Cascade Style Sheets

CSS (скорочення від англійських слів Cascade Style Sheets або каскадні таблиці стилів) властивості задають для web-документа його оформлення і поведінку блоків. Саме форматування сторінки за допомогою каскадних таблиць стилів забирає левову частку процесу верстки.

Властивості CSS умовно можна поділити на такі великі групи властивостей:

- задають позиціонування об’єктів;
- пов’язані із встановленням розмірів і відступів;
- керуванням «потоком» документа;
- форматування тексту;
- декоративне оформлення елементів;
- елементи анімації та динамічних ефектів.

Окремо варто виділити псевдоелементи та псевдокласи.

Синтаксис CSS є достатньо простим і його умовно можна подати так, як показано у прикладі 4.

Приклад 4.

```
селектор {
    властивість: значення;
    властивість: значення;
....
    властивість: значення;
}
```

Селектором є певне правило, яке визначає зовнішній вигляд та поведінку тега HTML, до якого застосовується. Опис, зрозумілий для браузера, подається між знаками відкритої і закритої фігурних дужок за допомогою запису пар `властивість – значення`. Кожен опис властивості завершується знаком крапкою із комою, а властивість і її значення розділяється двокрапкою.

Для коментування частини коду використовується наступний запис (Приклад 5).

Приклад 5.

```
/* Це коментар у CSS*/
```

Селектори можуть задаватися наступним чином.

Селектор, який задається іменем тега, визначає правило, яке буде застосовуватися для всіх тегів відповідної назви. Запис `p{…}` означає, що всі налаштування оформлення будуть застосовані до всіх тегів `p` заданої сторінки.

Селектор, запис якого починається із точки (клас), застосовується до всіх тегів HTML документу, які мають у описі атрибут class = ″назва_класу″ (приклад 6).

Приклад 6.

```
CSS
.redtext{
    color: red;
}
HTML
<p class = "redtext">Text</p>
```

Селектор id (задається через із використанням символу # на початку) може використовуватися лише один раз у рамках однієї web-сторінки для форматування або позначення певного унікального елемента HTML документа (приклад 7).

Приклад 7.

```
CSS
#uniq_id{
    text-align: center;
}
HTML
<div id = "uniq_id"></div>
```

Окрім стандартних записів правил оформлення зовнішнього вигляду документу в CSS є можливість комбінувати їх застосування. Розберемо їх докладніше на прикладах.

`a.help{…}` – правило буде застосовуватися до всіх тегів `<a></a>`, які у той же час матимуть атрибут `class = "help"`. Тобто буде виконуватися оформлення всіх тегів із назвою класу `help`.

Схожим чином буде працювати поєднання атрибуту та селектора `id – h2#chp_3{…}`. Стилі будуть застосовуватися до всіх тегів `h2`, у яких буде оголошено `id = "chp_3"`.

Важливою можливістю CSS є наявність вкладеного селектору або ж інша його назва контекстний селектор.

`card a{…}` дозволяє застосовувати правила оформлення до посилання a, що розміщення тега, який має клас `card`.

Аналогічно, може працювати вкладеність і для класів.

`.card .price{…}` виконує оформлення тегу із класом `price`, що розміщений у теге із класом `card`.

Псевдокласи CSS

Псевдоклас CSS – це ключове слово, яке додається до селектора та дозволяє визначити поведінку тега у певних ситуаціях (наприклад наведення вказівника миші та елемент сторінки). Серед ключових їх завдань є підсилення роботи звичних селекторів, дозволяють вибирати компоненти Web-документа із урахуванням стану і розмірів по відношенню до інших елементів.

Синтаксично псевдокласи задаються через двокрапку, після запису основного селектора (приклад 8).

Приклад 8.

```
.red_link:hover{…}
```

До найбільш вживаних псевдокласів варто віднести такі: `:link`, `:hover`, `:active`, `:focuse`, `:visited`, `:root`, `:first-child`, `:last-child`, `:nth-child`, `:valid`, `:invalid`, `:required`, `:optional`, `:checked`, `:disable`, `:enabled`.

Розглянемо докладніше кожен із них.

`:link` – дозволяє застосовувати набір правил до елементів, які виконують роль посилань на інші ресурси.

Приклад 9.

```
a:link {color: red;}
.sample: {font-size: 16px;}
```

`:hover` – елемент змінює свій зовнішній вигляд, тоді коли користувач наводить курсор миші на заданий елемент.

`:active` – застосовує оформлення до HTML елемента у той момент, коли він активується користувачем.

`:focus` – спрацьовує тоді, коли елемент активується для внесення певної інформації. Досить часто такі налаштування застосовуються до полів форми.

:visited – вибирає всі посилання, які були відвідані користувачем.

:root – визначає кореневий елемент дерева документа. Для web-сторінки це буде тег html. Цей псевдоклас часто використовують для задання CSS змінних.

:first-child – застосовує правила оформлення до першого елемента у заданому контейнері.

:last-child – працює аналогічно до попереднього, але у цьому випадку визначається останній елемент у заданому контейнері.

:nth-child – дозволяє знаходити один або більше елементів у контейнері.

:valid – застосовує описані правила оформлення до всіх полів введення, які проходять валідацію.

:invalid – визначає всі поля, які не проходять валідацію.

:required – вибирає всі теги <input>, які мають атрибут required.

:optional – вибирає всі теги <input>, які не мають атрибут required. Він дозволяє формам легко відзначати необов'язкові поля, і надавати їм відповідні стилі.

:checked – дозволяє застосовувати стилі оформлення до всіх елементів форми, які мають тип radio, checkbox або option.

:disabled – визначає всі елементи Web-сторінки, які є вимкнутими. Вимкнутими елементами вважаються ті, які не можуть бути активованими (їх не вибрати або натиснути на них кнопкою миші).

:enabled – діє аналогічно до попереднього, але у цьому випадку вибирає на сторінці чи у заданому контейнері всі активні елементи.

Окрім описаних вище існують такі як: :any, :any-link, :default, :defined, :dir(), :empty, :first, :first-of-type, :fullscreen, :indeterminate, :in-range, :lang(), :last-of-type, :left, :not(), :nth-last-child(), :nth-last-of-type(), :nth-of-type(), :only-child, :only-of-type, :out-of-range, :read-only, :read-write, :right, :scope, :target. Проте вони не часто використовуються для верстки сучасних web-документів.

Псевдоелементи CSS та їх застосування

Окрім псевдокласів у специфікації CSS присутнє таке поняття як псевдоелементи. Вони дозволяють утворювати нові віртуальні теги та стилізувати їх. Для їх оголошення використовують подвійну двокрапку. Серед найбільш використовуваних із них варто назвати такі як: `::after`, `::before`, `::first-letter`, `::first-line`, `::selection` тощо. Розглянемо їх особливості докладніше

`::after` – дозволяє створити псевдоелемент, який є останнім нащадком вибраного елемента. За замовчуванням є рядковим елементом.

`::before` – створює псевдоелемент, який є першим нащадком обраного елемента. Часто використовується для додавання додаткового контенту в елемент за допомогою властивості content. За замовчування є рядковим.

`::first-letter` – надає можливість застосовувати стилі до першої літери першого рядка контейнерного елемента, але тільки якщо немає іншого попередньо розміщеного зображення або таблиці.

`::first-line` – схожий за дією до попереднього, але дозволяє застосовувати стилі до першого рядка тексту контейнерного елемента. Варто зауважити, що довжина першого рядка залежить від багатьох факторів, включаючи ширину елемента, ширину документа і розмір шрифту тексту.

`::selection` – дозволяє застосувати стилі до частини документа, який був виділений користувачем (наприклад, за допомогою миші).

CSS властивості

У сучасних стандартах CSS3 існує дуже багато властивостей, які дозволяють форматувати та оформляти web-сторінки. Їх умовно можна розділити на такі великі групи: CSS властивості для встановлення розмірів та відступів; інструментарій для керуванням потоком; засоби форматування та декорування тексту; властивості для задання анімації

та динамічних ефектів тощо. Звичайно їх значно більше, проте вони діють за одним тими ж принципами.

Тому, перш ніж описати їх, розкриємо основні рекомендації, які варто дотримуватися при верстці HTML документів.

1. Якщо до одного і того ж тегу застосовуються декілька CSS правил (можуть бути різні класи), то у такому випадку властивості комбінуються.

2. У випадку конфлікту селекторів (наприклад у двох різних класах присутня однакова властивість із різними значеннями) браузер застосовує правила пріоритетів для визначення селектора, які будуть застосовані для оформлення тегу. Пріоритетність селекторів можна представити таким чином.

Найбільший пріоритет має селектор #id.

Вкладені класи завжди є важливішим за звичний клас

`.main.bage → .bage`

Вкладені теги важливіші за просто описаний тег у CSS

`.main a → a`

Тег із класом є пріоритетнішим за звичний клас

`p.title → .title`

Клас завжди є важливішим за звичний тег, означений у CSS

`.title → p`

Орієнтуючись на них завжди можна точно спрогнозувати поведінку браузера, а отже чітко форматувати саму сторінку сайта.

Основні типи значень у CSS

Важливим є розуміння, які значення можуть набувати властивості у селекторах. Загалом їх ділять на такі основні групи: абсолютні і відносні числові одиниці; ключові слова; кольори; функції та рядки. Дамо коротку характеристику для кожного із них.

Абсолютні величини задають точно значення властивостей, для яких вони є характері, у пікселях

Приклад 10.

```
height: 100px;
```

У цьому випадку вказується висота елемента у 100 пікселів.

Відносні одиниці вимірювання дозволяють вказати значення розмірів по відношенню до батьківського елемента. До них відносять: відсотки (розмір розраховується відносно батьківського елемента, наприклад для тега `body` таким буде `html`, а для нього власне сам браузер – точніше та частина, у якій відображається сам сайт); `vw` – розраховується як одна сота від ширини екрана; `vh` – аналогічна до попередньої відносна одиниця, тільки у цьому випадку за основу береться висота екрана; `em` – одиниця вимірювання для шрифтів, яка задає розмір тексту по відношенню до розміру шрифту батьківського елементу; `rem` – працює аналогічно, але розрахунок відбувається на основі базового розміру шрифту, який встановлений для тегу `body`.

Ключові слова задають певне значення, яке характеризує стан певної властивості (наприклад `display: flex;`).

Кольори у CSS можна описати трьома способами: за допомогою ключових слів, які позначають колір (наприклад `color:red;` – задає червоний колір тексту); у вигляді шістнадцяткового значення (`color:#23aaff;`); за допомогою ключого слова `rgb(255,40,50)`, де у дужках подаються значення трьох кольорів (червоного, зеленого і синього) за допомогою числа із інтервалу від 0 до 255 включно. Більш розширений формат `rgba`, дозволяє задати також прозорість четвертим числом із значенням від 0 до 1. Нуль відповідає відсутністю кольору, а 1 – повному його відображенню.

Також є можливість задати колір за допомогою моделі `hsl` або `hsla`. Ключовою відмінністю є те, що колір задається трьома значеннями у відсотках.

Функції CSS є спеціалізованими підпрограмами, які дозволяють генерувати та обраховувати значення. Найбільш застосовуваним із них є: `attr()`, `calc()`, `linear-gradient()`, `radial-gradient()`, `repeating-linear-gradient()`, `repeating-radial-gradient()` тощо.

Рядкові значення найчастіше зустрічаються при заданні сімейства шрифтів або у властивості content. Наприклад `font-family: ”Arial”, sans serif;` або ж content: `“Hello World!!!”`.

Спеціалізовані директиви CSS

Окремо розглянемо директиви CSS. Вони являються спеціалізованими правилами, які дозволяють змінити відображення або ж поведінку елементів сторінки у певній, завчасно визначеній ситуації.

Директиви розпочинаються із запису символа @, після якого записується одне із ключових слів. Їх ділять на стандартні правила та вкладені. Коротко охарактеризуємо перші.

`@charset` – задає кодування, яке у подальшому буде використовуватися браузером. Наприклад правило `@charset "UTF-8";` вказує, що сторінка використовує кодування символів UTF-8.

`@import` – дозволяє завантажити та використовувати зовнішній CSS файл (`@import "new.css";`).

`@namespace` дозволяє застосовувати CSS для таких мов розмітки як XML та XHTML.

Вкладені директиви містять у собі додаткові оголошення, які можуть бути описом певних завчасно визначених ситуацій.

`@document` – спеціалізована директива, яка визначає умови застосування стилей для всієї сторінки

Приклад 11.

```
@document(
    url(https://samplpe.com/)
    regexp("https:// ….”))
    {
        body{font-family: Arial;}
    }
```

`@font-face` – дозволяє завантажувати користувацькі шрифти.

Приклад 12.

```
@font-face{
    font-family:'MyWebFont';
    url('Fonts/MyWebFont.woff2')
}
```

`@media` – містить умовні інструкції, які застосовують задані стилі, у залежності від певної умови.

Приклад 13.

```
@media (max-width: 600px){
    .screen{
        display:none;
    }
}
```

Останній приклад скрипту дозволяє вимикати властивість `display` для значень ширини екрану 600px і менше.

Властивості керування потоком документу, заданням розмірів елементів

Поток документу HTML називається процес відображення елементів Web-сторінки у вікні браузера. Керування ним відбувається тільки за допомогою каскадних таблиць стилів. Основною одиницею для відображення матеріалу (текстового, графічного, відео та аудіо) є бокс – прямокутна область сторінки, у якій розміщується контент.

Поведінку блока на сторінці задається ключовою властивістю `display`. Серед основних значень, яких може набувати ця характеристика, є такі: `block, inline-block, inline, flex, grid, table`. Розглянемо їх детальніше.

Значення `block`. За замовчуванням браузер висловлює таке значення для тегів `header, section, footer, div, h1, …, h6, p, ul, ol` тощо. Для всіх тегів HTML, до яких застосовується властивість із описаним значенням характерні такі особливості.

1. Висота боксу підлаштовується під висоту контенту.
2. Займає весь доступний простір по ширині.
3. Для такого тегу є можливість задати значення висоти, ширини, внутрішні та зовнішні відступи.
4. Обов’язковим є примусове перенесення слів на новий рядок, якщо у попередньому речення не поміщається повністю.

Основні властивості, які задають параметри бокса, якщо для нього встановлено значення властивості `display: block`.

`width` – задає ширину контенту (зазвичай використовуються такі одиниці як px, %, auto).

`height` – висота контенту (px, auto).

padding – внутрішні відступи у боксі від меж до контенту. На рівні із цією властивістю також часто використовуються властивості padding-left, padding-right, padding-top, padding-bottom. Вони дозволяють задати внутрішній відступ з однієї сторони (зліва, справа, зверху, знизу).

margin – задає зовнішні відступи. Аналогічно дозволяється використовувати і такі властивості як: margin-left, margin-right, margin-top, margin-bottom. Їх дія аналогічна до попередньої властивості, з тією різницею, що у цьому випадку задається зовнішній відступ із певної сторони.

Властивість border задає для боксу рамки. Їх основними компонентами є: товщина, стиль накреслення і колір. Вони задаються за допомогою властивостей border-width, border-style та border-color. Окремі сторони дозволяється модифікувати за допомогою команд border-left, border-right, border-top, border-bottom.

Значення inline характерне для тегів, які використовуються для форматування фразового контенту. Типовими інструкціями HTML є: span, a, strong, em, b, i, time тощо.

Значення inline-block. Це гібридне значення, яке дозволяє розміщувати елементи із значенням block у рядок. Фактично поєднує особливості застосування першого та другого.

Flex – це більше ніж окреме значення для властивості display. Це взагалі окрема технологія, яка спростила створення гнучких та адаптивних сторінок для сайтів. Вона була розроблена як модель однорядкового макетування елементів інтерфейсу і, що також важливо, як один з методів розподілу простору між елементами у рамках контейнеру flexbox. Для неї характерно гнучкість у розміщенні компонентів на сторінці.

При верстці за допомогою цієї технології варто оперувати елементами із точки зору двох осей – основної та поперечної осей (рис. 2). Основна вісь визначається властивістю flex-direction, а поперечна вісь проходить перпендикулярно їй. Все, що ми робимо із компенентами flexbox, відноситься до цих осей, тому варто з самого початку зрозуміти, як вони працюють.

До основних властивостей, які дозволяють налаштувати розміщення елементів у flex-контейнері, відносять: `flex-direction`, `justify-content`, `flex-wrap`, `order`, `align-items`, `align-self`, `align-content`, `flex-grow`, `flex-shrink` і `flex-basis`.

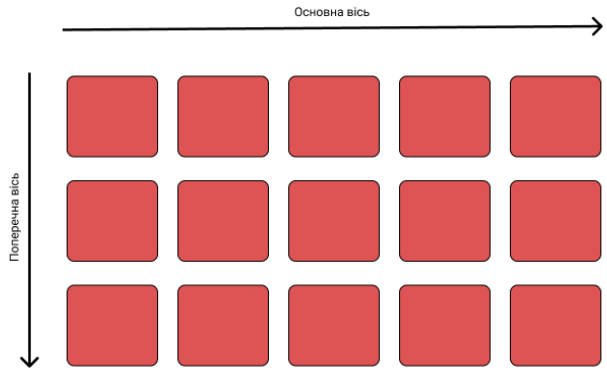

Рис. 2. Основна та поперечна вісь технології Flex

Значення `grid` є прикладом ще однієї інтегрованої технології, яка дозволяє виконувати позиціонування об’єктів на сторінці як по горизонталі так і по вертикалі. По своїй суті вона дуже схожа із таблицею, але має значно гнучкіший та простіший по свої суті функціонал.

Серед основних властивостей, які використовуються у рамках CSS Grid варто назвати: `grid-template-columns`, `grid-template-rows`, `grid-template-areas`, `grid-template`, `grid-column-gap`, `grid-row-gap`, `grid-gap`, `justify-items`, `align-items`, `justify-content`, `align-content`, `grid-auto-columns`, `grid-auto-rows`, `grid-auto-flow`, `grid`, `grid-column-start`, `grid-column-end`, `grid-row-start`, `grid-row-end`, `grid-column`, `grid-row`, `grid-area`, `justify-self`, `align-self`.

Властивості для форматування тексту

Форматування тексту є важливою частиною верстки сучасних web-сайтів. Для налаштування зовнішнього вигляду символів використовують такі властивості.

`font-family` – задає зовнішній вигляд та вказується сімейство шрифтів.

`font-size` – вказує для вибраного елемента розмір шрифту.

`font-style` – задає накреслення для шрифту (може мати такі значення `normal` – звичне накреслення, `italic` – курсив, `oblique` – похилий, `inherit` – наслідування шрифту батьківського елемента).

`font-variant` – вказує на представлення малих літер.

`font-weight` – дозволяє задати насиченість для шрифту (bold – напівжирний та значення від 100 до 900).

`letter-spacing` – відстань між символами.

`word-spacing` – задає додаткову відстань між словами.

`color` – вказується колір тексту.

`text-indent` – встановлюється відступ першого рядка для абзаца.

`text-decoration` – дозволяє представити текст наступним чином: `underline` – підкреслення, `overline` – лінія над текстом, `line-through` – перекреслений текст, `capitalize` – текст записується за допомогою великих літер, але розміром маленьких, `lowercase` – нижній регістр, `uppercase` – верхній регістр.

`vertical-align` – надає можливість записати текст у: `super` – верхньому та `sub` – нижньому індексі.

Властивості CSS для анімації та ефектів

CSS анімації дозволяє реалізувати анімацію переходів (transitions) від однєї конфігурації CSS стилю до іншої. Їх реалізація складаються з двох базових компонентів: стилю, котрий описує суть стилю та набору ключових кадрів (keyframes), які задають початковий та кінцевий стан стилю анімації. Присутня також можливість визначення проміжних точок анімації.

Анімація за елементів сторінки за допомого CSS має ряд важливих переваг над процесом її програмування за допомогою JavaScript.

1. Легкість створення простих анімацій, які при цьому не навантажують систему додатковими скриптами.

2. Анімації добре функціонують, навіть при помірному навантаженні на систему.

3. Процес анімації повністю контролюється браузером, а отже таким чином дозволяєте йому оптимізувати його роботу.

Властивості, які дозволяють описати анімацію у CSS, є наступними: `animation`, `animation-delay`, `animation-direction`, `animation-duration`, `animation-iteration-count`, `animation-name`, `animation-play-state`, `animation-timing-function`, `animation-fill-mode`.

Стилі кодування

При написанні будь-якої програми важливо дотримуватися певного стилю написання коду, який спрощує його розуміння як самим розробником так і сторонніми фахівцями. Стиль кодування включає стандартизацію форматування тексту програми, назви властивостей, порядок їх оголошення тощо.

Варто зауважити, що кожен фахівець самостійно вибирає способи оформлення власного коду програми, яку він пише, проте у рамках роботи команди програмістів, цей стиль узгоджується та стандартизується, з метою уникнення неузгодженостей.

Серед основних кодгайдів, які часто наслідують та дотримуються при розробці, варто вказати на такі: MDO, Google та Idiomatic CSS.