**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний університет «Одеська юридична академія»**

Кафедра інформаційних технологій

# Веб-дизайн та HTML-програмування

**Навчально-методичний посібник**
**для підготовки бакалаврів з галузі знань 06 «Журналістика»**
**за спеціальністю 061 «Журналістика»**

**Одеса 2017**

УДК 004.77 (075.8)
ББК 32.973.202
Т76

Автори:

**Трофименко О. Г.** – кандидат технічних наук, доцент, доцент кафедри інформаційних технологій Національного університету «Одеська юридична академія»

**Козін О. Б.** – кандидат технічних наук, доцент, доцент кафедри інформаційних технологій Національного університету «Одеська юридична академія»

**Рекомендовано навчально-методичною радою Національного університету «Одеська юридична академія», протокол 7 від 22 червня 2017 р.**

Рецензенти:

**Кузнєцова Т. В.** – доктор наук із соціальних комунікацій, професор, декан факультету журналістики Національного університету «Одеська юридична академія»

**Троянський О. В.** – кандидат технічних наук, доцент, директор інституту інформаційної безпеки, радіоелектроніки та телекомунікацій Одеського національного політехнічного університету

Посібник призначений для підготовки до лабораторних і самостійних робіт, які виконуються у межах навчальної дисципліни «Веб-дизайн та HTML-програмування». Розглянуто основні засоби проектування, макетування та редагування веб-сайтів (HTML, CSS, онлайн-конструктори сайтів) з метою набуття теоретичних знань і практичних навиків створювати якісні й цікаві вебсайти. Адже, такі вміння є важливою складовою інформаційної культури майбутнього журналіста, оскільки від того, як він зможе подати у світовому інформаційному просторі себе, коло своїх професійних та особистих інтересів або ж реалізувати в інтернеті той чи інший журналістський проект, багато в чому залежить успішність його кар’єри.

Подані завдання до тринадцяти лабораторних та десяти самостійних робіт, містить теоретичні відомості з детальним описом матеріалу та контрольні запитання, відповіді на які сприятимуть систематизації та закріпленню набутих знань у майбутніх журналістів, щоб якнайкраще підготувати їх до роботи в онлайн-ЗМІ, з сайтами редакцій та власними блогами.

Трофименко О. Г. Веб-дизайн та HTML-програмування: навчально-методичний посібник / Трофименко О. Г., Козін О. Б. – Одеса, 2017. – 220 с.

© Трофименко О. Г.,
Козін О. Б., 2017

# ПЕРЕДМОВА

Дисципліна «Веб-дизайн та HTML-програмування» вивчається студентами університету за напрямом бакалаврської підготовки галузі знань 06 «Журналістика» за спеціальністю 061 «Журналістика» на третьому курсі навчання.

***Метою*** викладання навчальної дисципліни «Веб-дизайн та HTML-програмування» є формування у майбутніх журналістів теоретичних знань та практичних навиків з проектування, макетування та редагування веб-орієнтованих програмних продуктів, підготувати майбутніх журналістів до роботи в онлайн-ЗМІ, з сайтом редакцій та власними блогами.

***Предметом*** вивчення навчальної дисципліни є веб-технології та принципи веб-дизайну, а також методи їх використання при розробці сайтів різноманітного призначення.

***Основними завданнями*** вивчення дисципліни «Веб-дизайн та HTML-програмування» є формування інформаційної культури студентів, розкриття можливості використання інформаційних технологій для розв'язування прикладних задач в галузі журналістики. Адже уміння створювати якісні й цікаві веб-сайти наразі стає одною з найважливіших складових інформаційної культури журналіста, оскільки від того, як він зможе подати у світовому інформаційному просторі себе, коло своїх професійних та особистих інтересів або ж реалізувати в інтернеті той чи інший журналістський проект, багато в чому залежить успішність його кар'єри.

Оволодіння курсом «Веб-дизайн та HTML-програмування» передбачає читання лекцій, проведення лабораторних занять, виконання самостійної роботи та складання заліку. Особлива увага приділяється прищепленню студентам практичних навиків із застосування сучасних інформаційно-комунікаційних технологій у майбутній професійній діяльності.

Згідно з вимогами освітньо-професійної програми після вчення дисципліни «Веб-дизайн та HTML-програмування» студенти повинні:

– **використовувати** на практиці сучасні інформаційні та комунікаційні технології, які застосовуються у пресі, на телебаченні, в радіомовленні, інтернет-ЗМІ і мобільних медіа;

– **володіти** основними методами, способами та засобами діставання, зберігання та опрацювання інформації;

– **застосовувати** знання щодо особливості створення і редагування творчих матеріалів, призначених для різних видів засобів масової інформації;

– **здійснювати** друкарську підготовку інформаційного продукту за допомогою спеціального програмного забезпечення.

## Перелік тем лабораторних робіт

*Лабораторна робота № 1.* Знайомство з HTML.
*Лабораторна робота № 2.* Каскадні таблиці стилів CSS.
*Лабораторна робота № 3.* Створення веб-сторінки з CSS-форматуванням тексту.
*Лабораторна робота № 4.* Створення і форматування таблиць на HTML-сторінці.
*Лабораторна робота № 5.* Розміщення зображень на веб-сторінці.
*Лабораторна робота № 6.* Створення форм на веб-сторінці.
*Лабораторна робота № 7.* Адаптивний веб-дизайн.
*Лабораторна робота № 8.* Створення веб-сайту за допомогою сайт-конструктора.
*Лабораторна робота № 9.* Створення блогу засобами Wordpress.
*Лабораторна робота № 10.* Дизайн сайту засобами плагінів.
*Лабораторна робота № 11.* Аналіз роботи веб-сайту та засоби його оптимізації.
*Лабораторна робота № 12.* Веб-аналітика та оптимізація роботи сайту.
*Лабораторна робота № 13.* Підсумкове заняття з презентацією та захистом створених сайтів.

Кожна із запропонованих до виконання робіт має теоретичні відомості з докладним описом порядку виконання та контрольні запитання для закріплення відповідного тематичного матеріалу.

## Перелік тем самостійної роботи

*Самостійна робота № 1.* Форматування веб-сторінки засобами CSS.
*Самостійна робота № 2.* Розміщення сайту журналіста на сервер.
*Самостійна робота № 3.* Вбудовування таблиць з обтіканням текстом.
*Самостійна робота № 4.* Розміщення медіаконтенту на веб-сторінці.
*Самостійна робота № 5.* HTML-документи на основі фреймів.
*Самостійна робота № 6.* Завантаження веб-сторінок на сервер.
*Самостійна робота № 7.* Підготовка матеріалу для сайту.
*Самостійна робота № 8.* Замовлення безкоштовного хостингу.
*Самостійна робота № 9.* Наповнення блогу, створеного за допомогою WordPress.
*Самостійна робота № 10.* Оптимізація роботи сайту.

Правильність виконаних лабораторних та самостійних робіт перевіряє викладач.

# ТЕМА 1
# ЗАСОБИ HTML ТА CSS ДЛЯ РОЗРОБЛЕННЯ САЙТІВ

*Лабораторна робота № 1*
## ЗНАЙОМСТВО З HTML

***Мета роботи*:** ознайомлення з мовою розмітки HTML і створення найпростішої веб-сторінки, набуття навиків форматування тексту й використання кольорів у форматі HTML.

### Навчальні питання

1. Структура HTML-документа.
2. Створення веб-сторінки типової структури.
3. Форматування тексту.

### Завдання

**1. Створення шаблону веб-сторінки**

1.1. Створити на диску папку і назвати її своїм прізвищем, надалі саме там зберігатимуться файли Ваших веб-сторінок.

1.2. Запустити текстовий редактор «Блокнот» або спеціалізований HTML-редактор «Brackets» (безкоштовно скачати останню версію можна на офіційному сайті http://brackets.io/) чи то редактор «SubLime Text 2»[1]. Також можна скористатись онлайн-редактором HTML, CSS, JavaScript коду (https://codepen.io/pen/).

1.3. Набрати шаблон веб-сторінки:

```
<!DOCTYPE html>
<meta charset= "utf-8">
<html>
  <head>
    <title>Заголовок веб-сторінки</title>
  </head>
  <body>
    Текст сторінки
  </body >
</html>
```

1.4. Зберегти цей файл з ім’ям **Прізвище_лб1**, тип файлу – html.

1.5. Переглянути результат у браузері, двічі клацнувши на імені файлу або запустивши для цього наявний веб-браузер та виконавши команду *Файл / Відкрити*, після чого віднайти свій файл. У вікні браузера має з’явитися текст сторінки, а сама сторінка мати заголовок.

---

[1] **SubLime Text 2** – редактор загального призначення з підтримкою HTML, CSS, JavaScript, який є повноцінною платформою розроблення сайтів.

**2. Створення на веб-сторінці заголовків різних рівнів**

2.1. Відкрити Ваш текстовий файл у редакторі та створити декілька заголовків різного рівня для Вашого тексту. Наприклад: заголовок першого рівня – «Це моя перша веб-сторінка» (без лапок), заголовок другого рівня – «Мета цього сайту», заголовок третього рівня – «Лабораторна робота № 1» тощо.

Для заголовків тексту використати теги h1...h6[2]. Зберегти змінення у файлі.

2.2. Щоб браузер показав модифікований документ, слід перезавантажити його у браузері. Для цього клацнути кнопку *Оновити*.

2.3. Використовуючи атрибут align, вирівняти заголовки різних рівнів відповідно за лівим, правим краєм та по центру. Формат заголовків має бути таким:

```
<h1 align=left>Це моя перша веб-сторінка </h1>
<h2 align=right>Мета цього сайту</h2>
<h3 align=center>Лабораторна робота № 1</h3>
```

2.4. Переглянути, як будуть виглядати заголовки, якщо змінити їхні рівні (використати всі рівні від h1 до h6).

**3. Застосування горизонтальних ліній у тексті**

3.1. Додати після заголовка тег <hr>, наприклад:

```
<h1 align=center>Текст заголовка</h1>
<hr>
```

Тег <hr> інтерпретується й відображається браузером як горизонтальна лінія уздовж усього вікна браузера.

3.2. Переглянути результат у браузері.

3.3. Використовуючи додаткові атрибути size (товщина горизонтальної лінії), width (довжина горизонтальної лінії), align (вирівнювання лінії, цей атрибут використовується в поєднанні з атрибутом width) змінити зовнішній вигляд лінії.

3.3.1. Задати значення атрибута size у пікселях в такий спосіб:

```
<hr size=80>
```

3.3.2. Переглянути результат у браузері.

3.3.3. Змінити значення у пікселях на інше й переглянути результат.

3.3.4. Задати довільне значення атрибута width.

Наприклад, тег

```
<h1 align=center>Текст заголовка</h1><hr width=”25%”>
```

сформує лінію, довжина якої становитиме 25% від ширини вікна браузера, а тег

```
<h1 align=center>Текст заголовка</h1><hr width=”300”>
```

сформує лінію довжиною у 300 пікселів.

3.3.5. Задати значення атрибута align, який задає спосіб вирівнювання лінії й використовується в поєднанні з атрибутом width, наприклад, так:

```
<h1 align=center>Текст заголовка</h1><hr width=”25%” align=left>
```

[2] Існує шість рівнів заголовків, які позначаються h1...h6. Заголовок рівня 1 (h1) забезпечує найвищий, а рівень 6 (h6) – найнижчий рівень заголовків.

3.3.6. Змінити значення атрибута align у різних заголовках на right і center та переглянути результати.

**4. Використання тегів абзацу й розриву рядка**

Щоб відокремити один блок тексту від іншого, слід сформувати новий абзац або новий рядок за допомогою дескрипторів <p> або <br> відповідно. Дескриптор <p> використовується для вставлення символу розриву абзацу, а <br> – символу розриву рядка. Вставляючи символ розриву абзацу, Ви тим самим даєте браузеру команду завершити поточний абзац і вставити порожній рядок перед наступним абзацом. Вставляючи символ розриву рядка, Ви даєте браузеру команду завершити поточний рядок і перейти на наступний.

4.1. Як приклад використання цих дескрипторів замінити вміст тегу <body> на такий:

```
<body>
  <h1 align=center>Розрив рядка й абзацу</h1>
  <br><br><br>
  <p>Символ розриву абзацу дає команду браузеру вставити порожній рядок
     перед початком наступного абзацу. </p>
  <p>Символ розриву рядка<br>
     дає команду браузеру<br>
     перейти на наступний рядок.</p>
</body>
```

4.2. Застосувати різні значення атрибута align для різних абзаців тексту, переглядаючи результати у браузері. Наприклад:

```
<p align=center> Цей текст буде вирівняний по центру. </p>
```

За замовчуванням текст абзацу вирівнюватиметься за лівим краєм, але можна вирівняти його і за правим краєм, і по центру, і за шириною вікна браузера. Для цього слід скористатися атрибутом align з одним зі значень: right, left, center або justify (за шириною).

**5. Використання кольорів для змінення фону і тексту**

5.1. Додати на веб-сторінку кольоровий фон (підібрати доцільний колір) за допомогою параметра bgcolor для тегу body (див. п. 2.2 теоретичних відомостей).

5.2. За допомогою параметра background для тегу body задати на веб-сторінці графічне фонове зображення, використовуючи доречне зображення (файл типу *jpg*) або файл анімації (файл типу *gif*).

5.3. Змінити колір шрифту різних фрагментів тексту на ті кольори, які будуть добре пасувати заданому фону.

5.4. Заголовок тексту відокремити горизонтальною лінією товщиною 3 пікселі, довжиною 50%, вирівняною по центру. При цьому текст заголовка має бути напівжирним шрифтом (див. п. 3 теоретичних відомостей).

5.5. Переглянути результат у браузері.

5.6. Вивчити теоретичні відомості до цієї роботи та підготувати відповіді на контрольні запитання. За потреби скористатись додатковою літературою.

5.7. Файл веб-сторінки Прізвище_лб1.html надати на перевірку викладачеві.

## Контрольні запитання

1. У чому суть гіпертекстової форми подання WWW-документів?
2. Які особливості синтаксису мови гіпертекстових директив?
3. Що таке HTML?
4. Що таке гіпертекстовий документ?
5. Якою є структура HTML-документа?
6. Які групи директив HTML існують?
7. Що таке тег? Які теги призначені для задавання структури документа?
8. Яка структура та формат запису парних тегів HTML? Навести приклади.
9. Що таке параметр тегу? Який формат запису параметра тегу HTML? Навести приклади.
10. Перелічити теги для подання текстової інформації та надати їхній опис.
11. У чому схожість та відмінність призначення тегів P та BR?
12. Яке призначення мета-тегів? Навести приклади мета-тегів.
13. Як подаються гіперпосилання в HTML-документі? Навести приклад тегів створення внутрішніх і зовнішніх посилань.
14. Перелічити різновиди списків, існуючих в HTML. Навести теги для списків в HTML.
15. Як долучаються графічні об’єкти до HTML-документа? Перелічити параметри тегу графічного об’єкта.
16. Навести приклади вставлення файлу зображення на веб-сторінку із зазначенням різних параметрів зображення та з альтернативним текстом.

## Теоретичні відомості

### 1. Структура HTML-документа

### 1.1. Використовувана термінологія

**Веб-сайт** – сукупність інформаційних ресурсів, доступних через інтернет, які пов’язані між собою гіпертекстовими та іншими навігаційними посиланнями, простіше – це певне місце в інтернеті, де можна розмістити будь-яку інформацію, зробивши її доступною з будь-якої точки світу.

**Браузер** (англ. *browser*) – програма для переглядання веб-сторінок.

**HTML** (від англ. *HyperText Markup Language* – мова розмітки гіпертекстових документів) – стандартний засіб створення веб-сторінок. Документ HTML обробляється браузером та відтворюється на екрані у звичному для людини вигляді.

**Гіпертекст** – це особливий текст, в якому є посилання на іншу веб-сторінку. Клацання по гіперпосиланню призведе до того, що браузер запитає документ, на який вказує посилання, та завантажить його у вікно браузера.

Для створення HTML-сторінок потрібен текстовий редактор, наприклад «Блокнот», або спеціальний HTML-редактор, наприклад «SubLime Text 2» або «Brackets». Крім того, можна скористатись онлайн-редактором «Codepen.io».

Створюючи веб-сторінку, час від часу варто перевіряти, як вона виглядатиме у браузері. Для цього використовується будь-який браузер, проте слід зважати на те, що у різних браузерах вигляд одної і тої самої веб-сторінки може відрізнятися.

**Дескриптор** (тег) – основний елемент кодування, прийнятий у стандарті HTML. Теги визначають межі дії елементів і відокремлюють елементи один від одного. Теги записуються у кутових дужках **< >**.

Існує два типи HTML-тегів:

1) **контейнер** – дескрипторна пара, що складається з початкового (відкривного) і кінцевого (закривного) дескриптора. Контейнери призначені для зберігання деякої інформації, приміром, тексту або інших HTML-дескрипторів. Наприклад: <html> і </html>,  і  тощо.

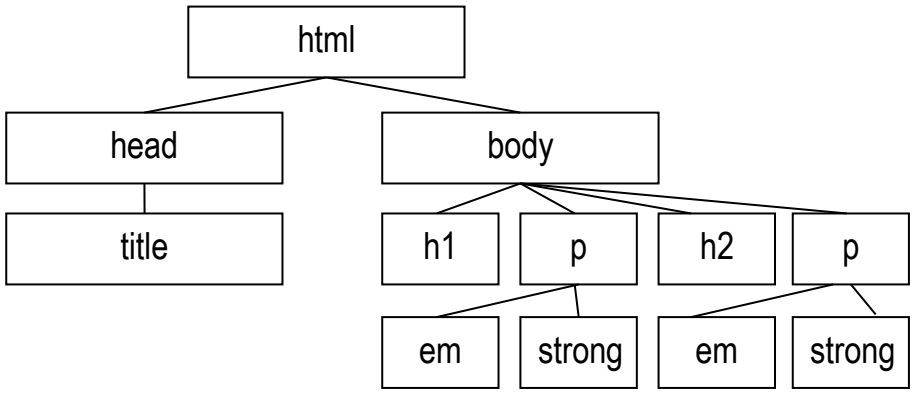
2) **порожній (одиночний) дескриптор** – складається тільки з початкового дескриптора й не містить ніякої інформації. Він виконує самостійне завдання, не пов’язане з конкретним текстом, наприклад: <br>, <hr> тощо.

### 1.2. Загальноприйнята структура HTML-документа

HTML-документ має визначену структуру, яку слід дотримуватись:

```
<!DOCTYPE html>
<meta charset="utf-8">
<html>
  <head>
    <title>Заголовок веб-сторінки </title>
  </head>
  <body>
    Текст на веб-сторінці
  </body >
</html>
```

Текст на веб-сторінці може складатися з різних за змістом та формою елементів: заголовків (теги від <h1> до <h6>), абзаців (тег ), таблиць (теги <table>, <tr>, <td> та ін.), вбудованих зображень (тег <img>) тощо. Для форматування кожного з цих елементів існують спеціальні теги, доречні саме для цього елемента.



### 1.3. Терміни наслідування

Для механізму наслідування використовують такі терміни:

- **предок** – елемент, який включає в себе інші (дочірні) елементи. Наприклад: html – предок для всіх, а head – лише для title;
- **потомок** – елемент, який розміщено всередині якогось елемента. Наприклад: title – потомок head, а head – потомок html;
- **батьківський елемент** – елемент, який стоїть вище на один рівень відносно того елемента, який зараз розглядається. Наприклад: html – батько для head і body, а body – для h1, h2, p;
- **дочірній елемент** – елемент, який стоїть на один рівень нижче елемента, який зараз розглядається. Наприклад: h1, h2, p є дочірніми тільки відносно body, але не до html, а body – дочірній до html;
- **сестринський елемент** – елемент, розміщений на одному рівні поруч з елементом, який розглядається. Наприклад: h1, h2, p – сестринські, також сестринськими є head і body, а ось head і title мають різних батьків і не є сестринськими.

## 2. Форматування тексту

### 2.1. Використання фізичних і логічних дескрипторів стилів

В HTML передбачено кілька способів форматування тексту. Це явне (або абсолютне) форматування за допомогою **фізичних стилів**; неявне (або відносне) форматування за допомогою **логічних стилів**; змінення розміру шрифту.

Текст, виділений фізичним стилем, у всіх браузерах відображається однаково. Логічні стилі у різних браузерах відображаються по-різному. Дескриптори фізичних стилів наведені у табл. 1.1.

*Таблиця 1.1* – **Дескриптори фізичних стилів**

| Дескриптор | Стиль |
|---|---|
| <b> | Напівжирний шрифт |
| <i> | Курсив |
| <tt> | Моноширинний шрифт |
| <u> | Підкреслення |
| <sub> | Підрядковий текст |
| <sup> | Надрядковий текст |
| <strike> | Перекреслення |

Дескриптори логічних стилів (табл. 1.2), як і дескриптори абзаців та заголовків, вказують на характер тексту, а не на точний спосіб його відображення у вікні браузера. При цьому браузер сам вирішує, як відформатувати текст оптимальним чином стосовно іншої частини веб-сторінки.

За допомогою елемента font можна визначити розмір і кольори шрифту. Атрибут size дозволяє вказати абсолютний розмір шрифту (він може набувати значення від 1 до 7) або відносний, стосовно розміру шрифту, використовуваного в основній частині сторінки (він може набувати значень від -4 до +4):

<font size=значення>

Для змінення кольорів символів використовується атрибут color, значення якого вказується назвою кольору або шістнадцятковим числом:

<font color=#rrggbb>текст</font>

або

<font color=red>текст</font>

*Таблиця 1.2* – **Дескриптори логічних стилів**

| Дескриптор | Стиль |
|---|---|
|  | Виділений текст |
|  | Сильно виділений текст |
| <cite> | Текст у вигляді цитати |
| <code> | Текст, що є фрагментом HTML-коду |
| <samp> | —""— |
| <dfn> | Текст, що є визначенням |
| <kbd> | Текст, що є назвою клавіші клавіатури |
| <var> | Текст, що визначає змінну або її значення |
| <acronym> | Абревіатура (акронім) та її розшифрування |

Запис #rrggbb складається з трьох двознакових шістнадцяткових чисел, які визначають співвідношення червоного, зеленого і блакитного кольорів, наприклад:

<font color=#a045ff>текст</font>

Для змінення розміру основного шрифту документа використовується елемент basefont:

<basefont size=значення>

Якщо треба розмістити на веб-сторінку інформацію, що вже міститься в деякому документі (але не в HTML-коді), то немає потреби витрачати час на повторний набір усього документа. Для цього випадку використовується контейнер <pre>. Цей контейнер дозволяє зберегти форматування тексту таким, яким воно було при набиранні, оскільки символи переходу на новий рядок він інтерпретує як розриви рядка. Пробіли усередині тексту інтерпретуються в точній відповідності до їхнього розташування у текстовому редакторі. Дескриптори абзаців і заголовків, що випадково потрапили до контейнера <pre>, не інтерпретуються. Але в контейнері <pre> можна використати дескриптори стиля й дескриптор <a> для створення прив'язок (посилань).

### 2.2. Використання кольорів для змінення тексту і фону сторінки

За допомогою атрибута bgcolor дескриптора <body> можна визначити кольори фону сторінки, вказавши або шістнадцяткове значення, або назву кольору. Цей атрибут має формат:

<body bgcolor=#rrggbb>
.....html – документ
</body>

Тут запис #rrggbb складається із трьох двознакових шістнадцяткових чисел, які визначають співвідношення червоного, зеленого й блакитного кольорів, наприклад:

<body bgcolor=#000000> – чорний колір;
<body bgcolor=#ffffff> – білий колір;
<body bgcolor=#ff0000> – червоний колір;
<body bgcolor=#702499> – фіолетовий колір.

Для створення графічного фону використовується атрибут background дескриптора <body>:

<body background="малюнок.gif">

Звісно зображення для фону має бути доречним і за тематикою, і за насиченістю, щоб у користувача не виникало бажання негайно покинути цю сторінку.

Після змінення кольорів фону документа або додавання фонового зображення можна змінити кольори об'єктів переднього плану, тобто тексту. Загальноприйнятим для більшості графічних браузерів є використання для тексту, за винятком гіпертекстових посилань, чорного кольору. Кольори основного тексту можна задавати за допомогою атрибута text дескриптора <body>:

<body text=#rrggbb>
.....HTML – документ
</body>

### 3. Застосування горизонтальних ліній у тексті

Для поділу тексту сторінки на частини можна скористатися горизонтальними лініями. За допомогою дескриптора <hr> можна створити лінію з тінню уздовж вікна браузера. Якщо користувач змінить розмір вікна, то відповідно змінитися й довжина даної лінії. Дескриптор <hr> є одинарним. При використанні горизонтальних ліній над і під ними автоматично додаються символи розриву абзацу. Додаткові атрибути (табл. 1.3) дозволяють змінити товщину лінії, її довжину й положення у вікні браузера.

*Таблиця 1.3* – **Опис деяких атрибутів**

| Атрибут | Опис |
|---|---|
| size | Визначає товщину горизонтальної лінії |
| width | Визначає довжину горизонтальної лінії |
| align | Визначає спосіб вирівнювання лінії |
| norhade | Визначає чорну лінію без ефекту гравірування |

Значення атрибутів size і width задають у пікселях або у пікселях та відсотках відповідно. Атрибут align може набувати значень: left, right та center.

Приклад:

<hr size=5 width="25%" align=center>

Такий запис задає горизонтальну лінію товщиною 5 пікселів, вирівняну по центру вікна завдовжки 25% від ширини вікна браузера.

*Лабораторна робота № 2*

# КАСКАДНІ ТАБЛИЦІ СТИЛІВ CSS

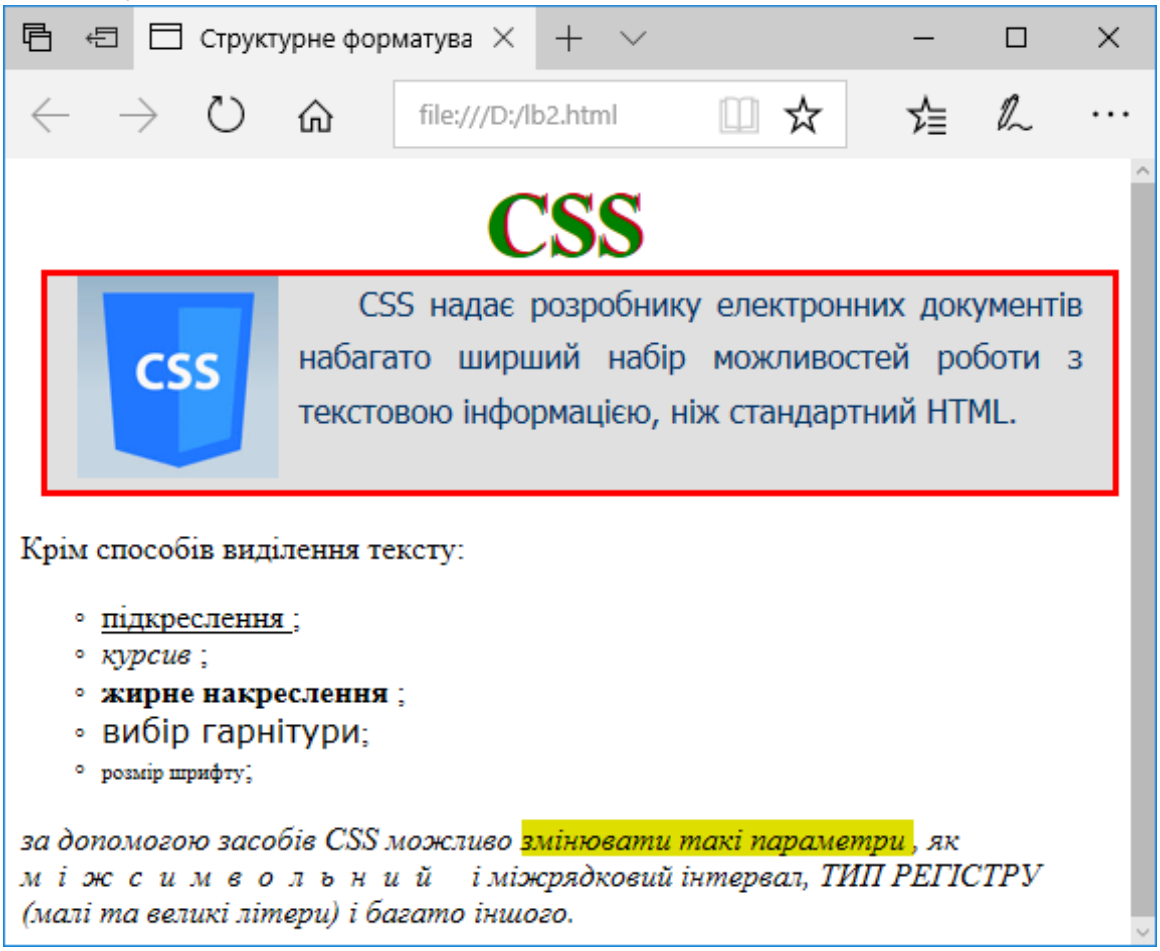
***Мета:*** ознайомлення зі специфікою та перевагами використання каскадних таблиць стилів CSS для змінення зовнішнього вигляду елементів веб-сторінок.

## Навчальні питання

1. Специфіка використання CSS.
2. Способи визначення таблиць стилів.
3. Структура CSS-файлів.

## Завдання

1. Вивчити теоретичні відомості до цієї роботи.
2. Використовуючи стилі CSS, створити веб-сторінку (файл **Прізвище_лб2.html**) з таким виглядом та вмістом:




Уточнення:

– тінь для слова «CSS» створити накладенням слів одне на одне із зсувом другого слова (значення відступів можуть бути як додатніми, так і від'ємними) та використанням змінених атрибутів кольору;

– у документі мають використовуватись тільки CSS-правила для форматування і не повинні використовуватися теги форматування з арсеналу класичного HTML (<b>, <i>, <u> тощо), а також атрибути тегу <font> (color, size тощо);
– слова «тип регістру» мають бути набрані малими літерами, а між літерами слова «міжсимвольний» не повинно бути пробілів.

3. На цій самій веб-сторінці (файл Прізвище_лб2.html) розмістити три зображення, використовуючи позиціонування на кшталт такого:

4. Створити файл Прізвище_лб2.css з визначенням стилів у селекторах body, p та img. При цьому для body задати гарнітуру шрифту Arial і відступ першого рядка (text-indent) 30 px. Визначення селекторів p та img задати самостійно.

У файлі Прізвище_лб2.html зробити посилання на цей CSS-файл.

5. Доповнити файл Прізвище_лб2.css описом псевдокласів гіперпосилань, які при наведенні вказівника миші на це гіперпосилання змінюватимуть його колір, розмір тексту, колір фону тощо.

6. Переглянути результат у браузері та надати його на перевірку викладачеві.

7. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1) Як розшифровується і що означає CSS?
2) Що таке стиль?
   а) метод перетворення текстових документів на HTML;
   б) набір правил форматування елементів веб-стрінки;
   в) спосіб зменшення HTML-коду веб-сторінки;
   г) мова розмітки гіпертекстових документів.
3) Що таке правило стиля? Описати різні компоненти і синтаксис.
4) Припустимо, що є набір правил стиля. Що треба зробити, щоб перетворити їх на зовнішню таблицю стилів?

5) У чому перевага об’єднання селекторів у групу?
6) Який атрибут використовується для визначення внутрішнього стиля?
   а) class; б) style; в) font; г) link.
7) Який варіант задавання кольору НЕ спрацює?
   а) color: #000; б) color: #aaa; в) color: #hhh; г) color: #aaaaaa.
8) Якщо межі сторінки не примикають до країв вікна браузера, куди слід додати властивість margin: 0?
   а) doctype; б) html; в) head; г) body.
9) Який CSS-код написано правильно?
   а)  {border: 1px solid #ccc;} ;
   б) div {border: 1px solid #ccc;} ;
   в)  {border: 1px solid #hhh;} ;
   г) div {border: 1px solid #hhh;}.
10) Коли імена ідентифікаторів і класів можна називати однаково?
   а) ніколи;
   б) якщо зустрічаються у коді лише раз;
   в) завжди;
   г) якщо використовуються для різних елементів.
11) Який CSS-код написано правильно?
   а) p { color: #ccc;} ;
   б) p: {border: 1px solid #ccc;} ;
   в) <p> { color: #ccc;} ;
   г) p { color: cccc;}.
12) Який CSS-код слід задати, щоб колір відвіданих і невідвіданих посилань був одним і тим самим?
   а) a:link, a:visited {color: yellow;} ;
   б) a:active, a:visited {color: yellow;} ;
   в) a:link {color: yellow;} ;
   г) a:link, a:active {color: yellow;}.
13) Як правильно вставляти коментарі в CSS-код?
   а) /* Мій коментар */; в) # Мій коментар;
   б) // Мій коментар; г) # Мій коментар #.
14) Яку помилку припущено у такому описі шрифту:
   font-family: Arial, Times New Roman, Helvetica, sans-serif;?
   а) замість font-family слід використовувати властивість font;
   б) слід поставити лапки для Times New Roman;
   в) не можна вказувати більше 3-х різних шрифтів;
   г) шрифту Helvetica не існує.
15) В якому з варіантів припущено явну помилку?
   а) p span { font-size: 150%;} ; в) p {font-size: 150%;} ;
   б) p span#text (font-size: 150%;} ; г) p text (font-size: 150%;}.

## Теоретичні відомості

### 1. Специфіка використання CSS

#### 1.1. Поняття CSS

Специфікація мови розмітки HTML дозволяє розробнику сайтів змінювати зовнішній вигляд елементів на веб-сторінках. Для цього складаються спеціальні правила відображення конкретного елемента в HTML-документі, які ще називають каскадними таблицями стилів (CSS) або стильовими шаблонами. Тобто HTML відповідає за зміст і структуру веб-сторінки (HTML-теги впорядковують інформацію, розбиваючи її на логічні елементи: заголовки (використовуються пошуковиками), абзаци, списки тощо), а CSS – за зовнішнє подання (форматування) цього матеріалу на веб-сторінці.

**CSS** (від англ. *Cascading Style Sheets*) – спеціальна мова, яка використовується для опису сторінок, написаних мовою розмітки даних. Можна сказати, що CSS – це спосіб додаткового форматування стандартних тегів HTML.

Розберемо складові слова поняття «каскадна таблиця стилів»:

– «**каскадна**» специфікація HTML дозволяє використовувати для одного й того самого елемента кілька стильових правил, інтерпретованих браузером послідовно, іншими словами – каскадом;
– «**таблиця**»: формат запису стильових правил CSS нагадує табличне подання даних. Заголовок таблиці відповідає назві елемента, класу або ідентифікатору стиля. В якості рядків і клітинок таблиці виступають стильові властивості та їхні значення, наприклад:

```
div {
  font-family: Tahoma;
  color: black;
  font-size: 12px;
}
```

– «**стиль**» – правило, яке описує форматування окремого фрагмента веб-сторінки. Таблиця стилів – набір певних стилів.

**Основні переваги CSS:**

– керування дизайном будь-якої кількості HTML-документів з однієї таблиці стилів;
– більш точний дизайн сторінок в усіх браузерах;
– поділ документа на дві складові: структуру (вміст веб-сторінки) і дизайн, завдяки чому вихідний код стає чистим і легко читається;
– розширені можливості порівняно зі звичайним HTML.

CSS нечутливий до регістру (великі та малі літери у ключових словах не розрізняються), переносу рядків, пробілів і символів табуляції.

#### 1.2. Рівні CSS

На сьогодні існують три рівні CSS, які визначаються наявністю завершеної редакції структури. Це: CSS1 (перший рівень структури стильових шаблонів,

остаточно затверджений 11 січня 1999 року), CSS2 (другий рівень стильових конструкцій, початок обговорення якого датується травнем 1998 року) і CSS3 (третій рівень стильового оформлення веб-сторінок, прийнятий до обговорення 23 травня 2001 року).

Саме третій рівень CSS3 позиціонується розробниками як єдина система подання стилів веб-сторінки, заснована на використанні спеціальних модулів.

CSS, як і будь-яка мова, має свій синтаксис. У ній немає ані елементів, ані параметрів, ані тегів. У ній є правила.

Правило складається з селектора і блоку оголошення стилів, укладеного у фігурні дужки:

```
h1 {color: red;  font-size: 14px;}
↑              ↑
селектор   блок оголошення стилів
```

*Селектор* визначає елемент для форматування, а *блок оголошень стилів* складається із властивостей та їх значень через крапку з комою:

```
h1 {color: red;       font-size: 14px;}
     ↑     ↑            ↑       ↑
властивість  значення  властивість  значення
```

У цьому прикладі задано червоний колір шрифту з розміром 14 пікселів для заголовка h1.

### 2. Способи визначення таблиць стилів

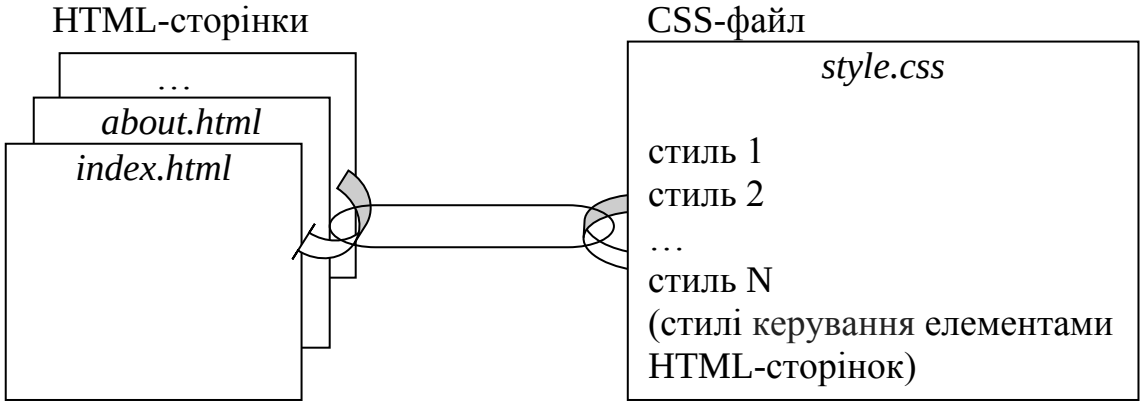
Як було сказано, будь-яка таблиця стилів CSS має інтерпретуватися браузером для того, щоб правила CSS, зазначені для конкретних елементів веб-сторінки (HTML-файлу), вступили в силу.

Визначити таблиці стилів (стильового шаблону) можна чотирма способами.

1) **Посилання на зовнішній файл (зовнішні стилі)**: усі стильові шаблони зі стилями форматування веб-сторінки виносяться в окремий текстовий файл з розширенням *css*, а в HTML-файлі в розділі head задається посилання на цей CSS-файл за допомогою тегу <link>:

```
<link   rel = "stylesheet"      type = "text/css"   href = "style.css">
          ↑                       ↑                   ↑
  Тип посилання              Тип даних файлу      Місце розташування файлу стилів
  «на таблицю стилів»
```

На один CSS-файл можна посилатися з декількох веб-сторінок сайту. Браузер, аналізуючи HTML-код, звернеться за вказаним шляхом і, виявивши вказаний файл стильового оформлення, відобразить елементи сторінки відповідно до певних правил CSS.




Суттєвим є те, що для підключення декількох файлів стилів, слід кожний файл прописувати окремим рядком (командою link). Крім того, один і той самий CSS-файл часто підключають до більшості веб-сторінок (HTML-файлів) сайту для дотримання одноманітного витриманого форматування. Самі CSS-файли рекомендовано розміщувати в окремій папці CSS.

2) **Внутрішні стилі**: стильові конструкції розташовують всередині HTML-сторінки у розділі head між тегами <style> і </style> з параметром type, який вказує, що підключається саме таблиця стилів CSS (взагалі-то існують й інші).

Наприклад:

```
<html>
  <head>
    <title>Підключення CSS до HTML</title>
    <meta http-equiv="Content-type"; content="text/html"; charset="windows-1251">
    <style type="text/css">
       h1 {
          color:red
        }
    </style>
  </head>
  <body>
     <h1>Заголовок червоного кольору 1</h1>
     <h1>Заголовок червоного кольору 2</h1>
     <h1>Заголовок червоного кольору 3</h1>
  </body>
</html>
```

Тепер всі заголовки h1 на сторінці будуть червоного кольору.

*Недолік* цього способу очевидний: таблицю стилів доведеться створювати для кожної сторінки. Це є однією з причин, через яку зручніше користуватися зовнішньою таблицею стилів (посиланням на зовнішній файл);

3) **Вбудовані стилі** (включення в тегові конструкції, inline-стиль): будь-який окремий HTML-елемент можна відформатувати засобами CSS. Для цього слід задати певне правило реалізації того чи іншого тегу, наприклад:

```
<p align = "justify" style = "color: #000000;  font-family: Verdana; ">
Текст параграфа ... </ p>
```

Тут задано окреме правило для конкретного параграфа. Суттєвим є те, що у цьому способі тег <style> є саме HTML-тегом, а не CSS, і тому після нього записують знак «=», а правило форматування записується у лапках (не у фігурних дужках, як у CSS).

Також можна створити стиль для одного конкретного заголовка h1:

```
<h1 style = "color: red"> Тема червоного кольору </h1>
```

Окремому HTML-елементу можна задати певний клас стильового шаблону:

```
<table>
   <tr>
      <td class="header" x/td>
      <td class="text" x/td>
   </tr>
</table>
```

Оголошення (опис) класів має дещо схожий вигляд з вбудовуванням стильових шаблонів у документ:

```
<style type="text/css">
   .header { font-weight: bold; color: gray; }
   .text { color: black; font-size: 11px; }
</style>
```

Клас header задаватиме жирний шрифт сірого кольору для тексту клітинок «шапки» таблиці, а клас text – звичайний чорний шрифт розміром 11 пікселів.

*Недолік* цього способу полягає в необхідності зазначення параметра style для кожного заголовка, а тому втрачається перевага використання CSS.

Якщо розробник захоче, щоб на сторінці всі заголовки рівня h1 були червоного кольору, а один з них став зеленого кольору, то простіше за все буде для цього одного заголовка задати внутрішній стиль:

```
</head>
   <body>
     <h1>Заголовок червоного кольору 1</h1>
     <h1 style="color:green">Заголовок зеленого кольору</h1>
     <h1>Заголовок червоного кольору 2</h1>
   </body>
</html>
```

Тут спрацює принцип каскадування, який усуне конфлікт, пов’язаний зі спробою одному й тому самому елементу задати два різні стилі. Оскільки зазначений всередині тегу стиль має вищий пріоритет, ніж загальний стиль, то лише один заголовок стане зеленим.

4) **Імпортування стильового шаблону CSS** по суті є подібним до посилання на зовнішній файл:

```
<style type="text/css">
   @import: url(style.css);
</style>
```

Іноді використання import уповільнює роботу, через що рекомендовано використовувати link, а не import. Використовувати import можна для підключення однієї таблиці стилів до іншої, з'єднуючи CSS-файли.

Усі чотири способи визначення стильового шаблону CSS можна використовувати одночасно в межах одного HTML-документа. Така можливість дозволяє задавати основне правило CSS, наприклад, у вигляді зовнішнього файлу шаблонів, а для виняткових або рідкісних HTML-елементів – окремі конструкції або у тегу <style>, або у кодових конструкціях самих тегів.

Щоб уникнути суперечливих ситуацій, пов'язаних зі спробою одному й тому самому елементу задати суперечливий стиль, слід розставити пріоритети. Порядок зростання пріоритетів такий:

1) посилання на зовнішній файл;
2) імпорт;
3) вбудовування в документ;
4) включення в тегові конструкції (inline-стиль).

Отже, inline-стиль має найбільший пріоритет.

Розглянемо на прикладі поєднання різних способів визначення стильового шаблону CSS:

```
<html>
  <head>
    <title>Поєднання різних способів визначення CSS</title>
    <link rel="stylesheet" type="text/css" href="style.css">
    <style type="text/css">
       p { text-align: justify; color: green; }
       .title { color: blue; font-weight: bold; font-size: 16px; }
    </style>
  </head>
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00"   vlink="blue">
    <font class="title"> Способи визначення шаблонів CSS</font>
. . . . . .
  </body>
</html>
```

### 3. Запис шаблону CSS

Розглянемо існуючі способи визначення стильових шаблонів CSS.

### 3.1. Групування та наслідування

Як зазначалося, будь-яке CSS-правило складається із селектора і визначення шаблону. Селектор виступає в ролі покажчика стильового правила для певного HTML-елемента або внутрішнього класу (ідентифікатора). Визначення шаблону – це опис стильових правил для певних HTML-елементів. Правила записуються через крапку з комою у фігурних дужках.

```
h3 {
    color: blue;
    font-family: Tahoma, Verdana, Arial;
}
```

У цьому прикладі селектором є елемент заголовка h3, для шаблону якого визначено шаблон з двох правил: колір – синій, гарнітура шрифту – Tahoma, Verdana або Arial. Це свідчить про те, що CSS дозволяє групувати кілька стильових правил для одного селектора в рамках єдиного опису шаблону.

Іншою особливістю CSS є властивість наслідування стильових правил для декількох селекторів (**групові селектори**) водночас, наприклад:

```
td, th, p, div {
    text-align: justify;
    color: gray;
    font-size: 10px;
}
```

Такий запис призначає єдиний стиль відображення текстової інформації для елементів таблиці (td, th), параграфів (p) і блоків (div), а саме: тип вирівнювання – за шириною, колір – сірий, розмір шрифту – 10 пікселів.

### 3.2. Види селекторів

Селекторами CSS можуть виступати:

**1) Селектори елементів (тегів) HTML** використовуються для визначення стилів конкретних елементів (тегів). В якості селектора виступає HTML-тег, наприклад:

```
body { color: orange; }
h2 { color: #0000FF; font-size: 18px; background-color: #0F0;}
```

Тут увесь текст у межах розділу body буде помаранчевим, а для заголовка, крім розміру шрифту, задано синій колір для шрифту і зелений – для фону. До речі, колір можна задавати константними назвами (red, blue, green, black тощо) або числовими значеннями палітри у формі від #000 (чорний) до #FFF (білий), або від #000000 (чорний) до #FFFFFF (білий). Числові значення дозволяють гнучко задавати відтінки тої чи іншої складової кольору: перша цифра у тризнаковому форматі (дві цифри у шестизнаковому) задає червону складову кольору від мінімального 0 (00) до максимального F[1] (FF[2]) значення, друга – зелену скла-

[1] F – значення 15 у шіснадцятковому форматі. Усі значення шіснадцяткового формату: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15).

дову, а третя – синю. До речі, потужним безкоштовним сервісом для добору кольорової палітри є https://color.adobe.com/create/color-wheel/ від компанії Adobe.

При додаванні нового елемента, наприклад таблиці, призначення стильового шаблону не спрацює для тексту всередині клітинок;

**2) Селектори класів** (**class**) дозволяють визначати стиль як для конкретного елемента, так і для будь-якого елемента, якому присвоєно даний клас. Ім'я класу починається з крапки і зазвичай пишеться малими літерами (допускається використання літер латиниці і цифр. При цьому наявність спеціальних символів, нижніх підкреслень та інших нестандартних елементів хоча і дозволено, але не рекомендується. Наприклад, визначення классу з ім'ям red-font у CSS-файлі матиме вигляд:

```
.red-font { color: red; }
```

Для застосування цього класу в HTML-файлі слід прописати його у будь-якому тегу:

```
<font class="red-font"> Текст червоного кольору </font>
```

або

```
<h1 class="red-font" </h1>
```

Якщо доповнити селектор класу назвою конкретного HTML-елемента, то дія стильового правила поширюватиметься лише на даний елемент:

```
h1.red-font { color: red; }
```

До речі, імена класів чутливі до регістру, тобто .red-font та .Red-font відрізняються.

При зазначенні класів стильового шаблону слід уважно стежити за тим, чи підтримує HTML-елемент тип, зазначений у визначенні. Наприклад, запис

```
hr { text-align: justify; }
```

буде безглуздим, оскільки горизонтальний розділювач стосується області структурного форматування і не може мати текст, для якого, відповідно до стильового правила, задано вирівнювання за шириною;

**3) Селектори ідентифікаторів** (**ID-селектори**) використовуються для форматування унікальних частин веб-сторінки (шапки, основної панелі навігації, додаткової (бокової) панелі навігації, зони контенту, «підвалу» тощо), позначених ідентифікатором ID. Ім'я ідентифікатора має бути унікальним у межах одного документа. Воно записується у відкривному тегу і може використовуватись як для форматування засобами CSS-стилів, так і для інших цілей, наприклад, для якірного посилання. ID-селектори записуються, починаючи з символу # (дієз) у назві:

```
#black { background-color: black; }
```

Наприклад, задавши цей ідентифікатор тегу, клітинки таблиці будуть залиті чорним кольором:

```
<td id="black">Клітинка чорного кольору</td>
```

Ще один приклад ID-селектора:

```
#header { background-color: #00F; }
```

[2] FF – значення 255 у шіснадцятковому форматі.

Цей ID-селектор задає синій колір фону. Використання його в HTML-файлі може мати вигляд:

```
<html>
  <head>
      . . . . .
  </head>
  <body>
      . . . . .
      <div id="header">
      . . . . .
      <div id="header">
      . . . . .
  </body>
</html>
```

**4) Універсальний селектор *** діє на всі або декілька тегів HTML-сторінки. Приклади:

```
 * { . . . . ; }                          // Стиль діятиме для усіх тегів веб-сторінки
 #header { background-color: black; }  // Цей стиль діятиме на всі теги всередині
                                         // ідентифікатора header
```

**Підіб'ємо підсумки:**

– якщо всі подібні елементи (наприклад, усі заголовки h1) повинні мати один стиль, то селектор складатиметься лише з цього елемента, наприклад:

```
h1 { color: black; }
```

– якщо один будь-який елемент (абзац, заголовок тощо) має відрізнятися від усіх інших, то йому присвоюється ідентифікатор (id), а роздільником у таблиці стилів є знак решітки (#), наприклад:

```
p # pink { color: pink; }
```

– якщо ж на сторінці буде кілька елементів з однаковим стилем, то їм присвоїться клас (class), а роздільником у таблиці стилів є крапка (.), наприклад:

```
p.pink { color: pink; }
```

– ідентифікатор має вищий пріоритет, аніж клас, тому, якщо для будь-якого елемента буде вказано і клас, і ідентифікатор (що суперечить принципам CSS), то застосовано буде стиль ідентифікатора.

### 3.3. Псевдокласи

**Псевдокласами** називають певні умови форматування HTML-елемента, відповідно до яких браузер підставляє стильові правила відображення даних. При цьому в початковій структурі веб-сторінки такі класи не присутні, вони створюються в процесі інтерпретації HTML-коду браузером. Здебільшого псевдокласи призначені для задавання типів форматування кільком різновидам елементів. Розглянемо функціональність псевдокласів на прикладі гіперпосилань.

Згідно зі специфікаціями HTML і CSS гіперпосилання може перебувати у чотирьох станах: невідвідуване посилання (link), відвідуване посилання (visited), активне посилання (active) і посилання при наведенні вказівника миші (hover). Перші три стани (link, visited, active) зазвичай прописуються у тегу <body> HTML-документа (рівень CSS1). Четвертий стан (hover) відноситься до рівня CSS2 і передбачає змінення кольору посилання при наведенні на нього вказівника миші.

Ці стани і будуть псевдокласами при записуванні правил відображення гіперпосилань у стильовому шаблоні:

```
a:link { color: blue; }
a:active { text-decoration: underline; }
a:visited { color: gray; }
a:hover { color: orange; }
```

Тут усі наявні на веб-сторінці гіперпосилання будуть відображатися відповідно до заданого стильового правила. Однак, доволі часто виникає потреба візуально виділити одні посилання щодо інших. Для цього поряд з псевдокласами використовуються звичайні селектори класів:

```
a:active.red { color: red; }
a:hover.red { color: blue; }
a:active.white { color: white; }
a:hover.white { color: black; }
```

## 4. Специфіка використання CSS

Перш за все, слід зазначити, що при визначенні стильових таблиць далеко не завжди властивості стандартного HTML-елемента відповідають опису шаблону стиля. Наприклад, в HTML для жирного накреслення використовується тег-контейнер <b> (або ), а в CSS – конструкція font-weight: bold для елемента або селектора. Для виділення тексту підкресленням в HTML передбачений тег <u>, а в CSS використовується запис text-decoration: underline тощо.

Зупинимося на деяких аспектах використання каскадних таблиць стилів, а саме на форматуванні тексту і структурному форматуванні.

### 4.1. CSS для форматування тексту

CSS надає розробнику веб-сторінок значно ширші можливості роботи з текстовою інформацією, аніж стандартний HTML. Крім способів виділення тексту (підкреслене, курсивне, жирне накреслення, вибір гарнітури і розмір шрифту), за допомогою засобів CSS можна змінювати міжсимвольний і міжрядковий інтервали, тип регістру (малі та великі літери) тощо.

### 4.2. Властивості форматування тексту та одиниці виміру

У CSS одиниці виміру властивостей елементів можна поділити на абсолютні і відносні (табл. 2.1).

*Таблиця 2.1* – **Одиниці виміру CSS**

| Абсолютні | Відносні |
|---|---|
| in (дюйм ~ 2,5 см) | em (висота шрифту елемента) |
| mm (міліметр) | ex (висота літери x) |
| cm (сантиметр) | px (піксель) |
| pt (пункт ~ 1/7 дюйма) | % (відсоткове співвідношення) |
| pc (піка = 12 пунктів) | |

Абсолютні одиниці виміру використовують, коли відомі характеристики того пристрою, який відображає інформацію.

Відносні одиниці виміру визначають масштаб форматованого елемента щодо інших елементів. Це дозволяє зберегти первозданність документа при виведенні на передавальний пристрій, характеристики якого заздалегідь не відомі.

«Em» – це масштабована одиниця, яка використовується у веб-документах. «Em» дорівнює поточному font-size і масштабується пропорційно. Наприклад, якщо font-size у документі 12pt, то 1em дорівнюватиме 12pt, 2em – 24pt, 0.5em – 6pt і т. д. Використання «em» стає все популярнішим у веб-документах через масштабованість, що суттєво при перегляданні сторінок на мобільних пристроях.

«Px» (pixel) – фіксований (немасштабаваний) розмір об'єктів, який використовується з метою створення піксель-ідеального (pixel-perfect) подання свого сайту у браузері на екрані комп'ютера. Один піксель дорівнює одній точці на екрані комп'ютера (найменший елемент роздільної здатності екрана. Одна з проблем використання «px» полягає в тому, що ці одиниці не дозволяють змінювати масштаб для слабозорих читачів та мобільних пристроїв.

«Pt» (pointspt) традиційно використовуються в друкованих ЗМІ (на папері). Один «pt» дорівнює 1/72 дюйма. «Pt», подібно до «px», має фіксований розмір і не може масштабуватися.

«%» (percents) – схожий на «em», за винятком кількох принципових відмінностей. По-перше, поточний font-size дорівнює 100% (тобто 12pt = 100%). По-друге, при використанні «%» текст стає повністю масштабованим для мобільних пристроїв і зручності користувача (accessibility).

Як правило, 1em = 12pt = 16px = 100%. Але, якщо font-size:14pt, то 2em становитиме 28pt.

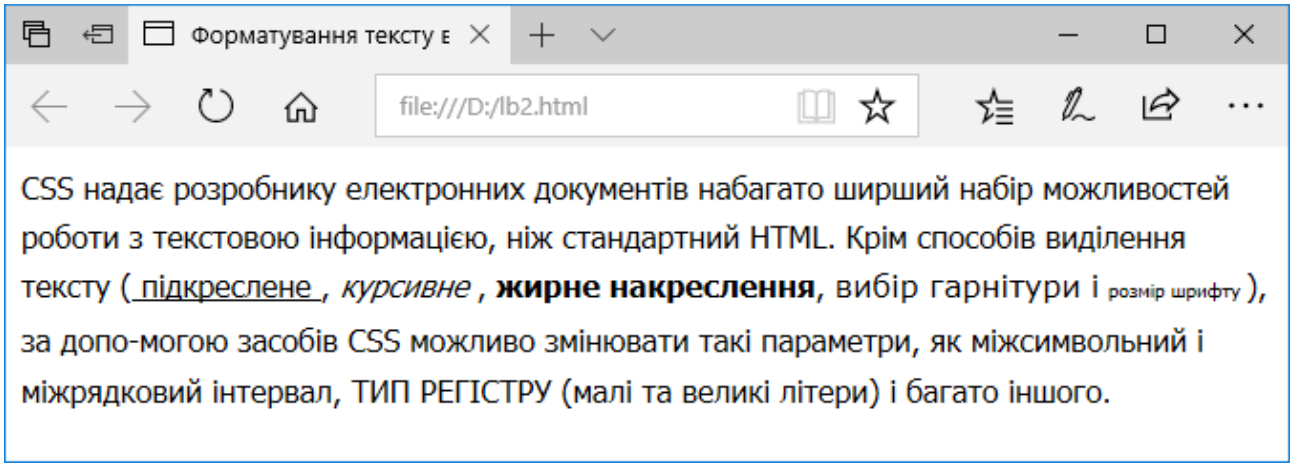
У табл. 2.2 наведено поширені властивості форматування тексту в CSS.

*Таблиця 2.2* – **Властивості форматування тексту в CSS**

| Властивість | Формат запису | Функція |
|---|---|---|
| font-family | font-family: Tahoma, Arial; | вибір гарнітури для відображення (допускається декілька назв через кому) |
| font-size | font-size: 11px; | розмір шрифту |
| font-style | font-style: italic; | вибір нахилу тексту (курсив) |
| font-weight | font-weight: bold; | наявність/відсутність жирного накреслення |
| font-variant | font-variant: small-caps; | модифікація усіх малих літер на заголовні зменшеного розміру |
| text-decoration | text-decoration: underline; | підкреслення тексту |
| text-align | text-align: right; | визначення типу вирівнювання тексту |
| text-transform | text-transform: uppercase; | усі великі літери |
| letter-spacing | letter-spacing: 1em; | міжсимвольний інтервал |
| line-height | line-height: 5mm; | міжрядковий інтервал |
| color | color: #ffffff | колір тексту |
| background-color | background-color: white; | колір фону тексту |
| text-indent | text-indent: 30; | відступ першого рядка абзацу |

Розглянемо на прикладі застосування цих властивостей:

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Форматування тексту в CSS</title>
    <style type="text/css">
      .text { font-family: Tahoma; color: 1003366;   /* визначення класу text */
              line-height: 7mm; font-size: 12pt;
             }
      #kursiv { font-style: italic; }                    /* визначення ID-селектора kursiv */
      span.font { font-size: 10px; }
      .color { background-color: #cecece; }
    </style>
  </head>
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
     <font class="text">                                 /* застосування класу text */
      CSS надає розробнику електронних документів набагато ширший набір можливос-
      тей роботи з текстовою інформацією, ніж стандартний HTML. Крім способів виді-
      лення тексту (<font style = "text-decoration: underline;"> підкреслене </font>, <font id =
      "kursiv"> курсивне </font>, <font style = "font-weight: bold;"> жирне накреслен-
      ня</font>, <font style = "font-family: verdana;"> вибір гарнітури </font> і <span class =
      "font"> розмір шрифту </span>), за допомогою засобів CSS можливо <font class =
      "color"> змінювати такі параметри</font>, як <font style = "letter-spacing: 3px;"> між-
      символьний </font> і міжрядковий інтервал, <font style = "text-transform: uppercase;">
      тип регістру </font> (малі та великі літери) і багато іншого.
    </font>
  </body>
</html>
```




### 4.3. Структурне форматування

Засоби CSS помітно розширили функціональність форматування структурних елементів веб-сторінки: p, div та ін. У табл. 2.3 наведено найпоширеніші властивості структурного форматування в CSS.

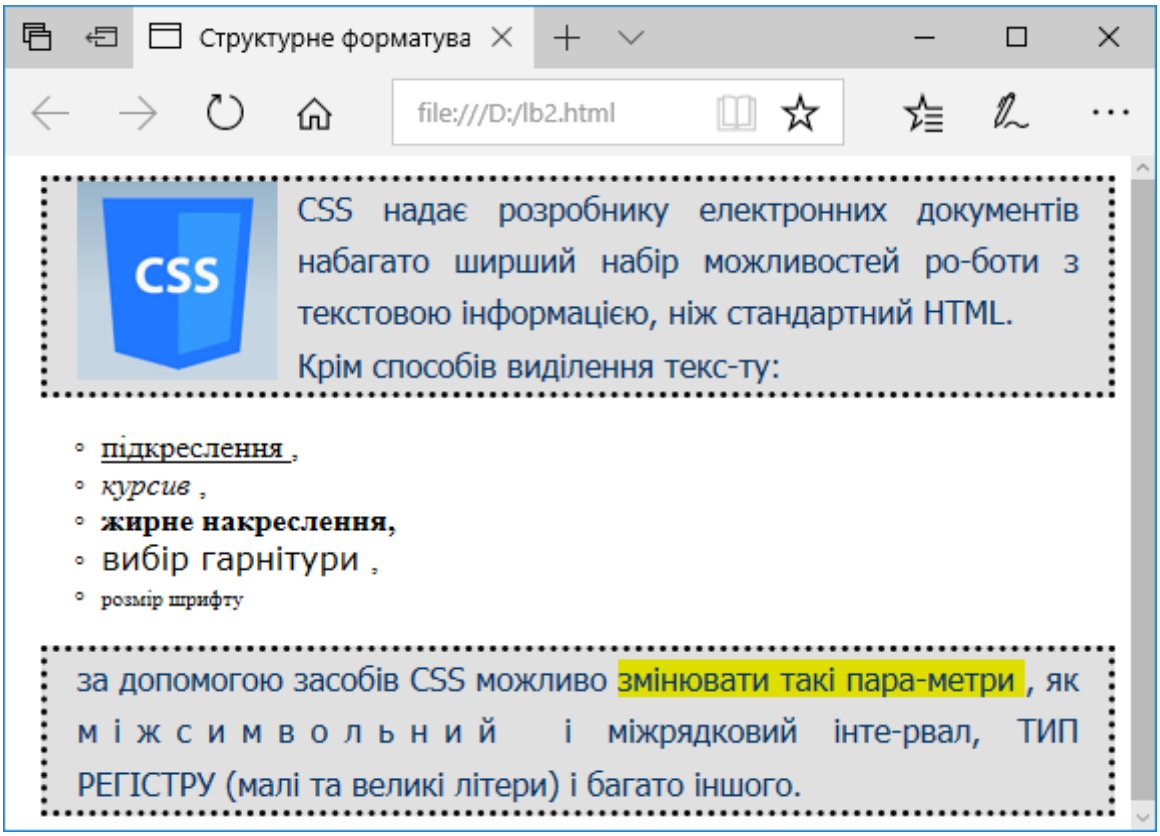
*Таблиця 2.3* – **Властивості структурного форматування в CSS**

| Властивість | Формат запису | Функція |
|---|---|---|
| border- width | border-width: 20px; | Ширина межі структурного елемента |
| border-style | border-style : solid; | Тип декоративного відображення межі елемента |
| border-color | border-color: gray; | Колір межі структурного елемента |
| list-style-type | list-style-type: square; | Тип нумерованого або маркірованого списку |
| list-style-image | list-style-image: url("bullet.gif"); | Зазначення шляху до графічного файлу |
| margin | margin: 1px 2px 3px 4px; | Визначення розміру поля відносно верхнього, правого, нижнього і лівого країв структурного елемента |
| padding | padding-top: 10em; padding-right: 25px; | Визначення відступу від верхнього, правого, нижнього і лівого країв структурного елемента |
| width | width: 300px; | Ширина структурного елемента |
| height | height: 120px; | Висота структурного елемента |
| background-color | background-color: #cecece; | Колір фону структурного елемента |
| float | float: left; | Плаваюче розміщення структурного елемента відносно інших елементів |

Розглянемо на прикладі властивості структурного форматування в CSS:

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Структурне форматування в CSS</title>
    <style type="text/css">
      .text { font-family: Tahoma; font-size: 12pt;
              color: #003366; line-height: 7mm;
              margin: 10px;
              padding-left: 15px; padding-right: 15px;
              border-color: black; border-style: dotted;
              background-color: #e0e0e0;
              width: 500px; text-align: justify;
            }
      #kursiv { font-style: italic; }
      span.font { font-size: 11px; }
      .color { background-color: #dede00; }
      img { float: left; width: 100px; height: 100px;  margin-right: 10px;}
      li { list-style-type: circle; }
    </style>
  </head>
```

```
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
    <p class="text">
        <img src="picture.jpg">
        CSS надає розробнику електронних документів набагато ширший набір можливостей
        роботи з текстовою інформацією, ніж стандартний HTML.<br>Крім способів виділення
        тексту:
    </p>
    <ul>
        <li><font style="text-decoration: underline;"> підкреслення </font>,
        <li><font id="kursiv"> курсив </font>,
        <li><font style="font-weight: bold;"> жирне накреслення, </font>
        <li><font style="font-family: verdana;"> вибір гарнітури </font>,
        <li><span class="font"> розмір шрифту </span>
     </ul>
     <p class="text"> за допомогою засобів CSS можливо <font class="color"> змінювати
        такі параметри </font>, як <font style="letter-spacing: 8;"> міжсимвольний </font> і
        міжрядковий інтервал, <font style="text-transform: uppercase;"> тип регістру </font>
        (малі та великі літери) і багато іншого.
     </p>
  </body>
</html>
```




З наведеного прикладу видно, що деякі елементи дозволяють застосовувати властивості структурного форматування CSS і для форматування тексту: text-align, background-color та ін. Отже, при форматуванні допускається використання властивостей CSS як для текстового, так і для структурного форматування.

## 5. Позиціонування об’єктів

Виходячи з концепції мови розмітки HTML, усі елементи документа виводяться браузером у тій послідовності, в якій розміщені тегові конструкції у коді. CSS дозволяє задавати порядок і послідовність відображення тих чи інших HTML-елементів залежно від певних подій на сторінці або маніпуляцій, здійснюваних з боку користувача. Інакше кажучи, за допомогою засобів CSS (спеціальних властивостей float і position) можна позиціонувати (задавати просторове розташування) об’єкти в межах веб-сторінки.

### 5.1. Позиціонування засобами float

Властивість **float** – доволі універсальний засіб позиціонування, який дозволяє позиціонувати елемент ліворуч або праворуч батьківського елемента (абзацу або клітинки таблиці), при цьому сам елемент прибирається зі звичайного потоку HTML-документа. Решта елементів на сторінці буду обтікати такий елемент.

Наприклад, абзаци обтікатимуть зображення, якщо для елемента <img> задана властивість float, а саме зображення (елемент) розташовуватиметься по краю батьківського елемента.

Якщо батьківського елемента немає, обтічний елемент розташовуватиметься по краю сторінки. Це призводе до того, що ширина цього елемента за замовчуванням стане шириною його вмісту. Іноді, наприклад, при створенні колонок для багаторазово використовуваного макета, така поведінка небажана. Це можна виправити шляхом додавання властивості width з фіксованим значенням для кожної колонки. Крім того, щоб обтічні елементи не стикалися один з одним, можна використовувати властивість margin для встановлення простору поміж елементами.

Найчастіше при позиціонуванні для float задають значення left і right:

```
img { float: left;}
```

Одночасне застосування float до декількох елементів дає змогу створити макет з елементами, розміщеними поряд або один напроти одного.

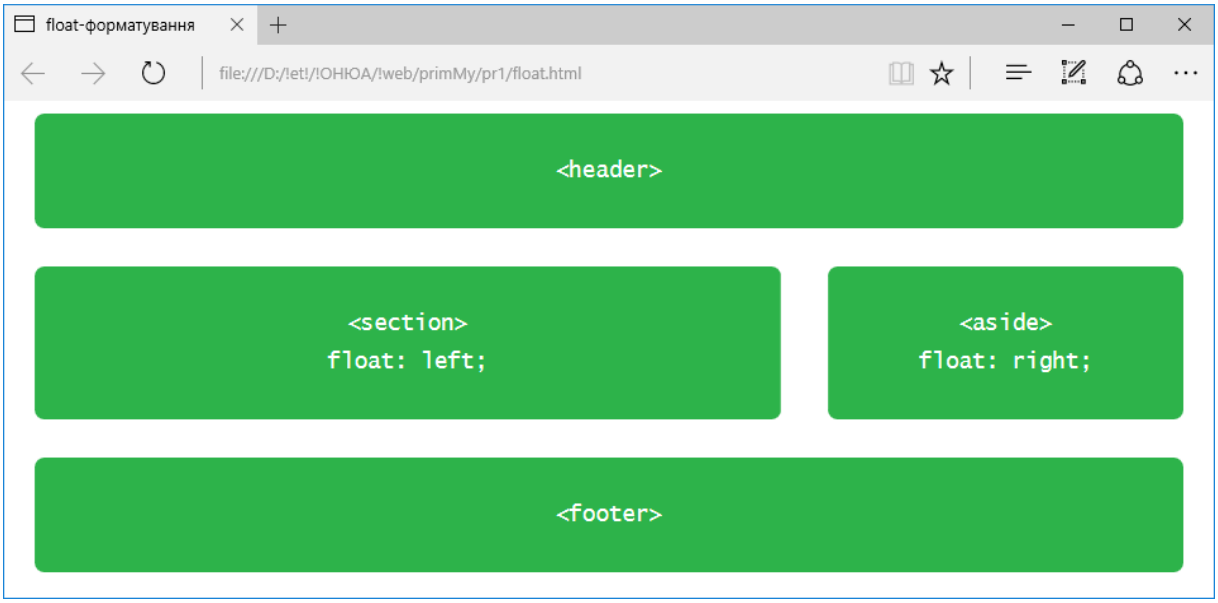
Так, для створення веб-сторінки з чотирьох основних розділів (блоків): шапки вгорі сторінки (<header>), двох колонок (<section>, <aside>) посередині і «підвалом» <footer> внизу слід у відповідному HTML-файлі всередині елемента <body> створити конструкцію на кшталт такої:

```
<header>...</header>
<section>...</section>
<aside>...</aside>
<footer>...</footer>
```

Щоби блокові елементи <section> та <aside> розмістити поруч «пліч о пліч», слід задати в CSS властивості float значення left для <section>, а для <aside> – значення right:

```
section {
  float: left;    margin: 0 1.5%;    width: 63%;
}
```

```
aside {
  float: right;   margin: 0 1.5%;   width: 30%;
}
```

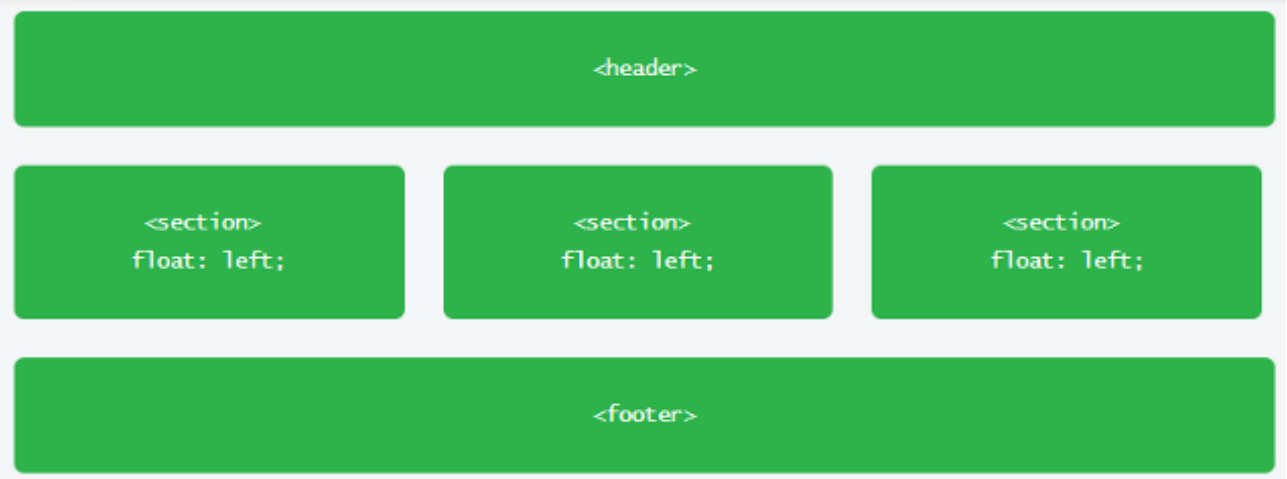



Для трьох (чи більшої кількості) колонок на веб-сторінці підхід дещо зміниться:

```
<header>...</header>
<section>...</section>
<section>...</section>
<section>...</section>
<footer>...</footer>
```

Для розміщення трьох елементів <section> у рядок з трьох колонок слід для всіх елементів <section> задати для властивості float значенн left і потрібну ширину з урахуванням зазорів.

```
section {
  float: left;   margin: 0 1.5%;
  width: 30%;
}
```




Як зазначалося, елемент при позиціонуванні через float прибирається зі звичайного потоку HTML-документа, це може призвести до того, що для батьківських елементів або елементів, розміщених поруч, не спрацюють задані стилі. Наприклад, можуть некоректно інтерпретуватися значення властивостей margin та padding, й елементи почнуть налізати один на одного. Для уникнення можливих некоректностей слід очистити float за допомогою властивості clear, яка набуває декілька різних значень: найчастіше використовуються значення – left, right та both.

```
div {
  clear: left;
}
```

Значення left очищує ліві float, у той час як значення right очищує праві float. Значення both очищує і ліві, і праві float і часто є найбільш ідеальним варіантом. Тому для сторінки з трьома однаковими колонками доречно задати очищення (властивість clear зі значенням both) для елемента <footer>. Важливо, щоб clear застосовувався до елемента, розміщеного після обтічних елементів, а не раніше, щоб повернути сторінку в її звичайний потік.

```
footer {
   clear: both;
}
```

Докладіше про це можна прочитати, скориставшись посиланням https://webref.ru/layout/learn-html-css/positioning-content.

### 5.2. Позиціонування засобами position

Властивість **position** визначає, як елемент позиціонуватиметься на сторінці і чи буде він відображатися у звичайному потоці документа. Вона застосовується у поєднанні з властивостями зміщення блоку – top, right, bottom і left, які точно визначають, де елемент буде розташований, шляхом переміщення елемента в різних напрямках. Можливими значеннями position є static, fixed, relative, absolute або inherit.

Якщо у батьківського елемента значення position задано як fixed, relative або absolute, то відлік координат ведеться від краю батьківського елемента. А коли у батьківського елемента значення position встановлено як static або батька немає, то відрахунок координат ведеться від краю вікна браузера.

За замовчуванням значення position задане як static. Це значення скасовує дію relative, absolute і fixed, а значення властивостей top, bottom, right, left ігнорує, тобто елемент існує в звичайному потоці документа без зміщень.

Існують два типи візуального позиціонування елементів: абсолютне і відносне.

### 5.3. Абсолютне позиціонування

Абсолютне позиціонування (**position: absolute**) чітко фіксує зазначений елемент на сторінці, незалежно від інших елементів документа, наприклад:

```
<img src = "picture.gif" width = "100" height = "100" alt = "Малюнок"
style = "position: absolute; top: 10px; left: 25px; ">
```

Тут графічне зображення абсолютно позиціоноване і розміщується в 10 пікселях від верхнього і в 25 пікселях від лівого краю свого батьківського елемента (за батьківський елемент виступає верхня ліва точка структури документа).

Розглянемо ще один приклад абсолютного позиціонування елемента div щодо батьківського елемента section з відносним позиціонуванням.

HTML-код:

```
<section>
  <div class=«offset»>...</div>
</section>
```

CSS-правила:

```
section {
  position: relative;
}
div {
  position: absolute;
  right: 20px;    top: 20px;
}
```

У результаті елемент  з'явиться посередині батьківського елемента <section> з відступами в 20 px справа і згори. При цьому  перекриватиме навколишні елементи.

### 5.4. Відносне позиціонування

Відносне позиціонування (**position: relative**) дозволяє розташувати зазначений об'єкт залежно від розміщення інших елементів документа (тобто відносно інших об'єктів сторінки), наприклад:

```
<div id = «text» style = «position: relative; top: 50px; left: 50px;»>
```

Значення relative дозволяє елементам відображатися у звичайному потоці сторінки, резервуючи місце для елемента, як це передбачалося, і не дозволяючи іншим елементам його обтікати. Однак, воно також дозволяє модифікувати розташування елемента за допомогою властивостей зміщення, наприклад:

HTML-код:

```
<div>...</div>
```

```
<div class=«offset»>...</div>
<div>...</div>
```

CSS-правила:

```
div {
  height: 100px;   width: 100px;
}
.offset {
  left: 20px;   position: relative;
  top: 20px;
}
```

Тут для другого елемента  з класом offset задано значення position як relative, а також дві властивості зсуву – left і top. Це зберігає вихідне розташування елемента, а іншим елементам заборонено займати цю область. Крім того, властивості зсуву переміщують елемент, виштовхуючи його на 20 пікселів від лівого і на 20 пікселів від верхнього початкового розташування. При такому зсуві елемент перекриватиме елемент під ним, а не зрушуватиме його вниз, як це роблять властивості margin або padding.

Ці особливості можна використати для створення тіні до заголовка:


HTML-код:

```
<h1>Одеса<div class="offset">Одеса</div></h1>
```

CSS-правила:

```
.offset  {  right: 2px;   position: relative;
            color: red;   font-style: bold;
            top: -48;
}
h1,  .offset {  font-size: 32pt;    text-align: center; }
```

### 5.5. Позиціонування через inline-block

Крім використання float та position, позиціонувати контент можна за допомогою властивості display в поєднанні зі значенням inline-block. Метод з inline-block, насамперед, корисний при компонуванні сторінок, а також для розміщення елементів в решенгу поруч один з одним.

Нагадаємо, що значення inline-block для властивості display відображає елементи в лінію і дозволяє їм набувати всіх властивостей блокової моделі, включаючи height, width, padding, border і margin. Застосування inline-block дозволяє повною мірою скористатися блоковою моделлю, не турбуючись про очищення будь-яких float.

Розглянемо наш приклад з трьома колонками у виконанні з inline-block:

```
<header> ... </ header>
```

```
<section> ... </ section>
<section> ... </ section>
<section> ... </ section>
<footer> ... </ footer>
```

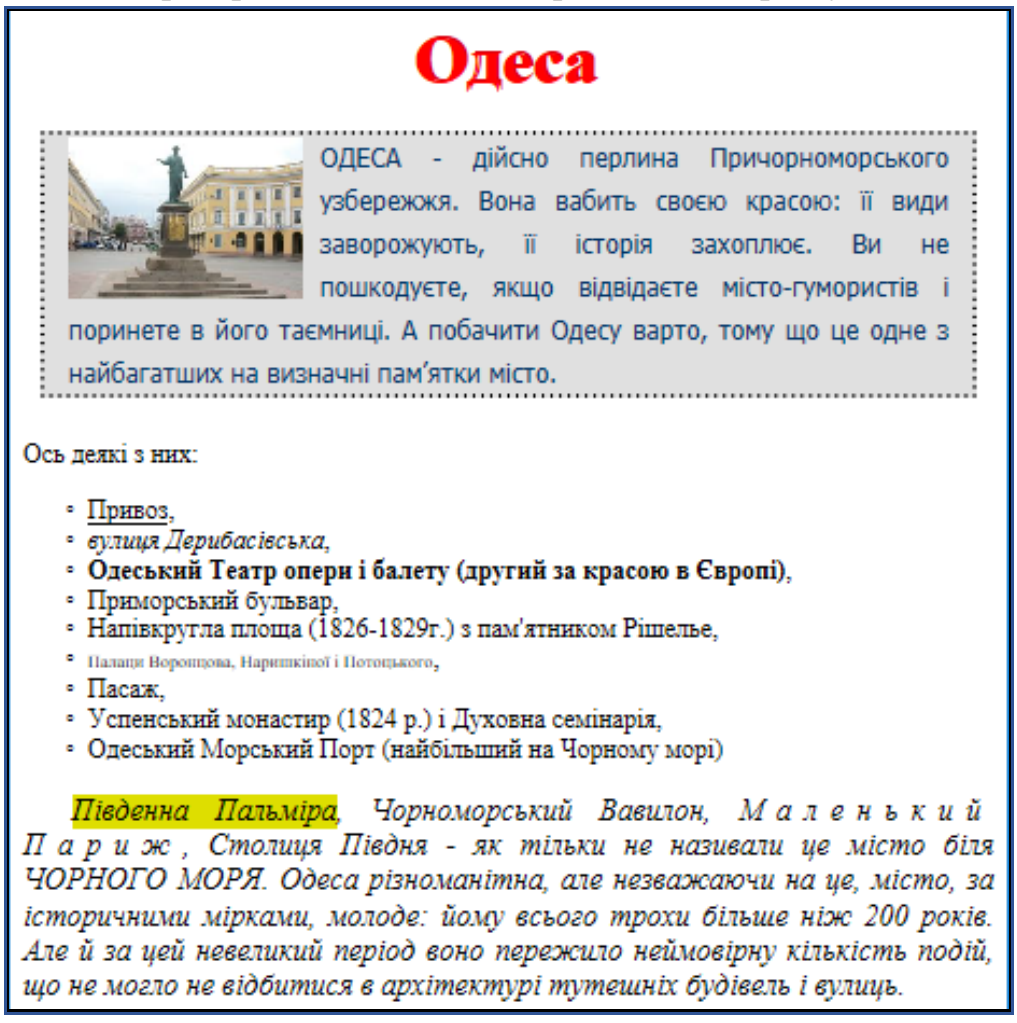
У CSS-правилах слід властивість float для трьох елементів <section> замінити на display: inline-block, залишаючи властивості margin і width тими, що були раніше. Як результат, наш CSS буде виглядати так:

```
section {
  display: inline-block;   margin: 0 1.5%;   width: 30%;
}
```

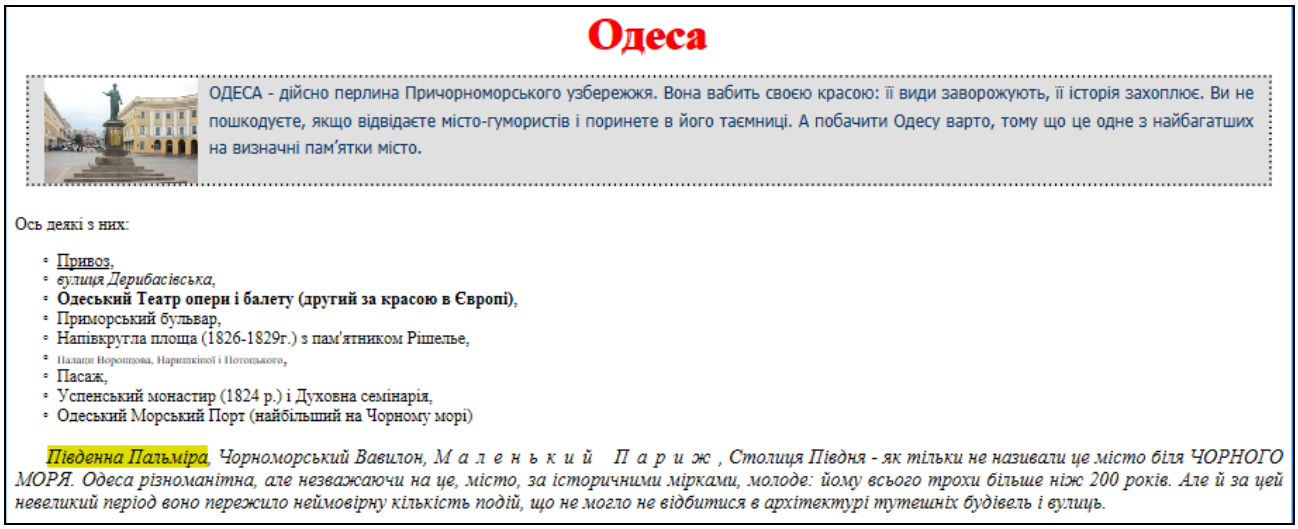
На жаль, при цьому останній елемент <section> виштовхнеться на новий рядок, оскільки загальна ширина стане занадто великою (розмір кожного окремого простору додається до ширини зі значенням горизонтального margin усіх елементів у рядку). Щоб відобразити всі елементи <section> в одному рядку, слід видалити порожній простір між кожним <section>.

## 6. Приклади

**Приклад 1**. В якості дещо схожого прикладу виконання першого пункту поставленого лабораторного завдання створимо веб-сторінку такого вигляду:



При зміненні (збільшенні) ширини вікна браузера зображення не вийде за рамки, відведені для першого абзацу, і весь вміст буде коректно відображатися на веб-сторінці:




```
<html>
  <head>
    <meta charset="utf-8">
    <title>Структурне форматування в CSS</title>
    <style type="text/css">
     .text-block { display: inline-block;
                   font-family: Tahoma;
                   font-size: 12pt;
                   margin: 10px;
                   color: #003366;
                   line-height: 7mm;
                   padding-left: 15px;
                   padding-right: 15px;
                   border-color: black;
                   border-style: dotted;
                   background-color: #e0e0e0;
                   text-align: justify;
                   }
     .text {
                   font-family: "Times New Roman";
                   text-indent: 30;                    /*Відступ першого рядка абзацу*/
                   font-size: 14pt;
                   font-style: italic;
                   text-align: justify;
                }
     #kursiv { font-style: italic; }
```

```
    span.font { font-size: 11px; }
    .color { background-color: #dede00; }
    .img-head { float: left;   height: 100;  margin-right: 10px;}
    li { list-style-type: circle; }
    .offset   {  right: 2px;
                 position: relative;
                 color: red; font-style: bold;
                 top: -48;
              }
    h1,.offset { font-size: 32pt;  text-align: center;
              }
    header { height: 5%;}
  </style>
</head>
<body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
 <header>
  <h1><font color="dark-grey"> Одеса</font><!-- --><div class="offset">Одеса</div></h1>
 </header>
  <p class="text-block"> <img src=«Odessa.jpg» class="img-head"> ОДЕСА – дійсно перлина Причорноморського узбережжя. Вона вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошкодуєте, якщо відвідаєте місто-гумористів і поринете в його таємниці. А побачити Одесу варто, тому що це одне з найбагатших на визначні пам’ятки місто.</p>
 <p>Ось деякі з них:</p>
 <ul>
  <li><font style="text-decoration: underline; ">Привоз</font>,</li>
  <li><font id="kursiv">вулиця Дерибасівська</font>,</li>
  <li><font style="font-weight: bold;">Одеський Театр опери і балету (другий за красою в Європі)  </font>,</li>
  <li>Приморський бульвар,</li>
  <li>Напівкругла площа (1826-1829г.) з пам’ятником Рішелье,</li>
  <li><span class="font">Палаци Воронцова, Наришкіної і Потоцького</span>,</li>
  <li><font style="font-family: verdana; ">Пасаж</font>,</li>
  <li>Успенський монастир (1824 р.) і Духовна семінарія,</li>
  <li>Одеський Морський Порт (найбільший на Чорному морі).</li>
 </ul>
 <p class="text">
  <font class="color">Південна Пальміра</font>, Чорноморський Вавилон, <font style="letter-spacing: 8;">Маленький Париж</font>, Столиця Півдня – як тільки не називали це місто біля <font style=" text-transform: uppercase;">Чорного моря</font>. Одеса різноманітна, але незважаючи на це, місто, за історичними мірками, молоде: йому всього трохи більше ніж 200 років. Але й за цей невеликий пе-
```

ріод воно пережило неймовірну кількість подій, що не могло не відбитися в архітектурі тутешніх будівель і вулиць.
</p>
</body>
</html>

**Приклад 2**. В якості дещо схожого прикладу виконання другого пункту поставленого завдання додамо на веб-сторінку чотири зображення та задамо позиціонування їх у такий спосіб:

Додамо в HTML-код такий фрагмент у розділ body:

```
<div>
   <img src="img1.jpg">
   <img src="img2.jpg" style = "position: relative; top: 50px; left: -50px;">
   <img src="img3.jpg" style = "position: relative; left: -100px;">
   <img src="img4.jpg" style = "position: relative; top: 50px; left: -150px;">
</div>
```

А до CSS-правил додамо:

```
div {
      text-align: center;
    }
img {
       display: inline-block;
       height: 200px;
       border: 3px solid blue; /* Параметри рамки */
       padding: 10px;          /* Поля навколо картинки */
    }
```

**Приклад 3**. В якості дещо схожого прикладу виконання четвертого пункту поставленого завдання додамо на веб-сторінку гіперпосилання на відкриття у новом вікні веб-сторнки, створеної на попередній лабораторній роботі. Доповнимо файл style.css описом псевдокласів гіперпосилань, які при наведенні вказівника миші на це гіперпосилання змінюватимуть його колір, розмір тексту, колір фону тощо.

Додамо в HTML-код такий фрагмент у розділ body:

```
<p><a href="lb1.html" target="_blank">Перейти на веб-сторінку лабораторної роботи №1 </a></p>
```

А до файлу CSS-правил додамо опис псевдокласів гіперпосилань:

```
a:link { color: blue; }
```

```
a:active { text-decoration: underline; }
a:visited { color: gray; }
a:hover { color: orange; }
a:active.red { color: red; }
a:hover.red { color: blue; }
a:active.white { color: white; }
a:hover.white { color: black; }
```

*Самостійна робота № 1*

# ФОРМАТУВАННЯ ВЕБ-СТОРІНКИ ЗАСОБАМИ CSS

## Завдання

1. Найти в інтернеті сайт з різноманітним за формою і кольором тексту та зображеннями. Узгодити вибір уподобаного сайту з викладачем. Зробити його скріншот і зберегти на комп'ютері з імям **Прізвище_зразок.jpg**. Використовуючи його (або якийсь його фрагмент) як зразок стильового шаблону, створити CSS-файл **Прізвище_sr1.css** з правилами подібного форматування.

2. Скопіювати HTML-файл лабораторної роботи № 1 **Прізвище_лб1.html**, перейменувати файл на **Прізвище_sr1.html** та видалити теги форматування (з арсеналу класичного HTML – <b>, <i>, <u> тощо, а також атрибути тегу <font> (color, size та інші)).

3. Зробити у HTML-файлі **Прізвище_sr1.html** посилання на CSS-файл **Прізвище_sr1.css**.

4. Файли створеної веб-сторінки надати на перевірку викладачеві.

*Лабораторна робота № 3*

# СТВОРЕННЯ ВЕБ-СТОРІНКИ З CSS-ФОРМАТУВАННЯМ ТЕКСТУ

***Мета:*** навчитись застосовувати засоби мови розмітки гіпертексту HTML та стилів відображення гіпертексту CSS для створення і форматування текстових веб-сторінок.

## Навчальні питання

1. Основні теги мови HTML.
2. Розроблення HTML-файлу зі стильовим CSS-форматуванням.
3. Розроблення CSS-файлу зі стилями форматування.
4. Підключення CSS-файлу до HTML-сторінки.

## Завдання

1. Вивчити теоретичні відомості до цієї роботи.

2. Засобами HTML та CSS створити веб-сторінку про місто, яке вказано в індивідуальному варіанті табл. 3.1, використовуючи:

- заголовки різних рівнів (не менше двох);
- основний текст із характерним описом міста (не менше 10 абзаців);
- спеціальні виокремлення шрифту у тексті для найважливіших об’єктів (напівжирне накреслення, курсив, колір тощо);
- маркірований список (перелік основних пам’ятних місць з можливими веб-посиланнями (переходами) на сторінки з описом цих місць);
- зображення (не менше трьох);
- гіперпосилання на сайти, з яких використовували матеріали;
- стильове форматування тексту та зображень здійснити в окремому файлі за допомогою засобів CSS;
- мета-теги з ключовими словами, використовувані у браузерах та пошукових системах;
- мета-теги з коротким описом сторінки, використовувані у пошукових системах;
- як заголовок у тег title, в якому міститься ключовий опис, вписати назву заданого міста та номер Вашого варіанта;
- для одного чи декількох абзаців тексту змінити колір і розмір шрифту, фону, рамки, відступів, ширини тощо, оформивши їх як примітки (див. приклад).

Для цього в редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) створити файл з ім’ям **lb3.html**, в якому використовувати теги html, head, title, body, щоб задати структуру документа, та теги p, br, ul, li, strong, i, b, em та інші – для основного тексту. Для заголовків використовувати теги h1 … h6, для яких задати доречне стильове оформлення (шрифт, колір тощо) та вирівняти їх по центру.

*Таблиця 3.1* – **Індивідуальні варіанти завдання**

| Варіант | Місто | Варіант | Місто |
|---|---|---|---|
| 1 | Париж | 9 | Краків |
| 2 | Барселона | 10 | Мадрид |
| 3 | Рим | 11 | Лісабон |
| 4 | Венеція | 12 | Мілан |
| 5 | Берлін | 13 | Таллінн |
| 6 | Мюнхен | 14 | Стокгольм |
| 7 | Варшава | 15 | Хельсінкі |
| 8 | Будапешт | 16 | Рига |

Для стильового форматування створити файл з ім'ям **lb3.css**, в якому задати потрібні стилі. Підключити до html-сторінки свій файл зі стилями за допомогою тегу:

```
<link rel="stylesheet" type="text/css" href="lb3.css">
```

3. Переглянути результат у браузері і надати його на перевірку викладачеві.
5. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Що таке HTML та CSS?
2. Що таке гіпертекстовий документ?
3. Що таке тег? Які теги призначені для задавання структури документа?
4. Яка структура тегу HTML та формат запису тегу HTML.
5. Навести структуру HTML-документа. Описати призначення відповідних тегів.
6. Що таке параметр тегу? Який формат запису параметрів тегів?
7. Перелічити параметри тегу.
8. Перелічити теги для подання текстової інформації та надати їхній опис.
9. Як подаються гіперпосилання в HTML-документі? Навести приклади тегів для внутрішніх і зовнішніх посилань.
10. Перелічити різновиди списків, існуючих в HTML. Навести приклади тегів для списків.
11. Що таке вкладені списки в HTML? Навести приклад для організації вкладеного списку в HTML.
12. Як вставити графічні об'єкти в HTML-документ? Назвати параметри тегу для графічного об'єкта.
13. Навести приклади тегів для вставлення файлу зображення на веб-сторінку із зазначенням різних параметрів зображення та з альтернативним текстом.
14. Навести приклади тегів для гіперпосилань на відкриття зображення із зазначенням різних параметрів зображення.
15. У чому схожість та відмінність призначення тегів <p> та <br>?
16. Яке призначення мета-тегів? Навести приклади мета-тегів.

## Теоретичні відомості

### Основні теги мови HTML

Розглянемо в алфавітному порядку основні теги HTML (навіть ті, з якими вже познайомились на минулих лабораторних роботах).

**1. Тег <a>**

Тег <a> призначений для створення гіперпосилань у тексті веб-сторінки. Оскільки гіперпосилання є одним з ключових елементів будь-якого веб-ресурсу, то важливість даного тегу важко переоцінити. Цей тег має такі атрибути:

– accesskey – призначення для гіперпосилання «гарячої» клавіші;
– href – визначення адреси, на яку веде гіперпосилання;
– name – назва областей веб-сторінки;
– tabindex – визначення порядку переходу за гіперпосиланням;
– target – визначення вікна для відображення об'єкта гіперпосилання.

**2. Тег <b>**

Парний тег <b> (подібно тегу ) задає тексту напівжирне накреслення: **текст**. Цей тег є тегом фізичної розмітки.

**3. Тег <basefont>**

За допомогою тегу <basefont> можна змінити зовнішній вигляд всього тексту. Цей тег має атрибут size, призначений для змінення розміру шрифту тексту.

**4. Теги <big> та <blockquote>**

Тег <big> дозволяє збільшити шрифт тексту порівняно з сусіднім текстом, а за допомогою тегу <blockquote> можна створити в документі блок цитати.

**5. Тег <br>**

Тег <br> створює новий рядок, тобто для переносить текст на наступний рядок без формування нового абзацу. Цей тег має один атрибут – clear, призначений для запобігання переносу слів тексту.

**6. Тег <body>**

Одним з ключових тегів мови HTML є структурний парний тег <body>. За допомогою цього тегу ідентифікується основний текст документа. Інакше кажучи, все, що треба помістити на веб-сторінку, треба розмістити між тегами <body> </body>. Цей тег має декілька атрибутів:

– alink, link і vlink – дозволяють задати колір гіперпосилань;
– background – задає графічного фону;
– bgcolor – змінює колір фону веб-сторінки;
– bgproperties – задає режим переміщення графічного фону при прокручуванні веб-сторінки;
– bottommargin – дозволяє редагувати висоту нижнього поля;
– leftmargin – дозволяє редагувати ширину лівого поля;
– marginheight – дозволяє редагувати висоту верхнього і нижнього полів;
– marginwidth – дозволяє редагувати ширину лівого і правого полів;
– rightmargin – дозволяє редагувати ширину правого поля;

– text – змінення кольору тексту;
– topmargin – дозволяє редагувати висоту верхнього поля.

**7. Тег <caption>**

За допомогою тегу <caption> можна задати для таблиці заголовок, а атрибут цього тегу align дозволяє задати вирівнювання заголовка таблиці.

**8. Тег **

Тег  призначений для центрування елементів веб-сторінки

**9. Тег <col>**

Тег <col> дозволяє створити неструктуровану групу стовпців таблиці. Він має такі атрибути:

– align – вирівнює дані у клітинках групи по горизонталі;
– bgcolor – задає колір фону клітинок групи;
– char – вибирає символ, який задає порядок вирівнювання даних у клітинках групи. Ця можливість доречна для вирівнювання колонки чисел по точці. Працює тільки, якщо значення атрибута align встановлено як char;
– span – задає кількість стовпців у групі;
– valign – вирівнює дані у клітинках групи по вертикалі.

**10. Тег <colgroup>**

За допомогою тегу <colgroup> створюється структурна група стовпців таблиці. Він має ті самі атрибути, що і тег <col>.

**11. Тег <dd>**

Тег <dd> дозволяє визначити елемент списку.

**12. Тег **

Блоковий тег  дозволяє сформувати на веб-сторінці розділи з різними стилями.

**13. Тег <font>**

Тег <font> змінює параметри шрифту тексту за допомогою таких атрибутів:

– color – змінює колір тексту;
– face – змінює шрифт тексту;
– size – змінюєся розмір шрифту.

**14. Тег <form>**

Тег <form> створює форми для пересилання даних від користувачів на сервер. Для введення даних форма може містити текстові поля, списки, прапорці, перемикачі, кнопки відправки даних тощо. Тег має атрибути, ось деякі з них:

– accept-charset – визначає тип кодування тексту даних;
– action – задає URL-адресу, куди пересилати дані форми;
– enctype – задає формат кодування даних форми до відправлення;
– method – вибирає спосіб (get або post) пересилання даних форми у мережі.

### 15. Тег <frame>

Тег <frame> визначає властивості фрейму (окремої прямокутної області, яка містить свій власний HTML-документ). Фрейми зручні для оформлення панелей навігації сайтом, нерухомих елементів інтерфейсу тощо. Цей тег має декілька атрибутів:

- bordercolor – задає колір ліній рамки фрейму;
- frameborder – дозволяє приховати рамку фрейму;
- marginheight – задає висоту верхнього і нижнього полів (відступів) фрейму;
- marginwidth – формує ліве і праве поля фрейму певної ширини;
- name – ім'я фрейму;
- noresize – запобігає зміненню розмірів фрейму;
- scrolling – задає відображення смуг прокручування ("yes", "no" або "auto" (за замовчуванням));
- src – визначає URL місцезнаходження джерела даних у фреймі.

HTML 5 не підтримує тег <frame>.

### 16. Тег <frameset>

Тег <frameset> замінює тег <body> і використовується для поділу веб-сторінки на фрейми. Він може використовуватись з такими атрибутами:

- border – дозволяє регулювати товщину ліній рамок фреймів;
- bordercolor – дозволяє змінювати колір ліній рамок фреймів;
- cols – призначений для формування стовпців рамок фреймів;
- frameborder – призначений для приховування рамок фреймів;
- framespacing – задає товщину ліній рамок фреймів;
- rows – призначений для створення рядків фреймів.

HTML 5 не підтримує тег <frameset>.

### 17. Теги заголовків

<h1>, <h2>, <h3>, <h4>, <h5>, <h6> – теги заголовків різних рівнів (відповідно від першого до шостого). Ці теги є парними. Теги заголовків можуть використовуватись з атрибутом align, який задає вирівнювання тексту заголовка.

### 18. Тег <head>

Парний тег <head> визначає структурний заголовок усієї веб-сторінки.

### 19. Тег <hr>

Тег <hr> призначений для створення горизонтальної лінії. Він може використовуватись з такими атрибутами:

- align – визначає ознаку вирівнювання лінії;
- noshade – дозволяє надати горизонтальній лінії об'ємний ефект;
- size – дозволяє задати товщину лінії;
- width – дозволяє задати ширину лінії.

### 20. Тег <html>

Структурний тег <html> призначений для ідентифікації веб-сторінки.

Програмний код будь-якої веб-сторінки має починатися тегом <html> і завершуватись тегом </html>.

**21. Тег <i>**

Тег <i> (він є парним) надає тексту курсивне накреслення: *текст*.

**22. Тег <iframe>**

Тег <iframe> дозволяє створити вбудований фрейм для відтворення різних незалежних сторінок-документів). Підтримується HTML5. Цей тег є блоковим елементом. Вигляд блоку визначається стилем, який можна винести у зовнішній css-файл, а для тегу визначити атрибути class або id з імям селектора.

**23. Тег <img>**

Тег <img> вставляє у документ графічний об'єкт (рисунок, фотографію, зображення). Цей тег може використовуватись з такими атрибутами:

- align – вирівнює текст відносно зображення (наприклад, щоб текст обтікав навколо зображення);
- alt (обов'язковий атрибут) – альтернативний текст за відсутності графічного об'єкта;
- border – дозволяє помістити графічний об'єкт у рамку;
- dynsrc – адреса *avi*-файлу для вбудовування відеооб'єкта;
- height, width – задають відповідно висоту і ширину графічного об'єкта;
- hspace, vspace – відступи (простір) навколо графічного об'єкта;
- ismap – призначений для створення карти посилань;
- lowsrc – визначає ім'я і місцезнаходження файлу зображення з низькою роздільною здатністю;
- name – задає ім'я графічному об'єкту;
- src (обов'язковий атрибут) – адреса файлу зображення;
- tabindex – задає порядок переходу по графічних об'єктах;
- usemap – задає ім'я карти посилань.

**24. Тег <li>**

Тег <li> визначає елемент списку. Використовується в обох типах списків – і нумерованих (<ol>), і маркірованих (<ul>).

**25. Тег <link>**

Тег <link> створює посилання на *css*-файл таблиці стилів. Має атрибути:

- href – URL-адреса файлу зовнішньої таблиці стилів;
- rel – визначає тип зв'язку.

**26. Тег <marquee>**

Цей тег дозволяє створити на веб-сторінці один з доволі поширених ефектів – рухомий рядок. Він може використовуватись з такими атрибутами:

- behavior – тип руху рядка[5];

[5] Атрибут behavior. [Электронный ресурс]. – Режим доступа: http://htmlbook.ru/html/marquee/behavior.

– bgcolor – колір фону рухомого рядка;
– direction – напрям руху рядка;
– height, width – висота і ширина рухомого рядка;
– hspace, vspace – поля навколо рядка;
– loop – кількість циклів переміщення рухомого рядка;
– scrollamount – крок одночасного переміщення руху рухомого рядка;
– scrolldelay – затримка між окремими тактами переміщення тексту рядка;
– truespeed – мінімальне значення інтервалу затримки scrolldelay.

**27. Тег <meta>**

За допомогою тегу <meta> у програмному коді виділяється службова інформація про веб-сторінку, яка призначена головним чином для пошукових систем.

**28. Тег <nobr>**

Дія тегу <nobr> протилежна дії тегу <br>, він забороняє перехід тексту на новий рядок (від англ. *no break* – немає розриву).

**29. Тег <noframes>**

Тег <noframes> включає відображення альтернативного тексту при неможливості показу фрейму.

**30. Тег <noscript>**

За допомогою тегу <noscript> включається відображення альтернативного тексту при неможливості виконання сценарію JavaScript.

**31. Тег <ol>**

Нумерований список. Кожний елемент списку має починатися тегом <li>.

**32. Тег <p>**

Новий абзац. Тег використовується з атрибутом align, який задає вирівнювання тексту абзацу.

**33. Тег <script>**

Тег <script> призначений для створення об’єкта сценарію JavaScript.

**34. Тег <small>**

Тег <small> зменшує розмір шрифту на одну умовну одиницю[6] порівняно з основним текстом.

**35. Тег <strike>**

Тег <strike> задає ознаку закресленого тексту: ~~текст~~.

**36. Тег **

Подібно до тегу  задає напівжирне накреслення тексту: **текст**. Перевага надається саме тегу  замість <b>. Відмінності цих тегів в тому, що

[6] В HTML розмір шрифту вимірюється в умовних одиницях від 1 до 7. Середній розмір тексту, використовуваного за замовчуванням, – 3.

 використовується для логічної розмітки. Саме тому пошукові системи вміст цього тегу інтерпретують як виділений текст, що позитивно позначається на релевантності індексованих сторінок.

### 37. Тег <style>

Тег <style> створює внутрішню (прямо в HTML-файлі) таблицю стилів.

### 38. Тег <sub>

Тег <sub> перетворює текст на нижній індекс.

### 39. Тег <sup>

Тег <sup> перетворює текст на верхній індекс.

### 40. Тег <table>

Тег <table> призначений для створення таблиць. Він має такі атрибути:
- align – задає обтікання таблиці текстом;
- background – визначає графічний фон таблиці;
- bgcolor – задає колір фону таблиці;
- border – дозволяє задати потрібну товщину ліній рамки;
- bordercolor – задає колір ліній рамки;
- cellpadding – задає відступи між вмістом клітинки та її межами;
- cellspacing – дозволяє задати відступи між клітинками таблиці;
- frame – дозволяє задати набір ліній рамки таблиці;
- height, width – задають висоту і ширину таблиці;
- rules – задає набір внутрішніх ліній таблиці.

### 41. Тег <tbody>

За допомогою тегу <tbody> ідентифікується група рядків таблиці. Тег може використовуватись з такими атрибутами.
- align – спосіб вирівнювання даних по горизонталі;
- bgcolor – колір фону;
- valign – спосіб вирівнювання даних по вертикалі.

### 42. Теги <td> і <th>

Теги <td> і <th> дозволяють створити клітинку даних і клітинку заголовка таблиці відповідно. Вони можуть використовуватись з атрибутами:
- align – спосіб вирівнювання даних по горизонталі;
- background – графічний фон клітинок;
- bgcolor – колір фону клітинок;
- colspan – об’єднує сусідні клітинки одного рядка таблиці;
- height – висота клітинки;
- width – ширина клітинки;
- nowrap – заборона переносу слів всередині клітинки на новий рядок;
- rowspan – об’єднання сусідніх клітинок одного стовпця;
- valign – спосіб вирівнювання даних по вертикалі.

### 43. Тег <tfoot>

Тег <tfoot> створює групу рядків підсумкових даних таблиці – «підвал» таблиці. Він може використовуватись з такими самими атрибутами, як у <tbody>.

### 44. Тег <title>

Тег <title> визначає назву сторінки в браузері і показує її в результатах видачі пошукових систем. Цей тег є обов'язковим для будь-якого HTML-документа і розташовується всередині тегу <html>. Він містить ім'я документа, яке буде відображатися в заголовку вікна браузера зліва вгорі в назві закладки. Якщо тег пропущено, то при валідації документа буде видано повідомлення про помилку.

### 45. Тег <tr>

Тег <tr> призначений для створення рядка таблиці. Він може використовуватись з такими самими атрибутами, як у тегу <tbody>.

### 46. Тег <tt>

Тег <tt> дозволяє відобразити текст моноширинним шрифтом (телетайп).

### 47. Тег <u>

Тег <u> задає підкреслений текст: текст.

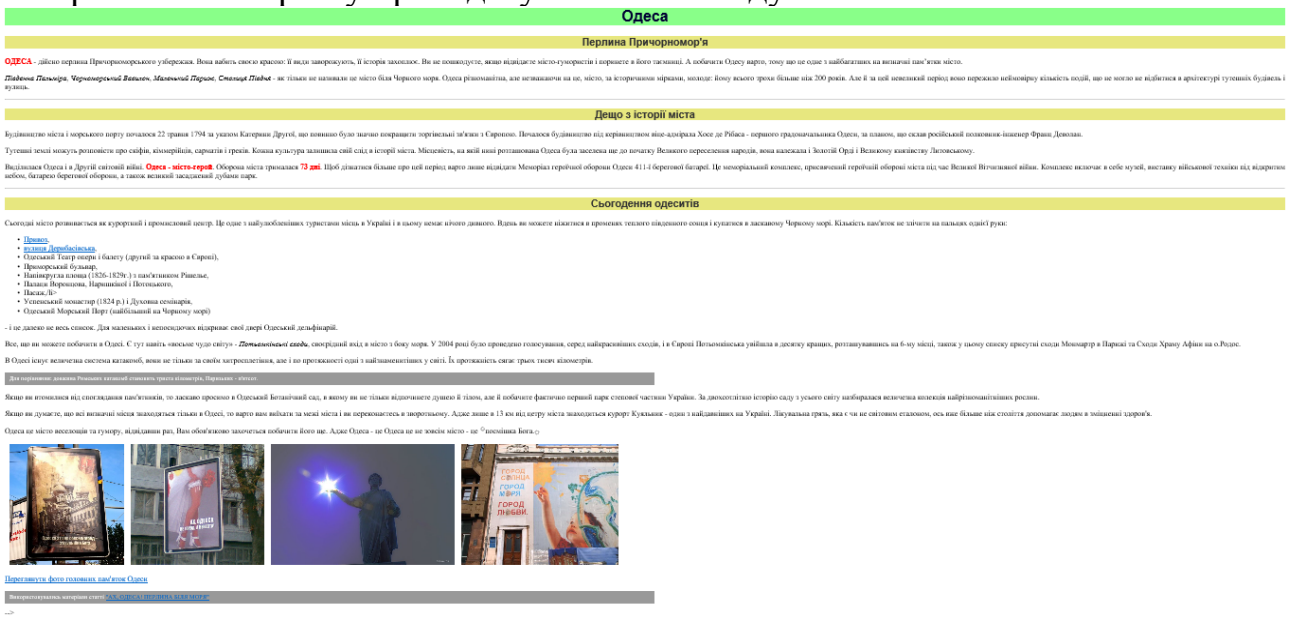
### 48. Тег <ul>

Тег <ul> дозволяє створювати маркіровані списки. Використання з цим тегом атрибута type дозволяє задати стиль маркірованого списку: disk – кружок, circle – коло, square – квадрат.

На практиці розглянутих тегів з атрибутами цілком достатньо для створення типових веб-сторінок мовою HTML.

### Приклад

Як приклад виконання завдання, поставленого у цій лабораторній роботі, створимо веб-сторінку про Одесу такого вигляду:



За браком місця наведемо детальніше вміст цієї веб-сторінки на рис. 3.1.

# Одеса

## Перлина Причорномор'я

**ОДЕСА** – дійсно перлина Причорноморського узбережжя. Вона вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошкодуєте, якщо відвідаєте місто-гумористів і поринете в його таємниці. А побачити Одесу варто, тому що це одне з найбагатших на визначні пам'ятки місто.

*Південна Пальміра, Чорноморський Вавилон, Маленький Париж, Столиця Півдня* – як тільки не називали це місто біля Чорного моря. Одеса різноманітна, але незважаючи на це, місто, за історичними мірками, молоде: йому всього трохи більше ніж 220 років. Але й за цей невеликий період воно пережило неймовірну кількість подій, що не могло не відбитися в архітектурі тутешніх будівель і вулиць.

### Дещо з історії міста

Будівництво міста і морського порту почалося 22 травня 1794 року за указом Катерини Другої, що повинно було значно покращити торгівельні зв'язки з Європою. Почалося будівництво під керівництвом віце-адмірала Хосе де Рибаса – першого градоначальника Одеси, за планом, що склав полковник-інженер Франц Деволан.

Тутешні землі можуть розповісти про скіфів, кіммерійців, сарматів і греків. Кожна культура залишила свій слід в історії міста. Місцевість, на якій нині розташована Одеса, була заселена ще до початку Великого переселення народів, вона належала і Золотій Орді, і Великому князівству Литовському.

Відзначилася Одеса і в Другій світовій війні. **Одеса – місто-герой**. Оборона міста тривалася **73 дні**. Щоб дізнатися більше про цей період варто відвідати Меморіал героїчної оборони Одеси 411-ї берегової батареї. Цей меморіальний комплекс, присвячений героїчній обороні міста під час Великої Вітчизняної війни. Комплекс включає в себе музей, виставку військової техніки під відкритим небом, батарею берегової оборони, а також великий засаджений дубами парк.

### Сьогодення одеситів

Сьогодні місто розвивається як курортний і промисловий центр. Це одне з найулюбленіших туристами місць в Україні, і в цьому немає нічого дивного. Тут Ви можете ніжитися в променях теплого південного сонця і купатися в ласкавому Чорному морі. Кількість пам'яток не злічити на пальцях однієї руки:

- Привоз,
- вулиця Дерибасівська,
- Одеський театр опери і балету (другий за красою в Європі),
- Приморський бульвар,
- Напівкругла площа (1826–1829) з пам'ятником Рішелье,
- палаци Воронцова, Наришкіної і Потоцького,
- Пасаж,
- Успенський монастир (1824) і Духовна семінарія,
- Одеський морський Ппорт (найбільший на Чорному морі)

- і це далеко не весь список. Для маленьких і непосидючих відкриває свої двері Одеський дельфінарій.

Все, що Ви можете побачити в Одесі. Є тут навіть «восьме чудо світу» – *Потьомкінські сходи*, своєрідний вхід у місто з боку моря. У 2004 році було проведено голосування, серед найкрасивіших сходів, і в Європі Потьомкінські увійшли до десятки кращих, посівши 6-те місце, також у цьому списку присутні сходи Монмартр у Парижі та Сходи Храму Афіни на о. Родос.

В Одесі існує величезна система катакомб, вони не тільки за своїм хитросплетінням, але й за протяжністю (≈ 3 000 км) – одні з найбільших у світі .

Для порівняння: довжина Римських катакомб становить 300 км, Паризьких – 500.

Якщо Ви втомилися від споглядання пам'ятників, то ласкаво запрошуємо до Одеського Ботанічного саду, в якому Ви не лише відпочините душею й тілом, а й побачите фактично перший парк степової частини України. За двохсотлітню історію в ньому з усього світу назбиралася величезна колекція найрізноманітніших рослин.

Одеса – це місто веселощів та гумору. Відвідавши його лише раз, Вам обов'язково захочеться побачити його ще. Адже Одеса – це не зовсім місто – це посмішка Бога.



Використовувались матеріали статті «АХ, ОДЕСА! ПЕРЛИНА БІЛЯ МОРЯ»

Рисунок 3.1 – Детальний вигляд вмісту створеної веб-сторінки

У редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) треба створити файл з ім'ям **lb3.html**, в якому записати таке:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Лабораторна робота № 3 Варіант 17</title>
    <link rel="stylesheet" type="text/css" href="lb3.css">
    <meta name="keywords" content="Одеса, Південна Пальміра, перлина Причорномор'я">
    <meta name="description" content=«Одеса – перлина Причорноморського узбережжя,
    це одне з найбагатших на визначні пам'ятки місто. Південна Пальміра, Чорноморський
    Вавилон, Маленький Париж, Столиця Півдня – як тільки не називали це місто біля Чо-
    рного моря. ">
  </head>
  <body>
    <h1>Одеса</h1>
    <h2>Перлина Причорномор'я</h2>
    <p><strong>ОДЕСА</strong> – дійсно перлина Причорноморського узбережжя. Вона
    вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошкодуєте, як-
    що відвідаєте місто-гумористів і поринете в його таємниці. А побачити Одесу варто,
    оскільки це одне з найбагатших на визначні пам'ятки місто.</p>
    <p><em>Південна Пальміра</em>, <em>Чорноморський Вавилон</em>,
    <em>Маленький Париж</em>, <em>Столиця Півдня</em> – як тільки не називали це
    місто біля Чорного моря. Одеса різноманітна, але незважаючи на це, місто, за історич-
    ними мірками, молоде: йому всього трохи більше ніж 220 років. Але й за цей невеликий
    період воно пережило неймовірну кількість подій, що не могло не відбитися в архітек-
    турі тутешніх будівель і вулиць.</p>
    <hr><h2>Дещо з історії міста</h2>
    <p>Будівництво міста і морського порту почалося 22 травня 1794 року за указом Като-
    рини Другої, що повинно було значно покращити торгівельні зв'язки з Європою. Поча-
    лося будівництво під керівництвом віце-адмірала Хосе де Рибаса – першого градона-
    чальника Одеси, за планом, що склав полковник-інженер Франц Деволан.</p>
    <p>Тутешні землі можуть розповісти про скіфів, кіммерійців, сарматів і греків. Кожна
    культура залишила свій слід в історії міста. Місцевість, на якій нині розташована Оде-
    са, була заселена ще до початку Великого переселення народів, вона належала і Зо-
    лотій Орді, і Великому князівству Литовському.</p>
    <p>Відзначилася Одеса і в Другій світовій війні. <strong>Одеса – місто-герой</strong>.
    Оборона міста трималася <strong>73 дні</strong>. Щоб дізнатися більше про цей пері-
    од варто відвідати Меморіал героїчної оборони Одеси 411-ї берегової батареї. Цей
    меморіальний комплекс, присвячений героїчній обороні міста під час Великої Вітчизня-
    ної війни. Комплекс включає в себе музей, виставку військової техніки під відкритим
    небом, батарею берегової оборони, а також великий засаджений дубами парк.</p>
    <hr> <h2>Сьогодення одеситів</h2>
    <p>Сьогодні місто розвивається як курортний і промисловий центр. Це одне з найулю-
    бленіших туристами місць в Україні, і в цьому немає нічого дивного. Тут Ви можете ні-
```

```
житися в променях теплого південного сонця і купатися в ласкавому Чорному морі. Кількість пам'яток не злічити на пальцях однієї руки:</p>
<ul>
   <li><a href="http://odessa-life.od.ua/article/5449-sekrety-odesskogo-rynka-privoz">Привоз </a>, </li>
   <li><a href="http://www.odessaguide.net/sights_deribasovskayastreet.ru.html">вулиця Дерибасівська</a>,</li>
   <li>Одеський театр опери і балету (другий за красою в Європі),</li>
   <li>Приморський бульвар,</li>
   <li>Напівкругла площа (1826-1829) з пам'ятником Рішелье,</li>
   <li>палаци Воронцова, Наришкіної і Потоцького,</li>
   <li>Пасаж,</li>
   <li>Успенський монастир (1824) і Духовна семінарія,</li>
   <li>Одеський морський порт (найбільший на Чорному морі),</li>
</ul>
<p>– і це далеко не весь список. Для маленьких і непосидючих відкриває свої двері Одеський дельфінарій.</p>
<p>Все, що Ви можете побачити в Одесі. Є тут навіть «восьме чудо світу» – <em>Потьомкінські сходи</em>, своєрідний вхід у місто з боку моря.  2004 року було проведено голосування, серед найкрасивіших сходів, і в Європі Потьомкінські увійшли до десятки кращих, посівши 6-те місце, також у цьому списку присутні сходи Монмартр у Парижі та Сходи Храму Афіни на о. Родос.</p>
<p>В Одесі існує величезна система катакомб, вони не тільки за своїм хитросплетінням, але й за протяжністю (≈ 3 000 км) – одні з найбільших у світі.</p>
<p class="note"> Для порівняння: довжина Римських катакомб становить 300 км, Паризьких – 500 км.</p>
<p>Якщо Ви втомилися від споглядання пам'ятників, то ласкаво запрошуємо до Одеського Ботанічного саду, в якому Ви не лише відпочините душею і тілом, а й побачите фактично перший парк степової частини України. За двохсотлітню історію саду з усього світу у ньому назбиралася величезна колекція найрізноманітніших рослин.</p>
<p>Одеса – це місто веселощів та гумору. Відвідавши його лише раз, Вам обов'язково захочеться побачити його ще. Адже Одеса – це не зовсім місто – це <sup> ☺ </sup>посмішка Бога.<sub font-color="red">☼</sub></p>
<img src="photo/img1.jpg" alt="Одеса – це не місто, а ...">
<img src="photo/img0.jpg">
<img src="photo/img00.jpg">
<img src="photo/img000.jpg">
<p class="note"> Використовувались матеріали статті
<a href="http://angatravel.com.ua/ua/ah-odesa-perlina-bilya-morya.html">«АХ, ОДЕСА! ПЕРЛИНА БІЛЯ МОРЯ»</a> </p>
  </body>
</html>
```

Створити ще один файл з ім'ям **lb3.css.** Не закриваючи попередній файл, розмістити у вікні обидва файли у дві колонки (рис. 3.2) командою *View* / *Layout* / *Columns 2* і перетянувши відповідну вкладку у праве поле. У файл **lb3.css** записати:

```
h1, h2, h3 {
    font-family: Arial, Helvetica, sans-serif;
    text-align: center;
}
h1 {  font-size: 160%;   color: #003;  }
h2 {  font-size: 135%;   color: #333;  }
h3 {  font-size: 120%;   color: #900;  }
p {  font-family: Times, serif;text-align:  justify; }
strong {  color: red;  }
.note {
    padding-left: 15px;   border-left-width: 10px;
    color: white;             background-color: #999999
    margin: 10px 0;         padding: 5px 10px;
    font-size: 80%;         width: 70%;
}
img {  height: 40%;       margin-left: 10px;  }
```

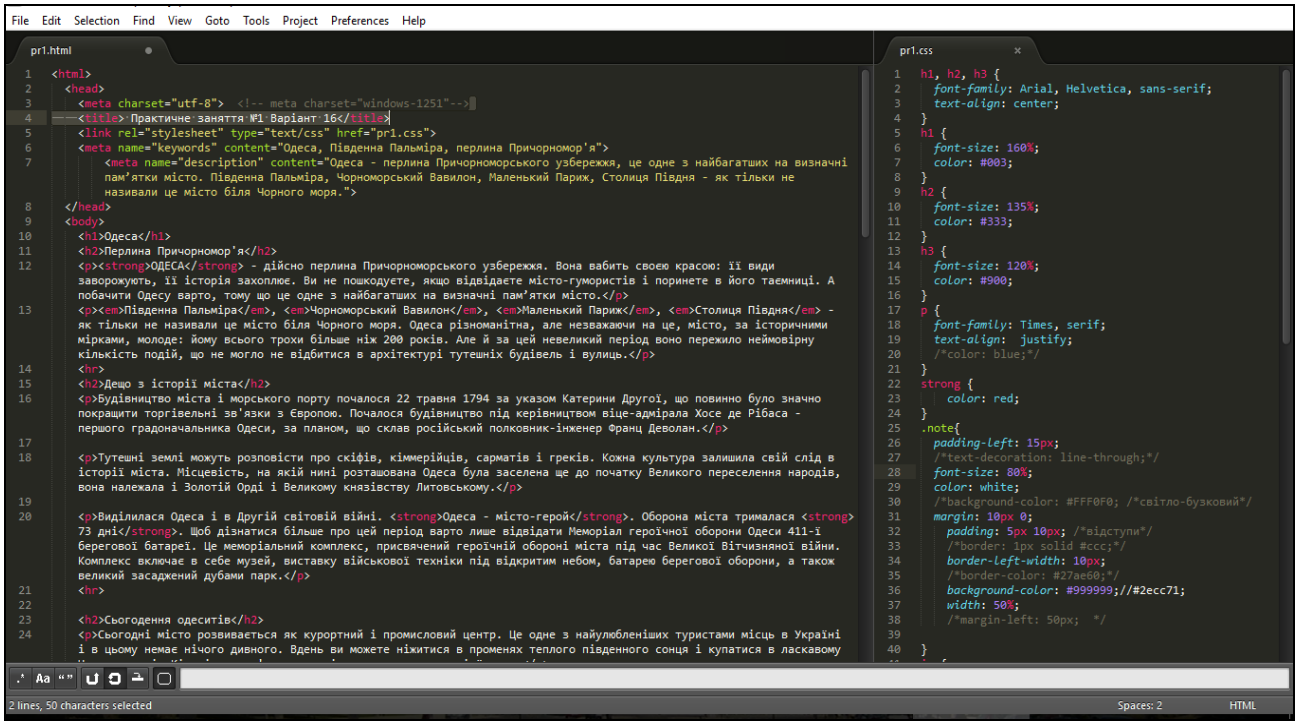



Рисунок 3.2 – Вигляд вікна редактора «SubLime Text 2»

*Самостійна робота № 2*

# РОЗМІЩЕННЯ САЙТУ ЖУРНАЛІСТА НА СЕРВЕР

## Завдання

1. Зареєструватися, замовити та отримати безкоштовний хостинг на www.ho.ua.
2. Перейменувати веб-сторінку, створену і лабораторній роботі № 3, на **index.html**.
3. Розмістити в свій сайт на сервері www.ho.ua створену веб-сторінку **index.html**. У подальшому створені в лабораторних роботах № 4 – 7 Ви довантажуватимете на цей сайт (див. самостійну роботу № 5).
4. Надати на перевірку викладачеві URL-адресу свого сайту.

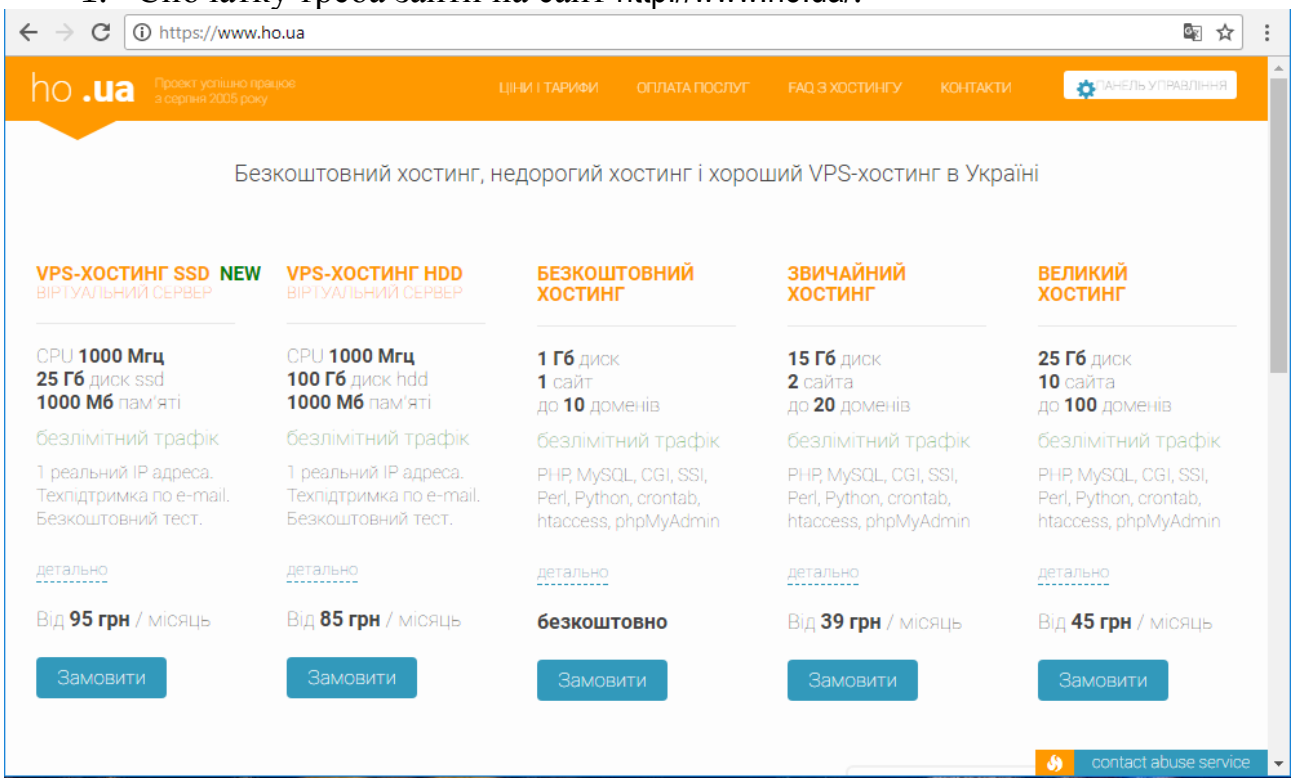
## Теоретичні відомості

### 1. Замовлення безкоштовного хостингу на www.ho.ua

**Хостинг** (англ. *hosting*) – послуга з надання ресурсів для розміщення інформації на сервер, який постійно перебуває в мережі (зазвичай інтернет).

Під час лабораторної роботи № 3 Ви почали створювати свій сайт, файли якого розмістили у власній папці (у наведеному прикладі файли були розміщені у папці prOdessa). Розглянемо послідовність завантаження (розміщення) сайту на сервер www.ho.ua. Особливістю цього сервера є те, що це український сервер, який надає послуги як платного, так і безкоштовного хостингу.

1. Спочатку треба зайти на сайт http://www.ho.ua/.



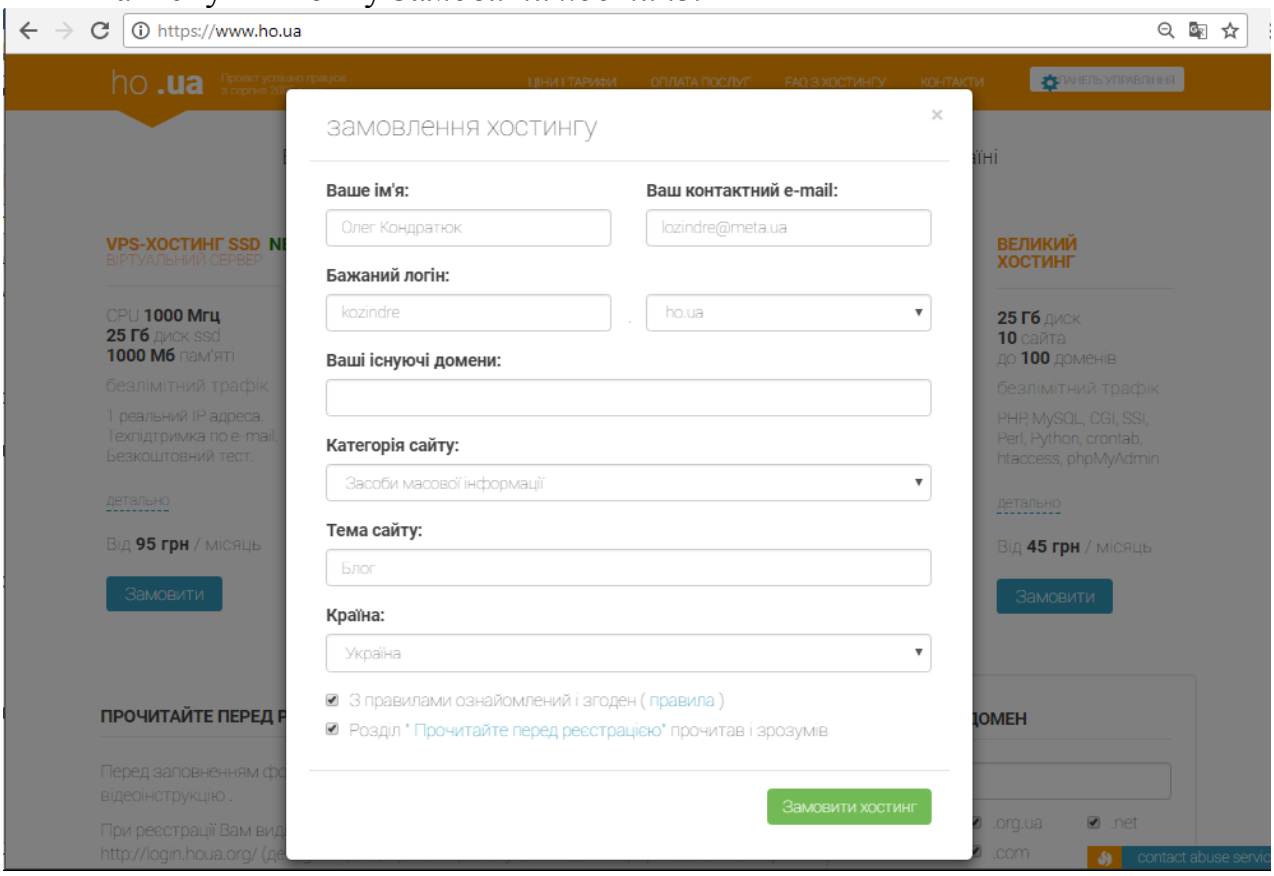
2. Замовити безкоштовний хостинг, натиснувши кнопку *Замовити* у колонці *Безкоштовний хостинг.*

3. Заповнити форму заявки:

– *Ваше ім'я* – як вас звуть (або як до вас звертатися);
– *Ваш контактний email* – електронна поштова скринька, на яку буде відправлений лист із підтвердженням реєстрації;
– *Бажаний логін* – ім'я (малі літери латиниці, цифри та тире) завдовжки до 16 символів, яке використовуватиметься для доступу по FTP. Це також частина адреси Вашого майбутнього хостингу (наприклад, *LOGIN.ho.ua,* де *LOGIN* – це вказаний Вами логін). Поруч з логіном пропонується на вибір два варіанти доменних імен для сайту (**.ho.ua* або **.houa.org*).
– *Категорія сайту* – вибір теми категорії майбутнього сайту;
– *Тема сайту* – короткий опис майбутнього сайту;
– *Країна* – це країна Вашого проживання.

Наприкінці реєстраційної форми слід уважно прочитати і встановити два прапорці, які означають, що Ви уважно прочитали і погодилися з правилами надання хостингу.

Натиснути кнопку *Замовити хостинг*.




Якщо ж при реєстрації виникли деякі помилки (неточності заповнення форми), про це буде повідомлено і запропоновано їх виправити. А якщо вказаний Вами логін вже зайнятий, то система запропонує ввести інший логін. Далі знову потрібно натиснути кнопку *Замовити хостинг*.

4. Якщо форму заповнено правильно і зазначений логін ніким ще незайнятий (такого немає у базі), Ви отримаєте повідомлення про те, що на Вашу електронну адресу відправлено листа з ключем (посиланням) активації хостингу. Вам слід уважно прочитати отриманий лист від noreply@ho.ua, в якому докладно описано два кроки для отримання замовленої послуги хостингу від ho.ua:

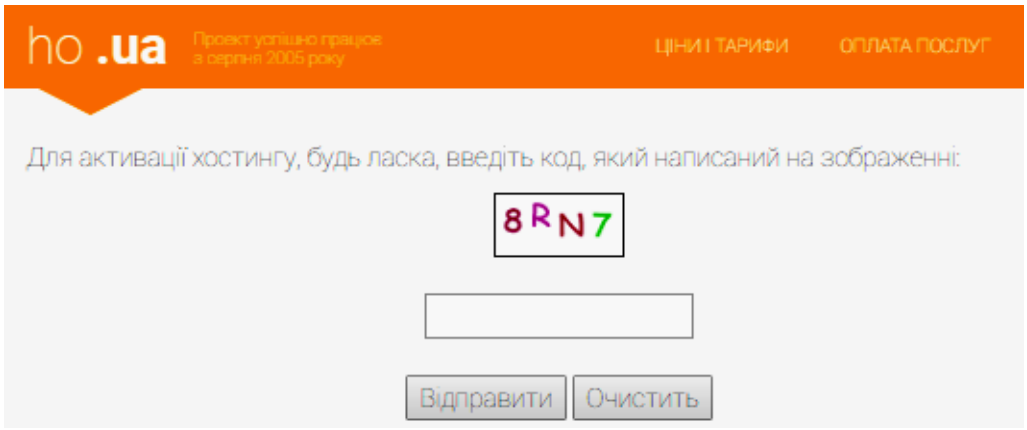
1) Підтвердження Вашої заявки через СМС. Для цього Вам треба відправити SMS-повідомлення на номер +380961020341, в якому вказати код Вашої заявки – …. (тут вказується надіслане чотиризнакове число, наприклад, 8855). Вартість СМС звичайна, залежить тільки від Вашого оператора мобільного зв'язку. Не треба відправляти SMS через веб-служби, оскільки вони приймаються лише з реальних мобільних телефонів.

2) Через деякий час (зазвичай 5 – 10 хвилин) залишиться активувати Ваш хостинг, перейшовши за надісланим посиланням: …… (тут вказується надіслане посилання, наприклад, http://www.ho.ua/cgi-bin/order.cgi?key=8855.

Після реєстрації можна відразу приступити до роботи з хостингом за тарифним планом «Звичайний хостинг», однак повністю веб-сервер буде налаштований до роботи протягом 1 години.

5. Отже, слід послати СМС по мобільному телефону і через 10 хв клікнути зазначене у повідомленні посилання (у нашому прикладі це http://www.ho.ua/cgi-bin/order.cgi?key=8855).

6. Система запропонує ввести зазначений код для активації хостингу:




7. У наступному вікні буде запропоновано зачекати три години, після чого перевірити свою електронну пошту.

8. У надісланому листі Вас повідомлять про успішне опрацювання заявки і створення сайту, який адмініструватиметься по FTP. Також будуть вказані дані сайту: логін, пароль, хостинг сервера, ім'я сервера та інші інструкції з подальшого наповнення сайту веб-сторінками з даними на кшталт такого повідомлення:

*Логін: kozindre*
*Пароль: GEWfeMUPrl*
*Хостинг сервера: s2.ho.ua*
*Ім'я сервера: kozindre.ho.ua*
*Сервер Alias: www.kozindre.ho.ua*
*Квота: 1*

*Всі сторінки слід поміщати в каталог HTDOCS. Зараз там містяться зразки документів і скриптів, роботу яких можна побачити:*

*http://kozindre.ho.ua/*
*http://kozindre.ho.ua/test.html*
*http://kozindre.ho.ua/test.shtml*
*http://kozindre.ho.ua/test.cgi*
*http://kozindre.ho.ua/test.php*

*У каталозі CGI-BIN сайту міститься інтерпретатор PHP. Він необхідний для роботи PHP.*

*У каталозі logs містяться файли протоколу роботи сайту. Розмір цих файлів враховується в загальній квоті. Для управління хостингом використовуйте сторінку https://www.ho.ua/cp/.*

*Доступ до БД MySQL Ви можете самостійно відкрити в Панелі управління хостингом. Для управління MySQL базою використовуйте сторінку https://www.ho.ua/phpMyAdmin.*

*Економ-пропозиція: зареєструй домен на ukraine-domain.com або перенеси своє доменне ім'я від іншого реєстратора та отримай безкоштовно півроку хостингу (за тарифним планом «Звичайний хостинг») http://ukraine-domain.com/ukrainian/.*

*Рекомендуємо реєструвати доменні імена тут: http://ukraine-domain.com/*

*Рекомендуємо почитати розділи FAQ «Початок роботи з хостингом» і «Редагування та налаштування сайту»: http://www.ho.ua/faq.html*

*А також лікнеп по хостингу: http://www.ho.ua/likbez.html*

*Не забудьте також стати членом Клубу користувачів www.ho.ua: http://www.ho.ua/forums/*

*У Клубі Ви завжди можете дізнатися багато цікавого, задати свої питання й отримати на них відповіді.*

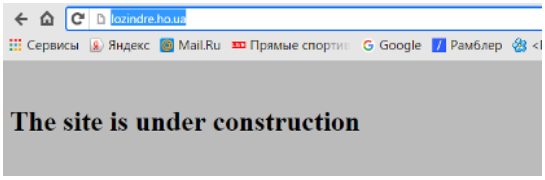
*Спробуйте наші економні віртуальні сервери (VPS): http://www.ho.ua/vps/*

*------*

*З повагою, команда підтримки*

## 2. Завантаження на сайт заздалегідь створеної веб-сторінки

1. У браузері ввести в адресному рядку URL-адресу свого сайту, вказану у повідомленні (у нашому прикладі це http://kozindre.ho.ua/). Оскільки сайт ще не заповнений, він матиме вигляд:

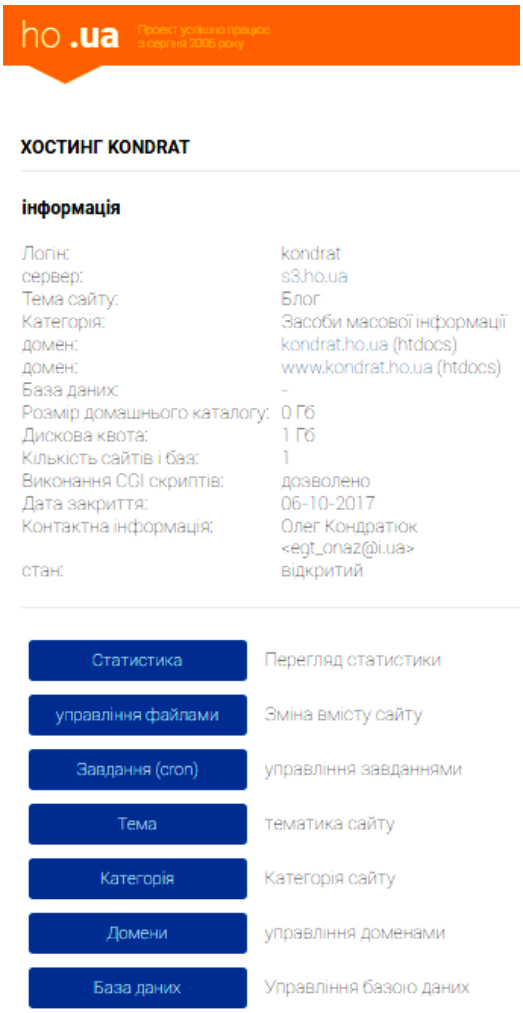



2. Ввести URL-адресу https://www.ho.ua/cp/, після чого ввести свій логін і пароль, вказані у листі.

3. Зайти до пункту *Управління базою даних* (сьома зверху синя кнопка *База даних*).

4. Ввести пароль двічі і натиснути *Зберегти*.

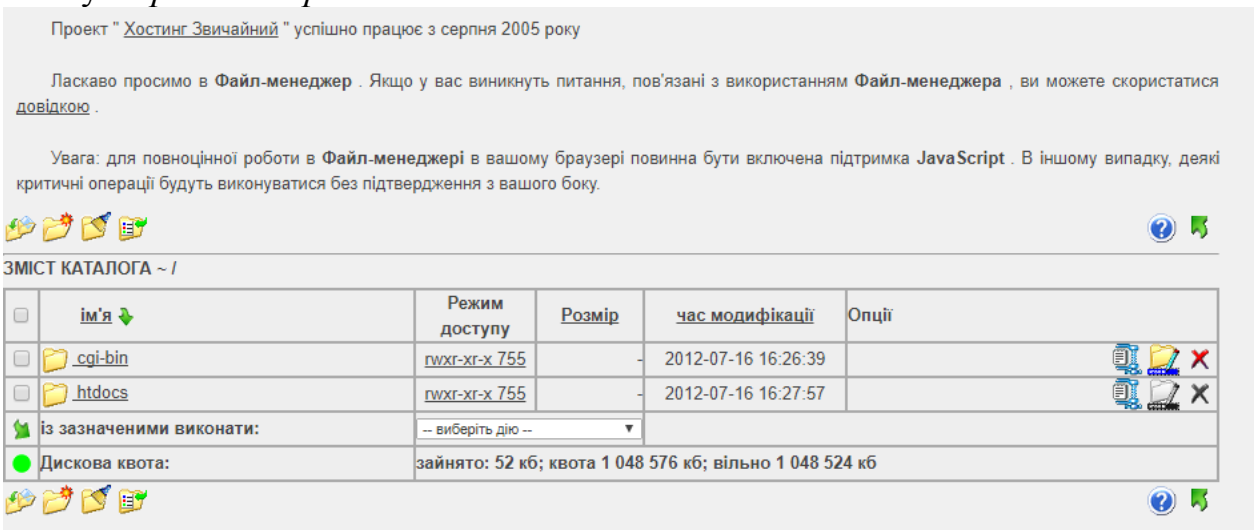
5. Якщо параметри бази даних Вас не задовольняють, можна змінити пароль, контактну інформацію, тематику сайту та інші налаштування, скориставшись відповідними опціями.



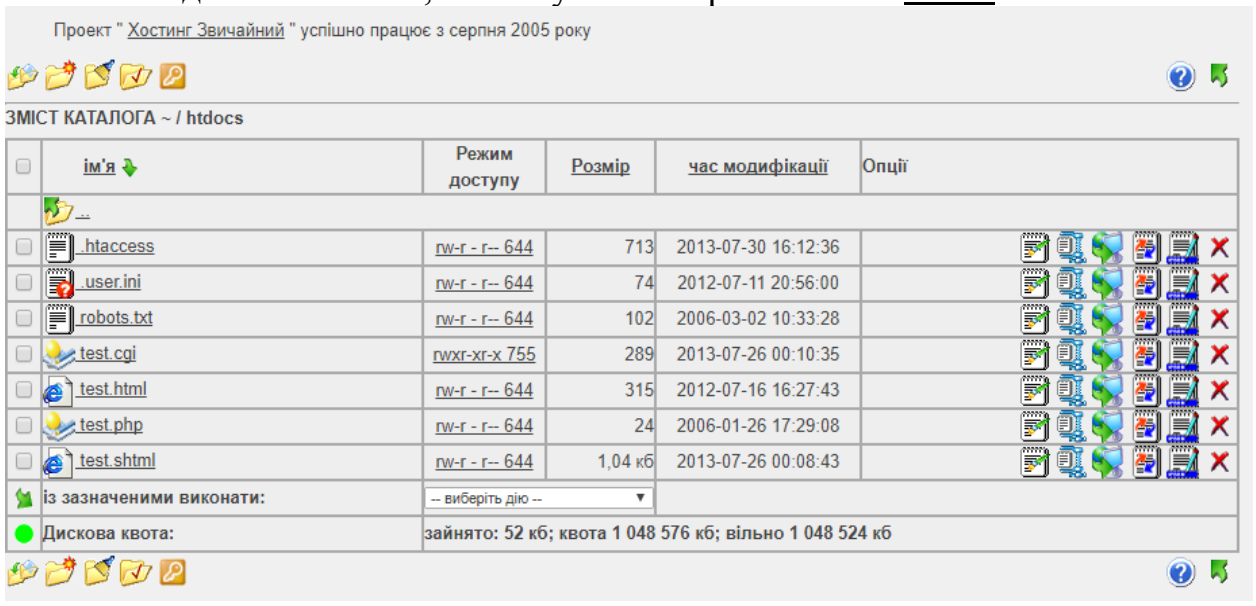
6. Згорнути вікно браузера та відкрити папку з файлами веб-сторінок для Вашого сайту (у нашому прикладі це папка prOdessa).

7. Тепер залишилося завантажити ці файли і папки на сервер. Для цього спочатку слід заархівувати всі вкладені файли і папки, які містяться всередині папки prOdessa (ім'я архіву у нашому прикладі prOdessa.zip, створювати треба саме *zip*-архів).

8. Повернутися до браузера на сторінку https://www.ho.ua/cp/ і натиснути кнопку *Управління файлами*.




Увійти до папки htdocs, натиснувши гіперпосилання htdocs.

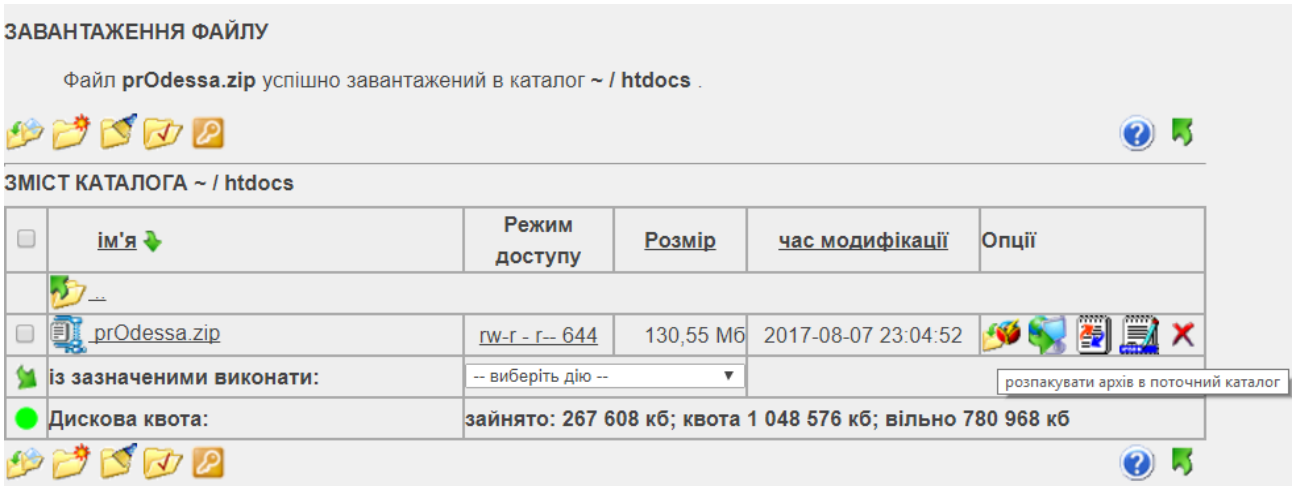



9. Видалити всі файли, для чого позначити ці файли і вибрати дію *Очистити*. Підтвердити видалення вибраних елементів, натиснувши *Так*.

10. При наведенні миші на лівий нижній значок з'явиться напис *Завантажити файл у каталог*. Натиснути на цей значок.

11. Клікнути кнопку *Вибрати файл*, що призведе до появи вікна вибору, в якому слід вибрати файл-архів (важливо використовувати саме *zip*-формат архіву), в нашому прикладі prOdessa.zip. Натиснути на кнопку *Відкрити*. Далі клікнути кнопку *Завантажити*.

12. У рядку з файлом prOdessa.zip праворуч є папка жовтого кольору, при наведенні миші на яку з'явиться підказка *розпакувати архів в поточний каталог*:




Слід клікнути цю папку для розпакування архіву та підтвердити дію, натиснувши *Так*. Файл розпаковано.

13. Тепер у новій вкладці інтернет-браузера слід набрати kozindre.ho.ua і натиснути *Enter*. За потреби оновити (перезавантажити) сторінку, натиснувши на клавішу *F5*. Тепер завантаження Вашого сайту на сервер виконано.

Якщо замість українських літер виводитиметься абракадабра, то слід змінити кодування браузера, наприклад, на Юнікод (UTF-8).

*Лабораторна робота № 4*

# СТВОРЕННЯ І ФОРМАТУВАННЯ ТАБЛИЦЬ НА HTML-СТОРІНЦІ

## Навчальні питання

1. Засоби HTML для створення таблиць.
2. HTML-атрибути та засоби CSS для форматування таблиць.
3. Особливості структурування таблиць.

## Завдання

1. Ознайомитись з інструментарієм HTML для створення і форматування таблиць, описаним у теоретичних відомостях до цієї лабораторної роботи.

2. Засобами HTML і CSS створити веб-сторінку з ім'ям **lb4.html**, де крім відповідного заголовка створити таблицю з даними про місто, для якого Ви розробили веб-сторінку в лабораторній роботі № 3. Організувати гіперпосилання для переходу між цими сторінками.

Дані у таблиці мають носити статистичний чи довідковий характер, а їхня кількість має бути такою, щоб веб-сторінка була рівномірно заповненою.

Надалі як приклад (п. 5 теоретичних відомостей) буде розглянуто засоби створення такої таблиці для міста Одеса:

<table>
<tr><td>Населення</td><td colspan="2">1 010 912 (01.05.2017)</td><td rowspan="2">Густота населення<br>6224 осіб/км²</td></tr>
<tr><td>Площа</td><td colspan="2">162.42 км²</td></tr>
<tr><td>Координати</td><td>46°29′08″ пн. ш.</td><td colspan="2">30°44′36″ сх. д.</td></tr>
<tr><td>Засновано</td><td colspan="3">XV століття</td></tr>
<tr><td>Міста-побратими</td><td colspan="3">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя, Єреван, Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель, Нікосія, Оулу, Пірей, Реґенсбург, Сеґед, Спліт, Стамбул, Хайфа, Циндао</td></tr>
<tr><td>Поділ міста</td><td colspan="3">4 райони</td></tr>
<tr><td>День міста</td><td colspan="3">2 вересня</td></tr>
<tr><td>Веб-сторінка</td><td colspan="3">http://omr.gov.ua</td></tr>
</table>

Для створення таблиці у файлі **lb4.html** використовувати тег <table>, який є контейнером для тегів <tr> (рядок), <th> (заголовок) і <td> (стовпець). За потреби об'єднання клітинок таблиці слід в якості параметрів тегів задати потрібні значення rowspan та colspan.

3. Для форматування створити CSS-стилі, які можна розмістити в HTML-коді сторінки всередині тегу <head> (або ж підключити їх як зовнішній файл, як це було у лабораторній роботі № 3). Переглянути результат у браузері.

4. Надати на перевірку викладачеві створені файл(и) веб-сторінки.

5. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Які теги призначені для задавання структури таблиці?
2. Який тег дозволяє додати таблицю на веб-сторінку? Яка його структура?
3. Яке призначення і структура тегу <th>? Навести приклади.
4. Яке призначення і структура тегу <tr>? Навести приклади.
5. Яке призначення і структура тегу <td>? Навести приклади.
6. У чому відмінність і схожість використання тегів <td> і <th>? Навести приклади.
7. Назвати теги, обов'язкові для створення таблиці.
8. Який атрибут застосовується для об'єднання декількох клітинок по стовпцям?
9. Яке призначення тегів <colgroup> та <col>? Навести приклади.
10. Яке призначення значення тегів rowspan та colspan? Навести приклади.
11. Як змінити ширину таблиці по всій ширині екрана? Яке значення ширини задається за замовчуванням?
12. Як для клітинок таблиці змінити фон?
13. Як задати змінювання кольору фону таблиці при наведенні на неї миші?
14. Назвати атрибути для форматування таблиці засобами HTML.
15. Назвати атрибути для форматування таблиці засобами CSS.
16. Яке призначення тегу <tfoot>?

## Теоретичні відомості

### 1. Засоби HTML для створення таблиць

Для створення таблиць призначений тег **<table>**, який є контейнером для елементів, що визначають вміст таблиці. Будь-яка таблиця складається з рядків і клітинок, які задаються за допомогою тегів <tr> і <td>. Усередині <table> допустимо використовувати теги (наведемо їх за абеткою):

- caption – заголовок таблиці;
- col – задає ширину та інші характеристики однієї чи декількох колонок таблиці;
- colgroup – задає ширину і стилі колонок (стовпців) таблиці.
- tbody – основна частина таблиці, її «тіло»;
- td – додати клітинку;
- tfoot – нижня частина таблиці, «підвал» таблиці. Звичайно дублює заголовок таблиці для великих за обсягом таблиць для зручності при прокручуванні;
- th – додати заголовну клітинку, звичайно використовується для першого рядка таблиці як «шапки» таблиці;
- thead – «шапка» таблиці – перший рядок(ки). Шрифт при цьому автоматично набуває напівжирного накреслення;
- tr – додати рядок.

До речі, таблиці з невидимою межею довгий час використовувалися для верстки веб-сторінок, дозволяючи розділяти документ на модульні блоки. Подібний спосіб застосування таблиць знайшов втілення на багатьох сайтах, поки йому на зміну не прийшли більш сучасні способи верстки.

Створення таблиці в HTML вимагає певної структури з наявністю обов'язкових тегів:

```
<table>          – відкрити таблицю;
 <tr>            – додати рядок;
  <td> ... </ td> – додати звичайну клітинку або заголовну через <th>;
 </ tr>          – кінець цього рядка;
</ table>        – кінець таблиці.
```

Закривний тег для <table> – обов'язковий.

Ця ієрархія обов'язкова і всі три елементи необхідні для побудови таблиці. При написанні коду треба визначати клітинки таблиці зліва направо і так до низу.

Наприклад:

```
<table>
 <tr>
  <td>Джон Леннон</td>     <td>Ритм-гітара</td>
 </tr>
 <tr>
  <td>Пол Маккартні</td>   <td>Бас</td>
 </tr>
 <tr>  <td>Джордж Харрісон</td>  <td>Соло-гітара</td>  </tr>
 <tr>  <td>Рінго Старр</td>      <td>Барабани</td>     </tr>
</table>
```

| Джон Леннон | Ритм-гітара |
|---|---|
| Пол Маккартні | Бас |
| Джордж Харрісон | Соло-гітара |
| Рінго Старр | Барабани |

Можна вдосконалити структуру (ієрархію) цієї таблиці, задавши головну (thead), основну (tbody) і нижню частини (tfoot) таблиці в такий спосіб:

```
 <style>
  table {
   border-collapse: collapse; /* Прибрати подвійні межі */
   border: solid thick;
  }
  td, th {
     border: 1px solid #333;
     padding: 2px; /* Поля в клітинках */
   }
   thead, tfoot {  background: #f0f0f0;  }  /* Колір фону */
   tbody:hover {
     background: #f5e0cd; /* Колір фону при наведенні вказівника миші */
     color: #ce232a; /* Колір тексту */
   }
</style>
<table>
 <caption>Група "The Beatles" </caption>
```

| Група «The Beatles» | |
|---|---|
| **Ім'я** | **Інструмент** |
| Джон Леннон | Ритм-гітара |
| Пол Маккартні | Бас |
| Джордж Харрісон | Соло-гітара |
| Рінго Старр | Барабани |
| **Ім'я** | **Інструмент** |

```
<thead>
 <tr>
  <th>Ім'я</th>      <th>Інструмент</th>
 </tr>
</thead>
<tfoot>
 <tr>   <th>Ім'я</th>      <th>Інструмент</th>   </tr>
</tfoot>
<tbody>
 <tr>
  <td>Джон Леннон</td>      <td>Ритм-гітара</td>
 </tr>
 <tr>
  <td>Пол Маккартні</td>    <td>Бас</td>
 </tr>
 <tr>
  <td>Джордж Харрісон</td>  <td>Соло-гітара</td>
 </tr>
 <tr>
  <td>Рінго Старр</td>       <td>Барабани</td>
 </tr>
</tbody>
</table>
```

Попри те, що <tfoot> розміщено перед </tbody>, все одно цей рядок буде відображено внизу таблиці.

При використанні тегів <thead>, <tbody> і <tfoot> слід дотримуватись правил:

- заголовок таблиці <caption> писати на початку таблиці, відразу після відкривного тегу <table>;
- тег <thead> писати у верхній частині сторінки, відразу після заголовка <caption>, якщо він присутній;
- тег <tfoot> вставляти перед <tbody>, але при цьому він завжди відображатиметься у нижній частині таблиці;
- елемент <tbody> для таблиць є обов'язковим, але для простих таблиць (без <thead> і <tfoot>) його можна не вказувати. Браузери самі навчилися вставляти його автоматично в код, розуміючи, що більшість розробників лінуються додавати елемент, який візуально ніяк не впливає на таблицю;
- тег <tbody> може бути один або кілька, розробник сам вирішує, за яким принципом їх додавати. Наприклад, якщо в деяких рядках є об'єднання клітинок, і, щоб рядки при наведенні на них вказівника миші виділялися, як нам потрібно (див. приклад вище), доречно їх об'єднати в <tbody>.

Для об'єднання стовпців і рядків таблиці слід скористатися атрибутами **colspan** і **rowspan** відповідно. Оскільки доволі складно відстежувати кількість потрібних клітинок при об'єднуванні, доречно спочатку створити повну таблицю, а тоді об'єднати певні клітинки, видаливши таким чином зайві клітинки.

Для декількох таблиць на веб-сторінці кожній з них доцільно поставити свій заголовок з назвою таблиці та/або її описом. Для цієї мети в HTML існує спеціальний тег **<caption>**, який задає текст заголовка. За замовчуванням заголовок розташовується вгорі таблиці по центру, як показано у вищенаведеному прикладі, його ширина не перевищує ширини таблиці, і в разі довгого тексту заголовка він автоматично переноситься на новий рядок. Для змінення вирівнювання заголовка використовується стильова властивість text-align.

Крім об'єднання групи рядків через <thead>, є ще один спосіб групування клітинок – по колонках за допомогою елементів **<col>** і **<colgroup>**. Кожна колонка (стовпець) таблиці зіставляється зі своїм елементом <col>, який пишеться після відкривного тегу <table>. Частину колонок можна об'єднати за допомогою атрибута span, його значенням є число поєднуваних колонок. Елементи <col> припустимо поміщати в один або кілька <colgroup>, об'єднуючи тим самим колонки.

Елементи <col> і <colgroup> ніяк не впливають на вигляд таблиці та її відображення у браузері і застосовуються тільки зі стилями, як буде показано далі у прикладі:

```
<section>
 <style>
  table {
     border-collapse: collapse; /* Прибрати подвійні межі */
     border: solid thick;
  }
  colgroup, tbody {
      border: solid medium;
  }
  td {
       border: solid thin;
       height: 1.4em; width: 1.4em;
       text-align: center;
       padding: 0; /* Поля в клітинках */
  }
 </style>
 <h1>Today's Sudoku</h1>
 <table>
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <tbody>
   <tr> <td> 1 <td>   <td> 3 <td> 6 <td>   <td> 4 <td> 7 <td>   <td> 9
   <tr> <td>   <td> 2 <td>   <td>   <td> 9 <td>   <td>   <td> 1 <td>
   <tr> <td> 7 <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td> 6
  <tbody>
   <tr> <td> 2 <td>   <td> 4 <td>   <td> 3 <td>   <td> 9 <td>   <td> 8
   <tr> <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td>
```

```
 <tr> <td> 5 <td>  <td>  <td> 9 <td>  <td> 7 <td>  <td>  <td> 1
<tbody>
 <tr> <td> 6 <td>  <td>  <td>  <td> 5 <td>  <td>  <td>  <td> 2
 <tr> <td>  <td>  <td>  <td>  <td>  <td> 7 <td>  <td>  <td>  <td>
 <tr> <td> 9 <td>  <td>  <td> 8 <td>  <td> 2 <td>  <td>  <td> 5
</table>
</section>
```

Зазначимо, що схожого результату можна досягти замінивши селектор <col> на <td> і <th>. У браузері ця таблиця матиме вигляд:

**Today's Sudoku**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | | 3 | 6 | | 4 | 7 | | 9 |
| | 2 | | | 9 | | | 1 | |
| 7 | | | | | | | | 6 |
| 2 | | 4 | | 3 | | 9 | | 8 |
| | | | | | | | | |
| 5 | | | 9 | | 7 | | | 1 |
| 6 | | | | 5 | | | | 2 |
| | | | | 7 | | | | |
| 9 | | | 8 | | 2 | | | 5 |

## 2. Атрибути для форматування таблиці

Назвемо HTML-атрибути форматування таблиць із зазначенням аналогічної CSS-властивості:

align – визначає вирівнювання таблиці (в CSS аналог – text-align);

background – задає фоновий рисунок у таблиці (в CSS – background-image);

bgcolor – колір фону таблиці (в CSS аналог – background-color);

border – товщина рамки у пікселях (в CSS аналог – border);

bordercolor – колір рамки (в CSS аналог – border-color);

cellpadding – відступ від рамки до вмісту клітинки (в CSS аналог – padding);

cellspacing – відстань між клітинками (в CSS аналог – border-spacing);

cols – кількість колонок (стовпців) у таблиці. Атрибут допомагає браузеру у подготовці до її відображення. Без цього атрибута таблиця буде показана тільки після того, як увесь вміст таблиці буде завантажено у браузер і проаналізовано. Використання атрибута cols дозволяє дещо прискорити відображення;

frame – повідомляє браузеру, як відображати межі таблиці (в CSS аналог – border). Значеннями можуть бути: void (без меж); border (межа навколо таблиці); above (межа по верхньому краю таблиці); below (внизу таблиці); hsides (лише горизонтальні межі вгорі і внизу таблиці); vsides (лише вертикальні межі ліворуч і праворуч таблиці); rhs (лише з правого боку); lhs (лише з лівого боку таблиці);

height – висота таблиці, задана цілим значення у пікселях або у відсотках від доступного простору (у CSS аналог – height);

rules – повідомляє браузеру, як відображати межі між клітинками (в CSS аналог – border). Значеннями можуть бути: all (лінія навколо кожної клітинки

таблиці); groups (лінія між групами, які утворені тегами <thead>, <tfoot>, <tbody>, <colgroup> або <col>); cols (лінія між колонками); none (без меж, за замовчуванням); rows (межа між рядками, створеними через тег <tr>);

summary – короткий опис таблиці. На відміну від тегу <caption> вміст summary не виводиться у браузері, однак може використовуватись пошуковими системами;

width – ширина таблиці. Також для цього тегу доступні універсальні атрибути і події (в CSS аналог – width).

Приклад можливого використання цих атрибутів:

```
<table bgcolor="#ffcc00" width="700" cellspacing="0" border="1" align="right" cols="10">
  <tr>
   <td> </td><td>1995</td><td>1996</td><td>1997</td><td>1998</td>
   <td>1999</td><td>2000</td><td>2001</td><td>2002</td><td>2003</td>
  </tr>
  <tr>
   <td>Золото</td><td>3</td> <td>6</td><td>4</td><td>6</td>
   <td>4</td><td>69</td><td>72</td><td>56</td><td>47</td>
  </tr>
 </table>
```

| | 1995 | 1996 | 1997 | 1998 | 1999 | 2000 | 2001 | 2002 | 2003 |
|---|---|---|---|---|---|---|---|---|---|
| Золото | 3 | 6 | 4 | 6 | 4 | 69 | 72 | 56 | 47 |

### 3. Особливості структурування таблиць

У кожного параметра таблиці є своє значення, задане за замовчуванням. З цього випливає, що якщо якийсь атрибут пропущений, то неявно він все одно присутній, причому з певним значенням. Тому вигляд таблиці може виявитися зовсім іншим, аніж припускав розробник. Щоб розуміти, що можна очікувати від таблиць, слід знати їхні явні і неявні особливості.

Так, одну таблицю допускається помістити всередину клітинки іншої таблиці. Це інколи потрібно у разі подання складних даних.

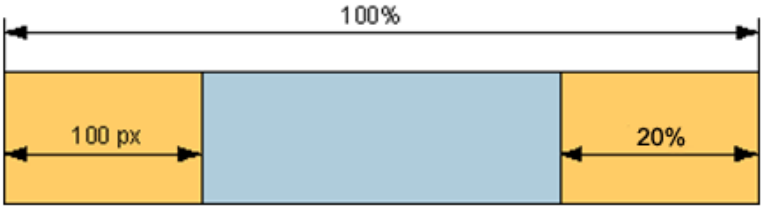
Розміри таблиці спочатку не задають, вониа обчислюються на основі вмісту клітинок. Так, загальна ширина визначається автоматично як сумарна ширина вмісту усіх клітинок плюс ширина меж між клітинками, полів навколо вмісту та відстані між клітинками.

Якщо для таблиці задано її ширину у відсотках або пікселях, то вміст таблиці підлаштовується під задані розміри. При цьому браузер автоматично задасть переноси рядків, щоб текст повністю вмістився у клітинку, а ширина таблиці залишилася без змін. Буває так, що ширину вмісту клітинки неможливо змінити, приміром, коли всередені клітинки є малюнок. У цьому разі ширина таблиці буде збільшена, дарма що точно зазначені розміри.

Допоки таблиця не завантажиться повністю, її вміст не буде відображатися. Справа в тому, що браузер, перш ніж показати вміст таблиці, має обчислити необхідні розміри клітинок, їхню ширину і висоту. А для цього треба знати, що в цих клітинках міститься. Тому браузер і чекає, поки завантажиться все, що є в клітинках, і тільки потім відображає таблицю.

### 4. Поєднання відсотків і пікселів при задаванні ширини

Розглянемо на прикладі варіант, коли ширина колонок задається і відсотками, і пікселями. Наприклад, у таблиці на весь екран (ширина 100%) із трьох колонок (стовпців) розмір першої колонки задається у пікселях, третьої – у відсотках, а ширина середньої колонки обчислюється автоматично як решта ширини таблиці.




Приклад програмного коду:

```
<!DOCTYPE html>
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Три колонки</title>
  <style type="text/css">
   table {    width: 100%;  }  /* Ширина таблиці */
   td {  vertical-align: top; }  /* Вирівнювання за верхнім краєм клітинки */
   #col1 {
      width: 100px;              /* Ширина першої колонки */
      background: #fc0;          /* Колір фону першої колонки */
     }
   #col2 {  background: #afccdb; }    /* Колір фону другої колонки */
   #col3 {  width: 20%; background: #fc0; } /* Ширина і колір третьої колонки */
   </style>
 </head>
 <body>
  <table cellpadding="5" cellspacing="0">
  <tr>
   <td id="col1">Колонка 1</td>   <td id="col2">Колонка 2</td>   <td id="col3">Колонка 3</td>
  </tr>
  </table>
 </body>
</html>
```

### 5. Приклад

Розглянемо засоби створення і форматування заявленої у завданні таблиці з чи статистично-довідковими даними про Одесу.

Для цього в редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) слід створити файл з ім’ям **lb4.html**, який зберегти поряд з файлом lb3.html, та записати в ньому таке:

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title> Лабораторна робота №4 Варіант 17</title>
      <meta name="keywords" content=«довідково-статистичні відомості про Одесу, дані
            довідкові про Одесу, статистичні дані про Одесу»>
      <style type="text/css">
      table { border: 2px solid gray;        /* стиль для таблиці */
              border-collapse: collapse;
              background-color: #F0F0FF; /*світло-блакитний*/
             }
      td     { border: 1px solid  gray;             /* стиль для клітинки */
               padding: 5px;
             }
      h1     { text-align: center;
               font-size: 28px;
               font-family: inherit;
               color: #618ad2;
             }
      </style>
   </head>
   <body background="photo/fon-110.gif">
   <h1>Довідково-статистичні відомості про Одесу</h1>
   <hr size=3 width="50%" align=center>
   <table >
      <tr>
         <td>Населення</td>  <td>1 010 912 (01.05.2017)</td>
         <td rowspan="2" width="30%">Густота населення<br>6224 осіб/км²</td>
      </tr>
      <tr>     <td>Площа</td>              <td>162.42 км²</td>         </tr>
      <tr>
         <td>Координати</td> <td> 46°29'08" пн. ш.</td>      <td> 30°44'36" сх. д.</td>
      </tr>
      <tr>
         <td>Засновано</td>
         <td colspan="2">XV століття</td>
      </tr>
      <tr>
         <td>Міста-побратими</td>
         <td colspan="2">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя, Єреван,
         Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель, Нікосія, Оулу,
         Пірей, Реґенсбург, Сеґед, Спліт, Стамбул, Хайфа, Циндао</td>
      </tr>
```

```
        <tr>
            <td>Поділ міста</td>
            <td colspan="2">4 райони</td>
        </tr>
        <tr>
            <td>День міста </td>
            <td colspan="2">2 вересня</td>
        </tr>
        <tr>
            <td>Веб-сторінка</td>
            <td colspan="2"><a href="http://omr.gov.ua">http://omr.gov.ua</a></td>
        </tr>
    </table>
    <hr>
        <p> <a href="lb3.html">Повернутись на головну сторінку </a> </p>
    </body>
</html>
```

Після запуску веб-сторінки у браузері вона матиме вигляд:

## Довідково-статистичні відомості про Одесу

| | | |
|---|---|---|
| Населення | 1 008 852 (01.05.2016) | Густота населення 6211 осіб/км² |
| Площа | 162.42 км² | |
| Координати | 46°29′08″ пн. ш. | 30°44′36″ сх. д. |
| Засновано | XV століття | |
| Міста-побратими | Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя, Єреван, Йокогама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель, Нікосія, Оулу, Пірей, Регенсбург, Сегед, Спліт, Стамбул, Хайфа, Циндао | |
| Поділ міста | 4 райони | |
| День міста | 2 вересня | |
| Веб-сторінка | http://omr.gov.ua | |

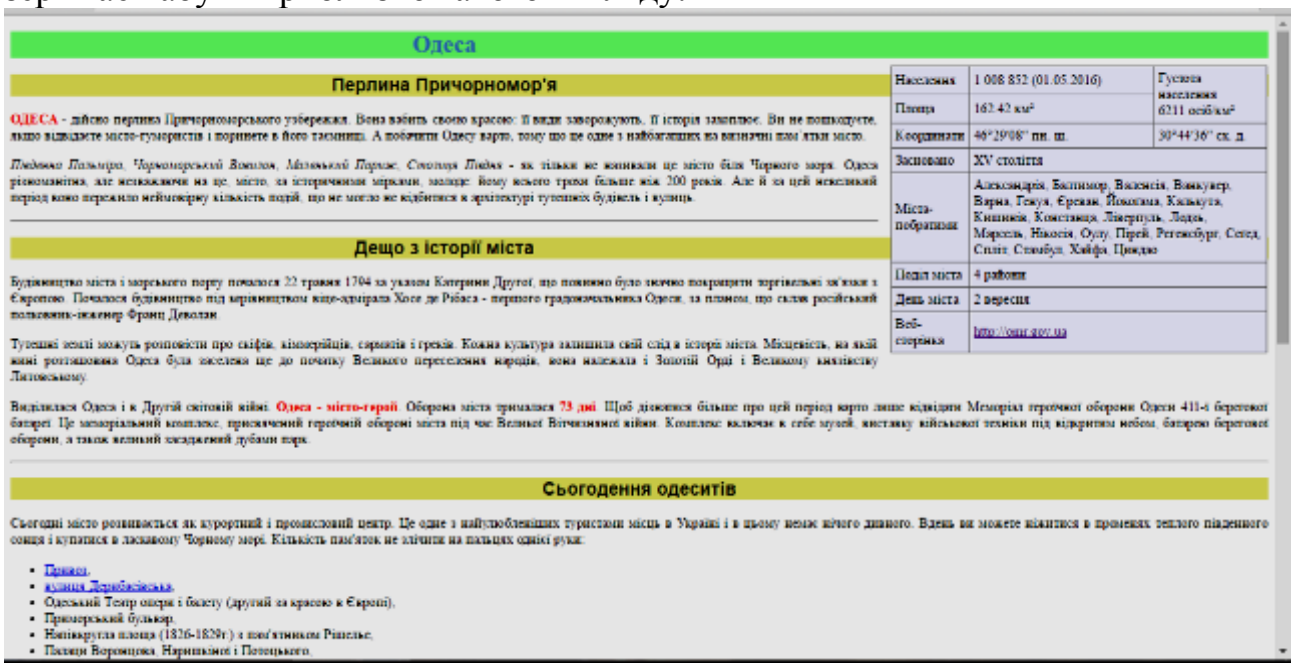
Повернутись на головну сторінку

*Самостійна робота № 3*

# ВБУДОВУВАННЯ ТАБЛИЦЬ З ОБТІКАННЯМ ТЕКСТОМ

## Завдання

Відкрити у HTML-редакторі веб-сторінку з ім'ям lb3.html, зберегти її з ім'ям **sr3.html** та розмістити на ній у правому верхньому куті поруч з текстом таблицю, створену в лабораторній роботі № 4. При цьому веб-сторінка у браузері має набути приблизно такого вигляду:




Для розміщення таблиці поруч з текстом використовувати структурний тег  (або тег ), для якого за допомогою атрибута <id> (унікальний ідентифікатор елемента по всьому документові) задати спеціально створений стиль розміщення.

Переглянути результат у браузері та надати його на перевірку викладачеві.

## Приклад

Розглянемо засоби розміщення таблиці, створеної під час лабораторної роботи № 4, у правому верхньому куті веб-сторінки, створеної в лабораторній роботі № 3.

Для цього в редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) слід відкрити файл з ім'ям lb3.html, який зберегти з ім'ям sr3.html поряд з файлом lb4.html та lb3.html, і скопіювати до цього файлу з файлу lb3.html фрагмент коду зі створенням таблиці таким чином (при цьому фрагмент коду з форматуванням скопіювати до файлу стилів lb3.css). Після усіх налаштувань HTML-файл набуде вигляду (нові рядки підкреслено):

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title> Самостійна робота №3 Варіант 17</title>
      <link rel="stylesheet" type="text/css" href="lb3.css">
      <meta name="keywords" content="Одеса, Південна Пальміра, перлина Причорномор'я">
      <meta name="description" content="Одеса – перлина Причорноморського узбережжя, це
            одне з найбагатших на визначні пам'ятки місто. Південна Пальміра, Чорно-
            морський Вавилон, Маленький Париж, Столиця Півдня – як тільки не назива-
            ли це місто біля Чорного моря.">
   </head>
   <body>
      <div id="placetabl">
         <table>
            <tr>
               <td>Населення</td>
               <td>1 010 912 (01.05.2017)</td>
               <td rowspan="2" width="30%">Густота населення<br>6224 осіб/км²</td>
            </tr>
            <tr>
               <td>Площа</td>
               <td>162.42 км²</td>
            </tr>
            <tr>
               <td>Координати</td>
               <td> 46°29'08" пн. ш.</td>
               <td> 30°44'36" сх. д.</td>
            </tr>
            <tr>
               <td>Засновано</td>
               <td colspan="2">XV століття</td>
            </tr>
            <tr>
               <td>Міста-побратими</td>
               <td colspan="2">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя,
               Єреван, Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Мар-
               сель, Нікосія, Оулу, Пірей, Реґенсбург, Сеґед, Спліт, Стамбул, Хайфа, Цин-
               дао</td>
            </tr>
            <tr>
               <td>Поділ міста</td>
               <td colspan="2">4 райони</td>
            </tr>
            <tr>
```

```
        <td>День міста </td>
        <td colspan="2">2 вересня</td>
      </tr>
      <tr>
        <td>Веб-сторінка</td>
        <td colspan="2"><a href="http://omr.gov.ua">http://omr.gov.ua</a></td>
      </tr>
    </table>
</div>
<h1>Одеса</h1>
<h2>Перлина Причорномор'я</h2>
<p><strong>ОДЕСА</strong> – дійсно перлина Причорноморського узбережжя. Вона вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошкодуєте, якщо відвідаєте місто-гумористів і поринете в його таємниці. А побачити Одесу варто, тому що це одне з найбагатших на визначні пам'ятки місто.</p>
<p><em>Південна Пальміра</em>, <em>Чорноморський Вавилон</em>, <em>Маленький Париж</em>, <em>Столиця Півдня</em> – як тільки не називали це місто біля Чорного моря. Одеса різноманітна, але незважаючи на це, місто, за історичними мірками, молоде: йому всього трохи більше ніж 220 років. Але й за цей невеликий період воно пережило неймовірну кількість подій, що не могло не відбитися в архітектурі тутешніх будівель і вулиць.</p>
<hr>
<h2>Дещо з історії міста</h2>
<p>Будівництво міста і морського порту почалося 22 травня 1794 року за указом Катерини Другої, що повинно було значно покращити торгівельні зв'язки з Європою. Почалося будівництво під керівництвом віце-адмірала Хосе де Рибаса – першого градоначальника Одеси, за планом, що склав полковник-інженер Франц Деволан.</p>
<p>Тутешні землі можуть розповісти про скіфів, кіммерійців, сарматів і греків. Кожна культура залишила свій слід в історії міста. Місцевість, на якій нині розташована Одеса, була заселена ще до початку Великого переселення народів, вона належала і Золотій Орді, і Великому князівству Литовському.</p>
<p>Відзначилася Одеса і в Другій світовій війні. <strong>Одеса – місто-герой</strong>. Оборона міста трималася <strong>73 дні</strong>. Щоб дізнатися більше про цей період варто відвідати Меморіал героїчної оборони Одеси 411-ї берегової батареї. Цей меморіальний комплекс, присвячений героїчній обороні міста під час Великої Вітчизняної війни. Комплекс включає в себе музей, виставку військової техніки під відкритим небом, батарею берегової оборони, а також великий засаджений дубами парк.</p>
<hr>
<h2>Сьогодення одеситів</h2>
<p>Сьогодні місто розвивається як курортний і промисловий центр. Це одне з найулюбленіших туристами місць в Україні, і в цьому немає нічого дивного. Тут Ви можете ніжитися в променях теплого південного сонця і купатися в ласкавому Чорному морі. Кількість пам'яток не злічити на пальцях однієї руки:</p>
<ul>
```

```
        <li><a href="http://odessa-life.od.ua/article/5449-sekrety-odesskogo-rynka-privoz">Привоз
        </a>, </li>
        <li><a
        href="http://www.odessaguide.net/sights_deribasovskayastreet.ru.html">вулиця Де-
        рибасівська</a>,</li>
        <li>Одеський театр опери і балету (другий за красою в Європі),</li>
        <li>Приморський бульвар,</li>
        <li>Напівкругла площа (1826-1829 р.) з пам'ятником Рішелье,</li>
        <li>палаци Воронцова, Наришкіної і Потоцького,</li>
        <li>Пасаж,</li>
        <li>Успенський монастир (1824 р.) і Духовна семінарія,</li>
        <li>Одеський морський порт (найбільший на Чорному морі),</li>
      </ul>
      <p>- і це далеко не весь список. Є тут навіть "восьме чудо світу" –
      <em>Потьомкінські сходи</em>, своєрідний вхід у місто з боку моря. У 2004 році
      було проведено голосування, серед найкрасивіших сходів, і в Європі Потьомкінські
      увійшли в десятку кращих, посівши 6-те місце, також у цьому списку присутні сходи
      Монмартр у Парижі та Сходи Храму Афіни на о. Родос.</p>
      <p>В Одесі існує величезна система катакомб, вони не тільки за своїм хитросплe-
      тінням, але й за протяжністю (≈ 3 000 км) – одні з найбільших у світі.</p>
      <p class="note"> Для порівняння: довжина Римських катакомб становить 300 км,
      Паризьких – 500 км.</p>
      <p>Якщо Ви втомилися від споглядання пам'ятників, то ласкаво запрошуємо до
      Одеського Ботанічного саду, в якому Ви не лише відпочините душею й тілом, а й по-
      бачите фактично перший парк степової частини України. За двохсотлітню історію у
      ньому з усього світу назбиралася величезна колекція найрізноманітніших рос-
      лин.</p>
      <p>Якщо Ви думаєте, що всі визначні місця знаходяться тільки в Одесі, то варто
      Вам виїхати за межі міста і Ви переконаєтесь в зворотному. Адже лише в 13 км від
      центру міста розташований курорт Куяльник – один з найдавніших на Україні. Ліку-
      вальна грязь, яка є чи не світовим еталоном, ось вже більше ніж століття допомагає
      людям у зміцненні здоров'я.</p>
      <p>Одеса – це місто веселощів та гумору. Відвідавши його лише раз, Вам
      обов'язково захочеться побачити його ще. Адже Одеса – це не зовсім місто – це по-
      смішка Бога.</p>
      <img src="photo/img1.jpg" alt="Одеса – це не місто, а ...">
      <img src="photo/img2.jpg">
      <img src="photo/img3.jpg">
      <img src="photo/img4.jpg">
      <p class="note"> Використовувались матеріали статті
      <a href="http://angatravel.com.ua/ua/ah-odesa-perlina-bilya-morya.html">"АХ, ОДЕСА!
      ПЕРЛИНА БІЛЯ МОРЯ"</a> </p>
    </body>
</html>
```

Після налаштувань CSS-файл набуде вигляду (нові рядки виокремлено жовтим маркером):

```
h1, h2, h3 {
    font-family: Arial, Helvetica, sans-serif;
    text-align: center;
}
h1 {
    font-size: 200%;
    color: #003;
    background-color: #89FF89;
}
h2 {
    font-size: 135%;
    color: #333;
    background-color: #E7E77F;
}
h3 {  font-size: 120%;  color: #900;  }
p {
    font-family: Times, serif;
    text-align:  justify;
}
strong {  color: red;}
.note {
    padding-left: 15px;
    font-size: 80%;
    color: white;
    margin: 10px 0;
    padding: 5px 10px;
    border-left-width: 10px;
    background-color: #999999;
    width: 50%;
}
img { height: 250px;    margin-left: 10px;  }
table {     /* стилі для таблиці */
    border: 2px solid gray;
    border-collapse: collapse;
    background-color: #F0F0FF;
}
td {   border: 1px solid  gray;    padding: 5px;  }
#placetabl {
    float: right;
    width: 30%;
    padding-left: 15px;
    padding-top: 40px;
}
```

*Лабораторна робота № 5*

# РОЗМІЩЕННЯ ЗОБРАЖЕНЬ НА ВЕБ-СТОРІНЦІ

***Мета:*** ознайомитися з основними засобами HTML та CSS для розміщення на веб-сторінці зображень та їхнього форматування.

## Навчальні питання

1. Засоби HTML для розміщення на веб-сторінці зображень.
2. Формати зображень.
3. Позиціонування зображень відносно інших елементів веб-сторінки.
4. Трансформація зображень.

## Завдання

1. Ознайомитись в теоретичних відомостях з основними засобами HTML та CSS для розміщення на веб-сторінці зображень та їхнього форматування.
2. Засобами HTML та CSS створити веб-сторінку з ім'ям **lb5.html**, де розмістити графічні зображення у вигляді ескізів за тематикою «Визначні місця…» того міста, для якого Вами розроблялась веб-сторінка в лабораторній роботі № 3. При наведенні мишею на будь-яке із зображень воно має збільшуватись. Організувати гіперпосилання для переходу між цією веб-сторінкою та сторінками, створеними під час лабораторних робіт № 3 і № 4.
3. Переглянути результат у браузері та надати його на перевірку викладачеві.
4. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Які засоби HTML призначені для вбудовування зображень у веб-сторінку?
2. Для чого призначений тег <img>? Який у нього синтаксис?
3. Які атрибути у тегу <img>? Який з них є обов'язковим, а які – ні?
4. Яке призначення атрибута alt у тегу <img>?
5. Навести приклади тегів для вставлення зображень у веб-сторінку із зазначенням різних параметрів зображення та з альтернативним текстом.
6. Для чого призначений тег <picture>?
7. Навести приклади гіперпосилання на відкриття зображення із зазначенням в атрибутах тегу різних параметрів зображення.
8. Які формати зображень можна використовувати на веб-сторінках? Які з них найпоширеніші?
9. У чому схожість та відмінність графічних форматів *gif* та *jpg*?
10. Коли доцільно використовувати формат *svg* для зображень на веб-сторінках?
11. Які засоби CSS існують для блокового позиціонування зображень на веб-сторінці без обтікання? Навести приклад.

12. Які засоби CSS існують для обтікання зображень текстом на веб-сторінці? Навести приклад.
13. Як для зображень використовувати CSS-властивості: margin. padding, border і background? Навести приклади.
14. Які засоби CSS існують для вирівнювання зображення відносно базової лінії тексту? Навести приклади.
15. Які засоби CSS існують для трансформації зображень? Які можливі візуальні модифікації можна задавати до елементів?
16. Навести приклад модифікації зображення при наведенні на нього вказівника миші.

## Теоретичні відомості

### 1. Засоби HTML для розміщення на веб-сторінці зображень

#### 1.1. Вбудовування зображень тегом <img>

Для розміщення зображень на веб-сторінці використовується **тег <img>**, який має два обов'язкових атрибути: src та alt. За потреби можна зробити так, щоб зображення було посиланням на інший файл, помістивши <img> всередину тегу <a>.

Також зображення можуть застосовуватися в якості карт-зображень, коли картинка містить активні області, що виконують функцію посилання. Така карта за зовнішнім виглядом нічим не відрізняється від звичайного зображення, але при цьому воно може бути розбите на невидимі зони різної форми, де кожна з областей слугує посиланням.

Розглянемо докладніше основні атрибути тегу <img>, які відповідають за відображення зображень, та їхні характеристики.

**Атрибут alt** задає альтернативний текст для зображень. Цей текст з інформацію про малюнок відображатиметься на сторінці в тому разі, коли не розпізнано зображення.

**Атрибути height** і **width** задають розміри відображення малюнка. Допускається використовувати значення у пікселях або відсотках. При цьому відсотковий запис задає розміри зображення щодо батьківського елемента – контейнера, де міститься елемент <img>. У разі відсутності батьківського контейнера, його роль відіграє вікно браузера, тобто запис width=100% означає, що зображення буде розтягнуте на всю ширину веб-сторінки.

Використання тільки одного атрибута width або height зберігає пропорції сторін зображення. Браузер при цьому очікує повного завантаження зображення, щоб визначити його первинну висоту і ширину.

```
<img src = "images / passag.jpg" alt = "Пасаж" width = "500px">
```

Одночасне задавання width і height може порушити пропорції зображення, в результаті чого воно буде спотворене: або витягнуте, або приплюснуте.

Крім того, для зазначення розмірів зображення можуть бути використані властивості width і height в CSS. Коли водночас використовуються властивості

CSS й атрибути HTML, то властивості CSS матимуть пріоритет над атрибутами HTML.

```
img {
  height: 200px;
  width: 200px;
}
```

Доцільно задавати розміри усіх зображень на веб-сторінці. Це дещо прискорює завантаження сторінки, оскільки браузеру немає потреби обчислювати розмір кожного зображення. Ширину і висоту зображення можна змінювати як в менший, так і в більший бік. Однак, на швидкість завантаження зображення це аж ніяк не впливає, оскільки розмір файлу залишається незмінним. Тому слід з обережністю зменшувати розміри зображень, оскільки це викликає подив читачів: чому таке мале зображення так довго завантажується? І навпаки, збільшення розмірів спричиняє зворотній ефект: файл зображення аналогічного розміру завантажується швидше, а якість малюнка при цьому погана.

**Атрибут src** задає адресу графічного файлу, який відображатиметься на веб-сторінці. Значенням може бути повний або відносний шлях до файлу.

Додавання зображення на сторінку з урахуванням цих атрибутів проказано у такому прикладі:

```
<! DOCTYPE html>
<html>
 <head>
  <meta charset = "utf-8">
  <title> Зображення </ title>
 < head>
 <body>
  <img src = "images/passag.jpg"  alt = "Одеський пасаж"  height = "375">
 </body>
</html>
```

Якщо після додавання зображення воно не відображається на веб-сторінці, слід перевірити такі моменти. Назва файлу має писатися з урахуванням регістру. Операційна система Linux, на базі якої переважно встановлюється хостинг, педантично ставиться до імен файлів, тому файли *img.png*, *IMG.png* та *Img.png* сприймаються, як різні. Щоб зазначена особливість не спричинила помилок з відображенням малюнків, слід задавати імена файлів зображень у нижньому регістрі. Інакше кажучи, слід записувати їх малими літерами. Графічні файли з символами кирилиці у назвах файлів не завжди коректно відображаються у браузерах. Краще використовувати для цього символи латиниці.

До речі, в HTML5 з'явилися додаткові атрибути тегу <img>:

sizes – задає розміри зображення для різних макетів сторінки;

srcset – шлях до графічних файлів з урахуванням розміру зображення і пристроїв.

### 1.2. Тег <picture>

По суті тег <picture> є контейнером для зберігання декількох тегів <source>, які підтримують тег <img>. Це дозволяє вказати різні зображення з урахуванням розміру екрана, щільності пікселів, співвідношення сторін екрана та інших параметрів. Ось декілька областей застосування <picture>:

- для екранів Retina планшетів iPad можна показувати картинку більшого розміру;
- можна виводити зображення різного розміру для мобільних і настільних пристроїв;
- відображати зображення різних пропорцій, які враховують орієнтацію пристрою;
- можна виводити зображення у векторному форматі *svg*, а для браузерів, які його не підтримують, показувати *png*.

Синтаксис тегу <picture> є таким:

```
<picture>
  <source>
  <img>
</picture>
```

Усередині <picture> може міститися нуль або декілька елементів <source>, які йдуть перед одним обов'язковим тегом <img>. Закривний тег </picture> є обов'язковим.

Наприклад:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>picture</title>
 </head>
 <body>
  <picture>
    <source srcset="image/html5-logo.svg">
    <img src="image/html5-logo.png" width="256" height="256" alt="HTML5">
  </picture>
 </body>
</html>
```

У цьому прикладі використовуються два зображення: одне у форматі *svg*, а друге у звичному *png*. Браузери, які підтримують тег <picture>, відобразять картинку у векторному вигляді. Браузер IE покаже зображення у форматі *png*. Для наочності логотип подано розміром 128×128 пікселів і збільшено до 256×256.

До речі, файли зображень для веб-сторінок створюваного сайту доречно розміщувати в окремій папці з назвою images, яку слід створити у папці, де розміщуються HTML та CSS-файли сайту. У свою чергу, у папці images слід створити інші папки з назвами веб-сторінок сайту спеціально для зображень на відповідних сторінках. Така структурованість допоможе не заплутатись та швидко віднайти потрібне зображення.

## 2. Формати зображень

Різні браузери можуть підтримувати (або не підтримувати) різні формати зображень на веб-сторінках. Найбільш поширеними і підтримуваними форматами зображень в інтернеті є *gif*, *jpg* і *png*. З них найбільш використовуваними сьогодні форматами є *jpg* і *png*. Є ще формат *svg*, про який слід сказати окремо.

**Формати *jpg*, *jpeg*** (від англ. *Joint Photographic Experts Group* – об'єднана група експертів в області фотографії) забезпечують якість зображення з високою кількістю кольорів, зберігаючи скромний розмір файлу, що ідеально підходить для швидкого завантаження на веб-сторінках. Формат *jpeg* підтримує 24-розрядний колір (тобто приблизно 16 мільйонів кольорів у зображенні, що цілком достатньо для збереження фотографічної якості зображення) і зберігає незмінними яскравість і відтінки кольорів у фотографіях. Формат *jpeg* не підтримує прозорість. Коли Ви зберігаєте фотографію у цьому форматі, прозорі пікселі заповнюються певним кольором.

**Формат *gif*** (від англ. *Graphics Interchange Format* – формат обміну графічними даними) використовує 8-бітовий колір (тобто кількість кольорів у зображенні може бути до 256) й ефективно стискає суцільні кольорові області, при цьому зберігаючи деталі зображення. Формат підтримує покадрове змінення зображень, що робить його популярним для створення банерів і простої анімації. Галузь застосування формату: логотипи, ілюстрації з чіткими краями, анімовані малюнки, зображення з прозорими ділянками, банери.

**Формат *png*** (від англ. *Portable Network Graphics* – портативна мережева графіка) добре підходить для зображень із прозорістю або малим числом кольорів. Формат за своєю дією схожий на *gif*, оскільки зберігає дрібні деталі зображення і підтримує багаторівневу прозорість, проте на відміну від *gif* не відображує анімацію ні в якому вигляді. Через те, що використовуваний алгоритм стискання зберігає всі кольори і пікселі в зображенні незмінними, порівняно з іншими форматами у *png*, кінцевий розмір файлу з фотографією виходить найбільшим. Звичайно зображення в *jpg* використовуються для фотографій, а зображення в *png* – для іконок, фонових візерунків, логотипів, ілюстрацій з чіткими краями, зображень із прозорістю.

**Формат *svg*** (від англ. *Scalable Vector Graphics* – масштабована векторна графіка) – це векторний формат, на відміну від попередніх растрових форматів. Растрове зображення складається з набору різнокольорових пікселів, які для людського ока зливаються в єдину картинку. Векторне ж будується з набору об'єктів, на зразок ліній, кривих, прямокутників, кіл тощо. При збільшенні масштабу векторне зображення збільшується пропорційно, зберігаючи свою високу якість. Написи в *svg*-зображенні залишаються звичайним текстом, їх можна виділяти, копіювати, вони читаються пошуковими системами при обході сайту. Слід зазначити, що формат не підтримується браузером Internet Explorer до версії 9.0. Галуззю застосування формату є масштабовані зображення, логотипи, ілюстрації, графіки і діаграми.

## 3. Позиціонування зображень відносно інших елементів веб-сторінки

### 3.1. Позиціонування (position) зображень

Існують різні підходи для позиціонування зображень на веб-сторінці. За замовчуванням зображення позиціонуються як рядково-блокові елементи, проте їхнє розташування може бути змінено за допомогою CSS, зокрема float, display і властивостями блокової моделі, включаючи padding, border і margin.

Щоб задати позиціонування елемента, при зазначення властивості **position** використовують властивості[7]: top (зверху), bottom (знизу), left (зліва) та right (справа). Наприклад, розмістити картинку у верхньому куті з відступами згори і справа у 30px можна в такий спосіб:

```
img {
        position: fixed;
        top: 30px;   left: 30px;
    }
```

### 3.2. Види позиціонування

**static** – статичне позиціонування, задається за замовчуванням. Елемент займає у розмітці місце відповідно до свого розташування в HTML-коді. На статично позиціонований елемент не діють властивості top, bottom, left та right;

**absolute** – абсолютне позиціонування відносно вікна браузера або свого батьківського елемента, для якого задано властивість **relative** або **absolute** (не **static**). Якщо в усіх батьківських елементів статичне позиціонування, батьківським елементом для абсолютного позиціонованого елемента буде елемент html. При прокручуванні абсолютно позиціонований елемент прокручуватиметься разом з текстом, вивільняючи місце наступним елементам. Абсолютно позиціоновані елементи можуть перекривати інші елементи.

```
h2 {
        position: absolute;
        top: 150px;   left: 100px;
    }
```

**relative** – відносне позиціонування щодо місця, де елемент виводиться у звичайному потоці даних. Відносно позиціоновані елементи можуть перекривати інші елементи. Відносно позиціоновані елементи часто використовують як контейнери для абсолютно позиціонованих елементів. Приклад:

```
h2.pos_left  { position: relative; left: -20px;   }
h2.pos_right { position: relative; right: 20px;   }
```

**fixed** – фіксоване позиціонування закріплює елемент у певному місці екрана. Тим самим таке позиціонування дозволяє розмістити елемент у будь-якій точці відносно вікна браузера. При прокручуванні фіксовано позиціонований

[7] Для усіх видів позиціонування, крім static.

елемент не прокручується і не змінює своє розташування. Фіксовано позиціоновані елементи можуть перекривати інші елементи.

```
p.pos_fixed {
        position: fixed;
        top: 40px;
        right: 5px;
    }
```

**inherit** – елемент наслідує значення від батьківського елемента.

### 3.3. Рядкове позиціонування зображень

Елемент <img> є за замовчуванням рядково-блоковим. Додавання зображення на сторінку без будь-яких стилів позиціонуватиме його на тому самому рядку, що і вміст навколо зображення. Крім того, зміниться висота рядка, в якому з'явиться зображення, щоб відповідати висоті зображення, і це може створити великі вертикальні зазори у рядку.

Лишати вставлені зображення позиціонованими за замовчуванням недоречно через нераціональне використання місця. Здебільшого зображення задаються як блокові або задається їхнє обтікання текстом з одного якогось боку, наприклад, з правого боку.

### 3.4. Блокове позиціонування зображень

Додавши властивість display до зображення та задавши її значення як block, можна змусити зображення бути блоковим елементом. Це розмістить зображення на окремому рядку, а навколишній вміст розташовуватиметься вище та нижче зображення.

```
img {
  display: block;
}
```

### Позиціонування зображень ліворуч або праворуч (обтікання зображення)

Для створення обтікання зображення текстом існує декілька способів. Найзручніший, звичайно ж, пов'язаний із застосуванням стилів.

Для обтікання картинки текстом застосовується стильова **властивість float**. Значення right вирівнюватиме зображення за правим краєм батьківського елемента або вікна браузера, а текст розміщувати ліворуч від зображення. Значення left, навпаки, вирівнюватиме зображення за лівим краєм, а текст – праворуч від малюнка. Елемент, для якого задане значення float, зазвичай називається **плаваючим**. Ця назва, звичайно ж, умовна і свідчить лише про те, що текст чи то інші об'єкти будуть обтікати його з різних боків.

Для забезпечення вільного простору (відступів) навколо зображення слід скористатися властивістю margin. Додатково можна застосовувати властивості padding, border і background, щоб створити рамку навколо зображення:

```
img {
   background: #eaeaed;
   border: 1px solid #9799a7;  /* сіра рамка */
   float: right;               /* вирівнювання за правим краєм */
   margin: 8px 0 0 20px;       /* відступ зверху і зліва */
   padding: 4px;
 }
```

Доволі часто при блоковому розміщенні декількох зображень на сторінці виникає потреба перевірки того, що вони не перевищують ширину відведених колонок (блоків). Для цього можна в окремо створеному класі з ім'ям effigy задати їхню максимальну ширину, як 100% від ширини блоку. Це дозволить усім зображенням підлаштовуватись під ширину відведених колонок. Крім того, можна трохи закруглити кути зображень і застосувати до них нижній відступ (властивість margin), наприклад, у 22 пікселі:

```
.effigy img {
              border-radius: 5px;
              display: block;
              margin-bottom: 22px;
              max-width: 100%
            }
```

До речі, якщо задати для властивості border-radius значення 50%, зображення перетвориться на круг. Для деяких зображень, наприклад для портретів, це доречно.

Розглянемо приклад застосування float, в якому створені два класи з ім'ям left і right, додавання яких до тегу <img> або <figure> вирівнюватиме їх за відповідним краєм. Щоб текст трохи відступав від картинки, використовується властивість margin:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Зображення</title>
  <style>
   figcaption {
    text-align: center;
   }
   .left {
    float: left;                 /* вирівнювання за лівим краєм */
    margin: 0 1em 1em 0;    /* відступ справа і знизу */
   }
   .right {
    float: right;                /* вирівнювання за правим краєм */
    margin: 0 0 1em 1em;    /* відступ знизу і зліва */
   }
  </style>
```

```
 </head>
 <body>
  <figure class="left">
   <img src="images/fig.jpg" width="200" alt=" " >
   <figcaption>Підпис знизу</figcaption>
  </figure>
  <p>Текст</p>
 </body>
</html>
```

**Вирівнювання зображення відносно базової лінії тексту**

Властивість **vertical-align** вирівнює зображення відносно свого батьківлького елемента, наприклад, відносно базової лінії тексту абзацу або клітинки таблиці.

За замовчуванням картинка вирівнюється за базовою лінією – невидимою горизонтальною лінією, що проходить по нижньому краю символів. Проте деякі літери (д, р, у, ф, ц, щ) мають нижні виносні елементи, які виходять за базову лінію.

Синтаксис властивості **vertical-align**:

vertical-align: baseline|bottom|middle|sub|super|text-bottom|text-top|top|inherit|*значення*|*відсотки*

Значеннями властивості vertical-align можуть бути:

baseline – вирівнює базову лінію поточного елемента по базовій лінії батька. Коли батьківський елемент не має базової лінії, то за неї береться нижня межа елемента;

bottom – вирівнює основу елемента по нижній частині елемента;

middle – вирівнює середню точку елемента по базовій лінії батька плюс половина висоти батьківського елемента;

sub – нижній індекс без змінення розміру шрифту;

super – верхній індекс без змінення розміру шрифту;

text-bottom – вирівнює нижню межу елемента за самим нижнім краєм поточного рядка;

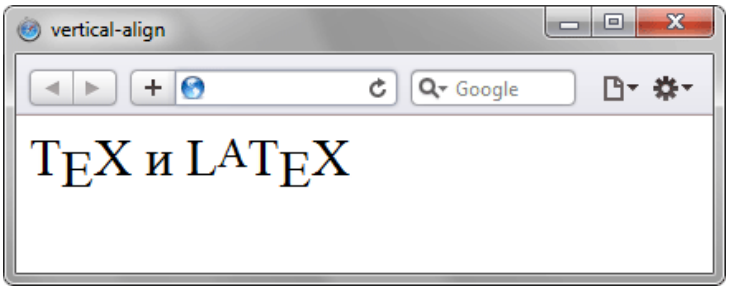
text-top – вирівнює верхню межу елемента за самим високим текстовим елементом поточного рядка;

top – вирівнює верхній край елемента по верху самого високого елемента рядка;

inherit – успадковує значення батьківського елемента.

Значення можна задавати у відсотках, пікселях або інших доступних одиницях виміру. Додатне число зсовує елемент догори відносно базової лінії, у той час як від'ємне число опускає його донизу. При використанні відсотків відлік ведеться від значення властивості line-height, при цьому значення 0% аналогічне значенню baseline.

Приклад:




```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>vertical-align</title>
 </head>
<body>
 <div style="font-size: 2em">
 T<span style="vertical-align: sub">E</span>X та L
 <span style="vertical-align: 5px;  font-size: 0.8em">A</span>T
 <span style="vertical-align: sub">E</span>X
 </div>
</body>
</html>
```

## 4. Трансформація зображень

**Трансформація** – це змінення вигляду елемента з такими можливими візуальними модифікаціями: поворот, масштабування, нахил і зсув. Для трансформації до селектора додається властивість transform, зазначається функція трансформації та її параметри. В загальному вигляді це записується так:

transform: функція (параметри);

Наведемо приклад програмного коду для змінення масштабу фотографій при наведенні на них вказівника миші.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="UTF-8">
  <title>Трансформація</title>
  <style>
   .thumbs {
    display: inline-block;
   }
   .thumbs:hover .thumb {
    transform: scale(0.9);  /* Зменшити всі фотографії */
   }
   .thumb {
    background: #297770; /* Колір рамки */
    padding: 10px;           /* Товщина рамки навколо картинки */
    transition: 1s;          /* Час переходу */
   }
   .thumb:hover {
    transform: scale(1.1) !important; /* Збільшити фотографію */
   }
  </style>
```

```
 </head>
 <body>
  <div class="thumbs»>
   <img src="images/thumb1.jpg" alt=" "  class="thumb">
   <img src="images/thumb2.jpg" alt=" " class="thumb">
   <img src="images/thumb3.jpg" alt=" " class="thumb">
  </div>
 </body>
</html>
```

При наведенні вказівника миші на будь-яку фотографію вона трохи (на 10%) збільшиться в розмірах, а решта фотографій, навпаки, зменшиться на 10%.

Для такого ефекту використовується псевдоклас :hover для класу thumbs і функція scale() для зменшення усіх зображень на 10% (аргумент 0.9). Для подальшого збільшення вибраної фотографії слід задіяти для неї :hover з аргументом 1.1. Правило !important застосовується для підвищення пріоритету стильового правила, оскільки зменшення зазвичай має вищий пріоритет.

Детальніше про види і засоби транформації див. за посиланням: http://shpargalkablog.ru/2011/07/transformaciya-css.html.

## 5. Приклад

Як приклад виконання поставленого завдання створимо нову веб-сторінку про видатні місця Одеси такого вигляду:


Для цього в редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) слід створити файл з ім’ям **lb5.html**, в якому записати таке:

```
<html>
   <head>
      <meta charset="utf-8">
      <title> Лабораторна робота № 5 Варіант 17</title>
      <link rel="stylesheet" type="text/css" href="lb5.css">
      <meta name="keywords" content="Одеса, видатні місця Одеси">
      <meta name="description" content="Видатні місця Одеси">
   </head>
   <body background="photo/fon-110.gif">
    <div class="thumbs">
      <h1 aligment="center">Видатні місця Одеси</h1>
      <img src="photo/img_lb5_1.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_2.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_3.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_4.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_5.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_6.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_7.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_8.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_9.jpg" alt=" "  class="thumb">
      <img src="photo/img_lb5_10.jpg" alt=" " class="thumb">
      <img src="photo/img_lb5_11.jpg" alt=" " class="thumb">
      <img src="photo/img_lb5_12.jpg" alt=" " class="thumb">
      <p><a href="lb3.html">Повернутись на головну сторінку сайту</a></p>
    </div>
  </body>
</html>
```

Також слід створити ще один файл з ім'ям **lb5.css**, в якому записати таке:

```
.thumbs {   display: inline-block;  }
.thumbs:hover .thumb {   transform: scale(0.9);   }    /* Зменшити всі фотографії */
.thumb {
   background: #297770; /* Колір рамки */
   padding: 10px;           /* Розмір рамки навколо картинки */
   transition: 1s;          /* Час переходу */
}
.thumb:hover {    transform: scale(1.2) !important; }      /* Збільшити фотографію */
img {
  height: 250px;
  margin-left: 10px;
}
h1  {
  text-align: center;
  font-size: 200%;
  color: #0000CC;
}
```

*Самостійна робота № 4*

# РОЗМІЩЕННЯ МЕДІАКОНТЕНТУ НА ВЕБ-СТОРІНЦІ

***Мета:*** ознайомитися з основними засобами HTML та CSS для розміщення на веб-сторінці медіаконтенту.

## Завдання

Засобами HTML та CSS створити веб-сторінку з ім'ям **sr4.html**, де розмістити аудіо- та відеоінформацію, підібрану за тематикою того міста, для якого Вами розроблялась веб-сторінка в лабораторних роботах № 3 – 5, із засобами їх відтворення на веб-сторінці (аудіо- та медіапрогравачі).

Організувати гіперпосилання для переходу між цією веб-сторінкою та сторінками, створеними під час виконання лабораторних робіт № 3 – 5.

Надати на перевірку викладачеві файл **sr4.html**.

## Теоретичні відомості

### 1. Доцільність використання медіаконтенту на веб-сторінках

Поряд з текстом, медіаконтент є величезною частиною інтернету. За останні роки використання зображень, аудіо та відео стрімко зростає і, скоріш за все, зростатиме надалі. Використання мультимедійних даних на веб-сторінках робить їх наочними, легкими для сприйняття. Крім того, завдяки поданню інформації не у вигляді тексту, який потрібно уважно читати, а у формі аудіозаписів, відеозображень і вбудованих фреймів, можна скоротити час перегляду. Завдяки спеціальним модулям, вбудованим у браузер, аудіо- та відеофайли можуть відтворюватися прямо у його вікні.

### 2. Формати аудіо- та відеофайлів

Щоб зберегти звук чи відеоряд у файлі, його потрібно закодувати. Спосіб кодування визначається форматом файлу, а програмний модуль, який здійснює кодування (що, як правило, супроводжується стисканням) та розкодування, називають **кодеком**. Розглянемо детальніше поширені формати аудіо- та відеофайлів.

#### 2.1. Формати відеофайлів

Формат ***avi*** (від англ. *Audio Video Interleaved* – аудіо- та відеодані, що чергуються) призначений для записування звуку і рухомих зображень. *Avi*-дані можна редагувати, експортувати, стискати, використовуючи відповідні програми.

***Mpeg*** (від англ. *Moving Pictures Experts Group* – група експертів з опрацювання рухомих зображень) розробила стандарт стиснення відео- та аудіоданих. Формат має кілька версій: від *mpeg*-1 до *mpeg*-4.

Технологія ***mov*** (або QuickTime) призначена для створення, зберігання та відтворення мультимедійних даних. Вона дає змогу об'єднувати звук, текст, анімацію та відео в одному файлі.

Формат ***asf*** (від англ. *Advanced Streaming Format* – розширений формат потокових даних), розроблений корпорацією Microsoft для файлів, що містять потокове аудіо та відео (потокову технологію буде розглянуто нижче).

### 2.2. Формати аудіофайлів ogg, mp3 і wav

У сучасних умовах для записування і відтворення звуку найчастіше використовують формати *wav*, *midi*, *mp3*. Розглянемо деякі з них.

Формат ***wav*** (Windows Audio) був створений корпорацією Microsoft і прийнятий як стандарт для звукового супроводу роботи системи і комп'ютерних ігор. *Wav*-файли містять інформацію про кількість доріжок, режим (моно або стерео), швидкість запису.

Формат ***mp3*** (*mpeg*-1 Audio Layer-3) має високий ступінь стискання даних і дає змогу створювати файли невеликого розміру. Завдяки цій властивості *mp3* сьогодні є найпоширенішим форматом зберігання аудіозаписів в інтернеті.

## 3. Технології та засоби відтворення мультимедіа

Для використання звукових та мультимедійних можливостей комп'ютера слід потурбуватися про те, щоб він був обладнаний звуковою картою та колонками. Щодо самого процесу відтворення, то він має низку особливостей. Файли більшості форматів починають відтворюватись лише після завершення їх завантаження. Є й інший спосіб передавання та відтворення файлів мультимедіа – у режимі реального часу. Таку технологію називають **потоковою**. Інформацію можна отримувати безпосередньо від джерела даних, зокрема з відеокамери або файлу на сервері. Дані відтворюються в міру їхнього надходження, копія на твердому диску комп'ютера користувача не створюється. Передавання потокових мультимедійних даних схоже на трансляцію телевізійних та радіопередач: користувач може приймати одну передачу, потім переключитися на інший канал або взагалі припинити прийом.

Приймання потокових мультимедійних даних має кілька **переваг** порівняно зі звичайним завантаженням файлів з веб-сервера:

– **негайне відтворення**: у разі прийому інформації в потоковому режимі фрагмент даних відтворюється відразу після його надходження;

– **можливість передавання поточних подій**: передавання даних у потоковому режимі зручно використовувати, наприклад, для новин або репортажів зі спортивних змагань;

– **можливість передавання великих обсягів даних**: у потоковому передаванні даних не діють обмеження на розмір файлу, що передається.

Однак, у цьому разі копії мультимедійного файлу на комп'ютері створено не буде, і для його повторного відтворення слід знову зв'язуватися з відповідним веб-сервером.

Наразі поширеними є такі **потокові технології**:

– *RealAudio/Video* – це технологія потокового передавання аудіо- та відеоінформації, розроблена компанією Progressive Networks. Для відтворення даних

необхідний додатковий програмний модуль RealPlayer. Файли, призначені для оброблення засобами *RealAudio/Video*, мають розширення *.ra, .ram, .rm, .rmm, .rmd*;

– *QuickTime Streaming Server* – підтримує потокове передавання відео, аудіо, тексту та MIDI-інформації;

– *Windows Media Server* – це комплект цифрових компонентів для підтримування роботи з мультимедійними даними, що надає користувачам повний набір засобів для роботи з мультимедіа. Вони забезпечують відтворення аудіоінформації, що записана на компакт-диску, дають змогу працювати з аудіо- та відеоданими, які розташовані на веб-сервері, підтримують завантаження таких даних у потоковому режимі, надають низку додаткових можливостей. Окрім формату Windows Media, ця технологія підтримує формати *wav*, *avi*, *midi*, *mpeg*, *vod*, *aiff*, *mps*.

Однією з найпопулярніших програм відтворення мультимедійних даних є програвач Windows Media (Windows Media Player). Це універсальний програвач для прослуховування аудіо та переглядання відеофайлів більшості популярних форматів. Файли можуть міститися як на комп'ютері чи компакт-дисках, так і на веб-сторінках.

### 4. Додавання аудіо

HTML5 пропонує швидкий і простий спосіб додавати аудіофайли на сайт через елемент <audio>. Як і тег <img>, для елемента <audio> слід задати URL джерела, вказаного в атрибуті src. При цьому, на відміну від тегу <img>, тег <audio> потребує відкривний і закривний теги.

```
<audio src="jazz.ogg"></audio>
```

### 4.1. Атрибути <audio>

Декілька інших атрибутів можуть супроводжувати атрибут src для тегу <audio>, найпопулярніші це: utoplay, controls, loop і preload. Атрибути autoplay, controls і loop – логічні атрибути, які не потребують наявності значень.

За замовчуванням елемент <audio> не відображається на сторінці. Якщо є логічний атрибут autoplay, на сторінці нічого не з'явиться, але аудіофайл почне відтворюватись автоматично після завантаження сторінки.

```
<audio src="jazz.ogg" autoplay></audio>
```

Для відображення елемента <audio> на сторінці необхідний логічний атрибут controls. Коли він застосовується до елемента <audio>, браузер покаже елементи керування за замовчуванням, включаючи відтворення, паузу, регулювання гучності тощо.

```
<audio src="jazz.ogg" controls></audio>
```


За наявності логічного атрибута loop для елемента <audio> аудіофайл повторюватиметься постійно, з початку і до кінця.

Атрибут preload для елемента <audio> допомагає визначити, яка інформація про аудіофайл, якщо вона є, має бути завантаженою до відтворення кліпа. Він може набувати одне з трьох значень: none, auto і metadata.

Значення none не завантажує жодної інформації про аудіофайл, у той час як значення auto завантажить усю інформацію про аудіофайл. Значення metadata є чимось середнім між значеннями none і auto та завантажує доступні дані про аудіофайл, наприклад довжину кліпу, але не всю інформацію.

Якщо атрибута preload немає в елементі <audio>, вся інформація про аудіофайл буде завантажена, немовби значення було задано як auto. З цієї причини використання атрибута preload зі значенням metadata або none доречне, коли аудіофайл не є необхідним для сторінки. Це допоможе не завантажувати канал і дозволить сторінці завантажуватись швидше.

### 4.2. Альтернатива аудіо та декілька джерел

В наш час різні браузери підтримують різні формати аудіофайлів, трьома найпопулярнішими з яких є *ogg*, *mp3* і *wav*. Для кращої підтримки браузерами доцільно використовувати декілька аудіофайлів, які будуть включені всередині елемента <audio> … </audio>.

Для початку видалимо атрибут src з елемента <audio>, а замість нього скористаємось елементом <source> з атрибутом src, вкладеним в <audio>, щоб визначити нове джерело.

Використовуючи елемент <source> і атрибут src для кожного формату файлу, можна навести декілька аудіофайлів один за одним.

Атрибут type допоможе браузеру швидко визначити, які типи аудіо доступні. Коли браузер розпізнає формат аудіофайлу, він завантажить файл і проігнорує решту.

Хоча тег <audio> підтримується в HTML5, деякі браузери його не підтримують і для них можна запропонувати посилання для скачування аудіофайлу після будь-якого елемента <source> всередині <audio>.

```
<audio controls>
  <source src="jazz.ogg" type="audio/ogg">
  <source src="jazz.mp3" type="audio/mp3">
  <source src="jazz.wav" type="audio/wav">
  Будь-ласка, <a href="jazz.mp3" download>скачайте</a> аудіофайл.
</audio>
```

Тут елемент <audio> з логічним атрибутом controls гарантує відображення аудіоплеєра в браузерах, які підтримують цей елемент. Тег <audio> не містить атрибут src, а замість цього має три різних елементи <source>. Кожний елемент <source> містить атрибут src із зазначенням різних форматів аудіофайлу та атрибут type з визначенням формату аудіофайлу. Останній рядок наведено як резерв – якщо браузер не розпізнає жоден із форматів аудіофайлів, буде показане посилання на скачування.

Додатково до елемента <audio> HTML5 також надав елемент <video>, в якого є доволі багато схожого з <audio>.

### 5. Додавання відео

Додавання відео в HTML5 дуже схоже на додавання аудіо, використовуючи елемент <video> замість елемента <audio>. Усі ті самі атрибути (src, autoplay, controls, loop і preload) можна застосовувати і тут.

Відмінністю є використання атрибута controls. Якщо для елемента <audio> без цього атрибута аудіокліп не відображатиметься, то для відео за відсутності атрибута controls, відео буде показане. Проте, його доволі важко переглянути, якщо логічний атрибут autoplay також не застосовується. Тому зазвичай атрибут controls включають, якщо немає серйозної причини не дозволяти користувачам запускати, зупиняти або повторювати відео.

Оскільки відео займає певне місце на сторінці, доречно визначити його розміри через властивості width і height в CSS. Це гарантуватиме, що відео не виявиться занадто великим і залишиться в межах макета сторінки. Крім того, зазначення розміру допомагає браузеру відображати відео швидше і дозволяє надати доречне за розмірами місце для демонстрації відео.

```
<video src="earth.ogv" controls></video>
```

#### 5.1. Атрибут poster

Атрибут poster для елемента <video> дозволяє задати зображення, яке буде показано до відтворення відео. У наведеному нижче прикладі скріншот із відео використовується як постер для відео.

```
<video src="earth.ogv" controls poster="earth-video-screenshot.jpg"></video>
```

#### 5.2. Альтернатива відео

Як і з елементом <audio>, для відео існують альтернативні варіанти відтворення. По-перше, як і для аудіо, можна використовувати елемент <source> для кожного типу файлу. По-друге, інколи як альтернативу елемента <video> використовують звичайний текст, наприклад:

```
<video controls>
  <source src="earth.ogv" type="video/ogg">
  <source src="earth.mp4" type="video/mp4">
  Будь-ласка, <a href="earth.mp4" download>скачайте</a> це відео.
</video>
```

Ще одним альтернативним варіантом є вбудоване відео з YouTube або Vimeo. Ці сайти відеохостингу дозволяють завантажить власні відео, надають стандартний відеоплеєр і можливість вбудувати відео на сторінку за допомогою вбудованого фрейму.

### 6. Додаткові варіанти використання мультимедіа на веб-сторінках

1) Браузери можуть завантажувати та відтворювати **фоновий звук**, для прослуховування якого не потрібно виконувати жодних дій. Звук зберігається у файлі. Для вставлення фонового звуку використовують тег такого формату:

```
<bgsound src="URL звукового файлу" loop=кількість відтворень>
```

Атрибут loop може набувати значень:

– true – повторення звуку доти, доки сторінка відображається на екрані;
– false – відтворення звукового файлу один раз;
– число – кількість відтворень.

Наприклад: <bgsound src="fonzvuk.mp3" loop=3>.

2) У HTML-документах можна також використовувати **посилання на звукові та відеофайли**, які відтворюватимуться лише у разі вибору цих посилань. Окрім того, є спеціальний тег для розміщення панелі програвача на сторінці відразу після її завантаження у браузер. Однак, слід пам'ятати, що мультимедійні файли можуть бути великими за обсягом, потребувати багато часу для завантаження, тому бажано повідомляти відвідувачів про розміри аудіо- та відеозаписів, щоб вони вирішили, чи варто витрачати свій час.

Розглянемо, як розміщують посилання на аудіо- та відеофайли. Якщо, наприклад, у поточній папці є файл кліпу lesson1.avi, то посилання на нього можна задати так:

<a href="lesson1.avi"> Відеокліп з лекцією (600 Кб) </a>

Після клацання мишею гіперпосилання та надання дозволу на відкривання файлу з'явиться вікно програвача для відтворення цього відеокліпу.

3) Атрибут **dynsrc** тегу **<img>** дає змогу вбудовувати відео у такий спосіб: на веб-сторінці міститься картинка, після наведення на яку вказівника миші починається відтворення відеокліпу. Ось зразок такого тегу:

<img src="1.jpg" dynsrc="video.avi" start=mouseover 100p=7>

4) Розглянемо приклад розміщення звукового файлу audio.wav за допомогою тегу **<embed>**, який дає змогу розмістити на веб-сторінці спеціальну панель програвача мультимедійних файлів. Для цього використовують парний тег <embed src=. . .> </embed>, наприклад:

<embed src="audio.wav"></embed>

Файл audio.wav у цьому прикладі має бути збережений у поточній папці (тій самій, що й HTML-документ).

Тег <embed> може мати такі атрибути:

– src (значення – URL-адреса) – адреса кліпу;
– align (набуває значень left, right, top, middle, bottom) – вирівнювання панелі програвача щодо тексту;
– width (у пікселях) – ширина програвача;
– height (у пікселях) – висота програвача;
– autostart (набуває значень true або false) – налаштовування автоматичного запуску після завантаження;
– repeat (значення true або false) – налаштовування повторного програвання;
– play_loop – кількість повторів;
– hidden (значення true або false) – показати або приховати панель.

Приклади використання тегу <embed>:

<embed src="filename.mp3"></embed>
<embed src="filename.avi" width="300" height=«160»
autostart="true" repeat="false" align="left"></embed>

5) Інший спосіб розміщення мультимедійного об’єкта на сторінці – це застосування більш універсального тегу **<object>**.

Наприклад:

```
<object data="pryklad.mp3" type="audio/wav"></object>
```

Атрибут data задає URL-адресу відтворюваного файлу, атрибут type визначає його формат. Ще для тегу <object> можна використовувати такі атрибути:

– align – вирівнювання відносно тексту;
– width – ширина;
– height – висота;
– hspace – відступ по горизонталі;
– vspace – відступ по вертикалі.

Як і в попередньому прикладі, об’єкт можна бачити на екрані у вигляді вбудованого програвача з елементами керування.

Можна вкладати кілька елементів <object> один в одного. Це призведе до такого результату: якщо браузер має засіб для переглядання зовнішнього об’єкта, то саме він і відображатиметься, а якщо ні – браузер спробує відобразити внутрішній об’єкт і т. д. Наприклад, можна написати так:

```
<object data="1.mp4" type="video/x-mpeg">
  <object data="2.mp3" type="audio/mp3">
  </object>
</object>
```

У цьому прикладі браузер спочатку спробує відтворити відеокліп (файл у форматі *mp4*). Якщо ця спроба буде вдалою, то все, що міститься всередині зовнішнього тегу <object>, браузер зігнорує, а якщо ні – спробує відтворити файл у форматі *mp3*.

У тегу <object> можна задавати атрибут standby, значення якого (текстовий рядок) відображатиметься на екрані доти, доки не завантажиться весь об’єкт. Наприклад, доцільно написати так:

```
<object data="1.wav" type="audio/wav" standby="Йде завантаження. Зачекайте. ">
```

Якщо файл 1.wav має великий розмір, відвідувач побачить повідомлення про те, що відбувається завантаження.

### 7. Додавання вбудованих фреймів

Ще один спосіб додавання вмісту на сторінку – це вбудувати іншу HTML-сторінку на поточну за допомогою вбудованого фрейму або елемента <iframe>. Для даного елемента в атрибуті src задається URL іншої HTML-сторінки, тим самим передається вміст із вбудованої HTML-сторінки для відтворення на поточній сторінці. Отже, елемент <iframe> використовується для вбудовування медіаконтенту на сторінку із зовнішнього сайту, на кшталт Google Maps, YouTube тощо.

```
<iframe src="https://www.google.com/maps/embed?... "></iframe>
```

Елемент <iframe> містить декілька стилів за замовчуванням, у тому числі border, width і height. Ці стилі можна регулювати за допомогою HTML-атрибутів frameborder, width і height або за допомогою CSS-властивостей border, width і height.

### 7.1. Атрибут seamless

Сторінки, на які посилається атрибут src елемента <iframe>, відображатимуться за своїми правилами, оскільки вони не успадковують стилі сторінки, на яку вказують.

Змінити поведінку і налаштувати вбудовані фрейми призваний атрибут **seamless** в елементі <iframe>, який дозволяє стилям сторінок успадковуватись сторінці всередині елемента <iframe>. Крім того, атрибут seamless дозволяє посиланням, нажатим на сторінці всередині <iframe>, відкриватися у тому самому вікні, що і вихідна сторінка з елементом <iframe>.

```
<iframe src="contact.html" seamless></iframe>
```

Атрибут seamless – це новий атрибут з HTML5. Хоча підтримка браузерів для цього атрибута зростає, він не працюватиме в старих браузерах. Рекомендуємо перевірити атрибут seamless перед його використанням.

### 7.2. Семантичні визначення figure та figcaption

З HTML5 також прийшли елементи <figure> та <figcaption>, які були створені для семантичної розмітки вмісту або медіаконтенту. До HTML5 це часто організовувалось за допомогою нумерованого списку і хоча такий список працював, розмітка була семантично неправильною.

**Блоковий елемент <figure>** застосовується для ідентифікації та обгортання незалежного вмісту, часто у вигляді медіаконтенту. Він може оточувати зображення, аудіокліпи, відео, блоки коду, діаграми, рисунки або інший самостійний медіаконтент. Всередині <figure> одночасно може міститися більше одного незалежного вмісту, наприклад, декількох зображень і відео. Якщо елемент <figure> переміщується з основної частини сторінки до іншого місця (наприклад, вниз сторінки), це не має порушувати вміст або читабельність сторінки.

```
<figure>
  <img src="фото1.jpg" alt="Тематичне фото">
</figure>
```

**Елемент <figcaption>** застосовується для додавання підпису або пояснень до елемента <figure>. Елемент <figcaption> може з'явитися згори, знизу або щедесь в елементі <figure>, але тільки один раз. Під час використання елемент <figcaption> слугує заголовком для всього контенту всередині елемента <figure>.

Крім того, <figcaption> може замінити атрибут alt елемента <img>, коли вміст елемента <figcaption> пропонує корисний опис візуального вмісту зображення.

```
<figure>
  <img src="event1.jpg">
  <figcaption>Фото з місця подій.</figcaption>
</figure>
```

Не всякий тип медіаконтенту має бути включений до елемента <figure> або включати <figcaption>, а тільки той, який є незалежним і відноситься до однієї групи.

## 8. Приклад

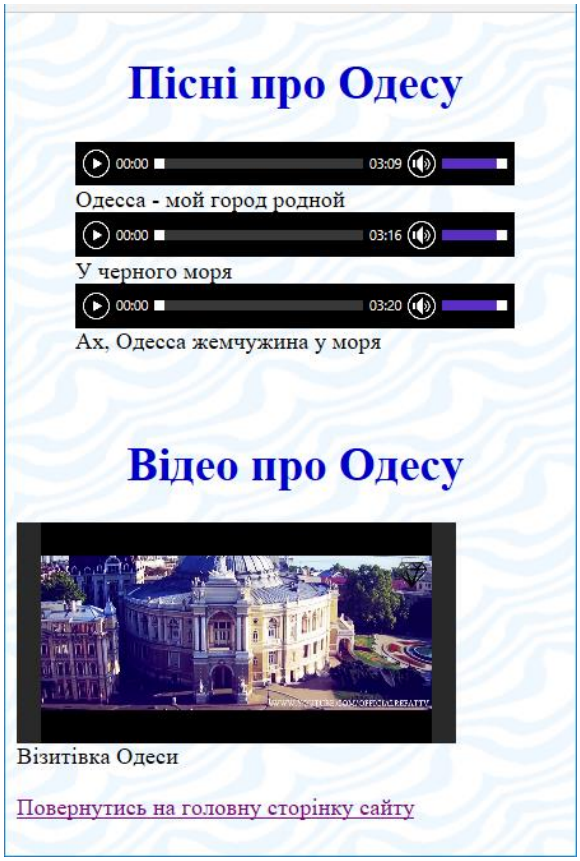
Як приклад виконання поставленого у роботі завдання створимо вебсторінку з піснями та відеовізитівкою про Одесу.

Для цього у папці поряд з іншими HTML-файлами свого сайту треба створити папку *audio*, в яку записати заздалегідь завантажені файли з піснями та відео про задане місто, віднайдені на просторах інтернету.

Далі у редакторі («Brackets», «SubLime Text 2», «Блокнот» тощо) слід створити файл з ім'ям **sr4.html**, в якому записати таке:

```
<html>
  <head>
      <meta charset="utf-8">  <!-- meta charset="windows-1251"-->
      <title>Медіаконтент про Одесу</title>
      <meta name="keywords" content="пісні про Одесу, відео про Одесу">
      <meta name="description" content="пісні та відео про Одесу">
  </head>
  <body background="photo/fon-110.gif">
    <div class="thumbs">
      <h1 aligment="center">Пісні про Одесу</h1>
         <figure>
             <audio src="audio/Odessa moy gorod rodnoy.mp3" controls></audio>
             <figcaption>Одесса – мой город родной</figcaption>
              <audio src="audio/у Черного моря.mp3" controls></audio>
             <figcaption>У черного моря</figcaption>
             <audio src="audio/ax, Одесса жемчужина у моря.mp3" controls></audio>
             <figcaption>Ах, Одесса жемчужина у моря</figcaption>
       </figure>
      <br>
      <h1 aligment="center">Відео про Одесу</h1>
      <object data="audio/aerovizitkaOdessa.mp4"
       type="video/mp4" standby="Йде завантаження. Зачекайте.">Візитівка Одеси
       </object>
      <figcaption>Візитівка Одеси</figcaption>
      <p><a href="lb1.html">Повернутись на головну сторінку сайту</a></p>
   </div>
  </body>
</html>
```

Вигляд веб-сторінки з піснями та відеовізитівкою про Одесу показано праворуч.



*Лабораторна робота № 6*

# СТВОРЕННЯ ФОРМ НА ВЕБ-СТОРІНЦІ

***Мета:*** ознайомитися з основними засобами HTML та CSS для розміщення на веб-сторінці діалогових форм.

## Навчальні питання

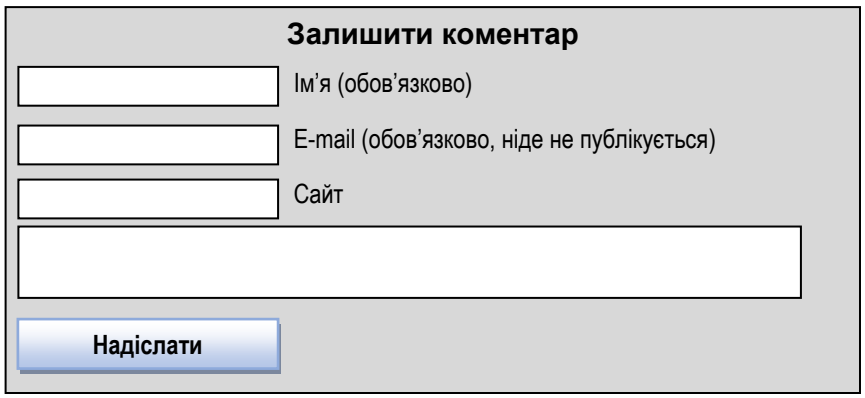
1. Засоби створення форми на веб-сторінці.
2. Теги створення текстових полів і текстових областей на формі.
3. Засоби створення на формі списків, кнопок, меню та інших елементів.

## Завдання

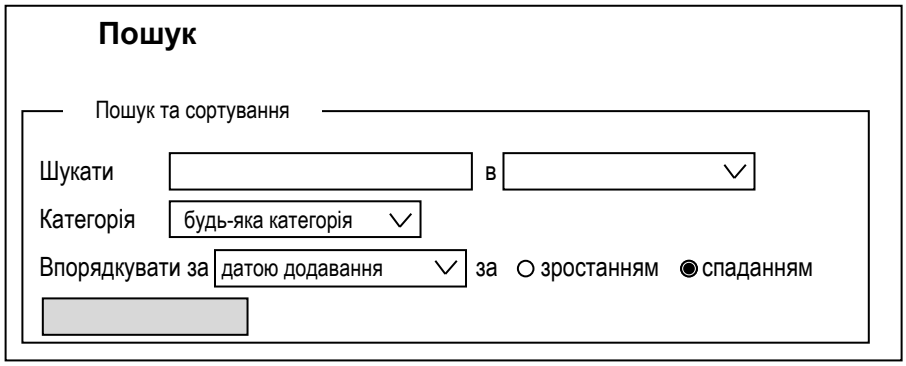
1. Ознайомитись в теоретичних відомостях до цієї роботи з основними засобами HTML та CSS для розміщення на веб-сторінці форм різної структури.
2. Засобами HTML та CSS створити веб-сторінку з ім'ям **lb6.html**, де розмістити форму, вигляд якої вибрати за індивідуальним варіантом.
3. Організувати гіперпосилання для переходу між цією веб-сторінкою та сторінками, створеними у попередніх лабораторних роботах.
4. Переглянути результат у браузері та надати його на перевірку викладачеві.
5. Підготувати відповіді на контрольні запитання.

### Індивідуальні варіанти до завдання

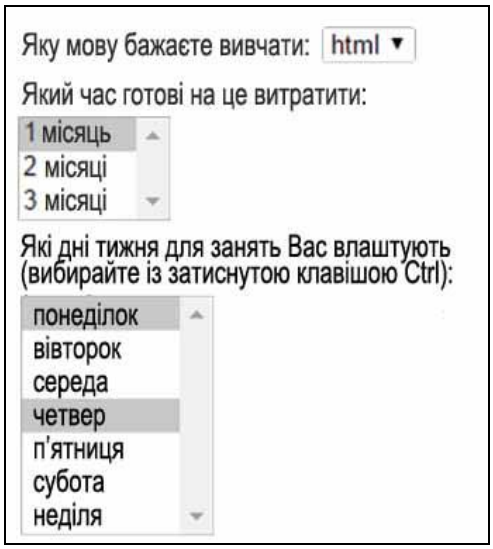
Варіант 1:



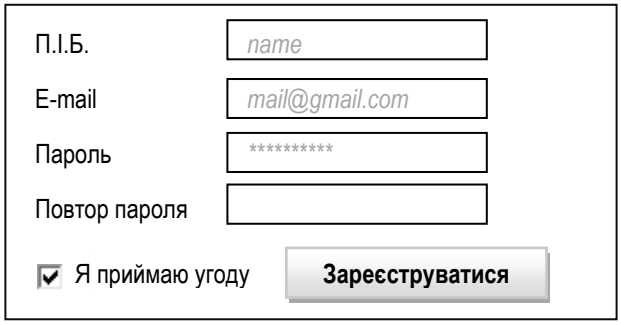

Варіант 2:



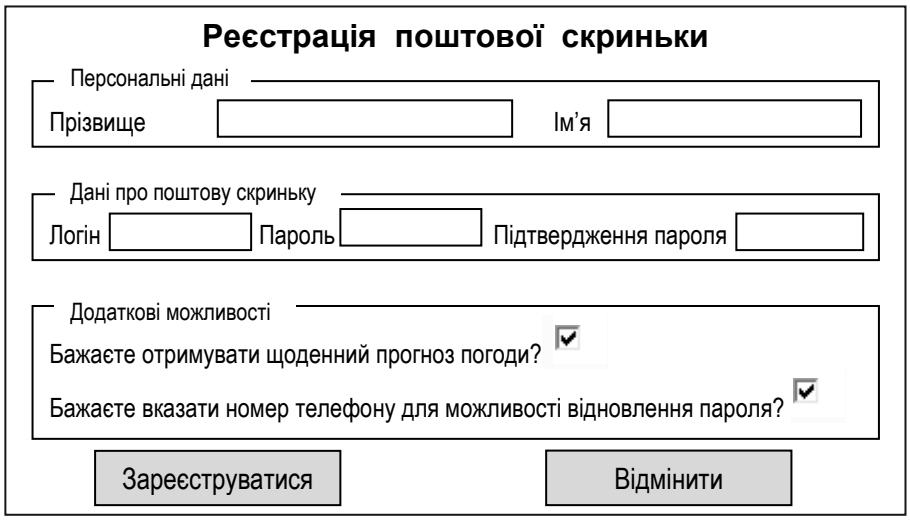
Варіант 3:



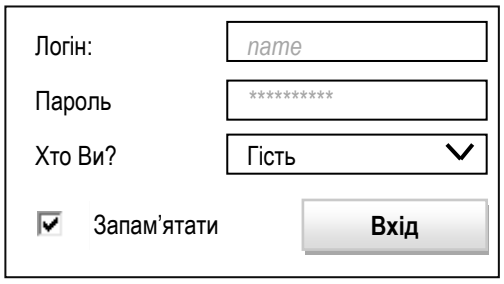

Варіант 4:



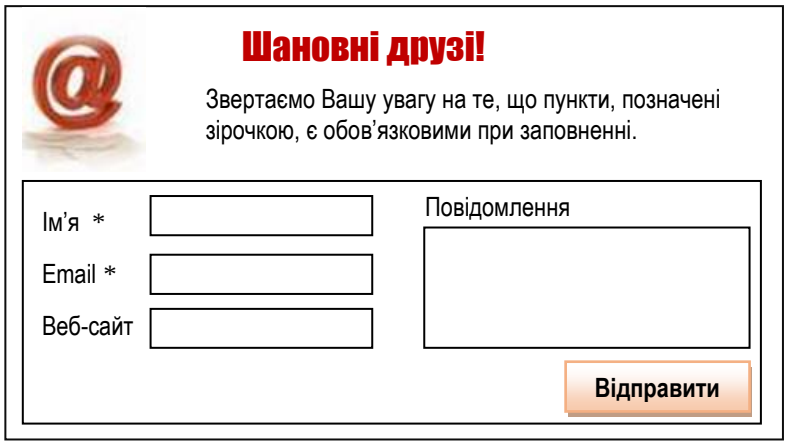

Варіант 5:



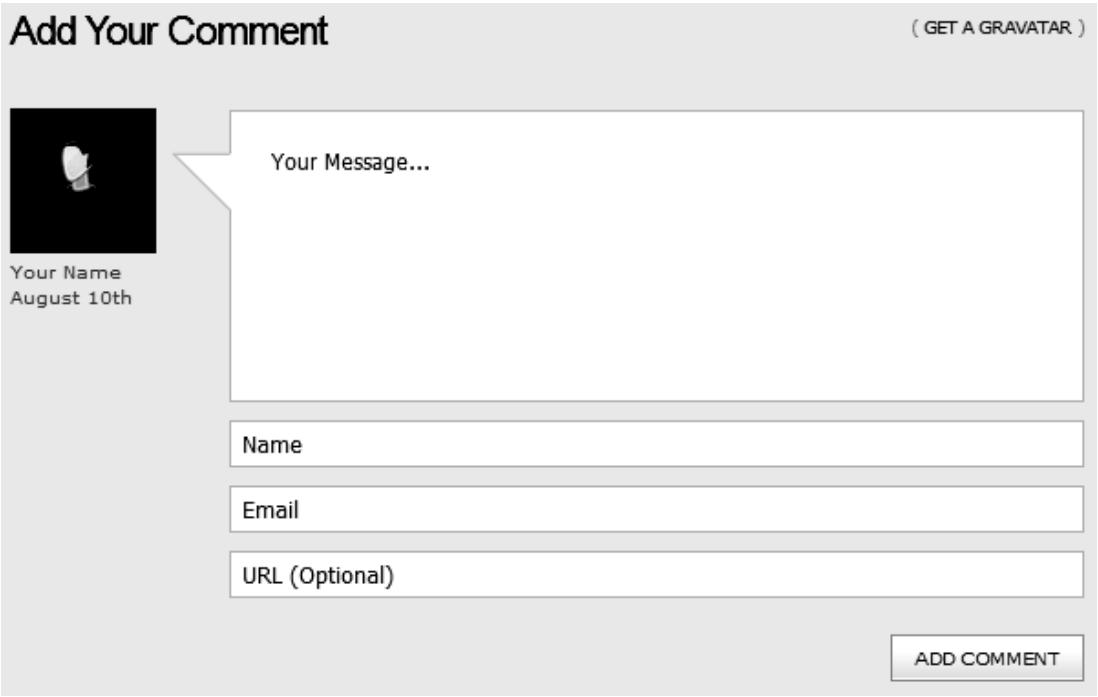
Варіант 6:




Варіант 7:

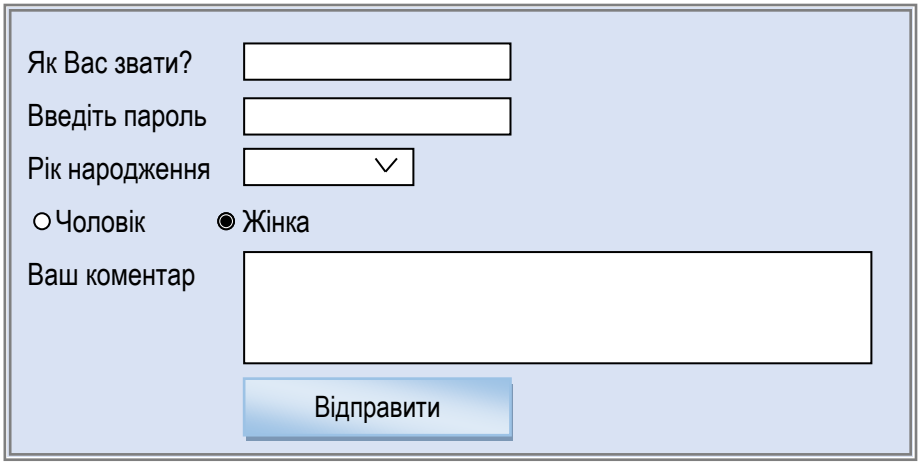



Варіант 8:



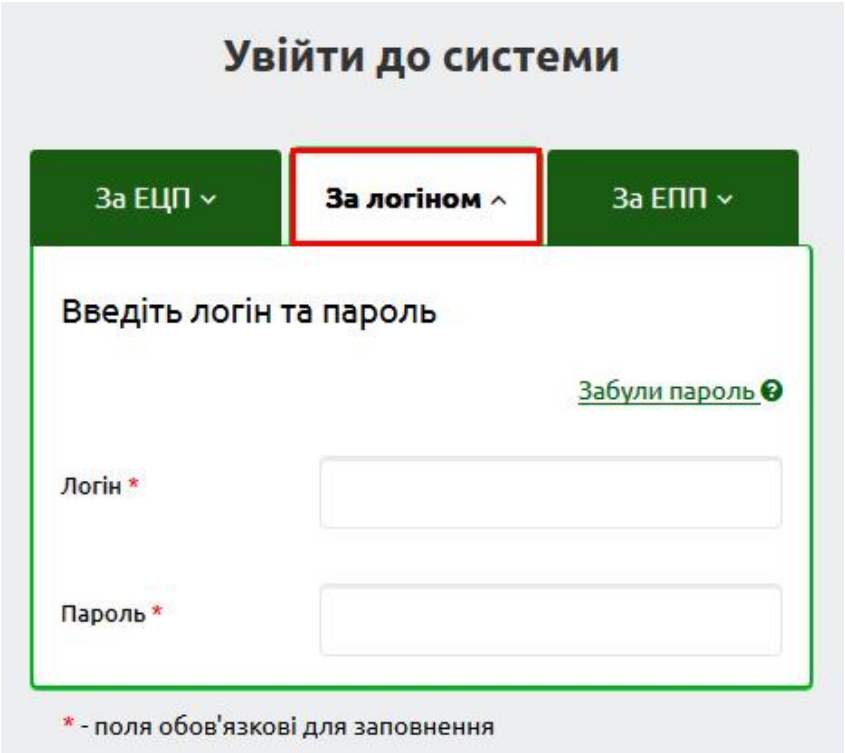
Варіант 9:


Варіант 10:

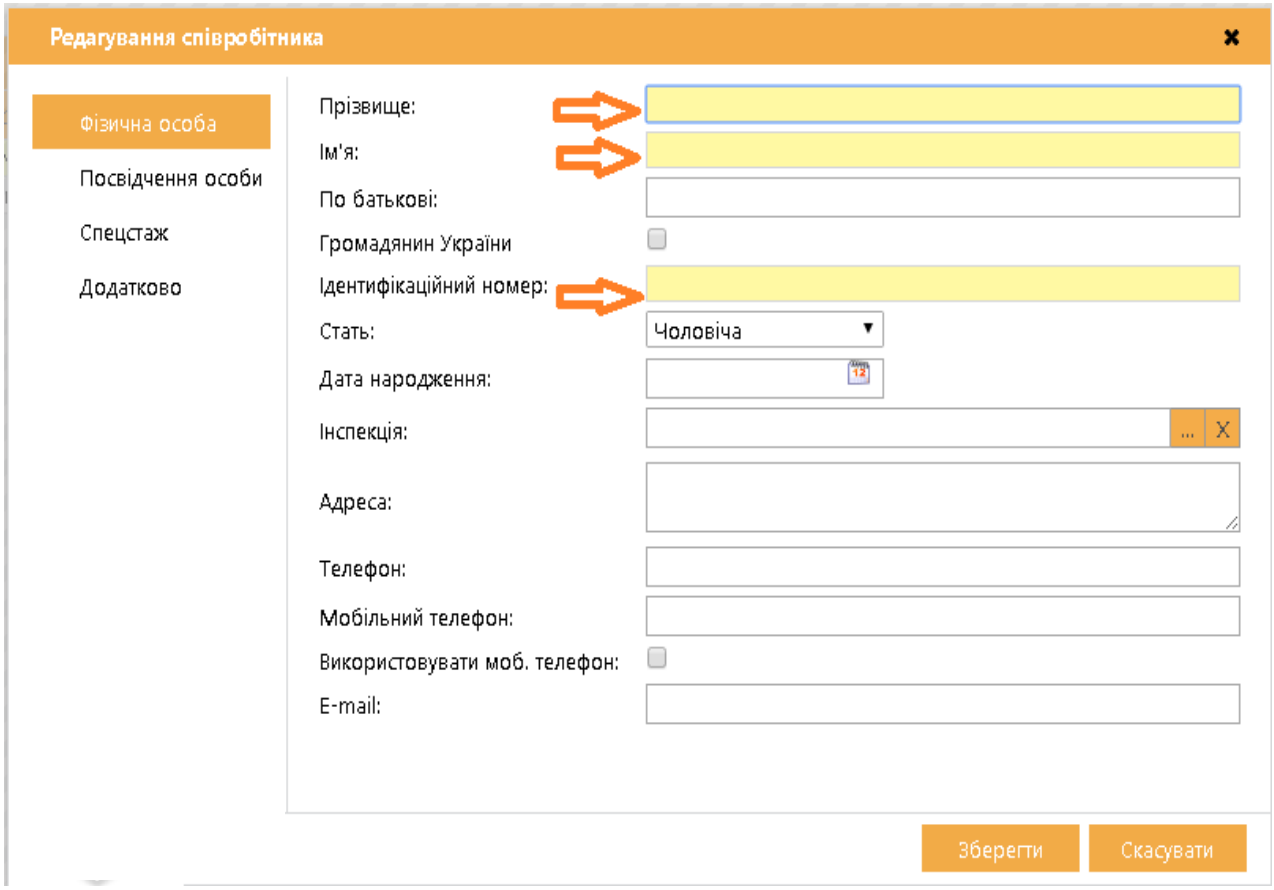



Варіант 11:

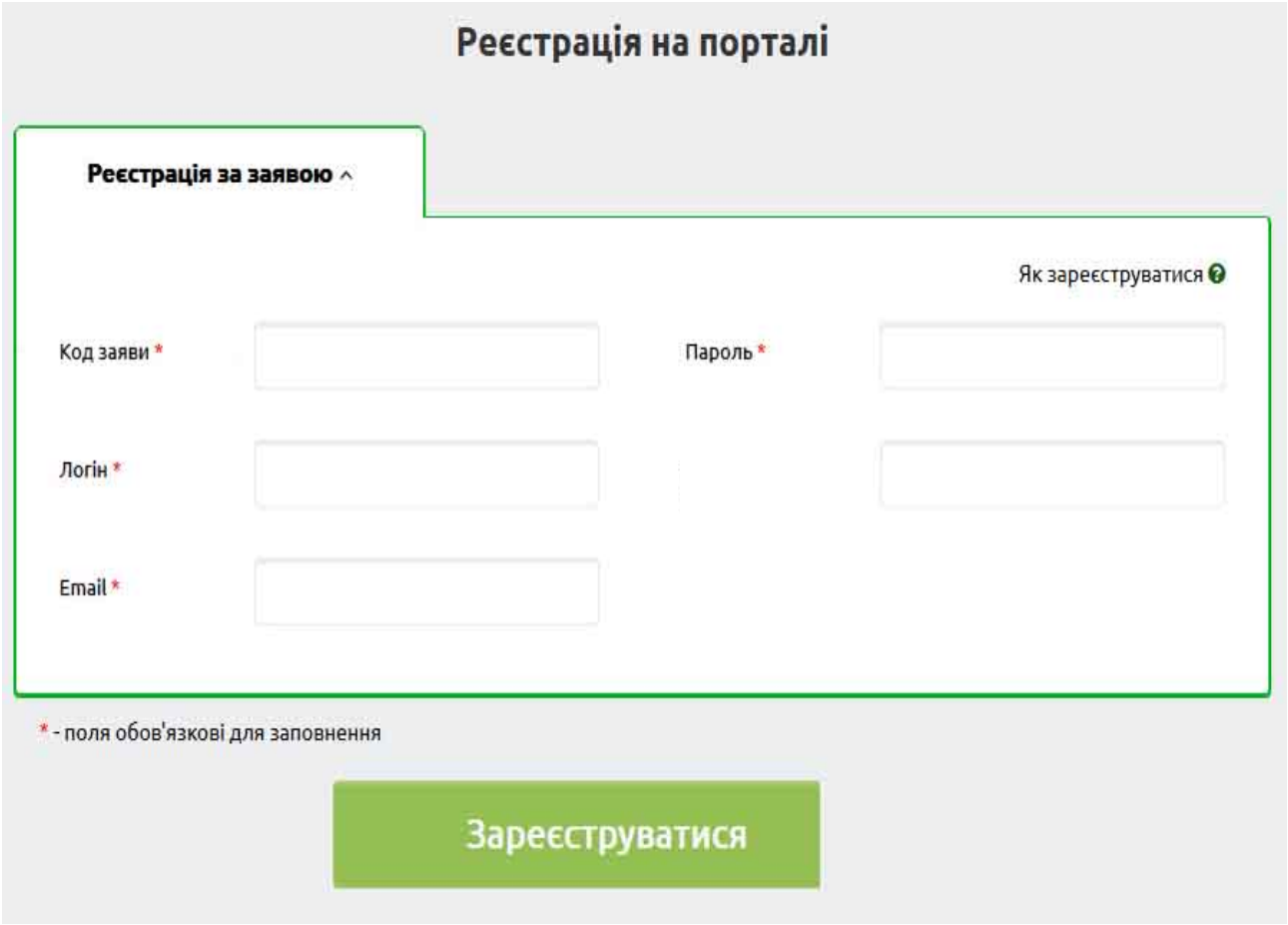
Варіант 12:



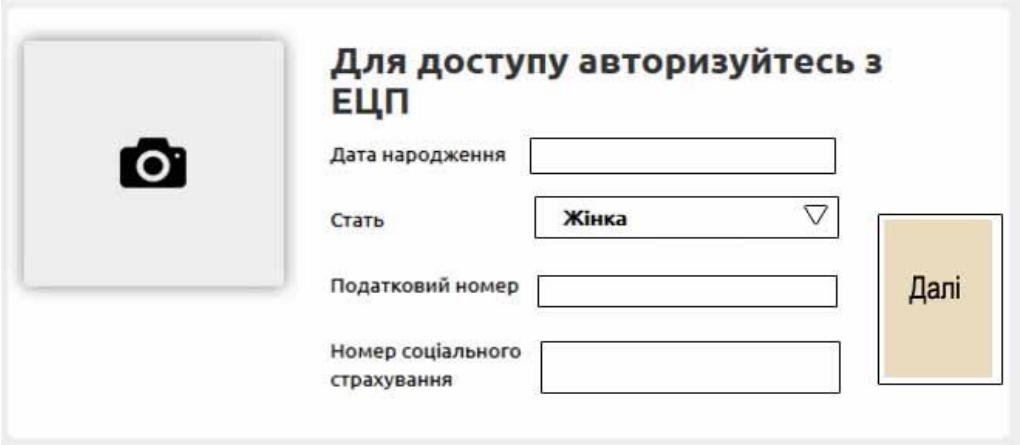

Варіант 13:



Варіант 14:




Варіант 15:




## Контрольні запитання

1. Яке призначення форм на сайтах?
2. Яке призначення тегу <form>?
3. Яке призначення тегу <input>? Який атрибут він має?
4. Навести приклади значень атрибута type для елемента <input>.
5. Який тег створює перемикач на формі?
6. Які параметри для перемикача на формі можна задавати?

7. Який тег створює розкривний список на формі?
8. Які параметри для розкривного списку на формі можна задавати?
9. Як на формі створити список для множинного вибору значень?
10. Як на формі створити елемент з прапорцями на кшталт такого: ☑ П'ятниця ☐ Субота ☐ Неділя ?
11. Який тег створює кнопку на формі?
12. Як на формі дозволити завантажувати файл?
13. Яке призначення тегу <label>?
14. Який тег групує поля форми в організовані розділи з контурною лінією?
15. Яке призначення тегу <legend>?

## Теоретичні відомості

Форми є невід'ємною частиною інтернету, оскільки вони пропонують сайтам метод збору інформації та опрацювання запитів від користувачів, а крім того, форми надають елементи керування практично для будь-якого можливого подальшого застосування. За допомогою елементів керування або полів форми можуть запитувати невеликий обсяг інформації (ім'я користувача, логін, електронну пошту, пароль тощо) або великий обсяг інформації (дані про посилку, платіжну інформацію, резюме або пропозицію роботи).

Розглянемо засоби створення форм для отримання вхідних даних від користувача. На цьому занятті ми розглянемо, як використовувати HTML для розмітки форми, які елементи використовувати для різних типів даних і як стилізувати форми за допомогою CSS. Ми не станемо занадто заглиблюватися в те, як інформація з форми обробляється на боці сервера.

### 1. Засоби створення форми на сторінці

Для створення форми на сторінці слід скористатися тегом <form>, який визначає, де на сторінці з'являться елементи керування. Крім того, <form> обгортає всі елементи форми.

```
<form action = "/ login" method = "post">
  ...
</ form>
```

Тег <form> може мати декілька атрибутів, найбільш поширеними з яких є action і method. Атрибут action містить URL, на яку інформація у формі буде відправлена для опрацювання сервером. Атрибут method є методом HTTP, який повинні використовувати браузери для відправки даних форми.

### 2. Теги створення текстових полів і текстових областей на формі

Коли справа доходить до збору текстової інформації від користувачів, є декілька різних елементів, доступних для отримання даних на формах. Зокрема, для збору даних на основі тексту застосовуються текстові поля і текстові області. Ці дані можуть включати в себе уривки тексту, паролі, номери телефонів тощо.

### 2.1. Текстове поле <input>

Одним з основних елементів, які використовуються для запиту даних у користувачів, є елемент <input>. Цей елемент має атрибут type для визначення типу інформації в елементі керування. Найпопулярніше значення атрибута type – text – означає введення одного рядка тексту.

Поряд з атрибутом type для тегу <input> також задають атрибут name. Значення атрибута name застосовується як ім'я елемента керування і відправляється разом з вхідними даними на сервер.

```
<input type = "text" name = "username">
```

Елемент <input> є самостійним, тобто він задіює тільки один тег і не обгортає контент. Значення елемента забезпечується його атрибутами та їх відповідними значеннями.

Спочатку було тільки два текстових значення атрибута type – text і password (для введення паролів), проте HTML5 дозволив кілька нових значень атрибута type. Ці значення були додані, щоб забезпечити чітке змістовне значення для полів введення, а також надати краще керування користувачам. Якщо браузер не розуміє одне з цих HTML5-значень атрибута type, він автоматично повернеться до значення text. Далі наведено список нових типів HTML5:

| | | | |
|---|---|---|---|
| color | email | range | time |
| date | month | search | url |
| datetime | number | tel | week |

Наступні приклади <input> показують використання деяких з цих значень атрибута type з HTML5, а на рисунках показано, як ці унікальні значення можуть виглядати в iOS. Зверніть увагу, що різні значення забезпечують різні елементи керування, всі вони роблять простішим процес збору даних від користувачів.

```
<input type="date" name="birthday">
<input type="time" name="game-time">
<input type="email" name="email-address">
<input type="url" name="website">
<input type="number" name="cost">
<input type="tel" name="phone-number">
```

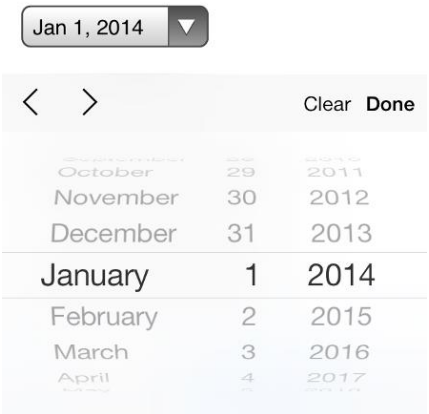



Рисунок 6.1 – Елемент <input> зі значенням date атрибута type для iOS7

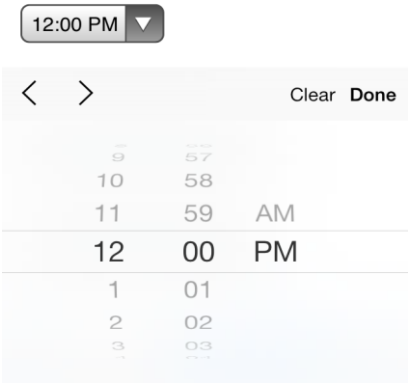



Рисунок 6.2 – Елемент <input> зі значенням time атрибута type для iOS7

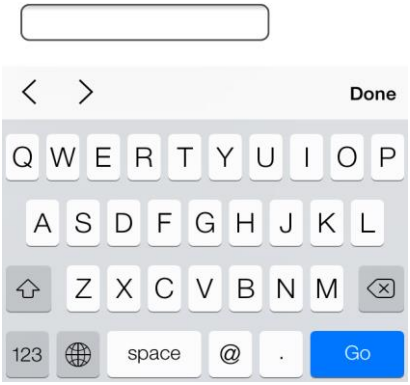



Рисунок 6.3 – Елемент <input> зі значенням email атрибута type для iOS7

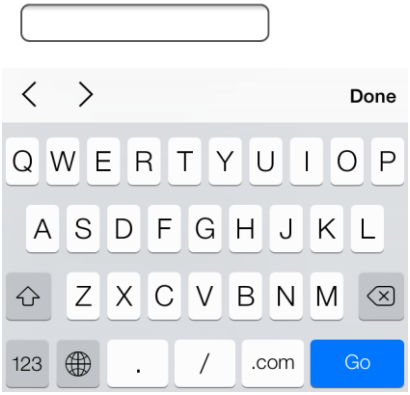



Рисунок 6.4 – Елемент <input> зі значенням url атрибута type для iOS7

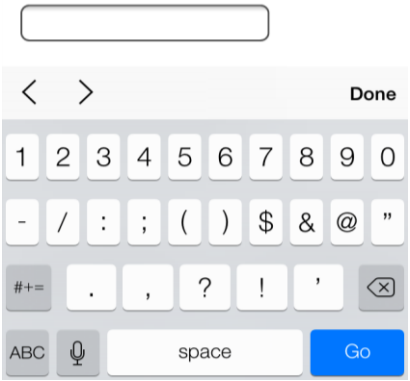



Рисунок 6.5 – Елемент <input> зі значенням number атрибута type для iOS7

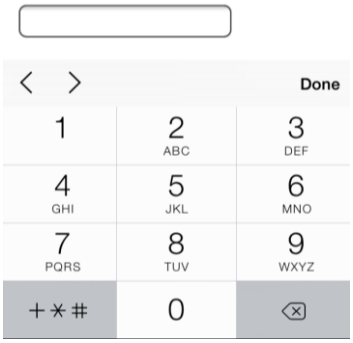
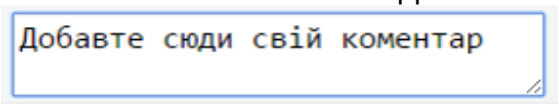



Рисунок 6.6 – Елемент <input> зі значенням tel атрибута type для iOS7

### 2.2. Тектсовий елемент <textarea>

Ще одним елементом, використовуваним для збору текстових даних, є тег <textarea>. Він відрізняється від <input> тим, що може збирати великі уривки тексту в декілька рядків. Елемент <textarea> також містить початковий і кінцевий теги, які можуть обгортати простий текст. Оскільки <textarea> бере тільки один тип значення, атрибута type тут немає, але атрибут name, як і раніше, використовується.

```
<textarea name="comment">Добавте сюди свій коментар</textarea>
```




### 3. Поля множинного вибору

Крім текстових полів, HTML дозволяє користувачам вибирати дані, використовуючи множинний вибір і розкривні списки. Є декілька різних варіантів полів для цих елементів форми, кожне з яких має свої відмінні переваги.

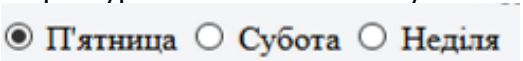
### 3.1. Перемикачі

**Перемикачі** – це простий спосіб, який дозволяє користувачам зробити швидкий вибір зі списку варіантів. Перемикачі дають користувачеві можливість вибрати тільки один варіант із запропонованих.

Щоб створити перемикач, можна використати тег <input> зі значенням radio в атрибуті type. Кожний перемикач повинен мати однакове значення атрибута name, щоб усі вони у групі були пов’язані один з одним.

Якщо у текстових полях значення є тим, що користувач в них набирає, то за допомогою перемикачів користувач робить конкретний вибір, тобто при розробці форми слід задати усі можливі значення для подальшого вибору користувачем. Для цього слід в атрибуті value записати конкретне значення для кожного елемента <input>. Крім того, задати значення перемикача за замовчуванням можна за допогоою логічного атрибута checked:

```
<input type="radio" name="day" value="Friday" checked> П’ятниця
<input type="radio" name="day" value="Saturday"> Субота
<input type="radio" name="day" value="Sunday"> Неділя
```



### 3.2. Прапорці

Прапорці дуже схожі на перемикачі. Вони використовують ті самі атрибути і шаблони, за винятком значення атрибута type. Різниця між ними полягає в тому, що прапорці дозволяють вибрати водночас кілька значень і пов'язати їх з одним ім'ям, у той час як перемикачі обмежують вибір одним значенням.

```
<input type="checkbox" name="day" value="Friday" checked> П'ятниця
<input type="checkbox" name="day" value="Saturday"> Субота
<input type="checkbox" name="day" value="Sunday"> Неділя
```

☑ П'ятниця ☐ Субота ☐ Неділя

### 3.3. Розкривні списки

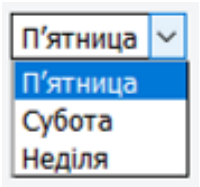
Розкривні списки надають користувачам можливість вибору потрібного значення зі списку можливих варіантів. Для створення списку застосовують елементи <select> та <option>. Елемент <select> обгортає всі пункти меню, а кожний пункт меню розмічений за допомогою елемента <option>.

Атрибут name розташовується в елементі <select>, а атрибут value розташовується в елементах <option>, вкладених в елемент <select>. Отже, атрибут value у кожному елементі <option> пов'язаний з атрибутом name елемента <select>.

Кожний елемент <option> обгортає текст (який бачить користувач) окремого пункту в списку.

Подібно до логічного атрибута checked у перемикачів і прапорців, для розкривного списку можна використовувати логічний атрибут selected, щоб попередньо виділити пункт для користувачів.

```
<select name="day">
  <option value="Friday" selected>П'ятниця</option>
  <option value="Saturday">Субота</option>
  <option value="Sunday">Неділя</option>
</select>
```

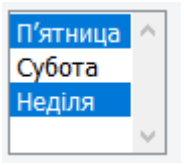



### 3.4. Список для множинного вибору

Логічний атрибут multiple в елементі <select> дозволяє користувачеві вибирати одночасно більше одного варіанта значень зі списку. Крім того, за допомогою логічного атрибута selected, доданого до більш ніж одного тегу <option>, у списку можна заздалегідь задати вибір кількох варіантів за замовчуванням.

Розмір елемента <select> можна змінювати за допомогою CSS, і він має бути скорегований відповідним чином для множинного вибору. Можливо є сенс інформувати користувачів про те, що для вибору декількох варіантів вони повинні утримувати клавішу *Shift* під час клацання.

```
<select name="day" multiple>
  <option value="Friday" selected>П'ятниця</option>
  <option value="Saturday">Субота</option>
  <option value="Sunday" selected >Неділя</option>
</select>
```



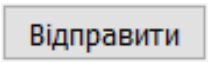
### 4. Кнопки на формі для відправлення даних

Після того, як користувач введе потрібну інформацію, натискання відповідної кнопки або поля дозволить відправити ці дані на опрацювання.

#### 4.1. Елемент <input>

Одним із засобів створення кнопки на формі є елемент <input> зі значенням submit в атрибуті type. Атрибут value задає текст на кнопці.

```
<input type="submit" name="submit" value="Відправити">
```



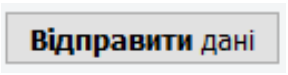

Кнопка для відправлення у вигляді елемента <input> є самодостатньою і не може обгортати інший контент. Якщо хочеться мати більше контролю над структурою та дизайном поля, поряд з можливістю обгортати інші елементи, доречно використовувати елемент <button>.

#### 4.2. Елемент <button>

Елемент <button> виконує те саме, що й елемент <input> зі значенням submit в атрибуті type. Однак, він містить відкривний і закривний теги, які можуть обгортати інші елементи. За замовчуванням елемент <button> діє зі значенням submit для атрибута type, а тому атрибут type разом зі значенням можна не вказувати.

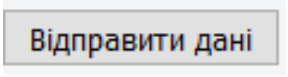
Текст на кнопці записується між відкривним і закривним тегами <button>:

```
<button name="submit">
  <strong>Відправити</strong> дані
</button>
```




або

```
<button > Відправити дані </button>
```




### 5. Інші поля форми

Крім вище розглянутих прикладів застосування, елемент <input> має кілька інших варіантів використання. Він дозволяє отримувати приховані дані та прикріплювати файли при опрацюванні форми.

#### 5.1. Приховане поле

Приховані поля дозволяють передавати дані на сервер без відображення їх користувачам. Приховані поля зазвичай використовуються для відстеження кодів, ключів або іншої інформації, яка не стосується користувача, але може бути корисною при опрацюванні форми. Ця інформація не відображається на сторінці, однак може бути віднайдена шляхом переглядання вихідного коду сторінки.

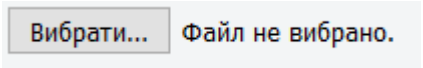
Щоб створити приховане поле, слід задати в атрибуті type значення hidden. Додатково приховане поле має відповідні значення атрибутів name і value:

```
<input type="hidden" name="tracking-code" value="abc-123">
```

#### 5.2. Поле для завантаження файлу

Щоб дозволити користувачам задавати на формі файл для завантаження, на кшталт прикріплення до листа, слід для атрибута type задати значение file.

```
<input type="file" name="file">
```



При натискані на кнопку *Вибрати* відкриється діалогове вікно для вибору файлу на ПК користувача.

На жаль, стилізація за допомогою CSS елемента <input>, для якого значення атрибута type задано як "file", є важким завданням. Різні браузери містять свої власні стилі поля за замовчуванням і жодний з них не надає можливості перевизначення цього стиля. JavaScript та інші рішення можуть бути використані для цього, але вони дещо складні для побудови.

## 6. Організація елементів форми

Знати, як отримати дані з полів форми, це лише половина справи. Інша половина – це організація елементів форми і полів у зручному порядку. При взаємодії з формами користувачі мають зрозуміти, що від них вимагається, і як надати запитувану інформацію.

Елементи <label>, <fieldset> та <legend> допомогають організовувати форми і скеровувати користувачів правильно їх заповнювати.

### 6.1. Елемент <label>

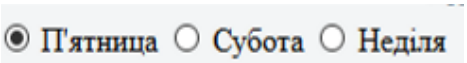
Елемент <label> використовують для надписів і заголовків на формі, створюючи тим самим зрозумлий інтерфейс форми.

<label>, крім тексту з тлумаченням вмісту полів на формі, може мати атрибут for з таким самим значеннмя, як і в атрибуті id елемента, пов'язаного з <label>. Відповідність значень атрибутів for та id пов'язує два елементи між собою, що дозволяє користувачам натиснувши на <label> передати фокус на потрібне поле форми.

```
<label for="username">Ім'я користувача</label>
<input type="text" name="username" id="username">
```


Крім того, елементом <label> можна обгорнути деякі поля форми (перемикачі або прапорці), що дозволить не вказувати явно атрибути for та id:

```
<label>
  <input type="radio" name="day" value="Friday" checked> П'ятница
</label>
<label>
  <input type="radio" name="day" value="Saturday"> Субота
</label>
<label>
  <input type="radio" name="day" value="Sunday"> Неділя
</label>
```

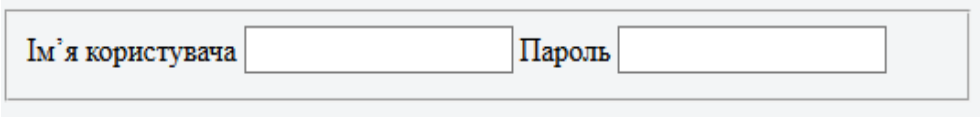



### 6.2. Елемент <fieldset>

<fieldset> групує поля форми в організовані розділи подібно <section> або іншим структурним елементам. Проте <fieldset> є блоковим елементом, який обгортає пов'язані елементи, зокрема у <form>, для їх кращої організації. Елемент

<fieldset> за замовчуванням включає в себе межі контуру, які можуть бути змінені за допомогою CSS.

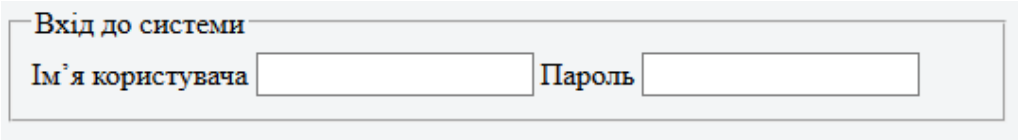
```
<fieldset>
  <label>
    Ім’я користувача
    <input type="text" name="username">
  </label>
  <label>
    Пароль
    <input type="text" name="password">
  </label>
</fieldset>
```




### 6.3. Елемент <legend>

Елемент <legend> в тегу <fieldset> формує рамку з надписом, тим самим тематично угруповує певні елементи керування на формі. Сам <legend> має розмущуватись відразу після відкривного тегу <fieldset>. Надпис рамки з’явиться у її лівому верхньому куті <fieldset>.

```
<fieldset>
  <legend>Вхід до системи</legend>
  <label>
    Ім’я користувача
    <input type="text" name="username">
  </label>
  <label>
    Пароль
    <input type="text" name="password">
  </label>
</fieldset>
```




## 7. Деякі поширені атрибути налаштування елементів на формі

Для налаштування елементів керування на формі існує ціла низка атрибутів. Розглянемо найпоширеніші з них.

### 7.1. Атрибут disabled

Логічний атрибут disabled вимикає елемент керування таким чином, що він стає недоступним для взаємодії або введення. Заблоковані елементи не будуть відправляти жодного значення на сервер для опрацювання форми.

Застосування атрибута disabled до елемента <fieldset> відключить усі елементи керування форми всередині групи.


```
<label>
   Ім'я користувача
   <input type="text" name="username" disabled>
</label>
```

### 7.2. Атрибут placeholder

Атрибут placeholder в HTML5 пропонує підказку або пораду всередині елемента <input> або <textarea>, яка зникне при клацанні по елементу керування або при наведенні фокусу. Це застосовується для поінформування користувачів, як саме слід заповнити поле форми, наприклад, вказати формат електронної пошти.


```
<label>
  Email
  <input type="email" name="email-address" placeholder="name@domain.com">
</label>
```

Основна відмінність між атрибутами placeholder і value в тому, що текст value залишається на місці, коли елемент керування отримує фокус, поки користувач не видалить його вручну. Це зручно для попередньо заповнених даних, наприклад, особистої інформації, але не для вказівок щодо заповнення.

### 7.3. Атрибут required

Логічний атрибут required в HTML5 вимагає, щоб елемент форми був заповнений значенням при відправці на сервер. Якщо елемент форми порожній, з'явиться повідомлення про помилку з проханням до користувача заповнити обов'язкове поле.

Нині стилі повідомлення про помилку контролюються браузером і не можуть бути оформлені засобами CSS. З іншого боку, некоректні елементи форми можуть бути стилізовані за допомогою псевдокласів :optional та :required.

Наприклад, елемент <input> зі значенням email в атрибуті type потребуватиме існування коректного значення у полі.


```
<label>
  Email
  <input type="email" name="email-address" required>
</label>
```

### 7.4. Додаткові атрибути

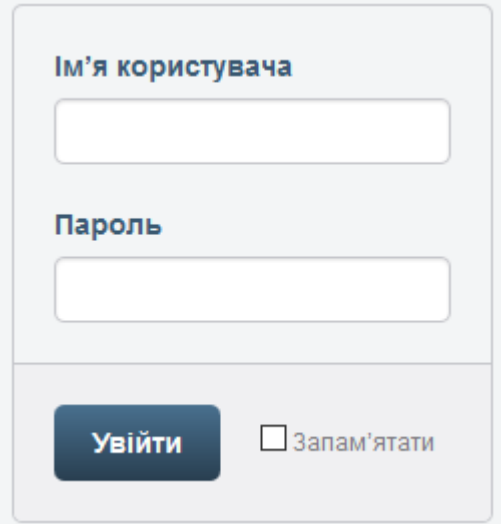
Також елементи форм можуть використовувати додаткові атрибути:

| | | | |
|---|---|---|---|
| accept | formenctype | max | readonly |
| autocomplete | formmethod | maxlength | selectionDirection |
| autofocus | formnovalidate | min | step |
| formaction | formtarget | pattern | |

## 8. Приклади форм для входу

### 8.1. Приклад 1

Розглянемо приклад форми для авторизації, яка міститиме в собі декілька різних елементів керування, тим самим проілюструємо вищеописані засоби. Стиль елементів задамо в CSS.




**HTML:**

```
<form>
  <fieldset class="account-info">
    <label>
      Ім'я користувача
      <input type="text" name="username">
    </label>
    <label>
      Пароль
      <input type="password" name="password">
    </label>
  </fieldset>
  <fieldset class="account-action">
    <input class="btn" type="submit" name="submit" value="Увійти">
    <label>
      <input type="checkbox" name="remember"> Запам'ятати
    </label>
  </fieldset>
</form>
```

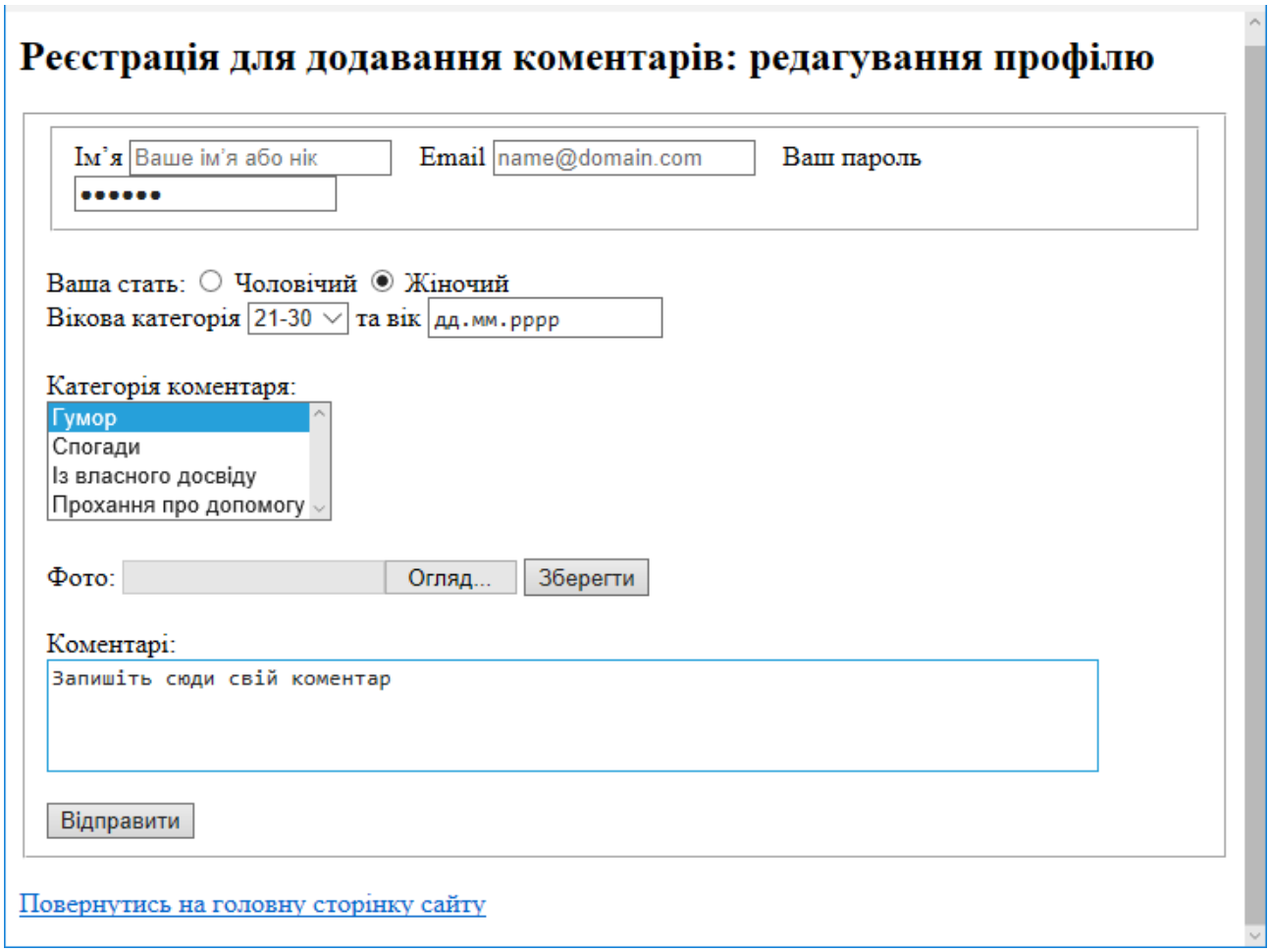
**CSS:**

```
*,
*:before,
*:after {
  box-sizing: border-box;
}
form {
  border: 1px solid #c6c7cc;
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  overflow: hidden;
  width: 240px;
}
fieldset {
  border: 0;
  margin: 0;
  padding: 0;
}
```

```
input {
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  margin: 0;
}
.account-info {
  padding: 20px 20px 0 20px;
}
.account-info label {
  color: #395870;
  display: block;
  font-weight: bold;
  margin-bottom: 20px;
}
.account-info input {
  background: #fff;
  border: 1px solid #c6c7cc;
  box-shadow: inset 0 1px 1px rgba(0, 0, 0, .1);
  color: #636466;
  padding: 6px;
  margin-top: 6px;
  width: 100%;
}
.account-action {
  background: #f0f0f2;
  border-top: 1px solid #c6c7cc;
  padding: 20px;
}
.account-action .btn {
  background: linear-gradient(#49708f, #293f50);
  border: 0;
  color: #fff;
  cursor: pointer;
  font-weight: bold;
  float: left;
  padding: 8px 16px;
}
.account-action label {
  color: #7c7c80;
  font-size: 12px;
  float: left;
  margin: 10px 0 0 20px;
}
```

### 8.2. Приклад 2

Як приклад виконання поставленого лабораторного завдання створимо веб-сторінку з формою такого вигляду:




Для цього слід створити файл з ім'ям **lb6.html**, в якому записати таке:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Профіль</title>
    <style>
       textarea {
            width: 90%;    height: 60px /* Ширина поля у відсотках  і висота у пікселях */
         }
    </style>
  </head>
  <body>
   <h1>Реєстрація для додавання коментарів: редагування профілю</h1>
   <form action="#" method="post">
    <fieldset>
      <fieldset>
```

```
        <label>  Ім’я
           <input type="text" name="name" placeholder="Ваше ім’я або нік" required>
        </label>
        <label>     Email
          <input type="email" name="email" placeholder="name@domain.com" required>
        </label>
       <label for="password-field">   Ваш пароль</label>
         <input type="password" name="password" id="password-field" value="123456">
       </fieldset>
       <br>      Ваша стать:
       <label>
         <input type="radio" name="question-one" value="m">        Чоловічий
       </label>
       <label>
         <input type="radio" name="question-one" value="w" checked>      Жіночий
       </label>
       <br>  Вікова категорія
      <select name="age">
         <option value="до 15">до 15</option>
         <option value="16-20">16-20</option>
         <option value="21-30" selected>21-30</option>
         <option value="31-40">31-40</option>
         <option value="41-50">41-50</option>
         <option value="51-60">51-60</option>
         <option value="61-70">61-70</option>
       </select>
       та вік      <input type="date" name="birthday">
       <br><br>     Категорія коментаря:   <br>
       <select name="tehn" multiple size="4">
         <option selected> Гумор</option>
         <option > Спогади</option>
         <option> Із власного досвіду</option>
         <option> Прохання про допомогу</option>
      </select>
      <br><br>   Фото:         <input type="file" value="Choose File">
      <input type="submit" value="Зберегти">        <br> <br>
     <label>  Коментарі:<br>
       <textarea name="comment">Запишіть сюди свій коментар   </textarea>
     </label>         <br><br>
     <input type="submit" name="submit" value="Відправити">
     </fieldset>
   </form>
   <p><a href="lb1.html">Повернутись на головну сторінку сайту</a></p>
  </body>
</html>
```

*Самостійна робота № 5*

# HTML-ДОКУМЕНТИ НА ОСНОВІ ФРЕЙМІВ

***Мета:*** навчитися створювати веб-сайт із фреймовою структурою.

## Навчальні питання

1. Фреймова структура: призначення та специфіка організації.
2. Області застосування фреймів.
3. Переваги і недоліки фреймової структури веб-сторінок.
4. Взаємодія між фреймами.

## Завдання

1. Ознайомитись в теоретичних відомостях з основними засобами HTML та CSS для створення веб-сторінок з фреймовою структурою.

2. Засобами HTML та CSS створити веб-сторінку з ім'ям **index.html**, структуру якої організувати за допомогою фреймів за заданою для Вашого варіанта схемою. Саме цей файл буде головним для веб-сайту.

Структура наповнення фреймів має бути такою:

- у верхньому фреймі розмістити **назву сайту** про місто, для якого Ви розробляли веб-сторінки в попередніх лабораторних роботах на кшталт такої: «… – найкраще місто в світі». Цей фрейм не буде оновлюватися;
- в одному із бокових фреймів створити список посилань з назвами вже створених Вами веб-сторінок. Цей фрейм буде своєрідною **навігаційною панеллю** і постійно відображатиметься на екрані;
- у найбільшому фреймі розмістити **основну інформацію** про місто з файлу lb3.html. Інформація в цьому фреймі оновлюватиметься залежно від вибору користувачем гіперпосилань зі списку інших Ваших веб-сторінок;
- у нижній частині екрана розташувати **«авторську» панель** з авторським знаком та Вашим прізвищем, наприклад: ©*Зайченко Маргарита*.

Для відображення вмісту у першому, другому та четвертому фреймі створити відповідні HTML-файли.

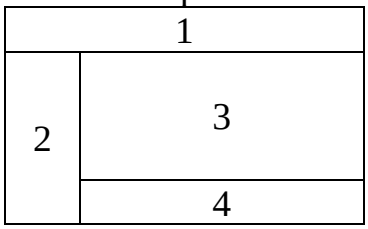
6. Переглянути результат у браузері та надати його на перевірку викладачеві.

7. Підготувати відповіді на контрольні запитання.

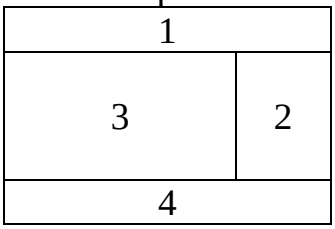
**Індивідуальні варіанти схем поділу сторінки**

(1 – назва сайту, 2 – навігаційна панель зі списком назв веб-сторінок, 3 – основна частина, 4 – дані про автора).

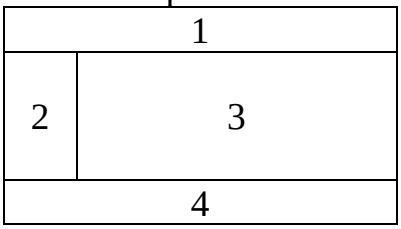
Варіант 1



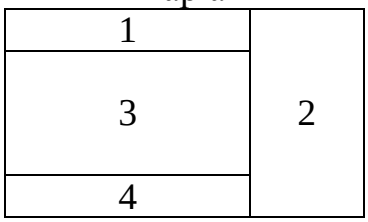

Варіант 2



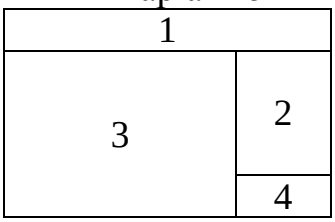

Варіант 3



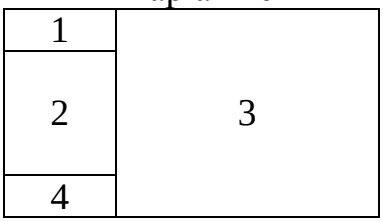
Варіант 4



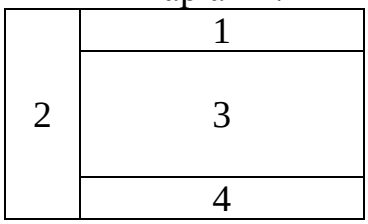

Варіант 5



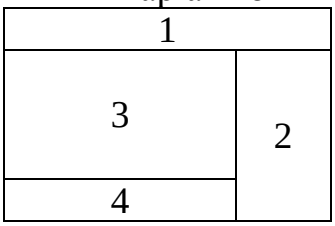

Варіант 6



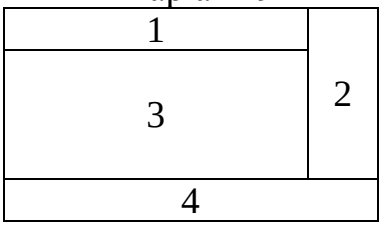

Варіант 7



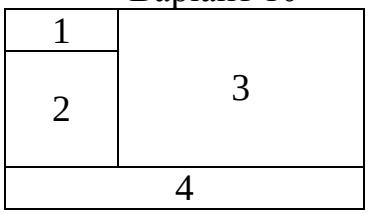

Варіант 8



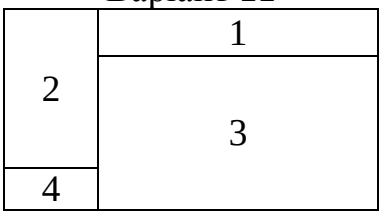

Варіант 9



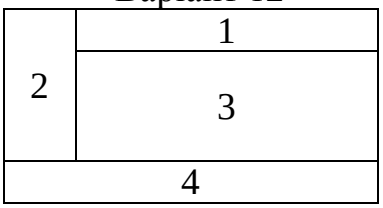

Варіант 10



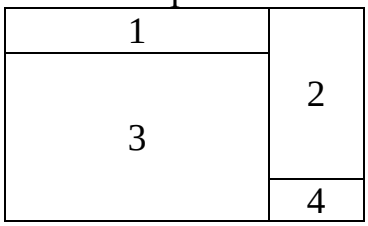

Варіант 11



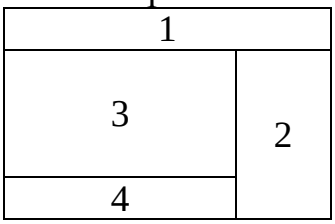

Варіант 12



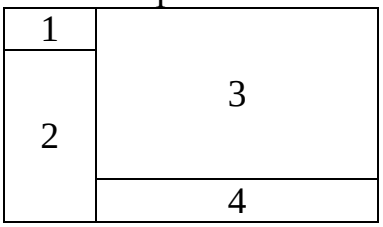

Варіант 13




Варіант 14




Варіант 15




## Контрольні запитання

1. Які засоби дозволяють організувати веб-сторінки, поділивши їх на частини для відображення в них різних блоків даних?
2. Що таке фрейм? Яке його призначення?
3. Коли доречно застосовувати фрейми?
4. Які переваги використання фреймової структури для сайтів?
5. Охарактеризувати недоліки використання фреймової структури.
6. Назвати теги, використовувані для організації фреймової структури.
7. Яке призначення тегу <frameset>? Навести приклад.
8. Яке призначення тегу <frame>? Навести приклад.
9. Охарактеризувати доступні атрибути тегу <frame>.
10. У чому відмінність застосування тегів <frame> та <frameset>?
11. Як у фрейм помістити дані з HTML-документа? Навести приклад.
12. Як надати фрейму певну назву на веб-сторінці?
13. Як організувати веб-сторінку з трьох стовпців різної ширини? Навести приклад.
14. Як задати певну висоту і ширину фрейму? Навести приклад.

15. Як організувати взаємодію між фреймами?
16. Чи можна у фрейми відразу вписувати інформацію, якщо передбачається, що інформація у фреймі не змінюватиметься? Якщо так, як це зробити, а якщо ні – то чому?

## Теоретичні відомості

### 1. Інструменти структурування даних

**Фрейм** (від англ. *frame* – рамка, каркас, кадр) – це окреме робоче вікно браузера, поділене ще на кілька різних за параметрами і розміром фреймів. Сукупність таких вікон прийнято називати фреймовою структурою.

Фреймова структура дозволяє розбивати основну область на будь-яке число складових областей (підфреймів), причому в разі потреби визначаючи внутрішню поведінку підфреймів. У кожну з таких областей завантажується самостійна веб-сторінка, яка задається за допомогою тегу **<frame>**. Отже, HTML-документ, створений на фреймовій основі, є набором взаємопов'язаних електронних документів, параметри і властивості яких визначаються налаштуваннями всієї фреймової структури. Механізм фреймів дозволяє відкривати документ в одному фреймі, за посиланням, натиснутим в абсолютно іншому фреймі. Допустимим є також використовувати вкладену структуру елементів, що дозволяє дробити фрейми на дрібніші області.

### 2. Основні теги для роботи з фреймами

#### 2.1. Тег <frameset>

Цей тег визначає структуру фреймів на веб-сторінці і по суті є контейнером для фреймів. Допустимим є використання вкладеної структури елементів, що дозволяє розбити один фрейм на декілька областей.

Особливістю веб-документа з фреймами є те, що в HTML-коді немає тегу-контейнера <body>, а тег **<frameset>** йде відразу після розділу <head>, тобто тег **<frameset>** замінює собою елемент <body> на веб-сторінці. Крім того, структурний HTML-документ (той, який визначає структуру фреймів) не може містити ані тегів форматування, ані будь-яких HTML-елементів.

**Синтаксис**:

```
<frameset>
  <frame>
</frameset>
```

**Атрибути:**

– border – товщина межі між фреймами;
– bordercolor – колір лінії межі;
– cols – задає ширину або пропорці фреймів у вигляді колонок;
– frameborder – визначає: чи відображати рамку навколо фрейму, чи ні;
– framespacing – аналог атрибута border, задає ширину межі;
– rows – задає розмір або пропорції фреймів у вигляді рядків.

Головними атрибутами є rows та cols, які задають кількість горизонтальних і вертикальних фреймів. Формат запису їхніх значень може бути у пікселях, відсотках або відносних одиницях, при чому кількість значень, записаних через кому, задає кількість фреймів.

Наприклад, тег

```
<frameset rows="30%, 70%">
```

утворить два горизонтальних фрейми, один з яких (верхній) займатиме 30% робочої області вікна браузера, а другий (нижній) – 70% (загальна сума завжди має складати 100%).

До речі, запис значень у пікселях не дуже практичний через те, що неможливо заздалегідь передбачити, на якому екрані (за розміром і за роздільною здатністю) буде переглядатися створюваний документ. У цьому сенсі оптимальнішим є зазначення відсоткового співвідношення – при зміненні розмірів вікна браузера розміри фреймів пропорційно змінюватимуться.

Розглянемо запис значення атрибута у відносних одиницях:

```
<frameset cols="*, 2*, 3*">
```

Тут символ «зірочки» (*) в якості значення атрибута cols є однією частиною цілого числа і здійснює пропорціональний поділ вікна браузера на зазначену кількість фреймів (у нашому прикладі це три вертикальних фрейми): перший фрейм займатиме 1/6 вікна, другий – 2/6 (або 1/3) вікна, а третій – 3/6 (або 1/2) вікна браузера. Відсутність числа перед символом «зірочки» інтерпретується як 1.

| | | |
|---|---|---|

Також можна для параметрів rows та cols задавати змішані значення:

```
<frameset rows="50, 50%, *, 3*">
```

Така структура має чотири горизонтальні фрейми: перший фіксований в 50 пікселів, другий – 50% від розміру вікна браузера, а два останніх фрейми поділять решту місця у співвідношенні 1/4 та 3/4.

Обов’язкового порядку для запису змішаних значень не існує, однак рекомендується найперше вказувати фіксовані значення (пікселі), тоді відсотки, а вже потім відносні одиниці.

Також у тег можна включати обидва параметри: і rows, і cols.

```
<frameset cols="70%, 30%" rows="*, 2*">
```

| | |
|---|---|
| | |

### 2.2. Тег <frame>

Тег **<frame>** визначає властивості окремого фрейму у складі фреймової структури. Цей елемент повинен розташовуватися в контейнері <frameset>, а кількість тегів <frame> має відповідає кількості організованих у <frameset> областей. У кожну з таких областей завантажується самостійна веб-сторінка, яка визначається атрибутом src. Хоча обов’язкових атрибутів у тегу <frame> немає, рекомендується задавати кожному фрейму його ім’я через атрибут name. Це є важливим, якщо треба за посиланням з одного фрейму завантажити документ в інший.

**Атрибути**:
- bordercolor – колір лінії межі;
- frameborder – ознака існування рамки навколо фрейму (так або ні);
- marginwidth і marginheight – визначення горизонтальних і вертикальних відступів всередині фрейму. Значення задаються у пікселях і є рівнозначними для обох сторін;
- name – унікальне ім'я фрейму;
- noresize – заборона можливості змінення розміру фрейму користувачем (атрибут не потребує значень). Слід пам'ятати про те, що за наявності цього атрибута змінювати розміри сусідніх фреймів теж неможливо;
- scrolling – спосіб відображення смуги прокручування у фреймі (значення yes – смуга відображатиметься завжди; значення no – смуга не з'явиться ніколи; значення auto – автоматична поява смуги прокручування, що дозволяє регулювати прокручування залежно від обсягів даних у фреймі);
- src – шлях до файлу, що завантажуватиметься у фрейм.

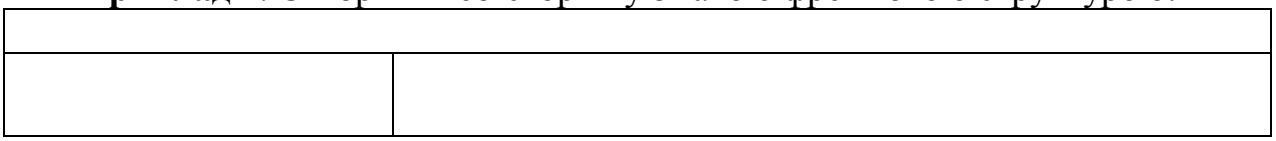
Наприклад:

```
<frame src="frames/menu.html" marginwidth="5" marginheight="3">
```

Тут задається шлях до файлу, який завантажуватиметься у фрейм, та задаються відступи: горизонтальний – 5 пікселів, вертикальний – 3 пікселі. Задавати відступи у фреймах слід обережно, пам'ятаючи про те, що такі самі (чи ще більші) відступи можуть бути задані і в HTML-коді документа вибраного фрейму (параметри leftmargin, rightmargin, topmargin, bottommargin, marginwidth та marginheight у тегу <body>).

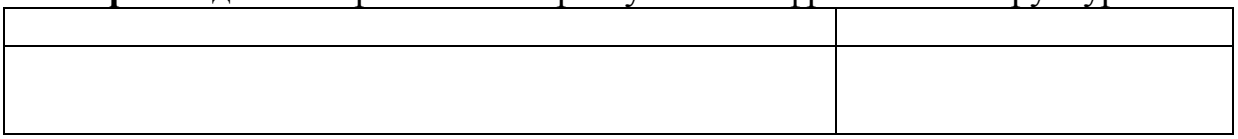
Отже, слід пам'ятати, що властивості документа, завантаженого у вікно фреймової структури, визначаються в HTML-коді саме документа, а не в межах конструкцій <frameset> або <frame>.

**Приклад 1**. Створити веб-сторінку з такою фреймовою структурою:



```
<!DOCTYPE html>
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Ter FRAME</title>
  </head>
  <frameset rows="80,* " cols="*"> <!-- заголовок сайту -->
    <frame src="nazva.html" name="topFrame" scrolling=no" noresize>
    <frameset cols="30%,*"> <!-- основна частина з двох фреймів (стовпців) -->
      <frame src="spisok.html" name="leftFrame" scrolling="no" noresize>
      <frame src="lb1.html" name="mainFrame">
    </frameset>
  </frameset>
</html>
```

**Приклад 2**. Створити веб-сторінку з такою фреймовою структурою:



```
<html>
  <head>
    <meta charset="utf-8">
    <title>Приклад фреймової структури</title>
  </head>
  <frameset cols="70%, 30%" frameborder="0" framespacing="0" border=" 0">
    <frame name="left" src="left.html" scrolling="yes" noresize marginwidth="10"
           marginheight="10">
    <frameset rows="*, 2*" frameborder="0" framespacing="0" border="0">
    <frame name="top" src="top.html">
    <frame name="bottom" src="bottom.html">
  </frameset>
  <noframes> Ваш браузер не підтримує фрейми. </noframes>
</html>
```

**Приклад 3**. Створити веб-сторінку з такою фреймовою структурою:

| Фрейм 1 | Фрейм 2 |
|---|---|
| Фрейм 3 | Фрейм 4 |
| Фрейм 5 | Фрейм 6 |

```
<frameset rows="33%,33%,*" cols="50%, 50%">
   <frame src="r1c1.html" name="Фрейм 1">
   <frame src="r1c2.html" name="Фрейм 2">
   <frame src="r2c1.html" name="Фрейм 3">
   <frame src="r2c2.html" name="Фрейм 4">
   <frame src="r3c1.html" name="Фрейм 5">
  <frame src="r3c2.html" name="Фрейм 6">
</frameset>
```

**Приклад 4**. Створити веб-сторінку з такою фреймовою структурою:

<table>
<tr><td rowspan="2">Фрейм 1</td><td>Фрейм 2</td></tr>
<tr><td>Фрейм 3</td></tr>
</table>

```
<frameset rows="*" cols="80,*">
 <frame src="frame1.html" name="Фрейм 1">
 <frameset rows="80,* ">
   <frame src="frame2.html" name="Фрейм 2">
   <frame src="frame3.html" name="Фрейм 3">
 </frameset>   </frameset>
```

### 2.3. Тег <iframe>

Тег **<iframe>** створює плаваючий фрейм всередині документа, який дозволяє завантажувати в область заданих розмірів будь-які інші документи. По суті тег <iframe> є контейнером, вміст якого ігнорується браузерами, які не підтримують даний тег. Для таких браузерів слід вказати альтернативний текст, який побачать відвідувачі. Він має розміщуватись між елементами <iframe> і </iframe>.

**Синтаксис:**

<iframe>...</iframe>

**Атрибути:**

– align визначає, як фрейм вирівнюватиметься, а також спосіб обтікання його текстом;
– allowtransparency задає прозорий фон фрейму, через який видно фон сторінки;
– frameborder – ознака наявності межі навколо фрейму;
– height – висота фрейму;
– hspace – горизонтальний відступ від фрейму до навколишнього контенту;
– marginheight – відступ згори і знизу від вмісту до межі фрейму;
– marginwidth – відступ зліва і справа від вмісту до межі фрейму;
– name – ім’я фрейму;
– sandbox задає обмеження на контент, завантажуваний у фрейм;
– scrolling – спосіб відображення смуги прокручування у фреймі;
– seamless відображає вміст фрейму так, неначе він є частиною документа;
– src – шлях до файлу, вміст якого буде завантажуватись у фрейм;
– srcdoc зберігає вміст фрейму безпосередньо в атрибуті;
– vspace – вертикальний відступ від фрейму до навколишнього контенту;
– width – ширина фрейму;

Також для цього тегу доступні універсальні атрибути і події.

**Приклад:**

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Тег IFRAME</title>
 </head>
 <body>
  <iframe src="banner.html" width="468" height="60" align="left">
   Ваш браузер не підтримує плаваючі фрейми!
  </iframe>
 </body>
</html>
```

### 2.4. Тег <noframes>

Вміст тегу **<noframes>** відображається у браузері, коли він не підтримує фрейми і не вміє їх інтерпретувати. Браузери, які працюють з фреймами, повністю ігнорують вміст тегу <noframes>. Зазвичай всередині цього тегу розташовується текст, який інформує користувача про те, що його браузер фрейми не підтримує або з пропозицією перейти на сторінку без фреймів.

```
<frameset cols="100,*">
  <frame src="menu.html"    name="leftFrame">
  <frame src="content.html" name="mainFrame">
  <noframes>Ваш браузер не підтримує фрейми.</noframes>
</frameset>
```

Цей тег ігнорується сучасними браузерами.

## 3. Області застосування фреймів

Фрейми є достатньо поширеними при організації системи навігації інтернет-ресурсу. Діапазон застосування фреймів не такий широкий, як, наприклад, у карт-зображень або меню навігації у вигляді звичайних текстових гіперпосилань. Найчастіше вдаються до фреймової структури в таких випадках:

- створення нерухомої або прокручуваної навігаційної панелі керування;
- одночасне відображення інформації у декількох місцях;
- постійна візуальна присутність певного текстового, графічного або іншого об’єкта;
- розроблення веб-інтерфейсу для онлайн-ігор.

Розглянемо докладніше кожну з названих областей застосування.

### 3.1. Панель навігації

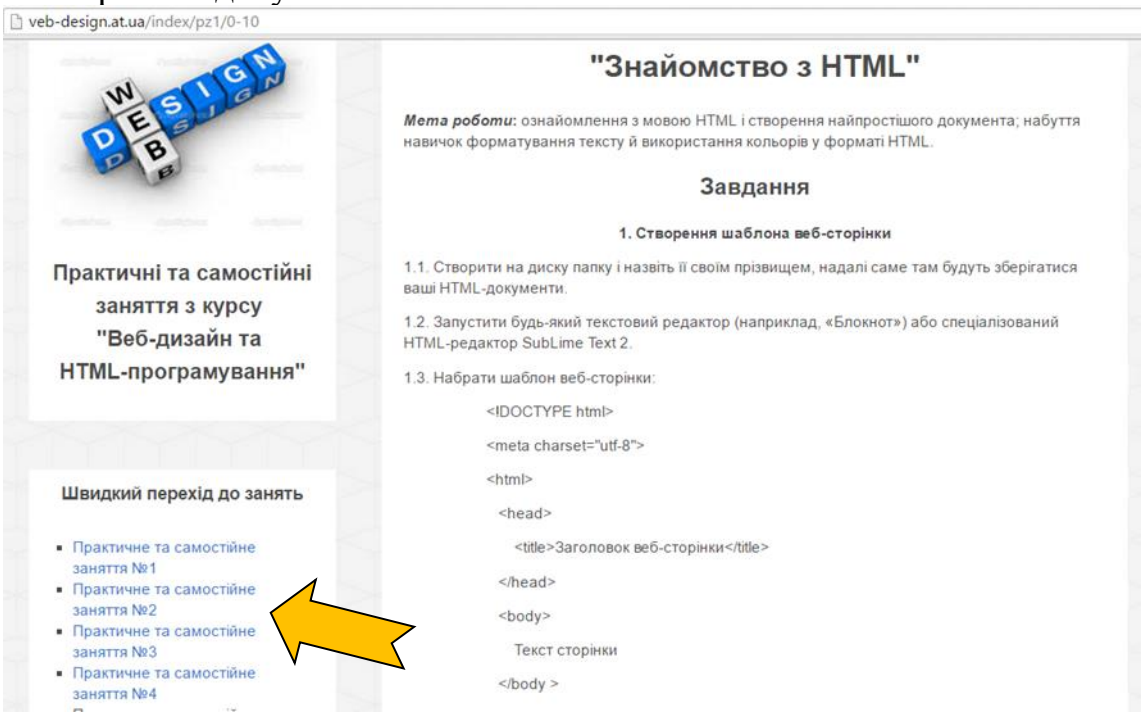
Незалежно від типу інтернет-ресурсу (корпоративний сервер, персональна сторінка, громадська організація тощо) останнім часом стало дуже популярним застосування фреймів для створення привабливих і зручних навігаційних панелей (меню). Ці панелі можуть бути статичними (тобто незалежними від дій відвідувача) і динамічними, коли будь-яка дія з боку користувача (натискання комбінації клавіш або кнопки миші чи інше) призводить до різних за видом і масштабом змін, починаючи із заміни зовнішнього вигляду навігаційних кнопок (ефект RollOver/MouseOver) і закінчуючи виконанням складних програмних сценаріїв.

Така реалізація фреймів зручна насамперед тим, що вона дозволяє відвідувачеві переміщатися по сайту і незалежно від його місця розташування (двадцять сторінок він перегорнув або всього одну) постійно мати перед очима панель керування з переліком усіх або основних розділів сайту.

Реалізувати таку панель керування можна в різний спосіб:

- у головному фреймі помістити тільки основні розділи веб-сайту, а підрозділи відкривати в іншому вікні;
- у головному фреймі розмістити посилання на документи підрозділів, тоді при натисканні на посилання конкретного підрозділу головний фрейм перезавантажиться і відкриється розширений варіант меню;

– у головному меню розташувати динамічне меню у вигляді розкривного списку, після натискання на пункти якого в іншому фреймі відкриється потрібний документ.




### 3.2. Одночасне відображення інформації

Інколи буває необхідно мати перед очима одночасно декілька текстових чи то інших інформаційних блоків, розташованих у різних вікнах, наприклад: порівняльні характеристики котирувань акцій (купівля/продаж), структуру книги (зміст, глави, розділи, підрозділи), перелік товарів та їхні різноманітні характеристики тощо.

### 3.3. Постійна візуальна присутність об’єкта

Іноді розробники HTML-документів ставлять перед собою задачу розміщення конкретного об’єкта на сторінці таким чином, щоб його було добре видно відвідувачам незалежно від їх дій (звичайно, в межах даного веб-сайту). Тобто, що б не робив відвідувач – переходив з одного розділу до іншого, залишав повідомлення в гостьовій книзі, копіював цікавий матеріал, переглядав документи тощо, – даний об’єкт у візуальному плані завжди повинен бути доступний. Такими об’єктами можуть виступати фірмові логотипи, емблеми, фотографії, рекламні навігаційні меню, написи, рубрикатори і багато іншого.

Здебільшого створюються навігаційні меню за допомогою технологій DHTML і JavaScript. Однак, не слід забувати про те, що багатьох постійне перебування на екрані реклами і меню сильно дратує, а часом заважає розглядати елементи веб-сторінки. Аналогічним варіантом реалізації ефекту постійної присутності об’єкта на сторінці є застосування багатовіконної (фреймової) структури.

### 3.4. Веб-інтерфейс для онлайн-ігор

Звичайно, фрейми застосовуються не лише на пізнавально-інформаційних сайтах і комерційних серверах. За їх допомоги можна створювати веб-інтерфейси для найрізноманітніших ігор у режимі онлайн.

### 4. Переваги і недоліки фреймів

Фреймова структура, як і будь-яке інше технологічне рішення, має своїх противників і прихильників. Комусь фрейми не до вподоби виключно з візуальних міркувань (натискаєш в одному місці, а все змінюється вже в іншому), іншим користувачам фрейми не до вподоби з технічних причин.

Розглянемо основні переваги та недоліки застосування фреймових структур. Почнемо, як водиться, з **позитивних сторін**:

- фрейми дозволяють економити на обсязі даних, які передаються користувачеві, оскільки після активізації посилання змінюється тільки один з них;
- фрейми помітно полегшують навігацію по електронних документах завдяки можливості переходу з інших посилань у межах інтернет-ресурсу;
- можливість роботи відразу з декількома інформаційними блоками у межах одного вікна дозволяє економити час;
- використання правил опису фреймових структур дозволяє розробнику HTML-документів як завгодно варіювати розміри полів фреймів, що дає більш широкий спектр можливостей для візуалізації просторових об’єктів.

А тепер декілька **недоліків** фреймів:

- деякі пошукові механізми не в змозі індексувати документи з фреймовою структурою або роблять це не зовсім коректно, що призводить до індексування не батьківського фрейму, як потрібно, а одного з його складових;
- пошукові системи погано працюють з фреймовою структурою, оскільки на сторінках, які містять контент, зазвичай немає посилань на інші документи;
- велике число фреймів вимагає для браузера виділення дещо більшої пам’яті, ніж звичайно;
- фрейми приховують адресу сторінки, на якій перебуває відвідувач, і завжди показують тільки адресу сайту. З цієї причини вподобану сторінку неможливо помістити у розділ браузера *Вибране*;
- деякі маловідомі браузери (а також ранні версії популярних) при спробі перейти назад до попереднього документа, який щойно був відображений, повертаються на початок веб-сайту. Те ж саме відбувається, якщо спробувати відновити сторінку з фреймовою структурою.

### 5. Взаємодія між фреймами

Робота з фреймовими структурами має свої особливості, які слід знати, коли є бажання їх використовувати як засіб навігації для HTML-документів.

Взаємодія між окремими фреймами відбувається за допомогою завантаження документа за відповідним посиланням у зазначене вікно. Реалізується ця дія за допомогою параметра target тегу <a>.

За замовчуванням викликаний за гіперпосиланням документ буде завантажуватися у поточний фрейм. Але часто потрібно зробити так, щоб сторінка з'являлася у сусідньому вікні. Існують спеціальні зарезервовані імена дій, за якими відбувається завантаження документів на сайтах з фреймової структурою: "_blank", "_self", "_parent" і "_top".

Дія

```
<a href="filel.html"  target="_blank">
```

здійснює завантаження документа у нове вікно без імені, що визначається параметром name тегу <frame>, тому цей варіант виключає змінення вмісту створеного вікна.

Дія

```
<a href="file2.html" target="_self">
```

відкриває документ у поточному вікні.

Дія

```
<a href="file3.html" target="_parent">
```

завантажує документ в область, яку займає батьківський фрейм поточного фрейму.

Дія

```
<a href = «file4.html» target = «_top»>
```

викликає завантаження документа на все вікно браузера. Тут не був задіяний параметр name у тегу опису поточного кадру <frame>.

Розглянемо кілька прикладів взаємодії між фреймами і відкриття окремих вікон браузера за допомогою параметра target.

Створимо файл з такою фреймовою структурою:

```
<frameset rows = "2 *, *">
<frame name="menu"  src="menu.html"  noresize  frameborder="1">
<frameset cols="50%, 50%">
<frame name="left"  src="left.html">
<frame name="right"  src="right.html">
</ frameset>
```

Верхній фрейм буде містити перелік гіперпосилань, а два нижніх фрейми призначені для відкриття в них вмісту конкретних посилань.

Документ menu.html містить список з шести гіперпосилань на один і той самий файл text.html.

**Лістинг файлу верхнього фрейму menu.html:**

```
<html>
  <body bgcolor="#ffffff"  text="black"  link="#ff0000"  alink="#00ff00"  vlink="blue">
  <h3>Посилання верхнього фрейму</h3>
  <hr>-
  <font face="Tahoma» size="2">
  <ul type="square">
   <li><a href="text.html" target="left">Файл з текстом у лівому нижньому фреймі</a>
   <li><a href="text.html" target="right">Файл з текстом у правому нижньому фреймі</a>
```

```
    <li><a  href="text.html"  target="menu">Файл з текстом у верхньому фреймі</a>
    <li><a  href="text.html"  target="_top">Файл з текстом у повному вікні</a>
    <li><a  href="text.html"  target="_blank">Файл з текстом у новому вікні</a>
    <li><a  href="text.html"  target="_self">Файл з текстом у поточному фреймі</a>
   </ul>
   </font>
   </body>
</html>
```

Перше посилання відкриється у лівому нижньому фреймі з причини зазначення конструкції target = "left" (тут "left" – це внутрішнє ім'я даного фрейму).

Друге посилання відкриється у правому нижньому фреймі (вказано внутрішнє ім'я фрейму "right").

Третє посилання буде відкрите у цьому самому вікні, оскільки "menu" – це ім'я поточного фрейму, з якого відкриваються гіперпосилання.

Четверте посилання відкриється на все вікно браузера (конструкція "_top").

П'яте посилання буде відкрите в окремому новому вікні поверх фреймової структури (дія "_blank").

Нарешті, останнє посилання відкриється в поточному фреймі (аналогічно дії target="menu").

### 6. Плаваючі фрейми

Суть плаваючих фреймів полягає в можливості вбудовувати звичайні фрейми (із зазначенням джерела), які є по суті HTML-документами, в інші електронні документи.

Плаваючі фрейми описуються тегом-контейнером і можуть мати параметри, властиві звичайним фреймам, а також параметри, аналогічні опису графічних зображень (а саме: align, width, height, hspace та vspace).

Приклад плаваючих фреймів:

```
<html>
 <head>
   <title>Приклад плаваючих фреймів</title>
 </head>
 <body bgcolor="#ffffff" text="black" link="#ff0000" alink="#00ff00" vlink="blue">
   <iframe src="iframe.html" name="iframe" width="150" height="250" hspace="5"
           vspace="5" scrolling="auto" align="right"> Ваш браузер не відображає плаваючі фрейми. </iframe>
  <p align="justify"> Плаваючі фрейми є стандартом, який підтримується браузером
     Internet Explorer. <br><br>
     Суть плаваючих фреймів полягає у можливості вбудовувати звичайні фрейми
     (із зазначенням джерела), які по суті є HTML- документами, в інші електронні
     документи.
  </p>
 </body>
</html>
```

Браузери, які не підтримують плаваючі фрейми, в зазначеному місці документа сформують повідомлення: «Ваш браузер не відображає плаваючі фрейми».

### 7. Приклад

Створимо сайт з фреймовою структурою: 1 – назва сайту; 2 – навігаційна панель зі списком назв веб-сторінок; 3 – основна частина, в яку завантажуватиметься вміст створених нами на попередніх заняттях веб-сторінок про Одесу; 4 – дані про автора.

<table>
<tr><td colspan="2">1</td></tr>
<tr><td rowspan="2">2</td><td rowspan="2">3</td></tr>
<tr><td>4</td></tr>
</table>

Після заповнення фреймів даними він набуде вигляду:


Для цього потрібно створити HTML-документ з ім'ям index.html як головний файл з фреймовою структурою, а також три HTML-файли для заповнення фреймів 1, 2 та 4: nazva.html (назва сайту), navig.html (навігаційна панель), author.html (дані про автора).

**Лістинг файлу index.html:**

```
<html>
  <head>
    <meta charset="utf-8">
    <title> Лабораторна робота № 7 Варіант 17</title>
    <link rel="stylesheet"  type="text/css"  href="lb4.css">
    <meta name="keywords" content="Одеса найкраще місто в світі">
    <meta name="description" content="Одеса найкраще місто в світі">
  </head>
  <frameset rows="130,*"  cols="*"  >        <!-- заголовок сайту -->
    <frame src="nazva.html"  name="topFrame"  scrolling="no"  noresize>
    <frameset cols="20%,*">                  <!-- основна частина з двох фреймів (стовпців) -->
      <frameset rows="90%,*"  >
        <frame src="navig.html"  name="leftFrame" scrolling="no" noresize >
        <frame src="author.html" name="leftFrame" scrolling="no" noresize>
```

```
      </frameset>
      <frame src="lb2.html" name="mainFrame">
    </frameset>
  </frameset>
</html>
```

**Лістинг верхнього фрейму nazva.html:**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Сайт про Одесу Варіант 17</title>
    <style type="text/css">
       h1 { text-align: center;    font-size: 300%;
            font-family: fantasy; color: #0000ff;
          }
      .gradient {                 /*градієнтна заливка*/
            background: -webkit-linear-gradient(top, #00c 0%,#fff 50%,#00f 100%,#fff 50%);
         }
    </style>
  </head>
  <body background="photo/fon-110.gif">
    <h1 class="gradient"><img src="photo/ship.gif" height="70px">
        ОДЕСА – найкраще місто в світі</h1>
  </body>
</html>
```

**Лістинг фрейму navig.html:**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Панель  навігації</title>
     <style type="text/css">
        p { font-size: 120%;
            margin: 20px;
            font-family: sans-serif;
            color: #0000ff;
        }
        .gradient {         /*градієнтна заливка*/
              background: -webkit-linear-gradient(left, #55c 0%,#fff 50%,#77f 100%,#fff 50%);
            }
      </style>
  </head>
  <body class="gradient">
    <p> <a href="lb2.html" target="mainFrame">Перлина у Чорного моря</a></p>
```

```
    <p> <a href="lb3.html" target="mainFrame">Довідково-статистичні відомості про Одесу
        </a> </p>
    <p> <a href="lb4.html" target="mainFrame">Видатні місця Одеси</a></p>
  </body>
</html>
```

**Лістинг фрейму author.html:**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Сайт про Одесу Варіант 17</title>
    <style type="text/css">
    </style>
  </head>
  <body background="photo/fon-110.gif">
    <p>  &#1693айченко Маргарита</p>    <!-- &#169 – код символу авторського знака -->
  </body>
</html>
```

При натисканні інших посилань на навігаційній панелі у третьому фреймі завантажуватиметься відповідний html-файл:

*Лабораторна робота № 7*

# АДАПТИВНИЙ ВЕБ-ДИЗАЙН

***Мета:*** навчитися створювати веб-сайт з адаптивною структурою.

## Навчальні питання

1. Концепція адаптивного веб-дизайну.
2. Правила розмітки флексбоксами.
3. Основні властивості flex-контейнера.
4. Основні властивості flex-елементів.

## Завдання

1. Створити папку ***lb7***, в якій створити файл ***style.css*** зі стилями флексбоксів (див. приклад 3 на с. 142).

2. Скопіювати у файл ***style.css*** стилі зображень, таблиці та інших об'єктів з веб-сторінок, створених під час виконання лабораторних робіт 3…6 та самостійної роботи № 4.

3. У папці ***lb7*** створити файл ***index.html*** з флексбоксами за схемою відповідно до варіанта (див. с. 113 – 114). Саме цей файл буде головним для веб-сайту. Структура наповнення флексбоксів має бути такою (див. приклад 3 на с. 142):
   - у верхньому флексбоксі розмістити назву сайту;
   - в одному із бокових флексбоксів створити панель навігації зі списком посилань на створені вами веб-сторінки:
     - текст посилання «Довідково-статистичні відомості про …» на файл *table.html*;
     - текст посилання «Видатні місця …» на файл *pictures.html*;
     - текст посилання «Пісні та відео про …» на файл *media.html*;
     - текст посилання «Зворотній зв'язок» на файл *form.html*;
   - у найбільшому флексбоксі розмістити основну інформацію про місто, скопіювавши її з файлу *lb3.html*;
   - у нижній частині екрана розташувати "авторську" панель з авторським знаком та Вашим прізвищем, наприклад: © *Зайченко Маргарита*.

4. Перевірити «гнучкість роботи» флексбоксів, змінюючи розміри вікна браузера.

5. Скопіювати щойно створений файл *index.html* та зберегти з ім'ям ***table.html***. Змінити в ньому вміст найбільшого флексбоксу, розмістивши таблицю з *lb4.html*. У навігаційні панелі відкоригувати посилання «Довідково-статистичні відомості про …» (файл *table.html*) на «Головна сторінка» (файл *index.html*).

6. Скопіювати щойно створений файл та зберегти з ім'ям ***pictures.html***. Змінити в ньому вміст найбільшого флексбоксу, розмістивши зображення видатних місць з *lb5.html*. За потреби додати використовувані стилі до файлу *style.css*. У навігаційні панелі відкоригувати посилання «Видатні місця …» (файл *pictures.html*) на «Довідково-статистичні відомості про …» (файл *table.html*).

7. Скопіювати щойно створений файл та зберегти з ім’ям ***form.html***. Змінити в ньому вміст найбільшого флексбоксу, розмістивши форму з *lb6.html*. У навігаційні панелі відкоригувати посилання «Зворотній зв'язок» (файл *form.html*) на «Видатні місця …» (файл *pictures.html*).

8. Перевірити у браузері працездатність панелі навігації на різних веб-сторінках сайту та «гнучкість роботи» флексбоксів, змінюючи розміри вікна браузера. Результат надати на перевірку викладачеві.

## Контрольні запитання

1) Що таке верстка сайту? У чому специфіка адаптивної верстки?
2) Що таке адаптивний веб-дизайн? Яка його мета?
3) Коли доречно застосовувати flexbox?
4) Які переваги використання flexbox для розмітки сайтів?
5) Що треба зробити, щоб задати контейнер флексбоксом?
6) Для чого задавати браузерні префікси у флексбоксах?
7) Назвати властивість, яка визначає, як елементи розташовуватимуться уздовж головної осі на поточній лінії.
8) Яке значення треба задати для властивості **justify-content**, щоб елементи рівномірно розподілялись на проміжку від початку до кінця?
9) Яке значення треба задати для властивості **justify-content**, щоб елементи рівномірно розподілялись на проміжку і утворювали рівномірні відступи між блоками та по пів відступу з країв?
10) Яка властивість визначає, як елементи розташовуватимуться уздовж переечної осі? Яке її значення центруватиме блоки по поперечній осі?
11) Яка властивість надає можливість одною властивістю описати напрямок головної і багаторядковість поперечної осі? Записати цю властивість зі значеннями, які зададуть розміщення блоків у стовпець без переносів.
12) Що задаватиме властивість flex-direction: column-reverse?
13) Що задаватиме властивість flex: 1 1 auto?
14) Яке призначення властивості **order**? Навести приклад.
15) Назвати flex властивості, які треба задавати тільки для контейнера.
16) Назвати flex властивості, які треба задавати тільки для елементів.

## Теоретичні відомості

### 1. Концепція адаптивного веб-дизайну

**Верстка сайту** – це кінцевий етап розробки сайту, створення структури сайту, яка визначатиме вигляд сайту в різних браузерах. Верстка сайту полягає у з'єднанні всіх його функціональних елементів і компонентів в єдине ціле.

Розрізняють «жорстку» (фіксовану) і адаптивну («гумову», змінювану) верстку сайту. **«Жорстка» верстка** для всіх елементів веб-сторінки задає фік-

совані розміри, незалежно від розміру екрана користувача й встановленої роздільної здатності. **Адаптивна верстка сайту** дозволяє гнучко підлаштовувати вміст веб-сторінки під різні розміри екранів.

**Адаптивний веб-дизайн** (англ. *Responsive web design*) – дизайн веб-сторінок, який забезпечує оптимальне відображення та взаємодію сайту з користувачем незалежно від роздільної здатності та формату пристрою, з якого здійснюється перегляд сторінки.

Метою адаптивного веб-дизайну є практичне відображення інформації та зручна навігація на всіх пристроях із доступом до інтернету (від стаціонарних ПК до мобільних телефонів). За технологією адаптивного веб-дизайну не потрібно створювати окремі версії веб-сайту. Один сайт може працювати на всьому спектрі пристроїв[1].

Популярність адаптивного веб-дизайну зростає з кожним днем, оскільки кількість мобільного трафіку є переважною в обсягах всього інтернет трафіку[2]. Ця тенденція настільки поширена, що Google 21 квітня 2015 року запустив у своїй пошуковій системі алгоритм оцінки сайту на відповідність принципам «дружного» до мобільних пристроїв інтерфейсу.[3] Від цього показника залежить те, як високо сторінка буде подана у результатах мобільного пошуку, а отже, ця оцінка частково діє як штраф для сайтів, які не відповідають стандартам інтерфейсу для мобільних пристроїв.

## 2. Правила розмітки флексбоксами

**Флексбокс** (від англ. **flexbox** – гнучкий блок) – це відносно новий засіб розмітки, створений для впорядкування елементів на сторінці з метою адаптивності сторінки під різні розміри екранів для різних пристроїв.

Використовуючи flexbox, контейнер і його елементи можуть бути розташовані у будь-якому напрямку: вліво, вправо і навіть вгору або вниз. Можна вибрати потрібний порядок елементів на сторінці та впорядкувати їх, вирівнювати вміст за шириною справа наліво за допомогою однієї властивості, і навіть додавати будь-яку кількість стовпців в розмітку сторінки свого сайту. Розмір блоків також є гнучким, оскільки елементи можуть збільшуватися, щоб зайняти вільний простір, або стискатися, щоб не допустити переповнення.

**Головною концепцією flexbox** є можливість змінення висоти та / або ширини його елементів, щоб найкраще заповнити простір будь-якого екрана. Така адаптація важлива при зміненні орієнтації (з вертикальної на горизонтальну чи навпаки) та розмірів екрана.

---

[1] Marcotte E. Responsive Web design. / Ethan Marcotte. – May 25, 2010. [Electronic resource]. – Mode of access: http://www.alistapart.com/articles/responsive-web-design.

[2] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper. – January 30, 2015. [Electronic resource]. – Mode of access: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[3] Official Google Webmaster Central Blog: Rolling out the mobile-friendly update. [Electronic resource]. – Mode of access: https://webmasters.googleblog.com/2015/04/rolling-out-mobile-friendly-update.html.

На сьогодні flexbox підтримується практично всіма сучасними браузерами, включаючи Android і iOS.

**Основні переваги flexbox:**

1. Створювані флексблоки (елементи) є «гумовими», тобто вони можуть стискатися і розтягуватися за заданими правилами, займаючи потрібний простір.

2. Вирівнювання по вертикалі і горизонталі щодо базової лінії тексту працює бездоганно.

3. Розташування елементів в HTML-коді не має вирішального значення. Його можна змінити засобами CSS. Це суттєво для деяких аспектів «чутливої» (responsive) верстки.

4. Елементи можуть автоматично вибудовуватися у кілька рядів чи то стовпців, займаючи весь наданий простір.

5. Безліч мов у світі використовують написання справа наліво rtl (right-to-left), на відміну від звичного нам ltr (left-to-right). Flexbox адаптований і для цього. У ньому є поняття початку і кінця, а не «права» чи «ліва». Тобто у браузерах з локаллю rtl всі елементи автоматично розташовуватимуться у реверсному порядку.

6. Синтаксис CSS-правил щодо флексбоксів доволі простий.

**Основна термінологія та правила розмітки флексбоксами:**

– Контейнер стає флексбоксом, якщо для нього задати властивість display зі значенням flex[4].

– Контейнер (flexbox) вміщує в собі елементи (flex items) (рис. 7.1).

– Головна вісь (main axis) – це вісь, уздовж якої вирівнюються елементи (може бути як горизонтальною, так і вертикальною). Властивість flex-direction встановлює main axis. Властивість justify-content визначає, як елементи розташовані вздовж головної осі на поточній лінії.

– Перехресна вісь (cross axis) (рис. 7.1) – вісь, перпендикулярна до головної осі. Властивість align-items визначає правила розташування елементів уздовж cross axis на поточній лінії. Властивість align-self задає вирівнювання елементів уздовж cross axis і змінює значення, задані в align-items.

– Головний початок / головний кінець (main start / main end**)** і перехресний початок / перехресний кінець (cross start / cross end) (рис. 7.1) – це сторони контейнера, що визначають початок і закінчення потоку елементів. Елементи (flex items) розміщуються у контейнері по головній і перехресній осях у заданому напрямі: зліва-направо (за замовчуванням), справа-наліво тощо, починаючи з боку головного початку (main-start) і до main-end.

– Перехресний початок і кінець (cross start / end): флекс лінії заповнюються items (елементами) і поміщаються у контейнер, починаючи з боку перехресного початку контейнера і до перехресного кінця.

– Властивість основного розміру (main size) – ширина або висота елемента, залежно від вибору головної осі, яка і є основним розміром елемента.

[4] Властивість flex задає гнучкість боку і вміщує в собі значення трьох властивостей: flex-grow, flex-shrink та flex-basis.

– Властивість перехресного розміру (cross size) – висота або ширина флекс елемента, залежно від вибору поперечної осі.

– Еквівалентами для розмірів (height і width) елементів є main size та cross size, які відповідно стосуються main axis і cross axis контейнера.

– Елементи можуть бути розміщені на одній чи кількох лініях відповідно до flex-wrap властивості, яка контролює напрямок перехресних ліній і напрямок, в якому складаються нові лінії.

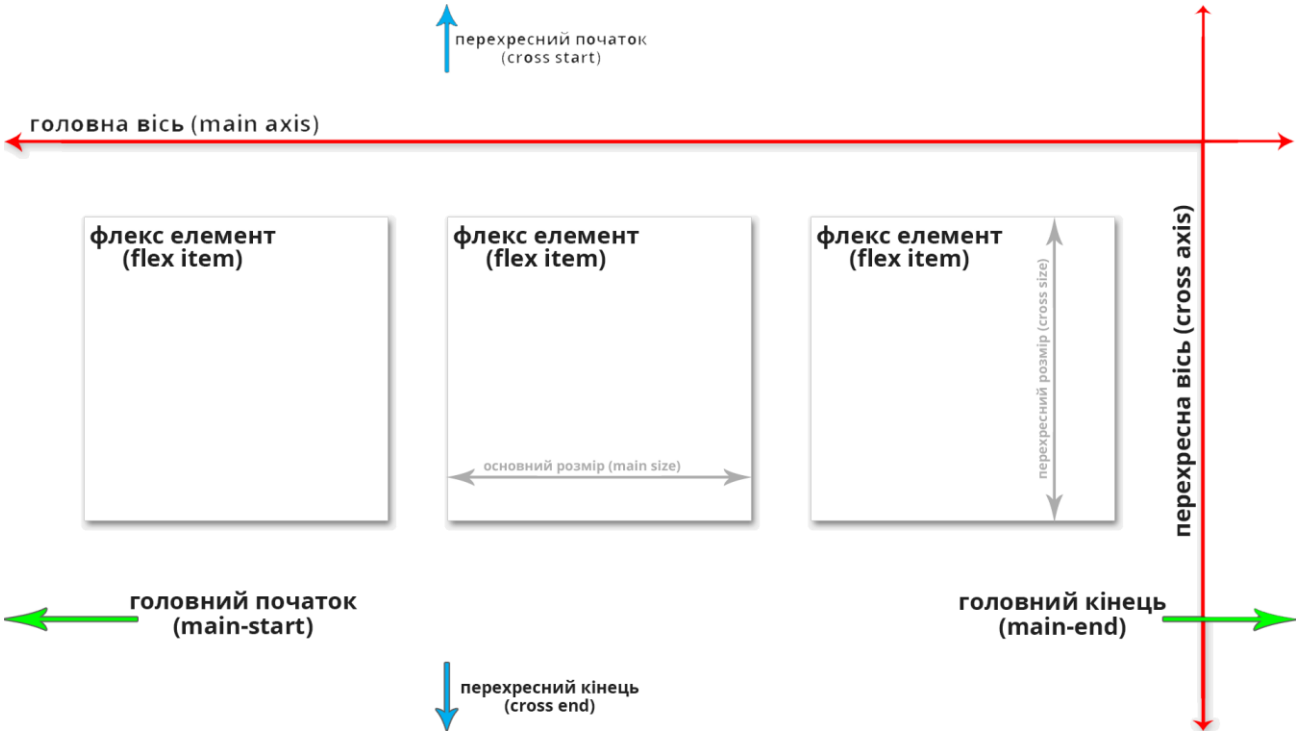



Рис. 7.1 – Графічний опис важливих концепцій для flexbox у CSS3[5]

Як приклад створимо всередині body HTML-файлу батьківський контейнер (флексбокс) класу main з трьома елементами класу blok з невеликою інформацією, і блоком з основним контентом (article) під ними:

```
<div class="main">
  <div class="blok"><p>Блок інформації №1</p></div>
  <div class="blok"><p>Блок інформації №2</p></div>
  <div class="blok"><p>Блок інформації №3</p></div>
  <article><p>Тут основний контент.</p></article>
</div>
```

Налаштування флексбоксу та його елементів задається в CSS (або в CSS-файлі, або в тегу <style></style> у розділі head).

Щоб створити flexbox, потрібно визначити для контейнера властивість display зі значенням flex. Крім того, щоб параметри flex спрацювали, слід задати розміри контейнера, наприклад, окремо ширину і висоту: width: 75%; height:

[5] Використання Flexbox в CSS3 для адаптивного дизайну. [Електронний ресурс] – Режим доступу: https://sebweo.com/vikoristannya-flexbox-v-css3-dlya-adaptivnogo-dizajnu.

450px; або хоча б один максимально допустимий розмір max-width: 800px (тоді висота залежатиме від наповнення контентом).

Наведемо вигляд початкового налаштування стилів контейнера main та його трьох елементів класу blok:

```
.main {  /* головний контейнер, в якому розміщуватимуться елементи */
   display: flex;         /* задати цей блок гнучким – флексбоксом */
   width: 75%;  height: 450px;
}
.blok {
   order: 1;
   flex: 1 1 30%;
}
article {
   order: 2;
   flex: 1 1 auto;
}
```

Щоб розібратися з використаними тут та іншими флекс властивостями, розглянемо їх детальніше.

### 3. Основні властивості flex-контейнера

**Властивість display** зі значенням flex або inline-flex задає батьківський контейнер гнучким.

```
.flex-container {
    display: -webkit-flex;    /* браузерний префікс */
    display: flex;              / * відображати контейнер як блоковий елемент * /
}
.flex-container {
    display: -webkit-inline-flex;  /* браузерний префікс */
    display: inline-flex;           / * відображати контейнер як рядковий елемент * /
}
```

Після встановлення наведених значень властивості display кожен дочірній елемент автоматично стає flex-елементом, шикуючись в ряд (уздовж головної осі) колонками однакової висоти по висоті блока-контейнера. При цьому блокові і дочірні елементи поводяться однаково, тобто ширина блоків дорівнює ширині їхнього вмісту з урахуванням внутрішніх полів і відступів елемента.

На сьогодні підтримка flexbox браузерами складає 86,3% без використання префіксів і 97,7% з ними[6]. Для підтримки flexbox у старих версіях популярних браузерів, наприклад, Internet Explorer (IE) і Safari, слід задати браузерні префікси[7]:

```
.main {
   display:-webkit-box;     /* OLD – iOS 6-, Safari 3.1-6 */
```

[6] CSS Flexible Box Layout Module. [Electronic resource]. – Mode of access: https://caniuse.com/#feat=flexbox.

[7] Using Flexbox: Mixing Old and New for the Best Browser Support. [Electronic resource]. – Mode of access: https://css-tricks.com/using-flexbox.

```
    display:-webkit-flex;     /* NEW – Chrome */
    display: -moz-box;            /* OLD – Firefox 19- (buggy but mostly works) */
    display:-ms-flexbox;     /* TWEENER – IE 10 */
    display: flex;         /* NEW, Spec – Opera 12.1, Firefox 20+ */
}
.main-content {
    -webkit-box-ordinal-group: 1;     /* OLD – iOS 6-, Safari 3.1-6 */
    -moz-box-ordinal-group: 1;  /* OLD – Firefox 19- */
    -ms-flex-order: 1;              /* TWEENER – IE 10 */
    -webkit-order: 1;               /* NEW – Chrome */
    order: 1;                   /* NEW, Spec – Opera 12.1, Firefox 20+ */
}
```

З цими та іншими браузерними префіксами докладніше можна ознайомитись за посиланнями: https://css-tricks.com/using-flexbox та http://blogo-daru.ru/2017/04/09/ prefiksy-dlya-flexbox-css-krosbrauzernyj-flexbox/. Сучасні тенденції показують на те, що потреба використання браузерних префіксів скоро мине.

**Властивість justify-content** визначає, як елементи розташовуватимуться уздовж головної осі на поточній лінії. Можливі значення цієї властивості:

- flex-start (за замовчуванням) – елементи притиснуті до початку головної осі;
- flex-end – елементи притиснуті до кінця головної осі;
- center – елементи розташовуватимуться по центру;
- space-between рівномірно розподіляє блоки на проміжку від початку до кінця;
- space-around рівномірно розподіляє блоки на проміжку, але утворює рівномірні відступи між блоками і по пів відступу з країв.

**Властивість align-items** визначає вирівнювання по вертикальній осі. Можливі значення цієї властивості:

- flex-start – блоки прижиматимуться до початку поперечної осі (догори);
- flex-end – блоки прижиматимуться до кінця поперечної осі (донизу);
- flex-center – блоки по центру поперечної осі;
- strench (за замовчуванням) – блоки розтягуватимуться, займаючи все доступне місце вздовж поперечної осі, при цьому все ж таки враховуватимуться min-width / max-width, якщо вони задані;
- baseline – блоки вирівнюватимуться за базовою лінією.

**Властивість flex-flow** поєднує в собі дві властивості flex-direction та flex-wrap. По суті, flex-flow надає можливість одною властивістю описати напрямок головної і багаторядковість поперечної осі. За замовчуванням – flex-flow: row nowrap.

**Властивість flex-wrap** зі значенням wrap дозволяє перенесення елементів на новий ряд, якщо вони не поміщаються. Значення wrap-reverse для цієї властивості задає зворотній порядок. За замовчуванням flex-wrap має значення nowrap і при цьому блоки розташовуються в одну лінію зліва направо.

**Властивість flex-direction** задає напрям головної осі. Можливі значення:

- row (значення за замовчуванням) – розміщення елементів в ряд зліва направо (в rtl справа наліво);

– row-reverse – розміщення елементів в ряд справа наліво (в rtl зліва направо);
– column змінює осі і формує відображення елементів у стовпчик зверху вниз по довжині поперечної осі;
– column-reverse – розміщення елементів знизу догори (зворотній порядок з притисканням донизу).

CSS властивості flex-direction, justify-content, align-items мають бути задані для flex-контейнера, а не для його дочірніх елементів. Наприклад:

```
.main {
  display: flex;
  flex-flow: row wrap;
  width: 75%;   height: 450px;
}
```

Наведений стиль main сформує відображення всіх елементів у контейнері в один ряд. Перенесення елементів по рядах контейнера задає властивість flex-flow зі значенням row wrap.

### 4. Основні властивості flex-елементів

**Властивість flex** з трьома значеннями (наприклад, flex: 1 1 30%; чи flex: 1 1 auto) поєднує в собі відразу три властивості flex-grow, flex-shrink і flex-basis. Для розуміння розглянемо їх детальніше:

– flex-grow – визначає, яку частину вільного простору може зайняти контейнер щодо інших контейнерів. Це може бути тільки додатне число;
– flex-shrink – властивість, яка визначає фактор стискання елемента. Значення flex-shrink (додатне число) визначає співвідношення заповнення контейнера, коли ширина елементів є ширшою, ніж сам контейнер;
– flex-basis – початковий розмір елемента (у пікселях або відсотках).

Так, властивість flex: 1 1 auto дозволить елементу стискатися і збільшуватися, підлаштовуючись під будь-який розмір екрана. Значення властивості flex: none або flex: 0 0 auto відключить гнучкість розміру і задасть розмір елемента фіксованим і нерегульованим для користувача при будь-якому розмірі екрана.

**Властивість flex-grow** визначає те, наскільки окремий елемент може бути більше сусідніх елементів, якщо це необхідно. Ця властивість показує, як захоплювати вільний простір елемента. Ціле додатне значення (від 0 – за замовчуванням і більше) показує пропорцію, в якій ділити вільний простір (якщо він є) контейнера між елементами.

Якщо всі елементи всередині контейнера мають flex-grow:1, то вони будуть однакового розміру. Якщо один з них має flex-grow:2, то він буде в 2 рази більше, ніж решта.

Якщо всі елементи всередині контейнера мають flex-grow:3, то вони будуть однакового розміру. Якщо один з них має flex-grow:12, то він буде в 4 рази більше, ніж решта.[8]

[8] Що таке Flexbox? Опис всіх css властивостей, основні принципи, переваги та недоліки. [Електроний ресурс]. – Режим доступу: http://html5.by/blog/flexbox/

Тобто абсолютне значення flex-grow не визначає точну ширину. Воно визначає його ступінь «жадібності» щодо інших елементів такого самого рівня.

**Властивість flex-shrink** – антипод flex-grow. Визначає, наскільки елемент буде зменшуватися щодо сусідніх елементів всередині контейнера в разі нестачі вільного місця. За замовчуванням має значення 1. При 0 блоки почнуть розтягуватись на всю ширину контейнера.

**Властивість flex-basis –** базовий розмір елемента по головній осі. Може задаватися у будь-яких одиницях виміру довжини (px, em, %, ...) або auto (за замовчуванням) – фіксований розмір залежно від значень властивостей width і height. Наприклад: flex-basis: 300px.

**Властивість order** повідомляє браузеру номери флекс елементів у порядку їхнього відображення. За замовчуванням дорівнює 0. До речі, для order можна задавати і від'ємні значення, наприклад, якщо потрібно додати елемент перед першим. Значення order не ставить абсолютну позицію елемента в послідовності. Воно визначає вагу позиції елемента.

**Властивість align-self** задає вирівнювання окремо взятого елемента по поперечній осі (cross axis) і перевизначає властивість align-items. Доступні значення align-self (ті самі 5 варіантів, що і для align-items: flex-start, flex-end, center, baseline, stretch (значення за замовчуванням).

**Властивість margin: auto** для елементів флексбоксу задає не вирівнювання по горизонталі, а центрує їх і по горизонталі, і по вертикалі.

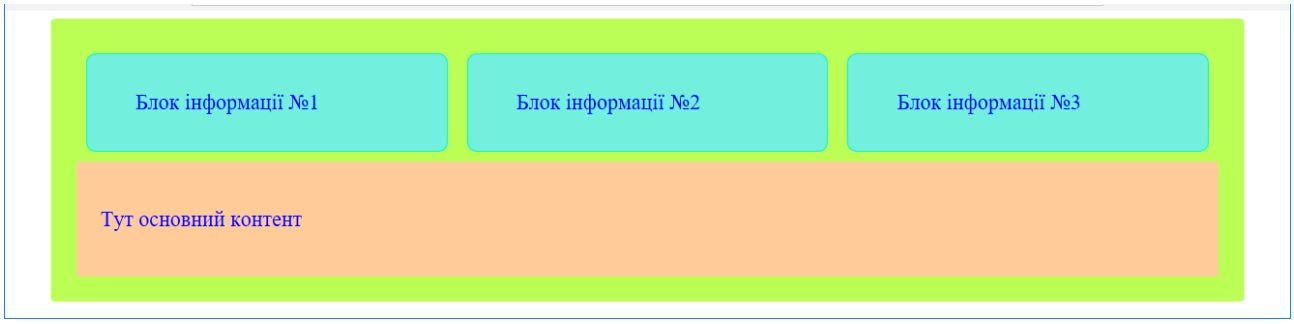
Для прикладу контейнера класу main з трьома елементами класу blok і блоком (article) під ними наприкінці п. 3 цих теоретичних відомостей створимо стилі з використанням розглянутих властивостей та різними кольорами для наочності:

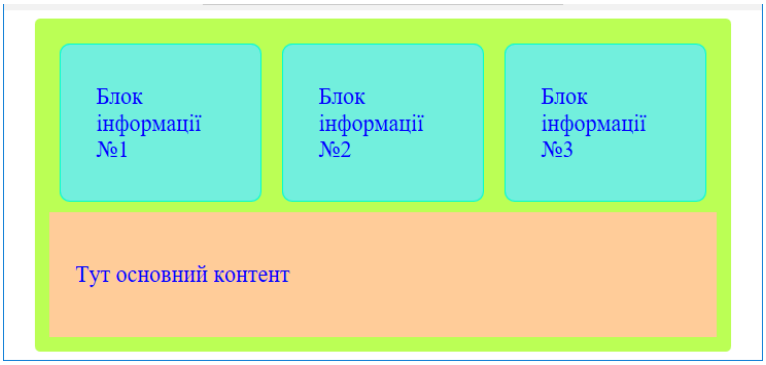
```
.main {
  display: flex;
  flex-flow: row wrap;
  max-width: 90%;              /* максимальна ширина блока */
  margin: auto;                /*   відступи всередині елемента */
  padding: 2%;                 /* відступи навколо елемента */
  background-color: #bf5;
  border-radius: 5px;          /* радіус заокруглень блока*/
}
.blok {
  order: 1;
  flex: 1 1 150px ;
  border-radius: 10px;
  border: 1px solid #0fc;      /* колір межі */
  background: #72efdd;
  padding: 3%;  margin: 10px ;
}
article {
  order: 2;
  flex: 1 1 100%;
```

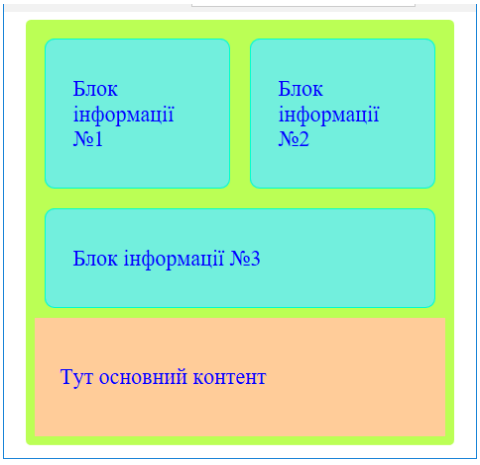
```
  background: #fc9;
  padding: 10px;
}
p {
  color: #00f;
  font-size: 22px;
  padding: 15px;
}
```

До речі, для адаптивності дизайну, краще використовувати ширину основного контейнера з використанням відсотків. Якщо задати жорсткі значення у пікселях, для маленьких екранів адаптивність не спрацює.

Далі наведено скріншоти такої веб-сторінки при зміненні розмірів вікна браузера.



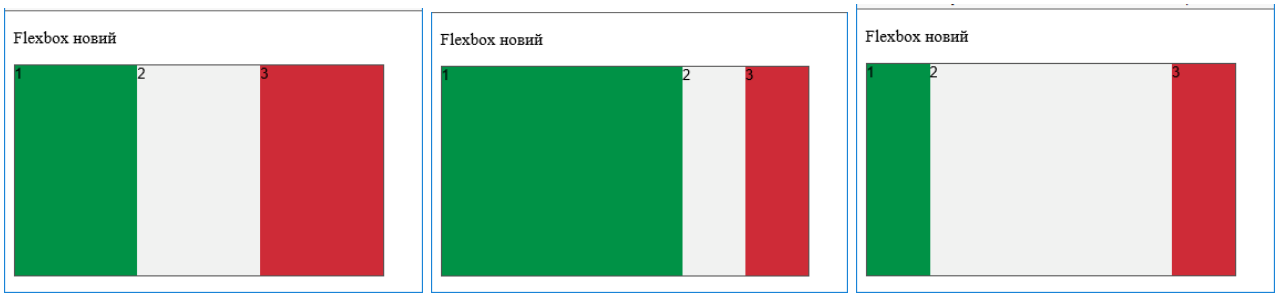




**5. Приклади**

**Приклад 1**: Створимо гнучкі елементи, динамічно змінювані при наведенні на них миші[9].

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <style>
   .flex {  /* basic styling */
     width: 350px;   height: 200px;
     border: 1px solid #555;
     font: 14px Arial;
     /* flexbox setup */
     display: -webkit-flex;
     -webkit-flex-direction: row;
     display: flex;
     flex-direction: row;
   }
   .flex > div {
    -webkit-flex: 1 1 auto;
     flex: 1 1 auto;
     width: 30px;
     -webkit-transition: width 0.7s ease-out;
     transition: width 0.7s ease-out;       /* Швидке змінення ширини за 0,7 секунди */
   }
   /* Кольори */
   .flex > div:nth-child(1) { background : #009246; }
   .flex > div:nth-child(2) { background : #F1F2F1; }
   .flex > div:nth-child(3) { background : #CE2B37; }
   .flex > div:hover   { /* При наведенні миші */
       width: 200px;           /* ширина блока збільшиться до 200 пікселів */
   }
  </style>
 </head>
 <body>
  <p> Новий flexbox </p>
  <div class="flex">
   <div>1</div>
   <div>2</div>
   <div>3</div>
  </div>
 </body>
</html>
```

[9] Використання CSS flexible-боксів. [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/uk/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes.



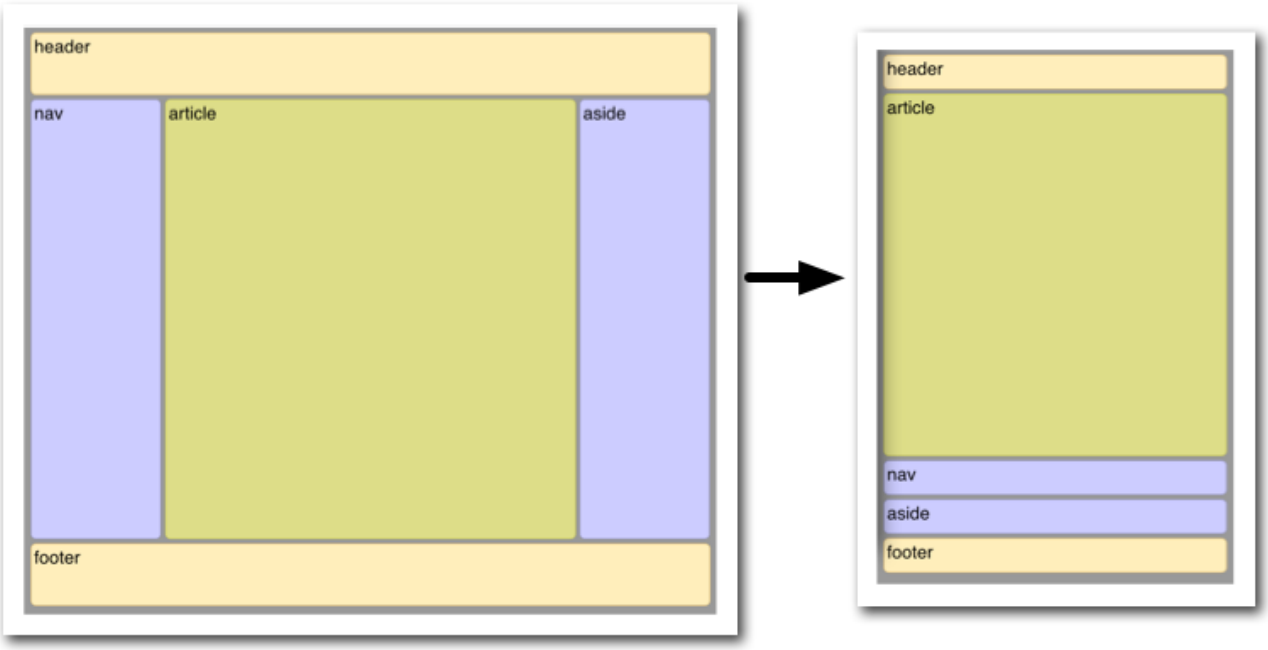

**Приклад 2:** Продемонструємо на прикладі[10], як flexbox надає можливість динамічно змінювати макет для різних розмірах екрана, тим самим оптимізувати вигляд веб-сторінки для вікна смартфону та монітору стаціонарного ПК.

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <style>
    body {
      font: 24px Helvetica;
      background: #999999;
    }
   #main {
      min-height: 800px;
      margin: 0px;
      padding: 0px;
      display: -webkit-flex;
      display:    flex;
      -webkit-flex-flow: row;
       flex-flow: row;
 }
 #main > article {
    margin: 4px;
    padding: 5px;
    border: 1px solid #cccc33;
    border-radius: 7pt;
    background: #dddd88;
    -webkit-flex: 3 1 60%;
    flex: 3 1 60%;
    -webkit-order: 2;
    order: 2;
 }
 #main > nav {
    margin: 4px;
    padding: 5px;
```

[10] Використання CSS flexible-боксів. [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/uk/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes.

```
      border: 1px solid #8888bb;
      border-radius: 7pt;
      background: #ccccff;
      -webkit-flex: 1 6 20%;
       flex: 1 6 20%;
      -webkit-order: 1;
      order: 1;
  }
 #main > aside {
      margin: 4px;
      padding: 5px;
      border: 1px solid #8888bb;
      border-radius: 7pt;
      background: #ccccff;
      -webkit-flex: 1 6 20%;
      flex: 1 6 20%;
      -webkit-order: 3;
      order: 3;
  }
 header, footer {
      display: block;
      margin: 4px;   padding: 5px;
      min-height: 100px;
      border: 1px solid #eebb55;
      border-radius: 7pt;
      background: #ffeebb;
  }
 @media all and (max-width: 640px) {
          #main, #page {
            -webkit-flex-flow: column;
            flex-direction: column;
      }
      #main > article, #main > nav, #main > aside {
            -webkit-order: 0;
            order: 0;
      }
      #main > nav, #main > aside, header, footer {
            min-height: 50px;
            max-height: 50px;
      }
   }
  </style>
</head>
```

```
<body>
<header>header</header>
<div id='main'>
  <article>article</article>
  <nav>nav</nav>
  <aside>aside</aside>
</div>
<footer>footer</footer>
 </body>
</html>
```




**Приклад 3**: Створити сайт з такою структурою: 1 – назва сайту; 2 – навігаційна панель зі списком назв веб-сторінок; 3 – основна частина, яка змінюватиметься, залежно від вибраної веб-сторінки сайту; 4 – дані про автора. Застосувати адаптивну верстку за допомогою флексбоксів.

| 1 | |
|---|---|
| 2 | 3 |
| 4 | |

Для цього треба створити HTML-файл з ім'ям *index.html* як головну веб-сторінку сайту. У цій веб-сторінці засобами флексбоксів слід утворити задану структуру, а стилі цих флексбоксів описати у файлі ***style.scc***. Готову веб-сторінку треба скопіювати чотири рази і зберегти копії з іменами: ***table.html***, ***pictures.html***, ***media.html*** та ***form.html***. Далі змінити в цих файлах вміст найбільшого флексбоксу на відповідні фрагменти HTML-файлів веб-сторінок, розроблених під час виконання лабораторних робіт 3…6 та самостійної роботи № 4. У навігаційні панелі цих файлів відкоригувати посилання. За потреби внести коригування стилів.

Наведемо можливий вміст усіх файлів веб-сторінок сайту.

**Лістинг файлу *style.scc***

```
body {
	margin: 0;
	padding: 0;
	font:1em/1.3em Arial, Tahoma, "Trebuchet MS", sans-serif;
}
.main { /*флексбокс, контейнер*/
	border: 1px solid #ccc;
	padding: 10px;
	border-radius: 3px;
	display:-webkit-box;
	display:-webkit-flex;
	display:-ms-flexbox;
	display: flex;
	max-width:1600px;
	margin: 20px auto;
}
.bl { /*флекс елементи (поля)*/
	padding: 5px 10px;
	background: #72efdd;
	border-radius: 5px;
	margin: 10px;
	flex: 1 2 1000px;
}
.fl1 {
	font-size: 12px;
	flex: 1 3 80px;
 }
.fl2 {
	border:1px solid #00d;
	background: #9feeff;
	font-size: 14px;
	flex: 5 1 900px;
}
.fl3 {
	font-size: 12px;
}
h1 {
	text-align: center;
	font-size: 200%;
	font-family: fantasy;
	color: #0000ff;
}
h2 {
	font-family: fantasy;
```

```
	font-size: 180%;
	color: #333;
	background-color: #fff;
}
.gradient {		/*градієнтна заливка*/
	background: -webkit-linear-gradient(top, #00c 0%,#fff 50%,#00f 100%,#fff 50%);
}
.note {
	padding-left: 15px;
	font-size: 90%;	color: white;
	margin: 10px 0;
	padding: 10px 10px; /*відступи*/
	border-left-width: 10px;
	background-color: #999999;
	width: 50%;
}
img {
	height: 200px;
	margin-left: 10px;
	vertical-align: middle;
}
.imgship { height: 70px;
}
table {
	border: 2px solid gray;	border-collapse: collapse;
	background-color: #F0F0FF; /*світло-блакитний*/
}
td {	/*стиль для клітинки таблиці*/
	border: 1px solid  gray;
	padding: 5px;
}
#tabl {
	float: right;	width: 30%;
	padding-left: 15px;
	padding-top: 40px;
}
```

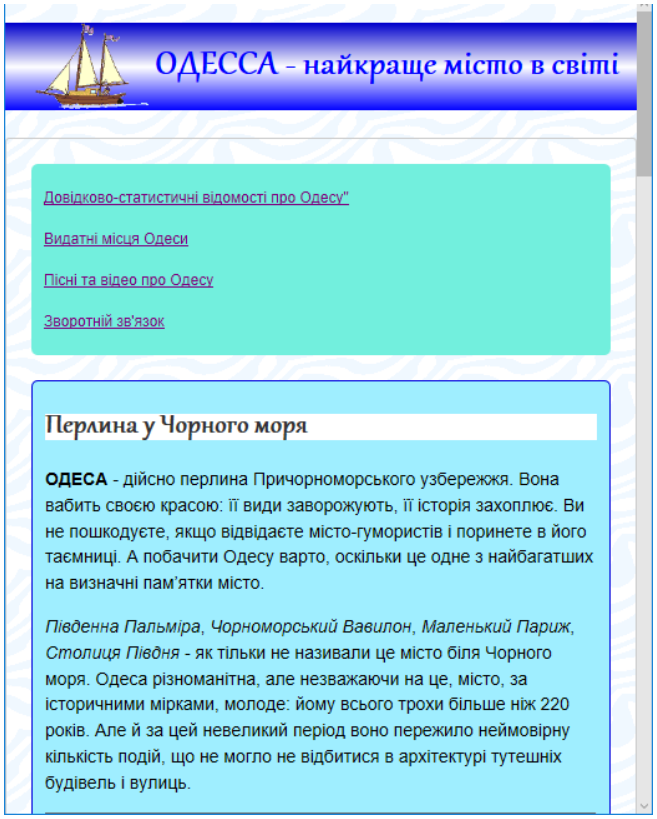
**Лістинг файлу *index.html***

```
<html>
  <head>
	<title>Flexbox на практиці</title>
	<meta charset="utf-8">
	<link rel="stylesheet" type="text/css" href="style.css">
	<meta name="keywords" content="Одеса найкраще місто в світі">
	<meta name="description" content="Одеса найкраще місто в світі">
  </head>
```

```
<body background="Photo/fon-110.gif" >
    <h1 class="gradient">
    <img src="Photo/ship.gif"  class="imgship">
    ОДЕССА - найкраще місто в світі
    </h1>
    <div class="main">
     <div class="bl fl1">
          <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
          Одесу"</a></p>
          <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
          <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
          <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
     </div>
     <div class="bl fl2">
          <h2>Перлина у Чорного моря</h2>
          <p><strong>ОДЕСА</strong> – дійсно перлина Причорноморського узбережжя. Во-
          на вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошкодує-
          те, якщо відвідаєте місто-гумористів і поринете в його таємниці. А побачити Одесу
          варто, оскільки це одне з найбагатших на визначні пам'ятки місто.</p>
          <p><em>Південна Пальміра</em>, <em>Чорноморський Вавилон</em>,
          <em>Маленький Париж</em>, <em>Столиця Півдня</em> – як тільки не називали
          це місто біля Чорного моря. Одеса різноманітна, але незважаючи на це, місто, за
          історичними мірками, молоде: йому всього трохи більше ніж 220 років. Але й за
          цей невеликий період воно пережило неймовірну кількість подій, що не могло не
          відбитися в архітектурі тутешніх будівель і вулиць.</p>
          <hr><h2>Дещо з історії міста</h2>
          <p>Будівництво міста і морського порту почалося 22 травня 1794 року за указом
          Катерини Другої, що повинно було значно покращити торгівельні зв'язки з Євро-
          пою. Почалося будівництво під керівництвом віце-адмірала Хосе де Рибаса –
          першого градоначальника Одеси, за планом, що склав полковник-інженер Франц
          Деволан.</p>
          <p>Тутешні землі можуть розповісти про скіфів, кіммерійців, сарматів і греків.
          Кожна культура залишила свій слід в історії міста. Місцевість, на якій нині
          розташована Одеса, була заселена ще до початку Великого переселення
          народів, вона належала і Золотій Орді, і Великому князівству Литовському.</p>
          <p>Відзначилася Одеса і в Другій світовій війні. <strong>Одеса – місто-
          герой</strong>. Оборона міста трималася <strong>73 дні</strong>. Щоб дізнати-
          ся більше про цей період варто відвідати Меморіал героїчної оборони Одеси
          411-ї берегової батареї. Цей меморіальний комплекс, присвячений героїчній
          обороні міста під час Великої Вітчизняної війни. Комплекс включає в себе му-
          зей, виставку військової техніки під відкритим небом, батарею берегової оборо-
          ни, а також великий засаджений дубами парк.</p>
          <hr> <h2>Сьогодення одеситів</h2>
          <p>Сьогодні місто розвивається як курортний і промисловий центр. Це одне з
          найулюбленіших туристами місць в Україні, і в цьому немає нічого дивного. Тут
```

```
        Ви можете ніжитися в променях теплого південного сонця і купатися в ласкаво-
        му Чорному морі. Кількість пам'яток не злічити на пальцях однієї руки:</p>
        <ul>
            <li><a href="http://odessa-life.od.ua/article/5449-sekrety-odesskogo-rynka-privoz">
            Привоз </a>, </li>
            <li><a href="http://www.odessaguide.net/sights_deribasovskayastreet.ru.html">
            вулиця Дерибасівська</a>,</li>
           <li>Одеський театр опери і балету (другий за красою в Європі),</li>
           <li>Приморський бульвар,</li>
           <li>Напівкругла площа (1826-1829) з пам'ятником Рішелье,</li>
           <li>палаци Воронцова, Наришкіної і Потоцького,</li>
           <li>Пасаж,</li>
           <li>Успенський монастир (1824) і Духовна семінарія,</li>
           <li>Одеський морський порт (найбільший на Чорному морі),</li>
        </ul>
        <p>– і це далеко не весь список. Для маленьких і непосидючих відкриває свої
        двері Одеський дельфінарій.</p>
        <p>Все, що Ви можете побачити в Одесі. Є тут навіть «восьме чудо світу» –
        <em>Потьомкінські сходи</em>, своєрідний вхід у місто з боку моря.  2004 року
        було проведено голосування, серед найкрасивіших сходів, і в Європі Потьо-
        мкінські увійшли до десятки кращих, посівши 6-те місце, також у цьому списку
        присутні сходи Монмартр у Парижі та Сходи Храму Афіни на о. Родос.</p>
        <p>В Одесі існує величезна система катакомб, вони не тільки за своїм хитро-
        сплетінням, але й за протяжністю (≈ 3 000 км) – одні з найбільших у світі.</p>
        <p class="note"> Для порівняння: довжина Римських катакомб становить 300 км,
        Паризьких – 500 км.</p>
        <p>Якщо Ви втомилися від споглядання пам'ятників, то ласкаво запрошуємо до
        Одеського Ботанічного саду, в якому Ви не лише відпочините душею і тілом, а й
        побачите фактично перший парк степової частини України. За двохсотлітню
        історію саду з усього світу у ньому назбиралася величезна колекція найрізно-
        манітніших рослин.</p>
        <p>Одеса – це місто веселощів та гумору. Відвідавши його лише раз, Вам
        обов'язково захочеться побачити його ще. Адже Одеса – це не зовсім місто – це
        <sup> ☺ </sup>посмішка Бога.<sub font-color="red">☼</sub></p>
        <img src="photo/img1.jpg" alt="Одеса – це не місто, а ...">
        <img src="photo/img0.jpg"> <img src="photo/img00.jpg">
        <img src="photo/img000.jpg">
        <p class="note"> Використовувались матеріали статті
        <a href="http://angatravel.com.ua/ua/ah-odesa-perlina-bilya-morya.html">
        «АХ, ОДЕСА! ПЕРЛИНА БІЛЯ МОРЯ»</a> </p>
      </div>
      <div class="bl fl3"><p > &#169 Зайченко Маргарита</p></div>
    </div>
  </body>
</html>
```

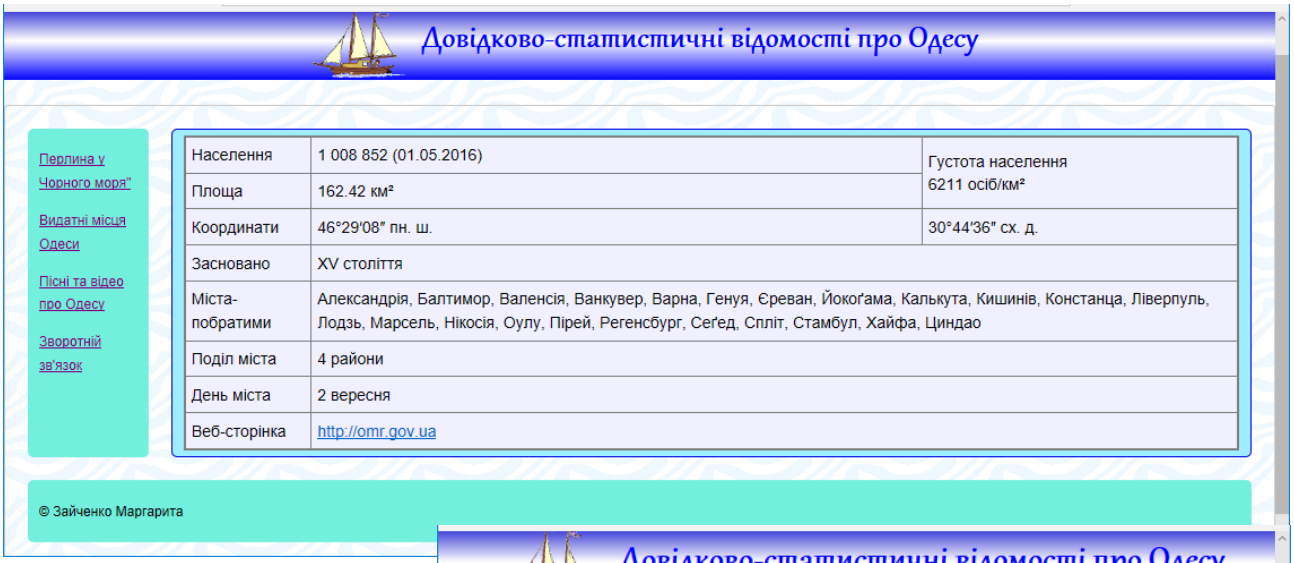

При зміненні розміру екрана браузера вміст веб-сторінки гнучко зміниться:



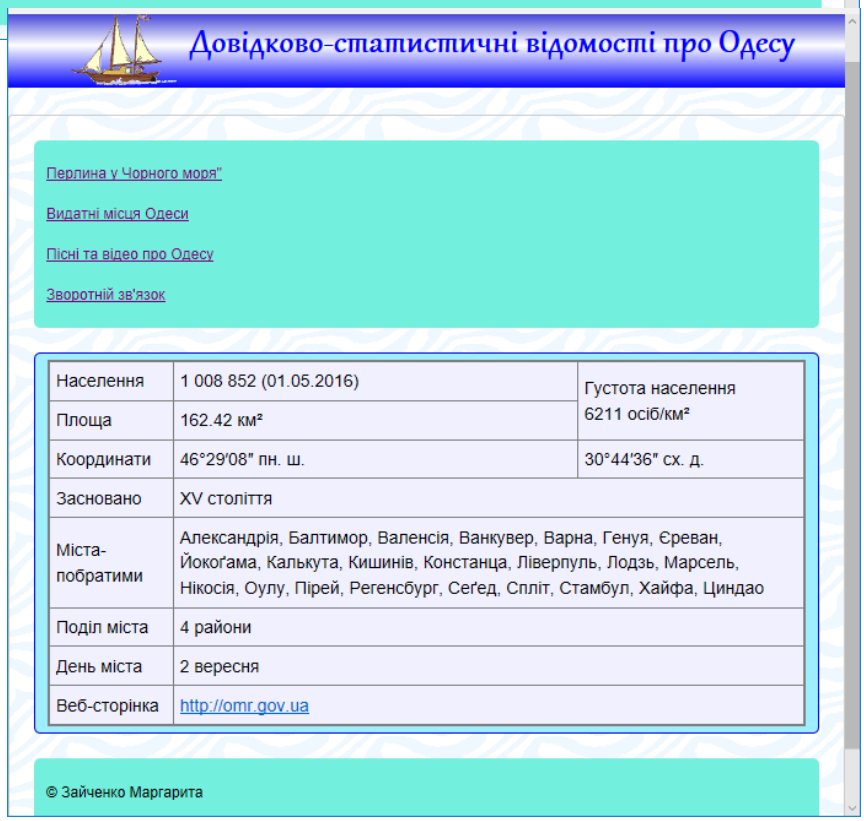
**Лістинг файлу *table.html***

```
<html>
 <head>
    <title>Flexbox на практиці</title>
    <meta charset="utf-8">
     <link rel="stylesheet" type="text/css" href="style.css">
     <meta name="keywords" content="Одеса найкраще місто в світі">
     <meta name="description" content="Одеса найкраще місто в світі">
 </head>
 <body background="Photo/fon-110.gif">
    <h1 class="gradient">  <img src="Photo/ship.gif"  class="imgship">
       Довідково-статистичні відомості про Одесу
    </h1>
    <div class="main">
     <div class="bl fl1">
      <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
      <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
      <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
      <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
     </div>
     <div class="bl fl2">
      <table >
    `   <tr>
          <td>Населення</th>    <td>1 008 852 (01.05.2016)</th>
          <td rowspan="2" width="30%">Густота населення<br>6211 осіб/км²</td>
        </tr>
        <tr>
          <td>Площа</td>   <td>162.42 км²</td>
        </tr>
        <tr>
          <td>Координати</td>
              <td> 46°29'08" пн. ш.</td>   <td> 30°44'36" сх. д.</td>
        </tr>
        <tr>
          <td>Засновано</td>    <td colspan="2">XV століття</td>
        </tr>
        <tr>
          <td>Міста-побратими</td>
          <td colspan="2">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя,
          Єреван, Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель,
          Нікосія, Оулу, Пірей, Регенсбург, Сеґед, Спліт, Стамбул, Хайфа, Циндао</td>
        </tr>
        <tr>
          <td>Поділ міста</td>    <td colspan="2">4 райони</td>
        </tr>
```

```
        <tr>
          <td>День міста </td>    <td colspan="2">2 вересня</td>
        </tr>
        <tr>
          <td>Веб-сторінка</td>
          <td colspan="2"><a href="http://omr.gov.ua">http://omr.gov.ua</a></td>
        </tr>
      </table>
     </div>
     <div class="bl fl3">  <p> &#169 Зайченко Маргарита</p>  </div>
    </div>
  </body>
</html>
```




При зміненні розміру екрана браузера вміст веб-сторінки гнучко зміниться:



**Лістинг файлу *pictures.html*:**

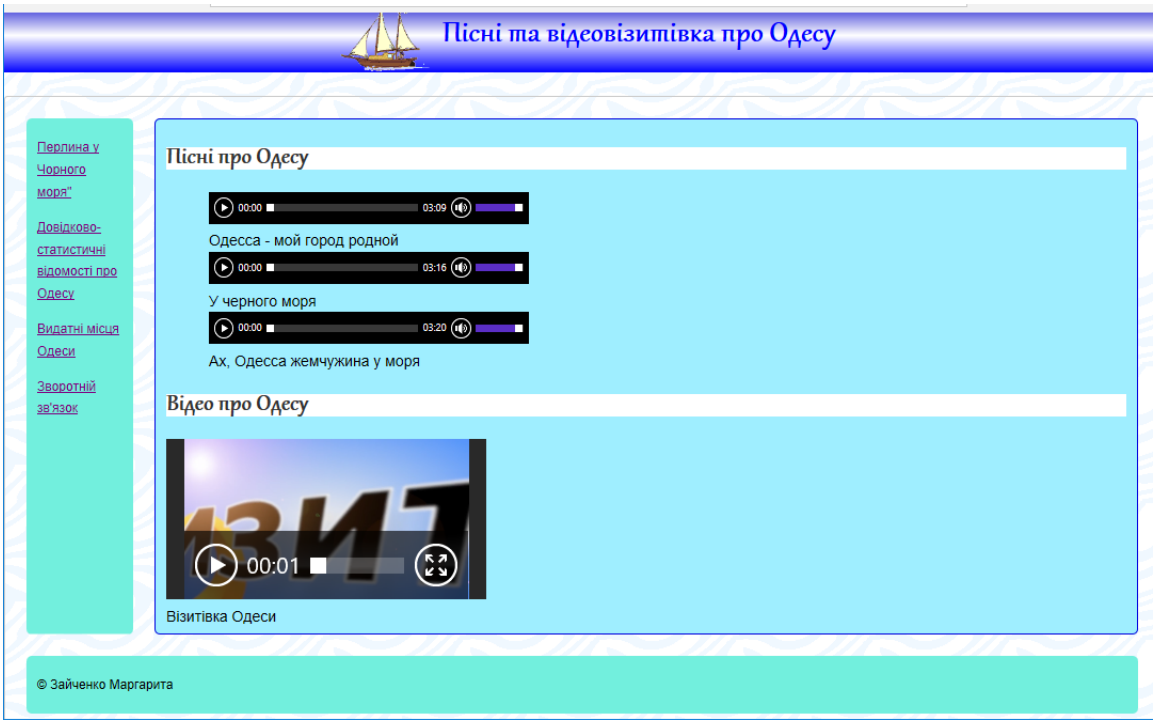
```
<html>
 <head>
    <title>Flexbox на практиці</title>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta name="keywords" content="Одеса найкраще місто в світі">
    <meta name="description" content="Одеса найкраще місто в світі">
    <style>
      .thumbs {
         display: inline-block;
         }
         .thumbs:hover .thumb {
         transform: scale(0.9);  /* Зменшити всі фотографії */
      }
    .thumb {
         background: #27f;       /* Колір рамки */
         padding: 10px;           /* Товщина рамки навколо картинки */
         transition: 1s;          /* Час переходу */
    }
      .thumb:hover {
         transform: scale(1.2) !important; /* Збільшити фотографію */
      }
      img {
         height: 220px;
         margin-left: 10px;
      }
    </style>
 </head>
 <body background="Photo/fon-110.gif" >
    <h1 class="gradient"> <img src="Photo/ship.gif"  class="imgship">
      Видатні місця Одеси
    </h1>
    <div class="main">
      <div class="bl fl1">
        <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
           <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості
           про Одесу</a></p>
        <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
        <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
      </div>
      <div class="bl fl2">
        <div class="thumbs">
         <img src="Photo/img_lb5_1.jpg" alt="" class="thumb">
         <img src="Photo/img_lb5_2.jpg" alt="" class="thumb">
```

```
          <img src="Photo/img_lb5_3.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_4.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_5.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_6.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_7.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_8.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_9.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_10.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_11.jpg" alt="" class="thumb">
          <img src="Photo/img_lb5_12.jpg" alt="" class="thumb">
        </div>
      </div>
      <div class="bl fl3">
        <p> &#169 Зайченко Маргарита</p></div>
    </div>
  </body>
</html>
```


**Лістинг файлу *media.html*:**

```
<html>
  <head>
    <title>Flexbox на практиці</title>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta name="keywords" content="Одеса найкраще місто в світі">
    <meta name="description" content="Одеса найкраще місто в світі">
  </head>
  <body background="Photo/fon-110.gif" >
    <h1 class="gradient"> <img src="Photo/ship.gif"  class="imgship">
      Пісні та відеовізитівка про Одесу
    </h1>
```

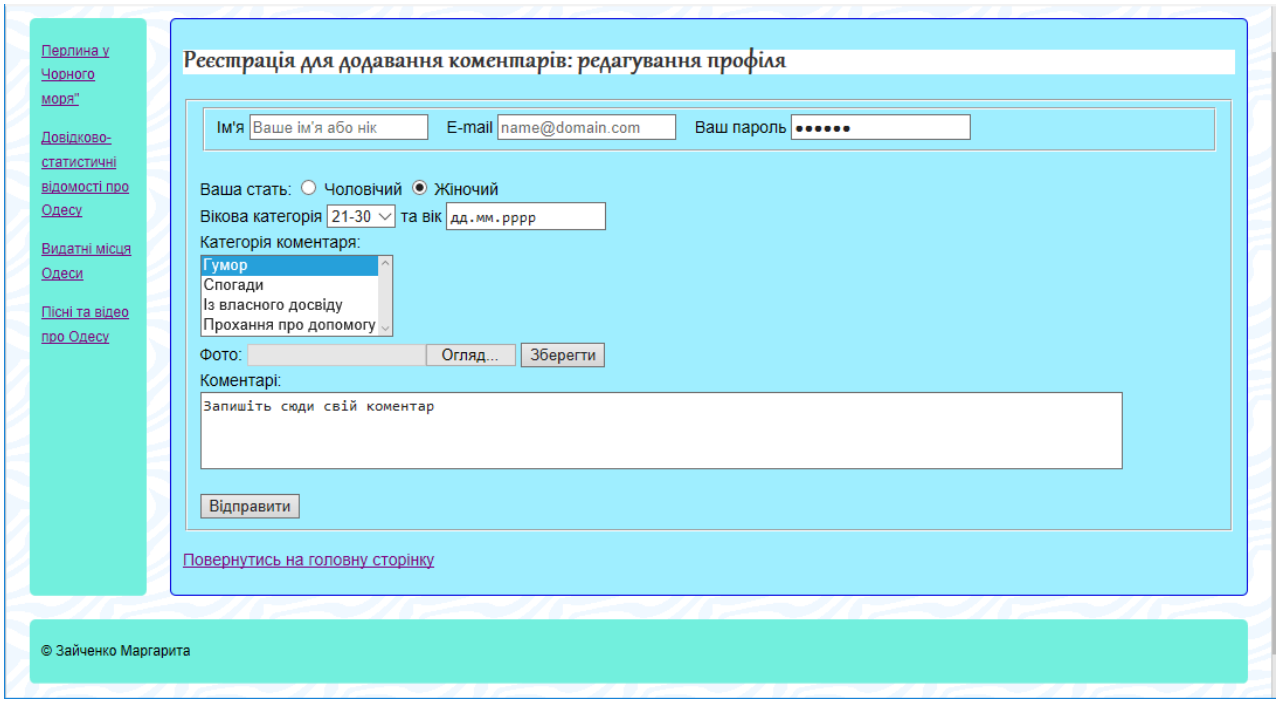
```
    <div class="main">
      <div class="bl fl1">
        <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
        <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
        Одесу</a></p>
       <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
       <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
      </div>
      <div class="bl fl2">
        <div class="thumbs">
          <h2">Пісні про Одесу</h2>
          <figure>
              <audio src="Audio/Odessa moy gorod rodnoy.mp3" controls></audio>
              <figcaption>Одесса - мой город родной</figcaption>
              <audio src="Audio/у Черного моря.mp3" controls></audio>
              <figcaption>У черного моря</figcaption>
              <audio src="Audio/ах, Одесса жемчужина у моря.mp3" controls></audio>
              <figcaption>Ах, Одесса жемчужина у моря</figcaption>
          </figure>
          <br> <h2>Відео про Одесу</h2>
          <object data="Audio/aerovizitkaOdessa.mp4" type="video/mp4"
          standby="Йде завантаження. Зачекайте.">Візитівка Одеси </object>
          <figcaption>Візитівка Одеси</figcaption>
        </div>
      </div>
      <div class="bl fl3"><p > &#169 Зайченко Маргарита</p></div>
    </div>
  </body>
</html>
```



**Лістинг файлу *form.html*:**

```
<!DOCTYPE html>
 <head>
  <meta charset="utf-8">
  <title>Реєстрація для додавання коментарів: редагування профіля</title>
  <link rel="stylesheet" type="text/css" href="style.css">
  <style>
     textarea {
        width: 90%;  height: 60px;
    }
  </style>
 </head>
 <body background="Photo/fon-110.gif">
  <div class="main">
     <div class="bl fl1">
        <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
        <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
        Одесу</a></p>
        <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
        <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
     </div>
     <div class="bl fl2">
       <h2>Реєстрація для додавання коментарів: редагування профіля</h2>
       <form action="#" method="post">
       <fieldset>
         <fieldset>
          <label>  Ім'я
            <input type="text" name="name" placeholder="Ваше ім'я або нік" required>
          </label>
          <label>     E-mail
            <input type="email" name="email" placeholder="name@domain.com" required>
          </label>
          <label for="password-field">   Ваш пароль</label>
          <input type="password" name="password" id="password-field" value="123456">
        </fieldset> <br> Ваша стать:
        <label>
            <input type="radio" name="question-one" value="m">  Чоловічий
        </label>
        <label>
            <input type="radio" name="question-one" value="w" checked>  Жіночий
         </label>
         <br> Вікова категорія
         <select name="age">
           <option value="до 15">до 15</option>
           <option value="16-20">16-20</option>
```

```
            <option value="21-30" selected>21-30</option>
            <option value="31-40">31-40</option>
            <option value="41-50">41-50</option>
            <option value="51-60">51-60</option>
            <option value="61-70">61-70</option>
          </select>
          та вік <input type="date" name="birthday">
          <br><br>Категорія коментаря:<br>
          <select name="tehn" multiple size="4">
            <option selected> Гумор</option>
            <option> Спогади</option>
            <option> Із власного досвіду</option>
            <option> Прохання про допомогу</option>
          </select>
          <br><br> Фото: <input type="file" value="Choose File">
          <input type="submit" value="Зберегти">  <br> <br>
          <label>  Коментарі:<br>
            <textarea name="comment">Запишіть сюди свій коментар  </textarea>
          </label>   <br><br>
          <input type="submit" name="submit" value="Відправити">
        </fieldset>
      </form>
      <p><a href="index.html">Повернутись на головну сторінку </a> </p>
    </div>
    <div class="bl fl3"><p> &#169 Зайченко Маргарита</p>  </div>
  </div>
 </body>
</html>
```



*Самостійна робота № 6*

# ЗАВАНТАЖЕННЯ ВЕБ-СТОРІНОК НА СЕРВЕР

## Завдання

1. Розмістити на свій сайт на сервер www.ho.ua, зарестрований при виконанні самостійної роботи № 2, файли усіх веб-сторінок, створених під час лабораторних робіт № 3 – 7 та самостійних робіт № 3 – 4. Попередній вміст свого сайту очистити (видалити). Послідовність дій щодо завантаження на сервер усіх розроблених веб-сторінок сайту описана у п. 2 теоретичних відомостей до самостійної роботи № 2 (див. с. 55).
2. Надати на перевірку викладачеві URL-адресу свого сайту.

## Теоретичні відомості

### Адміністрування сайту

Існування сайту та його повноцінна робота неможливі без регулярних заходів щодо його підтримки, а саме: наповнення сайту новим контентом, редагування наявних матеріалів та своєчасного оновлення необхідних компонентів і модулів, захисту від мережевих атак, резервного копіювання даних. Увесь комплекс подібних робіт – це ніщо інше, як **адміністрування сайту**.

Адміністрування сайту має здійснюватися на постійній основі, адже відвідувачам цікавий «живий» ресурс, що містить актуальну інформацію та виконує всі заявлені функції. До **завданнь адміністратора сайту** може входити його інформаційна та технічна підтримка, а також роботи з модернізації, які не потребують кардинальних змін дизайну або функціональності сайту.

Звичайно адміністрування сайту виконується безпосередньо силами його власника. Для цієї мети або наймається спеціальний співробітник або ж навчається якийсь з існуючих співробітників. При цьому якщо сайт виконаний на CMS (система управління контентом), то адміністрування сайту зазвичай не вимагає знань у сфері веб-дизайну та веб-програмування.

Втім, навіть використання CMS на сайті не дає підстави вважати, що адмініструванням можна знехтувати або ж покласти його на плечі одного з співробітників компанії-власника сайту як додаток до основних обов'язків. Насправді, в адмініструванні сайтів, як і в будь-якій іншій справі, **потрібен ретельно продуманий підхід**.[1]

[1] Адміністрування сайту. [Електронний ресурс] – Режим доступу: http://webstudio2u.net/ua/studio-web/673-administrirovanie-saita.html.

# ТЕМА 2
# ОНЛАЙН-КОНСТРУКТОРИ САЙТІВ

*Самостійна робота № 7*
## ПІДГОТОВКА МАТЕРІАЛУ ДЛЯ САЙТУ

### Завдання

Підготувати матеріал для наповнення сайту, який буде створено на подальшій лабораторній роботі № 8. Тематику матеріалу вибрати відповідно до індивідуального варіанта з табл. 8.1 (с. 130) лабораторної роботи № 8 (якщо у Вас є власні наопрацювання у вигляді курсових робіт, проектів, публікацій, які можна використати для наповнення сайту, слід узгодити тематику матеріалу з викладачем).

Підготовка матеріалу полягає у його доборі у потрібному обсязі (для 5-ти веб-сторінок сайту), систематизації, доборі графічного супроводу (фотографій, а можливо і діаграм), структуруванні та формулюванні заголовків різних рівнів для якнайкращого подання матеріалу на веб-сторінках сайту. Вміст кожної сторінки має розкривати якийсь із розділів (ракурсів) тематики розроблюваного сайту.

Слід дотримуватись певної послідовності кроків під час підготовки матеріалу при створенні веб-сайту, адже чим краще проведено підготовку матеріалу, тим швидше буде реалізовано задум:

1) визначитися з тематикою сайту: хто буде його аудиторією, кому він адресований;
2) розпланувати свій сайт по сторінках: попрацювати над структурою, розподілити контент, впевнитись, що його обсяг «знає почуття міри» щодо загальної площі сторінки;
3) продумати структуру заголовків різних рівнів, бажано щоб вони були креативними (особливо це актуально для заголовків першого рівня, які будуть індексуватися пошуковиками);
4) підібрати влучне графічне супроводження (фотографії, малюнки, схеми тощо) сайту, підрізати їх до оптимальних розмірів.

При використанні сторонніх (не власних) матеріалів слід посилатися на першоджерело, з якого вони були взяті.

### Теоретичні відомості

#### 1. Етапи створення сайту

Створення сайту умовно можна поділити на такі етапи[1]:

1. Планування – визначення тематики і призначення майбутнього сайту.

[1] Етапи створення веб-сайтів. [Електронний ресурс] – Режим доступу: http://edufuture.biz/index.php?title=Етапи_ створення_веб-сайтів.

2. Розробка – розробка структури сайту, добір матеріалів, вибір програмних засобів для його створення.

3. Створення окремих сторінок відповідно до структури, включення до них гіперпосилань.

4. Тестування – перевірка та редагування веб-сайту.

5. Розміщення – розміщення сайту в інтернеті.

6. Підтримка – оновлення вмісту сайту.

Створення веб-сайту починається зі створення інформаційної моделі сайту. Будь-яку веб-сторінку можна оцінити за двома параметрами: вміст та зовнішній вигляд. Проте спочатку потрібно вирішити, яку інформацію на ній розміщувати. Необхідно детально проаналізувати, скільки і якої інформації потрібно подати на веб-сторінці. Створюючи проект сайту, треба добре продумати його загальну структуру, зміст інформації та посилання.

## 2. Інформаційне наповнення сайту

### 2.1. Поняття вмісту

Увесь контент охороняється законом про авторське право, оскільки він є продуктом інтелектуальної праці і має своїх авторів і власників. Окрім якості контенту одним з важливих критеріїв є його доступність. Особливу важливість для користувача має актуальність контенту, його значущість на даний час і достовірність наданих даних, а також відповідність контенту щодо поставлених цілей.

Унікальний контент[1] (ексклюзивний контент) – це інформація, яка не має аналогів на ресурсах схожої тематики або розміщена на веб-сайті з дозволу правовласника, така, що є результатом інтелектуальної праці та охороняється законом про авторське право. Найчастіше цей термін застосовують до текстового наповнення сайтів (текстовий контент).

Унікальні статті, що написані для конкретного ресурсу, розміщуються на ньому і є першоджерелом, будь-який передрук допустимий лише з дозволу законного власника і за його умовами. Грамотне, якісно виконане і цікаве наповнення сайту здатне значно допомогти компанії та її сайту – підвищити відвідуваність, популярність, прибуток.

### 2.2. Рерайт статей (ReWrite)

Один зі способів отримати контент для сайту – рерайт готових статей. Рерайт це перерозподіл домінанти тексту, змінення форми без зміни змісту, так і розбавлення контенту необхідними ключовими фразами, за якими розробник подає сайт. При рерайті, так само, як і при копіпасті, постає питання законності й етики. Щоб не переступати межі етики, слід обов'язково посилатися на джерело цих ідей.

---

[1] Інформаційне наповнення сайту. [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/ Інформаційне_наповнення_сайту.

### 2.3. Правила і поради щодо якості контенту

– Текст має бути написаний цікаво, привертати увагу й затримувати відвідувача.

– У тексті не повинно бути орфографічних помилок, друкарських недоречностей, використання різних елементів у межах одного тексту (наприклад, різних термінів для одного й того самого поняття), оскільки це може заплутати читача.

– Текст має бути унікальним для пошукових систем. Брати чужий текст і ставити на свій сайт, змінивши лише назву компанії, неправильно. За це Ваш сайт може бути вилучено з баз пошукових систем.

– Текст слід оптимізувати під пошукові системи – додати (органічно вписати) достатньо багато ключових слів і виразів.

– Текст має бути «читабельним» для відвідувачів сайту. Часто власники сайту так «оптимізовують» текст, що жоден відвідувач не в силах затриматись на сторінці вже після першого речення. Не захоплюйтеся.

– Текст має виконувати задану для нього мету: інформувати, переконувати, продавати тощо.

– Текст слід зручно розміщувати на сторінці, щоб відвідувач не плутався у великому масиві тексту без єдиного виділеного рядка і не шукав довго те зображення, яке описується в тексті.

*Лабораторна робота № 8*

# Створення веб-сайту за допомогою сайт-конструктора

***Мета:*** навчитися застосовувати онлайн сайт-конструктори для створення сайту.

## Навчальні питання

1. Доменні імена та IP-адреси веб-сайтів.
2. Онлайн-конструктор сайтів.
3. Створення дизайну та структури сайту.
4. Публікація сайту.

## Завдання

За допомогою сайт-конструктора на сервері http://www.ucoz.ua створити веб-сайт за тематикою, вказаною для Вашого варіанта в табл. 8.1.

*Таблиця 8.1* – **Варіанти індивідуальних завдань**

| Варіант | Тематика сайту |
|---|---|
| 1 | Електронна журналістика – перспективи та реалії |
| 2 | Вимоги до основних елементів електронних видань |
| 3 | Проектування та створення електронного видання |
| 4 | Електронні видання та вибір типу видання |
| 5 | Програмне забезпечення для створення електронного видання |
| 6 | Дизайн електронних видань |
| 7 | Сучасні технології підготовки електронних видань |
| 8 | Електронні видання в сучасному інформаційному просторі |
| 9 | Типологія електронних видань |
| 10 | Електронне та паперове видання: плюси та мінуси |
| 11 | Програмні додатки для популяризації сайтів |
| 12 | Використання блогу для журналіста |
| 13 | Використання сервісів Google для видавця та журналіста |
| 14 | Статичні та динамічні сайти: плюси та мінуси |
| 15 | Елементи навігації по сайту та засоби їх створення |
| 16 | Використання хмарних технологій для журналіста |

Послідовність виконання завдання має бути такою:

1) зареєструватися на сайті;

2) вибрати дизайн сайту та його модулі із запропонованих сайт-конструктором шаблонів, доречний за тематикою для Вашого варіанта;

3) збудувати на сайті не менш 5-ти веб-сторінок з відповідними заголовками різних рівнів, скориставшись матеріалом, підібраним під час виконання са-

мостійної роботи № 6;

4) однією із веб-сторінок створити сторінку *Контакти*, на якій з використанням різних шрифтів, кольору тексту, розміру шрифту, видів накреслення (напівжирне, курсивне, підкреслене) і вирівнювання тексту записати власне прізвище, групу, потік, курс, вставити власну фотографію, вказати адресу електронної пошти, наприклад: sergienko@outlook.com;

5) останнім полем (рядком) кожної сторінки має бути прізвище розробника, тобто Ваше власне прізвище, наприклад: «Сайт журналіста Коваленко І.П.»;

6) надати на перевірку викладачеві адресу зареєстрованого сайту;

7) підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Що таке IP-адреса? Яка форма її запису?
2. Що називають доменним ім'ям? Які правила його запису?
3. Які існують рівні доменних імен? Навести приклади доменних імен різних рівнів.
4. На які групи поділяються доменні імена першого рівня?
5. Які Ви знаєте українські домени другого рівня?
6. Що називається конструктором сайтів? Які конструктори сайтів з україномовним інтерфейсом Ви знаєте?
7. Які інструменти звичайно пропонуються у конструкторах для роботи із сайтом?
8. Назвати основні етапи створення сайту за допомогою конструктора сайтів uCoz.
9. Що таке WYSIWYG-редактор?
10. Як змінити пункти меню сайту за допомогою конструктора сайтів uCoz?
11. Як активувати або видалити модуль в uCoz?
12. Що таке конструктор шаблонів в uCoz і як ним користуватися?
13. Що таке віджети, як їх підключити в uCoz?
14. Як налаштувати права доступу на сайті uCoz?
15. Які інструменти розкрутки інтегровані в uCoz?
16. Що таке robots.txt і чи потрібно його змінювати?

## Теоретичні відомості

### 1. Доменні імена та IP-адреси веб-сайтів

Кожному комп'ютеру при підключенні до інтернету надається власний унікальний номер, так звана IP-адреса. У кожного веб-ресурсу є також своя IP-адреса.

На сьогодні **IP-адреса** – це група з чотирьох чисел від 0 до 255, яка записується у вигляді 123.45.67.89. Так, IP-адресою офіційного сайту провайдера ООО «Укрнет» є 212.42.76.253, а компанії «Google Україна» – 172.217.18.67.

Число 172.217.18.67 важко запам'ятати, однак, якщо його написати в адресному рядку, то браузер відкриє саме сайт www.google.com.ua, адже доменне ім'я цього сайту www.google.com.ua відповідає IP-адресі 172.217.18.67.

Наявність доменного імені, замість числового еквівалента, дає можливість звертатися до комп'ютера за ім'ям, що ідентифікує власника IP-адреси.

**Доменне ім'я** (англ. *domain name*) – унікальний ідентифікатор у вигляді поєднання символів латинського алфавіту, який надається певній IP-адресі для ідентифікації сайту серед безлічі інших (двох однакових імен бути не може). Крім літер латиниці, в доменному імені можуть використовуватись цифри від 1 до 9 та символ дефісу «-», проте дефіс не може першим та/чи останнім символом в імені. Довжина імені домену може бути від 2 до 63 символів, інколи до 127 символів. Доменне ім'я виконує функцію унікального імені в інтернеті і по суті є більш простим і зручним варіантом запису IP-адреси сайту.

Цікаво, що перший зареєстрований у мережі домен – це Symbolics.com. Американська фірма Symbolics Inc. оформила його в 1985 році і він і зараз працює.

Для того щоб конкретній цифровій IP-адресі поставити у відповідність символьне доменне ім'я Вашого сайту, існують спеціальні сервери DNS[1].

**DNS-сервер** – програма, яка здійснює перетворення доменного імені на цифрову IP-адресу і навпаки. У пам'яті цих серверів зберігаються великі таблиці, в яких кожному доменному імені ставиться у відповідність IP-адреса.

Пам'ятайте: одній IP-адресі можуть відповідати безліч доменних імен.

Вся інформація про доменні імена зберігається в центральній базі даних DNS, яка розміщена на кількох потужних комп'ютерах, розкиданих по всьому світу. У цій базі зберігається інформація про дату реєстрації, фізичного або юридичного власника доменного імені, а також шлях до так званого сервера імен – NAMESERVER, де міститься інформація, на яку вказує доменне ім'я.

«Корпорація з керування доменними іменами і IP-адресами» (*Internet Corporation for Assigned Names and Numbers*, скорочено **ICANN**) – міжнародна некомерційна організація, створена 18 вересня 1998 року за участю уряду США для врегулювання питань, пов'язаних з доменними іменами, IP-адресами та іншими аспектами функціонування інтернету. ICANN займається акредитацією реєстраторів, а також наглядає за виконанням ними своїх функцій.

**. (крапка)** – домен нульового рівня. Зараз ICANN забезпечує підтримку та керування всім адресним простором DNS в інтернеті, крім доменів першого рівня обмеженого користування, які безпосередньо управляються американськими державними організаціями.

Єдиний інтернет-каталог, який визначає основу DNS, перебуває в державній організації SRI International-Menlo Park, CA, US (Менло-Парк, Каліфорнія, США).

Кожний раз, коли Ви набираєте доменне ім'я в браузері, служба DNS визначає, якій IP-адресі відповідає це ім'я і який саме ресурс потрібно Вам надати. Тому доменне ім'я сайту часто називають – доменною адресою сайту або доменним

[1] **DNS** (Domain Name Service) – служба доменних імен.

ім'ям сервера. По суті, це одне і те саме. Кожне доменне ім'я складається з кількох частин, розділених крапками – це **домени різних рівнів**. Число рівнів доменів звичайно обмежується двома-трьома. Довге доменне ім'я і велике число рівнів домену незручні для використання. Крайнє праве поле називається **доменом верхнього рівня**, далі, справа наліво, слідують імена доменів нижчого рівня.

Доменне ім'я або літерна адреса комп'ютера може бути:

– доменним ім'ям першого (верхнього) рівня (*first level domain*);

– доменним ім'ям другого рівня (*second level domain*);

– доменним ім'ям третього рівня (*third level domain*) тощо.

Доменні імена першого рівня поділяються на організаційні та географічні.

| **Організаційні доменні імена першого рівня в США** | | **Географічні доменні імена першого рівня** | |
|---|---|---|---|
| biz | businesses firms (комерційні) | ua | Ukraine (Україна) |
| com | commercial (комерційні) | uk | United Kingdom (Великобританія) |
| edu | US educational (освіта) | de | Germany (Німеччина) |
| gov | US goverment (уряд) | fr | France (Франція) |
| int | international (міжнародні) | ru | Russia (Росія) |
| mil | US military (військові США) | tv | Tuvalu (Тувалу) |
| nato | NATO (НАТО) | cc | Cocos Islands (Кокосові острови) |
| org | non-profit organization (некомерційні організації) | us | Сполучені Штати Америки |
| net | network (мережеві послуги) | ws | Western Samoa (Західна Самоа) |

На сайті https://www.nic.ru/cgi/na.cgi?step=n_a.all_world наведено понад 200 різних географічних доменних імен першого рівня.

Домени верхнього рівня ще називають доменними зонами.

Доменне ім'я першого рівня має вигляд: www.ua (українська зона інтернету).

Доменне ім'я другого рівня пишеться так – wikipedia.org. У домені wikipedia.org частина wikipedia є доменом другого рівня. Доменне ім'я третього рівня складається з домену другого рівня, до якого ліворуч додано піддомен, наприклад: cat.dog.ua.

**Українські домени:**

.ua – національний український домен першого (верхнього) рівня, призначений для організацій та зареєстрованих торгових марок. Реєструється тільки на основі свідоцтва на знак для товарів та послуг;

.com.ua – комерційний домен другого рівня для підприємств, які працюють на території України;

.net.ua – для мереж та організацій зв'язку на території України;

.org.ua – для організацій на території України;

.biz.ua – для фізичних та юридичних осіб, які ведуть підприємницьку діяльність на території України;

.edu.ua – для навчальних закладів на території України;

.in.ua – для приватних осіб;

.co.ua – призначений для будь-яких реєстрацій без обмежень.

**Сайт або веб-сайт** (від англ. *website* – місце, майданчик в інтернеті) – сукупність веб-сторінок, доступних в інтернет-мережі, які об'єднані як за змістом, так і навігаційно. Апаратні сервери для зберігання веб-сайтів називаються **веб-серверами**. Сама послуга зберігання називається **веб-хостингом.**

Фізично сайт може розміщуватися як на одному, так і на кількох серверах. Зараз сервери для зберігання тільки одного сайту називаються **виділеними** (англ. *dedicated*).

**Сторінки сайтів** – це набір текстових файлів, розмічених мовою HTML. Сторінки сайтів можуть бути простим статичним набором файлів (**статичні**) або створюватися спеціальною комп'ютерною програмою на сервері (**динамічні**). Зазвичай сторінки взаємопов'язані між собою. І відвідувач сайту сам вирішує, в якій послідовності їх переглядати. В інтернеті перегляд сторінок сайтів здійснюється через спеціальні програми – **браузери.**

Перший у світі сайт info.cern.ch з'явився 1990 року. Його створив Тім Бернерс-Лі[1] – батько сучасного інтернету. Автор опублікував на своєму сайті опис нової технології WWW (World Wide Web), основаної на протоколі передачі даних HTTP, системі адресації URI і мові розмітки HTML.

Здебільшого в інтернеті одному веб-сайту відповідає одне доменне ім'я. Саме за доменними іменами сайти ідентифікуються в глобальній мережі. Проте можливі інші варіанти: один сайт на декількох доменах або кілька сайтів під одним доменом. Зазвичай кілька доменів використовують великі сайти (веб-портали), щоб логічно відокремити різні види послуг (mail.google.com, news.google.com, maps.google.com). Нерідкі випадки виділення окремих доменів для різних країн або мов. Наприклад, google.ru, google.de і google.com.ua логічно є сайтом Google на різних мовах, але технічно – це різні сайти.

## 2. Конструктори сайтів

**Конструктори сайтів** – це онлайн-сервіси, за допомогою яких будь-хто може створити сайт, не володіючи при цьому спеціальними знаннями. Створення дизайну, заповнення та редагування контенту відбувається безпосередньо в браузері комп'ютера за допомогою різноманітних шаблонів галереї і дизайнерських інструментів, а тексти, зображення та інші модулі можна пересувати по сайту за допомогою миші.

Звичайно сайт-конструктор не вимагає скачування або встановлення спеціального програмного забезпечення, потрібні тільки комп'ютер, браузер та доступ до інтернету.

Більшість сучасних[2] потужних сайт-конструкторів з якісними, сучасними дизайнерськими шаблонами є платними (наприклад: Instapage (instapage.com),

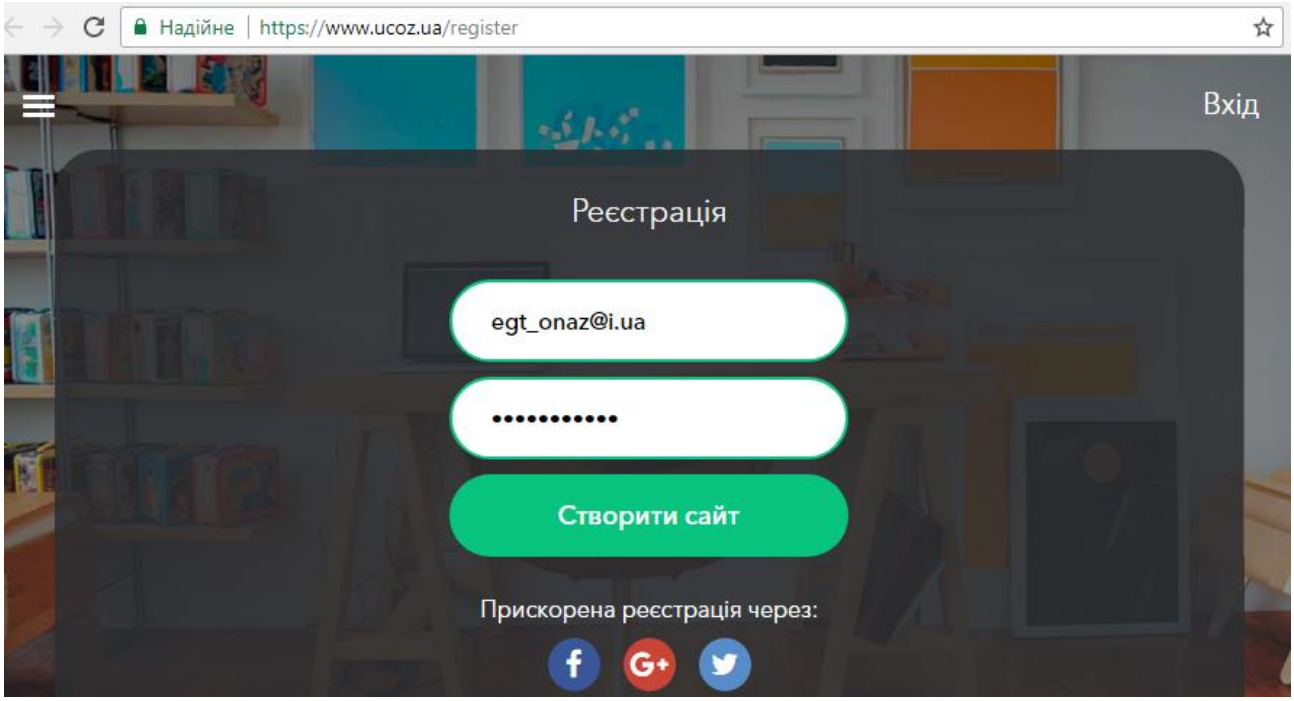
[1] Цікаво, що в 2004 році королева Великої Британії Єлизавета II посвятила Тіма Бернерс-Лі у лицарі за його винахід мови HTML та мережі WWW.

[2] Станом на 2017 рік.

Wix (ru.wix.com)) проте абонентська плата є помірною і становить в еквіваленті від $5 до $25 залежно від функціоналу обраного пакету. Деякі онлайн-конструктори сайтів є безкоштовними (наприклад: україномовний uCoz (ucoz.ua), Website Builder (websitebuilder.com), SiteBuilder (sitebuilder.com)), Sitey (sitey.com), і звичайно працювати з ними можна відразу ж після невеликої реєстрації. Інші ж можуть надавати основну частину функціоналу безкоштовно, а низку додаткових послуг і можливостей для сайту – на платній основі, наприклад: Weebly (weebly.com), Jimdo (ru.jimdo.com), Sitelio (sitelio.com). Також існують онлайн-конструктори сайтів з безкоштовним функціоналом конструкторів лише на певний термін, наприклад: Shopify (shopify.com) надає безкоштовний функціонал на 14 днів, а надалі вимагає оплати для можливості подальшої роботи.

Як приклад розглянемо сайт-конструктор uCoz на сервері www.ucoz.ua. Для реєстрації в системі uCoz слід зайти на сторінку http://www.ucoz.ua/register і вибрати один із доступних способів реєстрації.

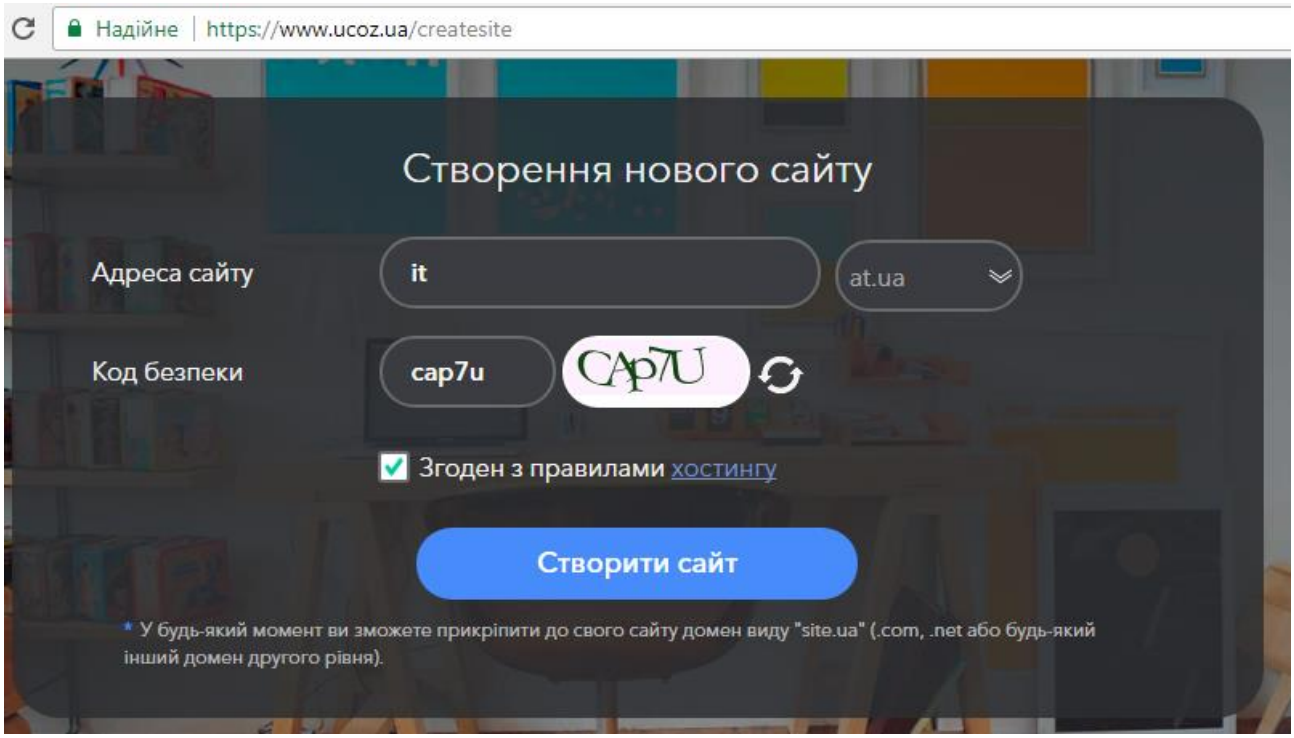
При першому знайомстві з uCoz, коли uID аккаунта ще немає, слід вибрати реєстрацію через email та задати пароль.



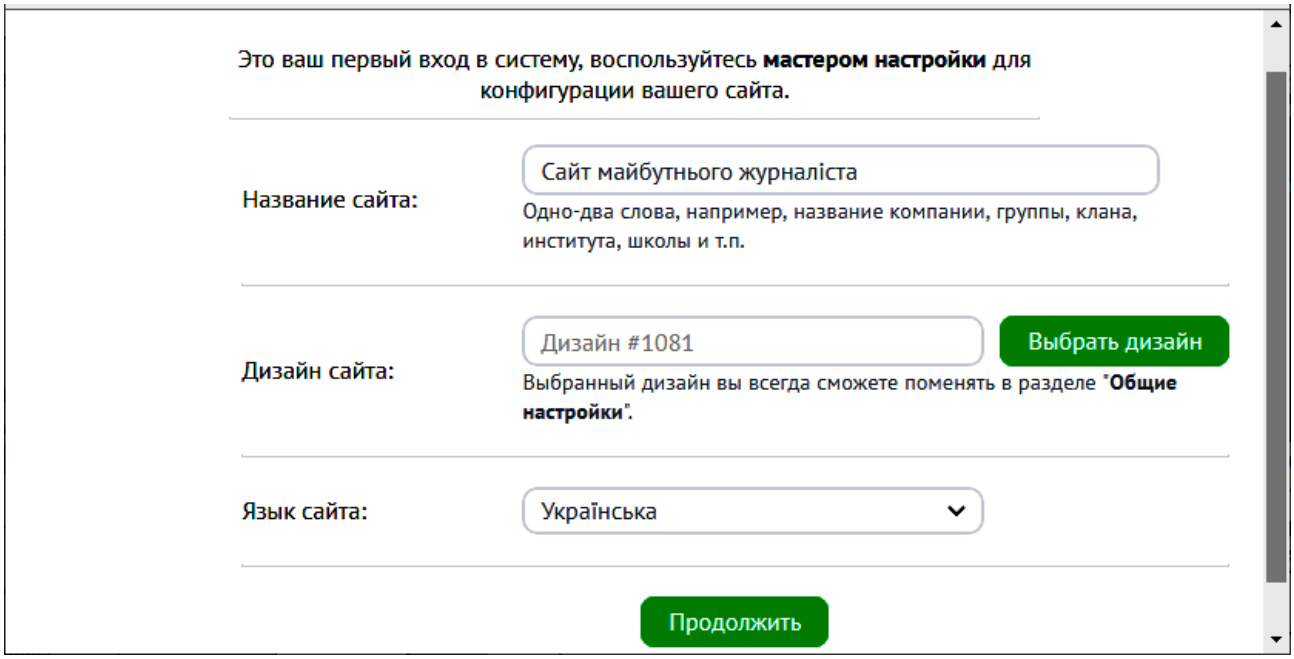

Крім того, користувачі соціальних мереж можуть скористатися прискореною реєстрацією, для цього треба натиснути на іконку відповідної соціальної мережі, в якій у вас є свій аккаунт, і слідувати вказівкам.

### 3. Створення дизайну та структури сайту на uCoz

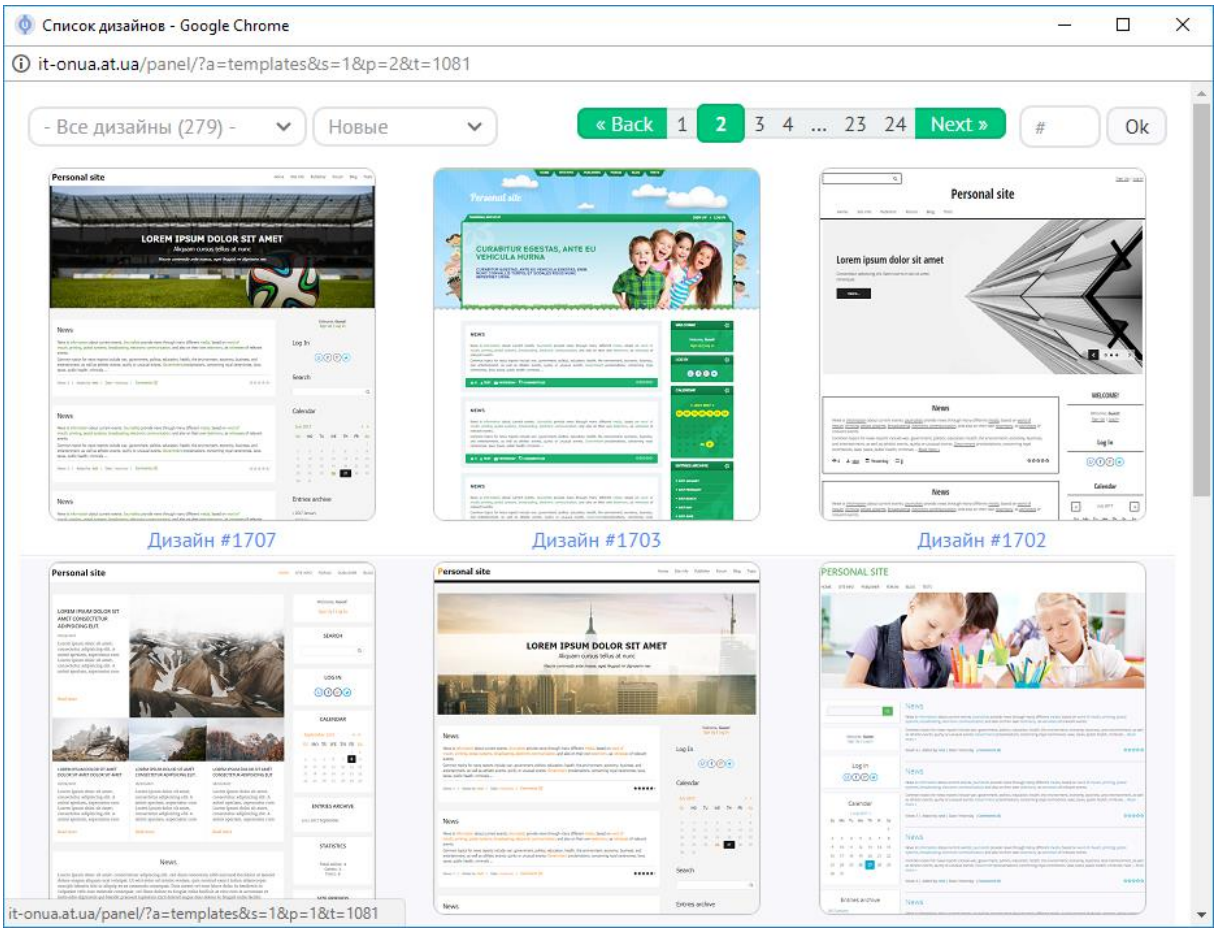
Після авторизації на uCoz можна приступити до створення сайту і вибрати для нього адресу. Саме за цією адресою відвідувачі бачитимуть Ваш сайт в інтернеті. Для придуманої вільним чином адреси сайту слід вибрати доменну зону (ucoz.com, ucoz.net чи ін.), ввести код безпеки і натиснути кнопку *Створити сайт*.



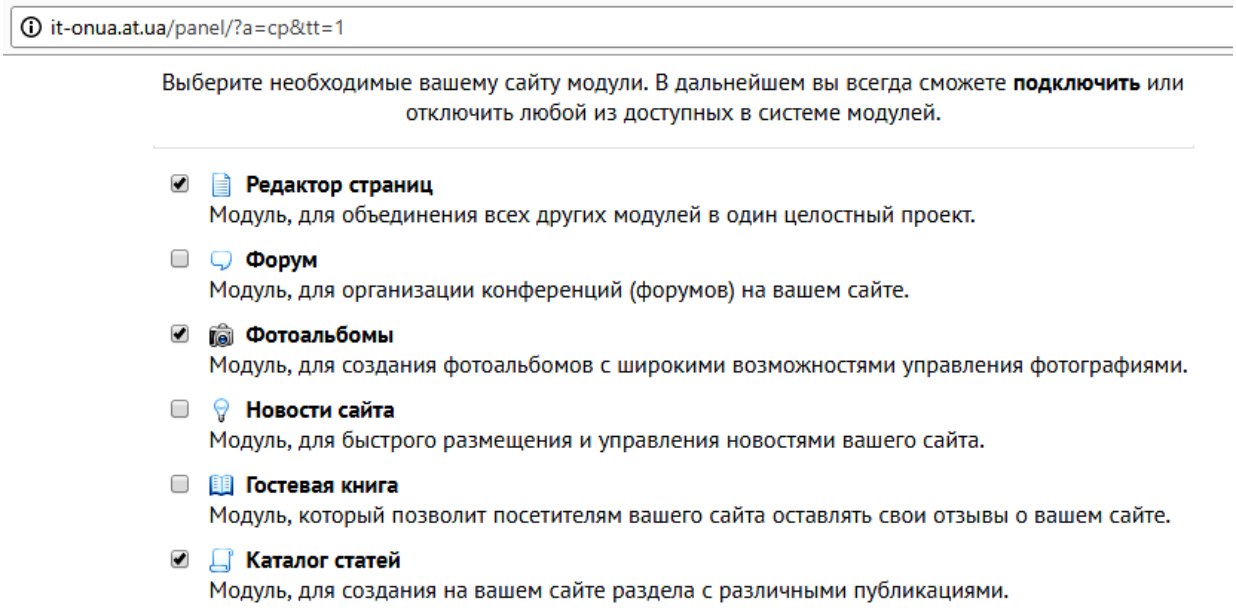

У наступному вікні треба ввести назву сайту, вибрати доречний дизайн відповідно до тематики майбутнього сайту та вибрати зі списку мову сайту. На жаль, станом на вересень 2017 система uCoz не надає можливості використання української мови при роботі з панелю керування, проте дає змогу вибору української мови для самого створюваного сайту.



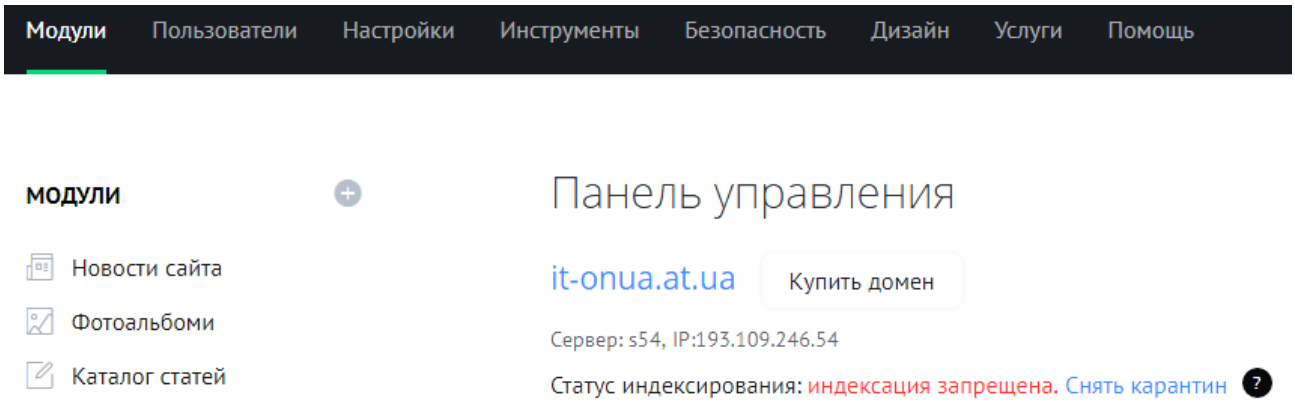

Дизайн на цьому етапі можна вибрати, скориставшись кнопкою *Выбрать дизайн*. При цьому з'явиться вікно зі списком дизайнів. Вибір уподобаного дизайну здійснюється простим клацанням по ньому. Після цього система повернеться до вищенаведеного вікна, де залишиться натиснути кнопку *Продолжить*.




Далі буде запропоновано вибрати модулі для створюваного сайту. Слід поставити галочки навпроти потрібних модулів. Згодом можна буде відключити чи підключити будь-який модуль системи uCoz.



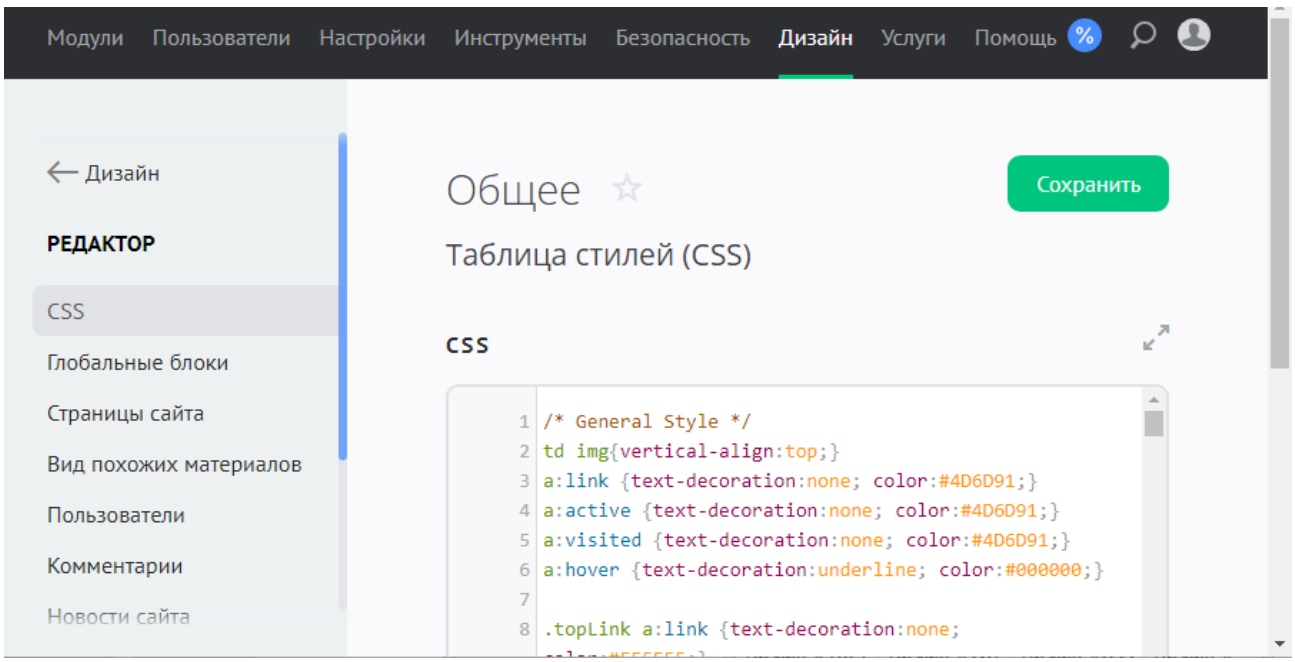

Після натискання кнопки *Продолжить*, розміщеної наприкінці переліку модулів, відбудеться перехід до панелі керування сайтом. Саме тут можна налаштувати всі використовувані модулі та вибрати або змінити дизайн сайту.



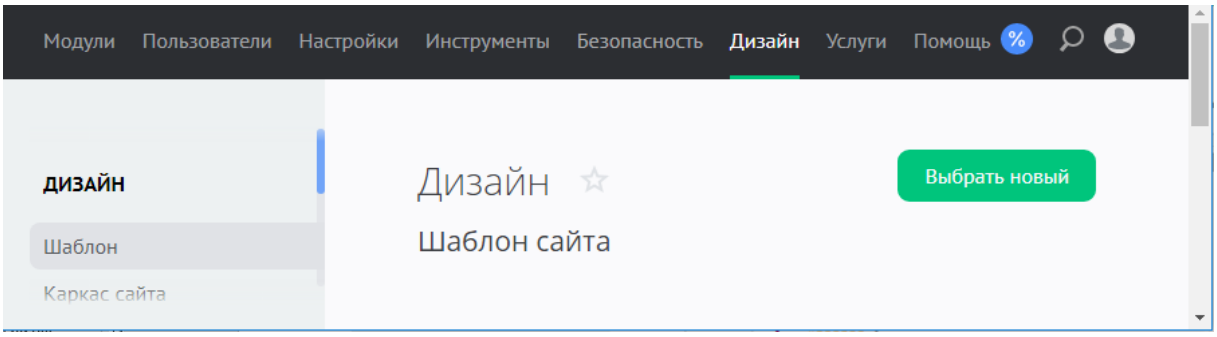

У панелі керування можна:

– додавати і видаляти модулі сайту;
– керувати основними налаштуваннями модулів;
– змінювати дизайн сайту;
– активувати додаткові можливості;
– створювати сторінки / категорії / інформери;
– робити резервні копії сайту;
– керувати налаштуваннями домену;
– змінювати права для груп користувачів;
– ... та багато іншого.

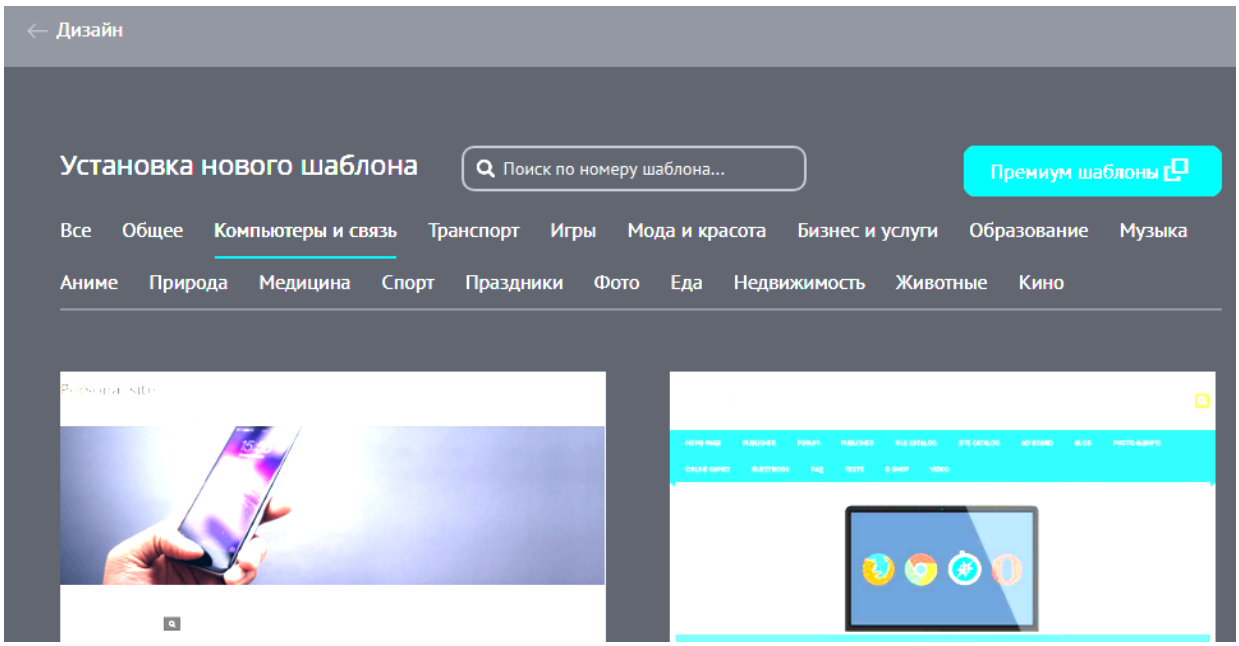
Приміром, щоб змінити дизайн сайту, слід перейти до розділу *Дизайн* (рис. 8.1). Далі натискання кнопки *Редактировать* дозволить гнучко відкоригувати CSS-стилі та HTML-код кожного з блоків на веб-сторінці.




Змінити раніше вибраний шаблон на інший можна скориставшись командою *Дизайн* та натиснувши кнопку *Выбрать новый*.

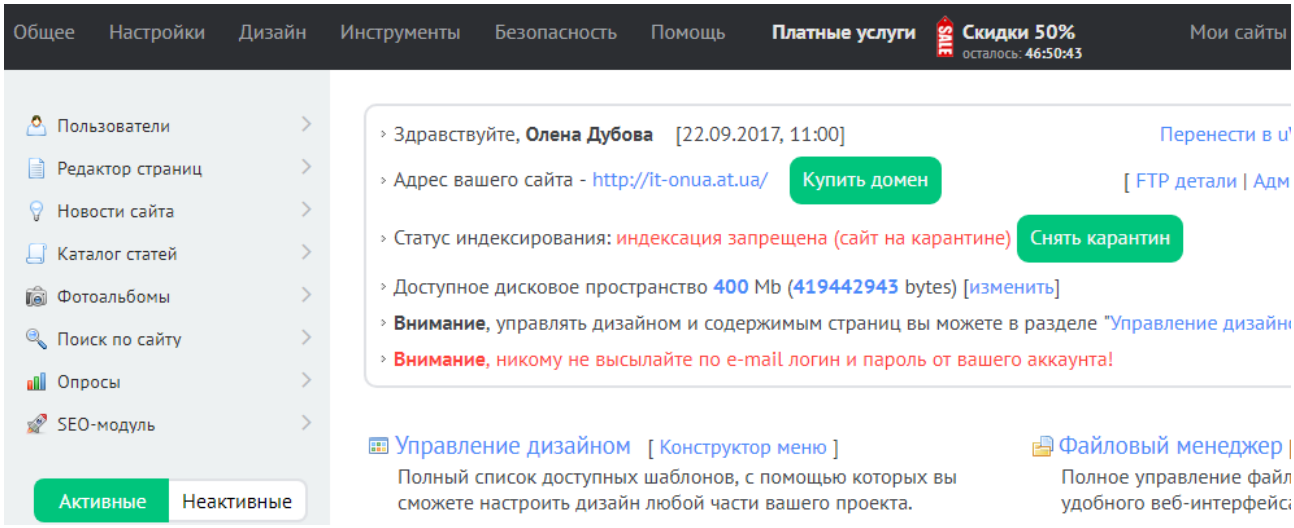



При цьому з’явиться вікно, де всі шаблони згруповані за тематичними рубриками.




Подальше натискання кнопки *Установить* на уподобаному шаблоні призведе до змінення стилю створюваного веб-сайту.

Зайти знову до панелі керування можна за адресою, сформованою з назви сайту на через косу риску /panel/#wpc-add-module (або /admin), наприклад: http://it-onua.at.ua/panel/#wpc-add-module.

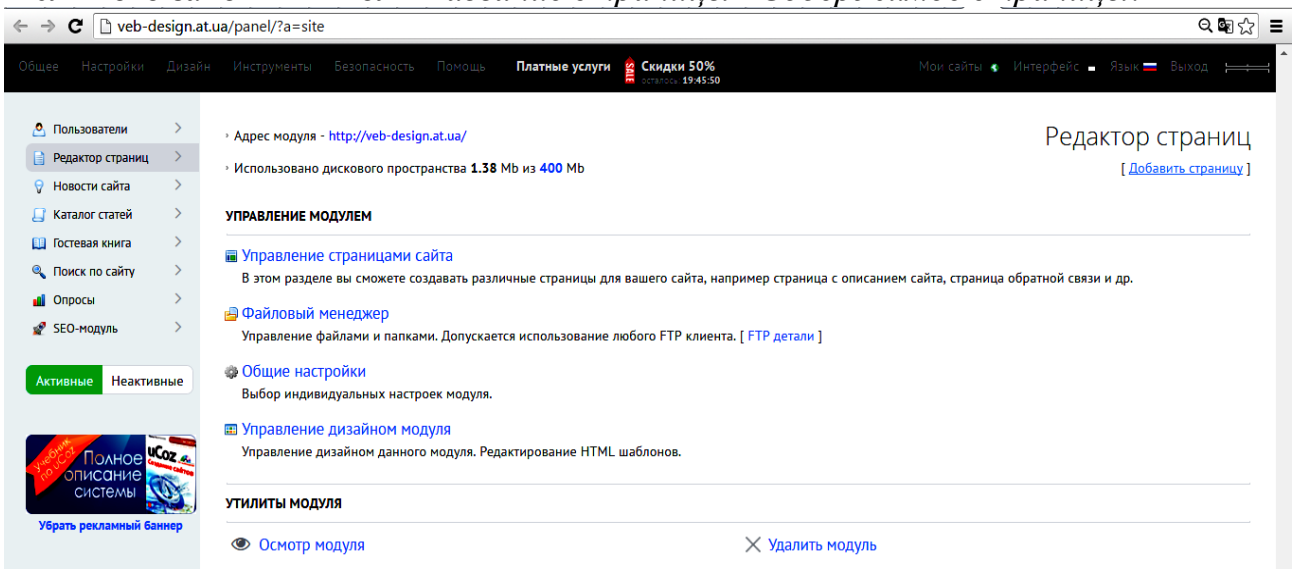


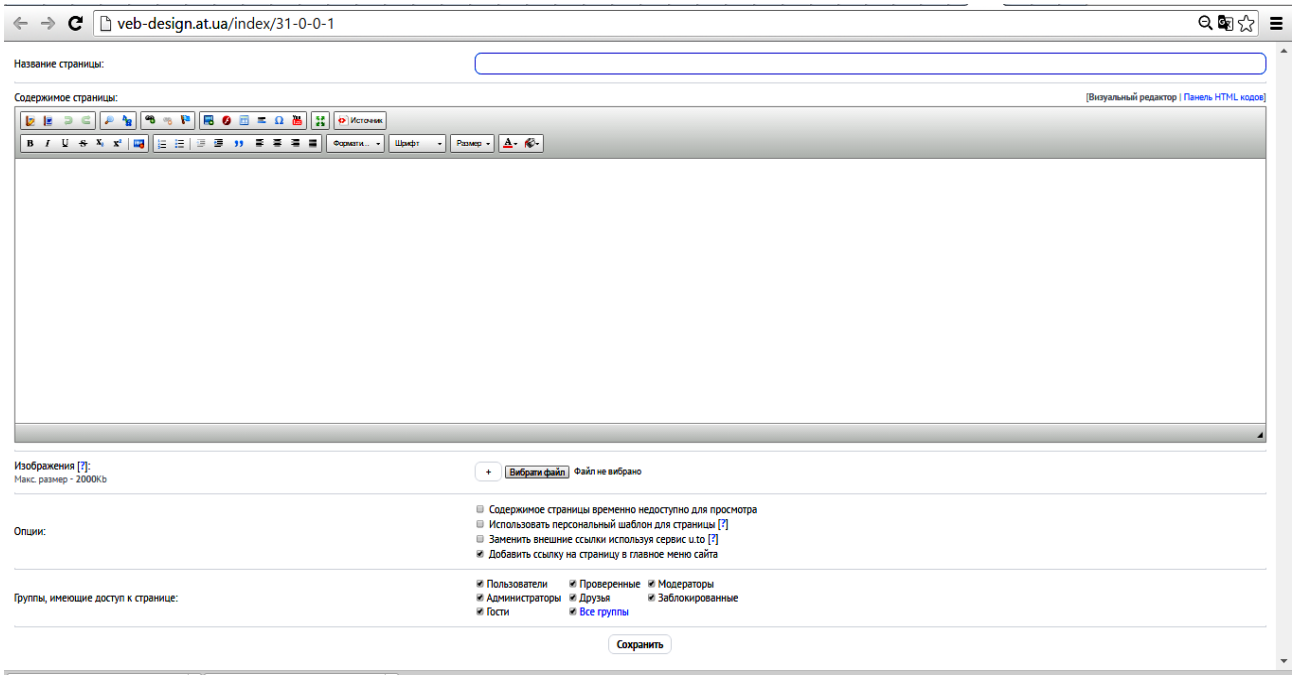
Докладний опис інструментів панелі керування описано за посиланням https://www.ucoz.ru/help/start/panel-upravlenija.

## 4. Публікація сайту

Сайт створено. Його можна відразу налаштувати та приступити до наповнення оригінальним авторським вмістом, при цьому публікація сайту відбуватиметься автоматично. У будь-який момент наповнення чи редагування сайту можна переглянути його вигляд, натиснувши на посилання адреси Вашого сайту. При цьому сайт відкриється у новій вкладці браузера.

Для додавання нової веб-сторінки слід спочатку зайти на сайт в ролі адміністратора за допомогою меню *Вход на сайт* / *Ввойти через* UID. Потім в меню керування сайтом скористатися командою *Редактор страниц* / *Добавить страницу*, після чого заповнити поля *Название страницы* і *Содержимое страницы*.



## 5. Робота з сайтом в сайт-конструкторі Ucoz

Для роботи з сайтом Ucoz пропонує кілька інструментів:

– конструктор, який дозволяє працювати безпосередньо зі сторінками сайтів у вікні браузера;
– редактор, який дозволяє працювати як з HTML і CSS-кодом сторінок, так і з кодом системи uCoz;
– візуальний WYSIWYG редактор, призначений для роботи з вмістом сайту без використання HTML-програмування.

Для редагування сайту потрібно увійти на сайт за допомогою логіна і пароля адміністратора, отриманого при створенні сайту.

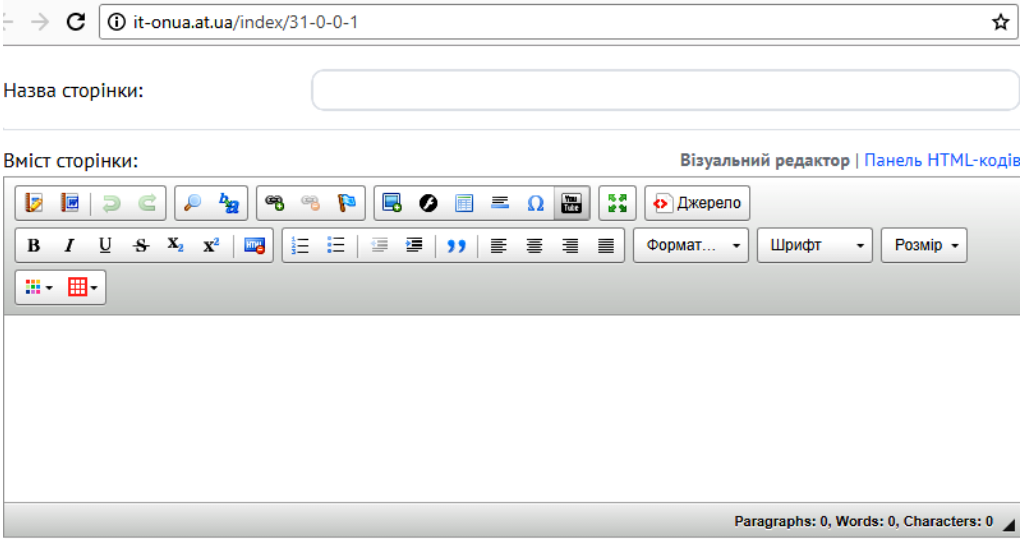
Для роботи з сайтом у режимі візуального редагування слід активувати режим роботи *«Конструктор»*. Зробити це можна з налаштувань панелі керування сайтом командою *Настройки / Общие настройки*, де встановити прапорець *Использовать «Конструктор» для управления дизайном сайта*.

Для змінення загальних даних треба зайти до панелі керування у розділ *Настройки / Общие настройки*. Далі можна змінити назву сайту, вибрати інший шаблон дизайну, налаштувати місцевий час і зробити багато чого іншого.

Використовуваний в системі uCoz **WYSIWYG-редактор** – це візуальний редактор, який відображає вміст у схожій з кінцевим виглядом формі. Основне завдання редактора – максимально спростити процес роботи з сайтом для людей, які не мають знань з HTML-програмування.

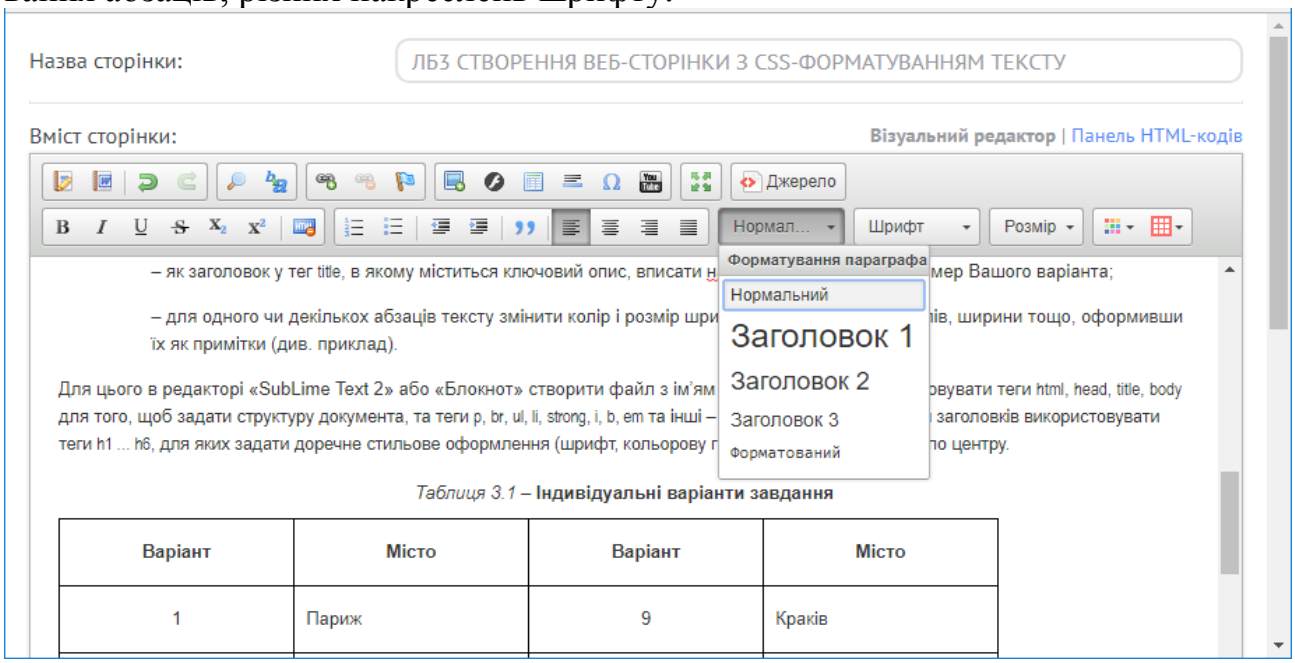
### 5.1. Робота з текстом у візуальному WYSIWYG-редакторі

Для додавання і редагування текстового наповнення веб-сторінок в Ucoz зручно використовувати візуальний редактор. Для цього слід спочатку зайти до панелі керування сайтом через відповідне посилання (у нашому прикладі – http://it-onua.at.ua/panel/#wpc-add-module). Далі виконати команду *Редактор страниц / Управление страницами сайта*, після чого можна як змінювати головну сторінку або додавати нову. Так, після натискання кнопки *Добавить страницу*, з'явиться вікно візуального редактора.



Далі текст можна вписувати на кшталт текстового редактора, але можна його й вставляти, копіюючи з інших джерел. Якщо джерелом є файл Word з форматованим відповідним чином текстом, списками, таблицями тощо, то для збереження параметрів форматування доцільно скористатись іконкою на панелі інструментів візуального редактора, після чого вставити скопійований фрагмент.

Візуальний редактор має доволі ширикий спектр засобів форматування. Так, для основного тексту доцільно використовувати нормальний стиль, крм того, існує три рівні заголовків. На панелі інструментів є іконки для вирівнювання абзаців, різних накреслень шрифту.




Візуальний редактор має доволі широкий спектр засобів форматування. Так, для основного тексту доцільно використовувати нормальний стиль, крім того, існує три рівні заголовків. На панелі інструментів є іконки для вирівнювання абзаців, різних накреслень шрифту.

Якщо треба швидко прибрати форматування для виокремленого фрагмента, слід скористатися іконкою .

Іконка дозволяє вставити спеціальні символи, наприклад: © або §.

Сформувати гіперпосилання з виокремленого тексту дозволить іконка .

Іконка призначена для створення і редагування якорів. Якір – це закладка з унікальним ім'ям на певному місці веб-сторінці, призначена для створення переходу до неї за посиланням. Якоря зручно застосовувати у документах великих розмірів, щоб можна було швидко переходити до потрібного розділу. Послідовність створення якоря може бути такою:

– поставити курсор на початок абзацу, на який треба організувати швидке переміщення;
– клацнути іконку *Вставити/редагувати якір*;
– ввести ім'я якоря та натиснути *ОК*, після чого на початку абзацу

з’явиться червоний прапорець (якір);
– далі перейти на місце, де треба організувати гіперпосилання для швидкого переходу на якір, виокремити відповідний текст і натиснути іконку *Вставити/редагувати посилання*;
– у полі *Тип посилання* вибрати *Посилання на якір у тексті* і вибрати потрібний якір.

У будь-який час можна переглянути HTML-код сторінки, клацнувши замість режиму візуального редактора на *Панель HTML-кодів*:

**Візуальний редактор** | Панель HTML-кодів

Після цього вікно перетвориться на HTML-редактор з тегами.

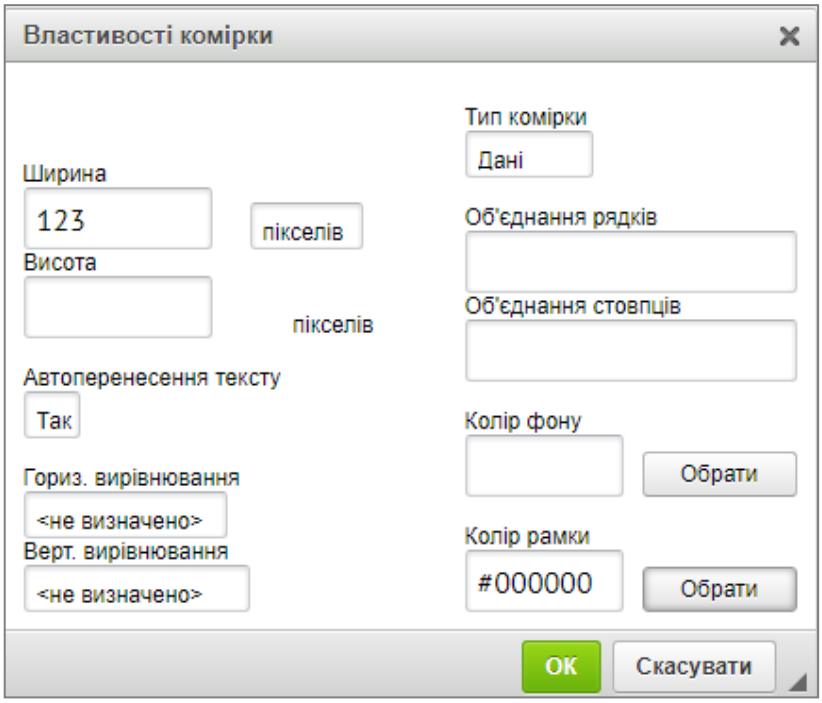
### 5.2. Робота з таблицями

Створити таблицю на веб-сторінці можна як копіюванням з файлу Word, якщо такий є, так і у візуальному редакторі за допомогою іконки *Таблиця*. Після натискання цієї іконки з’явиться діалогове вікно, в якому можна задати розміри (кількість стовпців і рядків) та інші параметри створюваної таблиці (відступи у комірках, вирівнювання тексту у них, колір та розміри меж, заголовки тощо).

Створену таким чином порожню таблицю слід заповнити даними.

Редагувати таблицю (додавати/видаляти/об’єднувати комірки, колонки, рядки тощо) можна, клацнувши по таблиці правою кнопкою миші та вибравши відповідну команду з контекстного меню.

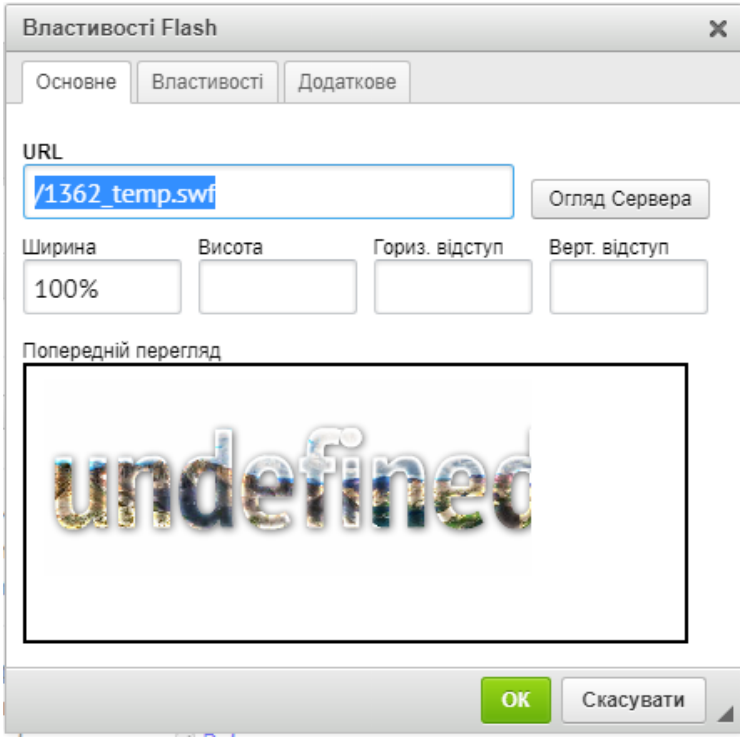
Для того, щоб змінити властивості комірки або всієї таблиці (змінити фон, колір меж тощо), слід виокремити потрібні комірки або всю таблицю, клацнути на них правою кнопкою миші та вибрати *Комірки / Властивості комірки*: Діалогове вікно *Властивості комірки* дозволить задати детальні параметри: ширину і висоту комірок у пікселях або відсотках, колір рамки (за замовчуванням від білий), тип комірок (дані або заголовок), ознаку автоперенесення тексту, вирівнювання тощо.




### 5.3. Робота з Flash

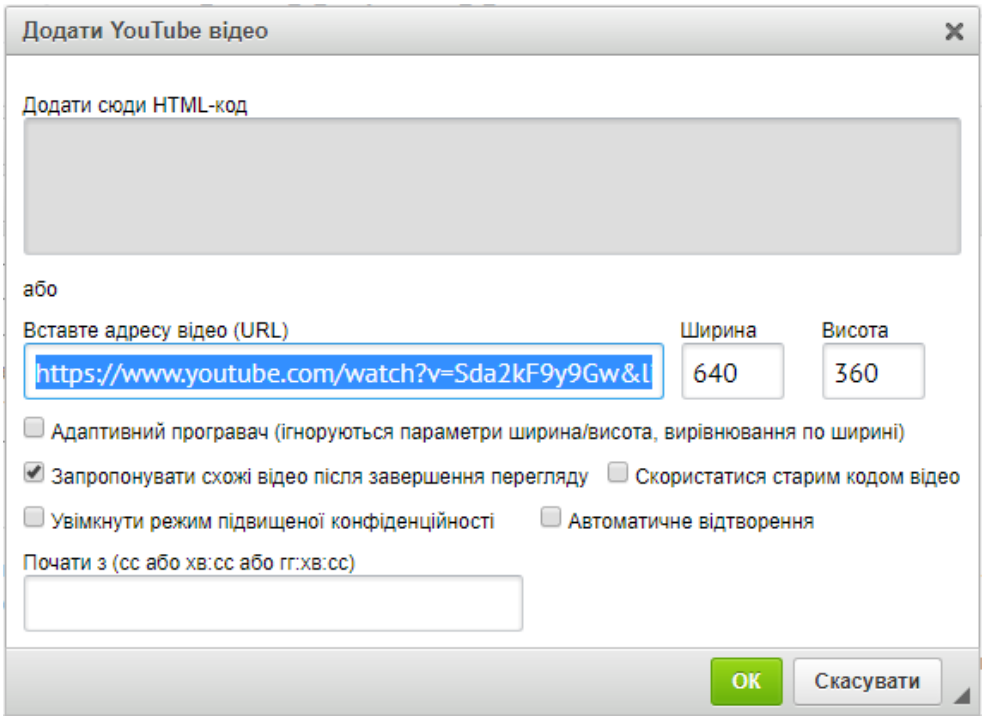
Додати flash-елемент на веб-сторінку дозволить іконка *Flash*. Натискання на неї спричиняє появу діалогового вікна *Властивості Flash* з трьома вкладками: *Основне*, *Властивості* і *Додаткове*. Натискання кнопки *Огляд сервера* до-

зволить вибрати та завантажити *swf*-файл flash-елемента. Додатково можна задати розміри та інші параметри елемента: масштабованість, вирівнювання, межі, режим відтворення тощо. У полі *Попередній перегляд* можна побачити сам flash-елемент.




### 5.4. Вставлення відео з YouTube

Вставити відео з сайту YouTube у візуальний редактор можна двома способами: 1) через вставлення HTML-коду або 2) посиланням на відео. Після натискання у візуальному редакторі на іконку *Додати YouTube відео* з'явиться вікно, в якому можна вставити URL-адресу відео та задати різноманітні опції його відтворення: показувати схожі відео після перегляду; включити режим підвищеної конфіденційності; автозапуск, ширину і висоту у пікселях тощо:



### 6. Поради дизайнерам-початківцям веб-сайтів

Головна помилка початківців дизайнерів – це акцентування уваги на малюванні екранів замість того, щоб думати про загальну картину (бізнес-завдання, сценарії взаємодії та очікування користувачів).

Грамотно побудований дизайн-процес позбавляє від типових помилок.

Для запобігання помилок взаємодії веб-дизайн сайту повинен відповідати користувачеві на два основних питання: «Як користуватися тим чи іншим інструментом на сайті для досягнення мети?» і «Чи працює інструмент так, як очікувалося?». І не варто при цьому розраховувати, що користувачі намагатимуться навчитися правильно працювати з сайтом, не отримав своїх відповідей, адже у них завжди є простий спосіб уникнути будь-яких проблем – піти на інший сайт

Завданням веб-дизайну є забезпечення зручної подачі інформації користувачеві сайту або веб-додатку, задоволення естетичного смаку аудиторії сайту.

Зробити перший ґрунтовний крок допоможе такий план:

1. **Постановка мети**: взяти існуючий сайт за основу або придумати новий, записати задачі так, ніби-то їх передав клієнт.

2. **Основні елементи**: вирішити, які блоки обов'язково будуть на сайті: меню, контент, фото тощо.

3. **Пошук зразків сайтів**: знайти, як мінімум, п'ять сайтів зі схожою структурою. Продумати, які сайти за структурою схожі на Ваш. Якщо Ваша мета зробити спортивний сайт, то можна брати за приклад новинний. Якщо це landing page, доречно заглянути на land-book.com.

4. **Проектування одного екрана**: за допомого ручки накидати на папері декілька варіантів розташування елементів. Тепер, коли у Вас є досвід, оскільки Ви вже переглянули багато схожих сайтів, Ви зможете вибрати один варіант, який, як Вам здається, краще виконуватиме головну задачу.

5. **Реалізація малюнка з виглядом екрана в Sketch або Photoshop**: при створенні зображення не треба відразу зациклюватися на кольорах. Можна спочатку зробити його чорно-білим, а потім додати акцентний колір для посилань і кнопок. Слід звертати увагу на розміри елементів, тексту і міжрядкові інтервали, оскільки погано зверстаний текст кидається в очі.

Дизайн – це талант? Якщо Ви маєте намір стати дизайнером, будьте готові, що перші роботи будуть жахливі. Доопрацьовуйте їх. Дивіться приклади і думайте, чому у інших вийшло, а у Вас – ні. Виправляйте. Знову практикуйтеся і коли-небудь Ви погляните на свою роботу і скажете: «Це круто».

*Самостійна робота № 8*

# ЗАМОВЛЕННЯ БЕЗКОШТОВНОГО ХОСТИНГУ

## Завдання

1. На кшталт того, як Ви реєструвалися і замовляли безкоштовний хостинг при виконанні п. 1 самостійної роботи № 2, Вам треба зареєструватися, замовити та отримати безкоштовний хостинг на www.ho.ua, але з іншої поштової адреси, оскільки з однієї поштової скриньки сервер не дозволить надати два безкоштовних хостинги. Тому, якщо у Вас поки що немає іншої поштової скриньки, слід попередньо її організувати. Наприклад, можна зареєструватися на gmail.com, i.ua, ukr.net або будь-якому іншому поштовому сервері. Надалі зареєстрований хостинг Вам буде потрібний при виконанні лабораторної роботи № 9.

2. Надати на перевірку викладачеві URL-адресу свого сайту.

*Лабораторна робота № 9*

# СТВОРЕННЯ БЛОГУ ЗАСОБАМИ WORDPRESS

***Мета:*** ознайомитися з засобами WordPress для створення власних блогів.

## Навчальні питання

1. Засоби створення блогу на WordPress.
2. Встановлення WordPress на хостинг.

## Завдання

1. Засобами сайт-конструктора WordPress створити власний блог. Для цього:
– скачати з сайту https://uk.wordpress.org/releases/ одну з версій WordPress (наприклад, файл wordpress-4.8.2-uk.zip);
– на сайті, створеному під час виконання самостійної роботи № 7, створити базу даних як місце, де зберігатиметься вся опублікована на сайті інформація (див. теоретичні відомості п. 6);
– встановити WordPress на свій хостинг (див. теоретичні відомості п. 7);.
2. Веб-адресу створеного блогу (сайту) надати на перевірку викладачеві.
3. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Що таке блог?
2. Яке призначення WordPress?
3. Що таке хостинг?
4. Якою є послідовність дій для створення блогу на WordPress?
5. Чому до вибору теми блогу слід ставитися серйозно і відповідально? Які кроки треба виконати, щоб правильно вибрати тему блогу?
6. Які поради є щодо вибору доменного імені?
7. Як вибрати надійний хостинг?
8. Для чого в WordPress потрібна база даних MySQL?
9. Який порядок створення бази даних у WordPress?
10. Як встановити WordPress на сервер?

## Теоретичні відомості

### 1. Основні поняття

**Блог** – це особливий тип веб-ресурсу, який характеризується як публічний онлайн-журнал, основним вмістом якого є систематичні записи, що містять в собі текст, зображення, відео, особисті фотографії.

Створення блогів є дуже і дуже популярним, хоча ще 10 років тому ніхто не чув і нічого не знав про слово «блог», яке прийшло до нас із Заходу. Зараз же, цим словом вже нікого не здивуєш. Особисті блоги ведуть багато популяр-

них журналістів, артистів, музикантів, спортсменів, політиків. За їх допомоги вони спілкуються зі своїми шанувальниками, розповідають про події з особистого життя, діляться своїми думками тощо.

За останні роки ведення блогу стало популярним й у простого люду. Люди заводять блоги і діляться розповідями про своє особисте життя, про подорожі, про свою роботу, про свої захоплення або навчають чогось інших. А деякі, крім ведення свого онлайн-журналу, ще й заробляють непогані гроші на ньому.

Існують численні інтернет-сервіси для створення сайтів як таких, так і блогів як різновиду сайту. Пропонуємо ознайомитись з засобами WordPress для створення блогів.

**WordPress** – безкоштовний, загальнодоступний конструктор сайтів, за допомогою якого можна створювати сайти «з нуля» й керувати їхнім вмістом без технічних навиків і знань. WordPress є системою керування вмістом сайту з відкритим вихідним кодом; написана на PHP; сервер бази даних – MySQL. Сфера застосування – від блогів до доволі складних новинних ресурсів і інтернет-магазинів. Вбудована система тем і плагінів разом з вдалою архітектурою дозволяє конструювати проекти широкої функціональної складності. WordPress іноді називають «серцем» або «движком» сайту.

Для використання WordPress слід завантажити з сайту https://uk.wordpress.org/releases одну з версій WordPress (наприклад, файл wordpress-4.8.2-uk.zip).

**Хостинг** (англ. *hosting*) – послуга надання ресурсів для розміщення інформації на сервер, що постійно перебуває у мережі (зазвичай інтернет).

Далі потрібно систему WordPress встановити на сервер. У цій лабораторній роботі ми її будемо встановлювати на сервер www.ho.ua, який надає послуги як платного, так і безкоштовного хостингу.

Але спочатку розглянемо теоретичні питання щодо можливої послідовності дій для створення власного блогу на WordPress, деякі з цих дій (3 і 4) Вами були уже виконані під час виконання самостійної роботи № 7.

## 2. Послідовність дій для створення блогу на WordPress

Для зручності розуміння процес створення блогу на WordPress розіб'ємо на кілька кроків:

1) вибір теми, якій буде присвячений блог;
2) замовлення тематичного домену;
3) вибір надійного хостингу;
4) створення бази даних;
5) встановлення WordPress на хостинг.

Виконавши ці прості 5 дій, можна створити свій особистий готовий блог. Залишиться лише зробити базові налаштування, встановити необхідні плагіни, створити сторінки і наповнити своє творіння корисним і якісним контентом.

## 3. Вибір теми, якій буде присвячений блог

Перш ніж робити свій особистий сайт, потрібно визначитися з його тематикою, оскільки саме від цього залежатиме світле майбутнє Вашого блогу. У цьому пункті надамо теоретичні поради щодо вибору теми блогу.

Тема вибирається один раз і назавжди, тому потрібно серйозно і відповідально поставитися до цього питання. Спочатку доцільно відповісти на кілька питань[1]:

– Яке у вас улюблене захоплення?
– В якій справі Ви є експертом?
– Чим би Ви щиро хотіли ділитися з іншими людьми?
– Що часто просять у Вас зробити друзі?
– Про що 70% життєвого часу Ви розмовляєте з усіма людьми?
– Що б Ви робили просто так, якщо Вам не потрібно було працювати заради забезпечення свого життя?

Відповіли самі собі на всі питання? Відмінно! Рухаємося далі. Коли у Вашого блогу буде висока відвідуваність і кілька тисяч постійних читачів, швидше за все Ви захочете його монетизувати. Тому потрібно обов'язково з'ясувати: чи користується попитом Ваша тема в інших людей?

А це, як Ви розумієте, найважливіший момент, тому що, якщо ви, наприклад, вибрали тему «розведення орангутангів у домашній оселі», то створювати блог на цю тему просто безглуздо. Людей, які цікавитимуться цим, просто не існує.

Тепер давайте проаналізуємо попит і популярність Вашої теми. Найкраще це робити декількома способами:

– пошукати в інтернеті форуми на обрану Вами тему (їхня присутність свідчить про популярність та актуальність);
– пошукати подібні тематичні групи у соціальних мережах, наприклад: відкрити Facebook або Twitter, в пошуку ввести ключові слова Вашої теми, оцінити кількість людей, які цікавляться цим напрямом, і виходячи з цього приймати рішення про звуження або розширення теми;
– пошукати подібні тематичні канали на YouTube;
– зробити аналіз популярних запитів для визначення ключових слів за різними показниками ефективності (кількість запитів, вартість за клік тощо) за допомогою таких онлайн-сервісів:
  – rGoogle Trends (https://trends.google.com/trends/explore?geo=UA);
  – rSerpstat (https://serpstat.com/uk/);
  – Buzzsumo (http://buzzsumo.com/);
  – SemRush.com (https://ru.semrush.com/);
  – Keyword Tool (http://keywordtool.io/ru).

Ці або інші подібні сервіси допоможуть віднайти топові публікації за заданими ключовими словами, визначити запити, які найчастіше задають користувачі інтернету з метою подальшого використання цих тем для написання статей блогу. Крім того, налаштування підписки в Google Alerts (https://www.google.com.ua/alerts) з оповіщеннями на теми і запити, які стосуються Вашої сфери, дозволить регулярно спостерігати за публікаціями конкурентів та тематичними порталами. Використання при цьому ефективного досвіду конкурентів сприятиме оптимізації веб-сторінок та виходу на топові позиції в результатах пошуку.

[1] Маслов И. Как создать блог на WordPress? Пошаговая инструкция от вебмастера! / Иван Маслов. [Электронный ресурс]. – Режим доступа: http://ivan-maslov.ru/kak-sdelat-sajt/kak-sozdat-blog-na-wordpress.html

Наприклад, для аналізу ключових запитів засобами сервісу Google Trends треба перейти за адресою https://trends.google.com/trends/explore?geo=UA (або https://trends.google.com/trends/hottrends#pn=p35) і ввести у поле назву вибраної Вами тематики. Також тут можна вибрати регіон та період часу, після чого натиснути кнопку пошуку. До речі, цей сервіс абсолютно безкоштовний, але потребує реєстрації. Сервіс є доволі зручним, оскільки дозволяє порівнювати водночас популярність декількох тем. Наведемо приклад аналізу тем популярних запитів. Так, серед запитів «політичний блог», «журналістське розслідування», «інфографіка» лідером є саме останній із запитів. Проте, і цей запит суттєво відстає від популярності запитів: «котики», «субсидії», «спорт», «HTML», «погода». При чому тут ці запити, як і попередні, розташовані саме у порядку зростання їх популярності. Це дозволить підібрати більш вдалу тематику блогу (сайту).

### 4. Замовлення тематичного домену

Отже, тему майбутнього інтернет-проекту вибрано. Тепер треба підібрати домен. Вибір домену для свого блогу – заняття цікаве і творче. Дамо лише кілька порад щодо вибору доменного імені. Домен має бути: бажано коротким, легко запам’ятовуватись і наочно характеризувати тематику блогу чи сайту.

Доменне ім’я свого сайту для виконання цієї лабораторної роботи Ви вже задали, коли замовляли безкоштовний хостинг на www.ho.ua при виконанні самостійної роботи № 7. Продовжуємо розбирати питання: як створити блог на WordPress і переходимо до наступного кроку.

### 5. Вибір надійного хостингу

Вибір хостингу – не менш важливий і серйозний пункт при створенні блогу на WordPress. Фахівці радять: ні в якому разі не публікувати свій веб-ресурс на безкоштовному хостингу. Це стосується блогів (і сайтів), які Ви плануєте використовувати як довгострокові проекти (а не лише як ця лабораторна робота). Якщо в майбутньому Ви не хочете проблем, то слід розміщувати свій блог відразу на хорошому і перевіреному хості.

Для цього завдання Ви завчасно зареєструвалися та отримали безкоштовний хостинг на www.ho.ua та доменне ім’я сайту під час виконання самостійної роботи № 7.

Якщо у браузері ввести в адресному рядку URL-адресу здобуту при виконанні самостійної роботи № 7 сайту, про що Вам було надіслано лист на поштову скриньку (в нашому прикладі це http://lozindre.ho.ua/), то відкриється поки що порожній сайт такого вигляду:

## 6. Створення бази даних на сайті

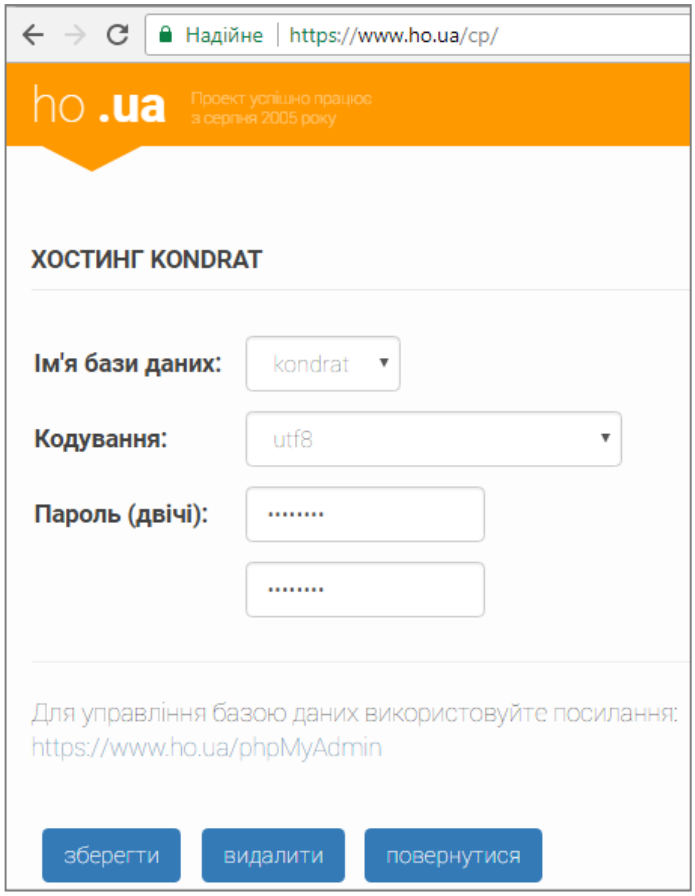
Для роботи блогу у системі WordPress потрібна база даних MySQL. Напевно у Вас відразу виникли питання: що таке база даних і навіщо вона потрібна?

**База даних** – це місце, де фіксується і зберігається вся опублікована інформація на сайті (паролі, тексти, сторінки, коментарі тощо). Отже, WordPress – це серце сайту, а ось база даних – це, абстрактно кажучи, мізки сайту.

Оскільки попередні пункти носили здебільшого теоретичний характер, зараз ми детально по пунктах розглянемо порядок створення бази даних на сайті.

1) Треба ввести URL-адресу https://www.ho.ua/cp/, ввести логін і пароль (див. лист, який Вам надіслали; в нашому прикладі логін *lozindre*), щоб увійти з правами адміністратора сайту.

2) Зайти до розділу *Управління базою даних*, натиснувши кнопку *База даних* (див. рис. на с. 56).




3) Далі відкриється вікно, в якому ввести придумане ім'я бази даних, задати кодування utf8, двічі ввести придуманий пароль і натиснути кнопку *Зберегти* для створення бази даних на своєму сайті. На інших хостах створення бази даних виглядає подібним чином.

4) Якщо параметри бази даних Вас задовольняють, натиснути кнопку *ОК*, а якщо ні – натиснути гіперпосилання поряд з *Управління базою*. Оскільки змінити опції можна буде і на подальших кроках, зупиняємось на цьому і натискаємо кнопку *Зберегти*. Після цього буде сформовано вікно на кшталт показаного праворуч.

На цьому базу даних успішно створено. Далі можливо, залежно від сервера, треба буде зачекати (до 10 хвилин) поки база підключиться.

## 7. Встановлення WordPress на хостинг

Спочатку на сайті https://uk.wordpress.org/releases/ треба скачати одну із версій WordPress (наприклад, файл wordpress-4.8.2-uk.zip). Далі послідовність дій може бути такою:

1) Розпакувати архів wordpress-4.8.2-uk.zip. Результат – папка з назвою *wordpress* з вкладеними файлами і папками.

2) Найти у папці *wordpress* файл wp-config-sample.php і відкрити його у будь-якому редакторі, щоб редагувати код. Наприклад, можна редагувати цей файл у текстовому редакторі «Блокнот», але після редагування слід задати кодування (при зберіганні файлу) UTF-8 (див. наступний рисунок).

Зараз відредагуємо файл в такий спосіб:

а) Найти рядок

define('DB_NAME', 'database_name_here')

і замість 'database_name_here' вставити ім'я своєї бази даних (у нашому прикладі 'lozindre'). Вийде так:

define('DB_NAME', 'lozindre');

б) Найти рядок

define('DB_USER', 'username_here');

і замість 'username_here' вставити ім'я користувача ('lozindre'). Вийде:

define('DB_USER', 'lozindre');

в) Найти рядок

define('DB_PASSWORD', 'password_here');

і замість 'password_here' вставити свій пароль, наприклад:

define('DB_PASSWORD', 'GEWfeMUPrl');

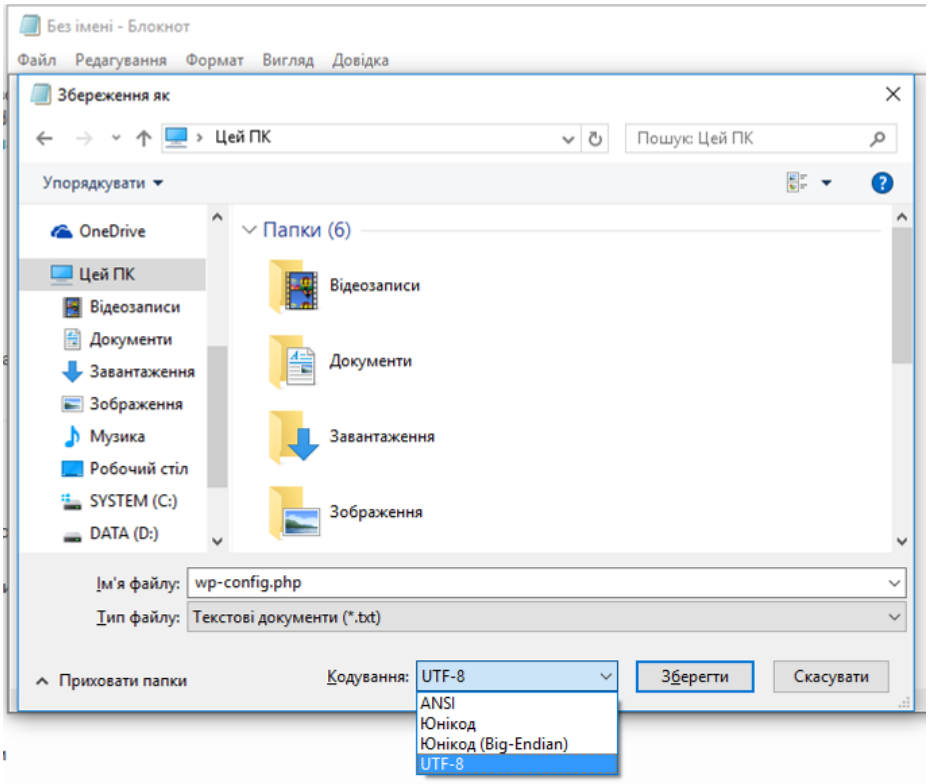
г) Найти рядок

define('DB_HOST', 'localhost');

і замість 'localhost' вставити сервер баз даних 'db2.ho.ua'. Вийде:

define('DB_HOST', 'db2.ho.ua');

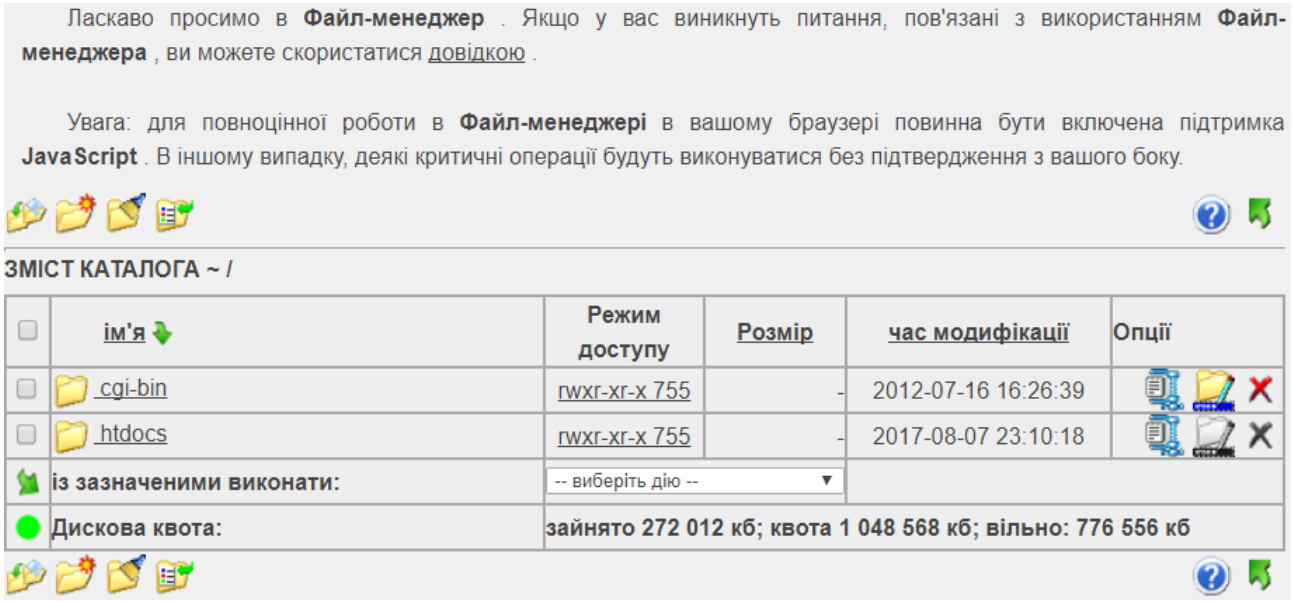
д) Зберегти цей файл як **wp-config.php**. Нагадуємо, що треба вибрати кодування UTF-8.




Далі завантажити цей дистрибутив на сервер та виконати встановлення.

3) Для цього спочатку треба заархівувати всі вкладені файли і папки всередині папки *wordpress*. В нашому прикладі ми створили архів з ім'ям wordpress.zip (суттєвим є використання саме *zip*-формату архіву).

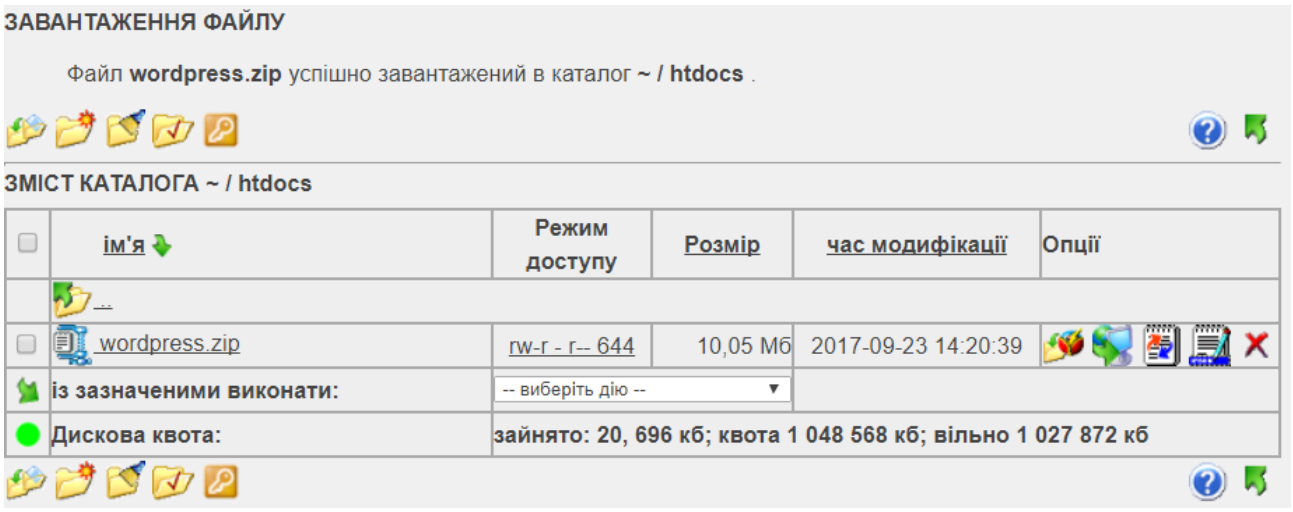
4) Далі треба повернутися на сторінку https://www.ho.ua/cp/, де натиснути кнопку «*Управління файлами*» *Зміна вмісту сайту.* – Відбудеться перехід до файлового менеджера.



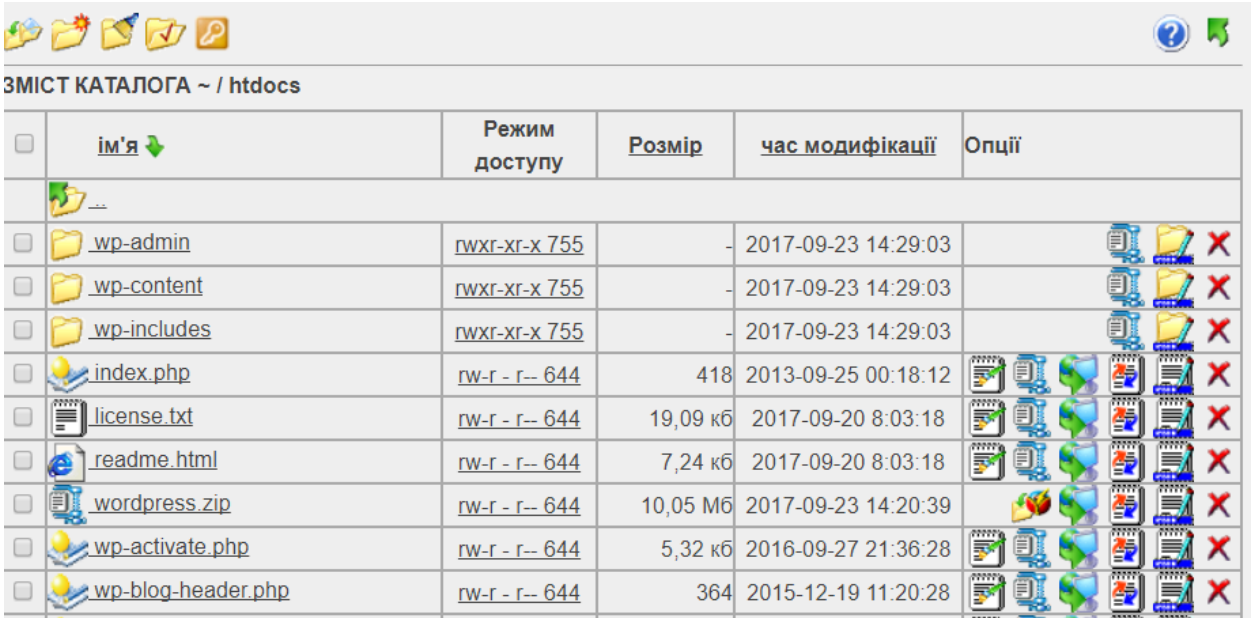

5) Увійти до папки *htdocs*, натиснувши на гіперпосилання *htdocs*. Виокремити та видалити усі файли. Підтвердити видалення у відповідному повідомленні. Подібні дії Ви виконували під час самостійних робіт № 2 і № 5.

6) Натиснути на значок  *Завантажити файл у каталог* (підказка з назвою значка з'явиться при наведенні вказівника миші на значок лівої нижньої папки).

7) Клацнути кнопку *Вибрати файл* та найти файл архіву wordpress.zip. Далі вибраний файл завантажити, натиснувши кнопку *Звантажити.*



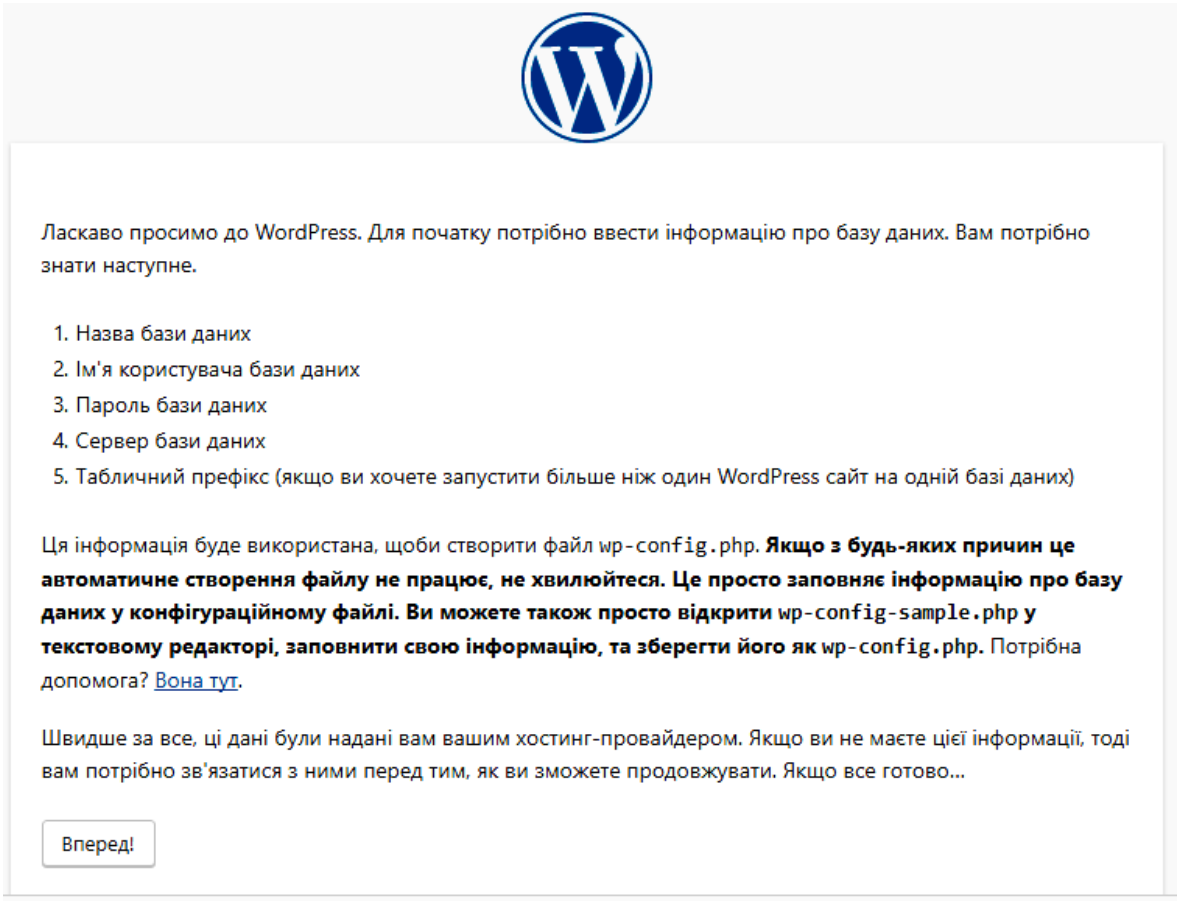
8) Щоб розпакувати архів, треба праворуч файлу wordpress.zip клацнути мишею папку жовтого кольору *Розпакувати архів у поточний каталог*.



ЗМІСТ КАТАЛОГА ~ / htdocs

| ім'я | Режим доступу | Розмір | час модифікації | Опції |
|---|---|---|---|---|
| ... | | | | |
| wp-admin | rwxr-xr-x 755 | - | 2017-09-23 14:29:03 | |
| wp-content | rwxr-xr-x 755 | - | 2017-09-23 14:29:03 | |
| wp-includes | rwxr-xr-x 755 | - | 2017-09-23 14:29:03 | |
| index.php | rw-r - r-- 644 | 418 | 2013-09-25 00:18:12 | |
| license.txt | rw-r - r-- 644 | 19,09 кб | 2017-09-20 8:03:18 | |
| readme.html | rw-r - r-- 644 | 7,24 кб | 2017-09-20 8:03:18 | |
| wordpress.zip | rw-r - r-- 644 | 10,05 Мб | 2017-09-23 14:20:39 | |
| wp-activate.php | rw-r - r-- 644 | 5,32 кб | 2016-09-27 21:36:28 | |
| wp-blog-header.php | rw-r - r-- 644 | 364 | 2015-12-19 11:20:28 | |

9) Тепер лишилося закрити вікно файл-менеджера, а в новій вкладці браузера набрати адресу свого сайту (у нашому прикладі lozindre.ho.ua) і натиснути *Enter*. Якщо з'явиться порожнє вікно, як при запуску у п. 5, слід оновити сторінку, натиснувши клавішу *F5*.



Ласкаво просимо до WordPress. Для початку потрібно ввести інформацію про базу даних. Вам потрібно знати наступне.

1. Назва бази даних
2. Ім'я користувача бази даних
3. Пароль бази даних
4. Сервер бази даних
5. Табличний префікс (якщо ви хочете запустити більше ніж один WordPress сайт на одній базі даних)

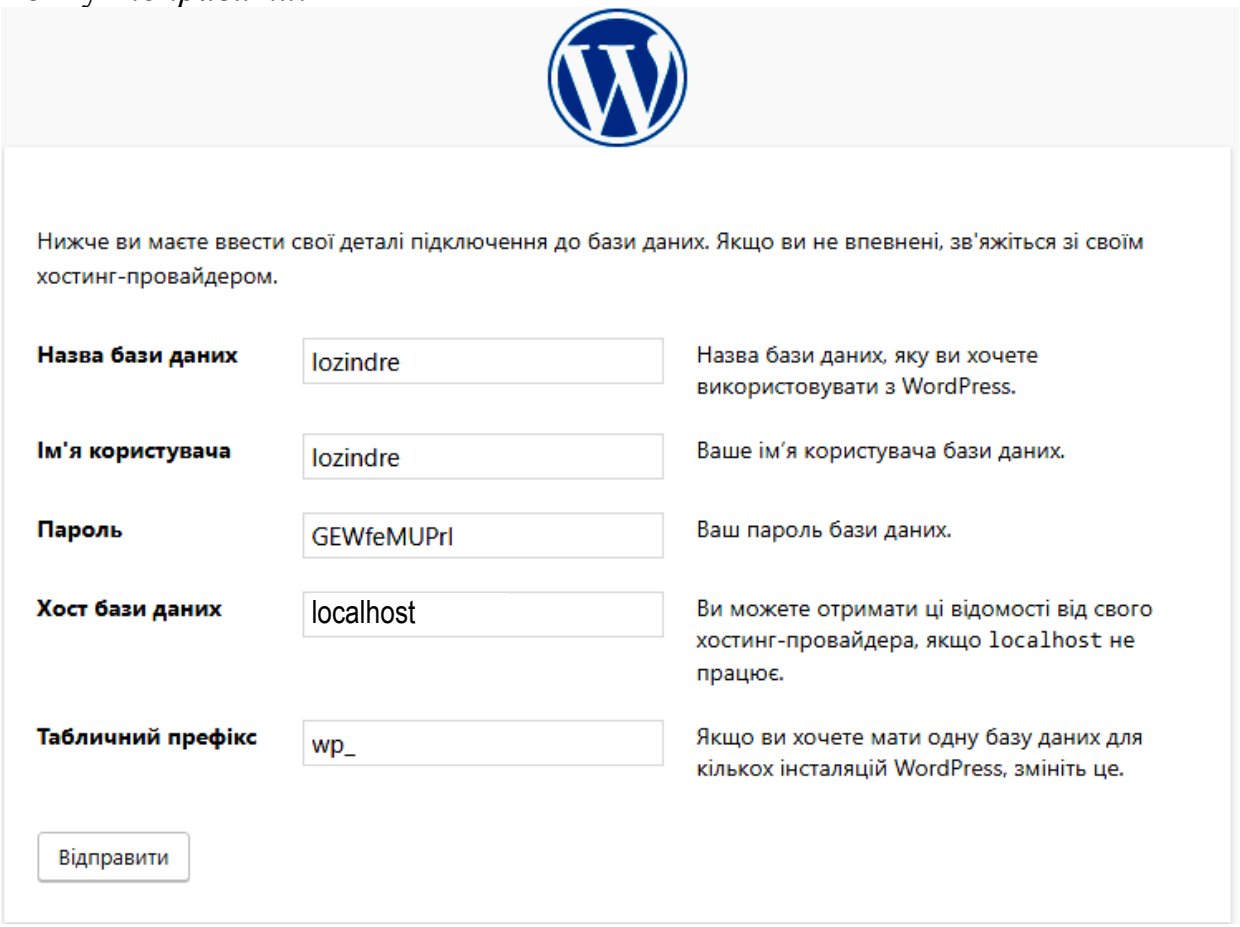
Ця інформація буде використана, щоби створити файл `wp-config.php`. **Якщо з будь-яких причин це автоматичне створення файлу не працює, не хвилюйтеся. Це просто заповняє інформацію про базу даних у конфігураційному файлі. Ви можете також просто відкрити `wp-config-sample.php` у текстовому редакторі, заповнити свою інформацію, та зберегти його як `wp-config.php`.** Потрібна допомога? Вона тут.

Швидше за все, ці дані були надані вам вашим хостинг-провайдером. Якщо ви не маєте цієї інформації, тоді вам потрібно зв'язатися з ними перед тим, як ви зможете продовжувати. Якщо все готово...

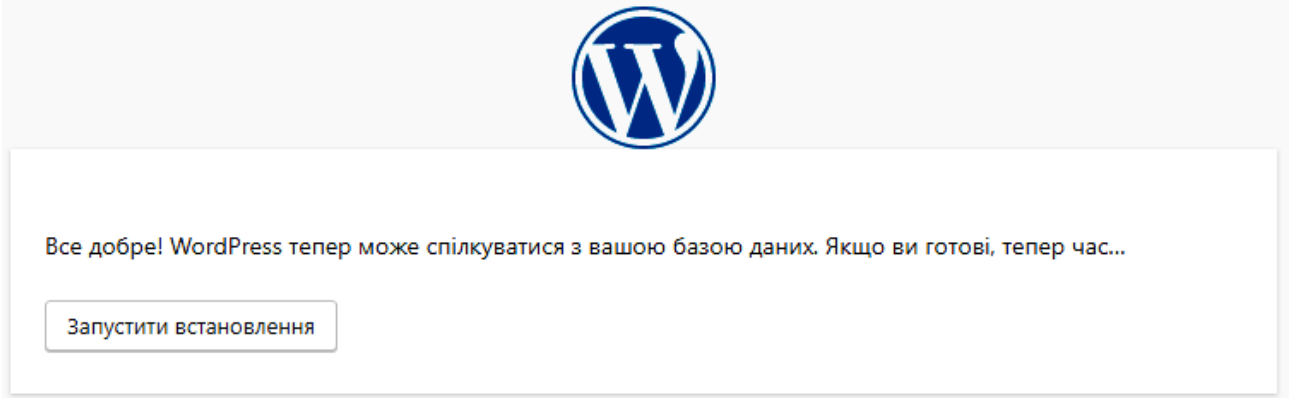
Вперед!

Натиснути кнопку *Вперед*.

10) Далі треба заповнити поля з назвою майбутнього сайту (наприклад, «Сайт журналіста») та іншими даними адміністратора. Наприкінці натиснути кнопку *Відправити*.




Далі з'явиться вікно, в якому натиснути кнопку *Запустити встановлення*.




Далі заповнити поля для реєстрації (назву сайту, ім'я користувача, пароль, email тощо) та натиснути кнопку *Встановити WordPress*.

Це все, *WordPress* встановлено. Можна увійти, зазначивши email та пароль.

На наступному занятті Ви будете займатися дизайном та наповненням створеного блогу. А поки що Ви можете найти в інтернеті різні сайти – блоги журналістів, щоб створити собі уявлення про можливі варіанти їх конструювання, наприклад: https://volodymyrboyko.com/.

## 8. Налаштування теми та структури блогу на Wordpress з можливою реєстрацією на wordpress.com

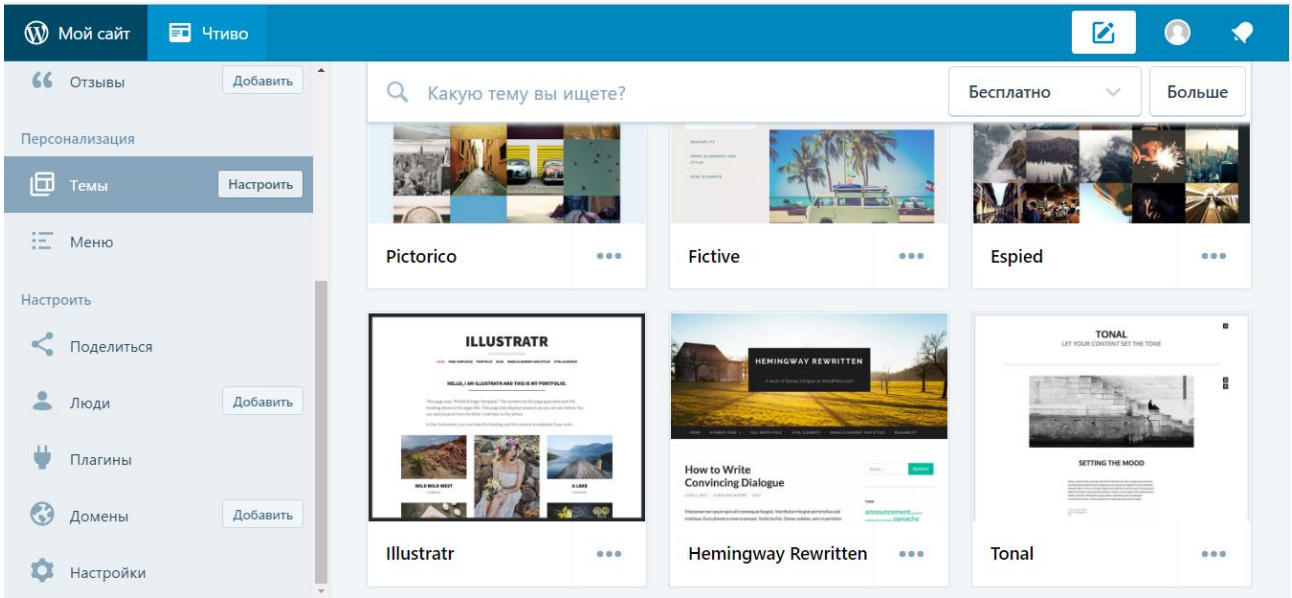
Крім встановлення Wordpress на певний хостинг, зареєструватися та отримати безкоштовний хостинг можна і на сайті https://wordpress.com. Для цього слід зайти на цей сайт і натиснути кнопку *Приступайте* (*Get Started*). На осінь 2017 року сервіс надає можливість вибрати одну з 18 мов інтерфейсу. На жаль, української серед них немає, але можна вибрати російську.

Далі пропонується 5 кроків для вибору спрямованості сайту або блогу:

1) оскільки завдання лабораторної роботи полягає у створенні блогу, натиснути кнопку *Начать с блога*;
2) вибрати одну із популярних тем (згодом після реєстрації можна буде вибрати іншу тему);
3) підібрати домене ім'я, тобто IP-адресу майбутнього сайту, або вказати адресу свого зареєстрованого хостингу, якщо таке є (здобуте при виконанні самостійної роботи № 7;
4) вибрати тарифний план (у нашій лабораторній роботі – безкоштовний);
5) вказати власне ім'я, адресу поштової скриньки та придумати пароль.

Далі на Вашу поштову скриньку прийде лист з посиланням для підтредження реєстрації.

Після цього можна приступити до налаштування відображення блогу та його наповнення. Наприклад, можна вибрати тему, створити меню певної структури, зайти в *Настройки* і задати імя. До речі, якщо натиснути на кнопку *Настроить* поряд з командою меню *Темы* у групі *Персонализация*, можна задати докладно різні налаштування, у тому числі і власний дизайн.



*Самостійна робота № 9*

# НАПОВНЕННЯ БЛОГУ, СТВОРЕНОГО ЗА ДОПОМОГОЮ WORDPRESS

## Завдання

1. Наповнити сконструйований під час виконання лабораторної роботи № 9 блог тематичними матеріалами (текстовими, графічними, відео матеріалами), відповідно до назви блогу.
2. Зазначити своє авторство, наприклад: «Блог Коваленко І.П.».
3. Надати на перевірку викладачеві URL-адресу свого блогу.

## Теоретичні відомості

### Типова анатомія блогу

Вміст блогу можна подібний до стрічки, на якій у хронологічному порядку згідно з датами публікації йдуть дописи блогера, так звані пости, один за одним. Оскільки з часом у блозі накопичується багато постів (зазвичай вони займають кілька веб-сторінок), то найновіший пост розміщується вгорі першої сторінки, а давніші переміщуються вниз. Наприклад, на першій сторінці розміщуватимуться пости цього тижня, друга сторінка присвячена постам за минулий тиждень, третя сторінка ще давнішим і так далі. Зазвичай сторінки блогу також містять посилання на архів блогу, тобто на попередні пости, згруповані за місяцями і роками. Отже, навігація по блогу в хронологічному порядку є доволі легкою.

Окрім того, у багатьох системах блогування можливо призначати категорії постам. Ці категорії відбивають тематику постів, як наприклад: «подорожі», «поетика», «сімейні справи» тощо. Тоді відвідувачі блогу, які цікавляться думками блогера щодо програмування, можуть за посиланням на цю категорію перейти до всіх наявних постів автора, присвячених цьому.

Типово окремий має свій заголовок, дату публікації, власне, зміст, який складається з гіпертексту (думки автора, цитати тощо), посилань на інші сайти та блоги в інтернеті, інколи зображень та відео. Також пост містить коментарі до нього, залишені відвідувачами та просту веб-форму, за допомогою якої вони долучають ці коментарі.[1]

[1] Блог. [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Блог.

# ТЕМА 3
# ЕЛЕМЕНТИ ВЕБ-ДИЗАЙНУ

*Лабораторна робота № 10*
## ДИЗАЙН САЙТУ ЗАСОБАМИ ПЛАГІНІВ

***Мета:*** ознайомитися з призначенням плагінів, навчитися їх встановлювати у WordPress і використовувати для дизайну та оптимізації для власних сайтів.

### Навчальні питання

1. Використання тем (шаблонів) WordPress для налаштування структури та зовнішнього подання сайту.
2. Використання плагінів.
3. Додавання форуму на сайт WordPress.

### Завдання

1. Для сайту (блогу), створеного на лабораторній роботі № 9 у сайт-конструкторі WordPress, вибрати одну із тем (шаблонів) для налаштування структури та зовнішнього подання сайту (тему вибрати та скачати з офіційного сайту WordPress https://wordpress.org/themes/browse/popular/) (див. теоретичні відомості п. 1).

2. Заповнити сайт декількома публікаціями (при цьому можуть використовуватись Ваші публікації минулих років, матеріали власних курсових робіт тощо), скориставшись на панелі навігації командою *Записи / Додати* (див. теоретичні відомості п. 2).

3. З офіційного сайту WordPress (https://wordpress.org/plugins/wptouch) скачати та встановити плагін *wptouch*, який є найпопулярнішим доповненням для створення мобільної версії Вашого WordPress-сайту (див. теоретичні відомості п. 3).

4. Для SEO-оптимізації встановити й активувати плагіни *Google XML Sitemaps* (https://wordpress.org/plugins/google-sitemap-generator), *All in One SEO Pack* (https://wordpress.org/plugins/all-in-one-seo-pack) (див. теоретичні відомості п. 4).

5. Створити форум на своєму сайті, скориставшись плагіном *bbPress* (див. теоретичні відомості п. 5). Заповнити форум декількома повідомленнями. Налаштувати відображення повідомлень.

6. Веб-адресу створеного блогу (сайту) надати на перевірку викладачеві.

7. Підготувати відповіді на контрольні запитання.

### Контрольні запитання

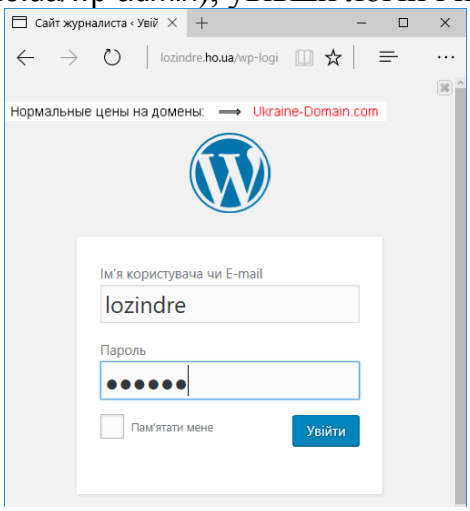
1. Яке призначення WordPress?
2. Які типи ролей для користувачів доступні на платформі WordPress?
3. Для чого використовуються готові макети (шаблони) WordPress?
4. Що розуміють під темою в WordPress?
5. Як додати нові теми у певний каталог на сервері WordPress?

6. Яке призначення плагінів у WordPress?
7. Що таке плагін для WordPress?
8. Як додати новий плагін на WordPress?
9. Описати інтерфейс WordPress.
10. Яка структура вікна *Майстерня* сайту WordPress?
11. Що таке SEO-оптимізація?
12. Яка основна мета SEO?
13. Як на сайті у WordPress створити форум?

## Теоретичні відомості

### 1. Використання тем (шаблонів) WordPress для налаштування структури та зовнішнього подання сайту

Зайти на свій сайт, створений на попередній лабораторній роботі, з правами адміністратора (до URL-адреси свого сайту дописується /wp-admin/, в нашому прикладі URL-адреса lozindre.ho.ua/wp-admin), увівши логін і пароль:

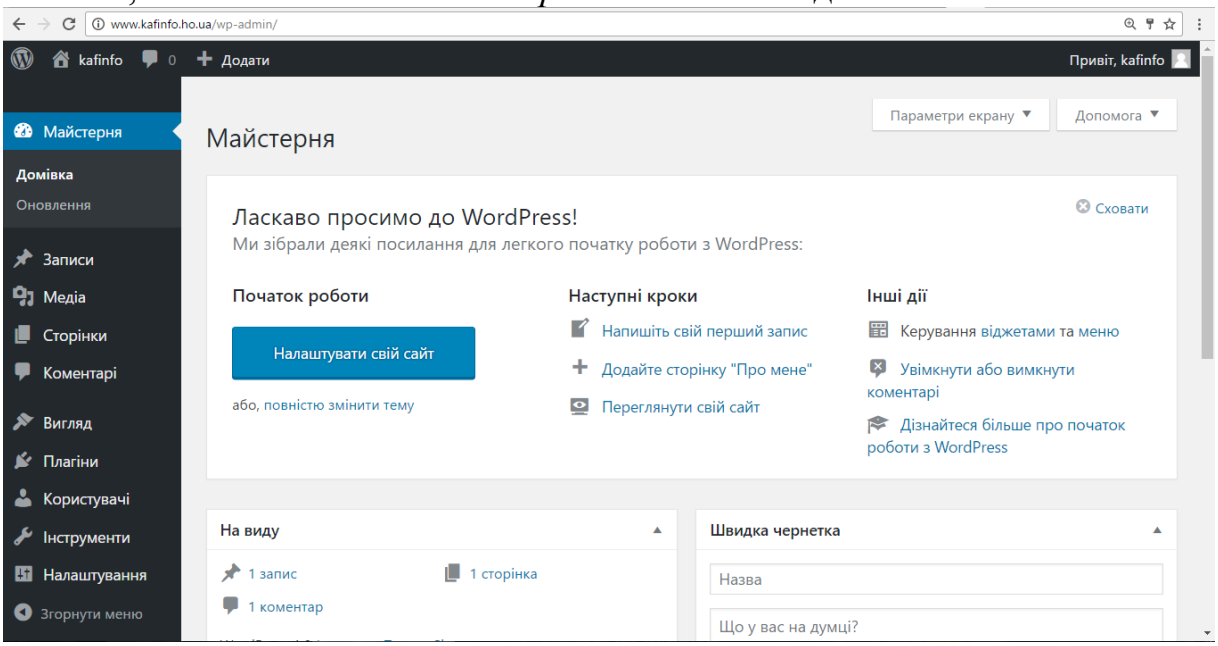



З'явиться *Майстерня* для оформлення сайту (вона завжди з'являється при вході на сайт з правами адміністратора). До речі, на платформі WordPress визначено п'ять типів ролей, доступних для користувачів:

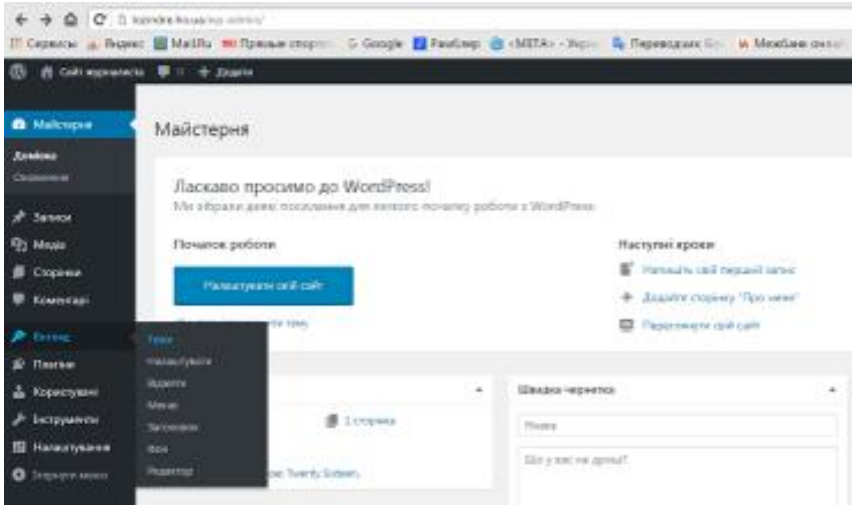
– адміністратор (administrator) з усіма можливими повноваженнями;
– редактор (editor), який володіє правами адміністратора, за винятком повноважень для внесення змін у конфігурацію веб-сервера;
– автор (author), який створює і публікує власні матеріали (пости);
– учасники (contributor) можуть створювати власні записи, але не мають права публікувати їх самостійно;
– передплатник (subscriber) може тільки читати записи у блозі і залишати коментарі.

Не рекомендується використовувати обліковий запис адміністратора для щоденної роботи, щоб уникнути компрометації системи. Більш правильним буде створити дублюючий обліковий запис з менш широкими правами і використовувати його.

Отже, початково вікно *Майстерні* матиме вигляд:




При наведенні на пункт меню *Вигляд* відкриється підменю, в якому нас цікавить команда *Теми*:

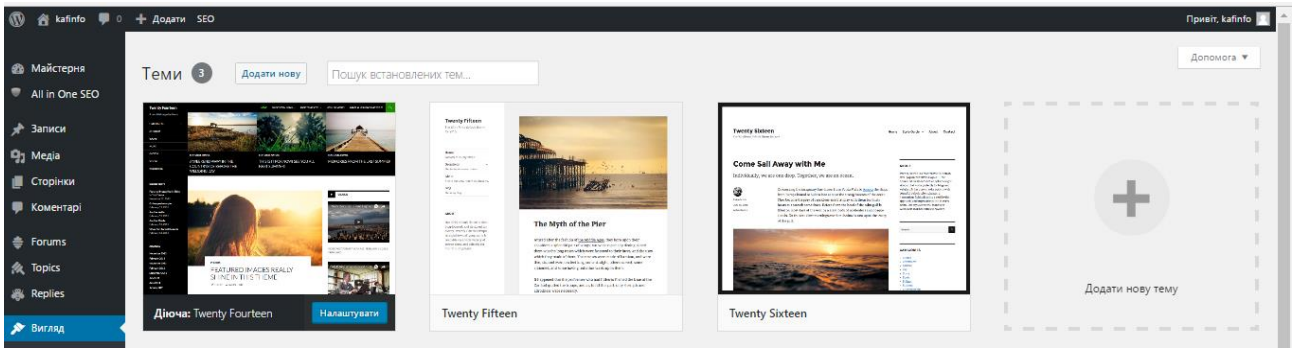



Команда *Теми* (або *Шаблони WordPress*) – це готові макети для популярної системи керування контентом, використовуваної для ведення блогів, новинних сайтів і проектів для електронної комерції. Ці шаблони можуть використовуватись для створення веб-проекту «з нуля» або модернізації вже існуючого.

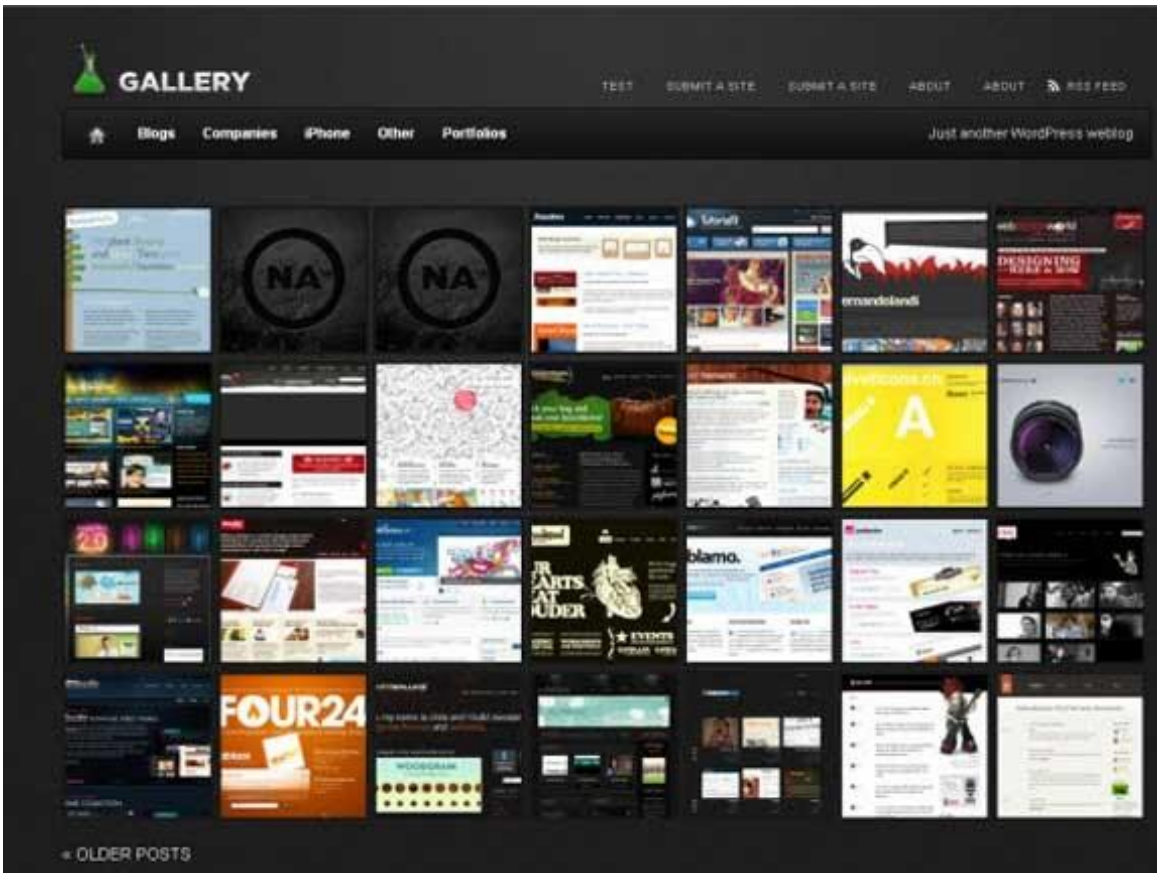
Під **темою** в WordPress розуміються файли (або шаблони), які дозволяють змінювати графічний інтерфейс і стиль відображення вмісту сайту без внесення будь-яких змін у програмний код. Тема складається з файлів шаблонів, зображень (**.jpg*, **.gif*), каскадних таблиць стилів (**.css*) і файлів з PHP-кодом (**.php*).

Вдало вибрана для блогу тема (шаблон оформлення) WordPress дозволить залучити трафік та аудиторію, надати привабливості і, навіть, набути престижності. Шаблони влаштовані таким чином, що дозволяють за потреби змінювати структуру або повністю оформлення сайту в будь-який час, за настроєм або відповідно до тематики ресурсу. Цей процес не займе багато часу.

Зокрема, командою *Теми* можна викликати заздалегідь заготовлені теми для WordPress, за допомогою яких можна змінити дизайн сайту.




Умережі існують численні варіації тем для сайтів, які без зусиль можна скачати і встановити на власний сайт, форум, блог. На сьогодні, тільки на офіційному сайті WordPress у папці Theme Directory (https://wordpress.org/themes/browse/popular/) налічується понад 2600 тем для вільного скачування. Після додавання нових тем у певну папку на сервері, вони з'являться у розділі *Теми*, як показано на рисунку:




Файли кожної нової теми містяться в окремій папці з відповідною назвою. Щоб тему можна було використовувати, вона повинна мати певний набір файлів:

– style.css – головний файл таблиці стилів;
– index.php – головний файл шаблонів;
– comments.php – шаблон коментарів;
– home.php – шаблон головної сторінки.

Для встановлення теми достатньо просто скопіювати її файли у папку ***themes*** або скористатися адміністративним інтерфейсом WordPress.

Побачити свій сайт у тому вигляді, як його бачитимуть відвідувачі, можна командою *Сайт журналіста* у верхньому лівому куті екрана. Зазначимо, під час вибору дизайну сайту доцільно періодично переглядати його вигляд на іншій вкладці браузера, оновлюючи її після того чи іншого змінення вигляду сайту.

При наведенні миші на команду *Сайт журналіста* відкриватиметься підменю, за допомогою якого можна знову повернутися до *Майстерні* для подальшого редагування сайту.

### 2. Використання плагінів

Отже, WordPress – зручна платформа для створеннґ веб-сайтів, яка користується популярністю серед розробників. Але, на жаль, її базова функціональність обмежена і може запропонувати розробнику тільки стандартний набір опцій. Тому для розширення можливостей WordPress використовуються спеціальні модулі (плагіни), які дозволяють модифікувати базову систему для створення сайтів, що відповідатимуть специфічним вимогам того чи іншого проекту.

**Плагін[1] для WordPress** – це невелика програма, яка встановлюється в систему у вигляді окремого блоку і допомагає розширити її можливості. Простіше кажучи, це доповнення до базової програми, метою якого є поліпшення функціональності веб-сайту або впровадження додаткових можливостей. Основною перевагою плагінів є те, що при їх впровадженні не потрібно вносити зміни в базову програму. Крім того, при оновленні базової системи плагіни не чіпають і вони функціонуватимуть в колишньому режимі.

Модулі є корисним доповненням до WordPress, оскільки вони не тільки використовуються для наповнення сайту інформацією, а й дозволяють розширити можливості основної системи. Більшість розробників не можуть обійтися тільки стандартними функціями WordPress, тому існує безліч плагінів для вирішення найрізноманітніших завдань. Але не варто забувати про той факт, що надмірне навантаження системи різними плагінами може серйозно позначитися на її продуктивності. Тому треба намагатися вибирати найбільш оптимальні реалізації плагінів, які відповідатимуть заданим вимогам, але не будуть надавати негативного впливу на роботу системи в цілому.

Отже, щоб встановити плагін, його спочатку треба скачати. Наприклад, можна скачати плагін *wptouch* (архівний *zip*-файл) з офіційного сайту WordPress (https://wordpress.org/plugins/wptouch/).

Після цього слід зайти у *Майстерню* (нагадуємо, *Майстерня* завжди відкривається при вході з правами адміністратора, у нашому прикладі, при введенні URL-адреси – lozindre.ho.ua/wp-admin/).

---

[1] **Плагін** (від англ. *plug-in* – підключати) – незалежно компільований програмний модуль, динамічно долучений до основної програми і призначений для розширення і/або використання її можливостей. Просто кажучи, плагін – це доповнення (розширення можливостей) для будь-якої програми на Вашому комп'ютері або движку сайту в інтернеті.


Виконати команду *Плагіни* / *Додати* і клацнути кнопку *Завантажити плагін*. Далі клацнути по меню *Плагіни* й активувати цей плагін за допомогою кнопки *Активувати*. Вибрати файл плагіна wptouch.4.2.5.zip за допомогою кнопки *Вибрати файл* і встановити його за допомогою кнопки *Встановити*.

### 3. Опис інтерфейсу WordPress

При вході в систему WordPress першою сторінкою є адміністративна *Майстерня*, на яку виводиться інформація про стан блогу – кількість коментарів, оновлення, новини WordPress. З цієї майстерні можна швидко перейти до інших розділів адміністрування інтерфейсу веб-сайту (рис. 10.1).

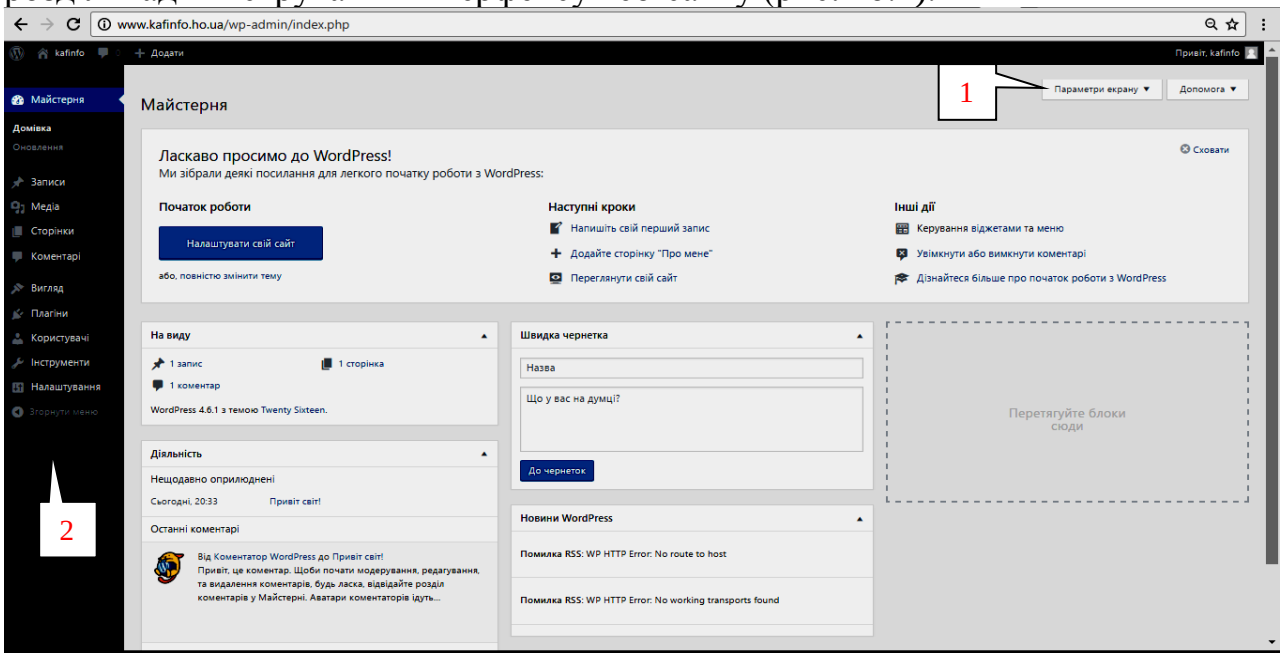
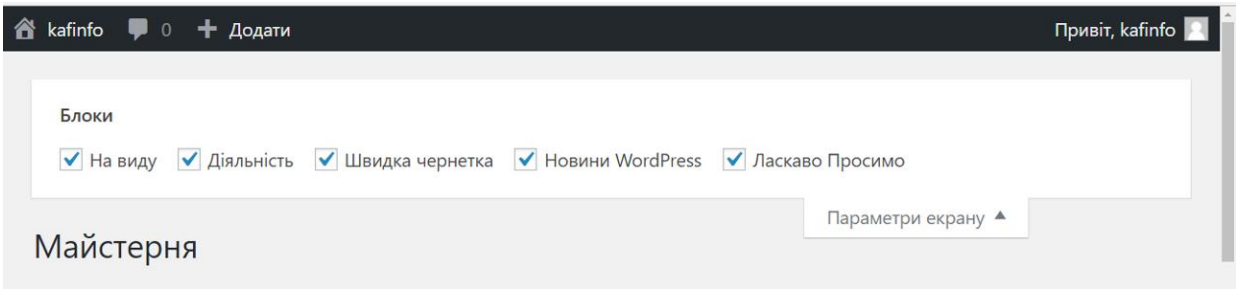



Рисунок 10.1 – Вигляд вікна *Майстерні* сайту

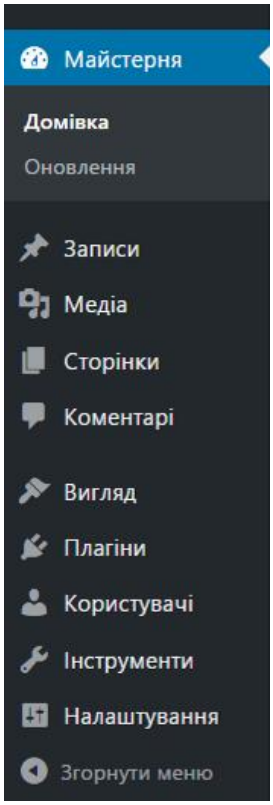
Під номером 1 на рис. 10.1 розташоване посилання *Параметри екрана*, яке дозволяє відключити або включити метабокси. Після клацання по цій кнопці відкриється зверху додаткова панель для керування метабоксами. Метабокси

(модулі) можна міняти місцями перетягуванням мишею. Нові метабокси можна додавати через плагіни або файл functions.php в темі сайту.




Панель навігації (номер 2 на рис. 10.1) використовується для швидкого доступу до найчастіше використовуваних дій в адміністративному інтерфейсі. На панелі навігації розміщено розділи для адміністрування інтерфейсу, а саме:

– *Записи* (Posts);
– *Медіа* (Media);
– *Сторінки* (Pages);
– *Коментарі* (Comments);
– *Вигляд* (Appearance);
– *Плагіни* (Plugins);
– *Користувачі* (Users);
– *Інструменти* (Tools);
– *Налаштування* (Settings).




Кожний розділ має декілька підрозділів, назви яких можна побачити при наведенні мишею на відповідний розділ. Приміром, для створення нового запису слід виконати команду *Записи / Додати*. Після цього з'явиться вікно з такими розділами (рис. 10.2):

1. *Заголовок* – поле для введення заголовка статті;
2. *Візуально* – переключення до візуального редактора для створення статей без знань HTML;
3. *Текст* – переключення для введення даних у вигляді HTML-коду;
4. *Оприлюднити* – для публікації запису або збереження як чернетки:
   - *Статус* – дає можливість вибрати параметри *Чернетка*, *До огляду*;
   - *Видимість* – можна вибрати рівень видимості: *Приватно*, *Захищено паролем*, *Публічно*;
   - *Оприлюднити негайно* – вибирається дата публікації;
5. *Формат* – можна використовувати для виведення різних типів матеріалу;
6. *Категорії* (під розділом *Формат*) – можна вибрати існуючу рубрику або додати нову;
7. *Мітки* (під розділом *Категорії*) – ключові слова, що стосуються статті.

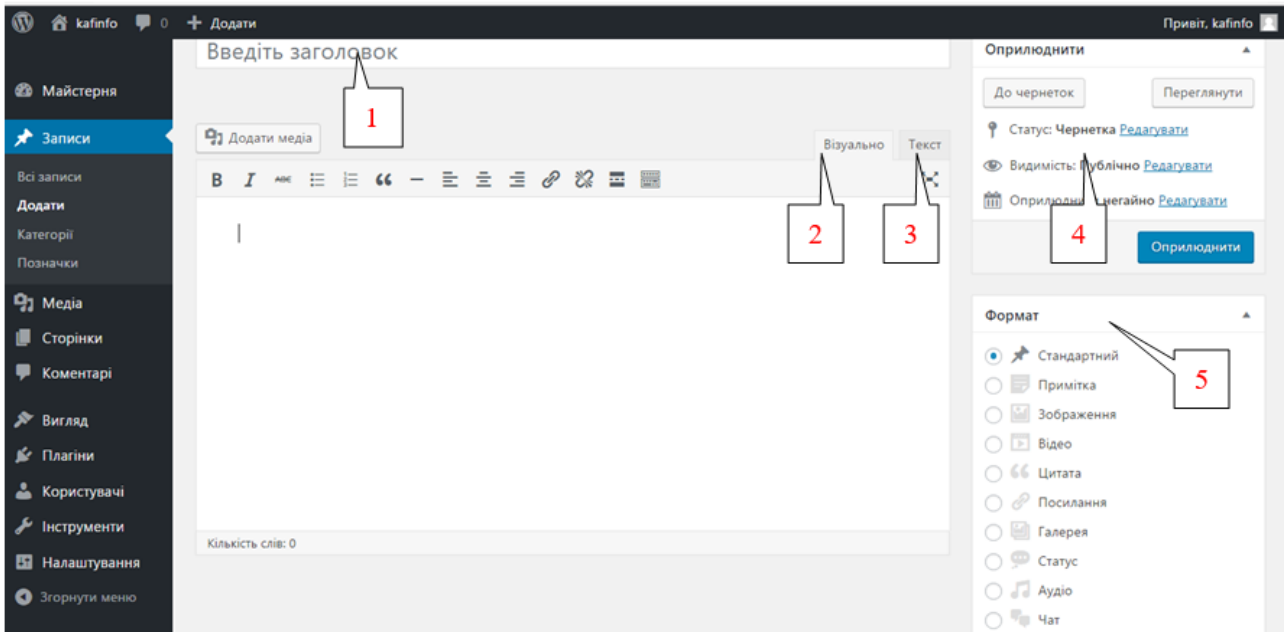
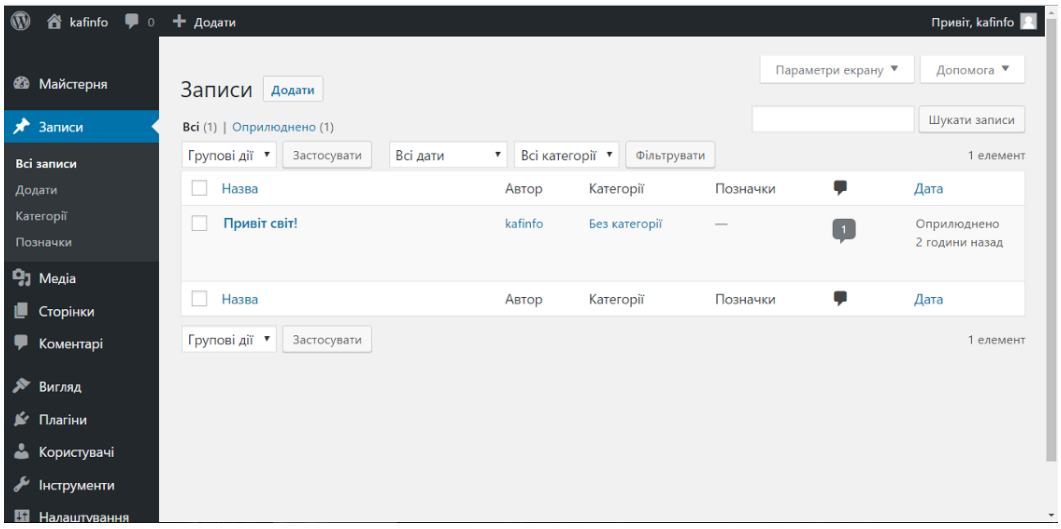



Рисунок 10.2 – Інтерфейс для створення нового запису

Після публікації запису можна зайти у ліве меню *Записи / Всі записи* та перевірити статус нового запису.




## 4. Налаштування сайту на базі Wordpress для роботи з пошуковими машинами

Щоб Ваш проект попав у перелік топ-ресурсів, які у числі перших видаються пошуковими системами, потрібно враховувати численні фактори з самого початку розроблення сайту.

Платформа WordPress має базові можливості для пошукової оптимізації. Так, назву кожного публікованого запису WordPress автоматично додає у тег <title> (разом з назвою сайту), а в самому тексті наділяє їй заголовний тег <h2>. При вставленні ілюстрацій можна заповнити поле *Тема зображення* й опціонально – поле *Підпис зображення*. Вміст цих полів запишеться в параметри title та alt тегу <img> і надасть пошуковим системам можливість визначити, що саме

зображено на ілюстрації. Також візуальний редактор WordPress може виділяти ключові слова безпосередньо в текстах публікацій.

Проте існують і спеціальні плагіни з більш серйозними можливостями для пошукової оптимізації, наприклад, плагін *All in One SEO Pack* для WordPress.

Також рекомендується виконати комплекс кроків для SEO-оптимізації:

– встановити й активувати плагіни *Google XML Sitemaps* (https://wordpress.org/plugins/google-sitemap-generator/), *All in One SEO Pack* (https://wordpress.org/plugins/all-in-one-seo-pack/);
– підготувати яскравий короткий опис сайту;
– створити анонси, які коротко описують опубліковані матеріали;
– створити сторінку *Про нас* (*About*).

Слід зазначити, що **SEO-оптимізація** (від англ. *Search Engine Optimization* – пошукова оптимізація) – це комплекс заходів, направлених на підняття позицій сайту в результатах видачі пошукових систем (Google, Yahoo та ін.) за певними запитами (ключовими фразами). Наприклад: у Вас є сайт, на якому Ви даєте корисні поради про пошукову оптимізацію, але за запитом «сео оптимізація» Ваш сайт займає 50-те місце у списку результатів. Відповідно це далеко не перша сторінка зі списком результатів, і дуже малоймовірно, що хтось із потенційних читачів, яким цікаво почитати про SEO (або СЕО), зайде на Ваш сайт з пошукової системи. А заходи, якими можна цю ситуацію виправити (підняти сайт вище у результатах пошуку) і називають одним словом SEO. Тобто **основною метою SEO** є збільшення притоку відвідувачів з пошукових систем.

Досвід показує, що сайти, зроблені на WordPress CMS, індексуються краще, ніж сайти засновані на інших платформах, за рахунок організації даних та їх відображення. Платформа WordPress продумана максимально лояльно до пошукових систем, тому проекти, реалізовані на ній, буде нескладно просувати у провідних пошукових системах.

### 5. Додавання форуму на сайт Wordpress

Для створення форуму потрібен плагін *bbPress 2.0*, який можна встановити через меню *Плагіни* в адміністративному інтерфейсі *Майстерні*. Як результат на адміністративній панелі навігації з’являться нові розділи, як показано на рисунку:


Для створення нового форуму треба клацнути посилання *New Forum* та пройти через серію екранів майстра створення форуму

Приклади вигляду різних форумів можна побачити за адресами: http://odesskiy.com/odesskiy-forum/; www.ukrcenter.com/Форум; http://forum.maidan.org.ua/.

Отже, платформа WordPress заслужено є однією з найпопулярніших і поширених CMS[1]. Так, стандартний блог можна створити підключенням всього декількох модулів, а зручність навігаційного меню і легкість налаштування надає змогу адаптувати сайт під конкретні завдання. На даний час для WordPress існує понад 52 тисяч модулів, крім того веб-сайти, розроблені на цій платформі, займають високі позиції в провідних пошукових системах.

WordPress можна назвати ідеальною платформою для початківців вебмайстрів, а до її недоліків можна віднести тільки складну структуру бази даних і великий обсяг пам'яті, необхідний для роботи даної CMS. Однак, у сучасних умовах ці проблеми легко розв'язні.

Розглянуті в роботі прийоми роботи з Wordpress можуть допомогти при створенні власних сайтів для блогів чи інших цілей, і, крім того, слугуватимуть відправною точкою для внесення змін у стандартний функціонал WordPress.

### 6. Плагіни веб-розробника для браузера Google Chrome

У кожного веб-розробника при виконанні тої чи іншої процедури завжди виникає потреба швидкого доступу до різної інформації: HTML-коду уподобаного сайту, кольору якогось блока, інформації про швидкість опрацювання і зчитування інформації з джерела тощо. Всі ці процедури можна виконувати вручну, що займатиме доволі багато часу, і процес може затягуватися на тижні.

На щастя, для всіх цих дій і маневрів існують різні плагіни, які інтегруються в різні браузери: Google Chrome, Mozilla FireFox, Opera тощо. Далі розглянемо популярні зараз плагіни для браузерів Firefox та Google Chrome.


1) Плагін ***Firebug Lite*** дозволяє переглядати і редагувати HTML, Java Script і CSS код будь-якого сайту буквально «на льоту», точно аналізувати використання і продуктивність мережі, не виходячи з Вашого браузера. Великим привілеєм є те, що редагувати і переглядати результат можна в самому плагіні, без внесення змін в оригінальні файли.

Скачати плагін для Firefox можна на сайті https://getfirebug.com/firebuglite, а для Google Chrome – на http://getfirebug.com/releases/lite/chrome/.

2) Плагін ***Video Slider*** для WordPress дозволяє без навичок програмування створити відео-слайдер, що сприятиме залученню більшої кількості людей на ваш веб-сайт. Плагін підтримує відео з Youtube, Vimeo, Vevo та MP4.

3) ***Image Slider*** для WordPress – це один з найкращих плагінів для створення повнофункціональних слайд-шоу із завантажених зображень. До тогож плагін дозволяє змінювати кольори, шрифти та розміри.

4) ***Photo Gallery*** – плагін для WordPress з широким спектром інструментів для роботи з галереєю зображень, включаючи ефекти висувних мініатюр, су-

[1] CMS (від англ. *Content Management System* – система керування контентом) – це програмне забезпечення, яке дозволяє користувачам розміщувати або змінювати вже розміщену на сайті інформацію без залучення розробників сайту. Іноді CMS попросту називають "движок сайту".

часних тенденцій для портфоліо, відео та фотогалереї. Джерелами відео можуть використовуватись Youtube або Vimeo медіа ресурси. Плагін має численні макети і гнучкий дизайн інтерфейсу.

5) Функціональність ***Autoptimize*** дозволяє аналізувати і виправляти проблеми продуктивності сайту, тобто оптимізувати його. Він може агрегувати, мінімізувати та кешировать скрипти та стилі, вставляти CSS у заголовок сторінки за промовчанням тощо. Він також мінімізує сам HTML-код, що робить Вашу сторінку дійсно легкою і покращує продуктивність сайту.

6) Плагін ***Yoast SEO*** дозволяє побачити, як виглядатиме ваша публікація чи сторінка у результатах пошуку. Плагін допоможе не тільки збільшити рейтинги, а й підвищити показник кількості кліків для звичайних результатів пошуку.

7) ***WP Fastest Cache*** – російськомовний плагін для WordPress з широким функціоналом для чищення та скорочення коду від зайвого сміття, яке накопичується у блозі. З його допомогою можна суттєво прискорити роботу блогу.

8) ***All-In-One-SEO-Pack*** (AIOSP) є одним з найпотужніших зараз плагінів для пошукової оптимізації. Завдяки даному плагіну можна максимально оптимізувати свій сайт, блог під пошукові машини. З його допомогою топ-видача буде гарантована. AIOSP підтримує Google Analytics, автоматично повідомляє пошукові системи типу Google і Bing про зміни на вашому сайті. Плагін автоматично оптимізує заголовки для Google та інших пошукових систем, автоматично генерує метатеги, дозволяє уникати типового дублювання контенту характерного для WordPress-блогів. Все це працює автоматично після встановлення і не потребує додаткових налаштувань, що дуже зручно для початківців.

9) Плагін ***Akismet*** допоможе позбутися від величезної кількості спаму на Ваш блог.

10) Плагін ***Logaster Logo Generator*** допоможе створити логотип для практично будь-якої теми WordPress.

11) Плагін ***Contact Form*** допоможе налаштувати форму зворотного зв'язку у блозі для відвідувачів.

12) Плагін ***Image Watermark*** допоможе нанести на Ваші зображення водяні знаки, тим самим убезпечить ваш блог від злодійства.

13) ***TinyMCE Advanced*** – зручний плагін для редагування постів.

14) ***WP Security*** – плагін для захисту блогу на WordPress. Дозволить захистити свій блог від хакерських атак, закриє «дірки» у вашому блозі, тим самим максимально захистивши його.

Ці та інші плагіни дуже корисні, а іноді просто незамінні для веб-розробника.

*Лабораторна робота № 11*

# АНАЛІЗ РОБОТИ ВЕБ-САЙТУ ТА ЗАСОБИ ЙОГО ОПТИМІЗАЦІЇ

***Мета:*** ознайомитися зі складовими аналізу роботи сайту та арсеналом засобів оптимізації веб-сайтів.

## Навчальні питання

1. Основні складові аналізу роботи сайту.
2. SEO-аналіз сайту.
3. Аналіз відвідуваності.
4. Юзабіліті-аналіз.

## Завдання

1. Підготувати відповіді на контрольні запитання.
2. Підготувати доповідь у формі реферату за тематикою, зазначеною в табл. 11.1 відповідно до індивідуального варіанта.

*Таблиця 11.1* – **Тематика доповідей (рефератів)**

| **Варіанти** | **Тема доповіді (реферату)** |
|---|---|
| 1 | Поетапний редизайн сайту: які у нього переваги |
| 2 | Як підвищити ефективність сайту |
| 3 | Що обрати: редизайн або створення сайту «з нуля» |
| 4 | Як оптимізувати сайт для мобільних пристроїв |
| 5 | Важливість редизайну сайту для просування |
| 6 | Як уникнути проблем при редизайні сайту |
| 7 | Коли необхідний редизайн системи навігації |
| 8 | В яких випадках виконується редизайн компонування |
| 9 | З яких етапів складається редизайн |
| 10 | Які завдання стоять перед редизайном |
| 11 | 10 помилок редизайну |
| 12 | Коли потрібний редизайн графіки сайту |
| 13 | Модернізація сайту: сучасні тенденції |
| 14 | Технічна оптимізація сайту: на що звернути увагу |
| 15 | Як підвищити релевантність контенту на сайті |
| 16 | Оптимізація сайту і правильна семантична розмітка |
| 17 | Що таке карта релевантності сайту |
| 18 | Оптимізація сайту: як збільшити швидкість завантаження сайту |
| 19 | Як перевірити швидкість завантаження сайту |
| 20 | Оптимізація сайтів: взаємодія title і h1 |
| 21 | Оптимізація контенту сайту для мобільного аудиторії |
| 22 | Оптимізація сайтів: робота з онлайн-формами |
| 23 | Новий домен: як змінити домен |

*Закінчення табл. 11.1*

| Варіанти | Тема доповіді (реферату) |
|---|---|
| 24 | Оптимізація сторінок: сервіси перевірки насиченості текстів ключовими словами |
| 25 | Перевірка індексації сайту |
| 26 | Як перевірити сайт на віруси |
| 27 | Аналіз посилань на сайті |
| 28 | Що таке SEO-аудит і навіщо він потрібний |
| 29 | Оптимізація сайтів на CMS WordPress |
| 30 | Показник відмов як критерій якості сайту |

## Контрольні запитання

1. Яке призначення оптимізації сайтів?
2. У чому полягає розкручування і просування сайтів?
3. Які складові включає аналіз роботи сайту?
4. Яке призначення SEO-аналізу сайту?
5. Яка мета SEO-просування?
6. Які можливості надає аналіз відвідуваності сайту?
7. Яке завдання веб-аналітики?
8. Чому повний аналіз та оцінка відвідуваності сайту може сприяти його розкрутці?
9. Що таке юзабіліті сайту?
10. Як провести юзабіліті аудит?

## Теоретичні відомості

### 1. Основні складові аналізу роботи сайту

Нині сайти для компаній є одним з найважливіших **інструментів бізнесу**: з його допомогою можна рекламувати свої товари та послуги та безпосередньо співпрацювати з клієнтами. Проте ефективність сайту як інструмента для бізнесу може знижуватися, якщо сайт працює не так, як було задумано. У зв'язку з цим важливим етапом у підтримці веб-сайту компанії є **аналіз роботи сайту**.

Аналіз роботи свойого сайту потрібен на регулярній основі, адже інакше неможливо оцінити: чи реалізує сайт усі 100% свого потенціалу або ж є просто «мертвим вантажем» для компанії. Також без регулярного аналізу неможливо вчасно помітити **проблеми або перебої в роботі сайту** і внести необхідні правки.

Аналіз роботи сайту включає **кілька основних складових**, серед яких можна назвати SEO-аналіз сайту, аналіз його відвідуваності, юзабіліті-аналіз, технічний аналіз тощо. Кожний з таких аналізів має важливе значення для загальної оцінки ефективності сайту і визначення подальшого шляху його розвитку.

1) **SEO-аналіз сайту** дозволяє визначити, як сайт «бачать» пошукові системи, виявити сильні і слабкі місця сайту щодо пошукового просування. Наприклад, за допомогою SEO-аналізу можна дізнатися, за якими з тематичних ключових слів сайт перебуває на верхніх рядках пошукової видачі, оцінити, наскільки релевантні ті чи інші сторінки сайту пошуковим запитам, під які вони просуваються, а також з'ясувати, чи добре перелінковані між собою внутрішні сторінки сайту.

2) **Аналіз відвідуваності сайту** дозволяє дізнатися детальну інформацію про те, як і звідки приходять відвідувачі на сайт, скільки часу проводять на його сторінках, з яких сторінок йдуть тощо. Такий аналіз є дуже важливим, адже з його допомогою можна не тільки оцінювати поточну відвідуваність сайту, а й прогнозувати майбутню.

Звичайно для аналізу відвідуваності сайтів користуються спеціальними онлайн-сервісами аналітики, такими як **Google Analytics**. Подібні сервіси дають розгорнуту інформацію про кількість відвіданих сторінок, ключові слова, за якими на сайт заходили відвідувачі, шляхи переходів по сайту, конверсію, час, проведений відвідувачами на сайті і про багато-багато іншого. Користуватися Google Analytics можна безкоштовно.

3) **Юзабіліті-аналіз сайту** та аналіз дизайну його сторінок – дві тісно взаємопов'язані складові у загальному аналізі роботи сайту. Завданням таких аналізів є виявлення проблем взаємодії відвідувачів з сайтом. Наприклад, за допомогою юзабіліті-аналізу можна визначити, чи добре видно на сторінках сайту важливу інформацію, чи зручна навігація по сайту, чи зрозумілі інструкції або заклики до дії. Аналіз дизайну сайту, у свою чергу, покликаний показати, які саме помилки в дизайні спричинили проблеми з юзабіліті.

4) **Технічний аналіз веб-сайту** потрібен для отримання відомостей про коректність роботи сайту з технічної точки зору. Так, у рамках цього аналізу може перевірятися коректність роботи різних скриптів на сайті, доступність різних сторінок сайту, швидкість завантаження даних. За потреби подібний аналіз може включати і перевірку програмного коду сайту для виявлення помилок, які можуть заважати нормальній роботі ресурсу.

Оцінюючи основні етапи проведення загального аналізу роботи сайту, можна помітити, що подібний аналіз – справа аж ніяк не проста. При цьому сам по собі аналіз є лише допоміжним засобом у роботі з сайтом, оскільки потрібно ще й **правильно інтерпретувати проаналізовані відомості**, щоб скласти якісний та ефективний план подальших дій з розвитку і просування сайту. Без цього проведення аналізу роботи сайту – марна витрата часу.

### 2. SEO-аналіз сайту

**SEO** (*Search Engine Optimization*, оптимізація для пошукових машин або скорочено оптимізація сайтів) – це оптимізація внутрішніх і зовнішніх параметрів сайту, які враховуються пошуковими системами для оцінки релевантності, з метою просування сайту в топ і зростанню відвідуваності сайту.

Отже, пошукова оптимізація або SEO полягає в комплексі робіт, покликаних підвищити рейтинг сайту в пошукових системах. Чим вищі позиції в результатах пошуку займає сайт, тим більше зацікавлених відвідувачів зможе його знайти. Метою SEO-просування є досягнення першої сторінки результатів пошуку за певним запитом. На першій сторінці розміщено лише 10 релевантних сторінок, але мільйони інших намагаються підняти себе сюди, до топ-3 або навіть першого місця (топ-1). Тому комплекс робіт з розкрутки та просування включає багато складових.

Зокрема, це аналіз (моніторинг, позиції та сторінки в пошуковиках, аудит, конверсія та юзабіліті тощо), внутрішня оптимізація (оптимізація HTML-коду, тексти (контент), заголовки, посилання, службові директиви тощо), зовнішнє просування (зовнішні посилання (беклінки), індексування, реєстрації тощо).

Ціна пошукової SEO-оптимізації залежить від конкретного завдання, сайту та конкуренції, і тому може відрізнятись, через що за висококонкурентними та високочастотними запитами Вам не вдасться піднятися безкоштовно. Якщо на сайті пропонуються певні послуги або продаються товари (сайти-візитки, корпоративні та комерційні сторінки, інтернет-магазин), то така реклама в інтернеті зможе значно підвищити рівень продажів, оскільки після професійного та ефективного розкручування значно більша кількість потенційних покупців буде переходити на Ваш ресурс по цільових запитах.

**3. Аналіз відвідуваності**

Дізнатися повну інформацію про відвідувачів Вашого сайту можна, використовуючи різні інструменти **веб-аналітики**. Її задачами є повний аналіз та оцінка відвідуваності сайту для подальшого його розкручування, аналіз ефективності рекламної кампанії, оптимізація сторінок сайту для підвищення коефіцієнта конверсії відвідувачів у покупців.

У сфері веб-аналітики застосовуються різні інструменти, серед яких популярністю користуються переважно безкоштовні. Наприклад, завдяки спеціальним веб-інструментам, таким як Google Analytics, можна не тільки довідатися кількість хітів (завантажень сторінки) і хостів (тобто відвідувачів з унікальною IP-адресою), а й провести детальний аналіз поведінки відвідувача на сайті. Крім того, існують спеціальні **аналізатори логів** (наприклад, Webalizer, AWStats) – локальні програми, які збирають записи подій (логи) сервера, на якому розміщений сайт.

Якщо потрібна грамотна оптимізація сторінок сайту або ж необхідно провести його **редизайн**, то найбільш корисними для цього будуть такі дані:

- з яких джерел відвідувачі приходять на сайт;
- які ключові слова залучають просто відвідувачів, а які – саме покупців;
- які сторінки сайту є найбільш відвідуваними й чому;
- які сторінки сайту є найменш відвідуваними й чому;
- на яких сторінках відвідувачі залишають сайт, не замовивши послуги або не зробивши покупку.

Так, знаючи, звідки приходять на сайт відвідувачі, можна точніше вибирати методи розкрутки. Якщо головне джерело – **пошукові системи**, то треба більше уваги приділяти пошуковій оптимізації сторінок сайту. Якщо ж, наприклад, більшість відвідувачів приходять із форумів, соціальних мереж і т. п., то, навпаки, варто більше займатися розкруткою сайту в цьому напрямі.

А, знаючи, наприклад, IP-адресу відвідувача, можна визначити його географічне місцерозташування. Це надасть власникові сайту можливість визначити, на яку аудиторію йому краще орієнтуватися: вітчизняну або іноземну.

Зібрати корисну для веб-аналітики інформацію нескладно й можна впоратися власними силами. А от проаналізувати здобуті результати й розробити необхідний план дій під силу тільки досвідченим професіоналам.

**4. Юзабіліті-аналіз**

**Юзабіліті** є показником зручності користування різними сайтами. За допомогою юзабіліті-тестування (юзабіліті-аудиту) можна даний показник оцінити. Юзабіліті-тестування може проводитися як на етапі розроблення сайту, так і протягом усього життєвого циклу сайту.

Юзабіліті-аудит допомагає дізнатися, чи потребує сайт редизайну або перепроектування інтерфейсу користувача. Такий аудит необхідний усім сайтам, але особливо його потребують інтернет-магазини, оскільки даний тип сайтів працює «напряму» з потенційними покупцями. Для інтернет-магазинів юзабіліті-аудит допомагає вирішити такі задачі:

- дізнатися, чому рівень продажів нижчий, ніж очікувалося;
- визначити, на якому етапі взаємодії у відвідувачів найчастіше виникають проблеми;
- виробити рішення для виявлених проблем.

Перш ніж проводити юзабіліті-аудит інтернет-магазину, та й будь-якого іншого типу сайтів теж, слід спочатку якомога конкретніше визначити його цільову аудиторію. Для цього зазвичай складають **портрет типового користувача сайту**, який демонструє[1]:

- вік і матеріальне становище;
- інтереси і властиву поведінку;
- професійні навики;
- рівень навиків інтернет-користувача.

**Юзабіліті-тестування** може бути якісним і порівняльним. При якісному юзабіліті-тестуванні вивчається поведінка і дії користувачів при роботі з сайтом, що тестується, а при порівняльному вивчаються дії користувачів при роботі відразу з декількома сайтами: тим, що тестується, і його конкурентами.

Необхідне **обладнання для юзабіліті-тестування сайту** включає комп'ютер користувача з доступом до інтернету, а також записувальну (реєструвальну) апаратуру для фіксації дій і поведінки респондента у час проведення тестуван-

[1] Що таке юзабіліті тестування (юзабіліті аудит). [Електронний ресурс]. – Режим доступу: http://webstudio2u.net/ua/design-web/655-usabiliti-testirovanie.html.

ня. В якості такої апаратури зазвичай застосовуються відеокамери або ж диктофони, ще може підійти для цієї мети веб-камера (слід вести запис потоку даних, що передається камерою).

Останнім часом для проведення юзабіліті-тестів стала використовуватися технологія **eye tracking** (айтрекінг), яка дозволяє визначити, як і куди по сторінці сайту рухається погляд користувача. Для застосування подібної технології потрібен спеціальний прилад eye tracker (айтрекер): за допомогою інфрачервоних променів цей прилад фіксує положення очей при перегляданні, і потім ці дані наносять на спеціальні карти. Технологія айтрекінга дозволяє отримувати дуже точні дані, однак вона доволі дорога.

Щоб **провести юзабіліті-аудит**, слід відібрати певну кількість респондентів (5 – 8 буде достатньо), що відповідають портрету типового користувача сайту, підготувати обладнання. Також слід правильно сформулювати завдання для респондентів і скласти сценарій їх взаємодії з сайтом, що тестується. Крім того, слід визначити перелік метрик, за якими буде проводитися аналіз зібраних у ході тестування даних.

Під час проведення юзабіліті-тестування важливо дати респондентам зрозуміти, що перевіряється саме сайт, а не вони самі, тому боятися зробити якусь помилку не потрібно. При тестуванні респонденти повинні обов'язково **промовляти свої зауваження**, емоції при взаємодії з сайтом, адже це теж характеризує рівень комфорту роботи з сайтом: якщо респондент супроводжує свої дії незадоволеними коментарями або ж злиться, то це може свідчити про те, що користування сайтом є незручним.

**Юзабіліті-аудит** не закінчується просто фіксацією даних про виявлені проблеми або помилки в роботі сайту. Після збору даних настає черга аналізу – найскладнішого етапу у всьому тестуванні. Фахівці з юзабіліті повинні вивчити всі здобуті дані, систематизувати їх і на їхній основі зробити висновки про те, чи вимагає сайт доопрацювання або перепроектування.

Окремо юзабіліті-тестуванням займаються компанії, які спеціалізуються у цьому напрямі. Також юзабіліті-аудит часто проводиться і веб-студіями при розробленні або редизайні сайтів.

*Лабораторна робота № 12*

# ВЕБ-АНАЛІТИКА ТА ОПТИМІЗАЦІЯ РОБОТИ САЙТУ

***Мета:*** ознайомитися з методами та засобами оптимізації власних сайтів.

## Навчальні питання

1. Оптимізація сайтів: як покращити готовий сайт.
2. Методи SEO-оптимізації сайту.
3. Використання безкоштовних інструментів для веб-аналітики.

## Завдання

1. Ознайомитись у теоретичних відомостях до цієї лабораторної роботи з основними засобами оптимізації сайтів.

2. У браузері зайти на сторінку сервісу веб-аналітики PR-CY (http://pr-cy.ru/) і виконати аналіз одного зі своїх сайтів (без реєстрації), просто ввівши його URL-адресу. Проаналізувати звіт за різними показниками: чи всі посилання працюють, наскільки доступна мобільна версія, який трафік, перевірку на віруси тощо. PR-CY сформує перелік рекомендації для оптимізації сайту. Зберегти звіт повністю, виконавши команду контекстного меню *Зберегти як*, для того щоб надалі використати рекомендації у звіті для оптимізації сайту.

3. Зробити скріншот екрана зі звітом Вашого сайту і зберегти його з ім'ям Прізвище_аналіз_1.jpg. На скріншоті має бути видно, що виконано аналіз саме Вашого веб-сайту.

4. Виконати ще одну веб-аналітику того самого або іншого зі створених під час попередніх лабораторних робіт сайтів за допомогою сервісу Google Analytics (https://analytics.google.com). Зробити скріншот екрана зі звітом Вашого сайту і зберегти його з ім'ям Прізвище_аналіз_2.jpg. На скріншоті має бути видно, що виконано аналіз саме Вашого веб-сайту.

5. Створені файли скіріншотів зі звітами надати на перевірку викладачеві.

6. Підготувати відповіді на контрольні запитання.

## Контрольні запитання

1. Чому потрібно проводити оптимізацію сайтів?
2. Назвати методи оптимізації сайту та його просування.
3. Охарактеризувати білу оптимізацію сайту.
4. Які методи сірої оптимізації сайту?
5. Що розуміють під помаранчевою оптимізацією сайтів?
6. Назвати методи чорної оптимізації сайтів.
7. Що таке SEO-аудит?
8. Коли і навіщо проводять SEO-аудит?
9. Який алгоритм виконання SEO-аудиту?
10. Назвати та охарактеризувати безкоштовні інструменти для веб-аналітики.

## Теоретичні відомості

### 1. Оптимізація сайтів: як покращити готовий сайт

Оптимізація – важлива частина життєвого циклу сайтів. **Оптимізація сайтів** допомагає покращити юзабіліті, ефективність методів розкручування й просування. В остаточному підсумку, оптимізація сайтів сприяє підвищенню їхньої прибутковості, а отже, позитивно впливає на фінансове благополуччя компанії. Як можна поліпшити вже готовий сайт?

Якщо оптимізація сайтів не була виконана ще на етапі створення цих сайтів, то як результат створені сайти можуть не задовольняти поставленим задачам і цілям. Однак, існує кілька простих і доступних методів, які дозволять оптимізувати сайти без їх повного або часткового редизайну, модернізації.

Наприклад, можна додати до зображень на сайті атрибути alt і title. Атрибут alt (від англ. *alternative*) містить текст, що у разі випадкового або спеціального відключення зображення (наприклад, у браузері) надасть необхідні пояснення відвідувачам сайту чи пошуковим роботам. Атрибут title, у свою чергу, відповідає за «підказку», яка з'явиться при наведенні вказівника миші на зображення. Атрибут title аналогічним чином працює і для гіперпосилань (тег <a>).

До слова, **гіперпосилання можна зробити більш зручними для відвідувачів**, якщо це не було зроблено раніше. Текстовий вміст посилань, так званий анкор, повинен чітко інформувати відвідувачів про сторінки, на які ведуть посилання. Крім того, оптимізація сайтів з використанням ключових слів в анкорах сприяє не тільки кращій навігації по сайті, а й пошуковому просуванню сайту.

Швидко й без шкоди для основного дизайну сайту, його контенту, можна **видалити фонову музику** або, якщо вона все-таки потрібна, прибрати автоматичне програвання музики при завантаженні сторінок сайту. Як показує практика, відвідувачів дратує музика, програвання якої починається автоматично, або ж музика, яку не можна відключити. А якщо відвідувачів щось дратує, вони залишають сайт.

Оптимізація сайтів у «лайт»-версії для підвищення ефективності пошукового просування виконується просто. Можна **відредагувати заголовки сторінок сайту** (title, відображається у лівій верхній частині вікна браузера) відповідно до головних ключових слів сайту. Якщо помістити у заголовок сторінки ключові слова, це допоможе пошуковому роботу проіндексувати сторінку й підвищить її шанси потрапити в топ пошукової видачі.

Можна **підсилити на сайті ключові слова**, це також не потребує «фундаментальних» змінень у структурі або вмісті сайту. Достатньо виділити ключові слова більшим за розміром шрифтом, курсивним або жирним накресленням. Для пошукових роботів величезне значення мають також заголовки у тексті. Однак, не можна використовувати занадто багато заголовків. Загальноприйняте правило: не більше 1 – 2 заголовків h1, трохи більше – h2 і т.д. Тільки грамотне використання заголовків допоможе підняти позиції сайту.

**Оптимізація сайтів без радикальних змінень структури або дизайну вже готового сайту можлива не завжди.** Наприклад, необхідність змінити всього

лише гарнітуру шрифту тексту на сайті може викликати потребу перегляду всієї концепції дизайну сайту. А необхідність додавання/видалення одного розділу призведе до змінення усієї структури сайту, появи «битих» посилань тощо.

Тому оптимізація сайтів має здійснюватись ще на етапах їх розроблення і тестування. У цьому разі існує набагато більше можливостей для виправлення виявлених проблем, забезпечення високого рівня юзабіліті сайту та його ефективності. Прикладами безкоштовних онлайн-засобів перевірки сайтів є *Уніфікований валідатор W3C* (http://validator.w3.org/unicorn/), *Browser Shots* (http://browsershots.org/).

## 2. Методи SEO-оптимізації сайту

Використовують кілька термінів: пошукова оптимізація, SEO-аудит, SEO-оптимізація[1]. Є різні методи оптимізації сайту і його просування, але виділяють чотири основних: біла, сіра, помаранчева й чорна. Розглянемо їх.

### 2.1. Біла оптимізація сайту

Біла оптимізація сайту – дозволена пошуковими системами. Вона використовує тільки легальні методи просування: змінення щільності ключових фраз у тексті, що не впливає на читабельність, обмін посиланнями з інтернет-ресурсами схожої тематики, коментарі в тематичних блогах і спеціалізованих форумах. Це природна оптимізація сайту, оскільки спроби вплинути на пошукові алгоритми не здійснюються. Така оптимізація сайту ресурсоємна за тимчасовими і матеріальними витратами, однак має максимально тривалий ефект.

### 2.2. Сіра оптимізація сайту

До сірої оптимізації сайту відносять використання doorway-сторінок без автоматичної переадресації, з посиланням-запрошенням на цільовий сайт; накручування лічильників відвідувань, створення сателітів (сайтів зі схожим вмістом, що просуваються чорними методами, з переадресацією на цільові сторінки основного ресурсу). Оптимізація сайтів сірими методами може вважатися вищим пілотажем, оскільки попри використання чорної оптимізації, штрафних санкцій щодо цільового сайту пошукові системи не нараховують.

### 2.3. Помаранчева оптимізація сайту

До помаранчевої оптимізації сайту відносять розміщення матеріалів, які не маючих жодного відношення до основної тематики сайту (наприклад, пізнавальних або статистичних фактів). Притягнутий додатковим матеріалом користувач може затриматися на сайті й придивитися уважніше до цільових сторінок, однак така оптимізація сайту достатньо трудомістка й вимагає спеціальних технологій.

### 2.4. Чорна оптимізація сайту

Чорна оптимізація сайту допускає використання методів, заборонених пошуковими системами, працюючи за принципом «на війні всі засоби підходять». Забороненими методами оптимізації сайтів є:

– використання сторінок сформованих лише набором ключових фраз, які не мають сенсу для відвідувачів сайту – пошуковий спам;

---

[1] Оптимізація сайтів. [Електронний ресурс]. – Режим доступу: http://webstudio2u.net/ua/optimization/97-seo.html.

– використання сторінок, які містять тільки посилання, linkfarm або посилальний спам;
– використання «невидимого» тексту (текст дрібним шрифтом, що зливається з кольором фону);
– використання клоакінга – підміни контенту сторінки так, що користувач і пошукова система бачать сторінки сайту зовсім по-різному;
– використання doorway-сторінок з автоматичним перенаправленням на цільовий сайт.

Використання цих методів просування швидко виводить сайт у топ, але виявлення пошуковою системою використання заборонених методів просування призводить до виключення сайту з пошукової видачі назавжди.

На сьогодні оптимізація сайту – не розкіш, а необхідність, для тих, хто піклується про просування сайту й розвиток свого бізнесу.

### 3. SEO-аудит

SEO-аудитом прийнято називати таку послугу, яка включає комплексну перевірку параметрів сайту, що має значення при його просуванні у пошукових системах. Список параметрів, що перевіряються, у кожному випадку може бути різним – це залежить і від типу сайту, який треба перевірити, і від його цілей і задач, і від компанії, що надає послуги SEO-аудиту.

Звичайно SEO-аудит виконується перед початком просування сайту – в цьому разі він потрібен для визначення готовності сайту до виконання активних дій з його розкручування. Також SEO-аудит може виконуватися і після певного періоду просування – для оцінки здобутих результатів.

Грамотно виконаний SEO-аудит дає власнику сайту багато корисних відомостей, допомагаючи йому зрозуміти, де саме на сайті є «слабкі місця», а також дізнатися, які кроки можна почати роюоти з усунення цих «слабких місць». Розглянемо приклад можливого алгоритму проведення SEO-аудиту.

**Алгоритм виконання SEO-аудиту:**

1) **SEO-аналіз**. На цьому етапі виконання SEO-аудиту проводиться аналіз основних факторів зовнішньої і внутрішньої оптимізації сайту:
– аналіз поточного статусу сайту в пошукових системах за тематичними ключовими словами;
– аналіз кількості та якості зворотних посилань на сайт;
– аналіз кількості та якості проіндексованих пошуковими системами сторінок сайту;
– аналіз показників трастовості сайту (PR[1], тІЦ[2]);

[1] **PR** (*Page Rank*) – алгоритм, який дозволяє визначити авторитетність сайту в пошуковій системі, наприклад, Google. Важливо, що PR розраховується для кожної сторінки сайту. Для простоти можна сказати так: чим більше у сторінки, яка на Вас посилається, Page Rank, тим імовірніше збільшення PR і Вашої сторінки. Значення PR мають діапазон від 0 до 10. Перераховується показник доволі рідко, за деякими даними – раз на 4 – 6 місяців.

[2] **тІЦ** – тематичний індекс цитування сайту. На відміну від PR, тІЦ розраховує авторитетність всього сайту, а не окремої сторінки. При цьому враховується не лише кількість посилань на Ваш сайт, а й, передусім, якість посилань, тобто – наскільки тематика сайту відповідає Вашій темі. Ось тому він і названий тематичним індексом цитування. По-простому можна сказати так: чим більше сайтів, близьких за тематикою, посилаються на Ваш сайт, тим вище у Вас тІЦ. Слід сказати, що тІЦ використовується лише для ранжирування сайту, але не в пошуковій видачі. Для збільшення тІЦ треба, щоб контент сайту був актуальним й унікальним.

– аналіз унікальності текстів на сторінках сайту;
– перевірка наявності заголовків (h1 – h6) на сторінках сайту;
– аналіз мета-тегів (title, keywords, description) кожної сторінки;
– аналіз ключових слів у текстах сторінок сайту;
– аналіз внутрішньої перелінковки сайту;
– аналіз sitemap сайту і файл robots.txt.

Для виконання усіх перевірок на цьому етапі можуть використовуватися різні ручні або автоматизовані методи. При цьому всі результати заносяться в тій чи іншій формі до спеціального звіту, що дозволить надалі скласти рекомендації щодо виправлення помилок і недоліків, якщо такі виявляться в процесі аналізу.

2) **Аналіз юзабіліті**. Цей етап SEO-аудиту передбачає ретельне вивчення юзабіліті сайту. Фахівці перевіряють, наскільки легко і зручно користуватися сайтом, наскільки зрозумілі і доступні користувачам дії, необхідні для здійснення купівлі або замовлення послуги. На даному етапі можуть виконуватися такі роботи:

– аналіз різних форм на сайті, включаючи реєстраційні, форми замовлення, форми зворотного зв'язку;
– аналіз кнопок купівлі товару/замовлення послуги, а також інших інтерактивних елементів на сторінках;
– аналіз загальної структури головної і другорядних сторінок сайту.

Результатом юзабіліті-аналізу в рамках проведення SEO-аудиту є звіт, в якому вказуються «плюси» і «мінуси» юзабіліті поточної версії сайту.

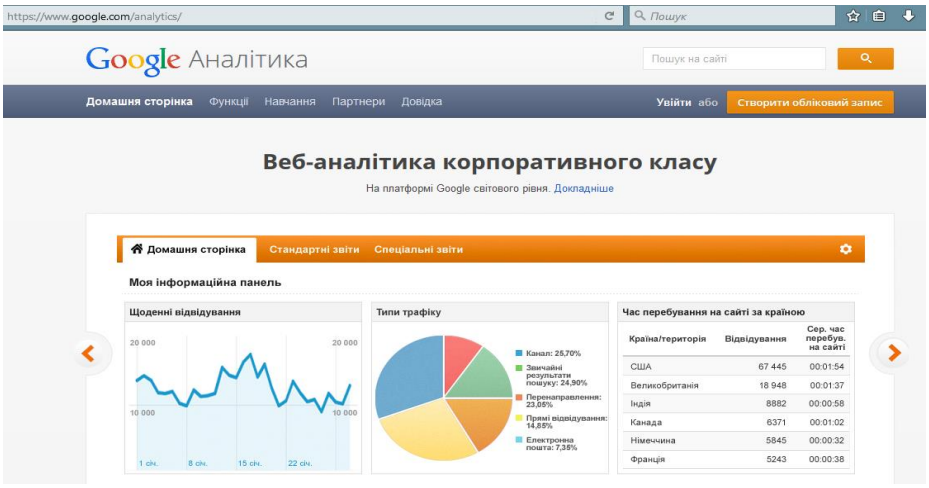
Перелік рекомендацій SEO-аудиту може бути різним – все залежить від конкретного виконавця. Найчастіше до списку рекомендацій потрапляють поради щодо заповнення мета-тегів сторінок сайту відповідно до вимог пошукових систем, поради з оптимізації текстів на сайті, поради з нарощування посилальної маси сайту тощо. На основі цих рекомендацій власник сайту приймає рішення про те, чи варто виконувати роботи з просування сайту.

### 4. Веб-аналітика засобами Google Analytics

**Google Analytics** – це онлайн-сервіс від Google для веб-аналітики для розкручування сайту. Якщо потрібно проаналізувати відвідуваність сайту, визначити з яких джерел приходять відвідувачі, розрахувати ефективність ключових слів, то без веб-аналізу не обійтися. На сьогодні існує безліч інструментів і методів веб-аналітики, проте лідируючі позиції по праву займає безкоштовний сервіс Google Analytics. Фахівці компанії **Google** постаралися втілити в цьому онлайн-сервісі максимум корисних можливостей для веб-аналітики. Google Analytics має інтуїтивно зрозумілий інтерфейс і наочність подання даних (графіки, діаграми, таблиці).

Щоб почати користуватися перевагами Google Analytics, слід зареєструватися в сервісі https://www.google.com/analytics/ або https://analytics.google.com. Для цього увійти до облікового запису на gmail.com (напис-посилання *Увійти* праворуч від помаранчової кнопки *Створити обліковий запис*). У вікні, що з'явиться, натиснути кнопку *Реєстрація*. Далі ввести власне ім'я, назву та URL-адресу

сайту й інші параметри, після чого натиснути кнопку *Отримати ідентифікатор відстежування*.




Після успішної реєстрації Google Analytics запропонує згенерований для Вашого сайту спеціальний **скрипт**, що слід вставити в HTML-код кожної сторінки, яку Ви хочете відслідковувати в Google Analytics. Цей скрипт слід помістити безпосередньо перед тілом сторінки, тобто перед тегом </body>. При цьому скрипт ніяк не впливає на завантаження сторінки, не вимагає завантаження додаткових файлів. Текст скрипту може бути таким:

```
<script>
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
  (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
  m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
  })(window,document,'script','https://www.google-analytics.com/analytics.js','ga');
  ga('create', 'UA-91614276-1', 'auto');
  ga('send', 'pageview');
</script>
```

Після зазначених дій та оновлення вмісту сайту з цим скриптом зайшовши на свою сторінку Google Analytics, можна побачити невелику зведену таблицю огляду цього сайту. Щоб дослідити інформацію про сайт докладніше, слід клацнути посилання *Подивитися звіт* у колонці *Звіти*. Перейшовши за посиланням, можна потрапити на сторінку, на якій доступні такі вкладки:

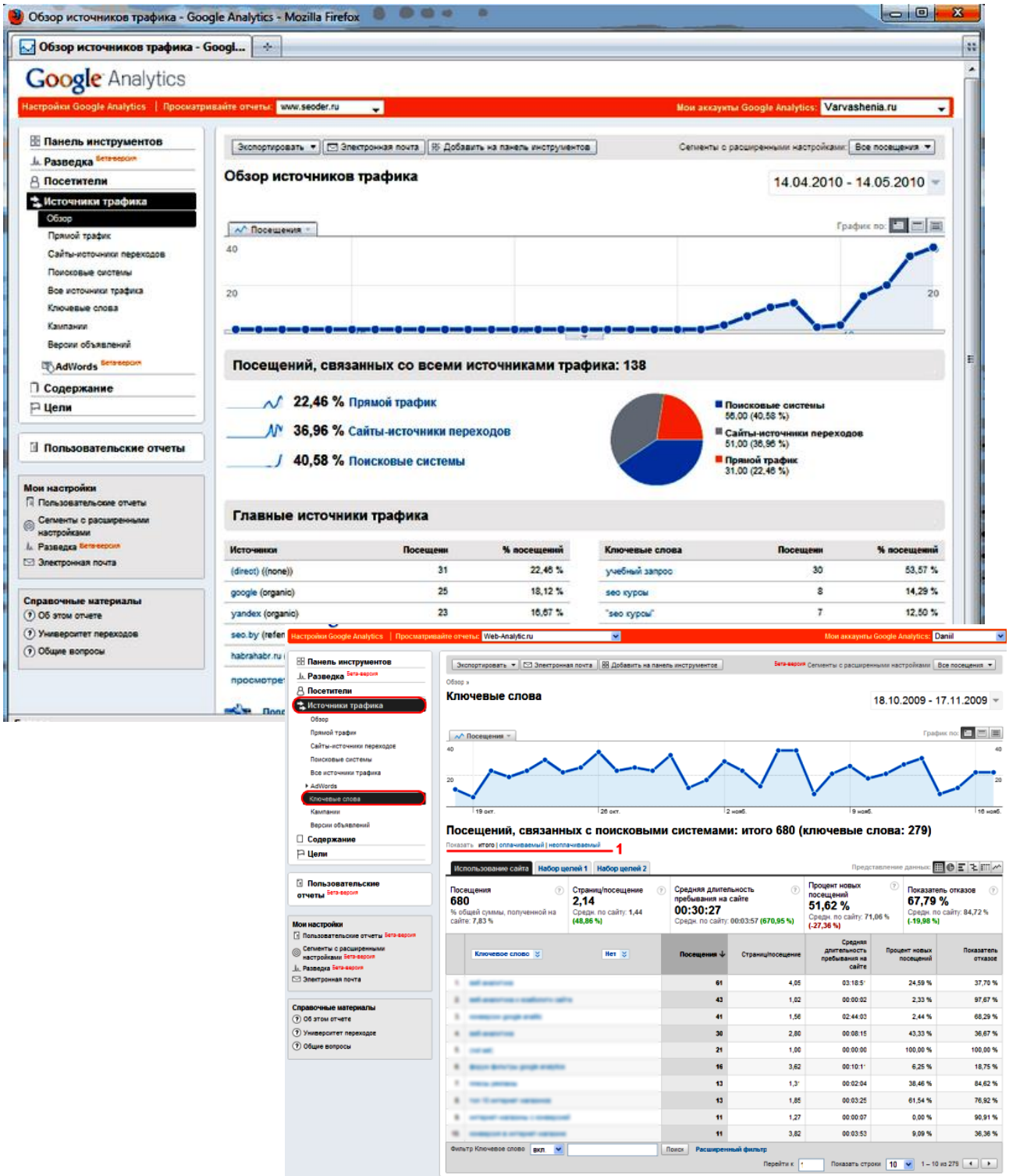
– вкладка ***Панель інструментів*** слугує для візуалізації звітів проведеного веб-аналізу. На панелі можна в довільному порядку і кількості розміщувати різні звіти, такі як огляд відвідувачів, огляд джерел трафіку, огляд браузерів тощо;

– вкладка ***Відвідувачі*** показує повну інформацію про відвідувачів сайту. За допомогою цієї вкладки можна дістати вичерпні відомості про кількість відвідувачів, тривалість їх перебування на сайті, про переглянуті ними сторінки, про можливості їх браузерів і параметри підключення до інтернету;

– вкладка ***Джерела трафіку*** дозволяє дізнатися, з яких сайтів приходять відвідувачі до Вашого сайту, якими пошуковими системами вони користуються, кількість прямого трафіка;

– графіки і таблиці вкладки ***Вміст*** показують, який вміст на сайті є найбільш популярним, які найпопулярніші сторінки входу і виходу, як відвідувачами здійснюється пошук по сайту. Також на цій вкладці можна відслідковувати дії користувачів на сайті;

– ***Цілі*** – це сторінки, на які потрапляє користувач, після здійснення покупки, реєстрації або виконання іншого бажаної дії. Вони дозволяють оцінити, наскільки сайт відповідає цілям компанії. У Google Analytics можна задавати до 20-ти таких цілей.

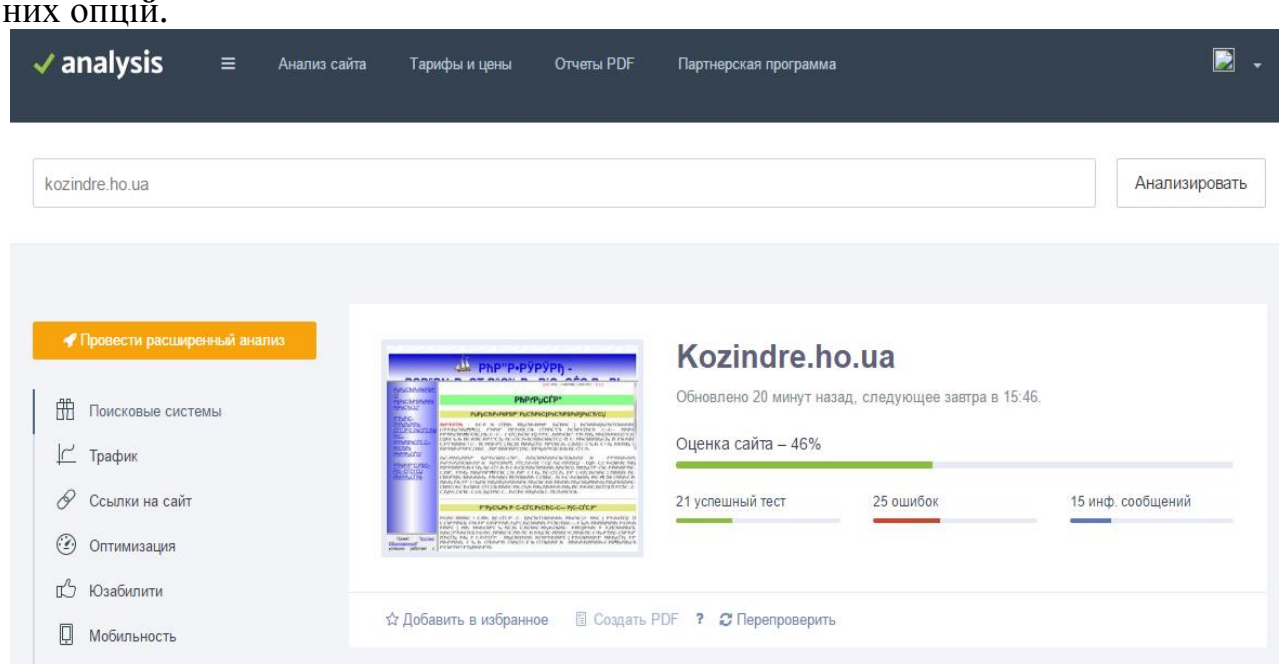


**5. Інші безкоштовні інструменти для веб-аналітики**

1) **Quill Engage** (https://www.quillengage.com/) – англомовний онлайн-сервіс, який пояснює дані Google Analytics і надає призначені для користувача звіти для Вас і Ваших клієнтів.

2) **Clicky** (https://clicky.com/) виконує аналіз дій відвідувачів, дозволяє подивитися теплову карту (принцип роботи теплової карти в тому, що на сторінку сайту накладається шар, де теплими кольорами показані ті області, на які найчастіше потрапляє курсор користувачів, холодними кольорами – зони сайту, на які не звертають уваги). Можна промоніторити конверсії або нагадування про Ваш бренд. Має англомовний інтерфейс. Інструментом можна користуватися безкоштовно, якщо аналізувати тільки один сайт.

3) **PR-CY** (http://pr-cy.ru/) пропонує перевірити сайт: його швидкість, виявити помилки, встановити динаміку зростання за ключовими словами. Дозволяє самостійно провести аналіз сайту навіть без реєстрації на сервісі, для цього потрібно ввести його адресу, і через кілька секунд можна побачити повний звіт. Можна дізнатися, чи всі посилання працюють, наскільки доступна мобільна версія, який трафік, здійснено перевірку на віруси та багато іншого. PR-CY сформує перелік рекомендації для оптимізації сайту. Існують широкий спектр платних опцій.



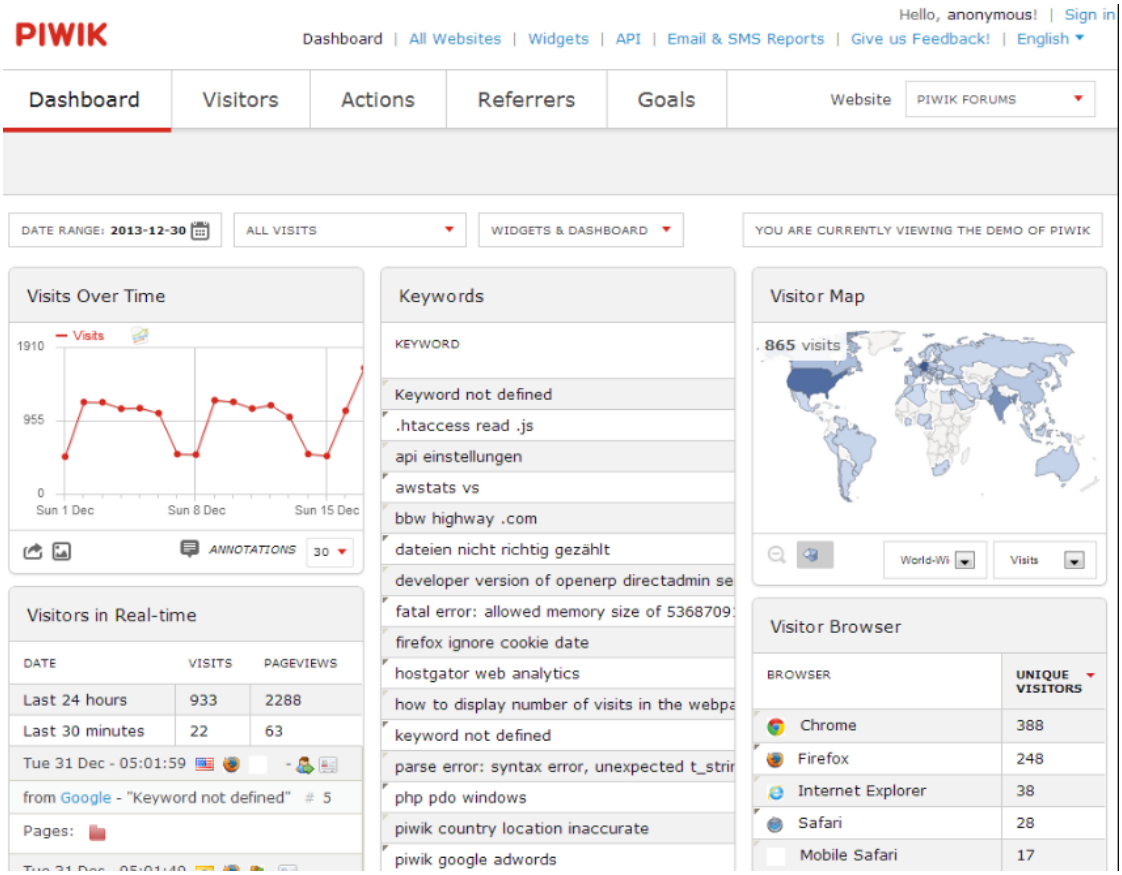

4) **Test My Site** (https://testmysite.thinkwithgoogle.com/intl/en-us) дозволяє перевірити швидкість завантаження того чи іншого сайту на мобільні пристрої, оскільки більшість сайтів через це втрачають половину своїх відвідувачів. Цей сервіс безкоштовно сформує та надішле на електронну пошту звіт з рекомендаціями щодо покращення ефективності веб-сайту на всіх пристроях.

5) **Open Web Analytics** (http://www.openwebanalytics.com/) – англомовний інструмент з відкритим вихідним кодом, який може бути завантажений і встановлений на Ваш хостинг. Його можна використовувати для декількох ресурсів: обмежень за кількістю даних немає. Доступною є інформація про останні відвідини ре-

сурсу (індивідуальні та достатньо докладні відомості про останнього відвідувача сайту із зазначенням місця розташування, типу браузера, переглянутих сторінок, тривалості відвідування). Є і теплова карта кліків і запис рухів курсору.

Версія з розширеним функціоналом доступна для сайтів на WordPress і MediaWiki. Плюс в тому, що всі дані зберігаються на Вашому власному сервері.

6) **Piwik** (https://piwik.org/**)** – аналітична платформа сайтів з відкритим вихідним кодом, яка завантажується і встановлюється на Ваш сервер, що є плюсом. Сервіс пропонує стандартний набір даних: запити і пошукові системи, веб-сайти, сторінки, операційні системи, браузери, дозвіл екрана, поведінку користувачів. Цікаво, що сервіс автоматично формує звіти в PDF або HTML і надсилає їх на електронну пошту.




7) **GoingUp** (https://www.goingup.com/) пропонує веб-аналітику і SEO-інструментарій, доступна теплова карта кліків. Разом з цим сервісом бажано використовувати й інші інструменти веб-аналітики, оскільки в GoingUp не такий широкий спектр функцій.

*Самостійна робота № 10*

# ОПТИМІЗАЦІЯ РОБОТИ САЙТУ

## Завдання

1. Провести оптимізацію одного (чи декількох) сайтів, з числа проаналізованих у пп. 2 і 4 лабораторної роботи № 12, скориставшись рекомендаціями у звітах аналітики.

2. Зробити повторний аналіз, скориставшись будь-яким з розглянутих безкоштовних веб-ресурсів веб-аналітики сайтів: PR-CY (http://pr-cy.ru/), Google Analytics (https://analytics.google.com) чи то іншим.

3. Зробити скріншот, в якому буде видно, що показники покращились порівняно з попереднім аналізом. Зберегти цей скріншот як зображення з ім’ям Прізвище_аналіз_3.jpg.

4. Створений файл скріншоту зі звітом та адресу свого оптимізованого веб-сайту надати на перевірку викладачеві.

*Лабораторна робота № 13*

# ПІДСУМКОВЕ ЗАНЯТТЯ З ПРЕЗЕНТАЦІЄЮ ТА ЗАХИСТОМ СТВОРЕНИХ САЙТІВ

## Навчальні питання

1. Виступи студентів із презентацією створених сайтів.
2. Систематизація набутих знань і практичних навиків зі створення сайтів різними засобами.
3. Колективний аналіз припущених у процесі розроблення сайтів помилок дизайну.

## Завдання

1. Зібрати, проаналізувати та систематизувати всі сайти, створені під час виконання лабораторних і самостійних робіт.

2. Підготувати невелику доповідь у вигляді коментарів та пояснень для презентації створених сайтів. Під час доповіді слід розповісти про причини актуальності вибраної структури створених сайтів, їхню оригінальність, описати ті проблеми, які виникали під час створення проектів, та шляхи їх усунення.

Після кожної доповіді передбачено обговорення даного проекту, де студенти висловлюватимуть свої зауваження та рекомендації.

# СПИСОК ВИКОРИСТАНОЇ ТА РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ


1) 10 бесплатных шаблонов на HTML5 и CSS3. [Электронный ресурс]. – Режим доступа: http://postovoy.net/10-besplatnyh-shablonov-na-html5-i-css3.html.
2) 10 бесплатных инструментов для веб-аналитики. [Электронный ресурс]. – Режим доступа: https://te-st.ru/2017/01/18/10-free-web-analytics-tool.
3) 23 бесплатных адаптивных HTML шаблона. [Электронный ресурс]. – Режим доступа: http://postovoy.net/18.html.
4) 37 адаптивных HTML5 CSS3 шаблона. [Электронный ресурс]. – Режим доступа: http://postovoy.net/25.html.
5) A vocabulary and associated APIs for HTML and XHTML. [Electronic resource]. – Mode of access: https://www.w3.org/TR/html5 – W3C. HTML5.
6) CodePen is a playground for the fron end web. [Electronic resource]. – Mode of access:http://codepen.io/.
7) CSS Tutorial. [Electronic resource]. – Mode of access: www.w3schools.com/css.
8) CSS справочник. [Электронный ресурс]. – Режим доступа: http://css.manual.ru.
9) CSS: Вікіпідручник. [Електронний ресурс]. – Режим доступу: http://uk.wikipedia.org/wiki/CSS.
10) CSS-LIVE: жизнь во фронтенде. [Электронный ресурс]. – Режим доступа: http://css-live.ru.
11) HTML Colors. [Electronic resource]. – Mode of access: http://www.w3schools.com/html/html_colors.asp.
12) HTML Tutorial. [Electronic resource]. – Mode of access: http://www.w3schools.com/html/default.asp.
13) HTML справочник. [Электронный ресурс]. – Режим доступа: http://html.manual.ru.
14) HTML: Вікіпідручник. [Електронний ресурс]. – Режим доступу: http://uk.wikipedia.org/wiki/HTML.
15) Аналіз роботи сайту. [Електронний ресурс]. – Режим доступу: http://webstudio2u.net/ua/design-web/713-analiz-raboty-saita.html
16) Вотролл Э. Изучаем веб-дизайн: учебн. пособие / Этан Вотролл, Джеф Сьярто; [пер. с англ.]. – М. : Эксмо 2010. – 496 с.
17) Вуль В. А. Электронные издания: учебник / В. А. Вуль. [Электронный ресурс]. – Режим доступа: http://www.hi-edu.ru/e-books/xbook119/01/title.htm.
18) Електронні видання: довідник / уклад. Т. Ю. Киричок. – К. : НТУУ «КПІ», 2011. – 400 с.
19) Квинт И. Создаем сайты с помощью HTML, XHTML и CSS на 100% / Игорь Квинт; [3-е изд.] – СПб. : Питер, 2014. – 448 с.
20) Коди спеціальних символів для використання в HTML. [Електронний ресурс]. – Режим доступу: http://vvz.nw.ru/Lessons/SymbolCodes/ symbolcodes.htm?n=1.

21) Маркотт И. Отзывчивый веб-дизайн / Итан Маркотт [пер. с англ. Павла Миронова]. – М. : Манн, Иванов и Фербер, 2012. – 163 с.
22) Маслов И. Как создать блог на WordPress? Пошаговая инструкция от вебмастера / Иван Маслов. [Электронный ресурс]. – Режим доступа: http://ivan-maslov.ru/kak-sdelat-sajt/kak-sozdat-blog-na-wordpress.html.
23) Мержевич В. Самоучитель по HTML / Влад Мержевич. [Электронный ресурс]. – Режим доступа: http://htmlbook.ru/samhtml.
24) Оптимізація сайтів: як покращити готовий сайт? [Електронний ресурс]. – Режим доступу: http://webstudio2u.net/ua/optimization/392-site-optimization.html.
25) Пасічник О. Г. Основи веб-дизайну: навч. посібник / О. Г. Пасічник, О. В. Пасічник, І. В. Стеценко. – К. : Вид. група BHV, 2009. – 336 с.
26) Плавная трансформация | CSS свойство transition. [Электронный ресурс]. – Режим доступа: http://shpargalkablog.ru/2011/07/transformaciya-css.html
27) Практические задания по теме «Web-дизайн и программирование». [Электронный ресурс]. – Режим доступа: http://www.modern-computer.ru/practice/web-design/prcatic-web-design-main.html.
28) Пушкар О. І. Мультимедійні видання : навч. посібник / Пушкар О. І., Климнюк В. Є., Браткевич В. В. – Х. : Вид. ХНЕУ, 2012. – 144 с.
29) Сабін-Вільсон Л. WordPress. Web design. / Ліза Сабін-Вільсон. – John Wiley & Sons Canada, 2011. – 384 с.
30) Сидерхолм Д. CSS3 для веб-дизайнеров / Дэн Сидерхолм ; [пер. с англ. Е. Кудашева]. – М. : Манн, Иванов и Фербер, 2013. – 144 с.
31) Сидерхолм Д. Пуленепробиваемый веб-дизайн. Библиотека специалиста / Дэн Сидерхолм; [3-е изд.] – СПб. : Питер, 2012. – 304 с.
32) Создание Web-сайта на базе WordPress CMS. [Электронный ресурс]. – Режим доступа: http://www.ibm.com/developerworks/ru/library/os-wordpress/index.html.
33) Справочники по HTML и CSS. [Электронный ресурс]. – Режим доступа: https://webref.ru/ref.
34) Уолтер А. Эмоциональный веб-дизайн / Аарон Уолтер ; [пер. с англ. Павла Миронова]. – М. : Манн, Иванов и Фербер, 2012. – 144 с.
35) Уроки по HTML и CSS. [Электронный ресурс]. – Режим доступа: https://webref.ru/layout/learn-html-css.
36) Учебник CSS. [Электронный ресурс]. – Режим доступа: http://ru.html.net/tutorials/css.
37) Учебник по HTML. [Электронный ресурс]. – Режим доступа: http://ru.html.net/tutorials/html.
38) Хоган Б. Книга веб-программиста: секреты профессиональной разработки веб-сайтов. / Б. Хоган, К. Уоррен, М. Уэбер, К. Джонсон, А. Годин. – СПб. : Питер, 2013. – 288 с.
39) Що таке юзабіліті тестування (юзабіліті аудит). [Електронний ресурс]. – Режим доступу: http://webstudio2u.net/ua/design-web/655-usabiliti-testirovanie.html.

# ЗМІСТ

*Трофименко О. Г., Козін О. Б.*

# Веб-дизайн та HTML-програмування

**Навчально-методичний посібник**