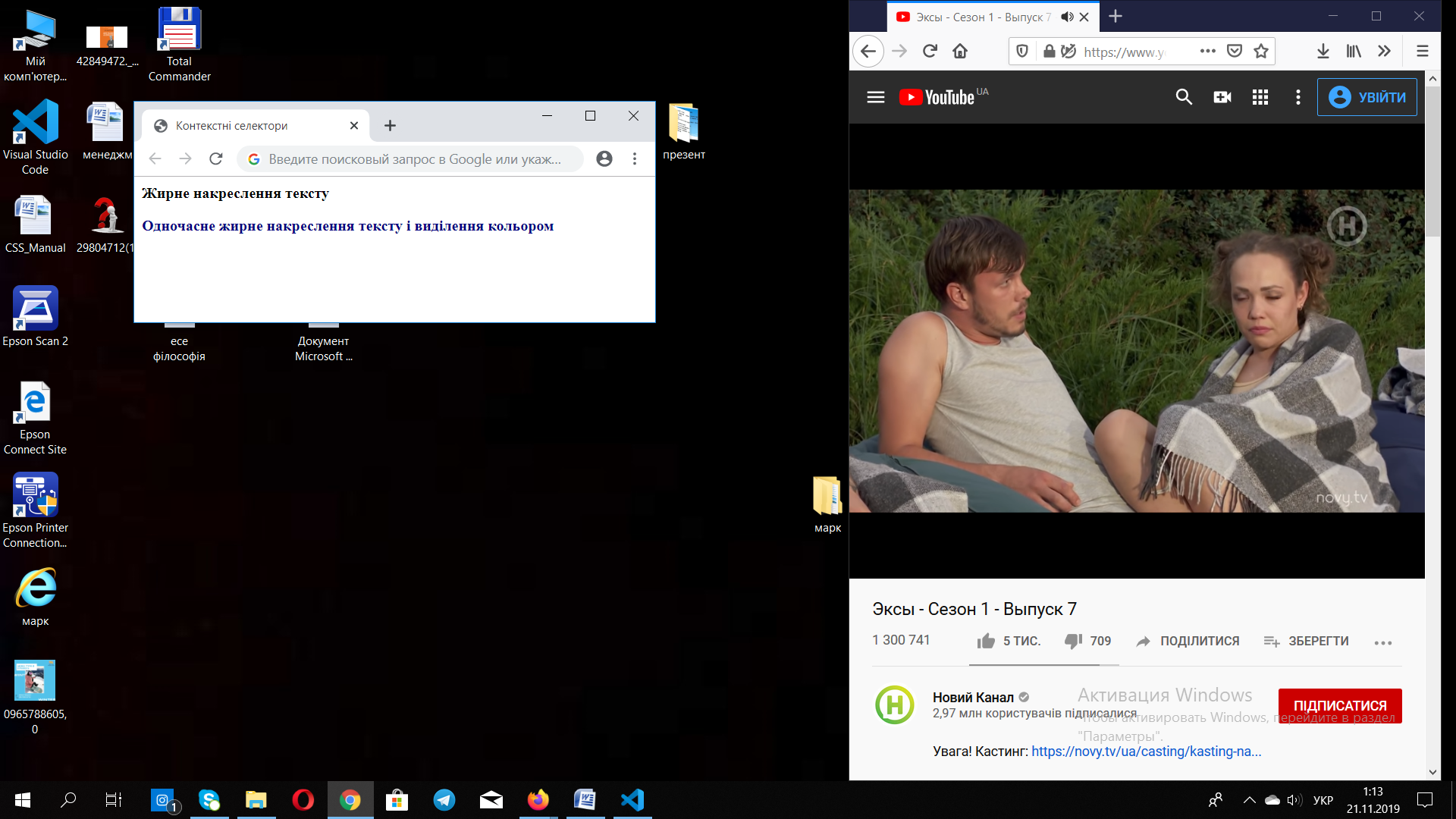
<Тег2> ... </Тег2>

</Тег1>

Використання контекстних селекторів продемонстровано в прикладі 1.35.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.35. Контекстні селектори | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Контекстні селектори</title>  <style type="text/css"> P B {  font-family: Times, serif; /\* Сімейство шрифту \*/ font-weight: bold; /\* Жирне накреслення \*/  color: navy; /\* Синій колір тексту \*/  }  </style>  </head>  <body>  <div><b>Жирне накреслення тексту</b></div>  <p><b>Одночасне жирне накреслення тексту і виділення кольором</b></p>  </body>  </html> | | | | | | | | | |

В даному прикладі показано звичайне застосування тега <b> і цього ж тега, коли він вкладений всередину абзацу <p>. При цьому змінюється колір і шрифт тексту, як показано на рис. 1.19.



*Рис. 1.19. Оформлення тексту в залежності від вкладеності тегів*

 Не обов'язково контекстні селектори містять тільки один вкладений тег. Залежно від ситуації допустимо застосовувати два і більш послідовно вкладених один в одного тегів.

Більш ширші можливості контекстні селектори дають при використанні ідентифікаторів і класів. Це дозволяє встановлювати стиль тільки для того елемента, який розташовується усередині певного класу, як показано в прикладі 10.2.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.36. Використання класів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Контекстні селектори</title>  <style type="text/css"> A {color: green; /\* Зелений колір тексту для всіх посилань \*/}  .menu {  padding: 7px; /\* Поля навколо тексту \*/ border: 1px solid #333; /\* Параметри рамки \*/ background: #fc0; /\* Колір фону \*/  }  .menu A {  color: navy; /\* Темно-синій колір посилання \*/}  </style>  </head>  <body>  <div class="menu">  <a href="1.html">Російська кухня</a> |  <a href="2.html">Українська кухня</a> |  <a href="3.html">Кавказька кухня</a>  </div>  <p><a href="text.html">Інші матеріали по темі</a></p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показано на рис. 1.20.



*Рис. 1.20. Посилання різних кольорів*

В даному прикладі використовується два типи посилань. Перше посилання, стиль якої задається за допомогою селектора A, буде діяти на всій сторінці, а стиль другого посилання (.menu A) застосовується тільки до посилань всередині елемента з класом menu.

При такому підході легко керувати стилем однакових елементів, на зразок зображень і посилань, оформлення яких має відрізнятися в різних областях веб-сторінки.

Сусідні селектори

Сусідніми називаються елементи веб-сторінки, коли вони йдуть безпосередньо один за одним в коді документа. Розглянемо кілька прикладів відносин елементів.

<p>Lorem ipsum <b>dolor</b> sit amet.</p>

У цьому прикладі тег **<b>** є дочірнім по відношенню до тегу **<p>**, оскільки він знаходиться всередині цього контейнера. Відповідно **<p>** виступає в якості батька **<b> .**

<p>Lorem ipsum <b>dolor</b> <var>sit</var> amet.</p>

Тут теги **<var>** і **<b>** ніяк не перекриваються і являють собою сусідні елементи. Те, що вони розташовані всередині контейнера **<p>**, ніяк не впливає на їхнє ставлення.

<p>Lorem <b>ipsum </b> dolor sit amet, <i>consectetuer</i> adipiscing <tt>elit</tt>.</p>

Сусідніми тут є теги **<b>** і **<i>**, а також **<i>** і **<tt>**. При цьому **<b>** і **<tt>** до сусідніх елементів не належать через те, що між ними розташований контейнер **<i>**.

Для управління стилем сусідніх елементів використовується символ плюса (+), який встановлюється між двома селекторами. Загальний синтаксис наступний.

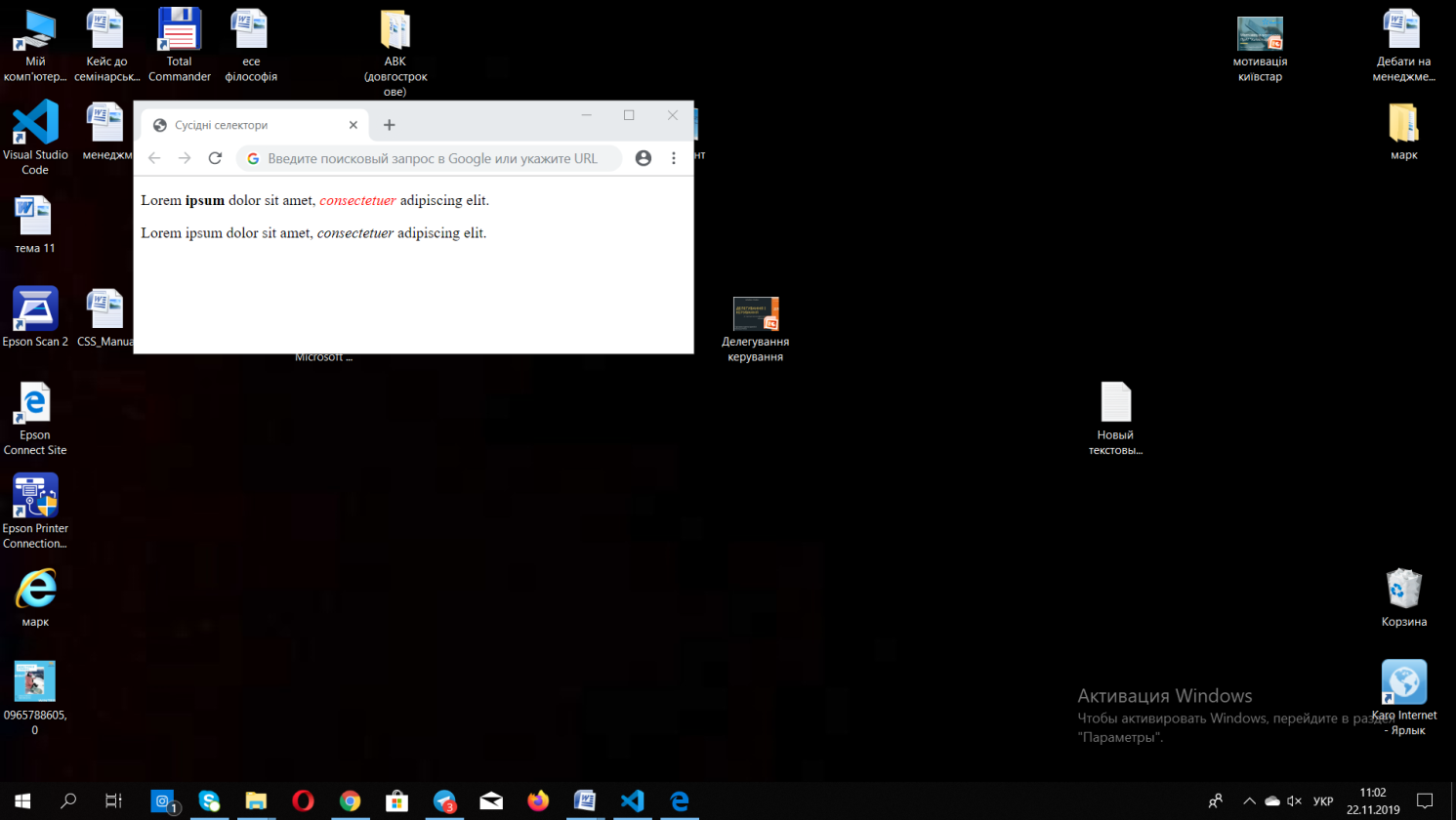
Селектор 1 + Селектор 2 { Опис правил стилю }

Прогалини навколо плюса не обов'язкові, стиль при такому записі застосовується до селектора 2, але тільки в тому випадку, якщо він є сусіднім для селектора 1 і слідує відразу після нього.

У прикладі 1.37 показана структура взаємодії тегів між собою.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.37. Використання сусідніх селекторів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Сусідні селектори</title>  <style type="text/css"> B + I {  color: red; /\* Червоний колір тексту \*/  }  </style>  </head>  <body>  <p>Lorem <b>ipsum </b> dolor sit amet, <i>consectetuer</i> adipiscing elit.</p>  <p>Lorem ipsum dolor sit amet, <i>consectetuer</i> adipiscing elit.</p>  </body>  </html> | | | | | | | | | |

Результат прикладу показано на рис. 1.22.



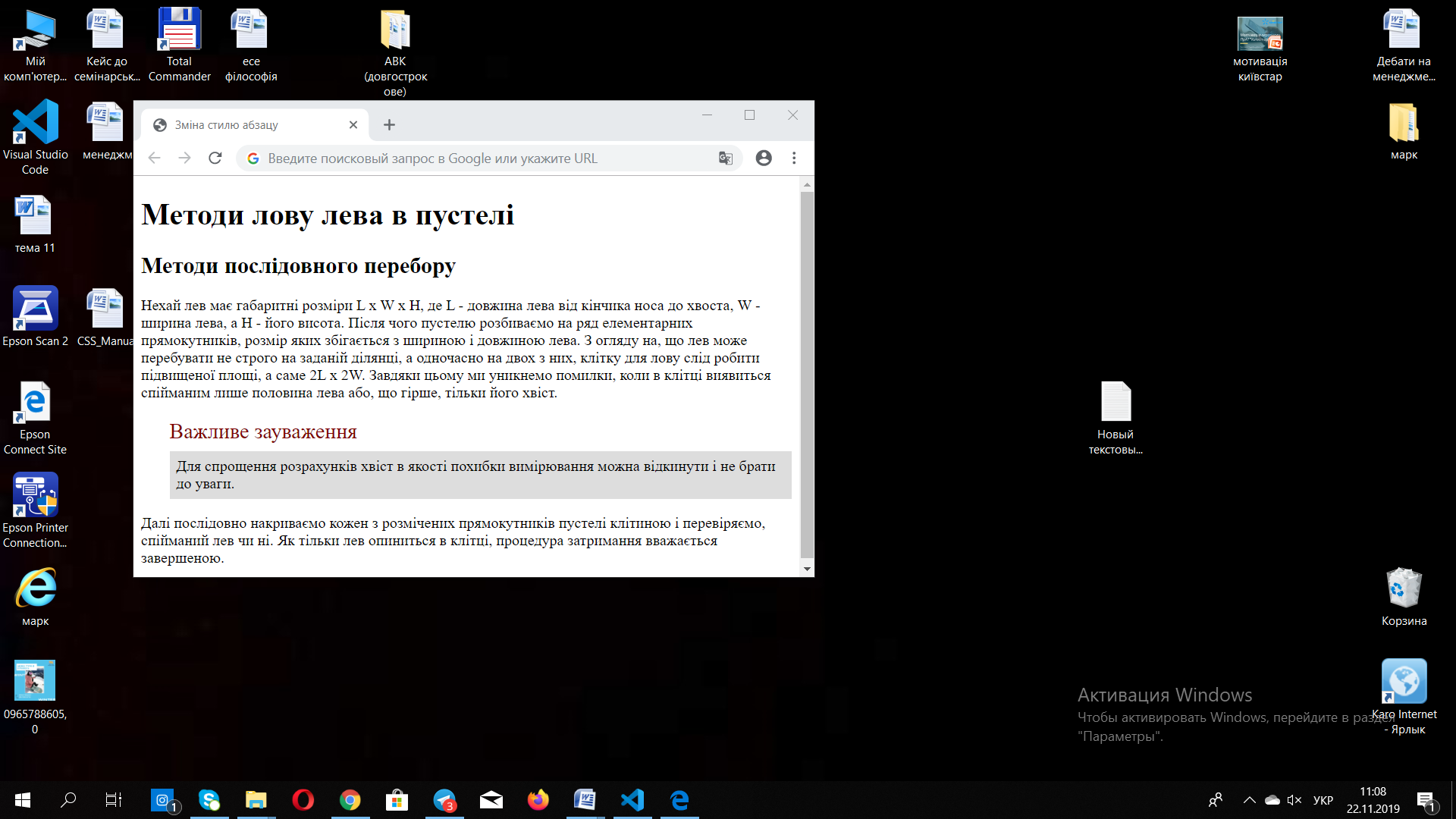
*Рис. 1.22. Виділення тексту кольором за допомогою сусідніх селекторів*

В даному прикладі відбувається зміна кольору тексту для вмісту контейнера **<i>**, коли він розташовується відразу після контейнера **<b>.**У першому абзаці така ситуація реалізована, тому слово «consectetuer» в браузері відображається червоним кольором. У другому абзаці, хоча і присутній тег **<i>**, але по сусідству ніякого тега **<b>** немає, так що стиль до цього контейнеру не застосовується.

Розберемо більш практичний приклад. Часто виникає необхідність в тексті статті включати різні виноски та примітки. Зазвичай для цієї мети створюють новий стильовий клас і застосовують його до абзацу, таким способом можна легко змінити вигляд тексту. Але ми підемо іншим шляхом і скористаємося сусідніми селекторами. Для виділення зауважень створимо новий клас, назвемо його sic, і будемо використовувати його до тегу **<h2>**. Перший абзац після такого заголовку виділяється кольором фону і відступом (приклад 1.38). Вид інших абзаців залишиться незмінним.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1. 38. Зміна стилю абзацу | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Зміна стилю абзацу</title>  <style type="text/css"> H2.sic {  font-size: 140%; /\* Размір шрифту \*/ color: maroon; /\* Колір тексту \*/  font-weight: normal; /\* Нормальне накреслення тексту \*/ margin-left: 30px; /\* Відступ зліва \*/  margin-bottom: 0px; /\* Відступ справа \*/  }  H2.sic + P {  background: #ddd; /\* Колір фону \*/ margin-left: 30px; /\* Відступ зліва \*/ margin-top: 0.5em; /\* Відступ зверху \*/ padding: 7px; /\* Поля навколо тексту\*/}  </style>  </head>  <body>  <h1> Методи лову лева в пустелі</h1>  <h2>Методи послідовного перебору</h2>  <p>Нехай лев має габаритні розміри L x W x H, де L - довжина лева від кінчика носа до хвоста, W - ширина лева, а H - його висота. Після чого пустелю розбиваємо на ряд елементарних прямокутників, розмір яких збігається з шириною і довжиною лева. З огляду на, що лев може перебувати не строго на заданій ділянці, а одночасно на двох з них, клітку для лову слід робити підвищеної площі, а саме 2L x 2W. Завдяки цьому ми уникнемо помилки, коли в клітці виявиться спійманим лише половина лева або, що гірше, тільки його хвіст.</p>  <h2 class="sic">Важливе зауваження</h2>  <p> Для спрощення розрахунків хвіст в якості похибки вимірювання можна відкинути і не брати до уваги.</p>  <p>Далі послідовно накриваємо кожен з розмічених прямокутників пустелі клітиною і перевіряємо, спійманий лев чи ні. Як тільки лев опиниться в клітці, процедура затримання вважається завершеною.</p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.23.



*Рис. 11.2. Зміна вигляду абзацу за рахунок використання сусудніх селекторів*

В даному прикладі текст відформатований із застосуванням абзаців (тег **<p>**), але запис H2.sic + P встановлює стиль тільки для першого абзацу що йде після тега **<h2>**, у якого доданий клас sic.

Сусідні селектори зручно використовувати для тих тегів, до яких автоматично додаються відступи, щоб самостійно регулювати величину відбиття. Наприклад, якщо поспіль йдуть теги **<h1>** і **<h2>**, то відстань між ними легко регулювати якраз за допомогою сусідніх селекторів. Аналогічно справа йде і для йдуть підряд тегів **<h2>** і **<p>**, а також в інших подібних випадках. У прикладі 1.39 таким чином змінюється величина відступів між зазначеними тегами.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.39. Відступи між заголовками і текстом | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Соседние селекторы</title>  <style type="text/css"> H1 + H2 {  margin-top: -10px; /\* Зміщуємо заголовок 2 вверх \*/  }  H2 + P {  margin-top: -1em; /\* Зміщуємо перший абзац вверх до заголовку \*/  }  </style>  </head>  <body>  <h1>Заголовок 1</h1>  <h2>Заголовок 2</h2>  <p>Абзац!</p>  </body>  </html> | | | | | | | | | |

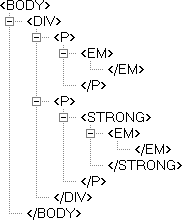
Оскільки при використанні сусідніх селекторів стиль застосовується тільки до другого елементу, то розмір відступів зменшується за рахунок включення від'ємного значення у властивості margin-top. При цьому текст піднімається вгору, ближче до попереднього елемента.

Дочірні селектори

Дочірнім називається елемент, який безпосередньо розташовується всередині батьківського елементу. Щоб краще зрозуміти відносини між елементами документа, розберемо невеликий код (приклад 1.40).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.40. Вкладеність елементів в документі | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Lorem ipsum</title>  </head>  <body>  <div class="main">  <p><em>Lorem ipsum dolor sit amet</em>, consectetuer adipiscing  elit, sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat.</p>  <p><strong><em>Ut wisis enim ad minim veniam</em></strong>,  quis nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>  </div>  </body>  </html> | | | | | | | | | |

В даному прикладі застосовується кілька контейнерів, які в коді розташовуються один в іншому. Найбільш наочно це видно на дереві елементів, так називається структура відносин тегів документа між собою (рис. 1.24).



*Рис. 1.24. Дерево елементів для прикладу*

На даному малюнку в зручному вигляді представлена вкладеність елементів і їх ієрархія. Тут дочірнім елементом по відношенню до тегу **<div>** виступає тег **<p>**. Разом з тим тег **<strong>** не є дочірнім для тега **<div>,** оскільки він розташований в контейнері **<p>**.

Повернемося тепер до селекторам. Дочірнім селектором вважається такою, що в дереві елементів знаходиться прямо всередині батьківського елементу. Синтаксис застосування таких селекторів наступний.

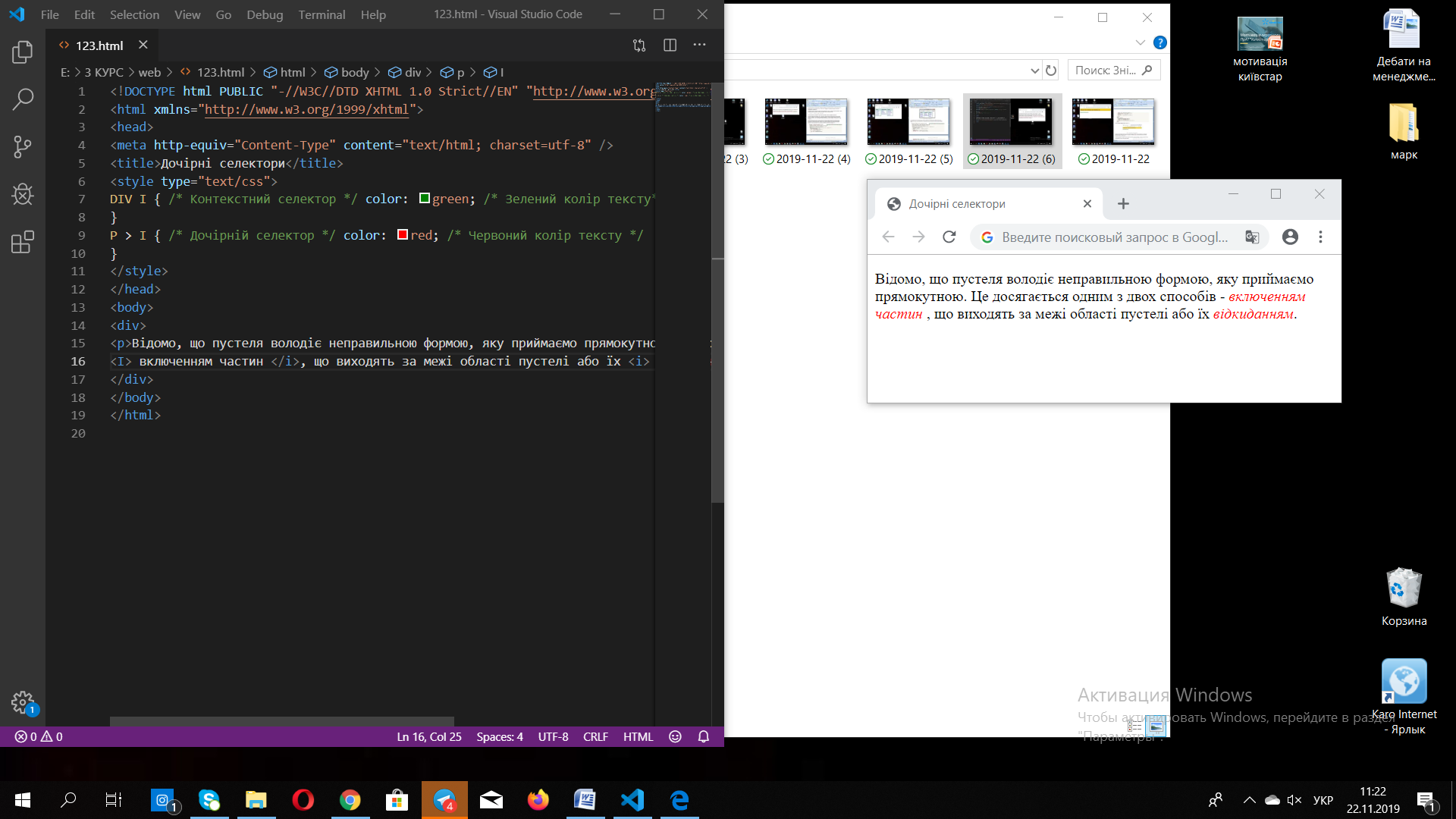
Селектор 1 > Селектор 2 { Опис правил стилю }

Стиль застосовується до селектора 2, але тільки в тому випадку, якщо він є дочірнім для селектора 1.

Якщо знову звернутися до прикладу 1.40, то стиль виду P > EM { color: red } буде встановлений для першого абзацу документа, оскільки тег **<em>** знаходиться всередині контейнера **<p>**, та не дасть ніякого результату для другого абзацу. А все через те, що тег **<em>** в другому абзаці розташований в контейнері **<strong>**, тому порушується умова вкладеності.

За своєю логікою дочірні селектори схожі на контекстні селектори. Різниця між ними така. Стиль до дочірнього селектору застосовується тільки в тому випадку, коли він є прямим нащадком, іншими словами, безпосередньо розташовується всередині батьківського елементу. Для контекстного селектора ж допустимо будь-який рівень вкладеності. Щоб стало зрозуміло, про що йде мова, розберемо наступний код (приклад 1.41).

Результат даного прикладу показано на рис. 1.25.



*Рис. 1.25. Колір тексту, заданий за допомогою дочірнього селектора*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.41. Контекстні та дочірні селектори | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Дочірні селектори</title>  <style type="text/css">  DIV I { /\* Контекстний селектор \*/ color: green; /\* Зелений колір тексту\*/  }  P > I { /\* Дочірній селектор \*/ color: red; /\* Червоний колір тексту \*/}  </style>  </head>  <body>  <div>  <p>Відомо, що пустеля володіє неправильною формою, яку приймаємо прямокутною. Це досягається одним з двох способів - <I> включенням частин </ i>, що виходять за межі області пустелі або їх <i> відкиданням</i>.  </div>  </body>  </html> | | | | | | | | | | |

На тег **<i>** в прикладі діють одночасно два правила: контекстний селектор (тег **<i>** розташований всередині  
**<div>**) і дочірній селектор (тег **<i>** є дочірнім по відношенню до **<p>**). При цьому правила є рівносильними, оскільки всі умови для них виконуються і не суперечать один одному. У подібних випадках застосовується стиль, який розташований в коді нижче, тому курсивний текст відображається червоним кольором. Варто поміняти правила місцями і поставити **DIV I** нижче, як колір тексту зміниться з червоного на зелений.

Зауважимо, що в більшості випадків від додавання дочірніх селекторів можна відмовитися, замінивши їх контекстними селекторами. Однак використання дочірніх селекторів розширює можливості по управлінню стилями елементів, що в підсумку дозволяє отримати потрібний результат, а також простий і наочний код.

Найзручніше застосовувати зазначені селектори для елементів, які мають ієрархічну структуру - сюди відносяться, наприклад, таблиці і різні списки. У прикладі 1.42 показано зміна виду списку за допомогою стилів. За рахунок вкладення одного списку в інший отримуємо різновид меню. Заголовки при цьому розташовуються горизонтально, а набір посилань - вертикально під заголовками (рис. 1.26).



*Рис. 1.26. Список в виді меню*

Для розміщення тексту по горизонталі до селектора **LI** додається стильова властивість float. Щоб при цьому розділити між собою стиль горизонтального і вертикального списку і застосовуються дочірні селектори (приклад 1.42).

В даному прикладі дочірні селектори потрібні, щоб розділити стиль елементів списку верхнього рівня і вкладені списки, які виконують різні завдання, тому стиль для них не повинен перетинатися.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.42. Використання дочірніх селекторів | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Дочірні селектори</title>  <style type="text/css"> UL#menu {  margin: 0; padding: 0; /\* Забираємо відступи \*/  }  UL#menu > LI {  list-style: none; /\* Забираємо маркери списку  \*/ width: 100px; /\* Ширина елемента в пікселях \*/  background: #b3d9d2; /\* Колір фону \*/ color: #333; /\* Колір тексту \*/ padding: 5px; /\*Поля навколо тексту \*/  font-family: Arial, sans-serif; /\* Рубаний шрифт \*/ font-size: 90%; /\* Розмір шрифта \*/  font-weight: bold; /\* Жирное накреслення \*/  float: left; /\* Розташовуємо елементи по горизонталі \*/  }  LI > UL {  list-style: none; /\* Забираємо маркери списку \*/  margin: 0; padding: 0; /\* Забираємо відступи навколо елементів списку \*/ border-bottom: 1px solid #666; /\* Межа внизу \*/  padding-top: 5px; /\* Добавляємо відступ зверху \*/  }  LI > A {  display: block; /\* Посилання відображається у вигляді блоку \*/ font-weight: normal; /\* Нормальне накреслення тексту \*/ font-size: 90%; /\* Розмір шрифту \*/  background: #fff; /\* Колір фону \*/  border: 1px solid #666; /\* Параметри рамки \*/ border-bottom: none; /\* Забираємо межу знизу \*/ padding: 5px; /\* Поля навколо тексту \*/  }  </style>  </head>  <body>  <ul id="menu">  <li>Правка  <ul>  <li><a href="#">Відмінити</a></li>  <li><a href="#">Вирізати</a></li>  <li><a href="#">Копіювати</a></li>  <li><a href="#">Вставити</a></li>  </ul>  </li>  <li>Накреслення  <ul>  <li><a href="#">Жирне</a></li>  <li><a href="#">Курсивне</a></li>  <li><a href="#">Підкреслене</a></li>  </ul>  </li>  <li>Розмір  <ul>  <li><a href="#">Маленький</a></li>  <li><a href="#">Нормальный</a></li>  <li><a href="#">Середній</a></li>  <li><a href="#">Великий</a></li>  </ul>  </li>  </ul>  </body>  </html> | | | | | | | | | | |

Селектори атрибутів

Більшість тегів розрізняються за своєю дією в залежності від того, які в них використовуються атрибути. Наприклад, тег **<input>** може створювати кнопку, текстове поле і інші елементи форми всього лише за рахунок зміни значення атрибуту type. При цьому додавання правил стилю до селектора **INPUT** застосує стиль одночасно до всіх створених за допомогою цього тега елементам. Щоб гнучко управляти стилем подібних елементів, в CSS введені селектори атрибутів. Вони дозволяють встановити стиль за присутністю певного атрибута тега або його значення.

Розглянемо кілька типових варіантів застосування таких селекторів.

Простий селектор атрибуту

Встановлює стиль для елемента, якщо заданий специфічний атрибут тега. Його значення в даному випадку не важливо. Синтаксис застосування такого селектора наступний.

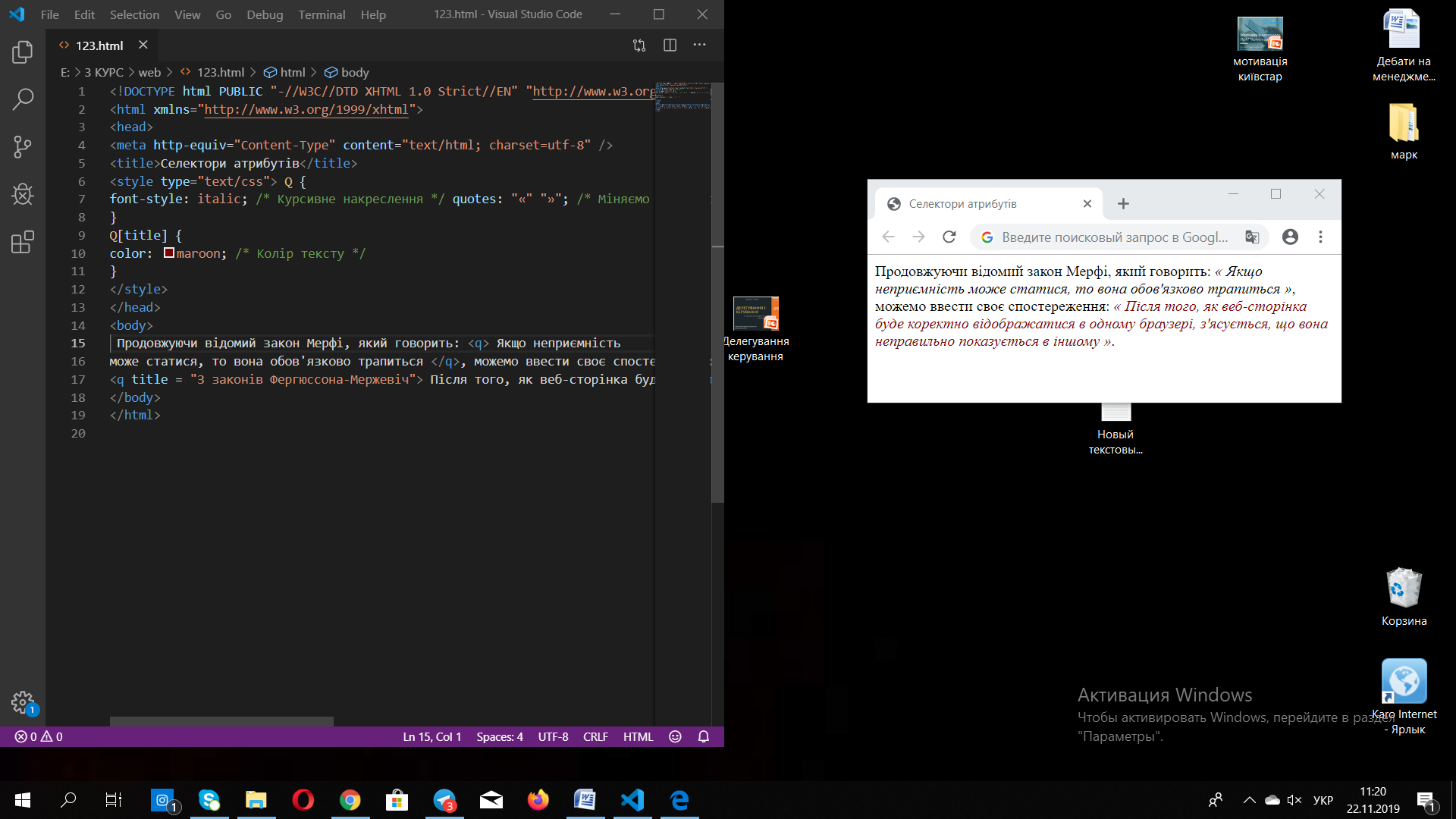
[атрибут] { Опис правил стилю } Селектор[атрибут] { Опис правил стилю }

Стиль застосовується до тих тегів, всередині яких був доданий вказаний атрибут. Пропуск між ім'ям селектора і квадратними дужками не допускається

У прикладі 1.43 показано зміну стилю тега **<q>**, в тому випадку, якщо до нього додано атрибут title.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.43. Вигляд елементу в залежності від його атрибута | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селектори атрибутів</title>  <style type="text/css"> Q {  font-style: italic; /\* Курсивне накреслення \*/ quotes: "«" "»"; /\* Міняємо вид лапок в цитаті \*/  }  Q[title] {  color: maroon; /\* Колір тексту \*/  }  </style>  </head>  <body>  <р> Продовжуючи відомий закон Мерфі, який говорить: <q> Якщо неприємність може статися, то вона обов'язково трапиться </ q>, можемо ввести своє спостереження: <q title = "З законів Фергюссона-Мержевіч"> Після того, як веб-сторінка буде коректно відображатися в одному браузері, з'ясується, що вона неправильно показується в іншому </ q>.</p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показаний на рис. 1.27. Браузер Safari і Chrome не виводять наші лапки в тексті, замінюючи їх на стандартні.



*Рис. 1.27. Зміна стилю елемента в залежності від застосування атрибута title*

В даному прикладі змінюється колір тексту усередині контейнера **<q>**, коли до нього додається title. Зверніть увагу, що для селектора **Q[title]** немає потреби повторювати стильові властивості, оскільки вони успадковуються від селектора **Q**.

Атрибут із значенням

Встановлює стиль для елемента в тому випадку, якщо задано певне значення специфічного атрибута. Синтаксис застосування наступний.

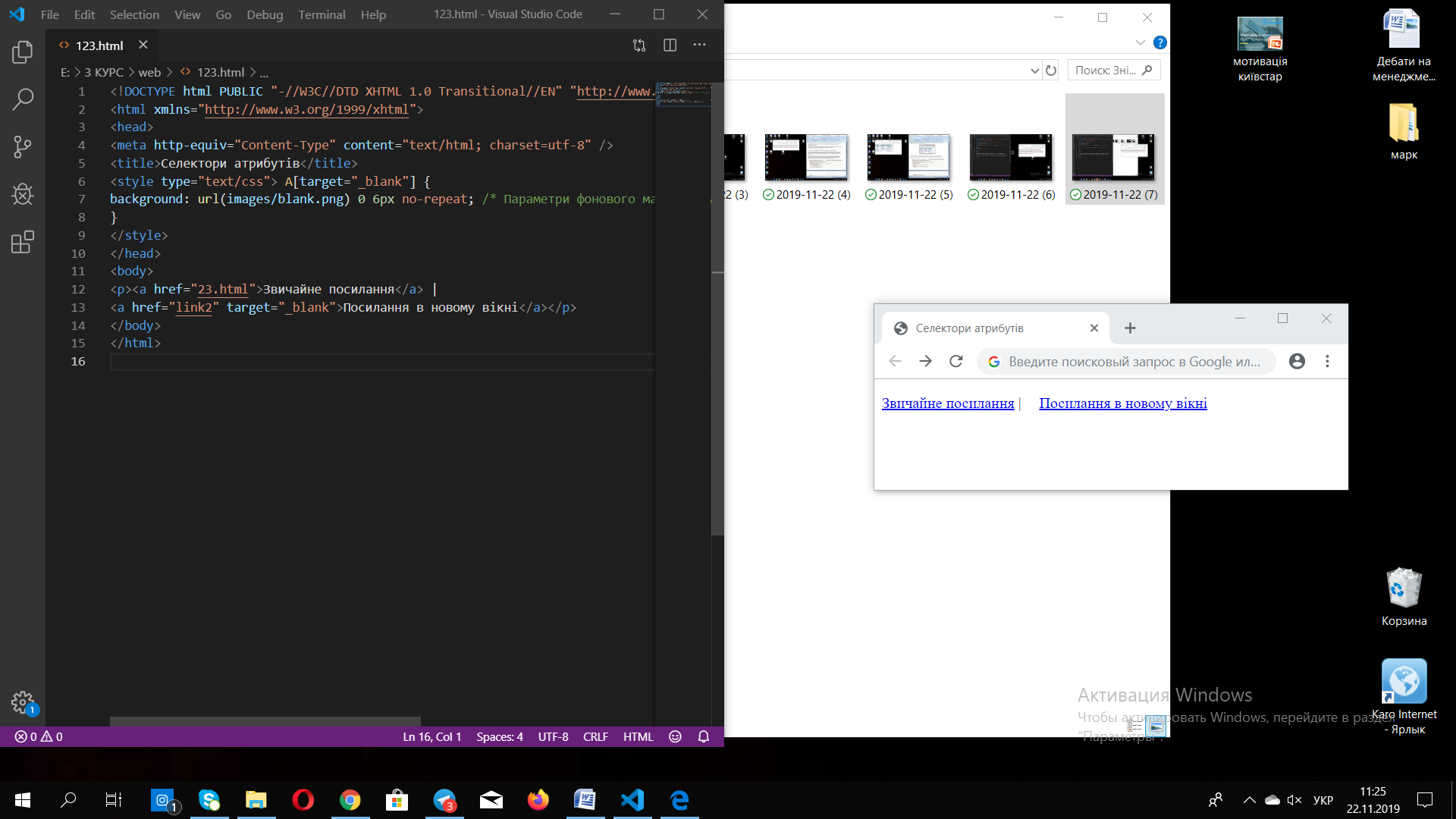
[атрибут="значення"] { Опис правил стилю} Селектор[атрибут="значення"] { Опис правил стилю}

У першому випадку стиль застосовується до всіх тегів, які містять вказане значення. А в другому - тільки до певних селекторів.

У прикладі 1.44 показано зміна стилю посилання в тому випадку, якщо тег **<a>** містить атрибут target зі значенням  
\_ blank. При цьому посилання буде відкриватися в новому вікні і щоб показати це, за допомогою стилів додаємо невеликий малюнок перед текстом посилання.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.44. Стиль для відкриття посилань в новому вікні | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd)-[transitional.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селектори атрибутів</title>  <style type="text/css"> A[target="\_blank"] {  background: url(images/blank.png) 0 6px no-repeat; /\* Параметри фонового малюнка \*/ padding-left: 15px; /\* Зміщує текст вправо \*/  }  </style>  </head>  <body>  <p><a href="1.html">Звичайне посилання</a> |  <a href="link2" target="\_blank">Посилання в новому вікні</a></p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано нижче (рис. 1.28).



*Рис. 1.28. Зміна стилю елемента в залежності від значення target*

В даному прикладі малюнок до посилання додається за допомогою властивості background. У його функції входить створення повторюваної фонової картинки, але повторення фону можна скасувати через значення no-repeat, що в підсумку дасть єдине зображення.

Значення атрибута починається з певного тексту

Встановлює стиль для елемента в тому випадку, якщо значення атрибута тега починається з вказаного тексту. Синтаксис застосування наступний.

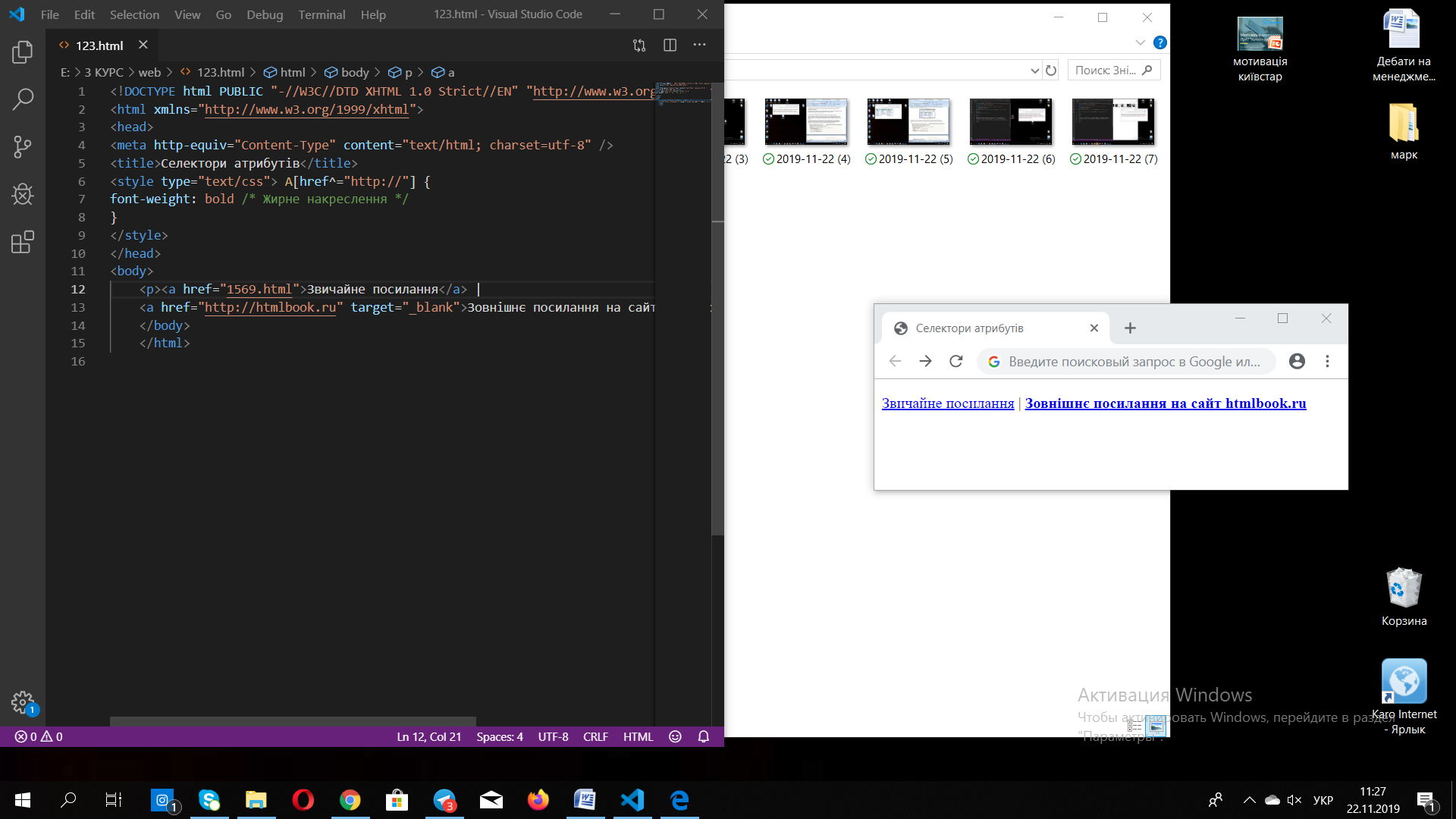
[атрибут^="значення"] { Опис правил стилю } Селектор[атрибут^="значення"] { Опис правил стилю }

У першому випадку стиль застосовується до всіх елементів, у яких значення атрибута починаються з вказаного тексту. А в другому - тільки до певних селекторів. Використання лапок не обов'язково.

Припустимо, що на сайті потрібно розділити стиль звичайних і зовнішніх посилань - посилання, які ведуть на інші сайти. Щоб не додавати до тегу **<a>** новий клас, скористаємося селекторами атрибутів. Зовнішні посилання характеризуються додаванням до адреси протоколу, наприклад, для доступу до гіпертекстових документів використовується протокол HTTP. Тому зовнішні посилання починаються з ключового слова http://, його і додаємо до селектора **A**, як показано в прикладі 1.45.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.45. Зміна стилю зовнішнього посилання | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селектори атрибутів</title>  <style type="text/css"> A[href^="http://"] {  font-weight: bold /\* Жирне накреслення \*/  }  </style>  </head>  <body>  <p><a href="1.html">Звичайне посилання</a> |  <a [href="http://htmlbook.ru"](http://htmlbook.ru/) target="\_blank">Зовнішнє посилання на сайт htmlbook.ru</a></p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано нижче (рис. 1.29).



*Рис. 1.29. Зміна стилю для зовнішніх посилань*

В даному прикладі зовнішні посилання виділяються жирним шрифтом. Також можна скористатися показаним в прикладі 1.44 прийомом і додавати до посилання невелике зображення, яке буде повідомляти, що посилання веде на інший сайт.

Значення атрибута закінчується певним текстом

Встановлює стиль для елемента в тому випадку, якщо значення атрибута закінчується зазначеним текстом. Синтаксис застосування наступний.

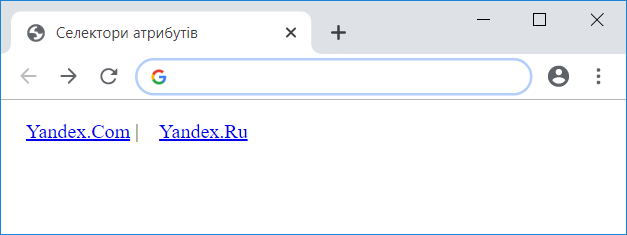
[атрибут$="значення"] { Опис правил стилю } Селектор[атрибут$="значение"] { Опис правил стилю }

У першому випадку стиль застосовується до всіх елементів у яких значення атрибута завершується заданим текстом. А в другому - тільки до певних селекторів.

Таким способом можна автоматично розділяти стиль для сайтів домена ru і для сайтів інших доменів на зразок com, як показано в прикладі 1.46.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.46. Стиль для різних доменів | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селектори атрибутів</title>  <style type="text/css">  A[href$=".ru"] { /\* Якщо посилання закінчується на .ru \*/  background: url(images/ru.png) no-repeat 0 6px; /\* Добавляємо фоновий малюнок \*/ padding-left: 12px; /\* Зміщуємо текст вправо \*/  }  A[href$=".com"] { /\* Якщо посилання закінчується на .com \*/ background: url(images/com.png) no-repeat 0 6px;  padding-left: 12px;  }  </style>  </head>  <body>  <p><a [href="http://www.yandex.com">Yandex.Com</a>](http://www.yandex.com/) |  <a [href="http://www.yandex.ru">Yandex.Ru</a></p>](http://www.yandex.ru/)  </body>  </html> | | | | | | | | | | |

В даному прикладі містяться два прямих посилань на різні домени - com і ru. При цьому до кожного такого посиланням за допомогою стилів додається своя фонова картинка (рис. 1.30). Стильові властивості будуть застосовуватися тільки для тих посилань, атрибут href яких закінчується на «.ru» або «.com». Зауважте, що додавши до імені домена слеш (http://www.yandex.ru/) або адресу сторінки (http://www.yandex.ru/fun.html), ми змінимо тим самим закінчення і стиль застосовуватися вже не буде . В цьому випадку краще скористатися командою \*=.



*Рис. 1.30. Добавлення картинки до посилань*

Значення атрибута містить вказаний текст

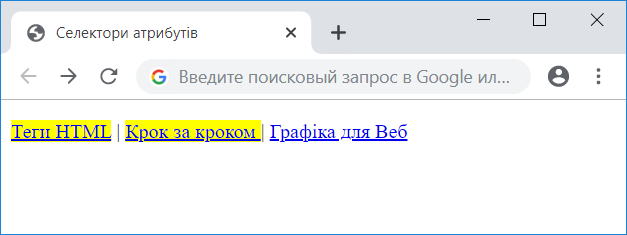
Можливі варіанти, коли стиль слід застосувати до тегу з певним атрибутом, при цьому частиною його значення є деякий текст. При цьому точно не відомо, в якому місці значення включений даний текст - на початку, середині або наприкінці. У подібному випадку слід використовувати такий синтаксис.

[атрибут$="значення"] { Опис правил стилю } Селектор[атрибут$="значение"] { Опис правил стилю }

У прикладі 1.47 показано зміна стилю посилань, в атрибуті href яких зустрічається слово «htmlbook».

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.47. Стиль для різних сайтів | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Селектори атрибутів</title>  <style type="text/css"> [href\*="htmlbook"] {  background: yellow; /\* Жовтий колір фону \*/  }  </style>  </head>  <body>  <p><a [href="http://www.htmlbook.ru/html/">Теги](http://www.htmlbook.ru/html/) HTML</a> |  <a [href="http://stepbystep.htmlbook.ru">Крок за кроком](http://stepbystep.htmlbook.ru/)  </a> |  <a [href="http://webimg.ru">Графіка](http://webimg.ru/) для Веб</a></p>  </body>  </html> | | | | | | | | | | |

Результат данного прикладу показано на рис. 1.31.



*Рис. 1.31. Зміна стилю для посилання, в адресі яких зустрічається «htmlbook»*

Одне з кількох значень атрибута

Деякі значення атрибутів можуть перераховуватися через пробіл, наприклад імена класів. Щоб задати стиль при наявності в списку необхідного значення застосовується наступний синтаксис.

[атрибут$="значення"] { Опис правил стилю } Селектор[атрибут$="значение"] { Опис правил стилю }

Стиль застосовується в тому випадку, якщо у атрибута є вказане значення або воно входить в список значень, що розділяються пропуском (приклад 1.48).

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.48. Стиль в залежності від ім‘я класу | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Блок</title>  <style type="text/css"> [class~="block"] h3 { color: green; }  </style>  </head>  <body>  <div class="block tag">  <h3>Заголовок</h3>  </div>  </body>  </html> | | | | | | | | | | |

В даному прикладі зелений колір тексту застосовується до селектора H3, якщо ім'я класу у шару задано як block. Відзначимо, що аналогічний результат можна отримати, якщо використовувати конструкцію \*= замість ~=.

Дефіс в значенні атрибута

В іменах ідентифікаторів і класів дозволено використовувати символ дефіса (-), що дозволяє створювати значущі значення атрибутів id і class. Для зміни стилю елементів, в значенні яких застосовується дефіс, слід скористатися таким синтаксисом.

[атрибут$="значення"] { Опис правил стилю } Селектор[атрибут$="значение"] { Опис правил стилю }

Стиль застосовується до елементів, у яких атрибут починається з вказаного значення або з фрагмента значення, після якого йде дефіс (приклад 1.49).

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.49. Дефіси в значеннях | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Блок</title>  <style type="text/css"> DIV[class|="block"] {  background: #306589; /\* Колір фону \*/ color: #acdb4c; /\* Колір тексту \*/ padding: 5px; /\* Поля \*/  }  DIV[class|="block"] A {  color: #fff; /\* Колір посилань \*/  }  </style>  </head>  <body>  <div class="block-menu-therm">  <h2>Термины</h2>  <div class="content">  <ul class="menu">  <li><a href="t1.html">Буквиця</a></li>  <li><a href="t2.html">Виворотка</a></li>  <li><a href="t3.html">Вирівнювання</a></li>  <li><a href="t4.html">Міжрядкові</a></li>  <li><a href="t5.html">Капітель </a></li>  <li><a href="t6.html">Накреслення </a></li>  <li><a href="t7.html">Відбиття </a></li>  </ul>  </div>  </div>  </body>  </html> | | | | | | | | | | |

В даному прикладі ім'я класу задано як block-menu-therm, тому в стилях використовується конструкція |="block", оскільки значення починається саме з цього слова і в значенні зустрічаються дефіси.

Всі перераховані методи можна комбінувати між собою, визначаючи стиль для елементів, які містять два і більше атрибута. У подібних випадках квадратні дужки йдуть підряд.

[атрибут1="значення1"][атрибут2="значення2"] { Опис правил стилю } Селектор[атрибут1="значення1"][атрибут2="значення2"] { Опис правил стилю }

Універсальний селектор

Іноді потрібно встановити одночасно один стиль для всіх елементів веб-сторінки, наприклад, задати шрифт або зображення тексту. В цьому випадку допоможе універсальний селектор, який відповідає будь-якому елементу веб-сторінки.

Для позначення універсального селектора застосовується символ зірочки (\*) і в загальному випадку синтаксис буде наступний.

\* { Опис правил стилю }

У деяких випадках вказувати універсальний селектор не обов'язково. Так, наприклад, записи \*.class і .class є ідентичними за своїм результатом.

У прикладі 1.50 показано одне з можливих додатків універсального селектора - вибір шрифту і розміру тексту для всіх елементів документа.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.50. Використання універсального селектора | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Універсальний селектор</title>  <style type="text/css">  \* {  font-family: Arial, Verdana, sans-serif; /\* Рубааний шрифт для тексту \*/ font-size: 96%; /\* Розмір тексту \*/  }  </style>  </head>  <body>  <p> Середній самець лева має довжину близько трьох метрів і важить від 180 до 230 кілограм.</p>  </body>  </html> | | | | | | | | | |

Зауважимо, що аналогічний результат можна отримати, якщо в даному прикладі поміняти селектор \* на **BODY**.

Універсальний селектор часто застосовується для обнулення відступів і полів у всіх елементів, як показано в прикладі 1.51.

Приклад 1. 51. Стиль для обнулення відступів і полів

\* { margin: 0;

padding: 0;

}

Універсальний селектор часто застосовується для обнулення відступів і полів у всіх елементів, як показано в прикладі 1.51.

Незважаючи на зовнішні зручності універсального селектора і його широке поширення, ні в якому разі не йдіть на поводу «моди» і не використовуйте селектор, як показано в прикладі вище. Висновки наступні.

* Універсальний селектор застосовує стиль до всіх елементів веб-сторінки, включаючи невидимі, що призводить до уповільнення браузера, оскільки йому потрібен якийсь час для побудови дерева елементів і додавання до них стилів. Чим більше елементів в коді, тим сильніше виражено уповільнення. У деяких крайніх випадках взагалі може з'явитися «зависання» браузера на декілька секунд.
* При неправильному використанні універсального селектора результат може виявитися непередбачуваним. Приклад нижче є повністю коректним з точки зору CSS, але призводить сторінку до парадоксального виду.

\* {

display: block;

border: 1px solid #c00;

}

* «Обнулення стилів» (див. Приклад 1.51) прищеплює у розробника погану манеру верстки. Замість того щоб знати, які значення властивостей встановлені за замовчуванням, розробник перекладає цю роботу на браузер, насильно встановлюючи всі значення в нуль. В результаті деякі значення застосовуються до властивостей, для яких не можуть встановлюватися або застосовуються до властивостей, у яких дане значення і так нульове. Це знову ж таки призводить до підвищення навантаження на браузер і уповільнення його роботи.
* Застосування одного стилю відразу до всіх елементів іноді призводить до помилок відображення елементів в  
   окремих браузерах. У прикладі нижче до посилань додається пунктирне підкреслення, яка не показується в IE7 через заданого обнулення полів у посилань.

\* { padding: 0;

}

a {

text-decoration: none;

border-bottom: 1px dashed red;

}

Незважаючи на зазначені особливості, універсальний селектор можна і потрібно включати в стиль, але в комбінації з іншими селекторами. Так, якщо потрібно змінити стиль всіх елементів в формі, то слід використовувати контекстний селектор FORM \* або селектор, показаний в прикладі 1.52.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.52. Використання універсального селектора | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Універсальний селектор</title>  <style type="text/css"> FORM P \* {  border: 1px solid #333; /\* Параметри рамки \*/ background: #ffe; /\* Колір фону \*/  }  </style>  </head>  <body>  <form action="handler.php">  <p><input type="text" /></p>  <p><input type="submit" /></p>  </form>  </body>  </html> | | | | | | | | | |

В даному прикладі рамка і фон додається до всіх елементів форми, розташованих усередині абзацу (тег **<p>**).

Псевдокласи

Псевдокласи визначають динамічний стан елементів, яке змінюється з часом або за допомогою дій користувача, а також положення в дереві документа. Прикладом такого стану служить текстове посилання, яка змінює свій колір при наведенні на неї курсора миші. При використанні псевдокласів браузер не перевантажує поточний документ, тому за допомогою псевдокласів можна отримати різні динамічні ефекти на сторінці.

Синтаксис застосування псевдокласів наступний.

Селектор:Псевдоклас { Описе правил стилю }

Спочатку вказується селектор, до якого додається псевдоклас, потім слідує двокрапка, після якого йде ім'я псевдокласу. Допускається застосовувати псевдокласи до імен ідентифікаторів або класів (A.menu:hover {color: green}), а також до контекстних селекторів (.menu A:hover {background: #fc0}). Якщо псевдоклас вказується без селектора попереду (:hover), то він буде застосовуватися до всіх елементів документа.

Умовно всі псевдокласи діляться на три групи:

* що визначають стан елементів;
* що мають відношення до дерева елементів;
* вказують мову тексту.

 Надалі імена псевдокласів писатимемо з двокрапкою попереду, щоб розрізняти між собою псевдокласи і стильові властивості.

Псевдокласи, що визначають стан елементів

До цієї групи належать псевдокласи, які розпізнають поточний стан елемента і застосовують стиль тільки для цього стану.

:active

Відбувається при активації користувачем елементу. Наприклад, посилання стає активним, якщо навести на нього курсор і клацнути мишкою. Незважаючи на те, що активним може стати практично будь-який елемент веб-сторінки, псевдоклас: active використовується переважно для посилань.

:link

Застосовується до невідвідування посиланнями, тобто таких посиланнях, на які користувач ще не натискав. Браузер деякий час зберігає історію відвідувань, тому посилання може бути позначена як відвідана хоча б тому, що по ній був зафіксований перехід раніше.

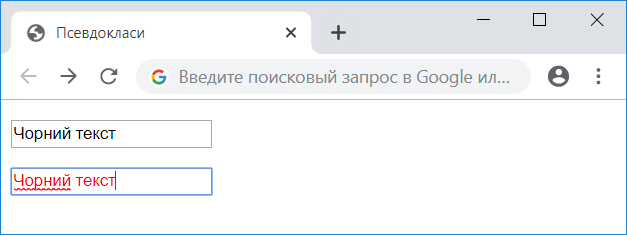
 Запис **A {...}** и **A:link** за своїм результатом рівноцінна, оскільки в браузері дає один і той же ефект, тому псевдоклас: link можна не вказувати. Винятком є якоря, на них дію**:link** не поширюється.

:focus

Застосовується до елемента при отриманні ним фокусу. Наприклад, для текстового поля форми отримання фокусу означає, що курсор встановлений в поле, і за допомогою клавіатури можна вводити в нього текст (приrkfl 1.53).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.53. Застосування псевдокласу focus | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> INPUT:focus {  color: red; /\* Червоний колір тексту \*/  }  </style>  </head>  <body>  <form action="">  <p><input type="text" value="Чорний текст" /></p>  <p><input type="text" value="Чорний текст" /></p>  </form>  </body>  </html> | | | | | | | | | |

Результат прикладу показаний нижче (рис. 1.32). У другому рядку знаходиться курсор, тому текстове поле отримало фокус.



*Рис. 1.32. Зміна стилю тексту при отриманні фокусу*

В даному прикладі в текстовому полі міститься попередній текст, він визначається значенням атрибута value тега **<input>**. При натисканні на елемент форми відбувається отримання полем фокусу, і колір тексту змінюється на червоний. Досить клацнути в будь-якому місці сторінки (крім текстового поля, природно), як відбудеться втрата фокусу і текст повернеться до первісного чорного кольору.

 Результат буде видно тільки для тих елементів, які можуть отримати фокус. Зокрема, це теги **<a>**,

**<input>**, **<select>** і **<textarea>**.

:hover

Псевдоклас :hover активізується, коли курсор миші знаходиться в межах елемента, але клацання по ньому не відбувається.

:visited

Даний псевдоклас застосовується до відвіданих посилань. Зазвичай таке посилання змінює свій колір за замовчуванням на фіолетовий, але за допомогою стилів колір і інші параметри можна задати самостійно (приклад 1.54).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.54. Зміна кольору посилань | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> A:link {  color: #036; /\* Колір невідвіданих посилань \*/  }  A:visited {  color: #606; /\* Колір відвіданих посилань \*/  }  A:hover {  color: #f00; /\* Колір посилань при наведенні на них курсора миші \*/  }  A:active {  color: #ff0; /\* Колір активних посилань\*/  }  </style>  </head>  <body>  <p><a href="1.html">Посилання 1</a> |  <a href="2.html"> Посилання 2</a> |  <a href="3.html"> Посилання 3</a></p>  </body>  </html> | | | | | | | | | |

Результат прикладу показано на рис. 1.33.



*Рис. 1.33. Вид посилання при наведенні на неї курсора миші*

В даному прикладі показано використання псевдокласів спільно з посиланнями. При цьому має значення порядок проходження псевдокласів. Спочатку вказується: visited, а потім йде: hover, в іншому випадку відвідані посилання не будуть змінювати свій колір при наведенні на них курсора.

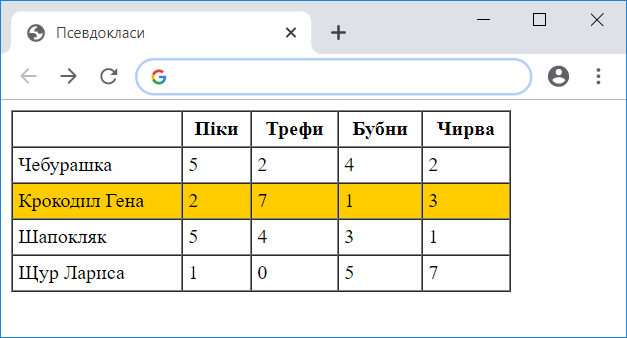
Селектори можуть містити більше одного псевдокласу, які перераховуються поспіль через двокрапку, але тільки в тому випадку, якщо їх дії не суперечать один одному. Так, запис **A:visited:hover** є коректною, а запис **A:link:visited** - немає. Втім, якщо підходити формально, то валідатор CSS вважає правильним будь-яке поєднання псевдокласів.

 Браузер Internet Explorer 6 дозволяє використовувати псевдокласи: active і: hover лише для посилань. Починаючи з версії 7.0 псевдокласи в цьому браузері працюють і для інших елементів.

Псевдоклас:hover не обов'язково повинен застосовуватися до посилань, його можна додавати і до інших елементів документа. Так, в прикладі 1.55 показана таблиця, рядки якої змінюють свій колір при наведенні на них курсора миші. Це досягається за рахунок додавання псевдокласу:hover до селектору **TR**.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.55. Виділення рядків таблиці | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> TR:hover {  background: #fc0; /\* Міняємо колір фону ряжка таблиці \*/  }  </style>  </head>  <body>  <table width="400" border="1" cellpadding="4" cellspacing="0">  <tr>  <th>&nbsp;</th>  <th>Піки</th><th>Трефи</th><th>Бубни</th><th>Черва</th>  </tr>  <tr>  <td>Чебурашка</td>  <td>5</td><td>2</td><td>4</td><td>2</td>  </tr>  <tr>  <td>Крокодил Гена</td>  <td>2</td><td>7</td><td>1</td><td>3</td>  </tr>  <tr>  <td>Шапокляк</td>  <td>5</td><td>4</td><td>3</td><td>1</td>  </tr>  <tr>  <td>Щур Лариса</td>  <td>1</td><td>0</td><td>5</td><td>7</td>  </tr>  </table>  </body>  </html> | | | | | | | | | |

Результат прикладу показано нище (рис. 1.34).



*Рис. 1.34. Виділення рядків таблиці при наведенні на них курсора миші*

Псевдокласи, що мають відношення до дерева документа

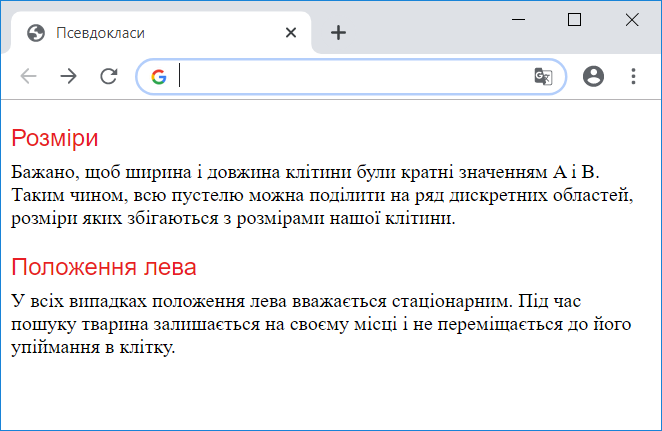
До цієї групи належать псевдокласи, які визначають положення елемента в дереві документа і застосовують до нього стиль в залежності від його статусу.

:first-child

Застосовується до першого дочірнього елементу селектора, який розташований в дереві елементів документа. Щоб  
 стало зрозуміло, про що мова, розберемо невеликий код (приклад 1.56).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.56. Використання псевдокласу first-child | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> div p:first-child {  font: 1.2em Arial, sans-serif; /\* Шрифт \*/ color: #E81E26; /\* Колір тексту \*/  margin-bottom: -0.5em; /\* Відступ знизу \*/  }  </style>  </head>  <body>  <div>  <p>Розміри</p>  <p> Бажано, щоб ширина і довжина клітини були кратні значенням A і B. Таким чином, всю пустелю можна поділити на ряд дискретних областей, розміри яких збігаються з розмірами нашої клітини.</p>  </div>  <div>  <p> Положення лева </ p> <P> У всіх випадках положення лева вважається стаціонарним. Під час пошуку тварина залишається на своєму місці і не переміщається до його упіймання в клітку. </p>  </div>  </body>  </html> | | | | | | | | | |

Результат прикладу показано нище (рис. 1.35).



*Рис. 1.35. Використання псевдокласу :first-child*

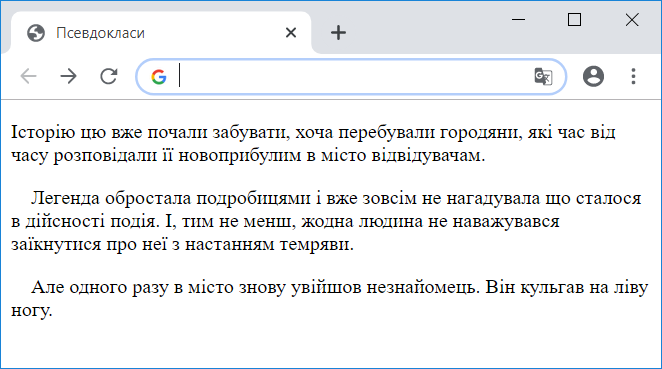
В даному прикладі псевдоклас:first-child додається до селектора **P** і встановлює для нього рубаний шрифт і червоний колір тексту. Хоча тег **<P>** всередині **<div>** зустрічається по два рази, стиль буде застосовуватися тільки до перших абзаців. В інших випадках стиль всередині **<P>** залишиться вихідним. З наступним тегом **<div>** все починається знову, оскільки батьківський елемент помінявся.

 Браузер Internet Explorer підтримує псевдоклас:first-child починаючи з версії 7.0.

Псевдоклас:first-child найзручніше використовувати в тих випадках, коли потрібно задати різний стиль для першого і інших однотипних елементів. Наприклад, за правилами типографіки червону рядок для першого абзацу тексту не встановлюють, а для інших абзаців додають відступ першого рядка. З цією метою застосовують властивість text-indent з потрібним значенням відступу. Але щоб змінити стиль першого абзацу і прибрати для нього відступ буде потрібно скористатися псевдокласом:first-child (приклад 1.57).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.57. Відступи для абзацу | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> P {  text-indent: 1em; /\* Відступ першого рядка \*/  }  P:first-child {  text-indent: 0; /\* Для першого абзацу відступ забираємо \*/  }  </style>  </head>  <body>  <p> Історію цю вже почали забувати, хоча перебували городяни, які час від часу розповідали її новоприбулим в місто відвідувачам. </ P> <p > Легенда обростала подробицями і вже зовсім не нагадувала що сталося в дійсності подія. І, тим не менш, жодна людина не наважувався заїкнутися про неї з настанням темряви. </p> <p > Але одного разу в місто знову увійшов незнайомець. Він кульгав на ліву ногу.</p>  </body>  </html> | | | | | | | | | |

Результат прикладу показано нище (рис. 1.36).



*Рис. 1.36. Зміна стилю першого абзацу*

В даному прикладі перший абзац тексту не містить відступу першого рядка, а для інших він встановлений.

Псевдокласи, що задають мову тексту

Для документів, які одночасно містять тексти на декількох мовах має значення дотримання правил синтаксису, характерні для тієї чи іншої мови. За допомогою псевдокласів можна змінювати стиль оформлення закордонних текстів, а також деякі настройки.

:lang

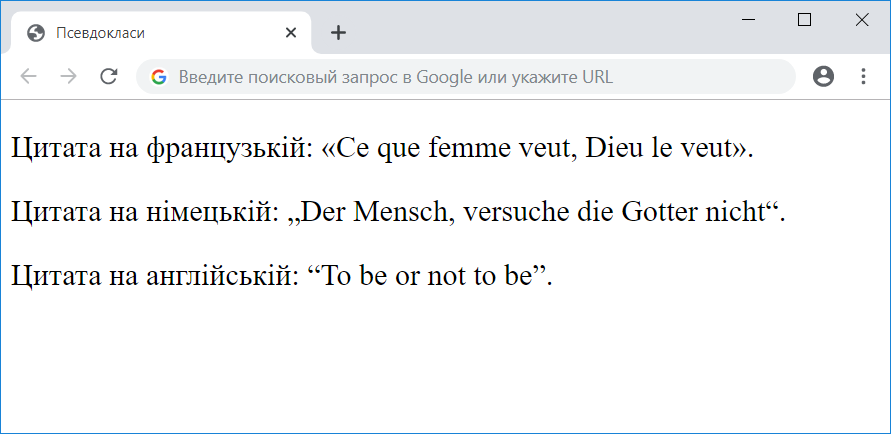
Визначає мову, який використовується в документі або його фрагменті. У коді HTML мова встановлюється через параметр charset тега **<meta>**. За допомогою псевдокласу:lang можна задавати певні настройки, характерні для різних мов, наприклад, вид лапок в цитатах. синтаксис наступний.

Элемент:lang(мова) { ... }

В якості мови можуть виступати наступні значення: ru - російська; en - англійська; de - німецька; fr - французька; it - італійська.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.58. Вид лапок в залежності від мови | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдокласи</title>  <style type="text/css"> P {  font-size: 1.50em; /\* Розмір тексту \*/  }  q:lang(de) {  quotes: "\201E" "\201C"; /\* Вид лапок для німецької мови \*/  }  q:lang(en) {  quotes: "\201C" "\201D"; /\* Вид лапок для англійської мови \*/  }  q:lang(fr), q:lang(ru) { /\* Вид лапок для російкої та французької мови \*/ quotes: "\00AB" "\00BB";  }  </style>  </head>  <body>  <p>Цитата на французій: <q lang="fr">Ce que femme veut, Dieu le veut</q>.</p>  <p>Цитата на німецьзькій: <q lang="de">Der Mensch, versuche die Gotter nicht</q>.</p>  <p>Цитата на англійській: <q lang="en">То be or not to be</q>.</p>  </body>  </html> | | | | | | | | | |

Результат даного прикладу показаний на рис. 1.37. Для відображення типових лапок в прикладі використовується стильова властивість quotes, а саме перемикання мови і відповідного виду лапок відбувається через атрибут lang, що додається до тегу **<q>**.



*Рис. 1.37. Різні лапки для різних мов*

Псевдоелементи

Псевдоелементи дозволяють задати стиль елементів не визначених в дереві елементів документа, а також генерувати вміст, якого немає у вихідному коді тексту.  
  
Синтаксис використання псевдоелементів наступний.

Селектор:Псевдоелемент { Опис правил стилю }

Спочатку слідує ім'я селектора, потім пишеться двокрапка, після якого йде ім'я псевдоелемента. Кожен псевдоелемент може застосовуватися тільки до одного селектору, якщо потрібно встановити відразу декілька псевдоелементів для одного селектора, правила стилю повинні додаватися до них окремо, як показано нижче.

.foo:first-letter { color: red }

.foo:first-line {font-style: italic}

В CSS3 щоб розрізняти псевдокласи і псевдоелементи, перед ім'ям псевдоелемента ставиться дві двокрапки (:: after). Internet Explorer ігнорує подібний запис, інші браузери коректно її розуміють.

 Псевдоелементи не можуть застосовуватися до внутрішніх стилів, тільки до таблиці пов'язаних або глобальних стилів.

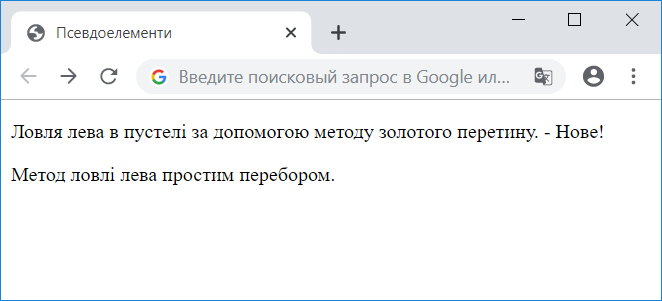
Далі перераховані псевдоелементи, їх опис та властивості.

:after

Застосовується для вставки призначеного контенту після елемента. Цей псевдоелемент працює спільно зі стильовою властивістю content, яка визначає вміст для вставки. У прикладі 1.59 показано використання псевдокласу: after для додавання тексту в кінець абзацу.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.59. Застосування псевдоелемента :after | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" ["http://www.w3.org/TR/html4/strict.dtd">](http://www.w3.org/TR/html4/strict.dtd)  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <title>Псевдоелементи</title>  <style type="text/css"> P.new:after {  content: " - Нове!"; /\* Добавляємо після тексту абзац \*/  }  </style>  </head>  <body>  <p class="new"> Ловля лева в пустелі за допомогою методу золотого перетину.</p> <p > Метод ловлі лева простим перебором.</p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано на рис. 1.38.



*Рис. 1.38. Добавлення тексту до абзацу за допомогою:after*

В даному прикладі до вмісту абзацу з класом new додається додаткове слово, яке виступає значенням властивості content.

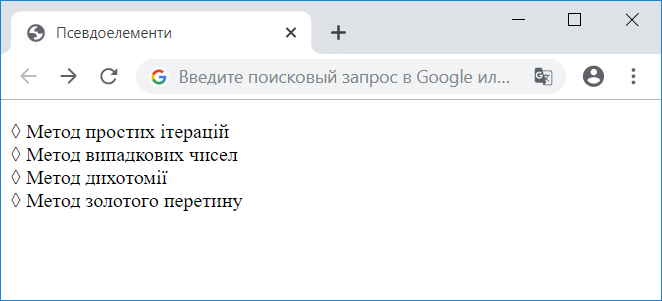
 Псевдоелементи :after і :before, а також стильова властивість content не підтримуються браузером Internet Explorer до сьомої версії включно.

:before

За своєю дією: before аналогічний псевдоелементу: after, але вставляє контент до елемента. У прикладі 1.60 показано додавання маркерів свого типу до елементів списку за допомогою приховування стандартних маркерів і застосування псевдокласу:before.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.60. Виористання псевдоелемента :before | XHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдоелементи</title>  <style type="text/css"> UL {  padding-left: 0; /\* Забираємо відступ зліва \*/  list-style-type: none; /\* Ховаємо маркери списку \*/  }  LI:before {  content: "◊ "; /\* Добавляємо перед елементом списку символ \*/  }  </style>  </head>  <body>  <ul>  <li>Метод простих ітерацій</li>  <li>Метод випадкових чисел</li>  <li>Метод дихотомії</li>  <li>Метод золотого перетину</li>  </ul>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано нище (рис. 1.39).



*Рис. 1.39. Зміна виду маркерів за допомогою :before*

В даному прикладі псевдоклас: before встановлюється для селектора **LI**, що визначає елементи списку. Додавання бажаних символів відбувається шляхом завдання значення властивості content. Зверніть увагу, що в якості аргументу не обов'язково виступає текст, можуть застосовуватися також символи юнікоду.

:first-letter

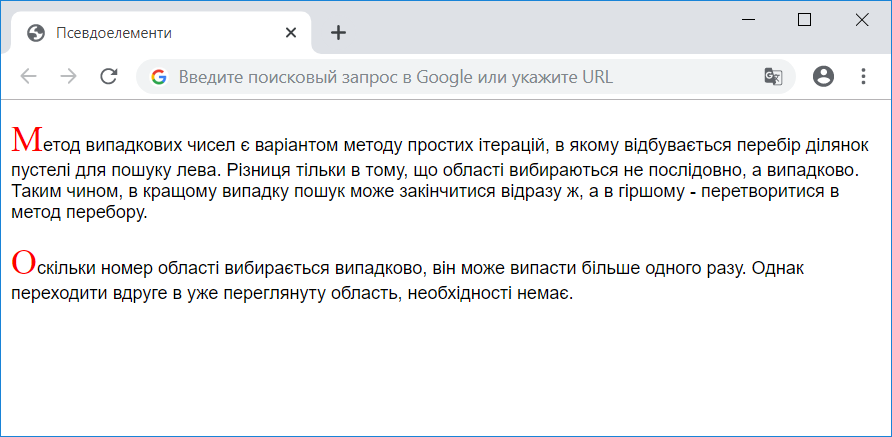
Визначає стиль першого символу в тексті елемента, до якого додається. Це дозволяє створювати в тексті буквицю і виступаючий ініціал.

 Буквиця являє собою збільшену першу літеру, базова лінія якої нижче на одну або кілька рядків базової лінії основного тексту. Виступаючий ініціал - збільшена прописна буква, базова  
лінія якої збігається з базовою лінією основного тексту.

Розглянемо приклад створення виступаючого ініціалу. Для цього потрібно додати до селектора **P** псевдоелемент  
:first-letter і встановити бажаний стиль ініціал. Зокрема, збільшити розмір тексту і поміняти колір тексту (приклад 1.61).

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.61. Використання псевдоелемента first-lette | rXHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдоелементи</title>  <style type="text/css"> P {  font-family: Arial, Helvetica, sans-serif; /\* Гарнітура шрифту основного тексту \*/ font-size: 0.9em; /\* Розмір шрифту \*/  color: black; /\* Чорний колір тексту \*/  }  P:first-letter {  font-family: 'Times New Roman', Times, serif; /\* Гарнітура шрифту першої літери \*/ font-size: 2em; /\* Розмір шрифту першого символу \*/  color: red; /\* Червоний колір тексту \*/  }  </style>  </head>  <body>  <p>Метод випадкових чисел є варіантом методу простих ітерацій, в якому відбувається перебір ділянок пустелі для пошуку лева. Різниця тільки в тому, що області вибираються не послідовно, а випадково. Таким чином, в кращому випадку пошук може закінчитися відразу ж, а в гіршому - перетворитися в метод перебору. </ р> <р> Оскільки номер області вибирається випадково, він може випасти більше одного разу. Однак переходити вдруге в уже переглянуту область, необхідності немає.</p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано нище (рис. 1.40).



*Рис. 1.40. Створення виступаючого ініціала*

В даному прикладі змінюється шрифт, розмір і колір першої літери кожного абзацу тексту.

:first-line

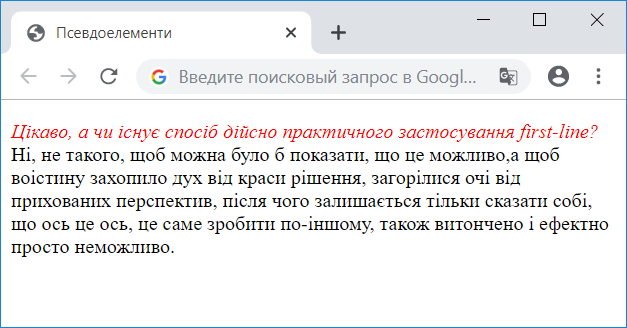
Визначає стиль першого рядка блокового тексту. Довжина цього рядка залежить від багатьох факторів, таких як використовуваний шрифт, розмір вікна браузера, ширина блоку, мови і т.д.

 До псевдоелементу: first-line можуть застосовуватися не всі стильові властивості. Допустимо використовувати властивості, що відносяться до шрифту, зміни кольору тексту і фону, а також: clear, line-height, letter-spacing, text-decoration, text-transform, vertical-align і word-spacing.

В прикладі 1.62 показано використання псевдоелемента: first-line стосовно абзацу тексту.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.62. Виділення першого рядка тексту | rXHTML 1.0 | CSS 2.1 | IE 6 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Псевдоелементи</title>  <style type="text/css"> P:first-line {  color: red; /\* Червоний колір тексту \*/  font-style: italic; /\* Курсивне накреслення \*/  }  </style>  </head>  <body>  <p>Цікаво, а чи існує спосіб дійсно практичного застосування first-line? Ні, не такого, щоб можна було б показати, що це можливо,а щоб воістину захопило дух від краси рішення, загорілися очі від прихованих перспектив, після чого залишається тільки сказати собі, що ось це ось, це саме зробити по-іншому, також витончено і ефектно просто неможливо.</p>  </body>  </html> | | | | | | | | | | |

Результат прикладу показано на рис. 1.41.



*Рис. 1.41. Результат застосування псевдоелемента :first-line*

В даному прикладі перший рядок виділяється червоним кольором і курсивним шрифтом. Зверніть увагу, що при зміні ширини вікна браузера, стиль першого рядка залишається постійним, незалежно від числа належних до неї слів.

Групування

При створенні стилю для сайту, коли одночасно використовується безліч селекторів, можлива поява повторюваних стильових правил. Щоб не повторювати двічі одні і ті ж елементи, їх можна згрупувати для зручності подання та скорочення коду. У прикладі 1.63 показаний звичайний запис, тут для кожного селектора наводиться свій набір стильових властивостей.

Приклад 1.63. Стиль для кожного селектора

H1 {

font-family: Arial, Helvetica, sans-serif; font-size: 160%;

color: #003;

}

H2 {

font-family: Arial, Helvetica, sans-serif; font-size: 135%;

color: #333;

}

H3 {

font-family: Arial, Helvetica, sans-serif; font-size: 120%;

color: #900;

}

P {

font-family: Times, serif;

}

З цього прикладу видно, що стиль для тегів заголовків містить однакове значення font-family. Групування якраз і дозволяє встановити одну властивість відразу для декількох селекторів, як показано в прикладі 1.64.

Приклад 1.64. Згруповані селектори

H1, H2, H3 {

font-family: Arial, Helvetica, sans-serif;

}

H1 {

font-size: 160%;

color: #003;

}

H2 {

font-size: 135%;

color: #333;

}

H3 {

font-size: 120%;

color: #900;

}

В даному прикладі font-family єдине для всіх селекторів застосовується відразу до декількох тегів, а індивідуальні властивості вже додаються до кожного селектора окремо.  
  
Селектори групуються у вигляді списку тегів, розділених між собою комами. До групи можуть входити не тільки селектори тегів, але також ідентифікатори і класи. Загальний синтаксис наступний.

Селектор 1, Селектор 2, ... Селектор N { Опис правил стилю }

При такому записі правила стилю застосовуються до всіх селекторів, перерахованих в одній групі.

Наслідування

Спадкування називається перенесення правил форматування для елементів, що знаходяться всередині інших. Такі елементи є дочірніми, і вони успадковують деякі стильові властивості своїх батьків, усередині яких розташовуються.  
  
Розберемо спадкування на прикладі таблиці. Особливістю таблиць можна вважати строгу ієрархічну структуру тегів. Спочатку слід контейнер **<table>** всередині якого додаються теги **<tr>**, а потім йде тег **<td>**. Якщо в стилях для селектора **TABLE** задати колір тексту, то він автоматично встановлюється для вмісту осередків, як показано в прикладі 1.65.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.65. Наслідування параметрів кольору | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Наслідуванняе</title>  <style type="text/css"> TABLE {  color: red; /\* Колір тексту \*/  background: #333; /\* Колір фону таблиці \*/  border: 2px solid red; /\* Червона рамка навколо таблиці \*/  }  </style>  </head>  <body>  <table cellpadding="4" cellspacing="0">  <tr>  <td>Комірка 1</td><td> Комірка 2</td>  </tr>  <tr>  <td> Комірка 3</td><td> Комірка 4</td>  </tr>  </table>  </body>  </html> | | | | | | | | | |

В даному прикладі для всієї таблиці встановлений червоний колір тексту, тому в осередках він також застосовується, оскільки тег **<td>** успадковує властивості тега **<table>**. При цьому слід розуміти, що не всі стильові властивості успадковуються. Так, border задає рамку навколо таблиці в цілому, але ніяк не навколо осередків. Аналогічно не успадковується значення властивості background. Тоді чому колір фону у осередків в даному прикладі темний, раз він не успадковується? Справа в тому, що у властивості background як значення за замовчуванням виступає transparent, тобто прозорість. Таким чином колір фону батьківського елементу «проглядає» крізь дочірній елемент.  
  
Щоб визначити, успадковується значення стильової властивості чи ні, потрібно заглянути в довідник за властивостями CSS і подивитися там. Підключати свою інтуїцію в подібному випадку марно, може і підвести.  
  
Спадкування дозволяє задавати значення деяких властивостей одного разу, визначаючи їх для батьків верхнього рівня. Припустимо, потрібно встановити колір і шрифт для основного тексту. Досить скористатися селектором **BODY**, додати для нього бажані властивості, і колір тексту усередині абзаців та інших текстових елементів зміниться автоматично (приклад 1.66).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.66. Параметри тексту для всієї веб-сторінки | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Наслідування</title>  <style type="text/css"> BODY {  font-family: Arial, Helvetica, sans-serif; /\* Гарнітура шрифту \*/ color: navy; /\* Синій колір тексту \*/  }  </style>  </head>  <body>  <p>Колір тексту цього абзацу синій.</p>  </body>  </html> | | | | | | | | | |

В даному прикладі рубаний шрифт і колір тексту абзаців встановлюється за допомогою селектора **BODY**. Завдяки спадкоємства вже немає потреби ставити колір для кожного елемента документа окремо. Однак бувають варіанти, коли потрібно все-таки змінити колір для окремого контейнера. В цьому випадку доведеться перевизначати потрібні параметри явно, як показано в прикладі 1.67.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.67. Зміна властивостей устадкування елемента | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Наслідування</title>  <style type="text/css"> BODY {  font-family: Arial, Helvetica, sans-serif; /\* Гарнітура шрифту \*/ color: navy; /\* Синій колір тексту \*/  }  P.red {  color: maroon; /\* Темно-червоний колір тексту \*/  }  </style>  </head>  <body>  <p>Колір тексту цього абзацу синій.</p>  <p class="red"> А у цього абзацу колір тексту вже інший.</p>  </body>  </html> | | | | | | | | | |

В даному прикладі колір першого абзацу успадковується від селектора **BODY**, а для другого встановлений явно через клас з ім'ям red.

Каскадування

Абревіатура CSS розшифровується як Cascading Style Sheets (каскадні таблиці стилів), де одним з ключових слів виступає «каскад». Під каскадом в даному випадку розуміється одночасне застосування різних стильових правил до елементів документа - за допомогою підключення декількох стильових файлів, успадкування властивостей і інших методів. Щоб в подібній ситуації браузер розумів, яке в підсумку правило застосовувати до елементу, і не виникало конфліктів в поведінці різних браузерів, введені певні пріоритети.  
  
Нижче наведені пріоритети браузерів, якими вони керуються при обробці стильових правил. Чим вище в списку знаходиться пункт, тим нижче його пріоритет, і навпаки.  
  
1. Стиль браузера.  
2. Стиль користувача.  
3. Стиль автора.  
4. Стиль автора з додаванням! Important.  
5. Стиль користувача з додаванням! Important.  
  
Найнижчим пріоритетом має стиль браузера - оформлення, яке за замовчуванням застосовується до елементів веб-сторінки браузером. Це оформлення можна побачити в разі «голого» HTML, коли до документу не додається ніяких стилів.

!important

Ключове слово !important грає роль в тому випадку, коли користувачі підключають свою власну таблицю стилів. Якщо виникає суперечність, коли стиль автора сторінки і користувача для одного і того ж елемента не збігається, то !important дозволяє підвищити пріоритет стилю або його важливість, іншими словами.  
  
При використанні для користувача таблиці стилів або одночасному застосуванні різного стилю автора і користувача до одного і того ж селектору, браузер керується наступним алгоритмом.

!important доданий в авторський стиль - буде застосовуватися стиль автора.  
!important доданий в призначений для користувача стиль - буде застосовуватися стиль користувача.

!important немає як в авторському стилі, так і стилі користувача - буде застосовуватися стиль автора.  
 !important міститься в авторському стилі і стилі користувача - буде застосовуватися стиль користувача.  
  
Синтаксис застосування !important наступний.

Властивість: значення !important;

Спочатку пишеться бажана стильова властивість, потім через двокрапку його значення і в кінці після пробілу ставиться ключове слово! Important.  
Підвищення важливості потрібно не тільки для регулювання пріоритету між авторською та користувальницької таблицею стилів, але і для підвищення специфічності певного селектора.

Специфічність

Якщо до одного елементу одночасно застосовуються суперечливі стильові правила, то вищий пріоритет має правило, у якого значення специфічності селектора більше. Специфічність це деяка умовна величина, що обчислюється в такий спосіб. За кожен ідентифікатор (надалі будемо позначати їх кількість через a) нараховується 100, за кожен клас і псевдоклас (b) нараховується 10, за кожен селектор тега і псевдоелемент (c) нараховується 1. Підсумовуючи зазначені значення, отримаємо значення специфічності для даного селектора.

\* {} /\* a=0 b=0 c=0 -> специфічність = 0 \*/

li {} /\* a=0 b=0 c=1 -> специфічність = 1 \*/ li:first-line {} /\* a=0 b=0 c=2 -> специфічність = 2 \*/ ul li {} /\* a=0 b=0 c=2 -> специфічність = 2 \*/ ul ol+li {} /\* a=0 b=0 c=3 -> специфічність = 3 \*/ ul li.red {} /\* a=0 b=1 c=2 -> специфічність = 12 \*/ li.red.level {} /\* a=0 b=2 c=1 -> специфічність = 21 \*/ #t34 {} /\* a=1 b=0 c=0 -> специфічність = 100 \*/ #content #wrap {} /\* a=2 b=0 c=0 -> специфічність = 200 \*/

Вбудований стиль, який додається до тегу через атрибут style, має специфічність 1000, тому завжди перекриває пов'язані і глобальні стилі. Однак додавання !important перекриває в тому числі і вбудовані стилі.

Якщо два селектора мають однакову специфічність, то застосовуватися буде той стиль, що визначений в коді нижче. У прикладі 1.68 показано, як впливає специфічність на стиль елементів списку.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.68. Колір списку | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Список</title>  <style type="text/css"> #menu ul li {  color: green;  }  .two { color: red;  }  </style>  </head>  <body>  <div id="menu">  <ul>  <li>Перший</li>  <li class="two">Другий</li>  <li>Третій</li>  </ul>  </div>  </body>  </html> | | | | | | | | | |

В даному прикладі колір тексту списку заданий зеленим, а другий пункт списку з допомогою класу two виділено червоним кольором. Обчислюємо специфічність селектора **#menu ul li** - один ідентифікатор (100) і два тега (2) в сумі дають значення 102, а селектор **.two** матиме значення специфічності 20, що явно менше. Тому текст фарбуватися червоним кольором не буде. Щоб виправити ситуацію, необхідно або знизити специфічність першого селектора, або підвищити специфічність другого (приклад 1.69).

Приклад 1.69. Зміна специфічності

/\* Знижуємо специфічність першого селектора \*/ ul li {...} /\* Прибираємо ідентифікатор \*/

.two {...}

/\* Підвищуємо специфічність другого селектора \*/ #menu ul li {...}

#menu .two {...} /\* Добавляємо ідентифікатор \*/

#menu ul li {...}

.two { color: red !important; } /\* Добавляємо !important \*/

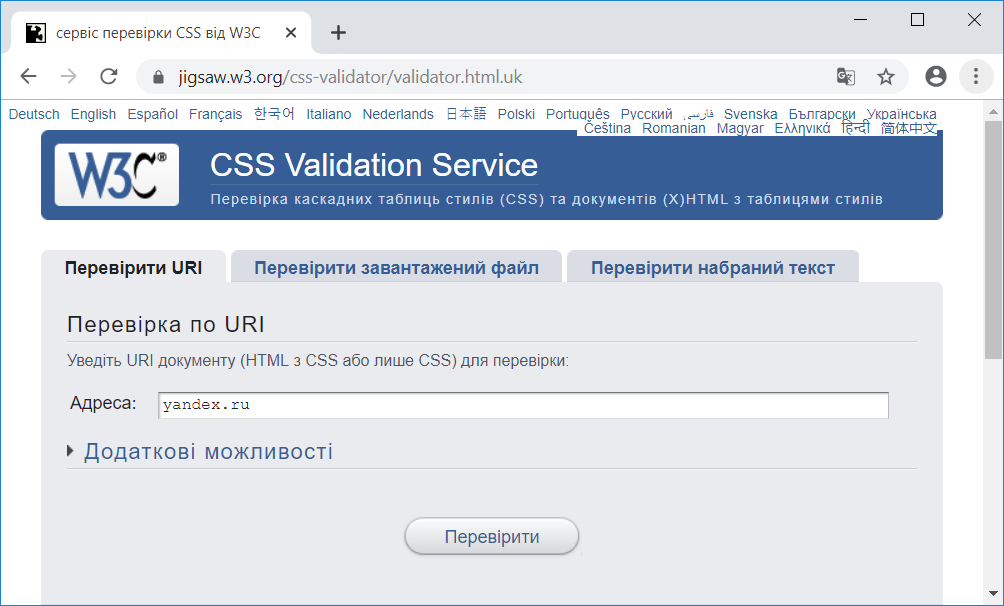
Додавання ідентифікатора використовується не тільки для зміни специфічності селектора, але і для застосування стилю тільки до зазначеного списку. Тому зниження специфічності за рахунок прибирання ідентифікатора застосовується рідко, в основному, підвищується специфічність потрібного селектора.

Валідація CSS

Валідацією називається перевірка CSS-коду на відповідність специфікації CSS2.1 або CSS3. Відповідно, коректний код, який не містить помилок, називається валідний, а не задовольняє специфікації - невалідний. Найбільш зручно робити перевірку коду через сайт <http://jigsaw.w3.org/css-validator/>,за допомогою цього сервісу можна вказати адресу документа, завантажити файл або перевірити набраний текст. Великим плюсом сервісу є підтримка російської та української мови.

Перевірити URL

Ця вкладка дозволяє вказувати адресу сторінки розміщеної в Інтернеті. Протокол http: // годі й писати, він буде доданий автоматично (рис. 1.42).

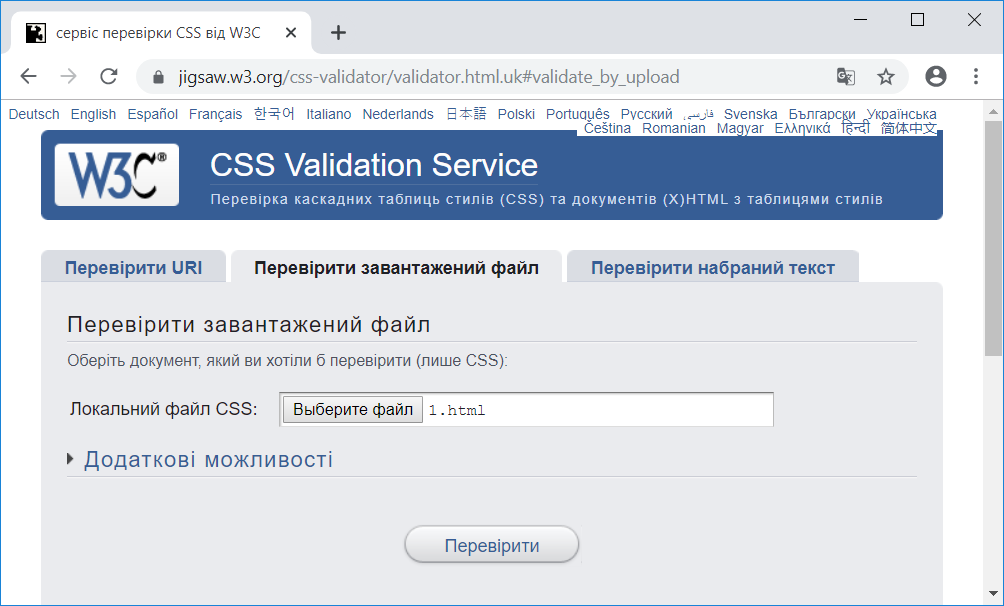


*Рис. 1.42. Перевірка документа за адресою*

Після введення адреси натисніть на кнопку «Перевірити» і з'явиться одна з двох написів: «Вітаємо! Помилок не виявлено» в разі успіху або «На жаль, ми виявили наступні помилки» при невалідності коду.  
Повідомлення про помилки або попередження містять номер рядка, селектор і опис помилки.

Перевірити завантажений файл

Ця вкладка дозволяє завантажити HTML або CSS-файл і перевірити його на наявність помилок (рис. 1.43).

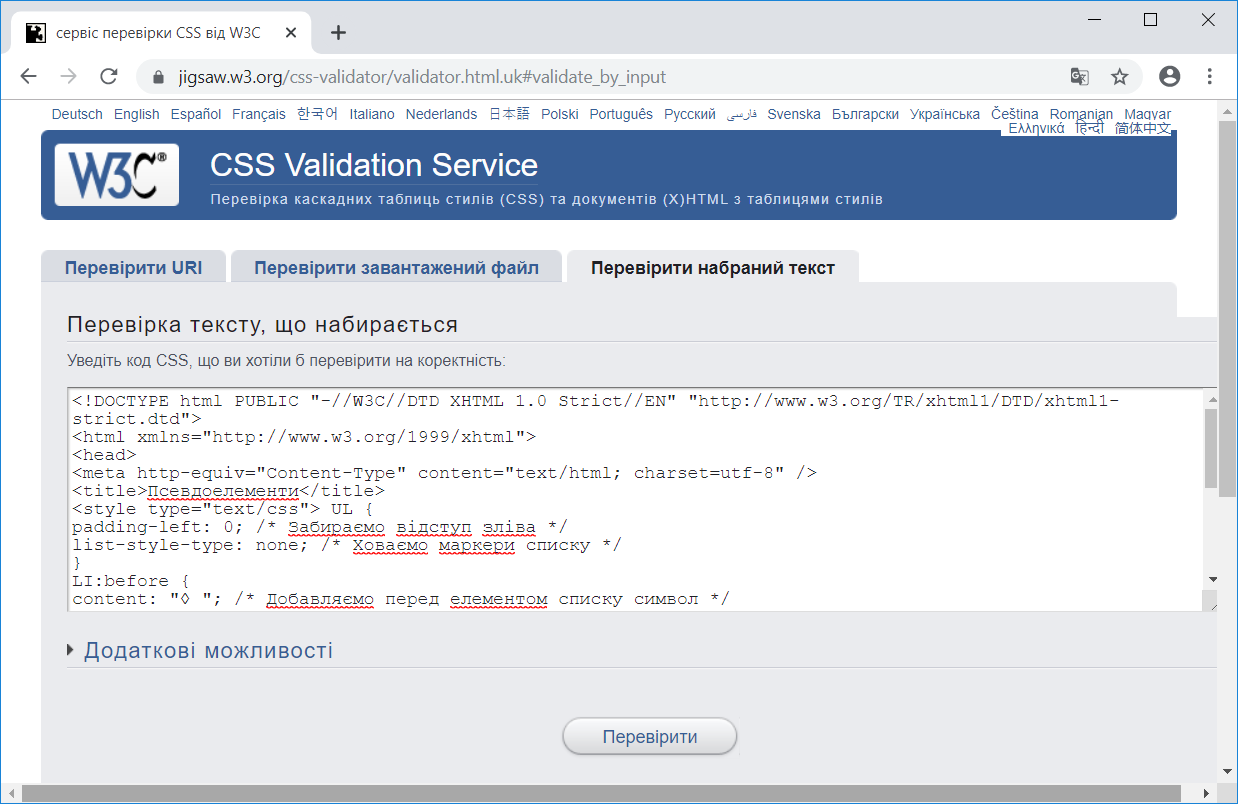


*Рис. 1.43. Перевірка файлу при його загрузці*

Сервіс автоматично розпізнає тип файлу і якщо вказано HTML-документ, виокремлює з нього стиль для валідації.

Перевірити набраний текст

Остання вкладка призначена для безпосереднього введення HTML або CSS-коду, при цьому перевірці буде підданий тільки стиль (рис. 1.44).

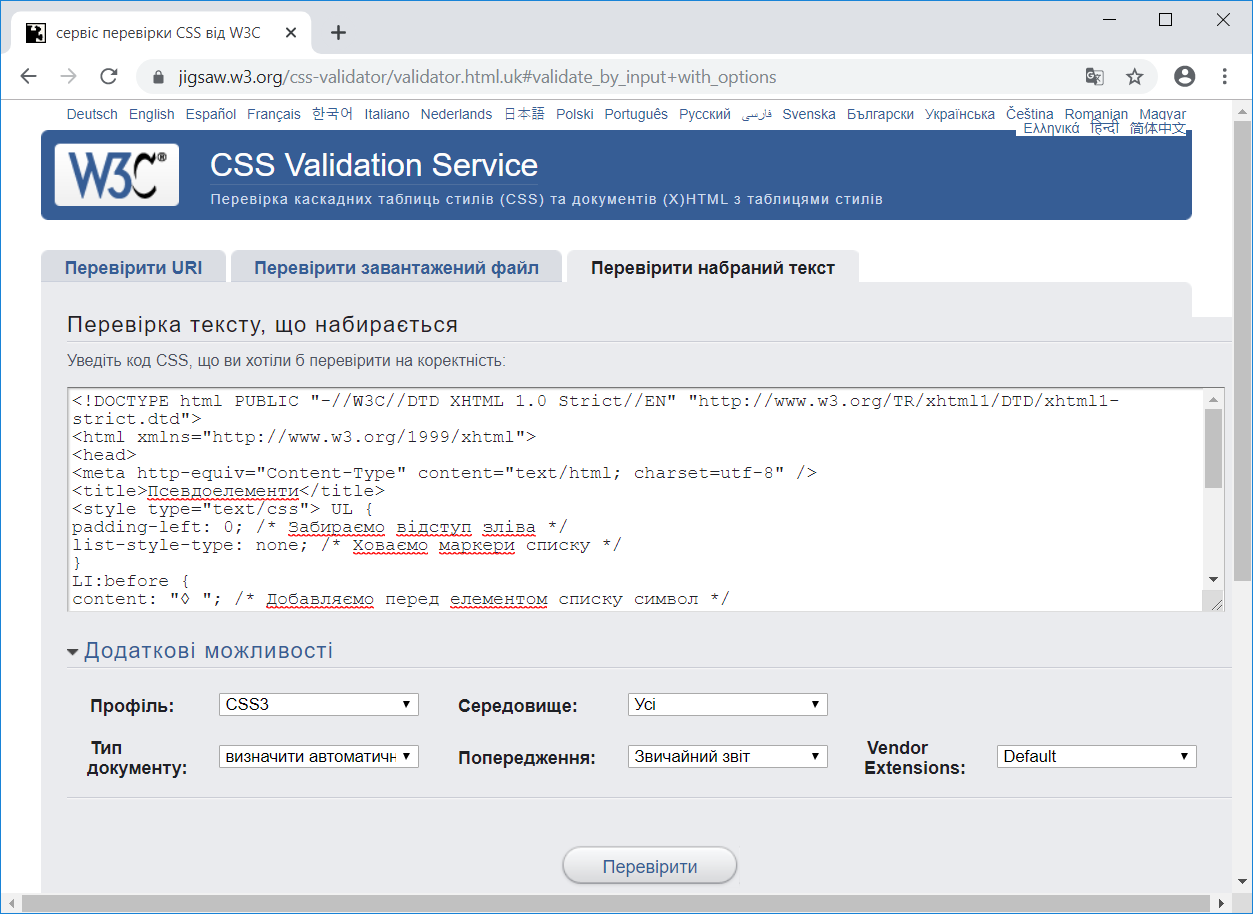


*Рис. 1.44. Перевірка введеного коду*

Цей варіант видається найбільш зручним для проведення різних експериментів над кодом або швидкої перевірки невеликих фрагментів.

Вибір версії CSS

В CSS3 додано багато нових стильових властивостей в порівнянні з попередньою версією, тому проводити перевірку коду слід з урахуванням версії. За замовчуванням в сервісі зазначений CSS2.1, так що якщо ви хочете перевірити код на відповідність CSS3, це слід вказати явно. Для цього клацніть по тексту «Додаткові можливості» і в блоці зі списку «Профіль» виберіть CSS3 (рис. 1.45).



*Рис. 1.45. Вказані версії CSS для перевірки*

Ідентифікатори і класи

Періодично піднімається суперечка про доцільність використання ідентифікаторів в верстці. Основний аргумент полягає в тому, що ідентифікатори призначені для доступу і управління елементами веб-сторінки за допомогою скриптів, а для зміни стилів елементів повинні застосовуватися виключно класи. Насправді немає різниці, через що задавати стилі, але слід пам'ятати про деякі особливості ідентифікаторів і класів, а також підводні камені, які можуть чекати розробників.  
  
Для початку перерахуємо характерні ознаки цих селекторів.  
  
**Ідентифікатори**  
  
У коді документа кожен ідентифікатор унікальний і повинен бути включений лише один раз. Ім'я ідентифікатора чутливе до регістру.  
Через метод getElementById можна отримати доступ до елементу за його ідентифікатором і змінити властивості елемента.  
Стиль для ідентифікатора має пріоритет вище, ніж у класів.  
  
**Класи**  
Класи можуть використовуватися в коді неодноразово. Імена класів чутливі до регістру.  
Класи можна комбінувати між собою, додаючи кілька класів до одного тегу.

Ідентифікатори

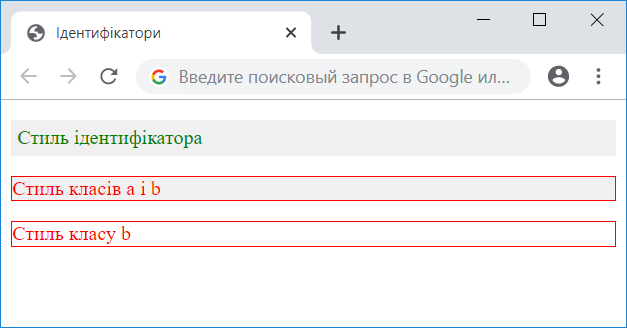
Якщо під час роботи веб-сторінки потрібно змінити стиль деяких елементів «на льоту» або виводити всередині них будь-якої текст, з ідентифікаторами це робиться набагато простіше. Звернення до елементу відбувається через метод getElementById, параметром якого служить ім'я ідентифікатора. У прикладі 1.70 до текстового поля форми додається ідентифікатор з ім'ям userName, потім за допомогою функції JavaScript робиться перевірка на те, що користувач ввів в це поле будь-якої текст. Якщо ніякого тексту немає, але кнопка Submit натиснута, виводиться повідомлення всередині тега з ідентифікатором msg. Якщо все правильно, дані форми відправляються за адресою, вказаною атрибутом action

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.70. Перевірка даних форм | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Перевірка форми</title>  <script type="text/javascript"> function validForm(f) {  user = document.getElementById("userName");  if(user.value == "") document.getElementById("msg").innerHTML = 'Будь ласка, введіть і’мя.'; else f.submit();  }  </script>  </head>  <body>  <form action="handler.php" onsubmit="validForm(this); return false">  <p>Введіть своє і’мя:</p>  <div id="msg"></div>  <p><input type="text" id="userName" name="user" /></p>  <p><input type="submit" /></p>  </form>  </body>  </html> | | | | | | | | | |

Оскільки ідентифікатори чутливі до регістру, має значення їх однотипне написання. усередині тега  
<Input> використовується ім'я userName, його ж слід вказати і в методі getElementById. При помилковому написанні, наприклад, username, скрипт перестане працювати, як потрібно.  
В наведеному вище прикладі стилі взагалі ніякої участі не брали, самі ідентифікатори були потрібні для роботи скриптів. При використанні в CSS слід враховувати, що ідентифікатори мають високий пріоритет в порівнянні з класами. Пояснимо це на прикладі 1.71.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Приклад 1.71. Поєднання стилів | XHTML 1.0 | CSS 2.1 | IE 7 | IE 8 | IE 9 | Cr 8 | Op 11 | Sa 5 | Fx 3.6 |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www.w3.org/TR/xhtml1/DTD/xhtml1](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)-[strict.dtd">](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)  <html [xmlns="http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml)  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Ідентифікатори</title>  <style type="text/css"> #A, .a {  border: none; background: #f0f0f0; color: green; padding: 5px;  }  .b {  border: 1px solid red; color: red;  padding: 0;  }  </style>  </head>  <body>  <p id="A" class="b">Стиль ідентифікатора </p>  <p class="a b">Стиль класів a і b</p>  <p class="b">Стиль класу b</p>  </body>  </html> | | | | | | | | | |

Для першого абзацу встановлюється стиль від ідентифікатора A і класу b, властивості яких суперечать один одному. При цьому стиль класу буде ігноруватися через особливості каскадирования і специфічності. Для другого абзацу стиль задається через класи a і b одночасно. Пріоритет у класів однаковий, значить, у випадку якщо будуть задіяні ті властивості, які вказані в стилі нижче. До останнього абзацу застосовується стиль тільки від класу b. На рис. 1.46 показаний результат застосування стилів.



*Рис. 1.46. Використання стилів тексту*

Специфічність в каскадуванні починає грати роль при розростанні стильового файлу за рахунок збільшення числа селекторів, що характерно для великих і складних сайтів. Щоб стиль застосовувався коректно, необхідно грамотно управляти специфічністю селекторів шляхом використання ідентифікаторів, підвищення важливості через !important, порядком проходження властивостей.

Класи

Оскільки до елементу одночасно можна додавати більше одного класу, це дозволяє завести кілька універсальних класів зі стильовими властивостями на всі випадки і включати їх до тегів при необхідності.  
Припустимо, що більшість блоків на сторінці мають закруглені куточки, причому деякі блоки ще мають червону рамку, а деякі ні. У цьому випадку можемо написати такий стиль (приклад 1.72).

Приклад 1.72. Використання класів

.r, .b { padding: 10px;

background: #FCE3EE;

}

.r {

border-radius: 8px;

-webkit-border-radius: 8px;

-moz-border-radius: 8px

}

.b { border: 1px solid #ED1C24; }

.n { border: none; }

### Вказуючи різні класи в атрибуті class ми можемо комбінувати набір стильових властивостей і отримати таким чином блоки з рамкою, блоки без рамки, із закругленими або прямими кутами. Це дещо схоже на групування селектор, але має більшу гнучкість.