10.2. Структура сторінки, DOM-дерево, застосування тегів

Типова структура веб-сторінки, поняття DOM-дерева та робота браузера з ним. Зміст веб-документа: doctype, блок header, блок body. Класифікація елементів HTML-сайту за завданнями та областями застосування.

1. Структура веб-сторінки та DOM-модель
2. Основні поняття DOM
3. Теги верхнього рівня: <html>, <head>, <body>
4. Заголовок. Теги.
5. Заголовок. Особливі технології:
    a. Open Graph Protocol
    b. Resource Description Framework
    c. RSS
6. Тіло веб-сторінки:
    a. Робота з текстом
    b. Посилання
    c. Зображення та мультимедіа
    d. Списки
    e. Таблиці
    f. Форми та поля введення
    g. Фрейми
    h. Стилі та семантика
    i. Програмування

Структура веб-сторінки та DOM-модель

Код, який можна знайти в Інструментах розробника (Developer Tools) будь-якого веб-браузера, не обов'язково показує вихідний HTML-документ. Сучасна розробка сайтів – процес складний та багаторівневий. Програмісти працюють не лише з HTML, CSS або JavaScript в межах кількох окремих файлів. Сьогодні їх застосовується багато. На остаточній публікації сайту в мережі інтернет-сервіс, що складається з великої кількості документів, об'єднується. Найчастіше застосовуються й інші мови програмування: Java,Python, C#. Вони можуть генерувати фрагменти HTML-коду, що складаються при запиті тієї чи іншої сторінки. Таким чином, HTML-сторінка, що відкривається в браузері є результатом:

1. З'єднання різних документів (HTML,CSS та скриптів) в єдину структуру.
2. Генерації тегів мовами програмування та їх фреймворками (Python і Django, JavaScript та Angular).
3. Звернення до даних із деякої бази (реляційної або NoSQL).
4. Впливу Javascript та AJAX-технології (дозволяє динамічно підвантажувати елементи сайту).
5. Роботи браузера з побудови дерева документа.

Останній пункт пов'язаний з так званою моделлю DOM (Document Object Model, об'єктна модель документа).

DOM – модель документа як об'єкта, створюється веб-браузером у пам'яті пристрою на підставі того коду HTML, який був отриманий від сервера. Безпосередньо сам документ у

вигляді HTML браузер не показують у вихідному вигляді. Спочатку йде запит до сервера, а отримана відповідь розбирається для побудови дерева сторінки або DOM-дерева.

Схема така:

1. Сервер створює HTML код сторінки або віддає його (якщо це простий статичний сайт).
2. Браузер отримує код та розбирає на елементи дерева.
3. За необхідності підключається JavaScript, якщо він використовується, щоб змінити поведінку тегів та їх вмісту залежно від впливу користувача.
4. DOM-дерево відображається у вкладці браузера у вигляді, який задуманий розробниками.

HTML-код, який пишуть програмісти – це лише текстовий файл певного формату, а DOM – результат дій браузера, який створює об'єкти при парсингу текстових файлів.

Основні поняття моделі DOM

Завдяки W3-консорціуму вироблено єдиний стандарт побудови та аналізу вмісту веб-сторінки. До цього різні браузери діяли по-своєму, що створювало багато незручностей для розробників. Ознайомитись з актуальним DOM-стандартом можна на офіційному сайті.

Основні структури, які використовуються в об'єктній моделі документа:

- Дерево. Веб-сторінка може бути представлена як ієрархічне перевернуте дерево, що починається з головного елемента і розширюється до низу. У кожного об'єкта дерева є батько, який може бути порожнім, тобто null, і нащадки, якщо він знаходиться не в самому низу.
- Нащадок. Будь-який дочірній елемент вузла дерева. Нащадок може бути інклюзивним, тобто не прямим. Приклад: тег <html> має дочірній елемент <body>, всередині якого міститься елемент <article>, тоді тег <article> це не прямий нащадок <html>.
- Батько. Якщо об'єкт DOM-дерева має нащадків, то від для них є батьком. Тут також можлива інклюзивність. З прикладу вище тег <article> має прямого нащадку – тег <body>, і навіть інклюзивний батько – тег <html>.
- Брати/сестри. У стандарті використовується термін sibling, що з англійської перекладається як рідний брат або сестра. Так вважається, що об'єкт має «сусідів», що стоять на тому ж рівні дерева і єдиного батька, що не є порожнім.
- Події. До об'єктів дерева можна застосовувати різні події за допомогою JavaScript. Вони можуть спрацьовувати на дії користувача або мережеву активність. Так, при наведенні миші, кліку, натисканні клавіші або обриві зв'язку можлива різна поведінка всього сайту або його елементів.
- Вузол. Всі елементи DOM-дерева є вузлами, нодами, nodes. Їх можна створювати, змінювати, викликати властивості та методи за допомогою JS.

Наведемо приклад простого DOM-дерева.

Приклад - HTML

```
<!DOCTYPE html>
<html lang="uk">
<head>
```

```
    <title>HTML</title>
</head>
<body>
    <span>Вивчаємо HTML</span>
</body>
</html>
```

Маємо веб-сторінку, що складається з:

- Оголошення стандарту HTML. У цьому випадку DOCTYPE свідчить про використання HTML5.
- Тегу <html>. Все укладене між ним буде сприйнято як мову розмітки.
- Тегу заголовків. <head>, що містить <title>, всередині якого текст.
- Тіла сторінки, що відображається користувачеві (позначений тегом <body>). Тут є лише один HTML-елемент – тег з деяким текстовим вмістом.

Така структура веб-документа є стандартною. Обов'язкове оголошення DOCTYPE (версії HTML), наявність тегу <html>, всередині якого містяться <head> та <body>. Цього мінімуму достатньо, щоби пройти валідацію коду.

Зазначене вище успадкування елементів сприймається браузером як дерево.

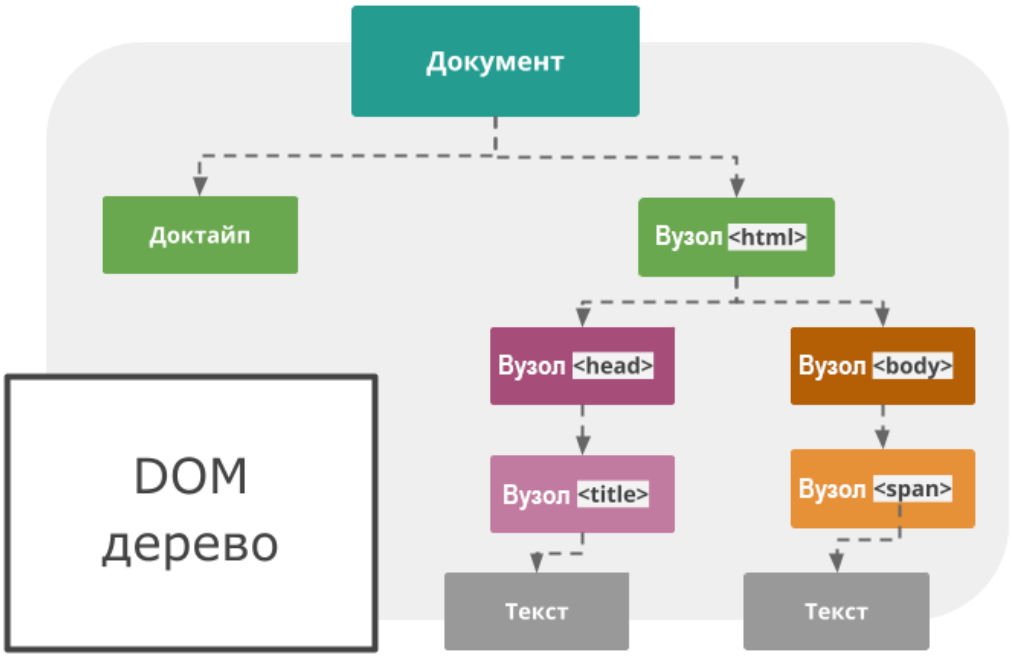

Рис. Структура DOM-дерева

Аналіз дерева дозволяє виділити такі об'єкти:

1. Вузол <html> є батьком для <body> і <head>, і навіть інклюзивним батьком для, наприклад, та тексту Вивчаємо HTML.
2. Вузол має нащадка, тобто текст HTML, а також прямого батька у вигляді <head>.
3. Вузли <body> та <head> відносяться до класу сіблінгів (брати або сестри), оскільки розташовані на одному рівні DOM-дерева і мають спільного батька <html>.

Інколи деякі верстальники можуть не слідувати канонічному шаблону представлення веб-сторінки. Сучасні браузери спроможні самі відновити відсутні елементи.

Звичайно, так робити не потрібно, адже якщо неправильно оформляти код сторінок, то і називати себе веб-розробником не варто. Тим більше, різні браузери можуть повести себе по-різному.

Теги верхнього рівня

Крім інформації, що демонструється відвідувачу сайту, на ньому має бути присутня і службова інформація. Вона містить оголошення DOCTYPE, заголовку, перелік мета-відомостей, посилань на CSS та JS-файли тощо. Нижче подивимося на цю групу тегів та інструментів.

<!DOCTYPE>

Доктайп задає стандарт, в якому створена веб-сторінка, щоб браузерам було простіше парсити вміст та побудувати DOM-дерево. На даний час можна зустріти 3 типи:

- HTML5: <!DOCTYPE html>
- HTML4: < !DOCTYPE ... "http://www.w3.org/TR/html4/loose.dtd">
- XHTML: < !DOCTYPE ... "http://www.w3.org/TR/. ../xhtml11.dtd">

Найпопулярнішим і функціональним є перший вид (HTML5), але зустрічаються й інші.

Кореневі елементи

Крім оголошення стандарту документа, типова веб-сторінка містить ще 3 теги:

Тег <html>

Кореневий тег, що вказує на вміст як HTML-код. Починає та закінчує будь-який веб-документ. Може мати глобальні атрибути. За необхідності містить унікальну властивість xmlns, якщо сторінка повинна дотримуватися стандарту XHTML. З глобальних атрибутів рекомендується вказувати lang, тобто. мова, яка застосовується для тексту сайту.

Приклад - HTML

```
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

      …

</html>
```

У прикладі задано як мову документа англійську та xml простір імен. Валідатор не зреагує, якщо пропустити xmlns властивість, оскільки вона за замовченням включається до XHTML.

Тег <head>

Це контейнер із метаданими, тобто містить інформацію про сам документ, яка безпосередньо не відображаються у вікні браузера. Особливих атрибутів не має, але може використовувати глобальні при необхідності. Всередині обов'язкова наявність тегу <title>.Може також містити елементи: <style>, <base>, <link>, <meta>, <script>, <noscript>.

Тег <body>

Основна частина коду HTML-сторінки, так зване тіло документа. Всі теги, що залишилися, будуть присутні тут. Звичайно, в документі може зустрічатися лише один раз. Допустимо застосування глобальних властивостей та властивостей дій.

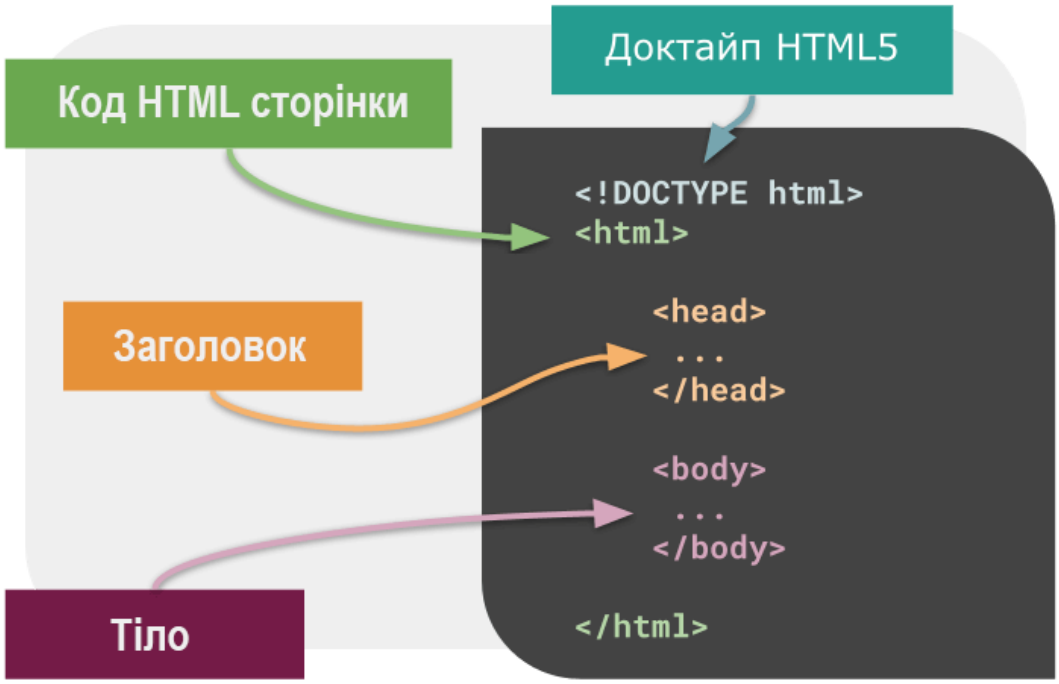

Рис. Теги верхнього рівня

Коментарі

Особливий тип даних, який зустрічається як у заголовній частині документа, так і в його тілі. Код не обробляється браузером безпосередньо, але досить важливий для розробників, оскільки містить пояснення, коментарі, уточнення.

Приклад - HTML

```
<!--Це коментар. Він не відображається браузером-->
```

Наявність коментарів у коді – хороша практика, що свідчить про професіоналізм верстальника та його повагу колег.

Заголовок веб-сторінки. Теги заголовки.

Отже, заголовна частина HTML-документа, що обертається тегом <head>, хоч і не видима користувачеві, має важливе значення. Вона суттєво розширює можливості розробників та дозволяє включати інформацію для браузера для правильної обробки коду. Охарактеризуємо ці елементи.

Тег <title>

Необхідний для відображення заголовка сторінки. Його видно у вкладці вікна. Контент – лише текст. Вміст заголовка важливий для SEO, оптимізації пошуку роботами сайту. Для кожної сторінки можливий лише у єдиному екземплярі.

Приклад - HTML

```
<head>
    <title>Заголовок сторінки</title>
</head>
```

Може містити глобальні атрибути. Повинен відображати суть сторінки (використовується при показі в результатах пошуку пошукової системи) і не бути дуже довгим (оскільки не поміститься у вкладці).

Тег <style>

Застосовується для вказування інформації про каскадні таблиці стилів. Коли цих стилів на веб-сайті не так багато, не обов'язково їх розміщувати в окремому файлі. Достатньо обернути на цей тег.

Приклад - HTML

```
<style>
    h1 {color:brown;}
    body {color:green;}
</style>
```

Вміст сторінки модифікується: всі заголовки <h1> стануть коричневими, а будь-який інший текст тіла документа буде зеленим.

За необхідності містить атрибути: media (вказує на тип пристрою, для якого призначаються ці властивості) та type (зі значенням text/css).

Тег <base>

Цей тег є поодиноким і необхідний для вказування основної адреси сайту. Всі відносні посилання всередині порталу будуть відштовхуватись від цього кореня.

Приклад - HTML

```
<base href="https://site.ua/" target="_blank">
```

Кореневою адресою для інших відносних посилань на сайті буде https://site.ua і всі вони будуть відкриватися в новій сторінці.

Тег <link>

Є одиночним елементом, але може бути присутнім у багатьох екземплярах у документі. Визначає відносини між поточним файлом та зовнішніми ресурсами. Необхідний для вказівки посилань на CSS-документи, шрифти, фавіконку.

Приклад - HTML

```
<!--Як таблицю стилів сайт використовує файл main.css.-->
<link rel="stylesheet" href="main.css">
<!--Зв'язаний зі шрифтами. Для документа застосовано шрифт від Google Lobster, до якого потрібно підключитися заздалегідь.-->
<link rel="preconnect" href="https://fonts.gstatic.com">
<link
href="https://fonts.googleapis.com/css2?family=Lobster&display=swap"
rel="stylesheet">
<!--іконка сайту-->
<link rel="shortcut icon" href="favicon.png">
```

```
<!-- RSS-канал. Властивість alternate надає альтернативну версію документа (Дзеркало, переклад іншою мовою, пристрій тощо).-->
<link rel="alternate" type="application/rss+xml" title="Новини сайту" href="https://site.ua/news.rss">
```

Тег <meta>

Одиночний тег, який використовується для вказування метаінформації про сторінку. Ці відомості використовуються браузером, пошуковими системами або іншими сервісами. Найчастіше ігнорується розробниками-початківцями.

Приклад - HTML

```
<!--Вказуємо ключові слова сторінки (важливі для просування сайту та кращого виявлення пошуковими машинами).-->
<meta name="keywords" content="HTML, tags, W3C">
<!--Опис документа. Мета – та сама. <meta name="description" content="Вивчаємо структуру web-сторінки">
<!--За необхідності вказують власника сайту або розробника-->
<meta name="author" content="Василь Василів">
<!--Примушує браузер перевантажувати сайт щохвилини-->
<meta http-equiv="refresh" content="60">
<!--Враховуємо різні пристрої, з яких відвідувачі можуть заходити на портал.
Якщо не вказати viewport, то на смартфонах сторінка буде масштабуватися і займати не весь доступний об'єм екрана.-->
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Докладніше з додатковими атрибутами тега <meta> можна ознайомитися в документації.

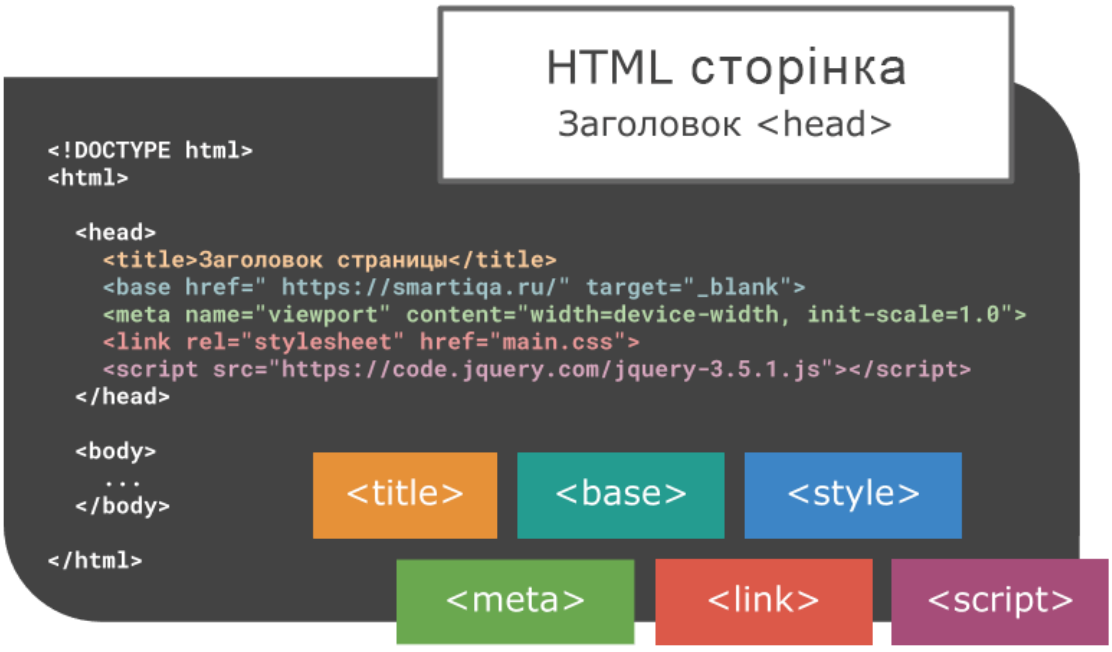

Рис. Заголовкові теги HTML сторінки

Тег <script>

Цей тег застосовується не лише в заголовній частині сторінки, але й у будь-якому іншому місці (залежно від завдання). Містить JavaScript код або посилання на js-скрипти.

Якщо потрібно, щоб скрипт спрацював із самого початку завантаження сайту, його підключають в заголовну частину. Якщо потрібний його функціонал після повного завантаження сторінки, зазвичай розміщується в самому кінці тегу <body>.

Приклад - HTML

```
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script>
    console.log('Все працює!')
</script>
```

Перший тег <script> говорить про те, що сайт використовує фреймворк jQuery, а в другому випадку ним обернуто код на мові JavaScript, який виводить у консолі інструментів розробника у браузері позначений текст.

Тег <noscript>

Не завжди браузери підтримують JavaScript (особливо старі версії), або ця опція вимкнена. Щоб не ламати функціонал сайту, дбають і про таких відвідувачів: їх або попереджають про обмежені можливості, або просять оновити програму, або переконують включити опцію підтримки JS.

Приклад - HTML

```
<noscript>
Ваш браузер не підтримує JavaScript. Оновіть його або активуйте скрипти.
</noscript>
```

Заголовок веб-сторінки. Особливі технології

HTML постійно розвивається та розширюється. З'являються нові потреби та технології, які можуть використовуватися розробниками для вдосконалення роботи сайту у різних умовах та середовищах. Вирізняють кілька таких можливостей, про які розповімо нижче.

Open Graph протокол

Соціальні мережі – невід'ємна частина Інтернету. Якщо їх ігнорувати, бізнес втрачає значну аудиторію.

Технологія Open Graph дозволяє впроваджувати вміст сторінки у стрічку соціальних мереж. Спочатку розроблено Facebook, але зараз підтримується та активно використовується й іншими соцмережами.

Веб-розробники застосовують цей протокол для публікації анонсів матеріалів сайтів. Він дозволяє коректно відображати вміст новини у межах конкретного ресурсу. Для цього використовується тег <meta>.Розглянемо приклад для наочнішої демонстрації можливостей.

Приклад - HTML

```
<meta property="og:title" content="У всі тяжкі">
<meta property="og:type" content="video.tv_show">
<meta property="og:url" content="https://www.imdb.com/title/tt0944947/">
<meta property="og:image" content="https://m.media-amazon.com/images/M/MV5BN2JmZGFiMDktYmRhYi00MDNhLWFhNmItNmEwMGZjMDE0N2U4XkEyXkFqcGdeQXVyNjg0Nzk2Nzc@._V1">
<meta property="og:description" content="Опис серіалу">
```

Тут вказано назву серіалу, тип, посилання на нього, адресу постера та опис. Цей контент можна без проблем розмістити у соціальній мережі, і він відобразиться так, як замислювався. Ознайомитись докладніше з роботою протоколу Open Graph можна на офіційному сайті .

Resource Description Framework

Технологія RDFa (Структура опису ресурсів в атрибутах) дозволяє розміщувати додаткові метадані на сторінці, не обов'язково лише у заголовній частині. Це рекомендація W3-Консорціуму, задумана для більш тонкого вказування пошуковим машинам ідей розробників.

Адже важливо не просто задовольнити відвідувача ресурсу, а й сервіс, який знаходить даний сайт. І чим більше відомостей про надано (із зазначенням типів даних, джерел, контактної інформації, дат), тим релевантнішою буде аудиторія.

На додаток до тегів <meta> може застосовуватися RDFa.

Приклад - HTML

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>RDFa</title>
    </head>
    <body vocab="http://site.com/">
        <p typeof="Blog">Welcome in
          <a property="url" href="https://site.ua/">site</a> about company
        </p>
    </body>
</html>
```

Це мінімально можлива структура з використанням RDFa. Підключено словник, чим перетворили властивості на терміни, а також вказали атрибут property="url" для посилання.

RSS канали

Не обов'язково відвідувати десятки сайтів щодня, щоб отримати інформацію про їх оновлення. RSS (Really Simple Syndication,«Дійсно просте агрегування»)– один із способів.

З'являється можливість надавати нові дані ресурсу та передавати їх на тисячі інших. RSS використовує формат xml, а за допомогою браузера або спеціальних програм ці зведення можна читати.

Приклад - HTML

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
<channel>
  <title>Стаття 1</title>
  <link>https://site.ua/</link>
  <description>Опис першої публікації</description>
  <item>
    <title>Стаття 2</title>
    <link>https://www.google.com/</link>
    <description>Опис другої публікації</description>
  </item>
</channel>
</rss>
```

Сьогодні RSS вже не настільки популярний (переважають соціальні мережі, пуш-повідомлення), але цілком актуальний.

Тіло веб-сторінки

Вміст веб-ресурсу розташовується всередині тегу <body>. Більшість елементів – його нащадки. Вони виконують різні функції: працюють із зображеннями або відео, форматують контент, подають посилання, зображують таблиці, малюють форми.

Робота з текстом

Оскільки велика частина заповнення веб-сайту - текст, то для роботи з ним передбачено значну кількість тегів. Вони можуть мати як стилістичний, так і семантичний контекст.

Розглянемо деякі:

Тег <address>

Як випливає з назви, всередині тегу наводиться контактна інформація. За замовченням текст всередині тегу виділяється курсивом.

Приклад - HTML

```
<address>
Написано <a href="mailto:webmaster@example.com">Юрчак І.Ю.</a><br>
Адреса:<br>
www.site.ua<br>
79013, Львів, а/с 100<br>
Україна
</address>
```

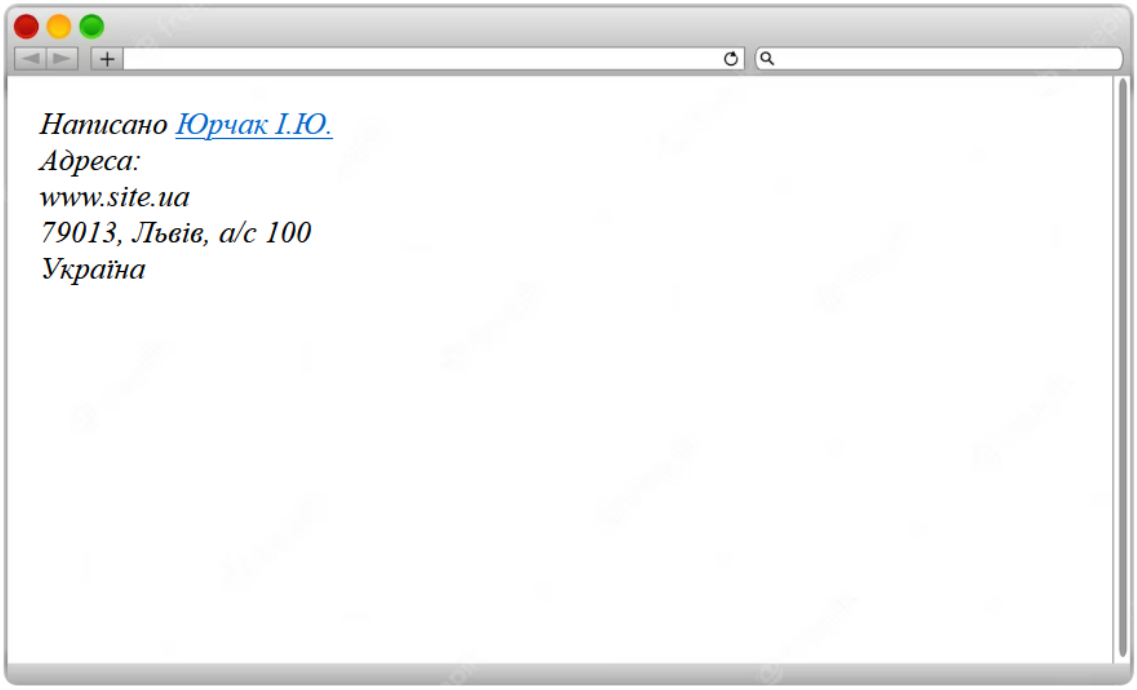

Рис. Тег <address>

**Тег **

Робить текст грубим, але не несе семантичного навантаження. Використовується для візуального ефекту. Тег є рядковим, тому текст всередині не переноситься на новий рядок.

Приклад - HTML

```
<body>
      Звичайний текст
      <strong>Виділений текст</strong>
      Звичайний текст
</body>
```

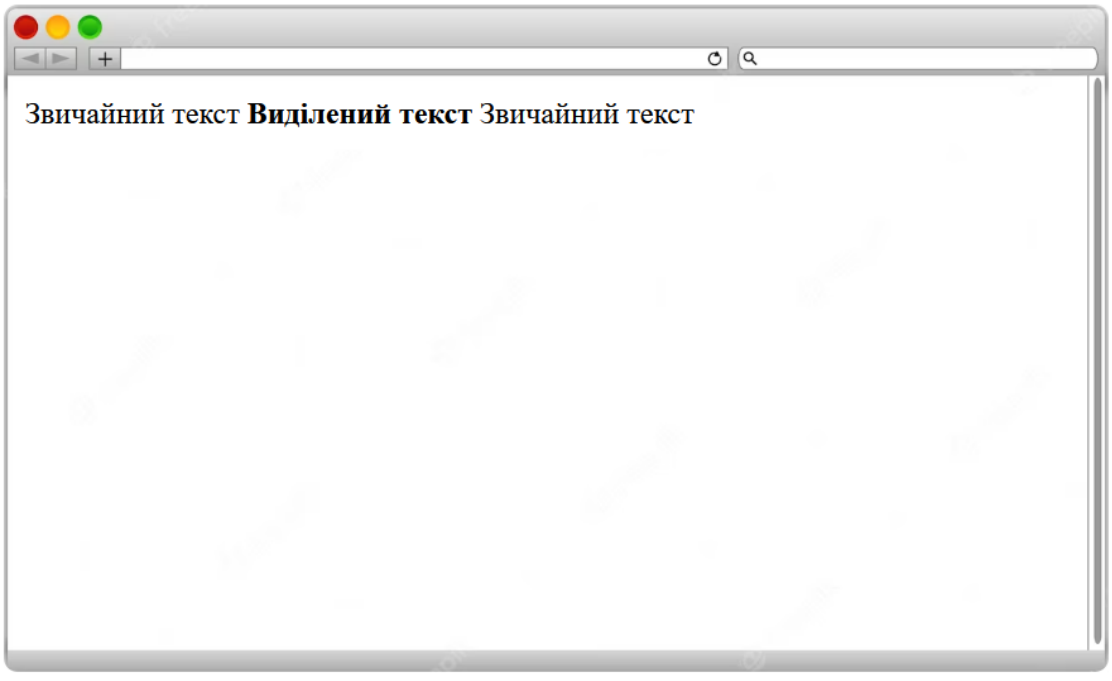

Рис. Тег

Тег <q>

Для позначення коротких цитат.

Приклад - HTML

```
<p>Закон Мерфі:
    <q>Коли запізнюєшся, маршрутка їде дуже повільно і зупиняється біля кожного світлофора.</q>
</p>
```

Тут застосовано 2 теги: (абзац, блоковий) і <q> (рядковий). Текст обрамляється лапками.

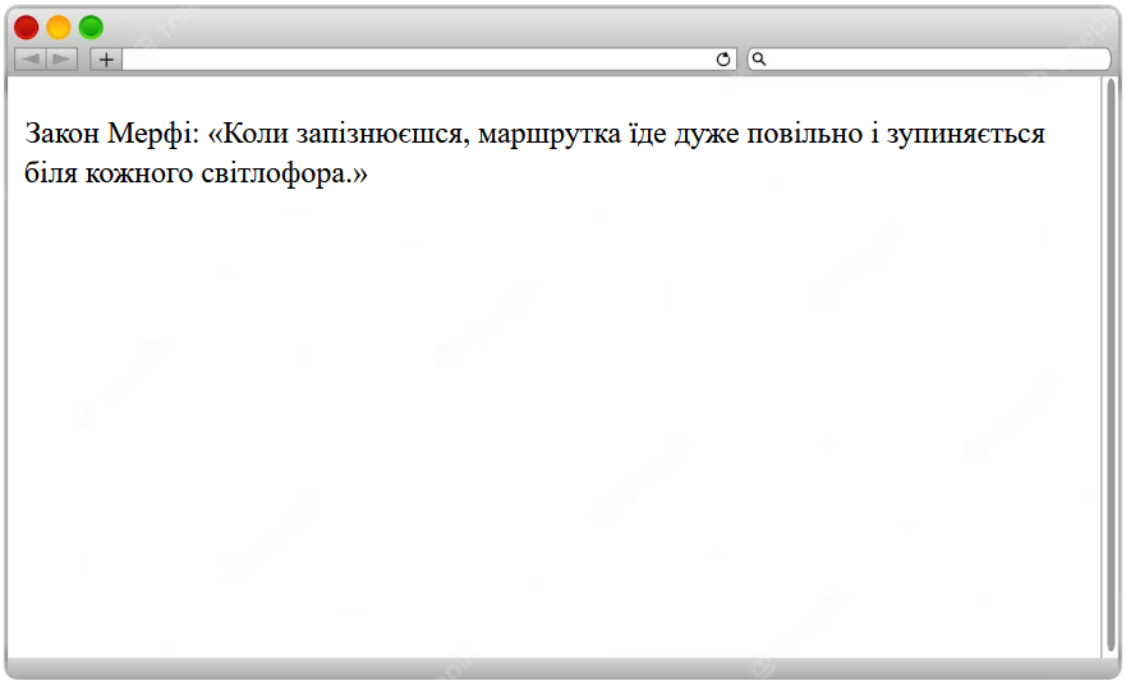

Рис. Тег <q>

Тег <abbr>

Використовується для виділення абревіатури. Можна застосовувати разом з елементом <dfn>, коли потрібно повідомити, що термін буде розшифровано або йому буде визначено.

Приклад - HTML

```
<p>
    <dfn>
        <abbr title="Мова гіпертекстової розмітки">
           HTML
        </abbr>
    </dfn> - є стандартом для створення веб-сторінок.
</p>
```

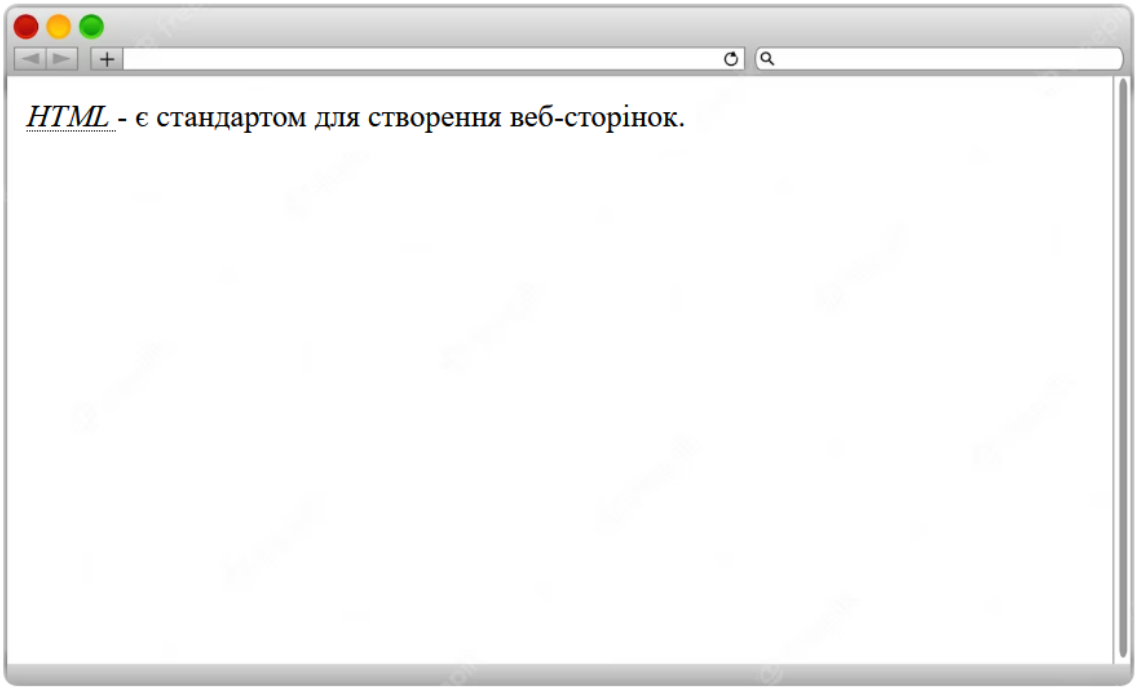

Рис. Тег <abbr>

Є й інші теги взаємодії з текстовими даними: <blockquote>, <cite>, , <i>, <mark>, <s>, , <sup>, <u> тощо.

Посилання

Поняття HTML включає так званий гіпертекст. Якщо текст посилається іншу частину документа чи інший ресурс, його позначають саме таким терміном.

Для позначення посилань використовується низка тегів.

Тег <a>

Визначає гіперпосилання. Має чимало особливих атрибутів (головний з яких – href), підтримує глобальні та дієві властивості. Поведінка тегу за замовченням у сучасних браузерах така:

- Не відвідане посилання виділяється синім кольором та підкресленням.
- Відвідане посилання – блакитного кольору та підкреслено.
- Активне посилання – червоне з підкресленням.

Приклад - HTML

```
<p>Текст</p>
<a href="https://site.com/" target ="_blank">Сайт</a><br>
<a href="mailto:user@gmail.com">Моя пошта</a><br>
<a href="tel:+38000000000" target="_blank">Мій телефон</a><br>
<a href="#part2" target="_blank">Якірне посилання</a><br>
<span id="part2">Текст</span>
```

1. Посилання можуть вести на інші сторінки сайту чи зовнішні ресурси, поштові скриньки, телефонні номери чи інші частини всередині документа.
2. Можна поставити і атрибут target, який відкриє нову вкладку або завантажить ресурс у поточні вкладці.

3. При натисканні на посилання з email відкриється поштова програма, яка використовується за замовченням.
4. Натискання на посилання телефону спробує зателефонувати на нього, використовуючи мобільний пристрій.
5. Для позначення посилання на іншу секцію сторінки (актуально для SPA, односторінкових сайтів) потрібно вказати її ID і знак #.Таке посилання називається якірним.

Тег <nav>

Цей тег сам собою не є посиланням, але служить обгорткою для головного меню сайту (елементи якого є посиланнями). Зазвичай застосовується у єдиному екземплярі на сторінці. Належить до блокових тегів.

Приклад - HTML

```
<nav>
    <a href="#">HTML</a> |
    <a href="#">CSS</a> |
    <a href="#">JavaScript</a> |
    <a href="#">Python</a>
</nav>
```

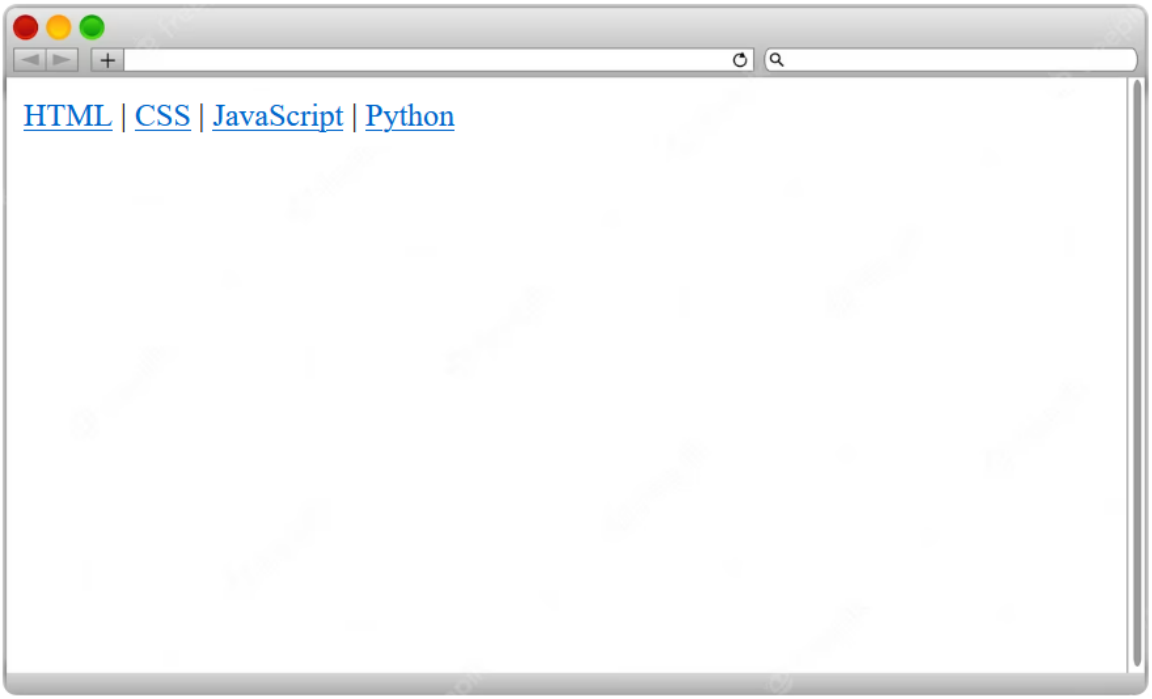

Рис. Тег <nav>

Всередині елемента перелічено посилання, які стосуються навігаційного меню сайту.

Зображення та мультимедіа

Сучасний сайт наповнений картинками, аудіо- та відео-даними, оскільки обмеження трафіку та повільний Інтернет для більшості користувачів – давно забуте явище.

Тег <img>

Будь-яке зображення растрового формату на сайті позначається тегом <img>. Рекомендується застосовувати його з атрибутами src (тут вказується адреса файла з зображенням) та alt (на випадок неможливості відобразити картинку виведеться текст).

Приклад - HTML

```
<img href="py.jpg" alt="Python на темному тлі" width="555" height="555">
```

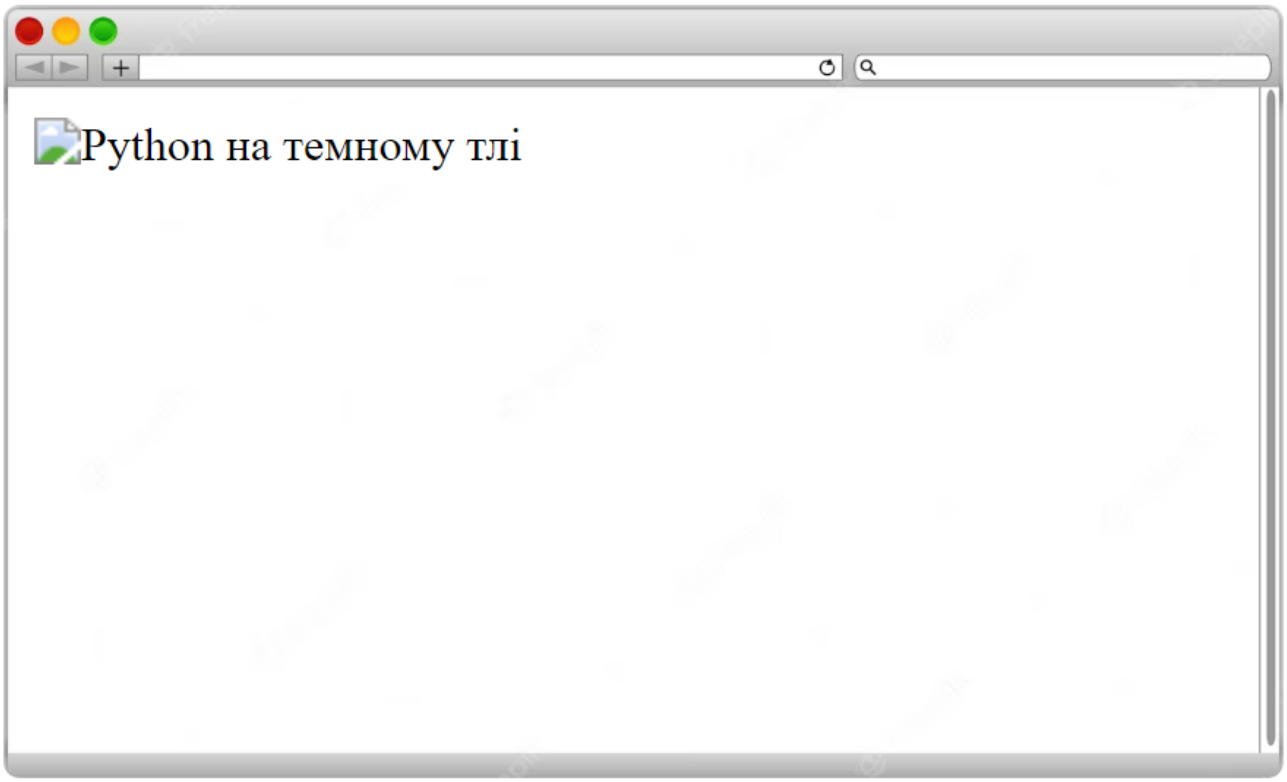

Рис. Відображення alt у разі відсутності файлу з зображенням

Оскільки зображення py.jpg немає, відобразився вміст властивості alt. Для зображень також можна вказувати розміри за шириною та висотою (у пікселях або відсотках).

Тег <figure>

Більше семантично грамотний спосіб позначати ілюстрації на сторінці. Ми не просто задаємо адресу зображення, а й підпис до нього (і повідомляємо тим самим, що цей напис відноситься до цієї картинки). Всередині містить теги <img> і <figcaption>.

Приклад - HTML

```
<figure>
    <img  src="https://logos-world.net/wp-content/uploads/2021/10/Python-Logo-700x394.png" alt="Python">
    <figcaption>Логотип Python на світлому фоні</figcaption>
</figure>
```

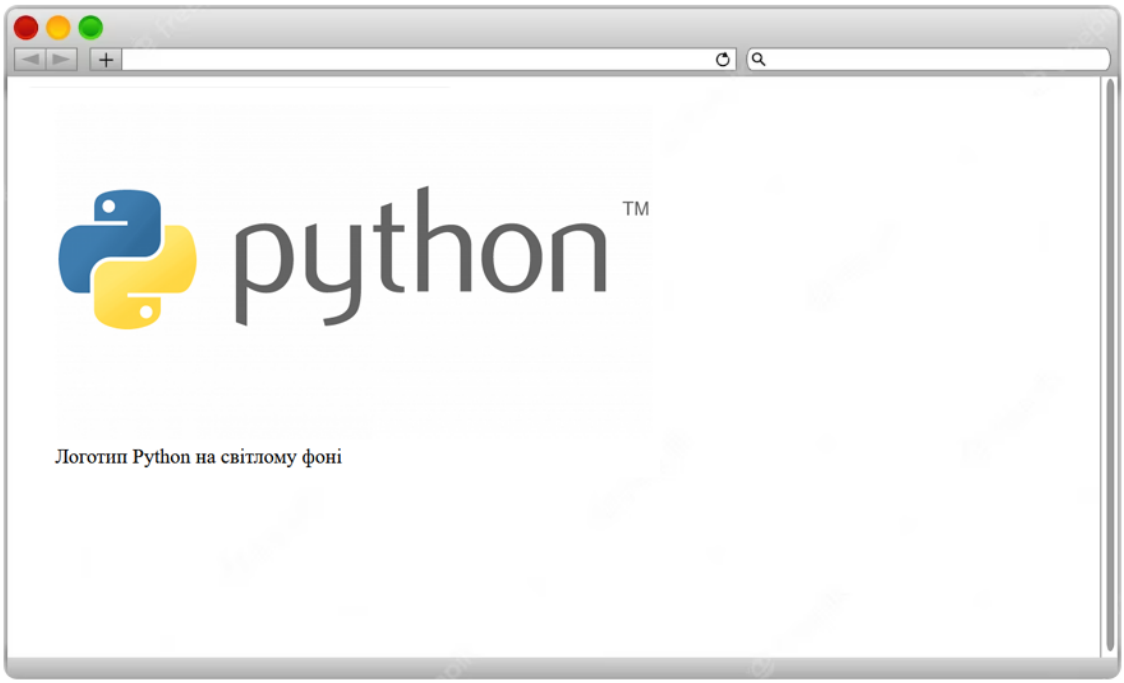

Рис. Тег <figure>

Особливих властивостей не має, але може підтримувати глобальні та атрибути дій.

Тег <picture>

Гнучкіша настройка відображення картинок. Дозволяє не просто масштабувати зображення залежно від пристрою або ширини екрану, а й відображати різні фотографії (якість, зміст). Для цього застосовують додатково теги <source>, <img>.

Приклад - HTML

```
<picture>
    <source media="(min-width:900px)" srcset="https://logos-world.net/wp-content/uploads/2021/10/Python-Logo-700x394.png">
    <source media="(min-width:600px)" srcset="
https://images.unsplash.com/photo-1649180556628-9ba704115795?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=crop&w=1462&q=80 ">
    <img src="
https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Python-logo-notext.svg/115px-Python-logo-notext.svg.png " alt="Python">
</picture>
```

У міру зміни розміру вікна браузера користувачеві демонструватиметься різні ілюстрації.

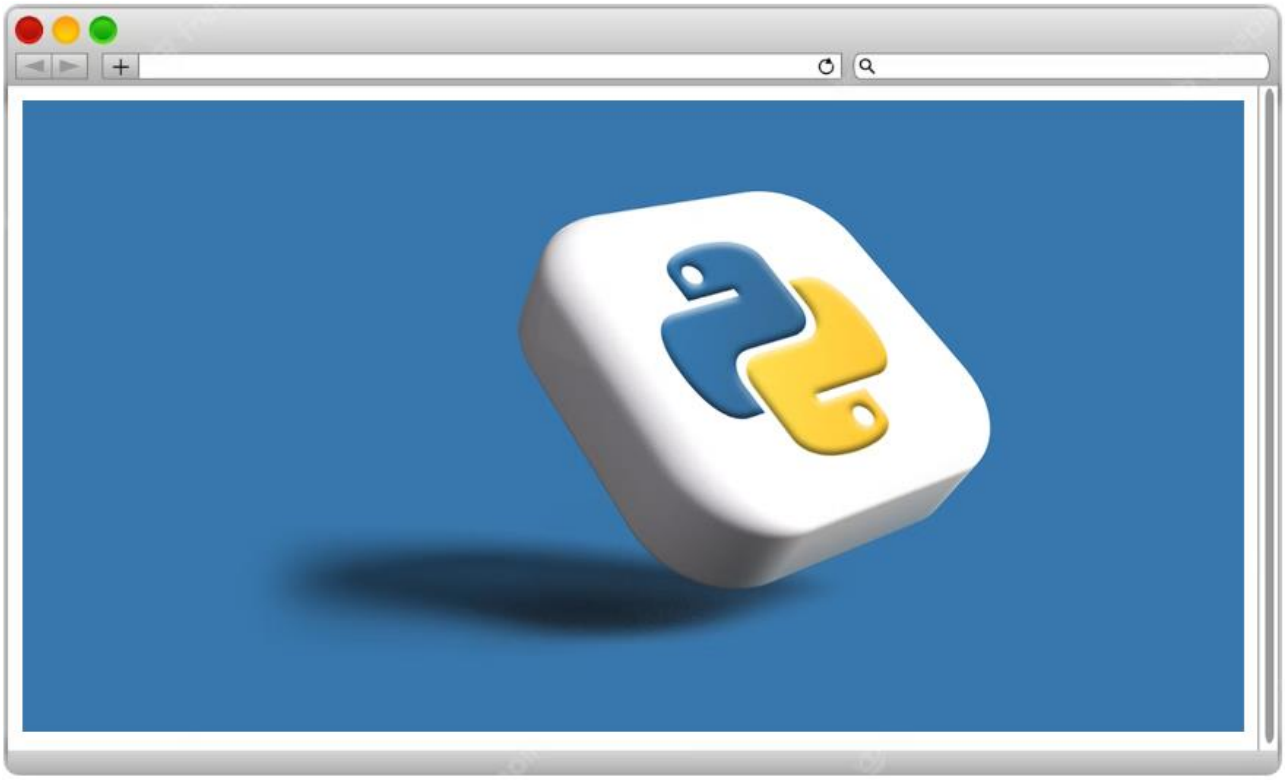

Рис. Роздільна здатність екрана > 900px

Рис. 900px > Роздільна здатність екрана > 600px

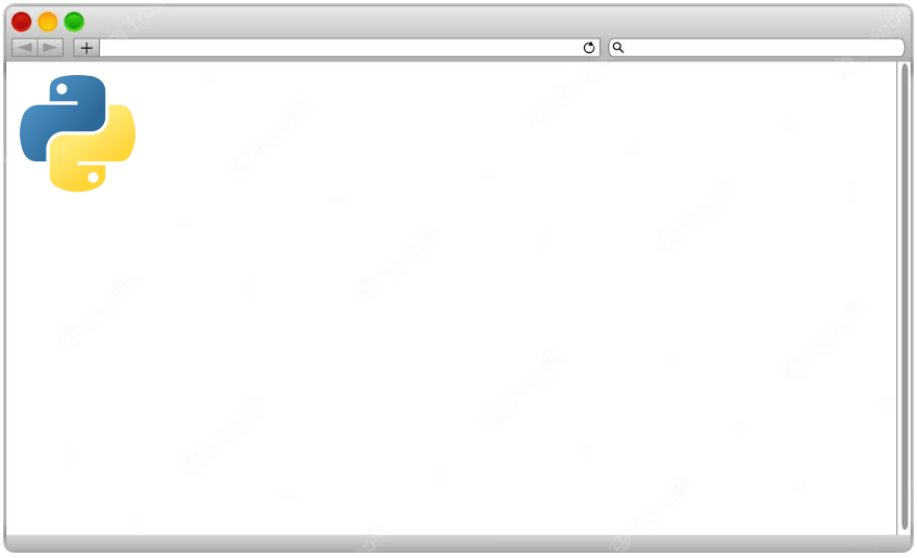

Рис. 600px < Роздільна здатність дисплея

Тег <video>

Використовується для впровадження на веб-сайт відео-вмісту. Всередині містить елементи <source>, які використовуються для роликів різних форматів: MP4, WebM, OGG .

Приклад - HTML

```
<video width="1300" height="800" controls>
      <source src="film.mp4" type="video/mp4">
      <source src="film.ogg" type="video/ogg">
      Ваш браузер не підтримує тег VIDEO
</video>
```

Спочатку браузер спробує відтворити перший файл, потім другий, а у разі неможливості обробити тег <video> повідомить про це відвідувача ресурсу.

Списки

Для наочного сприйняття структурованої інформації її можна подати як списки, які можуть бути нумерованими або ненумерованими. HTML дозволяє створювати їх за допомогою набору тегів: <ul>, <ol>, <li>, <dl>, <dt>, <dd>.

Нумеровані списки

Є перерахуванням елементів порядковими значеннями (числами, літерами).

Приклад - HTML

```
<ol start="26" reversed type="a">
    <li>Літера Z</li>
    <li>Літера Y</li>
    <li>Літера X</li>
</ol>
```

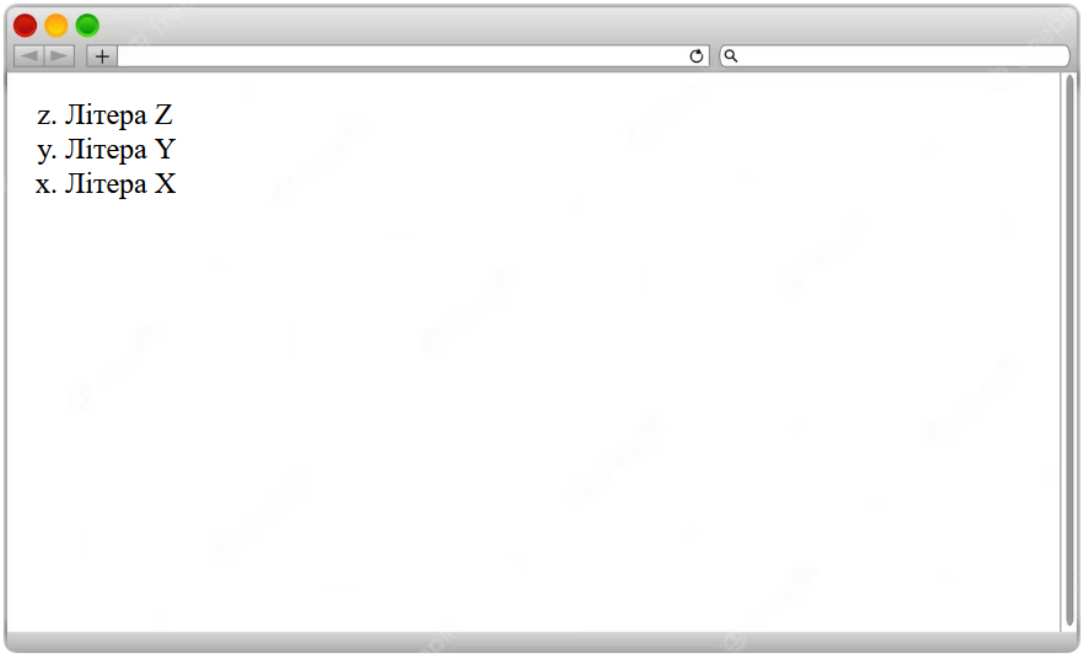

Рис. Нумеровані списки

Тег <li> Відображає окремі елементи списку. показує, що список нумерований, до якого застосували 3 атрибути:

- reversed (Зворотний порядок нумерації).
- start (почали з останньої літери англійського алфавіту).
- type – вказали, що як номери будуть використовуватися літери в нижньому регістрі.

Маркіровані списки

Як маркер використовуються спеціальні значки.

Приклад - HTML

```
<ul type="disc">
    <li>Молоко</li>
    <li>Чай</li>
    <li>Кава</li>
</ul>
```

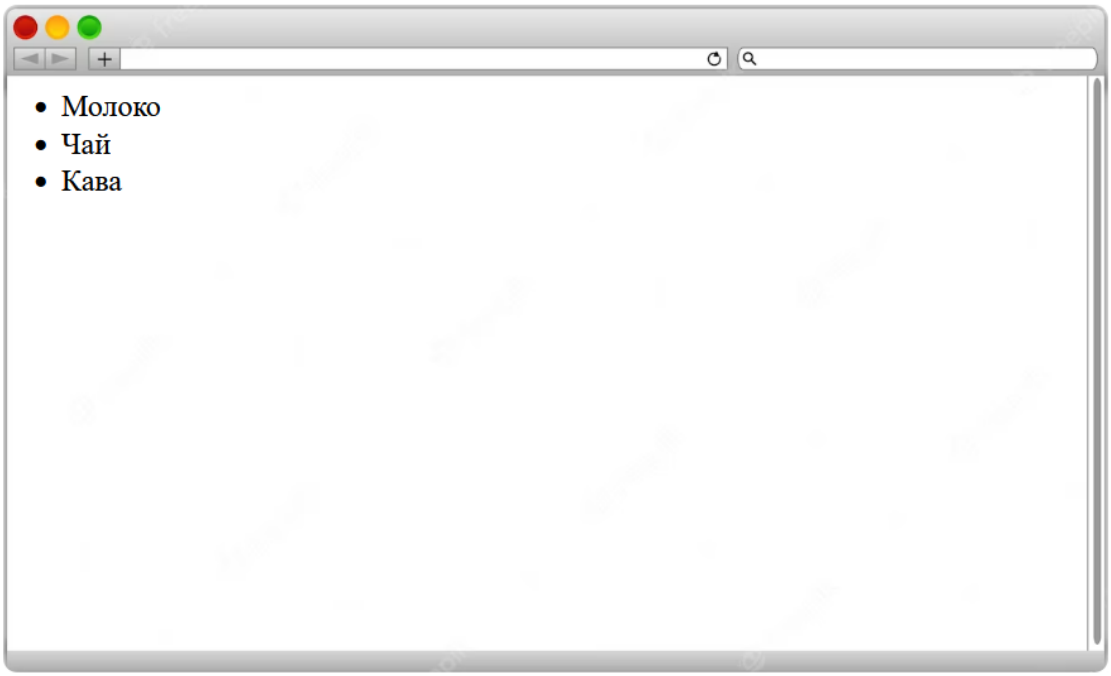

Рис. Маркіровані списки

За замовченням у ролі маркерів використовуються зафарбовані диски. Змінити тип маркера можна через CSS або через атрибут type (type="circle" – виведуться пусті кружечки, type="square" - квадратні маркери).

Списки з визначеннями

Особливий тип списків, у яких містяться терміни та їхнє трактування. Зручно застосовувати для формування тлумачного словника термінів.

Приклад - HTML

```
<dl>
    <dt>Гіпертекст</dt>
    <dd>Група документів, пов'язаних взаємними посиланнями</dd>
    <dt>HTML</dt>
    <dd>Мова гіпертекстової розмітки</dd>
</dl>
```

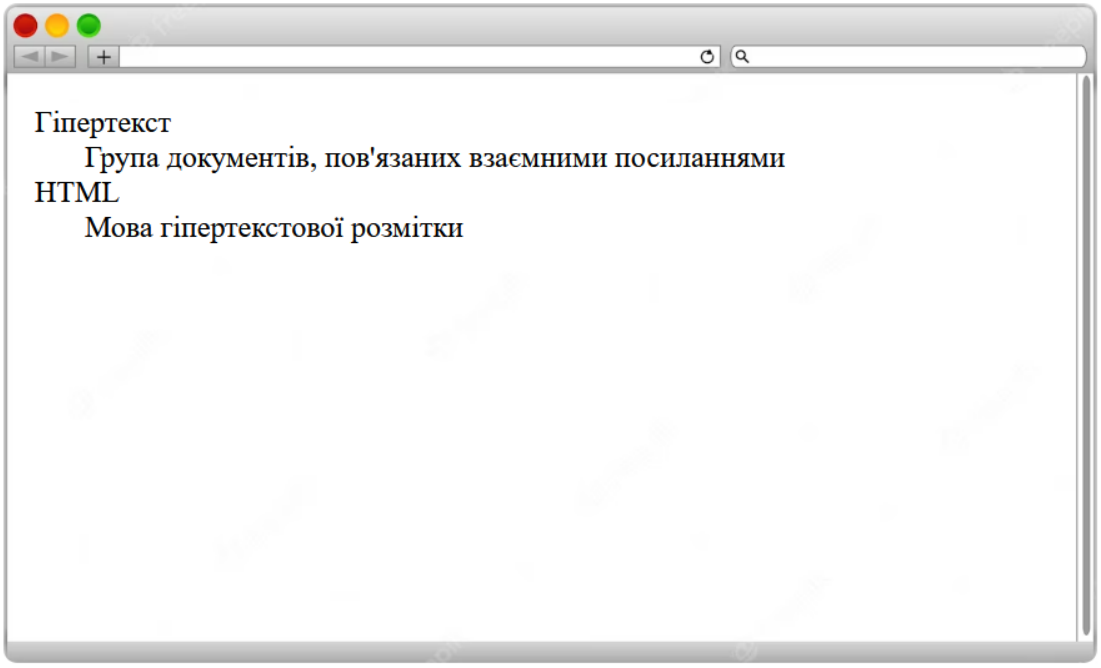

Рис. Списки з визначеннями

Таблиці

Для побудови таблиць застосовуються відповідні теги:

- Тег <table> (вміст таблиці).
- Теги <caption> (заголовок), <th> (заголовок),<thead> (групування заголовної частини таблиці).
- Тег <tbody> (основна частина).
- Тег <tr> (Рядок таблиці).
- Теги <col> (колонка таблиці), <colgroup> (угруповання колонок).
- Тег <td> (комірка).
- Тег <tfoot> (підсумковий вміст).

За допомогою властивостей rowspan та colspan у тегів <th> та <td> комірки таблиць можна об'єднувати.

Приклад - HTML

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Таблиця</title>
    <style>
        td {
            border: indigo solid 3px;
            width: 100px;
            height: 100px;
```

```
            }
    </style>
</head>
<body>
    <table>
        <caption>Приклад таблиці</caption>
        <tbody>
            <tr>
                <td rowspan="2">1</td>
                <td colspan="2">2</td>
                <td colspan="2">3</td>
            </tr>
            <tr>
                <td>4</td>
                <td>5</td>
                <td>6</td>
                <td>7</td>
            </tr>
            <tr>
                <td>8</td>
                <td colspan="4">9</td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

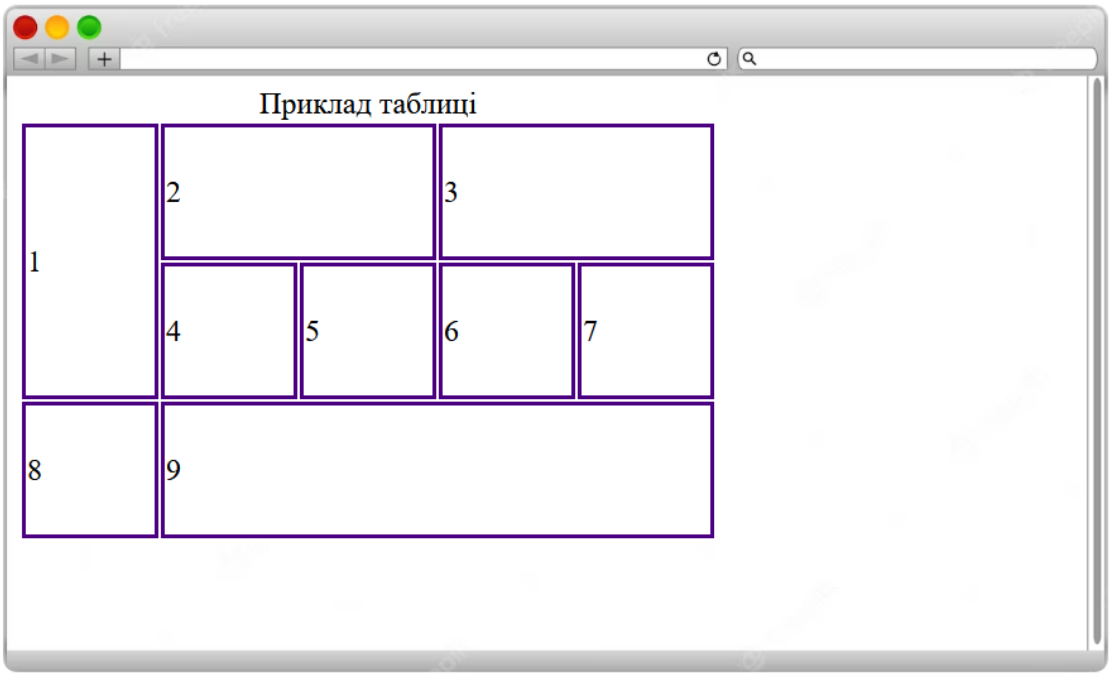

Рис. Таблиці

1. Для відображення меж комірок використані стилі (ми їх вивчимо в наступній темі, а поки що просто застосуємо для наочності результату).
2. Властивість rowspan поєднує комірки однієї колонки, а атрибут colspan - комірки різних колонок (підряд, що йдуть в обох випадках).
3. Якщо не вказано заголовні або підсумкові рядки таблиці, її вміст обертається тегом <tbody>.

Форми та поля введення

Форма є інтерактивним елементом, оскільки дозволяє користувачеві вводити деякі дані та надсилати їх до серверу.

Форми обертаються тегом <form>, всередині якого можлива наявність елементів:

1. Тег <input> (після введення).
2. Тег <textarea> (текстове поле для введення довгого тексту).
3. Тег <button> (кнопка).
4. Тег <select> (меню опцій).
5. Тег <option> (Певна опція).
6. Тег <optgroup> (групування опцій).
7. Тег <fieldset> (групування частини форми).
8. Тег <label> (підпис елемента).
9. Тег <output> (поле для результатів обчислення у відповідь на дії користувача).
10. Тег <legend> (заголовок для вмісту тега <fieldset>).
11. Тег <datalist> (Набір опцій для вибору в полі введення з властивістю list).

У тегу <form> вказують адресу скрипта, який обробляє результат заповнення форми, а також метод надсилання даних (GET або POST).

Приклад - HTML

```
<form action="/action.py" method="get">
      <label for="name">Ім'я:</label>
      <input type="text" id="fname" name="fname"><br>
      <label for="lname">Прізвище:</label>
      <input type="text" id="lname" name="lname"><br>
      <label for="age">Вік:</label>
      <input type="number" id="age" name="age" max="120" min="10"><br>
      <label for="browser">Який у вас браузер?</label>
      <input list="browsers" name="browser" id="browser"><br>
      <datalist id="browsers">
            <option value="Edge"></option>
            <option value="Firefox"></option>
            <option value="Chrome"></option>
            <option value="Opera"></option>
```

```
            <option value="Safari"></option>
        </datalist>
        <input type="submit" value="Submit">
</form>
```

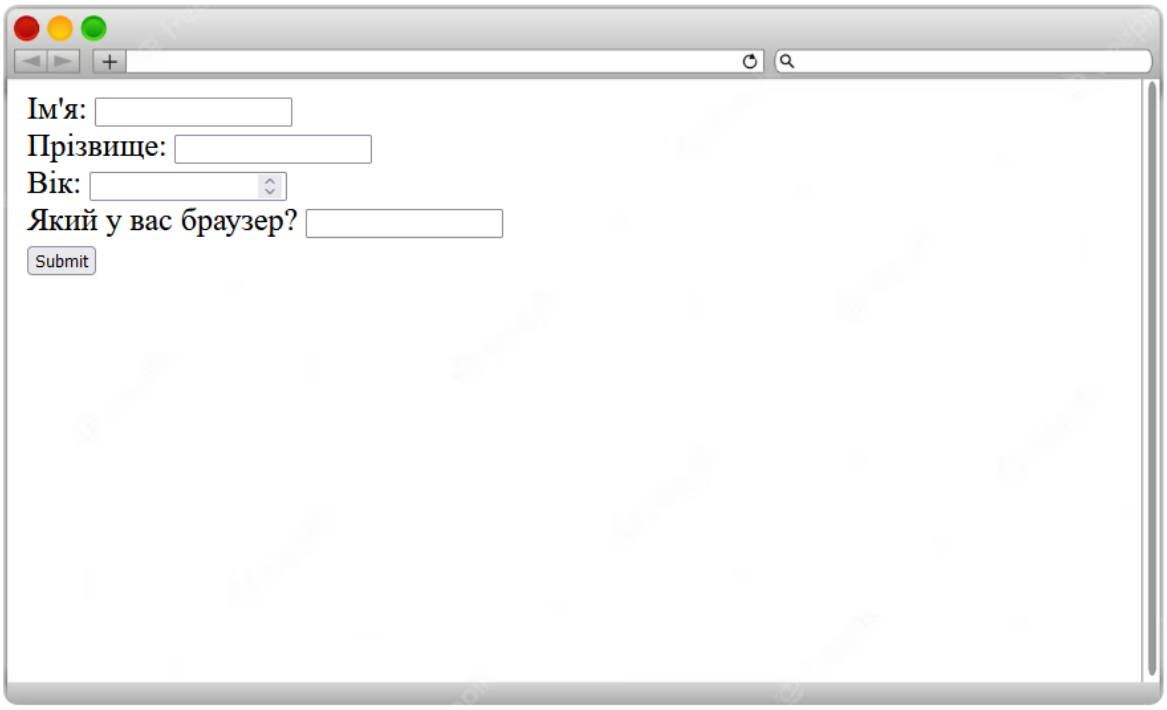

Рис. Форма з полями

- У формі маємо 2 текстові поля: для введення імені та прізвища.
- У полі для зазначення віку обмежено мінімальні та максимальні значення, щоб відвідувач не вводив неприпустимого числа число.
- Також створили список браузерів, з яких є можливість вибору (але можна ввести своє значення).
- Поле <input> з типом submit необхідно для надсилання даних на сервер. Задано неіснуючий скрипт, оскільки дані нема куди надсилати, але насправді використовують цю інформацію і додають до бази даних.

Фрейми

Фрейми надають відображати певні об’єкти (карти з GoogleMaps, відео з YouTube, окремий .html або .txt файли) в своїх рамках. Кожен із них завантажується ніби у своєму просторі і може окремо оновлюватись. Наразі представлений єдиним тегом: <iframe> .

Приклад - HTML

```
<p>Перший кадр:</p>
<iframe src="" width="50%" height="200">
</iframe>
<p>Другий кадр:</p>
<iframe src="" width="100%" height="300">
</iframe>
```

Наведений приклад досить умовний, так як посилання на кадр потрібно вказати конкретну адресу або документ. Сторонні сайти здебільшого не дозволяють встановити з'єднання з ними через кадр (спробуйте ввести будь-який). Для елементів можна задати ширину та висоту.

Рис. Фрейми

Стилі та семантика

Теги даної категорії несуть у собі сенс, зрозумілий розробникам, роботам, браузерам, але не відвідувачам сайту. Вони навіть не помітять, що на сайті є логічна верстка. У будь-якому випадку, щоб ресурс індексувався краще використання семантики є обов'язковим.

Що стосується стилів, то їх простіше заносити в окремі файли (простіше для редагування та наочніше).

Теги і

Це два нейтральних теги, які не мають семантичного навантаження. Основне завдання – з'єднати певний контент у єдине ціле (як рядок чи блок). Коли не можна за змістом віднести вміст до якогось семантичного елементу, але потрібна його стилізація, він обрамляється в один із цих тегів.

Приклад - HTML

```
<body>
    <div>Блок тексту</div>
    <span>Довільний текст у рядку</span>
</body>
```

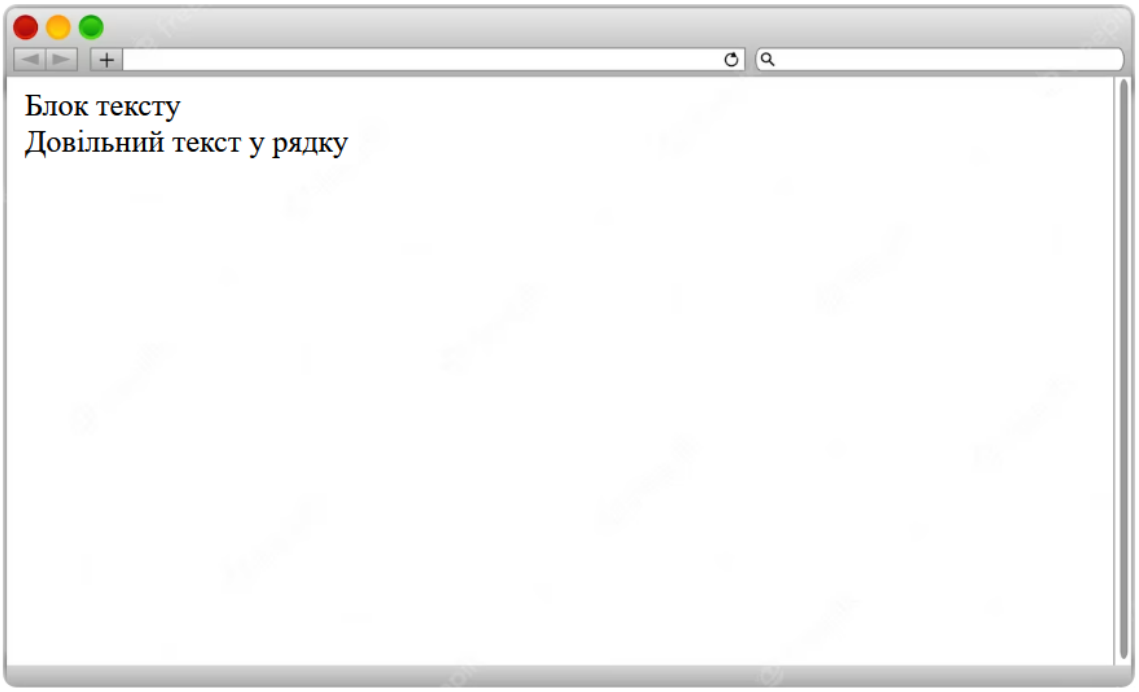

Рис. Блоковий та рядковий елементи

Семантичні теги

Такі теги не мають унікальних атрибутів, але можуть використовувати будь-які дієві чи глобальні.

Тег <details>

Призначається для даних, які користувач може розкривати чи ховати. Це зручно у разі додаткових пояснень, наявності необов'язкової інформації, яку можна пропустити. Застосовується разом із тегом <summary>.

Приклад – HTML

```
<details open>
    <summary>Додаткова інформація</summary>
    <p>Можна не читати</p>
</details>
```

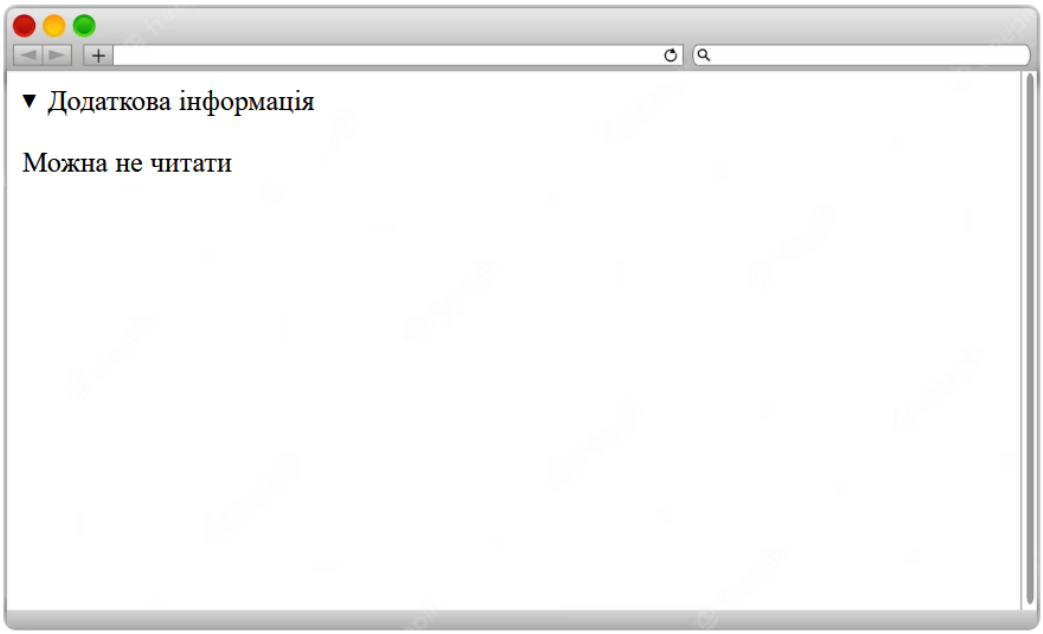

Рис. Тег <details>

Використано властивість open, яка примусово розкриває вміст елемента.

Тег <section>

Даним елементом виділяють окрему частину сторінки, що має закінчену думку. На відміну від <article> (який є самостійним та незалежним) застосовують для логічного розділу документа.

Наприклад, стаття складається з кількох розділів. Їх сміливо можна обернути у секції.

Приклад - HTML

```
<section>
    <h2>HTML</h2>
    <p>Мова розмітки дозволяє створювати власні веб-сторінки. Новий стандарт – HTML5.</p>
</section>
```

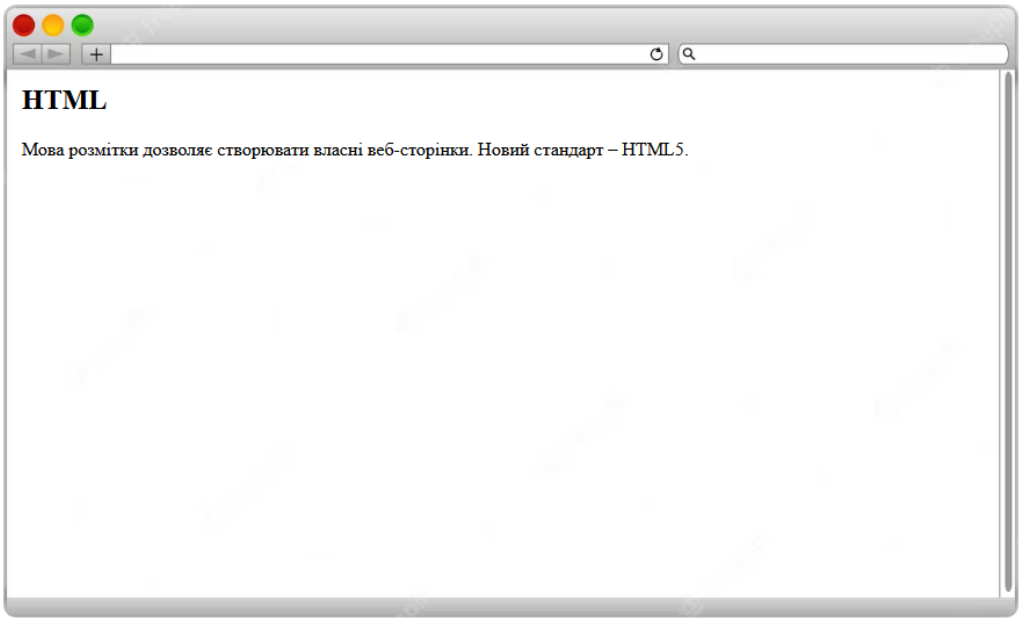

Рис. Тег <section>

Теги <h1>...<h6>

Як заголовки розділів і підрозділів використовуються теги групи h.H1 – найбільший заголовок, а h6 – найменший. Тут важливий не розмір тексту, а саме семантичне значення.

Так, на сторінці передбачається наявність одного заголовка h1, кількох h2 і решта рівнів. Укладати в ці теги необхідно не будь-який зміст, а важливий, оскільки пошукові системи орієнтуються на них у видачі результатів пошуку.

Приклад - HTML

```
<article>
    <h1>HTML</h1>
    <p>Текст</p>
    <h2>Історія розвитку</h2>
    <p>Текст</p>
    <h3>Основні елементи</h3>
    <p>Текст</p>
</article>
```

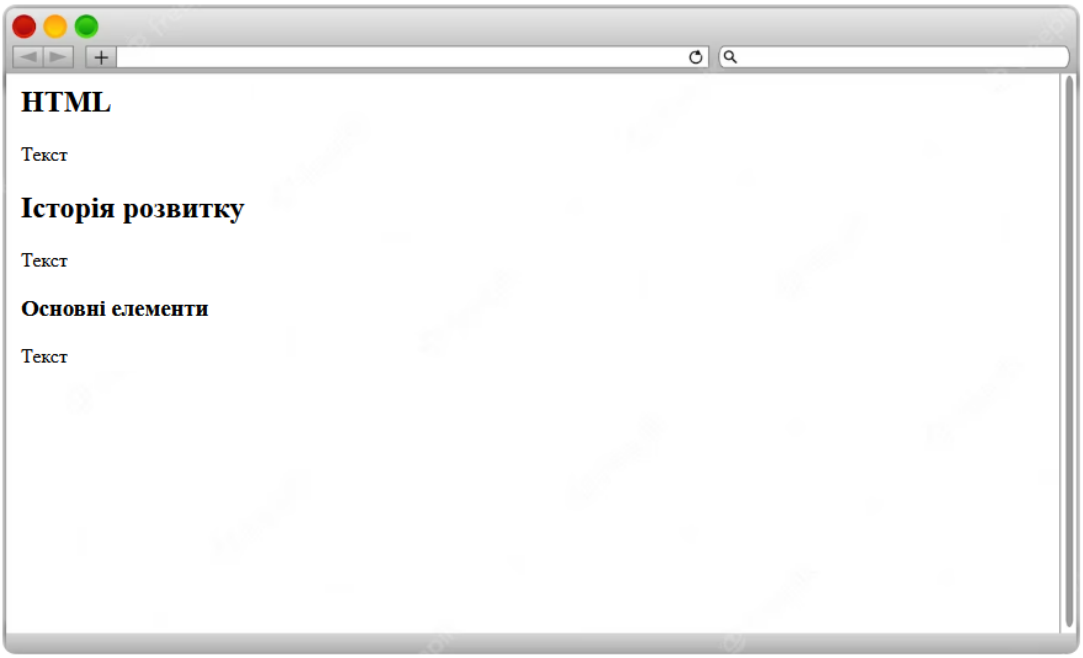

Рис. Теги заголовків

Сторінка містить інформацію про HTML, про що сказано в заголовку статті. Підзаголовками є розділи.

Програмування

Теги даного типу призначені для вбудовування до сторінки додаткового вмісту з використанням js або іншого типу. Найчастіше тут йдеться про елемент <script> та <noscript>. Всередині <script> міститься код для обробки веб-сторінки або її частин на стороні клієнта.

До цих тегів також відносять:

1. Тег <embed> (для відображення зовнішнього контенту).
2. Тег <object> (Відображає зовнішні ресурси).

3. Тег <param> (визначає параметри тега <object>).

Елементи <embed> та <object> практично ідентичні за своєю суттю. Перший з'явився раніше, а другий став альтернативою для більшої підтримки браузерами. Найчастіше вони замінюються більш відповідними тегами: <img>, <video>, <audio>, <iframe>.

Таким чином, веб-сторінка представляє сукупність тегів, властивостей і зовнішніх файлів, що підключаються. При створенні сайту обов'язково вказувати доктайп, весь код повинен знаходитися всередині тега <html>.Сторінка умовно поділяється на 2 блоки: заголовки та метадані, а також тіло документа з основним контентом.

Запитання

1. Чим відрізняється HTML-код від DOM-дерева?

HTML-код, який пишуть програмісти - це лише текстовий файл певного формату, а DOM - результат дій браузера, який створює об'єкти при парсингу HTML файлів.

2. У чому семантична відмінність тегів <article> і <section> ?

Тег <article> застосовується для незалежного, самостійного контенту, що займає основну частину конкретної сторінки. За допомогою тегу <section> виділяються окремі розділи сторінки, які не обов'язково є незалежними, але мають логічну завершеність.

3. Що означає запис <!DOCTYPE html> всередині HTML-файлу?

<!DOCTYPE html> вказує, що код файлу написаний відповідно до стандарту HTML5.

Завдання

Створіть форму, що дозволяє вибирати кілька з цих значень у полі хобі: кіно, музика, спорт. Використовуйте теги <label> та <input> .

Відповідь на завдання

Щоб користувач міг вибирати окремі елементи списку у формі, тегу <input> потрібно задати властивість checkbox .

Рішення - HTML

```
<form action="some.py">
     <input type="checkbox" id="cinema" name="cinema" value="Cinema">
     <label for="cinema">Люблю кіно</label><br>
     <input type="checkbox" id="music" name="music" value="Music">
     <label for="music">Люблю музику</label><br>
     <input type="checkbox" id="sport" name="sport" value="Sport">
     <label for="sport">Фанатею від спорту</label><br>
     <input type="submit" value="Підтвердити">
 </form>
```

Хороша практика під час створення форм – призначення підписів до полів (label для введення). За допомогою значення властивості name у тега <input> елемент <label> отримує доступ до

нього. Атрибути value та id використовуються JavaScript'ом та серверною частиною сайту для збереження значень у базі даних.

Джерела інформації

1. HTML: Структура сторінки, DOM-дерево, застосування тегів https://smartiqa.ru/courses/web/lesson-3-html