Все, що бачить користувач на сайті від шрифту, фону, меню, слайдеру створено за допомогою базової трійки засобів – мови HTML, таблиць стилів CSS і сценаріїв JavaScript.

HTML формує структуру сторінки (скелет), CSS визначає її зовнішній вигляд, а JS надає динаміку.

### HTML – мова розмітки тексту

Мова гіпертекстової розмітки HTML (HyperText Markup Language) є основним засобом для веб-сторінок, використовується для створення та візуального представлення веб-сторінок. Визначає структуру і описує зміст веб-сторінки в структурованій формі.

HTML-сторінка є звичайним текстовим документом, в якому використано спеціальні оператори – **теги** (*tag*) чи інша назва **дескриптори** (*descriptor*) для позначення, в якому вигляді буде виведено текстовий чи інший елемент у вікні браузера.

Прикладами таких операторів є <img>, <title>, <p>, <div>, <table> тощо.

HTML дозволяє формувати на сторінці сайту текстові блоки, додавати до них зображення, організовувати таблиці, додавати до дизайну сайту звуковий супровід, організовувати гіперпосилання з переходом до інших розділів сайту або ресурсів Інтернету і компонувати всі ці елементи між собою. За допомогою HTML можна створити як статичний так і динамічний сайт. Сторінки, які створено лише засобами HTML мають розширення .html.

***Гіперпосилання*** (*Hyperlink*) — це базовий функціональний елемент HTML-документу, який реалізовує зв'язок певного об'єкту веб-сторінки з іншим об’єктом. Для гіперпосилання може використовуватися як фрагмент тексту, так і графічний об'єкт, а сам зв'язок можна встановлювати як між об’єктами одного сайту, так і між об'єктами, що розміщені на різних сайтах Інтернету.

### *CSS – каскадна таблиця стилів*

CSS (Cascading Style Sheets) — це технологія опису зовнішнього вигляду документа, який створено засобами HTML.

CSS використовується для привласнення певних особливостей для елементів HTML-сторінки: колір, шрифт, розташування на сторінці тощо. З появою CSS стало можливим принципове розділення структури і опису документа. За рахунок такого розподілу стало можливим легке застосування єдиного стилю оформлення для кількох сторінок сайту, а також швидка зміна цього оформлення.

#### Переваги CSS:

* Застосування кількох варіантів дизайну сторінки для різних пристроїв перегляду, наприклад, для комп’ютера, планшету чи телефону.
* Зменшення часу завантаження сторінок сайту за рахунок перенесення правил опису даних до окремого CSS-файлу. В цьому випадку браузер завантажує лише структуру документа і дані, що містяться на сторінці. CSS-файл з правилами опису цих даних завантажується браузером лише один раз і зберігається в кеші браузера.
* Простота подальшої зміни дизайну. Не потрібно виправляти кожну сторінку, достатньо лише змінити кілька правил у CSS-файлі.
* Додаткові можливості оформлення. Наприклад, за допомогою CSS-правил можна застосувати обтікання певного блоку текстом або зробити так, щоб меню фіксовано знаходилося в певному місці при перегортанні сторінки.

### *JavaScript – мова сценаріїв*

JavaScript – це фрагменти програмного коду (скрипти), що надають динаміки для певних елементів сторінки. Програмний код на JavaScript вписується безпосередньо в текст HTML-сторінки і інтерпретується браузером в міру завантаження цього документа. За допомогою JavaScript можна динамічно реагувати на події, які пов'язані з діями відвідувача або змінами стану сторінки чи вікна.

Важливою особливістю JavaScript є об'єктна орієнтованість.

## **Серверні мови програмування**

Серверне програмування використовують для обробки дій користувача на динамічних складних проектах, таких як пошукові системи, електронна пошта, форуми, Інтернет-магазини тощо.

В цих випадках браузер приймає від відвідувача дані і надсилає їх до веб-серверу, який:

* Приймає запит на завантаження файлів (веб-сторінки, таблиці стилів, графічні зображення, фільми, звуки, архіви, виконувані файли тощо).
* Здійснює пошук цих файлів на дисках серверного комп'ютера.
* Скеровує завдання до відповідних програм, що виконують додаткові дії над файлами.
* Формує результати обробки серверних програм у HTML-код.
* Надсилає до браузера сформовану веб-сторінку в HTML-коді.

Для виконання завдань веб-сервера використовуються спеціальні програми, що працюють разом з веб-сервером на тому ж серверному комп'ютері. Вони називаються серверними програмами, не мають інтерфейсу. Цим вони відрізняються від клієнтських програм, що працюють безпосередньо з користувачем.

Сторінки, що формуються серверними програмами називаються динамічними, на відміну від статичних сторінок у форматі .html.

#### Серверні програми поділяються:

1. **Розширення веб-сервера**
2. **Активні серверні сторінки (РНР тощо).** Це звичайні статичні веб-сторінки, що збережені як файли, які, окрім звичайного HTML-коду, містять команди, що обробляються або самим веб-сервером або його розширеннями.
3. **Серверні сценарії, написані мовою, що інтерпретується (PНР, Python, Java, Ruby).** Сценарії оброблення даних при різних діях користувача на сайті.

**Розширення веб-сервера** - різновид серверних програм, що представляють бібліотеки, в яких реалізовано логіку серверної програми. Такі бібліотеки вбудовуються в програму веб-сервера і працюють як її невід'ємна частина.

Розширення веб-серверів Internet Information Server фірми Microsoft створюються у вигляді бібліотек DLL

Метою розширень веб-сервера є економна витрата системних ресурсів. Для обробки всіх наборів даних користувача запускається лише один екземпляр розширення, який забирає значно менше ресурсів, ніж багато запущених програм. Однак розширення важче створювати і налагоджувати, оскільки вони працюють як частина веб-сервера і будь-яка помилка в розширенні призведе до зависання сервера.

**Активні серверні сторінки** - це звичайні веб-сторінки, що містять особливі серверні сценарії (програмний код в HTML-коді) і виконуються самим веб-сервером або його розширеннями. Перевагами активних серверних сторінок є легкість та швидкість написання, і простота відлагодження.

**Серверні сценарії** подібні до активних серверних сторінок, проте, представляють собою "чистий" програмний код, без HTML-фрагментів. Інтерпретатор коду представлено як розширення веб-сервера. Сценарії, зазвичай, пишуться мовами програмування РНР, Java, Perl, Python, Ruby. Фактично писати сценарії можна будь-якою мовою програмування, для якої є інтерпретатор.

Серверні сценарії споживають значно більше системних ресурсів, ніж активні сторінки, бо для обробки кожного набору даних користувача запускається власна копія інтерпретатора, а інтерпретатор, в свою чергу, витрачає багато ресурсів на обробку сценарію. І все ж, незважаючи на це, сценарії - найпопулярніший спосіб створення серверних програм.

Слайд 1

# **Основи HTML**

HTML - не є мовою програмування, це мова розмітки, яка використовується для розміщення елементів (тексту, картинок, кнопок, таблиць тощо) на сторінці, тобто, для складання структури. Сторінка може мати складну або просту структуру, все залежить від досвіду веб-розробника.

HTML складається з ряду елементів, які використовують для того, щоб різні частини сторінки мали певний вид або спрацьовували певним способом. За призначення, вигляд та розташування елементу відповідають спеціальні оператори (теги), які представляють собою команду, що поміщена в кутові дужки. Теги, зазвичай, є скороченою назвою або абревіатурою того завдання, що має виконати браузер наприклад:

<p> - тег, що позначає абзац (paragraph).

<h1> - тег, що позначає заголовок 1 рівня (header 1 level). (1)

Для збільшення інформативності тегу використовуються атрибути, що вказують текстові чи числові значення.

<table width=”50%” align=”center”>…</table> (2)

В наведеному прикладі показано тег таблиці, яка має займати 50% від ширини вікна браузера (або блоку, в якому знаходиться) і розміщена по центру.

В мові HTML застосовують теги: одинарні, парні та коментарі.

### Одинарні теги HTML

Одинарні теги складаються з одного оператора. Наприклад, тег <br> - перенесення на новий рядок, <hr> - роздільна лінія, <img> - вставка зображення. Властивості для одиночного тегу вказуються всередині через атрибути, наприклад:

<img src=”image/picture.jpg” width=”200” border=”1”> (3)

В наведеному прикладі в одиночному тезі <img> прописано атрибути, які вказують на місце розташування і назву графічного файлу, ширину зображення в 200пікселів і наявність рамки навколо зображення шириною в 1 піксел.

### Парні теги HTML

Парних тегів є значно більше і вони вказують браузеру початок і закінчення певного структурного елементу: рядка, блоку, таблиці, заголовку

Наприклад, для того, щоб певний текст відобразився в браузері як рядок, слід застосувати теги, що позначають елемент-параграф (рис. 1.1):



Рисунок 1.1 ­ Склад HTML-елементу

Основними частинами елемента є:

* **Початковий тег:** Він складається з назви елемента (<p>) і є ознакою початку елемента, з цього моменту тег починає впливати на той вміст, що слідує за ним.
* **Кінцевий тег:** виглядає як і початковий, але містить слеш перед назвою тега (</p>). Він вказує на закінчення елементу. Закінчення в парних тегах є обов’язковими, інакше браузер може невірно відобразити вміст.
* **Вміст:** в цьому випадку вмістом є простий текст.
* **Елемент:** початковий тег + вміст + кінцевий тег = елемент.

Всередині елементів можна вкладати інші елементи - це називається вкладеністю. Наприклад, якщо потрібно зробити акцент на певному слові і виділити його грубішим шрифтом.

<p>Цей фрагмент є <strong>окремим рядком</strong> тексту</p> (4)

Якщо застосовується розміщення елементів всередині інших, слід обов’язково зберігати порядок вкладеності. В наступному прикладі наведено невірний запис:

<p>Цей фрагмент є <strong>окремим рядком тексту</p></strong> (5)

Елементи повинні відкриватися і закриватися правильно таким чином, щоб явно перебувати всередині або зовні один одного. Якщо вони перекриваються так, як в прикладі вище, то браузер сам приймає рішення як відображати, що може привести до непередбачуваного результату.

### Теги для коментарів

Коментування в HTML необхідно для покращення читабельності коду. В коментарях, зазвичай, вказується пояснення ділянки коду, що спрощує процес редагування HTML сторінки в подальшому. Розробнику легше орієнтуватися і вносити зміни, оскільки коли коду стає дуже багато, то в ньому легко заплутатися, можна поставити зайвий тег або, навпаки, не закрити його.

Тег для коментаря починається з кутової дужки, знак оклику, два дефіси, текст коментаря, два дефіси, закінчується кутовою дужкою.

<!-- текст коментаря --> (6)

Коментарі в HTML не відображаються на сторінці в браузері користувачеві, їх можна побачити лише в коді веб-сторінки. Для зручності слід коментувати не лише початок певного блоку коду, а й його завершення

Слайд 2

## **Структура HTML-документа**

Для сторінок будь якої складності існує стандартна структура, яка має обов’язкові елементи.

<!DOCTYPE html>

<html>

 <head>

 <meta charset = "utf-8">

 <title>Тестова сторінка</title>

 </head>

 <body>

 <p>Моя перша сторінка</p>

 </body>

</html>

* **<!DOCTYPE html>**: посилання на набір правил doctypes, яким має слідувати HTML-сторінка, також вказівка на автоматичну перевірку помилок і інші корисні речі.
* **<html>** є кореневим елементом документа. Всі інші елементи містяться всередині тегів <html> ... </html>. Все, що знаходиться за межами цих тегів, не сприймається браузером як код HTML і не обробляється..
* **<head> елемент** містить службову, довідкову та додаткову інформацію, яка призначена для браузера, пошукових систем, сервера і не відображається для відвідувача сторінки. В цьому елементі міститься технічна інформація про сторінку: ключові слова, короткий опис сторінки, під’єднані шрифти, стилі, скрипти, зазначається тип кодування шрифтів і багато іншого.
* **<body> елемент** містить весь контент, який бачать відвідувачі - текст, зображення, відео, анімація тощо.

Елементи, що знаходяться всередині елементу <html>, утворюють дерево документа, так звану об'єктну модель документа, **DOM (Document Object Model)**. Елемент <html> є кореневим елементом, body і head – структурними елементами.



Рисунок 1.2 ­ Найпростіша структура веб-сторінки

У взаємодії елементів веб-сторінки задіяні "родинні стосунки" між елементами. Відносини між множинними вкладеними елементами підрозділяються на батьківські, дочірні та сестринські.

* **Батьківський елемент** - елемент, що пов'язаний з іншими елементами нижчого рівня, і що знаходиться на дереві вище за них. На рис.1.2 <html> є батьківським тільки для <head> і <body>. Елемент <body> є батьківським для всіх елементів, що містяться в ньому: <h1>, <p>, <span>, <nav> і т.д. Тег <p> є батьківським тільки для <span>.
* **Дочірній елемент** - елемент, що розташований всередині іншого елементу. На рис.1.2 елементи <h1>, <h2>, <p> і <nav> є дочірніми по відношенню до <body>.
* **Сестринський елемент** - елемент, що має загальний батьківський елемент з даним, так звані елементи одного рівня. На рис.2.2 <head> і <body> - елементи одного рівня, так само як і елементи <h1>, <h2> і <p> є між собою сестринськими.

### Слайд 3

### Елемент <head>

Інформація всередині тегу не відображається у вікні браузера, однак, ці дані вказують браузеру, серверу чи пошуковій системі, як слід обробляти сторінку.

* **<title> елемент** є обов'язковим для розділу <head>, оскільки текст, що розміщений всередині нього, відображається у заголовку браузера. Текст заголовку повинен містити максимально повний опис вмісту сторінки.
* **<meta> елемент.** Елемент <head> може містити кілька елементів <meta>, які відповідно до використаних атрибутів несуть різну інформацію. За допомогою <meta> можна зазначити короткий опис сторінки і ключові слова для пошукових машин, автора html-документу та інші відомості.
* **<meta charset="тип кодування">** повідомляє браузеру кодування сторінки. Кодування HTML-сторінки потрібно вказувати для того, щоб веб-браузер міг правильно відображати текст на сторінці. Якщо браузер неправильно «вгадає» кодування, то замість тексту будуть відображатися ієрогліфи. Стандартом на сьогодні є кодування - utf-8, що містить символи зі всіх відомих природних мов, тому цей тег має вигляд <meta charset=" utf-8">.
* **Елемент <style>.** Всередині цього елемента задаються стилі, які використовуються на сторінці.
* **Елемент <link>.** Даний елемент визначає відношення між поточною сторінкою та іншими документами. Таких елементів на сторінці може бути кілька. Наприклад, вказати зв’язок сторінки з файлом зі стилями (окремий файл з розширенням .css), тоді посилання на файл зі стилями буде виглядати так

 <link href="style.css"> (7).

Обов’язковим є вказування атрибуту rel, де вказується як цей документ буде розпізнано браузером

(<link rel="stylesheet" href=”style.css”> (8)

- документ є таблицею стилів,

<link rel="icon" href="favicon.png"> (9)

- документ є іконкою). Додатково вказують тип документу атрибутом type

(<link rel="stylesheet" type="text/css" href="style.css">, (10)

 <link rel=" icon" type="image/x-icon" href="favicon.png">) (11)

* **Елемент <script>.** Дозволяє приєднувати до документа різні сценарії. Закриваючий тег є обов'язковим, при цьому текст сценарію може розташовуватися або всередині цього елемента, або в зовнішньому файлі. Якщо текст сценарію розташований в зовнішньому файлі, то він під’єднується за допомогою атрибутів елементу, наприклад:

 <script src=" scripts.js"></script> (12)

### Елемент <body>

Всередині елементу <body> міститься інформація, яка відображається в браузері. В таблиці <https://w3schoolsua.github.io/html/html_elements.html#gsc.tab=0> наведено повний перелік тегів.

**Слайд 4**

За призначенням теги можна умовно розподілити:

* **Контейнерні:** div, article, aside, header, footer, span, p, h1-h6, ul, ol
* **Функціональні:** a, img, table, form, input, audio, video, canvas
* **Фрейми:** iframe

У тегів можуть бути HTML атрибути, які повідомляють браузеру, яким чином має відображатися той чи інший елемент сторінки та особливості відображення даного елементу. Значення атрибута завжди заключають в лапки " ". Назви та значення атрибутів не є чутливими до регістру, але, рекомендується набирати їх у нижньому регістрі.

<тег атрибут1="значення1" атрибут2=”значення1 значення2”>контент</тег> (13)

Атрибути можна поділити на **універсальні (глобальні)** які можна використані для будь-якого HTML-елемента, та **власні**, які притаманні лише цьому тегу.

#### Універсальними атрибутами є

* **title** Вказує додаткову текстову інформацію про елемент, що відображається у спливаючій підказці над елементом
* **style** Вказує код CSS, що застосовується для оформлення даного елементу.
* **class** Вказує назву класу для елемента, сам клас описано в таблиці стилів.

Атрибут *class* застосовний до всіх HTML-елементів. Кожному конкретному елементу можна привласнити **кілька значень class**. Назви класу повинні складатися з букв, цифр, дефісів і нижніх підкреслень і починатися тільки з літери.

* **id** Вказує унікальний ідентифікатор елемента. Атрибут *id* застосовний до всіх HTML-елементів. Кожному конкретному елементу можна привласнити **лише одне значення id**. Назви id повинні складатися тільки з букв, цифр, дефісів і нижніх підкреслень і повинні починатися тільки з літер.

**Слайд 5**

#### Умовно елементи можна поділити на групи

* **Блокові елементи.** Характеризуються тим, що починаються з нового рядка, займають всю доступну ширину, висота елемента визначається його вмістом.
* **Рядкові елементи.** Є безпосередньою частиною іншого елемента, наприклад, текстового абзацу. В основному використовуються для зміни вигляду тексту, його логічного виділення чи акцентування.

### Блокові елементи

#### Загальні характеристики блокових елементів

* На початку і в кінці блокових елементів, автоматично ставиться перенесення рядка.
* За замовченням блокові елементи займають всю ширину батьківського елементу.
* В блокових елементах можна керувати шириною і висотою: width, height.
* В блокових елементах можна керувати зовнішніми і внутрішніми відступами.
* Об'єктами, що знаходяться всередині блокового елементу, можна керувати за допомогою горизонтального та вертикального вирівнювання.
* Всередині блокових елементів можна розміщувати інші блокові елементи, а також рядкові елементи.

#### Типові блокові елементи

* **<div>** Універсальний блоковий елемент. (14)

***Приклад:***

<style>

.block1 {

 width: 200px;

 background: #ccc;

 padding: 5px;

 padding-right: 20px;

 border: solid 1px black;

 float: left;

 }

 .block2 {

 width: 200px;

 padding: 5px;

 border: solid 1px black;

 float: left;

 position: relative;

 left: 10px;

 }

</style>

<p>Приклад використання тега div</p>

<div class="block1">Собаки здавна є помічниками людей.

 Вміють виконувати команди. Вони несуть службу на кордоні, митниці. Собаки - гарні рятівники. Вони допомагають шукати людей. </div>

 <div class="block2">Тварини служать поводирями для незрячих та немічних. А ще вони надійні друзі, сумують без своїх господарів.

 Про цих помічників знімають фільми, пишуть у книжках. </div>



Рисунок 1.3 ­ Приклад використання тега <div>

Слайд 6

* **<article>, <aside>, <header>, <footer>** Семантичні блоки для позначення структурних ділянок сторінки. (15)
1. **<article>**

Тег <article> визначає незалежний самодостатній вміст.

***Приклад:***

<article>

<h2>Google Chrome</h2>

<p>Google Chrome – це веб-браузер, розроблений Google, випущений у 2008 році. Chrome є найпопулярнішим веб-браузером у світі сьогодні!</p>

</article>

<article>

<h2>Mozilla Firefox</h2>

<p>Mozilla Firefox — веб-браузер із відкритим вихідним кодом. Firefox став другим за популярністю веб-браузером.</p>

</article>

<article>

<h2>Microsoft Edge</h2>

<p>Microsoft Edge — веб-браузер, розроблений Microsoft, випущений у 2015 році. Microsoft Edge замінив Internet Explorer.</p>

</article>

Слайд 7



Рисунок 1.4 ­ Приклад використання тега <article>

Слайд 8

1. **<aside>**

Тег <aside> визначає певний вміст, окрім вмісту, у якому він розміщений.

***Приклад:***

<style>

aside {

 width: 40%;

 padding-left: .5rem;

 margin-left: .5rem;

 float: right;

 box-shadow: inset 5px 0 5px -5px #29627e;

 font-style: italic;

 color: #29627e;

}

aside > p {

 margin: .5rem;

}

p {

 font-family: 'Fira Sans', sans-serif;

}

</style>

<p>Тег `aside` вказує, що контент, який міститься всередині цього елемента, пов'язаний з основним контентом даної сторінки, але не є його частиною</p>

<aside>

 <p>HTML тег `aside` визначає блок збоку від контенту для розміщення рубрик, посилань на архів, міток та іншої інформації.</p>

</aside>

<p>У певному сенсі його застосування аналогічно впровадженню коментаря в тіло тексту (подібно даному коментарю). Контент в круглих дужках надає додаткову інформацію про елемент, що містить цей контент.</p>

Слайд 9



Рисунок 1.5 ­ Приклад використання тега <aside>

Слайд 10

1. **<header>**

Тег <header> призначений для того, щоб позначити розділ HTML-сторінки як заголовок (контейнер для вступного вмісту) або набір навігаційних посилань.

***Приклад:***

<style>\*{

margin:0;

width:100%;

height:100%;

}

header{

width:100%;

height:10%;

background:#f0cfcf;

}

section{

width:100%;

height:80%;

background:#00cf0f;

}

</style>

<body>

 <header>Заголовок

 <nav>

 <a href='#'>Головна</a>

 <a href='#'>Каталог</a>

 <a href='#'>Контакти</a>

 </nav>

 </header>

 <section>

 <p>

 Стаття перша

 </p>

 </section>

 </body>

Слайд 11



Рисунок 1.6 ­ Приклад використання тега **<header>**

Слайд 12

1. **<footer>**

Тег <footer> визначає нижній колонтитул для документа або розділу.

Елемент <footer> зазвичай містить:

* інформація про авторство
* інформація про авторські права
* контактна інформація
* карта сайту
* посилання на початок списку
* пов’язані документи

***Приклад:***

<style>\*{

margin:0;

width:100%;

height:80%;

}

header{

width:100%;

height:10%;

background:#f0cfcf;

}

section{

width:100%;

height:60%;

background:#00cf0f;

}

footer{

width:100%;

height:10%;

background:#c0cf0f;

}

</style>

<body>

 <header>Заголовок

 <nav>

 <a href='#'>Головна</a>

 <a href='#'>Каталог</a>

 <a href='#'>Контакти</a>

 </nav>

 </header>

 <section>

 <p>

 Стаття перша

 </p>

 </section>

 <section>

 <p>

 Стаття перша

 </p>

 </section>

 <footer>Футер</footer>

 </body>

Слайд 13



Рисунок 1.7 ­ Приклад використання тега **<footer>**

Слайд 14

* **<p>, <h1>-<h6>, <blockquote>** Елементи для HTML тексту. (16)
1. Тег <p> визначає параграф (абзац).

***Приклад:***

<p style="text-align:right">Це деякий текст в параграфі.</p>



Рисунок 1.8 ­ Приклад використання тега **<p>**

1. Теги від <h1> до <h6> використовуються для визначення заголовків HTML.

<h1> визначає найважливіший заголовок. <h6> визначає найменш важливий заголовок.

***Приклад:***

<h1 style="background-color:DodgerBlue;">Слава Україні!</h1>

<h2 style="color:Tomato;">Слава Україні!</h2>



Рисунок 1.9 ­ Приклад використання тега <h1> до <h6>

Слайд 15

1. **<blockquote>**

Тег <blockquote> визначає розділ, що цитується з іншого джерела.

***Приклад:***

<blockquote cite="http://developer.mozilla.org">

 <p>This is a quotation taken from the Mozilla Developer Center.</p>

</blockquote>

URL, що вказує на вихідний документ або повідомлення, звідки було взято цитату. Цей атрибут призначений для того, щоб послатися на інформацію, яка пояснює контекст, або посилання, з яких було взято цитату.

Слайд 16

* **<hr>** Горизонтальна лінія. (17)

***Приклад:***

<h1>Основні мови Інтернету</h1>

<p>HTML є стандартною мовою розмітки для створення вебсторінок. HTML описує структуру вебсторінки та складається з ряду елементів. Елементи HTML повідомляють браузеру, як відображати вміст.</p>

<hr>

<p>CSS — це мова, яка описує, як елементи HTML мають відображатися на екрані, папері чи в інших носіях. CSS заощаджує багато роботи, оскільки він може керувати макетом кількох веб-сторінок одночасно.</p>

Рисунок 1.10 ­ Приклад використання тега **<hr>**

Слайд 17

* **<ol>, <ul>** Встановлює нумерований (маркірований) список. (18)

***Приклад:***

<ol>

 <li>Coffee</li>

 <li>Tea</li>

 <li>Milk</li>

</ol>

<ol start="50">

 <li>Coffee</li>

 <li>Tea</li>

 <li>Milk</li>

</ol>



Рисунок 1.11 ­ Приклад використання тега </ol>

Тег <ul> визначає невпорядкований (маркований) список.

***Приклад:***

<ul>

 <li>Coffee</li>

 <li>Tea</li>

 <li>Milk</li>

</ul>



Рисунок 1.12 ­ Приклад використання тега <ul>

Слайд 18

* **<table>** Створює таблицю. (19)

Тег <table> визначає HTML таблицю.

Таблиця HTML складається з одного елемента <table> і одного або кількох елементів [<tr>](https://w3schoolsua.github.io/tags/tag_tr.html), [<th>](https://w3schoolsua.github.io/tags/tag_th.html) та [<td>](https://w3schoolsua.github.io/tags/tag_td.html).

Елемент <tr> визначає рядок таблиці, елемент <th> визначає заголовок таблиці, а елемент <td> визначає клітинку таблиці.

***Приклад:***

<html>

<head>

<style>

table, th, td {

 border: 1px solid black;

 border-collapse: collapse;

}

</style>

</head>

<body>

<table>

 <tr>

 <th>Month</th>

 <th>Savings</th>

 </tr>

 <tr>

 <td>January</td>

 <td>$100</td>

 </tr>

 <tr>

 <td>February</td>

 <td>$80</td>

 </tr>

</table>

</body>

</html>



Рисунок 1.13 ­ Приклад використання тега <table>

Слайд 19

* **<form>** Встановлює форму на веб-сторінці. (20)

***Приклад:***

HTML-форма з двома полями введення та однією кнопкою надсилання:

<form action="/action\_page.html" method="get">

 <label for="fname">Ім’я:</label>

 <input type="text" id="fname" name="fname"><br><br>

 <label for="lname">Прізвище:</label>

 <input type="text" id="lname" name="lname"><br><br>

 <input type="submit" value="Надіслати">

</form>



Рисунок 1.14 ­ Приклад використання тега **<form>**

### Рядкові елементи

#### Загальні характеристики рядкових елементів

* На відміну від блокових, рядкові елементи слідують один за одним і НЕ переносяться на новий рядок.
* В рядкових елементах не можна керувати шириною і висотою: width, height.
* В рядкових елементах можна встановлювати лише внутрішні відступи (padding) та зовнішні бічні відступи (margin).
* Рядкові елементи НЕ займають всю ширину батьківського елементу.
* Оскільки рядкові елементи НЕ займають всю ширину батьківського елементу, то елементами, що знаходяться всередині рядкових елементів, не можна керувати за допомогою горизонтального чи вертикального вирівнювання.
* Всередині рядкових елементів можна розміщувати інші рядкові елементи, якщо розмістити там блоковий елемент, то він розірве логічний потік і розміститься з нового рядка.

Слайд 20

#### Типові рядкові елементи

**<a>** Є одним з важливих елементів HTML і призначений для створення посилань. (21)

Тег <a> визначає гіперпосилання, яке використовується для переходу з однієї сторінки на іншу.

Найбільш важливим атрибутом елемента <a> є атрибут href, який вказує на призначення посилання.

***Приклад:***

<a href="https://www.w3schools.com">Посетите W3Schools.com!</a>

****

Рисунок 1.15 ­ Приклад використання тега **<a>**

**<span>** Універсальний рядковий елемент. (22)

Тег <span> — це вбудований контейнер, який використовується для позначення частини тексту або частини документа.

Тег <span> дуже схожий на елемент [<div>](https://w3schoolsua.github.io/tags/tag_div.html), але <div> є блоковим елементом, а <span> є вбудованим (рядковим) елементом.

Елемент <span>, який використовується для забарвлення частини тексту:

***Приклад:***

<p>Слава <span style="color:blue">Україні</span> </p>



Рисунок 1.16 ­ Приклад використання тега <span>

Слайд 21

**<br>** Розірвання рядка. (23)

***Приклад:***

<p>Щоб примусово<br> розбивати рядки<br> в тексті,<br> використовуйте елемент br<br>.</p>



Рисунок 1.16 ­ Приклад використання тега **<br>**

**<img>** Вставлення зображення. (24)

Тег <img> використовується для вставлення зображення на сторінку HTML.

Тег <img> має два обов’язкові атрибути (згідно специфікації HTML5):

* src - Вказує шлях до зображення
* alt - Визначає альтернативний текст для зображення, якщо зображення з якоїсь причини не може бути відображено.

***Приклад:***

<style>

.fit-picture {

 width: 250px;

}

</style>

<img

 class="fit-picture"

 src="/Основи інтернет технологій/images/grapefruit.jpg"

 alt="Grapefruit slice atop a pile of other slices" />



Рисунок 1.17 ­ Приклад використання тега **<img>**

Слайд 22

## **HTML текст**

Елементи для форматування тексту несуть змістовне навантаження і, зазвичай, задають для тексту структурне та стильове оформлення, наприклад, виділяють текст грубішим шрифтом або відображають його іншим шрифтом.

Грамотно відформатований текст надає для пошукових систем відомості, які фрагменти тексту мають важливий зміст, за якими з них слід ранжирувати веб-сторінку в пошуковій видачі.

#### Елемент абзацу <p>

Розділяє текст на окремі абзаци, відокремлюючи їх збільшеною відстанню. Браузер автоматично додає верхній і нижній відступ.

#### Елементи заголовків <h1>-<h6>

Заголовки є важливим елементом веб-сторінки, вони допомагають впорядкувати текст, сформувати його структуру. У специфікації HTML існує шість рівнів заголовків, завдяки яким можна легко виділяти теми і підтеми.

Заголовки є блоковими елементами, завжди починаються з нового рядка. Браузер автоматично додає перед заголовком і після нього збільшені відступи.

Текст в заголовках впливає на індексацію сайту пошуковими системами, оскільки багато роботів звертають увагу саме на вміст заголовків сайту, тому, краще завжди використовувати ці елементи, міняючи зовнішній вигляд і розмір за допомогою стилів.

При використанні заголовків необхідно враховувати їх ієрархію, тобто за <h1> повинен слідувати <h2> і т.д. Також, не рекомендується вкладення інших елементів в <h1> ... <h6>.

#### Елементи для форматування тексту

* **<strong>** задає грубіше накреслення шрифту, вказуючи браузеру на важливість тексту. Є сучасною альтернативою для попереднього елементу <b>.

Вміст усередині зазвичай відображається **жирним шрифтом**.

***Приклад:***

<strong>Це важливий текст!</strong>



Рисунок 1.18 ­ Приклад використання тега **<strong>**

* **<em>** відображає текст нахиленим шрифтом (курсивом). Є сучасною альтернативою для елементу <i>, який на сьогоднішній день охоче застосовують для розміщення іконок. Для нахилу тексту можна використати елемент <cite>.

***Приклад:***

<p>Ти <em>маєш</em> поспішати яскраво жити!</p>

<p>Ми <em>не можемо</em> дозволити собі так погано жити.</p>



Рисунок 1.19 ­ Приклад використання тега **<em>**

Слайд 23

* **<q>** додавання зовнішніх лапок до фрагменту.

***Приклад:***

<p>WWF's goal is to:

<q>Build a future where people live in harmony with nature.</q>

We hope they succeed.</p>



Рисунок 1.20 ­ Приклад використання тега **<q>**

* **<code>** зміна шрифту на більш технічний шрифт.

***Приклад:***

Визначає текст як комп’ютерний код у документі:

<p>HTML тег <code>button</code> визначає клікабельну кнопку.</p>

<p>CSS властивість <code>background-color</code> визначає колір фону елемента.</p>



Рисунок 1.21 ­ Приклад використання тега **<code>**

Слайд 24

* **<del>** перекреслений текст.

HTML тег <del> (від англ. deleted - видалений) використовується для виділення тексту, який був вилучений у новій версії документа.

Подібне форматування дозволяє відстежити, які зміни в тексті документа були зроблені.

***Приклад:***

<h3>Приклад використання тегу <code>del</code></h3>

<p>Кількість студентів, що навчаються в педагогічному університеті станом на 2019 рік становить <del>4589</del> </del> <ins>5004</ins>. Вибуло <del>240</del> студентів. </p>



Рисунок 1.22 ­ Приклад використання тега **<del>**

* **<ins>** підкреслений текст.

Тег <ins> визначає текст, який було вставлено в документ. Браузери зазвичай підкреслюють вставлений текст.

***Приклад:***

<p>Мій улюблений колір <del>синій</del> <ins>червоний</ins>!</p>



Рисунок 1.23 ­ Приклад використання тега **<ins>**

Слайд 25

* **<sub>** використовується для створення нижніх індексів. Зсуває текст нижче рівня рядка, зменшуючи його розмір.

***Приклад:***

<p>Цей текст містить <sub>підрядковий</sub> текст.</p>



Рисунок 1.24 ­ Приклад використання тега **<sub>**

* **<sup>** використовується для створення ступенів. Зсуває текст вище рівня рядка, зменшуючи його розмір.

***Приклад:***

<p>Цей текст містить <sup>надрядковий</sup> текст.</p>



Рисунок 1.25 ­ Приклад використання тега **<sup>**

Слайд 26

* **<pre>** дозволяє вивести текст на екран, зберігши початкове форматування. Пробіли і переноси рядків при цьому не видаляються.

***Приклад:***

<pre>

Текст у попередньому елементі

відображається шрифтом

фіксованої ширини, і він зберігає

обидва пробіли та

розриви рядків

</pre>



Рисунок 1.26 ­ Приклад використання тега **<pre>**

Слайд 27

* **<blockquote>** Використовується для оформлення фрагменту особливим блоком, наприклад, цитати, виділяючи його відступами і переносами рядків.

***Приклад:***

<blockquote cite="https://www.worldwildlife.org/who/index.html">

Ось вже 50 років WWF захищає майбутнє природи. Всесвітній фонд дикої природи (WWF), найбільша у світі природоохоронна організація, працює в 100 країнах, і її підтримують 1,2 мільйона членів в США та близько 5 мільйонів в усьому світі.

</blockquote>



Рисунок 1.27 ­ Приклад використання тега **<blockquote>**

Слайд 28

* **<br>** Переносить текст на наступний рядок, створюючи розрив рядка.

***Приклад:***

<p>Щоб примусово<br> розбивати рядки<br> в тексті,<br> використовуйте елемент br<br>.</p>

******

Рисунок 1.28 ­ Приклад використання тега **<br>**

**Слайд 29**

* **<hr>** Використовується для поділу контенту на веб-сторінці. Відображається у вигляді горизонтальної лінії.

***Приклад:***

<h1>Основні мови Інтернету</h1>

<p>HTML є стандартною мовою розмітки для створення вебсторінок. HTML описує структуру вебсторінки та складається з ряду елементів. Елементи HTML повідомляють браузеру, як відображати вміст.</p>

<hr>

<p>CSS — це мова, яка описує, як елементи HTML мають відображатися на екрані, папері чи в інших носіях. CSS заощаджує багато роботи, оскільки він може керувати макетом кількох веб-сторінок одночасно.</p>

******

Рисунок 1.29 ­ Приклад використання тега **<hr>**