ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«МЕТІНВЕСТ ПОЛІТЕХНІКА»

Конспект лекцій з дисципліни веб-дизайн та веб-розробка: HTML, CSS (частина 1)

за освітньо-професійною програмою
першого (бакалаврського) рівня спеціальності
122 «Компютерні науки»

Запоріжжя 2025

УДК 004.9 (072)
К64

Рекомендовано Науково-методичною радою
ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»
(протокол № 8 від 27.06.2025 р.)

Укладач
Гурковська С.С., канд. техн. наук, доцент.

К64 Конспект лекцій з дисципліни веб-дизайн та веб-розробка: HTML, CSS (частина 1) за освітньо-професійною програмою першого (бакалаврського) рівня спеціальності 122 «Комп'ютерні науки» / уклад. С. С. Гурковська. Запоріжжя : ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА», 2025. 39 с.

Конспект лекцій містить основні теоретичні відомості з дисципліни «Веб-дизайн та веб розробка», перелік рекомендованої літератури.
Рекомендовано для студентів спеціальності 122 Комп'ютерні науки першого (бакалаврського) рівня освіти.

УДК 004.9 (072)

ЗМІСТ

ВСТУП

У сучасному цифровому суспільстві створення та оформлення веб-ресурсів є однією з ключових компетенцій для фахівців у сфері інформаційних технологій. Щодня мільйони користувачів взаємодіють із веб-сайтами, інформаційними порталами, освітніми платформами, електронними сервісами — усі ці ресурси формують візуальне середовище інтернету. Для розробки таких систем необхідне глибоке розуміння структури веб-документів та принципів їх візуального подання. Саме тому вивчення основних мов веб-розробки — HTML (HyperText Markup Language) та CSS (Cascading Style Sheets) — є фундаментальною складовою професійної підготовки в галузі комп'ютерних наук.

Метою даної дисципліни є формування у студентів базових теоретичних знань і практичних навичок, необхідних для створення сучасних, структурованих і візуально привабливих веб-сторінок. У процесі вивчення курсу студенти оволодіють інструментами розмітки та оформлення веб-контенту, зрозуміють логіку побудови інтерфейсів, навчаться застосовувати стилі до різних елементів документа та розробляти макети з урахуванням адаптивності, зручності користування та принципів доступності.

Перший етап вивчення — мова HTML — забезпечує структурну розмітку веб-сторінки. Вона дозволяє описувати зміст документа у вигляді заголовків, абзаців, списків, таблиць, форм, зображень та гіперпосилань. HTML є основою, на якій базується весь веб-документ, і забезпечує логічну побудову контенту. Особлива увага приділяється семантичному HTML, що є надзвичайно важливим з огляду на SEO-оптимізацію, доступність для людей з інвалідністю та ефективну індексацію пошуковими системами.

Другий важливий компонент — мова CSS — відповідає за візуальне оформлення HTML-елементів. За допомогою CSS можна визначати кольори, шрифти, відступи, розміри, розташування елементів на сторінці, межі, фони, а також будувати складні макети за допомогою систем Flexbox або Grid. CSS дозволяє створювати адаптивні інтерфейси, що динамічно підлаштовуються під розмір екрана користувача, забезпечуючи комфортну взаємодію як на комп'ютерах, так і на мобільних пристроях.

Особливістю цього курсу є практична спрямованість: студенти не лише засвоюють теоретичні основи, а й активно застосовують знання під час виконання завдань — створюють власні HTML-документи, оформляють їх за допомогою CSS, експериментують із макетами, структурами і стилями. Такий підхід сприяє розвитку технічного мислення, вмінню вирішувати прикладні задачі, самостійно знаходити та виправляти помилки, а також формуванню естетичного чуття, необхідного для створення зручних, зрозумілих і привабливих інтерфейсів.

Сучасна веб-розробка вимагає не лише знання технологій, а й вміння працювати з ними системно, методично і з увагою до деталей. Саме вивчення HTML і CSS дозволяє закласти міцну основу для подальшого освоєння складніших технологій — таких як JavaScript, серверна розробка, бази даних тощо. Однак вже на рівні HTML і CSS студент отримує потужний інструментарій для самостійного створення повноцінних веб-сторінок і набуває необхідних навичок для подальшого професійного розвитку.

Таким чином, вивчення основ HTML та CSS є необхідним етапом формування компетентного веб-розробника, здатного створювати сучасні цифрові продукти, адаптовані до вимог сьогодення.

Розділ 1. Основи HTML

HTML-документи — це текстові файли, які містять в собі інструкції спеціальної мови розмітки HTML, звані *тегами*. За допомогою тегів можна задавати форматування тексту, додавати вміст мультимедіа (зображення, аудіо, анімацію Flash), створювати гіперпосилання на інші ресурси, оформлювати таблиці та форми для введення даних.

Файли HTML мають розширення **.htm** або **.html**. Код HTML можна створювати і редагувати у звичайному текстовому редакторі (наприклад, Блокнот), а переглядати результати — через веббраузер.

1. Структура HTML-тега

Тег має такий загальний вигляд:

<ім'я_тега атрибут1 атрибут2="значення2" ...>

Тег складається з імені, після якого можуть іти атрибути, записані всередині кутових дужок <>. Атрибути дозволяють керувати поведінкою та виглядом елемента. Вони можуть мати значення, які задаються через знак = і беруться в лапки (одинарні або подвійні). Атрибути розділяються пробілами, а порядок їх розташування не має значення. Імена тегів та атрибутів не залежать від регістру (великими чи малими літерами — неважливо).

Приклад:

<font color="red" face="Arial">

Цей тег FONT визначає стиль тексту: червоний колір та шрифт Arial.

Типи тегів

Існують парні та непарні теги:

- Парні теги мають відкривальний і закривальний тег. Закривальний тег позначається так само, але з косою рискою перед іменем (наприклад,). Текст або інші теги розміщуються між ними. Атрибути вказуються лише у відкривальному тегу. Цей синтаксисбуде виглядати наступним чином:

Початок речення середина речення кінець речення

В браузері буде відображен напис саме в такому вигляді:

Початок речення **середина речення** кінець речення

- Непарні теги не мають закривальної частини. Наприклад, тег
 вставляє розрив рядка. Звичайне натискання клавіші Enter у HTML не створює нового рядка в браузері (так само, як і кілька пробілів або табуляцій поспіль — вони ігноруються).

Наприклад, якщо в html-документі буде написано:

Текст першого абзацу

Текст другого абзацу, то

у браузері відобразиться як:

Текст першого абзацу Текст другого абзацу

Для того, щоб отримати два абзаци потрібно записати
Текст другого абзацу
 Текст другого абзацу
При такій формі запису в браузері відобразиться:
Текст другого абзацу
Текст другого абзацу

Вкладені теги

Парні теги потрібно обов'язково закривати. Якщо, наприклад, не закрити тег , то весь текст після нього буде жирним. При вкладенні тегів їх потрібно закривати в зворотному порядку.

Наприклад, потрібно написати текст жирним курсивом. Тег використовується для виділення тексту курсивом.

В такому разі неправильно буде оформити наступним чином, оскільки не зберігається порядок закриття тегів:

жирний курсив

Правильна форма запису:

жирний курсив

В цьому випадку в браузері відображається: **жирний курсив**

2. HTML-документ та його структура

HTML-документ починається з тегу <html> і складається з двох основних частин: **заголовка** та **тіла**. Заголовок розміщується між тегами <head> і містить назву сторінки та деякі додаткові параметри. Основна частина вмісту – **тіло документа** – розміщується в межах тегу <body> і включає текст та HTML-теги, які відображає браузер.

```
<html>
<head>                    } заголовок документа
  …
</head>
<body>
  …
</body>                   } тело документа
</html>
```

Текст, що міститься в елементі <title>, зазвичай виводиться в заголовку вікна браузера, а також відображається в результатах пошуку в інтернеті.

Приклад простого HTML-документа:

```
<html>
  <head>
    <title>Заголовок</title>
```

```
  </head>
  <body>
    Мій перший <i>HTML-документ!</i><br>
    (це приклад)
  </body>
</html>
```

В браузері

Рис. 1. Результат виконання HTML-документа

3. Кольори в HTML

У HTML кольори можна задавати за допомогою англійських назв кольорів, наприклад:

Таблиця 1

Назва в HTML	**Українська назва**	**RGB-код**
aqua	морська хвиля	#00ffff
black	чорний	#000000
blue	синій	#0000ff
fuchsia	фуксія	#ff00ff
grey	сірий	#808080
green	зелений	#008000
lime	яскраво-зелений	#00ff00
maroon	темно-бордовий	#800000
navy	темно-синій	#000080
olive	оливковий	#808000
purple	пурпурний	#800080
red	червоний	#ff0000
silver	сріблястий	#c0c0c0
teal	бірюзовий	#008080
white	білий	#ffffff
yellow	жовтий	#ffff00

Однак комп’ютер здатен відображати значно більше кольорів – близько 16 мільйонів. Інший спосіб задання кольору – використання шістнадцяткового коду в системі RGB (Red, Green, Blue).

У цій системі кожен колір утворюється шляхом змішування трьох основних кольорів – червоного, зеленого та синього. Колір задається 6-значним шістнадцятковим числом:

Перші 2 символи – інтенсивність червоного,
Наступні 2 – зеленого,
Останні 2 – синього.
Наприклад:
#FF0000 – яскраво-червоний (red)
#00FF00 – яскраво-зелений (green)
#0000FF – яскраво-синій (blue)
#FFFF00 – жовтий (red + green)
#000000 – чорний
#FFFFFF – білий

Значення 00 означає відсутність компоненти, а FF (255) – її максимум.

Приклади використання кольору в HTML:

1. Задача створити текст сірого кольору:

```
<html>
  <head>
    <title>Заголовок</title>
  </head>
  <body>
    <font color="#808080">Мій перший <i>HTML-документ!</i><br>(це приклад)</font>
  </body>
</html>
```

Рис. 2. Результат виконання HTML-документа

4. Основні теги: робота з текстом, списки

Остання офіційно затверджена версія HTML - HTML5, у якій 107 офіційно підтримуваних тегів. Додатково, у специфікації HTML5 є 12 тегів, які офіційно не підтримуються, але підтримуються в більшості браузерів. Таким чином, всього в HTML5 можна використовувати 119 тегів.

Теги структури документа

Ці теги визначають логічну структуру HTML-документа. Вони не впливають на зовнішній вигляд сторінки у браузері, але є обов'язковими для правильної побудови документа.

- <html>…</html> – охоплює весь вміст сторінки, включаючи теги <head> та <body>.
- <head>…</head> – містить службову інформацію (метадані), наприклад, заголовок сторінки.
- <title>…</title> – задає назву сторінки. Відображається у заголовку вікна браузера.
- <body>…</body> – основна частина: містить усі елементи, що відображаються на сторінці.

В середині тега <body> вказуються атрибути для надання певних властивостей сторінки**:**

- bgcolor="колір" – задає колір фону сторінки.
- text="колір" – задає колір звичайного тексту.

Важливо: Теги <html>, <head>, <title> і <body> використовуються в документі лише один раз.

Теги для роботи з текстом

HTML дозволяє змінювати зовнішній вигляд тексту за допомогою спеціальних тегів:

… – жирний текст

… – курсив

<u>…</u> – підкреслення

_… – підрядковий (нижній) індекс

Приклад:
Код: 101₂ = 5
Вивід: $101_2 = 5$

[…] – надрядковий (верхній) індекс

Приклад:
Код: 2⁸ = 256
Вивід: $2^8 = 256$

* … * – вирівнює текст по центру

… – задає шрифт, розмір та колір тексту
Атрибути тега <font>:

- color="колір" – задає колір тексту

- face="шрифт" – вказує гарнітуру (назву шрифту). Можна вказати кілька шрифтів через кому – використовується перший доступний.
- size="1-7" – розмір шрифту (1 – найменший, 7 – найбільший)

Приклад:
HTML-код
<FONT face="Tahoma" size="2" color="gray">текст</FONT>

Теги для роботи з параграфами та заголовками

* … * – створює параграф (абзац)

Атрибут цього тегу align="..." потрібен для вирівнювання тексту:

- left – по лівому краю (за замовчуванням)
- center – по центру
- right – по правому краю
- justify – вирівнювання по ширині

<hN>…</hN> – заголовки різного рівня. Всього існує 6 рівнів заголовків: <h1> – найбільший, <h6> – найменший.

*
* – перенесення рядка

Тег *<hr>* додає горизонтальну розділювальну лінію на сторінці.

Атрибути:

- align="..." – вирівнювання:
  - left – по лівому краю
  - center – по центру (типово)
  - right – по правому краю
- noshade – лінія буде суцільною, без об'ємного ефекту
- size="N" – товщина лінії у пікселях

Розміри та лінії

width="N" – задає ширину горизонтальної лінії (<hr>) у пікселях або у відсотках від ширини вікна браузера.

Піксель – це найменший елемент зображення на екрані (одна точка). Роздільна здатність екрану вказує кількість пікселів по горизонталі і вертикалі, наприклад: 1024 x 768.

Робота зі списками

HTML дозволяє створювати:

- нумеровані списки (із цифрами або літерами)
- маркіровані списки (з маркерами: кружечками, квадратиками тощо)

Нумерований список:

*
…
* – створює впорядкований список

Атрибути:

- start="N" – задає початкове число для нумерації
- type="..." – визначає тип нумерації:

1 – арабські цифри (1, 2, 3...) *(типово)*

A – великі латинські літери (A, B, C...)
a – малі латинські літери (a, b, c...)
I – великі римські цифри (I, II, III...)
i – малі римські цифри (i, ii, iii...)

Маркірований список:

*
…
* – створює **невпорядкований список**

Атрибут:

- type="..." – визначає вигляд маркера:

disk – закрашений кружок (типово)
circle – порожнє коло
square – квадрат

Елемент списку:

*
…
* – використовується для кожного пункту списку

Атрибути:

- type="..." – індивідуальний стиль маркера або номера для елемента
- value="N" – змінює номер поточного елемента у нумерованому списку

Приклад HTML-коду зі списками:

```
<ol>
<li>арабские цифры (по умолчанию)</li>
<li type="A">прописные буквы</li>
<li type="a">строчные буквы</li>
<li type="I">прописные римские цифры</li>
<li type="i">строчные римские цифры</li>
</ol>
<ul>
<li>диск (по умолчанию)</li>
<li type="circle">окружность</li>
<li type="square">квадрат</li>
</ul>
```

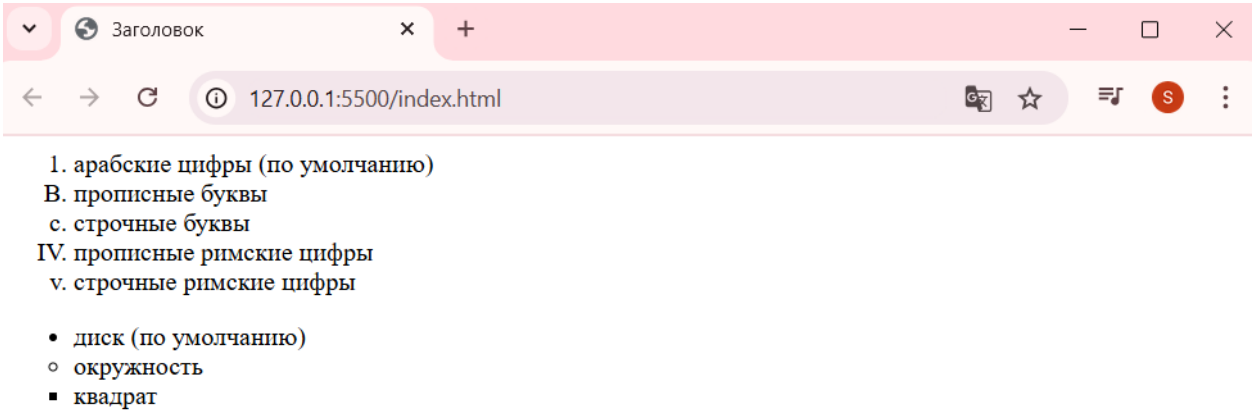

Рис. 3 Результат виконання HTML-документа

5. Гіперпосилання в HTML

Для створення гіперпосилань у HTML використовується тег *<a>...*, де всередині між відкриваючим і закриваючим тегом розміщується текст, на який користувач може натиснути. Основним атрибутом цього тега є href, який вказує адресу ресурсу, на який веде посилання.

Існує два основні типи адрес: абсолютні та відносні. Абсолютна адреса містить повний шлях до зовнішнього ресурсу разом із назвою домену. Наприклад:

Пошук у Google

Таке посилання дозволяє перейти на будь-який сайт в Інтернеті, незалежно від того, де знаходиться сам HTML-документ.

Відносна адреса використовується для навігації в межах сайту або локальної структури файлів. Вона задається шляхом відносно поточного документа. Наприклад, щоб перейти до файлу about.html, розташованого в тій самій папці, використовують:

Про нас.

Якщо сторінка знаходиться в іншій папці, шлях буде виглядати як docs/about.html або ../about.html — для переходу на рівень вище.

HTML також дозволяє створювати посилання всередині тієї ж сторінки, що дуже корисно для навігації в довгих документах. Для цього потрібно призначити елементу унікальний ідентифікатор за допомогою атрибута id, а потім створити посилання на цей ідентифікатор із використанням символу решітки. Наприклад:

Перейти до розділу 1

і відповідний блок тексту далі у сторінці позначити так:

<h2 id="section1">Розділ 1</h2>.

Після натискання на посилання браузер автоматично прокручує сторінку до позначеного елементу.

Таким чином, HTML-посилання можуть вести як на зовнішні сайти, так і на інші документи чи окремі частини поточної сторінки. Усі ці можливості дозволяють організувати зручну та логічну навігацію в межах вебресурсу.

6. Вставка зображень у HTML-документ

Одним із важливих елементів веб-сторінки є зображення. Їх додавання здійснюється за допомогою непарного тегу *<img>*. Основним атрибутом є src, який вказує шлях до файлу зображення. Шлях може бути як відносним (стосовно місця розташування HTML-документа), так і абсолютним (із вказівкою на повний URL в Інтернеті). Наприклад:

```
<img src="images/photo.jpg" alt="Моє фото">
<img src="https://example.com/pics/tree.png" alt="Дерево">
```

Найчастіше використовуються наступні формати графічних файлів:

GIF — підтримує до 256 кольорів, добре підходить для невеликих іконок або простих зображень. Підтримує анімацію і прозорість.

JPEG (або JPG) — призначений для фотографій та зображень з великою кількістю кольорів. Використовує стиснення з втратами, що дозволяє зменшити розмір файлу.

PNG — підтримує дві версії: PNG-8 (до 256 кольорів, без анімації) та PNG-24 (до 16 мільйонів кольорів, без втрат якості). Підтримує прозорість.

Характеристика зображення	**Рекомендований формат**
Анімоване зображення	GIF
Маленьке зображення з малою кількістю кольорів	GIF або PNG-8
Прозоре або напівпрозоре зображення	PNG
Фотографія або зображення з багатьма кольорами	JPEG
Скриншоти, зображення з текстом і дрібними деталями	PNG-24

Не рекомендується використовувати формати BMP, TIFF тощо, оскільки вони можуть не підтримуватися браузерами або мати надто великий розмір.

Атрибути тега <img>

src="шлях" — шлях до файлу зображення (обов’язковий атрибут).

alt="текст" — альтернативний текст, який відображається, якщо зображення не завантажується.

align="..." — вирівнювання зображення відносно тексту: top, middle, bottom (за замовчуванням), left, right.

width і height — ширина і висота зображення (у пікселях або %). Рекомендується задавати для прискорення відображення сторінки.

border="N" — товщина рамки навколо зображення (0 – без рамки).

Наприклад:

```
<img src="logo.png" alt="Логотип" width="100" height="50" align="left" border="0">
```

Масштабування зображень

За допомогою width і height можна штучно змінити розмір зображення. Але це може призвести до втрати якості (розмитість, деформація). Краще використовувати зображення, які вже мають потрібні розміри.

Зображення як посилання

HTML дозволяє використовувати зображення як гіперпосилання. Для цього тег <img> розміщується всередині тега <a>:

<img src="icon.png" alt="Перейти" border="0">

У цьому випадку зображення стає клікабельним. Щоб прибрати рамку, слід вказати border="0" у <img>.

Фонове зображення сторінки

Фонове зображення задається через атрибут background тега <body>:

<body background="bg.jpg">

Зображення автоматично повторюється по горизонталі та вертикалі, якщо його розміри менші за розміри вікна браузера. Це корисно для створення текстурного фону.

Приклади

1. Відносне посилання на зображення в папці:
   <img src="images/picture.jpg" alt="Фотографія">

2. Зображення з зовнішнього джерела:
   <img src="https://example.com/pics/tree.gif" alt="Дерево">

3. Фонове зображення на сторінці:
   <html>
   <head>
   <title>Фонове зображення</title>
   </head>
   <body background="bg1.jpg">
   </body>
   </html>

7.Робота з таблицями в HTML

HTML дозволяє створювати таблиці, які використовуються для структурованого представлення інформації. Таблиця в HTML створюється за допомогою тега <table>, а її вміст формується через рядки <tr> та комірки <td> або <th>.

Основні теги для створення таблиць

- *<table>…</table>* — основний тег, що огортає всю таблицю.

- *<tr>…</tr>* — рядок таблиці (table row).
- *<td>…</td>* — звичайна комірка таблиці (table data).
- *<th>…</th>* — комірка заголовка таблиці (table header); текст у ній за замовчуванням відображається жирним і вирівнюється по центру. Приклад найпростішої таблиці:

```
<table>
        <tr>
          <th>Ім’я</th>
          <th>Прізвище</th>
        </tr>
        <tr>
          <td>Анна</td>
          <td>Іванова</td>
        </tr>
        <tr>
          <td>Олег</td>
          <td>Петренко</td>
        </tr>
</table>
```

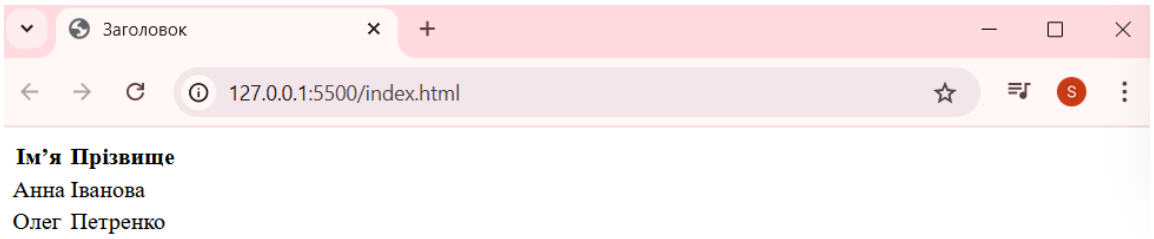

Рис. 4 Результат виконання найпростішого HTML-документа для створення таблиці

Атрибути тега *<table>:*

border="N" — товщина рамки таблиці у пікселях (наприклад, border="1").

width="N" — ширина таблиці у пікселях або у відсотках від ширини вікна.

align="…" — вирівнювання таблиці на сторінці: left, center, right.

cellspacing="N" — відстань між комірками.

cellpadding="N" — відступ між вмістом комірки та її межами.

bgcolor="…" — фоновий колір таблиці.

bordercolor="…" — колір рамки таблиці.

Приклад з атрибутами:

```
<table border="1" width="80%" cellspacing="5" cellpadding="10" align="center" bgcolor="#f0f0f0">
        <tr>
          <th>Ім’я</th>
          <th>Прізвище</th>
        </tr>
```

```
            <tr>
              <td>Анна</td>
              <td>Іванова</td>
            </tr>
            <tr>
              <td>Олег</td>
              <td>Петренко</td>
            </tr>
        </table>
```

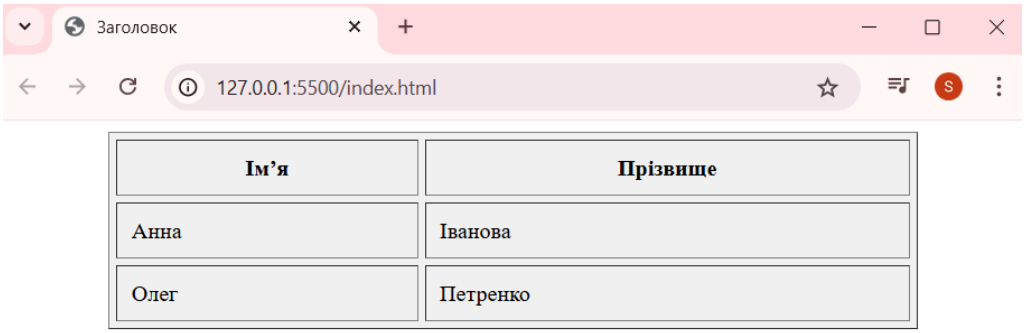

Рис. 5 Результат виконання HTML-документа для створення таблиці

Об'єднання комірок

colspan="N" – розтягує комірку на N стовпчиків ліворуч

```
<html>
  <head>
    <title>Заголовок</title>
  </head>
  <body>
    <table cellpadding="15" border="1">
       <TR><TD colspan="2">1</TD></TR>
       <TR><TD>2</TD><TD>3</TD></TR>
    </table>
   </body>
</html>
```

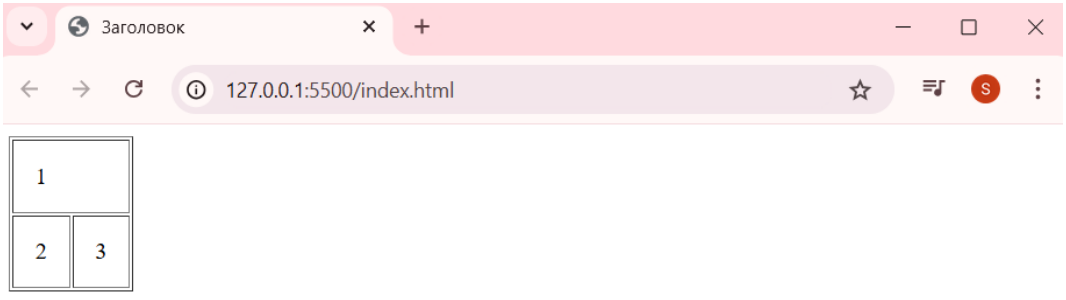

Рис. 6 Результат виконання HTML-документа для створення таблиці з об'єднаними по горизонталі комірками

rowspan="N" – розтягує комірку на N рядків нижче

```
<table cellpadding="15" border="1">
    <TR><TD rowspan="2">1</TD><TD>2</TD></TR>
    <TR><TD>3</TD></TR>
 </table>
```

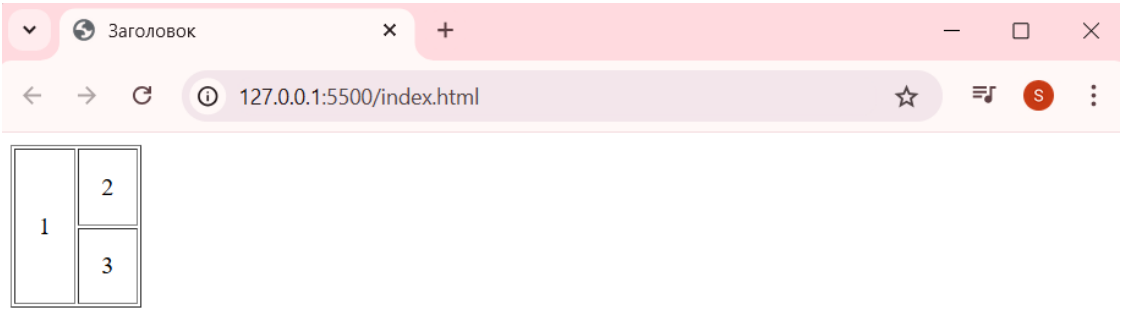

Рис. 7 Результат виконання HTML-документа для створення таблиці з об'єднаними по вертикалі комірками

Ширина таблиці

За замовченням, якщо ширина таблиці не задана окремо, то її максимальная ширина не буде перевіщувати ширини вікна браузера.

Але ширину таблиці можна задавати атрибутом **width**. В цьому випадку браузер розставляє переноси слів в тексті комірки таким чином, щоб відтворити заданий розмір ширини.

```
<table width="100" border="1">
<tr><td> браузер розставляє переноси слів в тексті </td></tr>
</table>
```

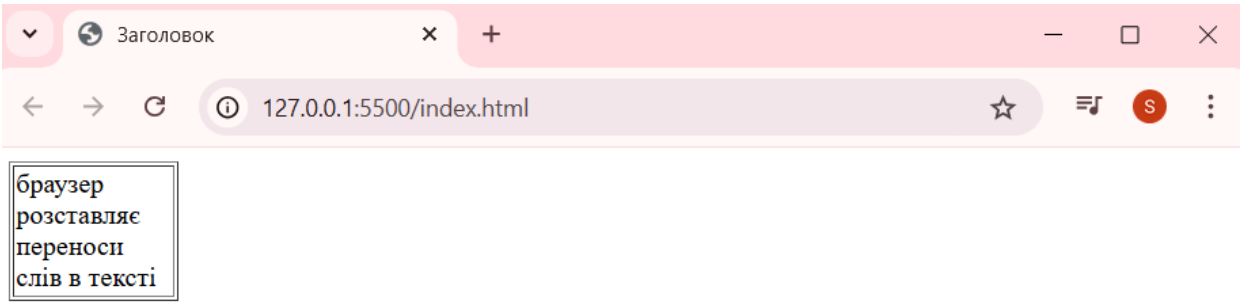

Рис. 8 Результат виконання HTML-документа для створення таблиці з заданою шириною

Семантична структура таблиці

HTML дозволяє логічно структурувати таблицю для покращення читабельності та підтримки доступності:

- *<caption>…</caption>* — заголовок таблиці, розташовується одразу після відкриваючого тега <table>.
- *<thead>* — визначає заголовок таблиці.
- *<tbody>* — основна частина таблиці (тіло).
- *<tfoot>* — нижній підсумковий блок таблиці.

Ці елементи не змінюють вигляд таблиці у звичайному браузері, але важливі для доступності та обробки таблиць скриптами.

Приклад із семантичними тегами:

```
<table border="1">
  <caption>Результати опитування</caption>
  <thead>
    <tr>
      <th>Питання</th>
      <th>Так</th>
      <th>Ні</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Ви задоволені курсом?</td>
      <td>80%</td>
      <td>20%</td>
    </tr>
  </tbody>
</table>
```

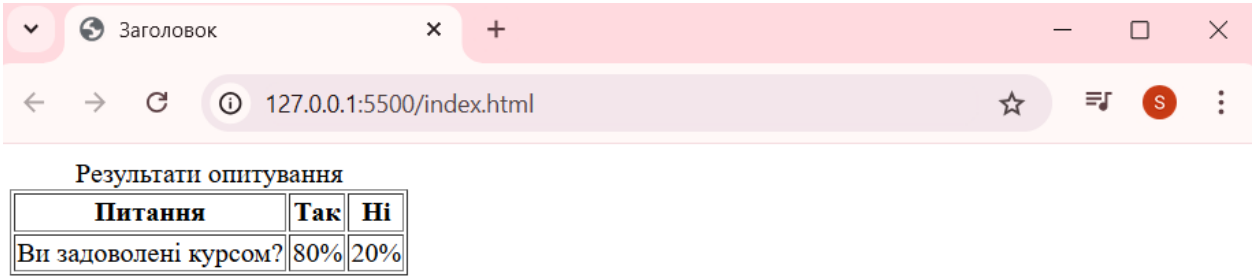

Рис. 9 Результат виконання HTML-документа для створення таблиці з семантичними тегами

Розділ 2. Основи CSS

CSS (Cascading Style Sheets, або «каскадні таблиці стилів») — це мова стилізації, яка використовується для візуального оформлення HTML-документів. HTML визначає структуру сторінки, а CSS — її зовнішній вигляд: кольори, шрифти, розміри, розташування елементів, фони, відступи тощо.

Важливою особливістю CSS є каскадність — можливість поєднувати декілька джерел стилів, які можуть перекривати один одного. Тому порядок оголошення стилів і пріоритетність джерел стилів мають ключове значення.

1.Основи синтаксису CSS

Кожне CSS-правило складається з двох основних частин: селектора і блоку декларацій. Селектор визначає, до якого елементу HTML застосовуватиметься стиль, а блок декларацій містить одну або кілька пар «властивість–значення», що описують, як саме потрібно оформити цей елемент.

Загальна форма запису CSS-правила:

```
селектор {
  властивість1: значення1;
  властивість2: значення2;
  ...
}
```

Наприклад:

```
p {
  color: blue;
  font-size: 14pt;
}
```

У цьому прикладі CSS-правило задає для всіх абзаців (тег <p>) синій колір тексту та розмір шрифту 14 пунктів.

Селектори

Селектор визначає, які HTML-елементи потрібно стилізувати. Існує кілька типів селекторів:

Селектори за тегом — застосовуються до всіх елементів певного типу. Наприклад, h1, p, table тощо.

Селектори за класом — задаються з використанням крапки перед назвою класу. Наприклад, .highlight застосовується до будь-якого елемента з класом highlight.

Селектори за ідентифікатором — задаються з використанням решітки. Наприклад, #main застосовується до елемента з атрибутом id="main".

Складені селектори — дозволяють вибирати елементи за їхнім розташуванням у структурі HTML або поєднувати кілька умов. Наприклад:

div p — всі абзаци всередині блоку div;

ul > li — всі елементи li, що є безпосередніми нащадками списку ul;

h2, h3 — одночасно стилізує заголовки рівня 2 і 3.

Класи та ідентифікатори

Для точного та багаторазового застосування стилів у HTML-документі використовуються класи та ідентифікатори.

Атрибут class задається у будь-якому HTML-елементі й дозволяє застосовувати один і той самий стиль до багатьох елементів. Наприклад:

В index.html:

Увага! Це важливе повідомлення.

В main.css:

```
.notice {
  color: red;
  font-weight: bold;
}
```

Атрибут id використовується для унікального позначення елемента. На відміну від класу, id не повинен повторюватися в межах одного документа:

В index.html

```
<div id="header">Заголовок сторінки</div>
```

В main.css:

```
#header {
  background-color: lightgray;
  padding: 10px;
}
```

Способи підключення CSS-стилів

Існує три основні способи додавання CSS до HTML-документа: зовнішні стилі, стилі рівня документа (внутрішні), та вбудовані стилі.

Зовнішні стилі

Це найбільш рекомендований спосіб. CSS-код розміщується в окремому файлі з розширенням .css, який підключається до HTML-документа за допомогою тегу <link> у розділі <head>.

В index.html:

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

Переваги:

- зручно підтримувати та оновлювати;
- забезпечує повторне використання стилів у кількох документах;
- зменшує обсяг HTML-коду.

Внутрішні стилі (стилі рівня документа)

Описуються всередині тегу <style> у заголовку документа. В index.html:

```
<head>
  <style>
    h1 {
      color: darkblue;
    }
  </style>
</head>
```

Цей метод зручний для коротких стилів, специфічних лише для однієї сторінки.

Вбудовані стилі

Описуються безпосередньо в атрибуті style HTML-елемента. Наприклад (в index.html):

Це текст зі стилем, вбудованим у тег.

Такий підхід не рекомендується у великих проєктах, бо порушує принципи відокремлення структури і стилю, ускладнює підтримку.

Порядок застосування стилів (каскадність)

CSS розшифровується як *Cascading Style Sheets*, тобто *каскадні таблиці стилів*. Каскадність означає, що, якщо до одного й того ж елемента застосовується кілька правил, браузер вибере те, яке має більший пріоритет. Правила оцінюються за специфічністю та порядком розміщення.

Пріоритет правил:

1. **Вбудовані стилі** (style в тегу HTML) мають найвищий пріоритет.
2. **Внутрішні стилі** (в межах <style>) мають середній пріоритет.
3. **Зовнішні стилі** (у файлі .css) мають нижчий пріоритет.

Також важливу роль відіграє специфічність селектора: чим точніше селектор, тим вище його пріоритет. Наприклад, правило з селектором #main має вищу специфічність, ніж правило з селектором .menu, а останнє — вищу, ніж просто div.

Якщо специфічність однакова, перевага надається тому правилу, яке з’являється пізніше у коді.

2.CSS-властивості: розміри, кольори, шрифти, текст

Встановлення розмірів елементів

У CSS ми можемо явно задавати ширину та висоту блоку за допомогою властивостей width і height. Як правило, розміри встановлюються в пікселях (px), відсотках (%), ем (em), rem, pt (пунктів) тощо. Наприклад в main.css:

```
div {
  width: 400px;
  height: 200px;
}
```

Тут ми задаємо блоку ширину 400 пікселів і висоту 200 пікселів.

Якщо використовується значення у відсотках, то воно визначається відносно розміру батьківського елемента:

```
div {
  width: 80%;
```

```
}
```

Це означає, що елемент div займатиме 80% ширини свого контейнера.

Для забезпечення гнучкої адаптації можна також використовувати min-width, max-width, min-height, max-height, які задають мінімальні та максимальні межі розмірів елемента. Наприклад в main.css:

```
img {
  max-width: 100%;
  height: auto;
}
```

Цей підхід дозволяє зображенням масштабуватись відповідно до ширини батьківського блоку, не виходячи за його межі.

Колір у CSS

Кольорове оформлення відіграє важливу роль у візуальному сприйнятті сторінки. Для встановлення кольорів найчастіше використовуються такі CSS-властивості:

- *color* — колір тексту;
- *background-color* — колір фону елемента;
- *border-color* — колір рамки.

Колір можна задати у різних форматах:

- Назва кольору: red, blue, green, black, white тощо;
- Шістнадцяткове значення (hex): #ff0000, #00ff00;
- Функція RGB: rgb(255, 0, 0);
- Функція RGBA (з прозорістю): rgba(0, 0, 255, 0.5).
-

У наступному прикладі заголовок матиме темно-синій колір тексту та світло-блакитне тло. Наприклад в main.css:

```
h1 {
  color: #003366;
  background-color: #ccddff;
}
```

Робота зі шрифтами

Оформлення тексту за допомогою шрифтів відбувається за допомогою ряду властивостей:

- font-family — визначає тип шрифту. Рекомендується вказувати кілька варіантів через кому: від бажаного до універсального.
- font-size — задає розмір шрифту.
- font-weight — визначає товщину шрифту: normal, bold, lighter, bolder, або числові значення від 100 до 900.
- font-style — встановлює стиль шрифту: normal, italic, oblique.
- font-variant — використовується для оформлення малих капітальних літер: small-caps.

Наприклад в main.css:

```
p {
  font-family: "Times New Roman", serif;
  font-size: 16pt;
  font-weight: bold;
  font-style: italic;
}
```

Це правило зробить текст абзацу жирним курсивом розміром 16 пунктів у шрифті Times New Roman або іншому схожому з родини serif.

Форматування тексту

CSS дозволяє контролювати положення, міжрядковий інтервал, відступи та інші параметри тексту за допомогою таких властивостей:

- *text-align* — вирівнювання тексту (left, right, center, justify);
- *text-indent* — відступ першого рядка абзацу;
- *line-height* — міжрядковий інтервал;
- *letter-spacing* — інтервал між літерами;
- *word-spacing* — інтервал між словами;
- *text-transform* — регістр тексту (uppercase, lowercase, capitalize);
- *white-space* — керує відображенням пробілів і перенесенням рядків (normal, nowrap, pre, тощо);
- *text-decoration* — оформлення тексту: підкреслення, перекреслення тощо (underline, line-through, overline, none).

Наприклад в main.css:

```
h2 {
  text-align: center;
  text-transform: uppercase;
  letter-spacing: 2px;
}
```

Це правило зробить заголовок другого рівня великими літерами з відступом між літерами 2 пікселі та вирівняє його по центру.

Інший приклад:

```
p {
  text-indent: 20px;
  line-height: 1.5;
  text-align: justify;
}
```

Абзац з таким оформленням матиме відступ першого рядка 20 пікселів, міжрядковий інтервал 1.5 і вирівнювання по ширині.

Вставлення URL (фонові зображення та шрифти)

Іноді властивості CSS дозволяють використовувати URL-адреси, наприклад, для фонових зображень або підключення шрифтів.

Наприклад в main.css:

```
body {
```

```
  background-image: url("images/bg.jpg");
  background-size: cover;
}
```

Фонове зображення завантажується за вказаною адресою та розтягується на весь фон сторінки.

Інший приклад — підключення вебшрифтів за допомогою правила *@font-face:*

```
@font-face {
  font-family: "CustomFont";
  src: url("fonts/customfont.woff2") format("woff2");}
p {
  font-family: "CustomFont", sans-serif;
}
```

Цей приклад завантажує користувацький шрифт із сервера і застосовує його до абзаців.

3. CSS-властивості: поля, заповнення, границі

У веброзробці надзвичайно важливо мати змогу точно керувати простором навколо елементів сторінки. Цей простір дозволяє організувати вміст, створити структуру, виділити важливі блоки, забезпечити зручність сприйняття інформації. Для цього у CSS передбачено три основні групи властивостей: поля (margin), заповнення (padding) і границі (border).

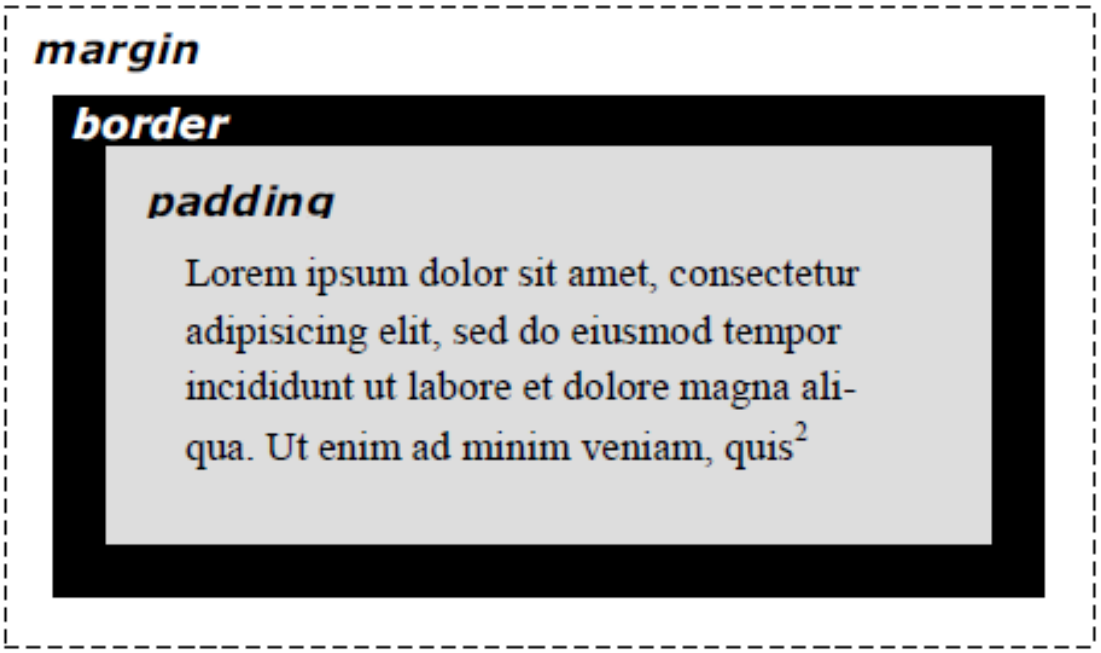

Рис. 10 Бокс елементу

Поля (margin)

Поле — це відстань від межі елемента до сусідніх елементів. Воно «відштовхує» елемент від усього, що його оточує. Поля є зовнішнім

простором і не мають кольору або іншого візуального оформлення — це просто порожній простір.

CSS дозволяє задавати поля окремо для кожної сторони:

- *margin-top* — верхнє поле;
- *margin-right* — праве поле;
- *margin-bottom* — нижнє поле;
- *margin-left* — ліве поле.

Також існує універсальна властивість *margin*, яка дозволяє скорочено записати значення полів одразу для всіх сторін.

Приклад:

```
p {
  margin: 20px;
}
```

Цей запис еквівалентний наступному:

```
p {
  margin-top: 20px;
  margin-right: 20px;
  margin-bottom: 20px;
  margin-left: 20px;
}
```

Інші варіанти скороченого запису:

- margin: 10px 20px; — верхнє і нижнє поле по 10px, ліве і праве — по 20px;
- margin: 5px 10px 15px; — верхнє 5px, ліве і праве — 10px, нижнє — 15px;
- margin: 5px 10px 15px 20px; — верхнє, праве, нижнє, ліве відповідно.

Також можна задати margin: auto; — браузер сам визначить значення, наприклад, для центрування елемента по горизонталі в межах контейнера.

Центрування блоку по центру:

```
div {
  width: 400px;
  margin: 0 auto;
}
```

Заповнення (padding)

Заповнення — це відстань між вмістом елемента і його межею. На відміну від полів, заповнення є внутрішнім простором і може мати фон, тобто є частиною елемента.

Аналогічно до полів, заповнення задається окремо:

- padding-top, padding-right, padding-bottom, padding-left.

Або скорочено:

```
div {
  padding: 10px;
```

```
}
```
Це означає 10 пікселів заповнення з усіх боків.

Скорочений запис працює аналогічно до margin:

- padding: 10px 20px; — зверху й знизу 10px, ліворуч і праворуч 20px;
- padding: 5px 10px 15px; — зверху 5px, ліворуч і праворуч 10px, знизу 15px;
- padding: 5px 10px 15px 20px; — по всіх сторонах у порядку за годинниковою стрілкою.

Приклад:

```
p {
  padding: 15px;
  background-color: #f0f0f0;
}
```

Цей приклад створює відступ між текстом абзацу та його межею, і фон буде видно також у зоні заповнення.

Границі (border)

Границя — це лінія, що обмежує елемент ззовні. Вона відділяє вміст разом із заповненням від полів. У CSS границю можна повністю налаштувати: її товщину, стиль, колір, а також задати індивідуальні параметри для кожної сторони.

Основні властивості:

- border-width — товщина границі;
- border-style — стиль лінії (solid, dotted, dashed, double, groove, ridge, inset, outset, none);
- border-color — колір границі.

Найзручніше використовувати скорочений запис:

```
div {
  border: 2px solid black;
}
```

Цей запис одночасно задає товщину, стиль та колір.

Якщо потрібно задати границі тільки для окремих сторін:

```
p {
  border-top: 1px dashed gray;
  border-bottom: 1px dashed gray;
  border-left: none;
  border-right: none;
}
```

Порядок розрахунку загального розміру елемента

Слід пам’ятати, що властивості width та height визначають **тільки** розмір вмісту. Щоб дізнатися загальний розмір блоку, треба враховувати ще заповнення, границі та поля.

Загальна ширина елемента:

width + padding-left + padding-right + border-left + border-right + margin-left + margin-right

Загальна висота елемента:

height + padding-top + padding-bottom + border-top + border-bottom + margin-top + margin-bottom

Приклад:

```
div {
  width: 300px;
  padding: 10px;
  border: 2px solid black;
  margin: 20px;
}
```

Загальна ширина блоку становитиме:

300 (вміст) + 20*2 (поля) + 10*2 (заповнення) + 2*2 (границі) = 364px

Автоматичне розрахування простору

Для полів і заповнень можна задавати значення auto, що дозволяє браузеру самостійно визначити оптимальну ширину простору. Найчастіше це використовується для центрування блоків.

4. CSS-властивості: фон, оформлення таблиць

У сучасній верстці важливо не лише правильно структурувати елементи, але й надати їм привабливого вигляду. Саме для цього служать властивості, що дозволяють оформити фон елементів, а також покращити вигляд таблиць.

Оформлення фону

Фон (background) — це область, що лежить під вмістом елемента, заповненням (padding) і границями (border). CSS дозволяє за допомогою властивостей фону задати кольори, зображення, їх положення, повторення та інші параметри.

Основні властивості, що відповідають за оформлення фону:

- *background-color* — колір фону;
- *background-image* — фонове зображення;
- *background-repeat* — повторення зображення;
- *background-position* — положення зображення;
- *background-size* — розмір зображення;
- *background-attachment* — спосіб прокручування фону;
- *background* — скорочений запис усіх перелічених властивостей.

Встановлення кольору фону

Найпростіше задання фону — це суцільний колір.

```
body {
  background-color: #f5f5f5;
}
```

Можна використовувати ключові слова (red, blue), шістнадцяткові коди (#ff0000), або функції RGB / RGBA / HSL.

```
div {
  background-color: rgba(255, 0, 0, 0.5); /* напівпрозорий червоний */
}
```

Фонове зображення

За допомогою background-image можна встановити зображення як фон. Шлях до зображення задається у форматі URL:

```
body {
  background-image: url("images/pattern.png");
}
```

За замовчуванням зображення повторюється по горизонталі і вертикалі. Щоб цього уникнути:

```
body {
  background-image: url("bg.jpg");
  background-repeat: no-repeat;
}
```

Можна також вказати повторення лише по одній осі:

```
background-repeat: repeat-x; /* лише по горизонталі */
background-repeat: repeat-y; /* лише по вертикалі */
```

Положення і розмір зображення

Місце розташування фонового зображення визначає властивість background-position.

```
background-position: center center; /* по центру */
background-position: top right;     /* вгорі праворуч */
```

А розмір — background-size:

```
background-size: 100% 100%; /* розтягнути зображення на весь елемент */
background-size: cover;     /* покрити всю площу елемента, зберігаючи пропорції */
background-size: contain;   /* вмістити повністю у межі */
```

Закріплення фону

Щоб зробити фон нерухомим при прокручуванні сторінки використовується наступна властивість:

```
background-attachment: fixed;
```

Скорочений запис фону

Щоб задати всі властивості фону одночасно, використовується наступна форма запису:

```
div {
  background: url("bg.jpg") no-repeat center center / cover #f0f0f0 fixed;
}
```

Оформлення таблиць

Хоча таблиці у HTML здебільшого застосовуються для подання табличних даних (а не для верстки!), їхнє правильне оформлення залишається важливим етапом дизайну.

HTML-таблиця складається з елементів <table>, <tr> (рядок), <td> (комірка), <th> (заголовкова комірка). Стилізація цих елементів виконується через CSS.

Межі таблиці

Щоб таблиця мала чіткі лінії, використовують властивість border. За замовчуванням браузери не відображають внутрішні границі між комірками.

```
table, th, td {
  border: 1px solid black;
}
```

Для того щоб усі границі виглядали як єдина лінія, використовують:

```
table {
  border-collapse: collapse;
}
```

Альтернативно:

```
table {
  border-collapse: separate;
  border-spacing: 10px; /* відстань між комірками */
}
```

Відступи в комірках

Внутрішній простір у комірках задається через padding:

```
td {
  padding: 8px;
}
```

Для вирівнювання тексту використовується властивість text-align:

```
th {
  text-align: center;
}
td {
```

```
  text-align: left;
}
```

Вирівнювання тексту в комірці по вертикалі задається через vertical-align:

```
td {
  vertical-align: middle;
}
```

Фон рядків або комірок

Можна змінювати фон окремих комірок або рядків, наприклад, для чергування кольорів:

```
tr:nth-child(even) {
  background-color: #f2f2f2;
}
```

Приклад оформленої таблиці

В index.html:

```
<table>
  <thead>
    <tr>
      <th>Ім’я</th>
      <th>Прізвище</th>
      <th>Вік</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Іван</td>
      <td>Петренко</td>
      <td>20</td>
    </tr>
    <tr>
      <td>Олена</td>
      <td>Коваль</td>
      <td>22</td>
    </tr>
  </tbody>
</table>
```

В main.css:

```
table {
  border-collapse: collapse;
  width: 100%;
}
```

```
th, td {
  border: 1px solid #999;
  padding: 10px;
  text-align: center;
}

tr:nth-child(even) {
  background-color: #f9f9f9;
}

th {
  background-color: #333;
  color: white;
}
```

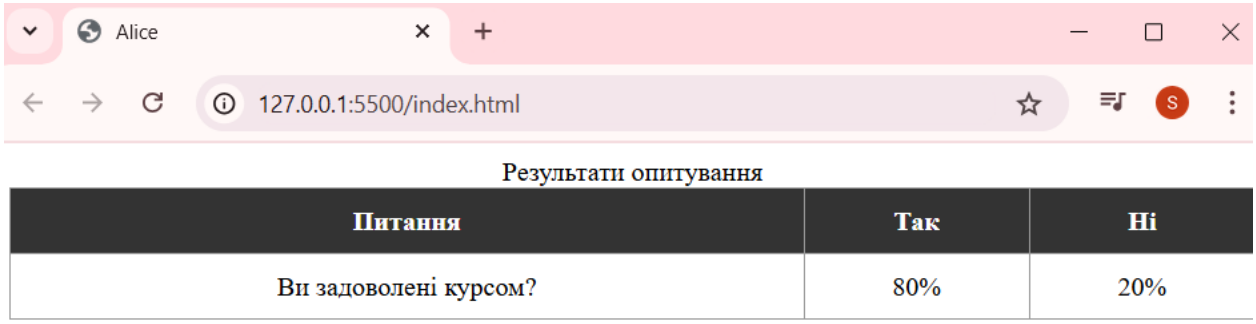

Рис.11 Результат формування таблиці із зазначеним форматуванням в main.css

5. Теги DIV та SPAN, псевдокласи

У веброзробці теги DIV та SPAN є універсальними інструментами для структурування контенту та застосування стилів. Вони самі по собі не несуть семантичного значення, проте надзвичайно важливі для організації макету сторінки, групування елементів і селективного застосування стилів.

Тег DIV

Тег — **блочний** елемент. Це означає, що він займає весь доступний простір по ширині батьківського контейнера, навіть якщо містить мінімум вмісту. Тег часто використовується для обгортання цілих секцій, блоків контенту, з метою подальшого застосування CSS-стилів або JavaScript-скриптів. У наступному прикладі ми групуємо заголовок і абзац в один блок, застосовуючи до нього загальні стилі.

В index.html:

```
<div class="container">
  <h2>Новини</h2>
```

```
    <p>Сьогодні відбудеться презентація нових можливостей CSS.</p>
  </div>
```

В main.css:

```
.container {
  background-color: #eef;
  padding: 10px;
  border: 1px solid #99c;
}
```

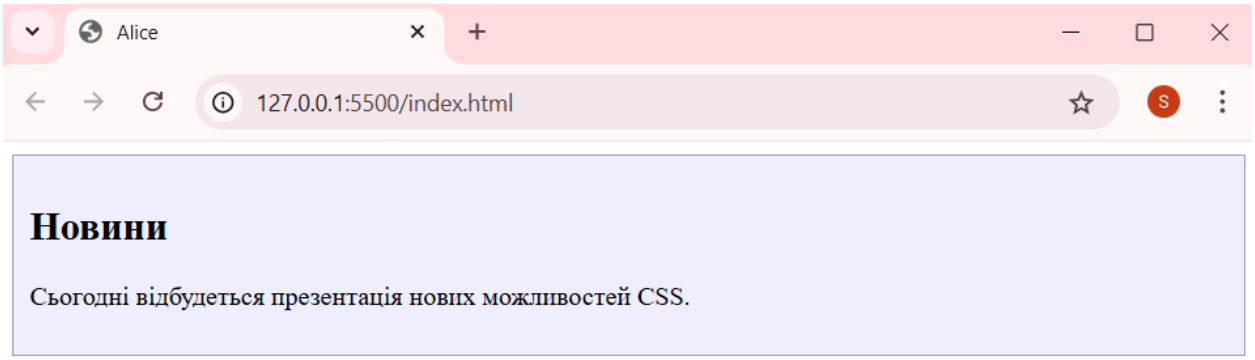

Рис. 12 Результат виконання коду з тегом

Тег SPAN

На відміну від div, тег — рядковий елемент, який не порушує потоку тексту. Його використовують для стилізації окремих фрагментів всередині абзаців, заголовків або інших елементів.

В наступному прикладі тег span дозволяє виділити слово «CSS» червоним кольором та жирним шрифтом, не змінюючи структуру решти абзацу.

В index.html:

```
<p>Моя улюблена мова — <span class="highlight">CSS</span>.</p>
```

В main.css:

```
.highlight {
  color: #c00;
  font-weight: bold;
}
```

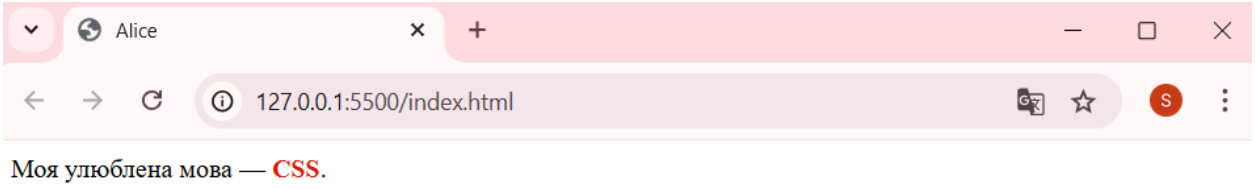

Рис. 13 Результат виконання коду з тегом

Псевдокласи у CSS — це механізм для вибору елементів у певному **стані**, або за логічними умовами, які не можна описати лише за допомогою HTML-структури.

Найбільш поширені псевдокласи:

- :hover — коли користувач наводить курсор на елемент.
- :active — коли елемент активується (натиснуто мишею).
- :focus — коли елемент отримує фокус (наприклад, поле вводу).
- :first-child, :last-child, :nth-child(n) — для вибору елементів у списках.
- :visited, :link — для стилізації посилань.

Приклад:

В index.html:

```
<a href="#">Наведіть курсор на це посилання</a>
```

В main.css:

```
a {
  color: blue;
  text-decoration: none;
}
a:hover {
  color: red;
  text-decoration: underline;
}
```

Коли користувач наводить курсор, посилання змінює колір на червоний і підкреслюється.

Псевдокласи дозволяють створити інтерактивний і динамічний інтерфейс, без використання JavaScript.

6. CSS-властивості: позиціювання

Позиціювання елементів у CSS — ключова складова управління розміщенням контенту. За допомогою відповідних властивостей ми можемо розміщувати блоки відносно вікна браузера, інших елементів або потоку документа.

Властивість position

Ця властивість задає спосіб позиціювання елемента. Вона може приймати такі значення:

static — за замовчуванням, елемент розташовується у звичайному потоці.

relative — відносне позиціювання: елемент зміщується відносно свого початкового місця.

absolute — абсолютне позиціювання: елемент зміщується відносно найближчого батьківського елемента з позиціюванням (не static).

fixed — фіксоване позиціювання: елемент розташовується відносно вікна браузера, навіть при прокручуванні.

sticky — поєднання relative та fixed: елемент прилипає до певної позиції при прокрутці.

Встановлення координат

Для елементів з позиціюванням relative, absolute, fixed, sticky доступні властивості:

top, right, bottom, left — зміщення у відповідних напрямках.

Приклад абсолютного позиціювання. У цьому прикладі блок .box буде розташований на 50 пікселів нижче та на 100 пікселів правіше верхнього лівого кута блоку .wrapper.

В index.html:

```
<div class="wrapper">
  <div class="box">Абсолютний блок</div>
</div>
```

В main.css:

```
.wrapper {
  position: relative;
  width: 400px;
  height: 300px;
  background-color: #ddd;
}

.box {
  position: absolute;
  top: 50px;
  left: 100px;
  width: 150px;
  height: 100px;
  background-color: #6cf;
}
```

Рис. 14 Результат виконання коду для створення блоку з абсолютним розташуванням

Плаваючі елементи (float)

Властивість float дозволяє розміщувати елементи **вплав** ліворуч або праворуч, дозволяючи іншим елементам обтікати їх.

Значення: left, right, none.

Приклад:

В index.html:

```
<img src="photo.jpg" class="float-img">
<p>Цей текст буде обтікати зображення праворуч.</p>
```

В main.css:

```
.float-img {
  float: left;
  margin-right: 10px;
}
```

Щоб **скасувати обтікання**, використовують властивість clear:

В main.css:

```
.clearfix {
  clear: both;
}
```

Важлива заувага!

Після плаваючих елементів структура документа може порушитися. Щоб цього уникнути, часто використовують спеціальний клас .clearfix, що застосовується до контейнера:

```
.clearfix::after {
  content: "";
  display: block;
  clear: both;
}
```

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Базові

1 Баран С. В. Основи web-програмування : навч. посіб. Кривий Ріг, 2023. 316 с.

2 Nixon R. Learning PHP, MySQL, JavaScript, CSS & HTML5. A Step-by-Step Guide to Creating Dynamic Websites. 3rd Edition. O'Reilly, 2019. 786 p.

3 O'Kane M. A Web-Based Introduction to Programming: Essential Algorithms, Syntax, and Control Structures Using PHP, HTML, and MariaDB/MySQL. Carolina Academic Press, 2021. 740 p.

4 Веб-технології та веб дизайн : навч. посібник / О. Г. Трофименко, О. Б. Козін, О. В. Задерейко, О. Є. Плачінда. Одеса : Фенікс, 2019. 284 с.

Web-ресурси

1. Довідник по HTML тегам : CSS: Український веб-довідник : веб-сайт. URL: https://css.in.ua/html/tags (дата звернення: 02.06.2025).

2. Електронний HTML і CSS довідник українською мовою : інтернет-довідник : веб-сайт. URL: https://html-css.co.ua/ (дата звернення: 02.06.2025).

Навчально-методичне видання

Гурковська Світлана Сергіївна

Конспект лекцій з дисципліни веб-дизайн та веб-розробка: HTML, CSS (частина 1)

за освітньо-професійною програмою
першого (бакалаврського) рівня спеціальності 122 «Компютерні науки»

Самостійне електронне мережеве видання
Публікується в авторській редакції