

Н. В. Речич

Вебтехнології

вибірковий модуль

10-11



Інформатика
рівень стандарту



Н. В. Речич

Вебтехнології

вибірковий модуль



10–11
КЛАСИ

ВИДАВНИЦТВО
РАНОК

УДК 004.738.5(075.3)
Р46

Схвалено для використання у загальноосвітніх навчальних закладах
(лист Інституту модернізації змісту освіти Міністерства освіти і науки України
від 17.12.2019 № 22.1/12-Г-1175)

Речич Н. В.

**P46 Інформатика : вебтехнології (вибірковий модуль для 10–11 класів, рівень стандарту)/ Н. В. Речич. — Харків : Вид-во «Ранок», 2020. — 160 с.
ISBN 978-617-09-6223-2**

Зміст посібника відповідає вимогам навчальної програми вибірково-обов'язкового курсу з інформатики для учнів 10–11 класів загальноосвітніх навчальних закладів (рівень стандарту) і призначений для вивчення вибіркового модуля «Вебтехнології».

Посібник складається з п'яти розділів і містить основні теоретичні відомості про онлайн-інструменти для розробки структури сайту, опис базових тегів мови HTML, алгоритми проектування та створення сайту, початкові відомості з вебпрограмування, приклади застосування базових правил ергономіки сайту та пошукової оптимізації тощо. Видання містить запитання для контролю знань, завдання для самостійного виконання, диференційовані за рівнем складності, та практичні роботи.

Призначено для учнів 10–11 класів закладів загальної середньої освіти та вчителів інформатики.

УДК 004.738.5(075.3)

Запитання для перевірки знань і практичні завдання для самостійного виконання відповідають рівням навчальних досягнень:

- — початковий і середній рівні
- — достатній рівень
- — високий рівень

У тексті використано такі позначки:

-  — означення, висновок
-  — зверніть увагу



Разом дбаємо
про екологію та здоров'я

Інтернет-підтримка

Електронні матеріали
до посібника розміщено на сайті
interactive.ranok.com.ua



ISBN 978-617-09-6223-2

© Речич Н. В., 2020

© ТОВ Видавництво «Ранок», 2020

Передмова

Пропонований вам посібник присвячено одному з напрямків інформаційних технологій, який наразі розвивається найбільш динамічно — це вебтехнології. Важко уявити без них хоча б один аспект людської діяльності: працюємо ми чи навчаемось, відпочиваємо чи спілкуємося з друзями. Саме завдяки цим технологіям ми готуємося до ЗНО, вивчаємо іноземні мови, використовуючи різноманітні онлайн-платформи, обмінюємо «сторіс» в Інстаграмі, «лайкаємо» сторінки у Фейсбуці, замовляємо квитки на потяг або концерт, слухаємо музику, переглядаємо фільми, читаемо книжки, купуємо будь-що або просто, не виходячи з власної квартири, мандруємо залами видатних музеїв або віддаленими куточками нашої планети.

Посібник складається з п'яти розділів, кожен з яких присвячено певному аспекту розробки сайта.

В *розділі 1*, який називається «Напрямки та інструменти вебдизайну», ви познайомитеся із сучасними тенденціями вебдизайну, класифікацією та структурою сайтів, отримаєте відомості про види цільової аудиторії та механізми її визначення, дізнаєтесь про сервіси для роботи з соціальними медіа та інструменти веброзробника. Стислий огляд популярних редакторів коду допоможе обрати найкращий для подальшої роботи саме для вас. Також ви навчитеся власноруч розробляти структуру сайту за допомогою онлайн-інструментів.

Розділ 2 «Проектування та верстка вебсторінок» присвячено безпосередньо проектуванню та верстці сайту. В ньому розглядаються основні теги мови гіпертекстової розмітки, правила застосування каскадних таблиць стилів, наводяться основи адаптивної верстки та способи досягнення кросбраузерності. Особлива увага приділяється сучасному стандарту HTML5+CSS3, який базується на відокремленні дизайну документа від його вмісту.

Родзинкою посібника є екскурс в історію: створення та становлення Всесвітньої павутини, «війна браузерів», розвиток верстки — від давньої фіксованої до сучасної чуйної (*responsive*).

Розділ 3 «Графіка та мультимедіа для вебсередовища» продовжить знайомити вас із версткою сайта, але тепер мова піде про використання статичних зображень, анімаційних ефектів та підключення аудіо- та відеофайлів до сайту.

Зверніть увагу на те, що кожен параграф розділів 2 та 3 супроводжується практичними роботами, присвяченими темам параграфів, що дає змогу одразу застосувати отримані знання на практиці.

У розділі 4 «Вебпрограмування» ми зосередимося на тому, який вигляд має наш сайт із позицій бекенда, тобто з боку сервера. Будемо говорити про вебпрограмування, створення інтерактивних сторінок, валідацію та види хостингу. Невеликим бонусом буде знайомство з досить популярним наразі API Canvas, який дозволяє створювати растрові зображення та анімацію за допомогою JavaScript.

І нарешті в *розділі 5 «Основи дизайну та просування вебсайта»* мова піде про базові правила ергономіки сайта й пошукову оптимізацію.

Таким чином, використовуючи посібник, ми пройдемо весь шлях від створення макета до перевірки на правильність коду та підключення до хостингу. Познайомимося з основними трендами у вебдизайні, на прикладах переконаємося в необхідності орієнтації на мобільні сайти, з'ясуємо, що таке кросбраузерність і чому необхідно її підтримувати, отримаємо початкові відомості про вебпрограмування.

Мабуть, не можна говорити про створення та супроводження сайтів без достатнього ілюстративного матеріалу. Оскільки жоден друкований підручник не зможе відобразити відео, кожен параграф посібника містить посилання (у вигляді QR-коду) на презентацію з відеорядом, який відповідає змісту параграфа.

У посібнику також включено 12 практичних робіт, які дозволять опанувати викладений у ньому необхідний навчальний матеріал.

Авторка сподівається, що цей посібник стане вашим надійним путівником у світі вебтехнологій і допоможе опанувати теоретичні основи модуля за допомогою цікавих практичних завдань і вправ.

НАПРЯМКИ ТА ІНСТРУМЕНТИ ВЕБДИЗАЙНУ

- 1.1** Основні тренди у вебдизайні
 - 1.2** Види сайтів та цільова аудиторія
 - 1.3** Інформаційна структура сайта
- Практична робота № 1*
- 1.4** Інструменти веброзробника



**1.1**

Основні тренди у вебдизайні



Тренд (від англ. *trend* — напрям, тенденція) — можливий (ймовірний) вектор розвитку подій стосовно якогось проміжку часу.

Загальна кількість користувачів мережі інтернет постійно збільшується. Щодня — на 1 млн нових користувачів. Розглянемо статистичні дані, наведені на рис. 1.1. Упродовж останніх років незмінними залишаються два тренди: необхідність швидкого завантаження сайтів та орієнтація на мобільні версії сайтів.

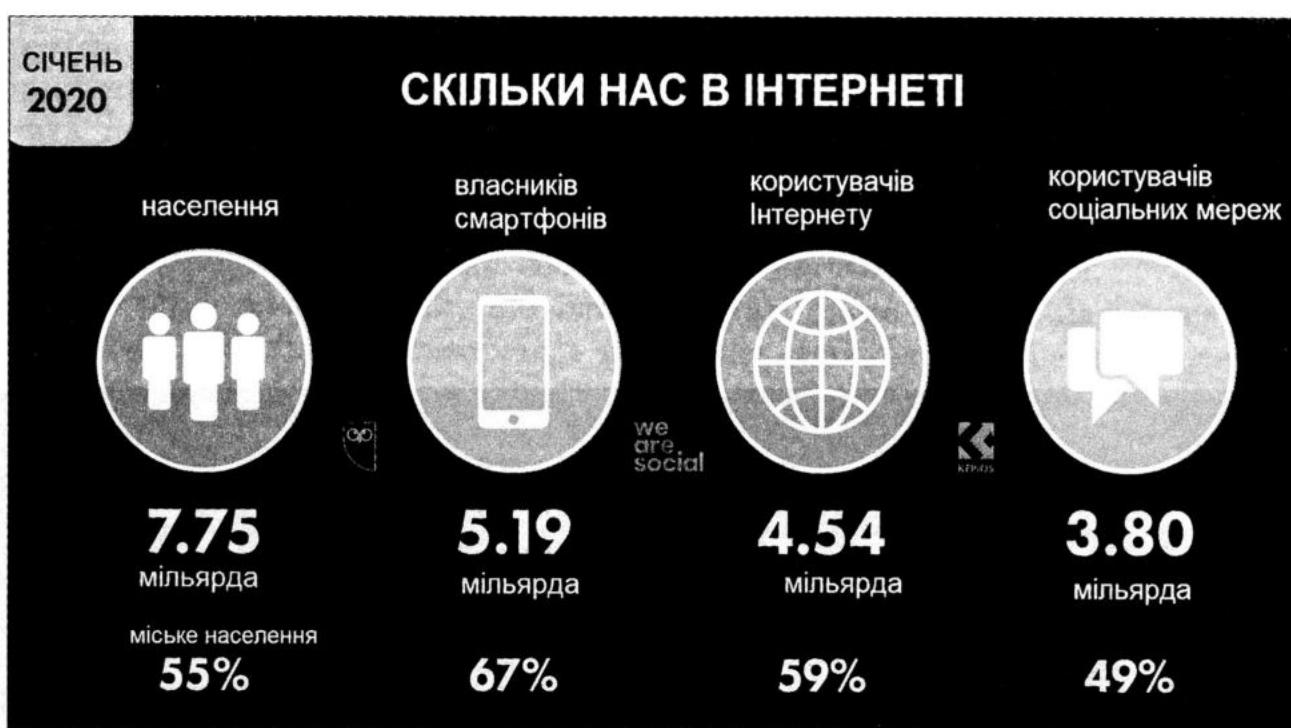


Рис. 1.1. Статистичні дані користування інтернетом у 2019 році за даними Аналітичної агенції We Are Social і SMM-платформи Hootsuite

- ♦ **Швидкість завантаження сайтів**

Під час завантаження сайта саме швидкість має вирішальне значення. У ході досліджень доведено, що людині достатньо трьох секунд для того, щоб справити враження на оточуючих. Це стосується й сайта: якщо за цей час він так і не завантажиться, великою є ймовірність того, що відвідувач далі не чекатиме.

- ♦ **Орієнтація на мобільні сайти**

Усе більше користувачів використовують для серфінгу інтернетом саме смартфони. 85 % користувачів впевнені, що на смартфоні сайт повинен виглядати не гірше (а то й краще), ніж на моніторі стаціонарного комп’ютера чи ноутбука.



Мобільний перегляд впевнено випереджає десктопний, і загальний дизайн сайтів стає все більш зручним для пальців. Дедалі частіше доведеться стикатися з навігацією, пристосованою до великого пальця.

Джош Кларк, фахівець у галузі мобільного дизайну, «дизайнер взаємодії», як він себе називає, який вдало поєднує ергономіку та психологію мобільного дизайну, у книзі «Проектування для дотику» (*Josh Clark, Designing for Touch*) досліджує, як користувачі тримають свої мобільні телефони та як їхні рухи мають оброблятися в процесі роботи з сайтом. На його думку, меню «гамбургер» (рис. 1.2), яке традиційно розташовується вгорі сайтів, буде перенесено в нижню частину мобільних екранів.



Рис. 1.2. Приклад меню «гамбургер»

Успіх сайта значною мірою залежить від якості дизайну, отже, дуже важливо стежити за трендом. Потрібно розуміти, які напрями ввійшли в моду у сфері сайтобудування, а які вже вважаються застарілими і від яких варто відмовитися. Адже тренди у вебдизайні постійно змінюються, удосконалюються, модернізуються.

Розглянемо напрями, які на початок 2020 року є найбільш популярними за версіями інтернет-журналів AWWWARD і Designmodo.

- ◆ **Візуалізація даних** — інфографіка сприймається як частина дизайну. Візуалізація даних є цікавим і наочним способом подання великих обсягів інформації.

Цей формат є дуже популярним серед користувачів, адже його можна застосовувати всюди: від чисел і карт до складних алгоритмів. Рис. 1.1 є яскравим прикладом подібної інфографіки.

- ◆ **Персоналізація сайтів** — найуспішнішими стануть сайти, орієнтовані на користувача, тобто ті, які надають інформацію з урахуванням статі, віку і навіть особистих уподобань користувача. Відповідно електронна комерція стане ще більш витонченою.

- ◆ **Зростання ролі типографіки** — зараз у тренді незвичайні шрифти, різноманітні художні елементи, креативні прийоми на кшталт складання текстів із «газетних вирізок». З'явився термін «вінтажна типографіка» (стиль старовинної типографії).

Серед вебдизайнерів набуває популярності використання шрифтів із зарубками (serif) (рис. 1.3), заокруглених плит і текстових елементів, які здаються більш старими.

Свого часу вважалося, що більш читабельними на екранах є шрифти без зарубок (рис. 1.4). Проте завдяки еволюції девайсів вибір шрифтів



перестає бути проблемою. Набувають популярності так звані варіативні шрифти. Змінні літери, які поступово збільшуються і зменшуються, привабливі для користувачів.

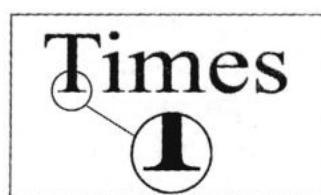


Рис. 1.3. Приклад шрифтів із зарубками

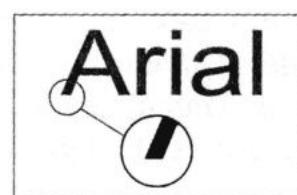


Рис. 1.4. Приклад шрифтів без зарубок

- **Пріоритетність відео-контенту** — на YouTube щодня переглядають близько 500 млн годин відео.

При цьому передбачається, що в 2020 році на призначений для користувача контент припадатиме 80 % усього інтернет-трафіку. На лідируючі позиції серед інших компонентів вебдизайну вийдуть повноекранні відео, оскільки вони є інформативними та зручними.

- **Синемаграфіка** — ці статичні картинки містять один анімований (динамічний) елемент. Наприклад, це може бути статична композиція з паруючою чашкою кави.

Таке поєднання фотографії та анімації є особливо привабливим з точки зору маркетологів.

- **Геометричні форми** (рис. 1.5) — використання у вебдизайні різних композицій геометричних форм було започатковано в 2016 році і, за прогнозами, у 2020-му лише набиратиме популярності.



Рис. 1.5. Використання геометричних форм у вебдизайні



- **3D-графіка** (рис. 1.6) — актуальний тренд, який можна назвати природною еволюцією плоского дизайну.

Тривимірні візуалізації забезпечують оновлення такого дизайну. Результатом є поєднання 3D-реалістичних і плоских інтерфейсів, які є складними та візуально цікавими. Вони «чіпляють» і привертають пильну увагу користувача.

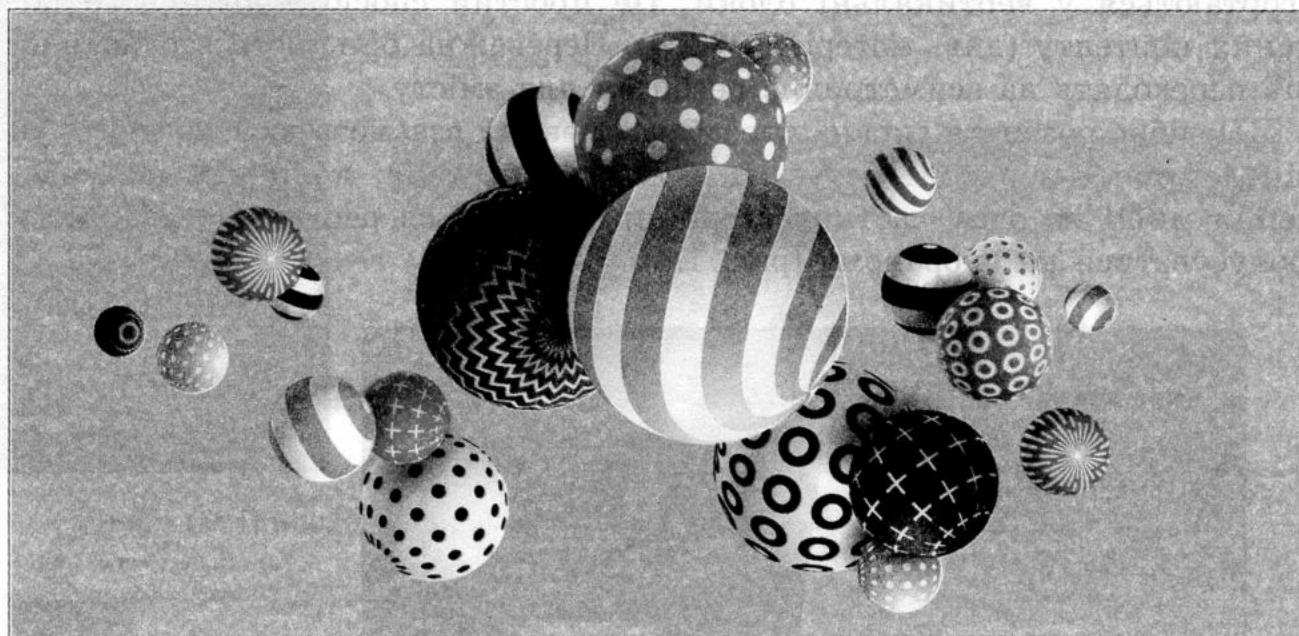


Рис. 1.6. Приклад тривимірної графіки

• Яскраві кольори

Це тренд, який ніколи не втрачає актуальності. Використання градієнту є багатофункціональною кольоровою тенденцією, що працює в дизайні будь-якого типу. Це може бути цікавим способом розбити елементи тексту, або виділити певний контент, або створити неординарне тло для розміщення продукту. З точки зору колористики застосування градієнтів (рис. 1.7) є чудовою ідеєю. Тло сторінки, у створенні якого задіяний градієнт, викликає враження унікальності й свіжості.

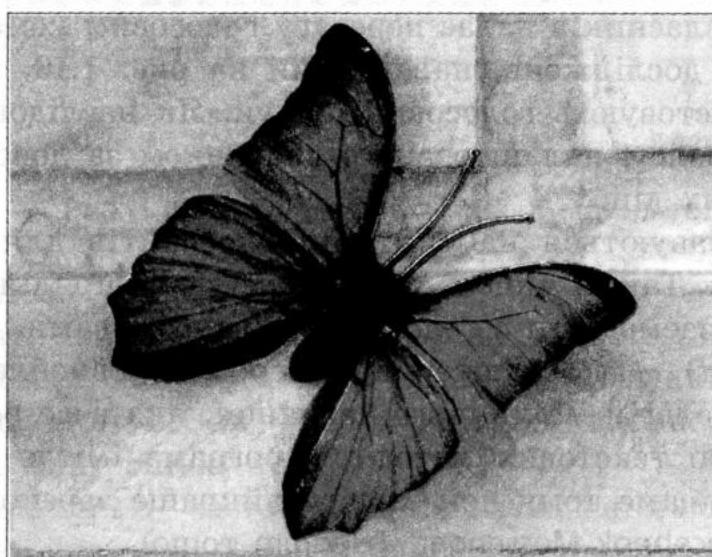


Рис. 1.7. Приклад використання градієнта



- ◆ **Застосування розділеного екрана (спліт-екрана)**

Ми неодноразово наголошували на тому, що дизайн сайтів має орієнтуватися на мобільних користувачів. Відповідно з'являються шаблони розділених екранів. Естетичний поділ екрана (насамперед за рахунок кольору) передбачає, що сайт, призначений для перегляду на великому моніторі, розподілено на дві панелі, які на екранах менших пристрій згортаються у вертикальні блоки. Це простий спосіб адаптивного подання контенту (див. матеріал § 2.4). Переважна більшість вебдизайнерів переходять на асиметричне розбиття для змісту.

Подібне розбиття екрана передбачає певний взаємозв'язок контенту — він доповнює або контрастує. Популярною є схема, коли під час наведення миші на одну частину сторінки затінюється решта, що дає змогу фокусуватися на вибраному (рис. 1.8).



Рис. 1.8.
Спліт-екран

- ◆ **Поширення використання штучного інтелекту**

Штучний інтелект так чи інакше вже використовують Amazon, Netflix та ін. Насамперед це стосується чат-ботів, які надають потрібну інформацію, наприклад під час обслуговування клієнтів (рис. 1.9).

Із зростанням кількості користувачів, які використовують смарт-пристрой у своїх будинках, і смартфонів для управління цими пристроями переважна більшість власників надає перевагу голосовим командам.

Згідно з результатами досліджень, наведеними на рис. 1.10, майже 43 % користувачів використовують голосовий пошук. Як наслідок зростає роль голосового інтерфейсу, налаштованого власником за допомогою ключових слів і відповідних дій.

Голосові інтерфейси базуються на «читанні» вебсайтів для отримання інформації та даних. Таким чином, все більше уваги приділяється розробці сайтів, здатних взаємодіяти з голосовими інтерфейсами.

Чат-бот (англ. *Chatbot*) — комп’ютерна програма, розроблена на основі нейромереж та технологій машинного навчання, яка веде розмову за допомогою слухових або текстових методів. Програма імітує розмову з людиною в інтернеті, саме тому цей сервіс найкраще зарекомендував себе в месенджерах (Facebook Messenger, Telegram тощо).



IBM Developer IBM Developer Courses

Build Chatbots with Watson Assistant

Course Overview

- + Get started
- + Lab 1: create a Watson Conversation instance
- + Intents
- + Lab 2: import and create intents
- + Entities

Course overview

Self-paced 5 hours Free

Chatbots and Watson: Let's talk abou...

Watson Conversation
Creating conversational experiences with Watson

Рис. 1.9. Приклад створення чат-боту засобами IBM Watson Assistans

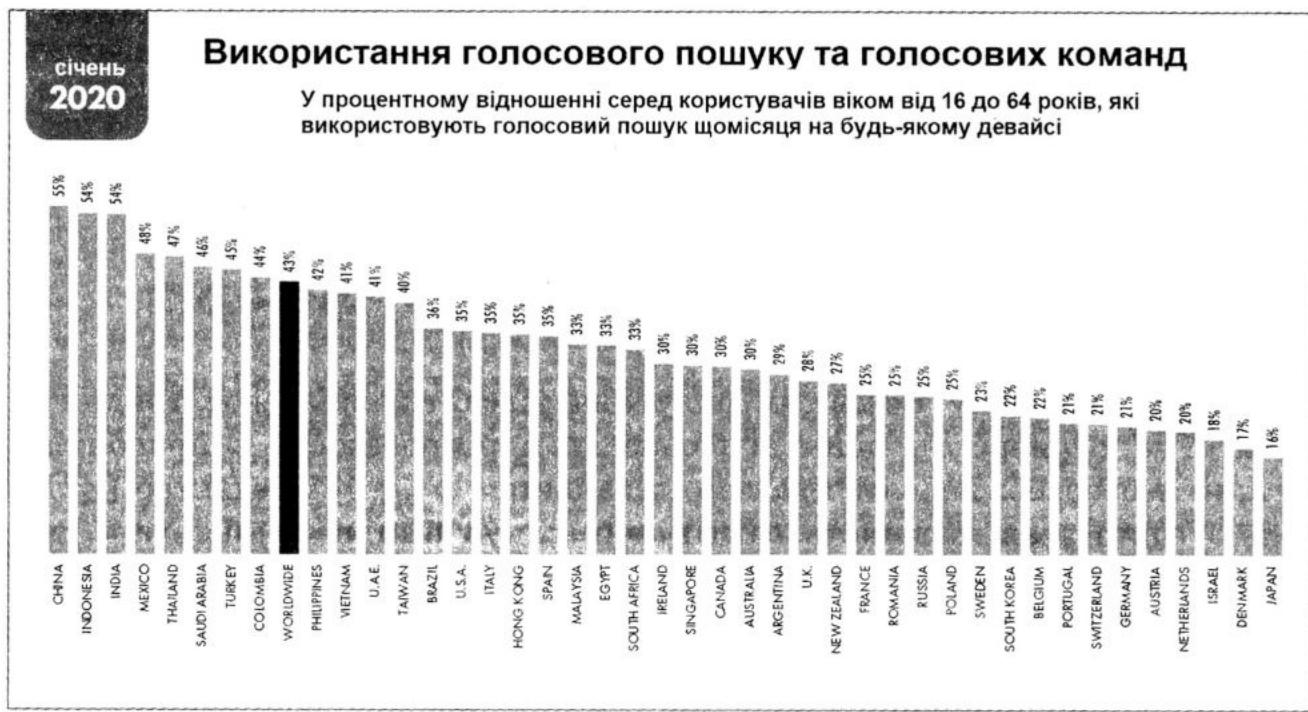


Рис. 1.10. Дослідження з використання голосового пошуку

Нині все більшої популярності набуває візуальний пошук. Його дозволяють використовувати Snapchat, Pinterest (соціальний фото-сервіс), Lens, AliExpress та Google Lens. Піонером у візуальному пошуку був Amazon із його функцією Flow. У 2014 році він використовував розпізнавання зображень для пошуку товару.

Google Lens — застосунок для визначення об'єктів в об'єктиві камери. Він може виділяти назви в тексті і навіть телефони, відразу дозволяючи



зателефонувати за вказаним номером. Він розпізнає більш ніж 1 млрд об'єктів.

У вересні 2018 року Snapchat (мультимедійний мобільний додаток обміну фото та відео) у співробітництві з Amazon оголосив про розробку нової функції. Вона дозволяє користувачеві «сканувати» штрих-коди за допомогою камери.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке чат-бот? Де використовують чат-боти?
2. Що таке синематографіка?
3. Наведіть приклади використання штучного інтелекту в сучасних онлайн-платформах.
4. У чому перевага розподілених екранів?
5. Знайдіть в інтернеті статистику за 2017 та 2018 роки, аналогічну наведеній на рис. 1.1. Яку тенденцію ви можете простежити?
6. Які тренди, на вашу думку, будуть актуальними в найближчому майбутньому?



Дивіться презентацію «Основні тренди у вебдизайні».

1.2

Види сайтів та цільова аудиторія

У березні 2019 року Всесвітня павутина відсвяткувала своє тридцятиріччя. Тім Бернерс-Лі, один із головних її розробників, запропонував перший вебсервер, перший браузер (детальніше в § 2.5) і редактор (*World Wide Web*). Наразі кількість сайтів перевищує півтора мільярда і продовжує зростати. Щосекунди у світі з'являються два сайти (рис. 1.11). Дізнатись кількість існуючих сайтів можна завдяки проекту Internet Live Stats (<http://www.internetlivestats.com/total-number-of-websites/>).



Рис. 1.11. Кількість сайтів на сьогодні



Ознайомимося з класифікацією сайтів (рис. 1.12).

Інформаційні сайти є одними з найбільш популярних у мережі інтернет і призначені для донесення до користувача будь-якої інформації.



До інформаційних сайтів відносять тематичні інформаційні, новинні та блоги.



Рис. 1.12. Класифікація сайтів

Тематичні інформаційні сайти — ресурси, в яких практично всі сторінки присвячені одній певній проблематиці або темі. У свою чергу різноматичні сайти охоплюють широке коло інформаційної спрямованості і можуть висвітлювати велику кількість тематик і напрямків.

Новинні сайти виконують дуже важливу завдання — вони повинні донести до користувача різні новини, що відбувалися, відбудуться чи відбуваються в цей момент. Такі сайти можуть спеціалізуватись на одній певній тематиці або розповідати інтернет-спільноті про новини з різних життєвих сфер людини.

Блоги зустрічаються у Всесвітній павутині досить часто, рівень їхньої популярності стає дедалі вищим. Як і новинні сайти, блоги призначені для того, щоб донести інтернет-спільноті ту чи іншу інформацію, але з однією істотною відмінністю: автор описує свою особисту думку про те, що відбувається.



Умовно блоги діляться на дві категорії: особисті та корпоративні, кожна з яких виконує певну функцію.

Особисті блоги — категорія сайтів, у яку входять ресурси, створені окремими користувачами, з метою донести широкому колу громадськості думку автора з певної теми.

Корпоративні блоги зазвичай створюються певними компаніями як додатковий ресурс, який розкручує бренд, і часто є доповненням до комерційного або корпоративного сайта. Вони виконують одне з двох певних завдань: зміцнення зв'язку між філіями та відділеннями або підвищення іміджу в очах потенційних та існуючих клієнтів.

Залежно від того, яку зі згаданих проблем має вирішувати корпоративний ресурс, сайти поділяють на іміджеві та інформаційні. *Іміджевий корпоративний сайт* виконує одну з найважливіших функцій для компанії — рекламну. На ньому зазвичай описується історія підприємства, компанії або торгової марки, надаються докладні поточні відомості, контактна інформація, містяться відомості про товари, що виробляються та розповсюджуються, про послуги, які надаються.

На *іміджевих сайтах* можуть публікуватися різні новини, інформація про акції, знижки — все те, що характеризує компанію перед клієнтом. Такі сайти відрізняються оригінальним ексклюзивним дизайном, поєднанням нестандартних елементів оформлення і рішень.

Інформаційний корпоративний сайт виконує не менш значущі функції — автоматизації та зміцнення зв'язку між відділами та філіями, обігу документів, управління персоналом тощо.

Пересічні користувачі інтернету зазвичай не мають відкритого доступу до інформаційного корпоративного сайта, оскільки він призначений виключно для співробітників однієї організації.

Сайти-портфоліо і **сайти-візитки** сьогодні все частіше створюються приватними особами, особливо фрілансерами, які намагаються привернути увагу нових замовників і клієнтів, розрекламувавши свою роботу і надавши результат її виконання.

Сайти-візитки призначені для того, щоб коротко подати інформацію про свого власника широкому колу користувачів. Такі сайти налічують більше ніж 20 сторінок.

Сайти-портфоліо призначені привертати увагу широкого кола потенційних користувачів шляхом рекламивання роботи, виконаної компанією, щоб клієнт зміг вирішити, чи потрібно користуватися послугами цієї компанії.

Комерційні сайти — одна з найбільш поширених категорій сайтів. Їх основне призначення — продаж товарів інтернет-користувачам.



Комерційні сайти поділяють на кілька підкатегорій, найбільш поширеними з яких є сайт-вітрина, промо-сайт, інтернет-магазин.

Перші системи електронної комерції з'явилися понад 50 років тому у США для замовлення транспортних квитків. А вже у 1990-х виникли



перші інтернет-магазини, де люди могли купувати товари та оплачувати їх банківськими картками. Наразі це є однією з найпотужніших індустрій із величезним грошовим обігом, яка швидко розвивається (рис. 1.13). І слід відзначити, що практично кожен українець від 1 до 12 разів на рік купує онлайн.

Сайти-вітрини, або *посадкові сторінки (landing page)* — сайти, основним призначенням яких є не продаж готової продукції, а її реклама. Найбільшого поширення вони набули серед компаній і підприємств, які виробляють товари.



Рис. 1.13. Купівля онлайн

За допомогою *сайта-вітрини* неможливо здійснити операцію купівлі-продажу, оскільки він лише надає докладну інформацію про товар. Але іноді на них зазначаються місця, де можна купити товар, який зацікавив.

Промо-сайти — інтернет-ресурси, на яких рекламиують певні послуги або товари, зупиняючи особливу увагу на їхніх перевагах. Вони характеризуються досить простою структурою, невеликим обсягом (до 10 сторінок) і нетривалим терміном існування.

На сторінках промо-сайта містяться різні графічні матеріали, контактні дані компанії тощо. Якщо необхідно провести масштабну компанію певного товару, промо-сайт стане чудовим помічником.

Інтернет-магазини призначені для продажу товарів різних категорій у мережі в режимі онлайн. Відвідавши такий сайт, кожен користувач Всесвітньої павутини може вибрати товар, що його цікавить, дізнатися про нього докладну інформацію і, в разі необхідності, здійснити замовлення.

Дуже часто компанії пропонують користувачам усі свої лінійки товарів на одному сайті. Значно рідше створюються кілька ресурсів, на яких представлено одну певну категорію поширюваної продукції.

Соціальні проєкти включають форуми, спеціалізовані соціальні мережі, мережі загальної спрямованості, сайти-спільноти та ін.

Форуми — спеціалізовані сайти, на яких користувачі мережі інтернет можуть безперешкодно обговорювати різні новини, проблеми, життєві ситуації тощо.

Форуми можуть бути як *спеціалізованими* (обговорюються одна або кілька проблем, пов'язаних між собою), так і *загальноспрямованими* (обговорюються проблеми, абсолютно різні за тематикою, часом зовсім не пов'язані одна з одною).

Соціальні мережі покликані надавати інтернет-користувачам можливість знаходити одне одного і вести між собою спілкування в режимі онлайн. Кількість користувачів соціальних мереж невпинно зростає (рис. 1.14).

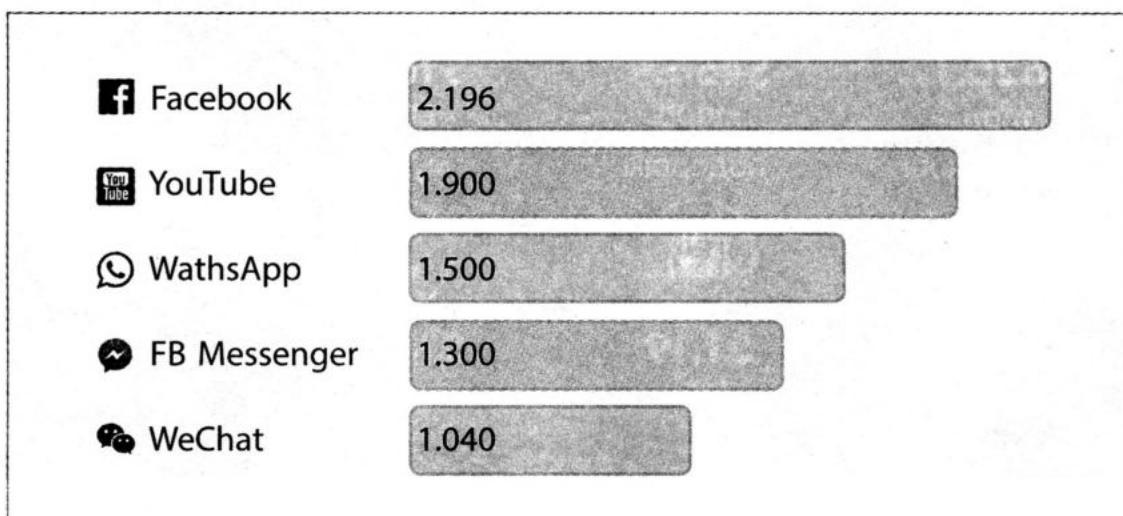


Рис. 1.14. Кількість користувачів соціальних мереж
(у мільярдах)

Існують спеціалізовані соціальні мережі, які об'єднують людей зі спільними інтересами (хобі, професіями) (рис. 1.15).

Соціальні мережі загальної спрямованості збирають в одне коло для спілкування людей, які зайняті в абсолютно різних сферах і мають різні інтереси і хобі (рис. 1.16).

Окрему категорію сайтів становлять **освітні ресурси**. У 9 класі ви ознайомилися з класифікацією освітніх та навчальних ресурсів. Пригадаємо, що освітніми інтернет-ресурсами називають ресурси освітнього характеру, розміщені у вебпросторі.

Особливу увагу наразі приділяють відкритим онлайн-курсам на платформах Prometheus, EdEra, EdX, Coursera (рис. 1.17).



Рис. 1.15.
Спеціалізована
соціальна мережа
LinkedIn

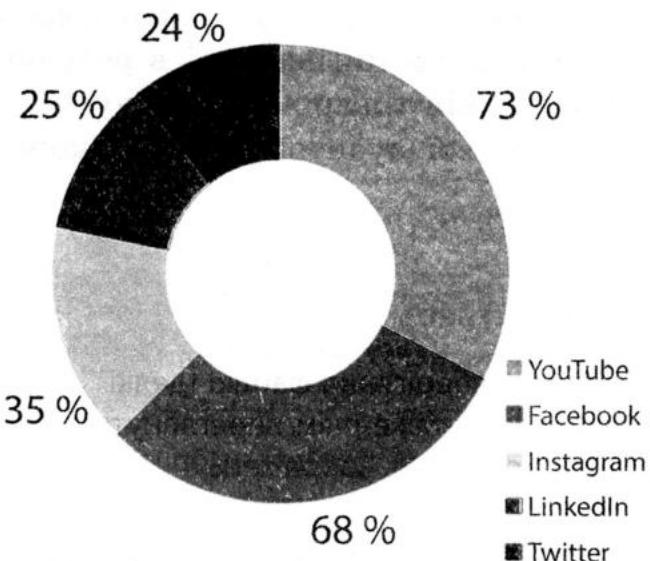


Рис. 1.16. Співвідношення користувачів найбільш впливових соціальних мереж

Розважальні портали впевнено тримають лідерство за популярністю нарівні з соціальними мережами. Найпопулярнішим є *відеохостинг*. Найбільш відвідуваними є ресурси, що безкоштовно пропонують відвідувачам онлайн-перегляд і скачування фільмів, відеороликів. На другому місці — *сайти музичного напрямку*. За структурою вони подібні до відеохостингів, але пропонують безкоштовне прослуховування й скачування мелодій. Третє місце посідають класичні комплексні *сайти розваг*. Тут користувачі шукають інформацію відповідно до своїх інтересів: це може бути графіка, відео, тексти різної тематики і, безумовно, ігри.

Ігровий портал — складний розважальний інтерактивний проект, що передбачає велику відвідуваність і ресурсомісткість. На порталі зазвичай є розділ для пошуку потрібної інформації за темою, блок новин і величезна кількість посилань на ігри з їхнім описом. Часто ігрові портали містять спеціалізовані й добре модеровані форуми.

Вебсервіси нині отримали значне поширення (рис. 1.18).

Сенс існування будь-якого сайта — це його відвідуваність. І якщо блогер публікує свою думку, розраховуючи на зворотний зв'язок у вигляді коментарів або лайків, то, наприклад, інтернет-магазин існує за рахунок реальних покупців, які шукають і, головне, купують товар.



Рис. 1.17. Платформи з онлайн-курсами

Цільова аудиторія сайта, цільові відвідувачі — група інтернет-користувачів, на яку сфокусовано зміст сайта; коло відвідувачів, зацікавлених в інформації, товарах або послугах, що презентують на сайті.

Цільові відвідувачі точно знають, в отриманні якої інформації вони зацікавлені і який саме товар або послугу бажають придбати.



Виділення цільової аудиторії з усіх відвідувачів сайта дозволяє точніше направити інформаційний або рекламний вплив і в результаті веде до розвитку бізнесу (збільшення продажів товарів або послуг).

Будь-який сайт, крім власне цільової аудиторії, має також побічну і випадкову аудиторію.



Рис. 1.18. Класифікація вебсервісів

Побічною аудиторією є користувачі, які приходять із пошуку по запитах, суміжних із семантичним ядром сайта. Те саме можна сказати і про людей, які начебто «автоматично» клащають на рекламу, ще не знаючи, потрібна їм послуга чи ні. Вони можуть стати клієнтами, проте це відбувається нечасто. Отже, побічна аудиторія теж є цільовою.

Визначення цільової аудиторії полягає в складанні приблизного портрета цільового відвідувача сайта (так званого портрета клієнта).

Для отримання даних, які складають портрет цільового відвідувача, використовують різні способи збирання інформації.

Існує кілька способів збирання інформації про аудиторію сайта.

Спосіб 1	Лог-аналізатор сервера і дані лічильника відвідувань (дозволяють вивчити всі дії користувачів на сайті й конкретизувати розподіл аудиторії сайта за регіонами, часом тощо)
Спосіб 2	Опитування аудиторії сайта (анкетування унікальних відвідувачів із використанням опитувальної форми або реєстрації на сайті)

Закінчення таблиці

Спосіб 3	Опитування аудиторії на сайтах опитувань, поєднання панельних даних і даних лічильника відвідувань (анкетування відбувається на сайті панелі)
Спосіб 4	Системи аудиту й традиційні опитування дослідницьких компаній (агентств)
Спосіб 5	Моніторинг соціальних мереж

Панель — це збирання даних із певної групи користувачів, яке повторюється через рівні проміжки часу. Тобто панель — це різновид безперервної вибірки, яка дозволяє зафіксувати зміни характеристик, що цікавлять дослідника. Панельне опитування використовують при вивчені думок споживачів певної групи за визначений проміжок часу, коли визначаються потреби, звички, смаки та рекламиації цих споживачів.

Дослідження великих аудиторій показують, що комунікацію легше налагоджувати з малими групами покупців, об'єднаними в одну цільову аудиторію.

Існує чотири основні принципи сегментації цільової аудиторії сайту (рис. 1.19).

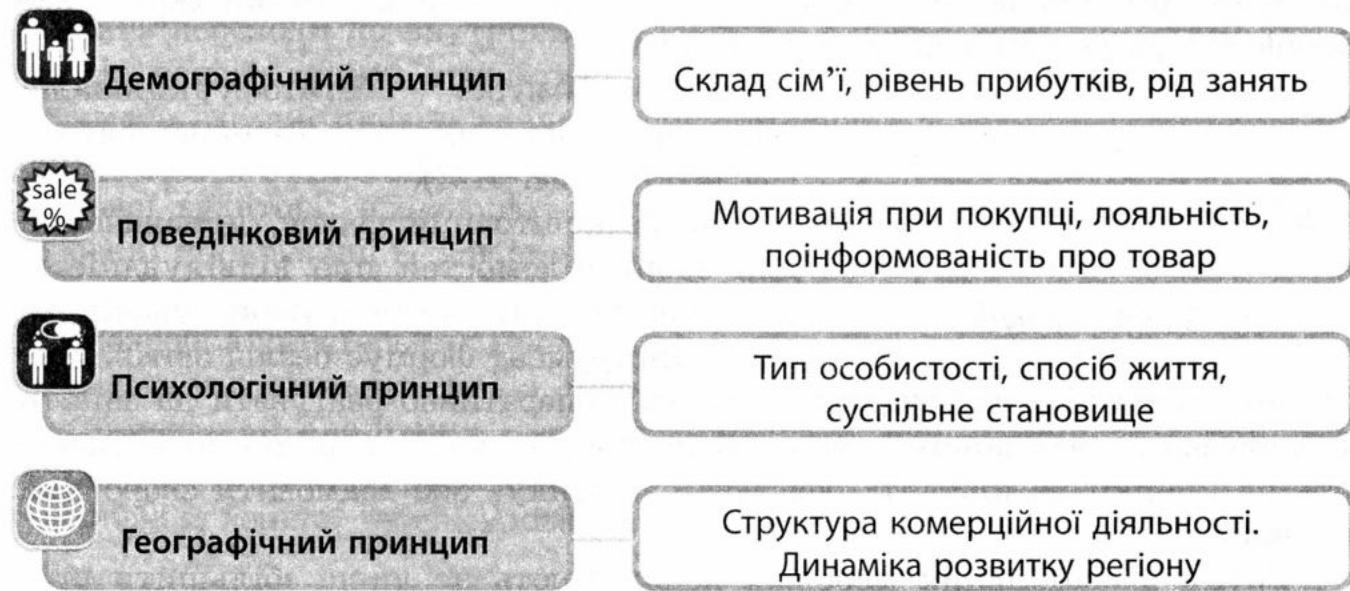


Рис. 1.19. Сегментація цільової аудиторії

Досить цікаву сегментацію запропонував Марк Шерінгтон, засновник всесвітньо відомої бренд-консалтингової компанії Added Value, назвавши її 5W (рис. 1.20). Він виявив ефективну формулу визначення цільової аудиторії, що базується на п'яти ключових питаннях.

1. **Що?** (Який товар або послугу надають — сегментація за типом товару.)
2. **Хто?** (Потенційний клієнт — сегментація за типом споживача.)
3. **Чому?** (Що мотивує, «підштовхує» клієнта придбати товар, який пропонується, — сегментація за типом мотивації здійснення покупки.)

4. **Коли?** (У яку пору року/час дня покупець здійснює покупку — сегментація за ситуацією, коли здійснюється покупка.)
5. **Де?** (Де здійснюється придбання товару, за яких умов — сегментація за каналами продажу.)



Рис. 1.20. Сегментація 5W

Важливим джерелом статистичних даних про цільову аудиторію сайта є статистика запитів пошукових систем. Оцінити величину цільової аудиторії можна за кількістю пошукових запитів. Такий сервіс має Google (<https://www.google.com/analytics>).



Рис. 1.21. Google Analytics

Google Analytics — багатофункціональний сервіс для аналізу інтернет-сайтів і додатків (рис. 1.21).

Це безкоштовний сервіс, призначений для збору відомостей про відвідуваність, а також про дії інтернет-користувачів на сайті. Інструмент формує безліч звітів. Усебічний аналіз цільової аудиторії дозволяє оперативно реагувати на запити користувачів. Веб-майстер зможе дізнатися, як часто відвідувачі натискають на кнопки на кшталт «оформити покупку» або «замовити зворотний дзвінок».

Групи в соціальних мережах допомагають не лише збільшити продажі компанії, а й просувати сайт компанії в пошукових системах. Існує безліч сервісів моніторингу авторитету сайта за кількістю згадок про нього в соціальних мережах, особливо в таких, як Facebook, а також твіти із зазначеними посиланнями на сайт.

Hootsuite (<http://hootsuite.com>) — відмінний і, мабуть, один із найпопулярніших багатофункціональних сервісів для роботи з соціальними медіа. Акцент у сервісі зроблено на роботу з Twitter. У першу чергу Hootsuite буде корисний тим, хто веде кілька акаунтів відразу. Hootsuite успішно працює також з акаунтами Facebook, LinkedIn, MySpace і Foursquare, з блогами на WordPress; підключається до Ping.fm, що дозволяє оновлювати сторінки понад 40 соціальних мереж.



У Hootsuite є багато можливостей для різноманітної аналітики. Наприклад, можна підключити Google Analytics зі свого сайта й дивитися графіки, порівнюючи з кількістю твітів і популярністю ваших посилань.

Socialmention (<http://www.socialmention.com/>) — одна з найпотужніших платформ для безкоштовного пошуку й аналізу інформації в соціальних мережах. Система шукає згадки у відібраних сервісах або в усіх відразу. Крім того, пропонує аналіз тональності згадувань, пов'язані ключові слова, популярні джерела та багато іншого.

Socialmention шукає по більш ніж 100 соціальних медіа, включаючи мережі, соціальні закладки, блоги, форуми, соціальні сервіси та ін.

Google Alerts (<https://www.google.com/alerts>) — простий і зрозумілий безкоштовний сервіс, який сканує публікації за заданими ключовими словами, висилаючи сповіщення електронною поштою відповідно до частоти, налаштованої користувачем. Є можливість фільтрувати результати залежно від регіону, мови та джерел, у тому числі блогів і форумів.

Socialbakers (<http://www.socialbakers.com>) — масштабний статистичний сервіс, який називає себе «Серце статистики Facebook». Крім Facebook, Socialbakers безкоштовно надає моніторинг інформації за різними показниками в Twitter, Google+, LinkedIn тощо. Socialbakers відомий своїми рейтингами брендів на Facebook. Він збирає статистику по сторінках інтернет-ресурсу. Власник сайта зможе дізнатися, як часто відвідувачі роблять переходи по внутрішніх і зовнішніх посиланнях.

На основі даних, отриманих у результаті збору інформації, можна отримати зведені цифри, вивчити закономірності поведінки груп користувачів і оцінити ефективність рекламного впливу. Характеристики портрета цільової аудиторії враховуються при розробці дизайну і структури, при внесенні коригувань в інформаційне наповнення сайта з метою залучення більшої кількості відвідувачів, що входять у коло цільової аудиторії.

Розробка українських програмістів — система моніторингу та аналізу посилань у соціальних медіа **YouScan** (<https://youscan.io>) — має потужний інструментарій аналітики і вміє розпізнавати зображення за допомогою штучного інтелекту. Серед клієнтів сервісу є такі брендові компанії, як McDonald's, Ferrero, Bosch, Vodafone.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Скільки груп сайтів ви можете класифікувати?
2. Що входить до соціальних проектів?
3. Що таке цільова аудиторія сайта? Назвіть її основні характеристики.
4. Які основні групи цільової аудиторії сайта?
5. Які перспективні способи збирання інформації про цільову аудиторію?
6. Візуалізуйте будь-яким засобом результати вашого опитування щодо того, які сайти є вподобаннями ваших друзів.



Дивіться презентацію «Види сайтів та цільова аудиторія».

1.3

Інформаційна структура сайта

Для користувача простота навігації сайтом — важливий чинник, що позитивно впливає на поведінкові фактори і, як наслідок, на видимість, позиції і трафік.



Інформаційна структура сайта — спосіб організації інформаційних даних на вебсайті, а також структура взаємодії різних блоків інформації один з одним.

В ідеалі така структура повинна бути максимально зручною, щоб користувач міг швидко знайти потрібну йому інформацію. Інакше кажучи, структура сайта — це логічна схема побудови сторінок сайта з розподілом за теками і категоріями.

Сайти можуть мати *лінійну* та *ієрархічну* структуру (рис. 1.22). Також є сайти з *довільною* структурою, на яких навігація здійснюється в довільному порядку. Сайтом із довільною структурою є, наприклад, Вікіпедія. Структура є дуже важливим фактором ранжування, покращуючи характеристики всього сайта.

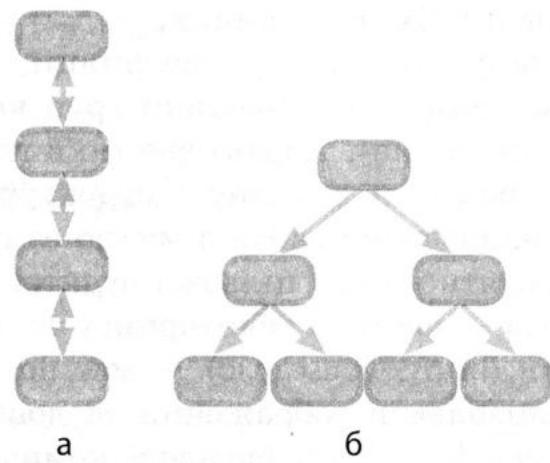


Рис. 1.22. Структура сайтів: лінійна (а) та ієрархічна (б)

Основною складовою будь-якого сайта є документ — це сторінка сайта, що має унікальну адресу в інтернеті. Документи можуть бути як простими інформаційними сторінками, так і картками товару або, наприклад, лістингами товарів в інтернет-магазині.

Якщо сайт являє собою ієрархічну (деревоподібну) структуру, то головну сторінку можна уявити як «стовбур», а розділи і статті — як гілки й листя. Наскільки широкою буде структура, залежить від формату і типу сайта. Але навіть сайт, що складається з однієї сторінки, вже є основою, або «стовбуром», такої структури, від якої він може розвиватися в різних напрямках.

Ієрархічна структура є базисом для будь-яких інших видів структури, в яких змінюється вид зв'язку документів — чи це лінійний зв'язок, де кожен документ пов'язаний з наступним і з головною сторінкою, чи більш широкий, де документи пов'язані між собою і з головною сторінкою.



Принцип побудови зв'язків між документами залежить від типу сайта. Загалом можна виділити такі типи сайтів, які мають відмінності в структурах: сайт-візитка (рис. 1.23), комерційний сайт (рис. 1.24), інтернет-магазин, інформаційний сайт. Кожен тип передбачає свій вид структури.



Рис. 1.23. Структура сайта-візитки

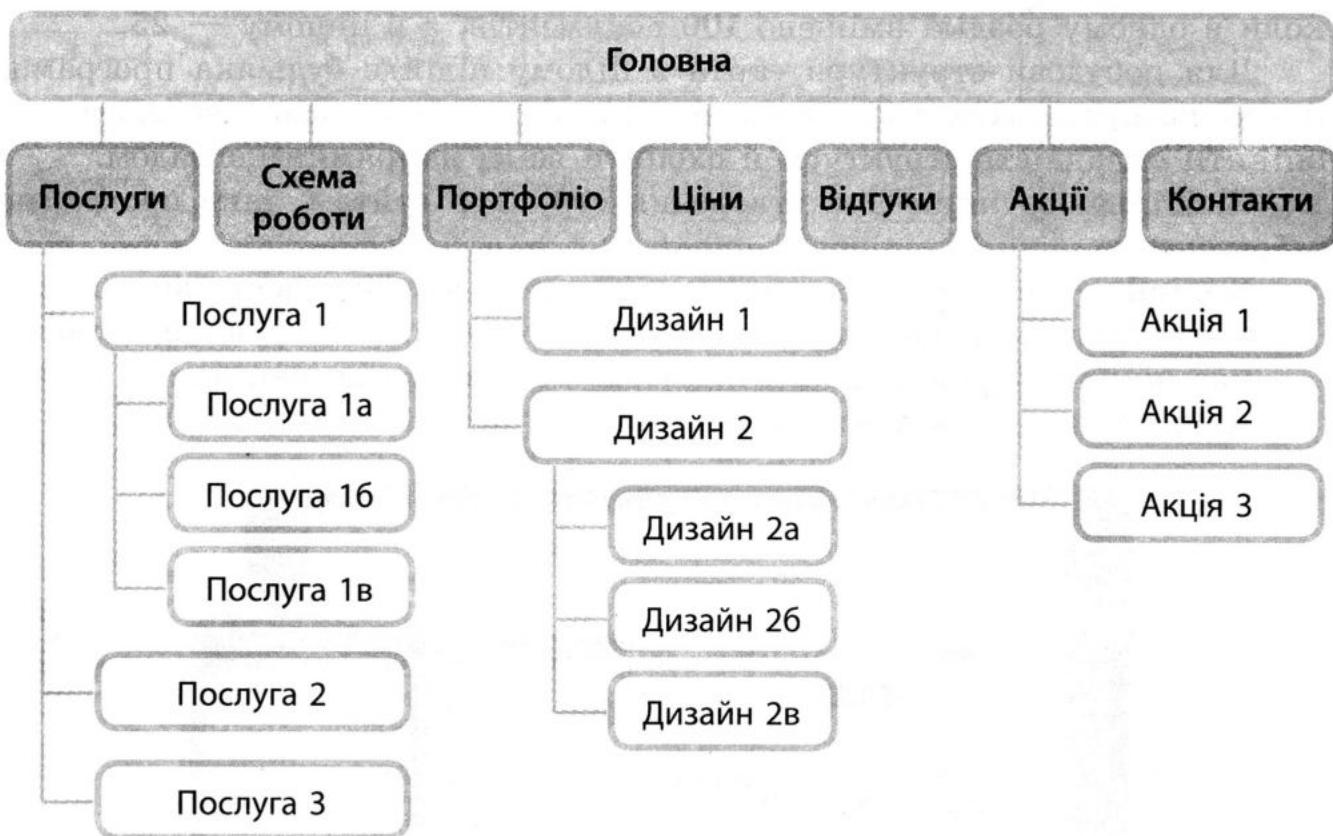


Рис. 1.24. Структура комерційного сайта

Наразі все більшу популярність набувають так звані *посадкові*, або *лендінг* (від англ. *landing page*), сторінки. Посадкова сторінка розробляється з урахуванням психології відвідувачів, вона має чіпляти їх до самого моменту покупки (або реєстрації). Товарна сторінка служить для продажу товару. Це просунута картка, яку можна зустріти

в будь-якому інтернет-магазині, з описом товару, умовами доставки, контактним телефоном. Головною тут буде кнопка Купити. Це може бути передплатна сторінка, завдання якої зводиться до отримання електронної адреси користувача, який зайдов на сторінку. Таким чином, збирається база даних з інформацією про потенційних покупців.

Структура необхідна для індексації сайта пошуковими системами. У пошукових систем є ціла низка своїх вимог до структури. Чим більш правильно її логічно вона спроектована, тим простіше пошуковій системі проіндексувати сторінки і надати їх користувачеві.

Правильно розроблена структура сайта дає можливість пошуковим системам швидко здійснити обхід ресурсу, а відвідувачам — легко переміщуватися між вебсторінками.

Складність структури сайта визначається двома параметрами: рівнем вкладеності і збалансованістю.

Рівень вкладеності — це кількість переходів, які потрібно зробити з головної сторінки до найдальшого документа в структурі. Не рекомендується, щоб це значення було більше від 3.

Збалансованість визначають «на око», оцінюючи кількість документів усередині розділів і рубрик. Не повинно бути помітних розбіжностей, коли в одному розділі вміщено 100 документів, а в іншому — 25.

Для побудови структури сайта в цілому підійде будь-яка програма, що може працювати з діаграмами. Створення схеми дозволяє візуально оцінити складність структури й охопити всі її напрямки поглядом.

Найбільш зручним для створення структур сайта є таке програмне забезпечення.

- **X-mind** — безкоштовна програма для побудови інтелект-карт, структур і діаграм різних процесів. Вона досить проста у використанні, має широкий функціонал і підтримується всіма операційними системами, використовується на мобільних пристроях (рис. 1.25).

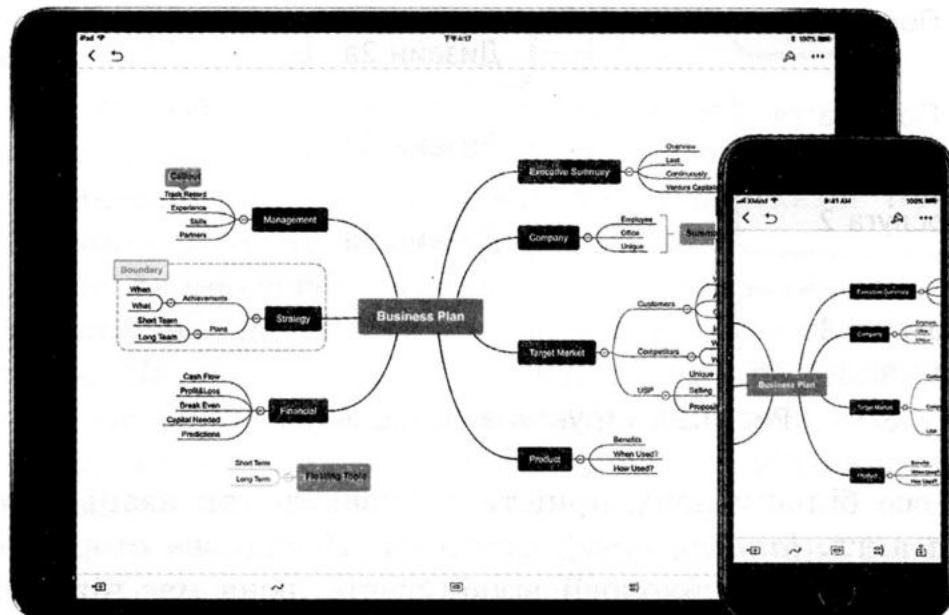


Рис. 1.25. Використання X-mind на мобільних пристроях



- **Draw.io** — безкоштовний онлайн-сервіс для створення структур і діаграм. Він вимагає наявності пошти від Google. В арсеналі має більше інструментарію, ніж у X-mind, наявна низка технічних і інженерних шаблонів.

Ще один плюс Draw.io — можливість завантаження результату на Google-диск та інші сервіси нарівні із завантаженими на комп'ютер.

На рис. 1.26 подано структуру сайта вивчення інформатики, розроблену за допомогою Draw.io.

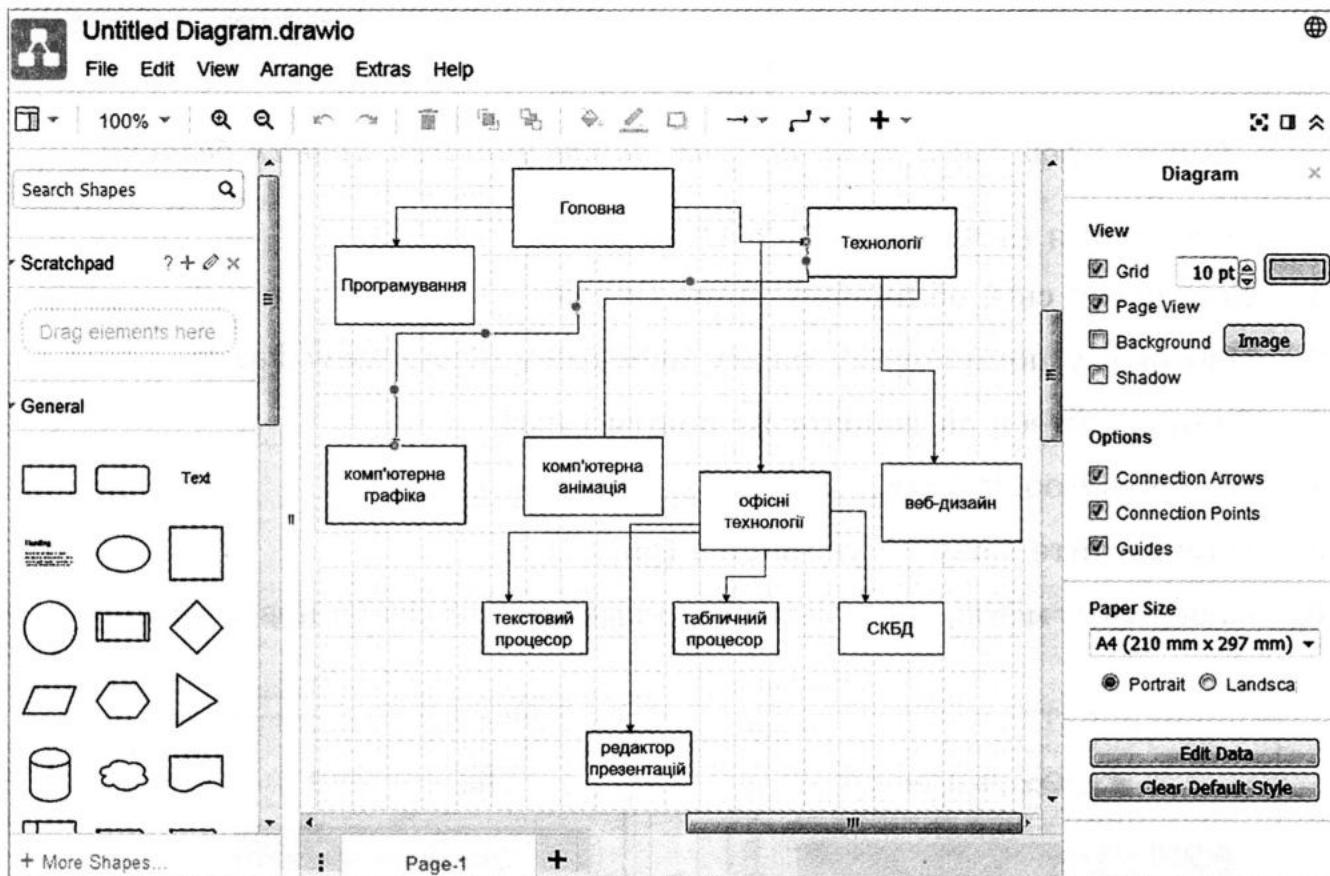


Рис. 1.26. Структура сайта вивчення інформатики, зроблена за допомогою Draw.io



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке інформаційна структура сайта?
2. Назвіть параметри складності структури. Як вони розрізняються?
3. Поміркуйте, яка структура характерна для будь-якого сайта.
Опишіть цю структуру.
4. Яка мета створення посадкової сторінки?
5. Обґрунтуйте, чим важлива структура сайта.
6. Створіть за допомогою будь-якого онлайн-сервісу структуру інформаційного сайта; інтернет-магазину.



Дивіться презентацію «Інформаційна структура сайту».

Практична робота № 1

ТЕМА. Створення структури сайта за допомогою онлайн-сервісу Draw.io

ЗАВДАННЯ: розробити структуру шкільного сайта підтримки викладання інформатики засобами онлайн-сервісу Draw.io з відображенням основних напрямків навчання.

ОБЛАДНАННЯ: комп’ютер із доступом до мережі інтернет.

Хід роботи

Під час роботи з комп’ютером дотримуйтесь правил безпеки.

Налаштування онлайн-сервісу Draw.io

1. Увійдіть у свій обліковий запис Google.
2. Наберіть у пошуковому рядку url-адресу <https://www.draw.io>.
3. Авторизуйтесь за допомогою пошти Gmail.
4. Виберіть Google Диск.
5. Налаштуйте мову — українська (рис. 1).
6. Виберіть у меню, що з’явилося, команду Створення нової діаграми.

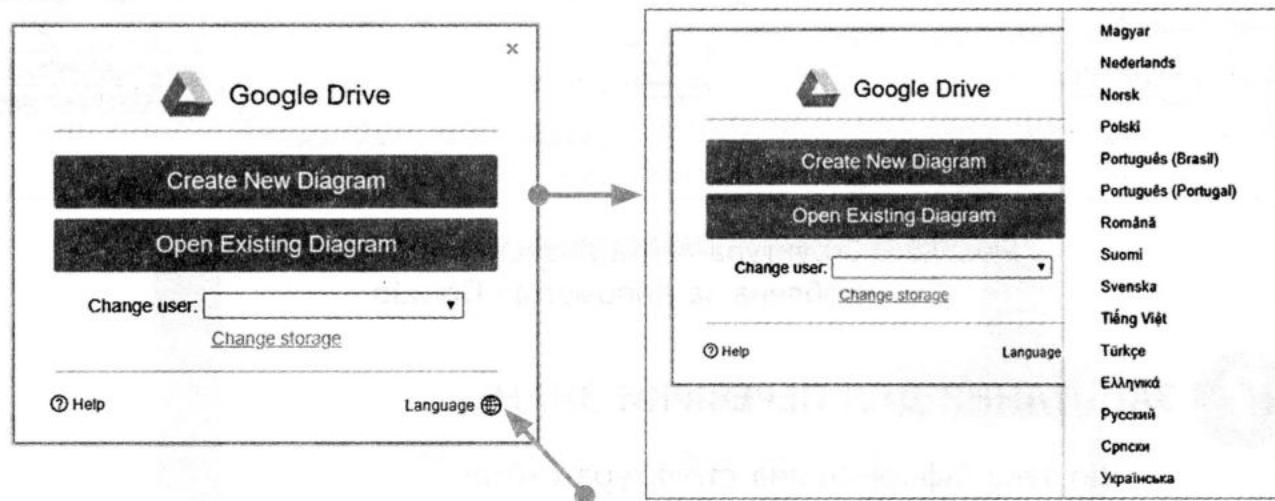


Рис. 1. Налаштування мови та вибір команди Створення нової діаграми

Створення структури

1. Виберіть із фігур, розташованих у лівому полі програми, прямокутник і перетягніть його на поле діаграми (рис. 2).
2. За допомогою меню на правому полі Стиль та Текст (рис. 3) відформатуйте прямокутник (виберіть колір тла й розмір, колір і шрифт тексту).

3. Додайте на прямокутник текст Головна сторінка.
4. Аналогічно створіть елементи другого рівня Програмування та Інформаційні технології. Виберіть для них інший стиль форматування.

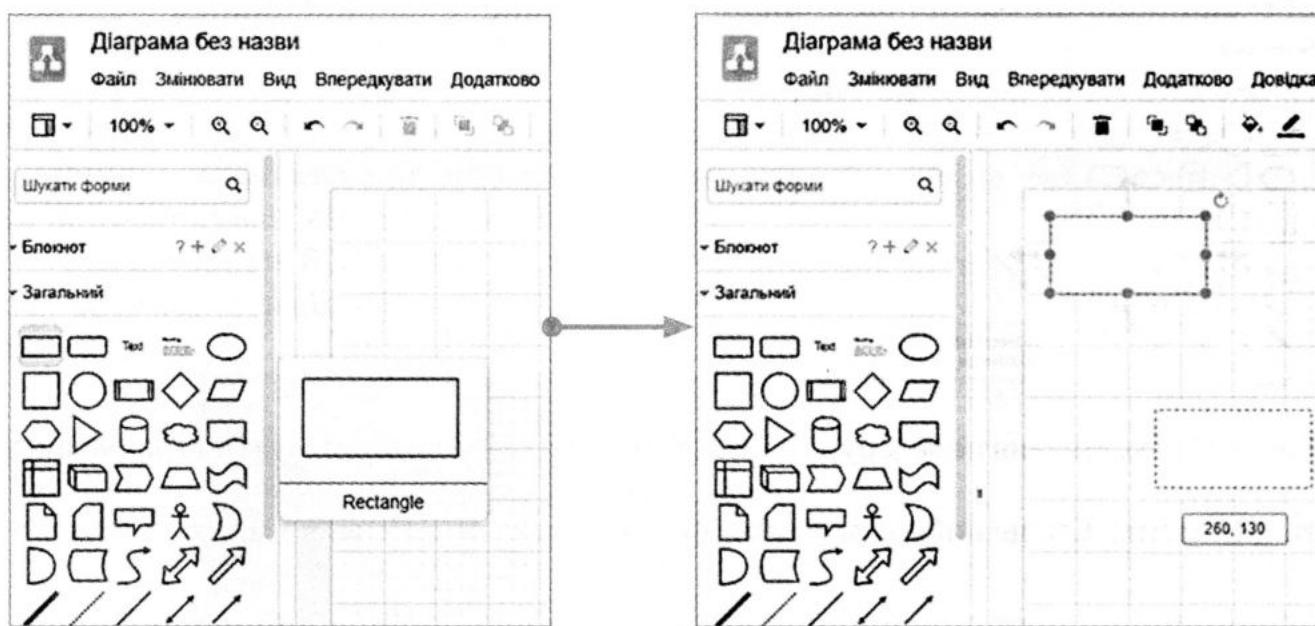


Рис. 2. Вибір елемента діаграми

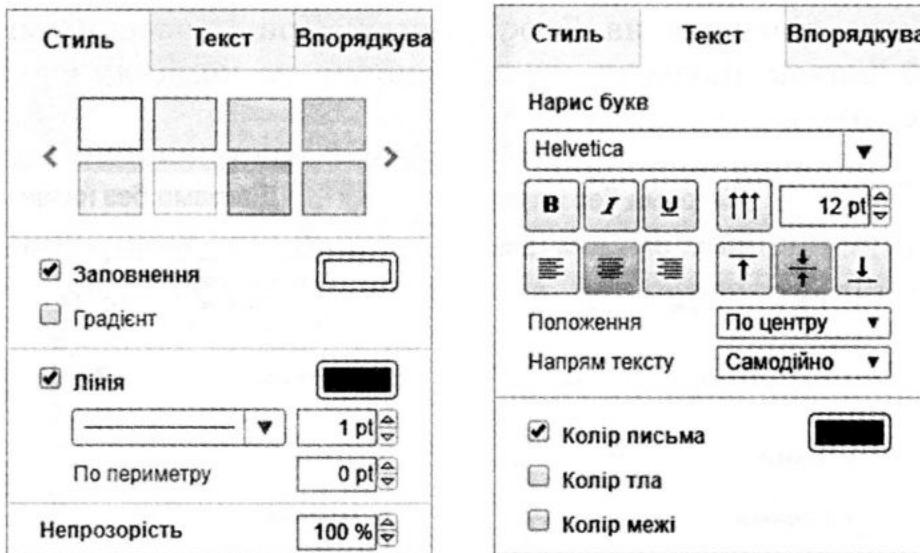


Рис. 3. Меню форматування елемента діаграми

5. Для полегшення роботи використовуйте команду клонування (перетягування вибраної фігури із натиснутими клавішами Ctrl і лівою кнопкою миші).
6. Створіть елементи третього рівня. Поміркуйте, як їх назвати та скільки їх, на вашу думку, має бути.
7. Інструментом Стрілка налаштуйте зв'язки між сторінками (з яких і на які сторінки здійснюватиметься перехід (рис. 4)).

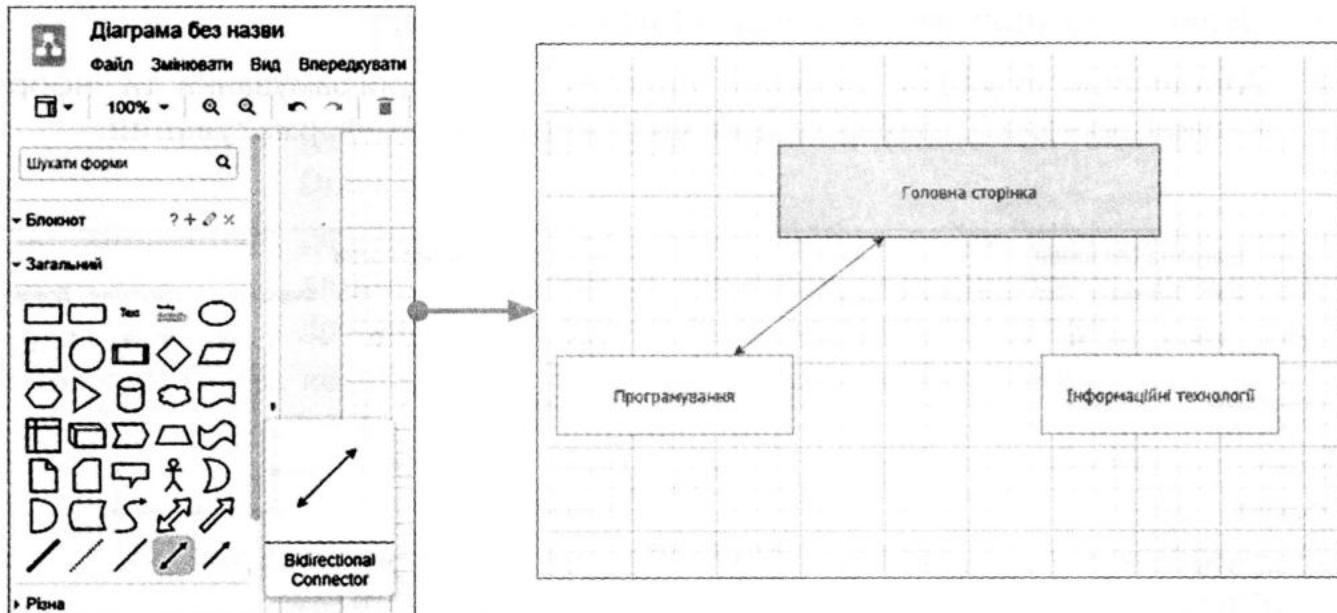


Рис. 4. Налаштування переходу з головної сторінки сайту на сторінку «Програмування»

8. У меню Стиль виберіть стиль форматування для зв'язків.

Збереження створеної діаграми

1. Збережіть розроблену структуру сайта на своєму Google Диску (File → → Save або сполучення клавіш Ctrl + S).
2. Перегляньте діаграму на Google Диску. При наведенні миші на колірковий значок діаграми він змінюється на команду Відкрити Google Диск (рис. 5).

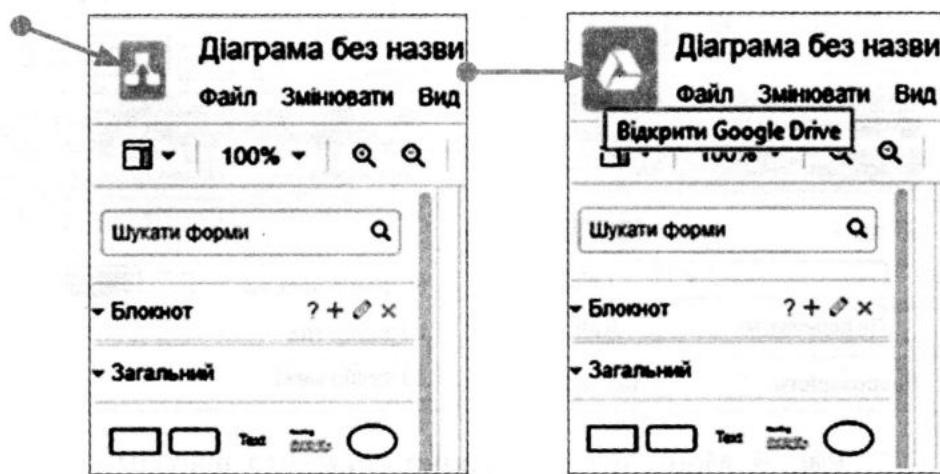


Рис. 5. Кнопка перемикання на Google Диску

3. За допомогою команди на Google Диску або засобами сервісу (File → → Share) отримайте посилання на розроблену вами структуру сайта.
4. Надішліть посилання вчителю й трьом однокласникам та/або однокласницям.

Зробіть висновок за результатами порівняння структури сайтів, створених вашими однокласниками.

1.4

Інструменти веброзробника

Є цілий клас завдань (створення сайта, перевірка його працездатності, супровід, оптимізація та просування), виконання яких неможливе без участі фахівців різної спеціалізації.

Який вигляд матиме сайт — цим питанням опікується вебдизайнер. Веброзробник отримує результат його роботи у вигляді макета. Його завдання — написати код, що відобразить браузер у саме такому вигляді, як запропонував вебдизайнер (рис. 1.27).

Усі сайти будуються з допомогою мови розмітки HTML та каскадних таблиць стилів CSS.

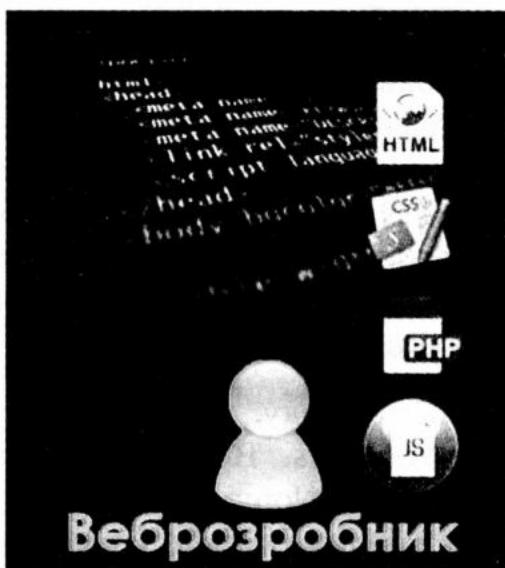


Рис. 1.27. Інструменти веброзробника

Редактор коду — це перший інструмент веброзробника. Сучасні редактори надають широкий асортимент інструментів, які полегшують і прискорюють процес розробки коду.

Розглянемо найбільш поширені нині редактори коду згідно зі статистикою сайта Stack Overflow (рис. 1.28).

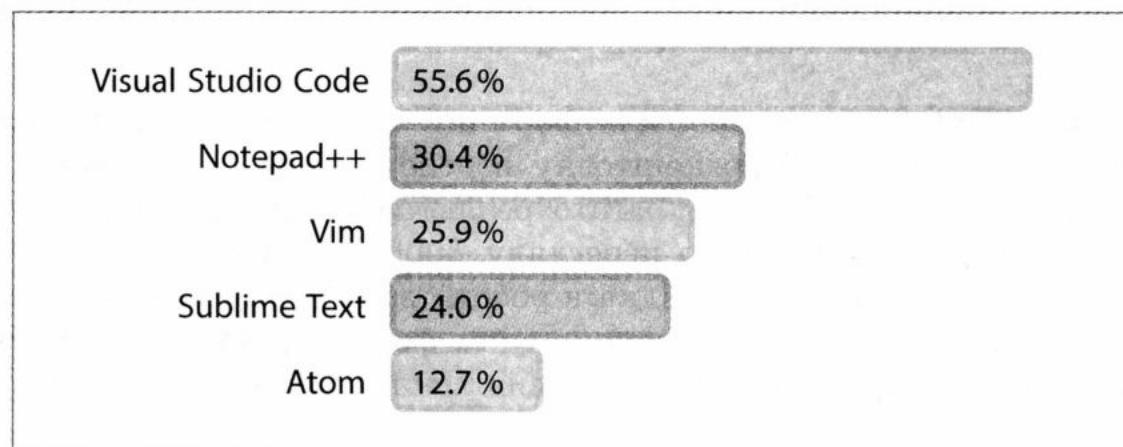


Рис. 1.28. Рейтинг популярності редакторів коду



Рис. 1.29. Сайт Stack Overflow

Сайт Stack Overflow (рис. 1.29) створено у 2008 році Джефом Етвудом і Джоелом Спольські як ресурс для програмістів, що намагаються розв'язати якусь проблему. Це сервіс питань і відповідей з програмування, який входить до сотні найбільш відвідуваних сайтів. Щомісяця — близько 50 млн користувачів, 72 % із яких є веброзробниками.

Нинішній майбутні фахівці обговорюють сотні мов програмування, фреймворків, платформ тощо. Причому якість відповідей на Stack Overflow тримається на рівні кращих академічних стандартів. Тому компанії, запрошуючи на роботу розробника ПЗ, розглядають активний профіль на Stack Overflow як важливу частину резюме.

- ◆ **Visual Studio Code** — це безкоштовний крос-платформний редактор коду, розроблений корпорацією Microsoft. Виходячи з опитування, проведеного Stack Overflow у 2020 році, Visual Studio Code є найпопулярнішим редактором коду, ним користуються понад 55 % розробників.

Програма має відкритий вихідний код. Вона оснащена доступним набором інструментів для редагування й налаштування, легко інтегрується з іншими сервісами, її властивості можна легко розширити.

- ◆ **NotePad++** — це розвинутий редактор коду, випущений у 2003 році і доступний лише на платформі Windows. Виходячи з опитування, проведеного Stack Overflow у 2020 році, NotePad++ займає другу позицію в рейтингу популярних редакторів коду. Ним користуються понад 30 % розробників.

Програма дуже швидка, підтримує зовнішні плагіни, включаючи макроси, її інтерфейс передбачає редагування в різних вкладках, є опція перетягування для початківців. NotePad++ підтримує повноекранний режим, робить автоматичні відступи та автодоповнення, має дуже продумане підсвічування синтаксису.

Програма дозволяє здійснювати пошук і заміну тексту, перевірку правопису з порівнянням файла, використовувати фолдинг (функціонал згортання).

- ◆ **Sublime Text 3** — це крос-платформний редактор коду, що має як преміум-версію, так і безкоштовну версію, яку можна завантажити на офіційному сайті. Існує багато редакторів коду, які підтримують чорне тло для розширеного перегляду. Sublime Text 3 — один із них.

Програма легка й дуже швидка в роботі, за замовчуванням надає автодоповнення, підсвічування синтаксису та фолдинг, має зручний інтерфейс для початківців, виявлення та виділення синтаксичних помилок, дозволяє одночасно редагувати багато рядків.

У Sublime Text 3 можна додатково налаштувати плагін Package Control, додавши таким чином інструменти налаштування, нові теми.



- ◆ **Atom** — це безкоштовний крос-платформний редактор із відкритим вихідним кодом. Це розробка GitHub. Інтерфейс Atom виглядає як клон редактора Sublime Text, проте працювати в ньому набагато комфортніше, оскільки він більш простий і зрозумілий.

Крім того, якщо розробник під час програмування стикається з труднощами, спільнота GitHub досить активно розв'язує цю проблему.

Програма за замовчуванням надає підсвічування синтаксису, доповнення й згортання коду, а також має вбудовану підтримку десятків мов програмування. Atom постачається із вбудованим менеджером пакетів, завдяки чому можна здійснювати пошук, установлювати або створювати власні пакети для розширення функціоналу редактора.

Подібно до Visual Studio Code, Atom оснащений потужним інструментом для парного програмування — Teletype. Це дає можливість кільком розробникам приєднуватися до ізольованої сесії й працювати спільно.

- ◆ **Vim** — це редактор (від англ. *vi improved* — покращений *vi*), який уперше був випущений наприкінці 1991 року Брамом Мооленааром. Справжню міць Vim демонструє під час роботи зі структурованими текстами, він є незамінним для програмістів і верстальників, його використовує кожен четвертий веброзробник (вебдевелопер).

Vim є клоном текстового редактора vi для Unix, який написав Біл Джой у 1976 році. Тоді ж був придуманий його незвичайний інтерфейс, заснований на поділі режимів роботи: стандартний, режим вставки, режим командного рядка.

- ◆ **Brackets** — це редактор коду, створений з нуля спеціально для вебдизайнерів і фронтенд-розробників, які працюють переважно з JavaScript, HTML і CSS.

Програма поставляється з основними стандартними властивостями, включаючи автодоповнення, підсвічування синтаксису для багатьох мов програмування, підтримку швидкого редагування й різноманітних пре-процесорів.

У Brackets є властивість extract, що дозволяє підтягувати кольори, розміри, градієнти, шрифти та інші важливі дані з PSD-файла в CSS-файл, готовий до використання. Brackets дуже популярний серед українських веброзробників.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. У чому полягає різниця між роботою вебдизайнера і веброзробника?
2. Якими інструментами користується в роботі веброзробник?
3. Назвіть найпопулярніший, на вашу думку, редактор коду.
4. Яке призначення редактора коду?
5. Що об'єднує всі редактори коду?

6. На рис. 1.30 наведено логотипи інструментів веброзробника. Знайдіть в інтернеті відомості про ці інструменти. Створіть їх класифікацію.



Рис. 1.30. Логотипи до завдання 6



Дивіться презентацію «Інструменти веброзробника».



Розділ 2

ПРОЕКТУВАННЯ ТА ВЕРСТКА ВЕБСТОРІНОК

2.1 Мова гіпертекстової розмітки

Практична робота №2

2.2 Каскадні таблиці стилів

Практична робота №3

2.3 Проектування та верстка вебсторінок

Практична робота №4

2.4 Адаптивна верстка

Практична робота №5

2.5 Кросбраузерність

Практична робота №6



2.1

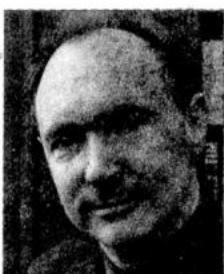
Мова гіпертекстової розмітки

Для розуміння процесу створення сайта необхідне опанування мови розмітки гіпертексту.



HTML (англ. *HyperText Markup Language*) — стандартна мова розмітки документів у Всесвітній павутині, яка обробляється спеціальними програмами (браузерами) і відображається у вигляді документа у зручній для людини формі.

Слід зазначити, що HTML не є мовою програмування, вона призначена лише для розмітки сторінки, надання певного вигляду її складовим.

**ЦІКАВІ ФАКТИ**

Мову HTML розробив британський учений Тім Бернерс-Лі приблизно в 1989–1991 роках під час роботи в Європейській лабораторії з ядерних досліджень ЦЕРН у Женеві (Швейцарія).



Гіпертекст — електронний документ, який містить зв'язки з іншими електронними документами. Такі зв'язки називаються гіперпосиланнями.

HTML створювалася як мова для обміну науковою та технічною документацією, як один із компонентів технології розподіленої гіпертекстової системи World Wide Web (яку ми звичайно називати Всесвітньою павутиною).

Ідея така: користувач має можливість переглядати документи (сторінки тексту) у найзручнішому для себе порядку, а не послідовно, як це узвичаєно під час читання книжок. Це досягається створенням спеціального механізму зв'язування різних сторінок тексту за допомогою гіпертекстових посилань.

Будь-який документ мовою HTML є набором елементів, водночас початок і кінець кожного елемента позначається спеціальними позначками — тегами.

Теги — команди мови HTML. HTML-теги — це ключові слова або символи, які записуються в кутових дужках.

Теги бувають двох видів: парні й непарні (їх ще називають поодинокими). Парні теги складаються з відкриваючого і закриваючого тегів. Теги нечутливі до регистра (тобто регистронезалежні), тому можуть бути написані як великими, так і малими літерами.



Однак існують правила запису тегів (більш детально — див. § 4.6), за якими теги пишуться виключно малими літерами.



Теги визначають, де починається й де закінчується HTML-елемент.

Текстові документи, що містять розмітку мовою HTML (такі документи зазвичай мають розширення .html або .htm), опрацьовуються спеціальними застосунками, які відображають документ у його відформатованому вигляді. Такі застосунки називаються браузерами.

Пригадаємо, що *браузером* (або *вебпереглядачем*) називають програмне забезпечення для комп’ютера або іншого електронного пристрою, під’єднаного до інтернету, який дає змогу користувачеві взаємодіяти з текстом, малюнками або іншою інформацією на гіпертекстовій вебсторінці.

Згідно з даними щорічних досліджень інформаційної агенції Hootsuite та соціального проекту We are social (рис. 2.1.), у 2020 році безперечним лідером серед браузерів залишається Google Chrome, який забезпечує 63 % всього глобального трафіку. Друге місце посідає Safari, третю позицію утримує Mozilla Firefox, активно нарощує свій трафік Samsung Internet. Opera та Internet Explorer (Microsoft Edge) посідають відповідно шосте та сьоме місця, втрачаючи своїх прихильників (більш детально про браузери — див. у § 2.5).



Рис. 2.1. Розподіл вебтрафіку між браузерами

За допомогою тегів браузер розпізнає структуру документа. Отримавши цю інформацію, браузер використовує вбудовані в нього за замовчуванням правила про те, як відображати контент сторінки.



Контент (англ. *content* — вміст) — це інформаційне наповнення сайта.

Контент може містити:

- текст;
- статичну графіку (§ 3.1);
- анімаційні елементи (§ 3.2);
- мультимедійні елементи: відео, аудіо (§ 3.3).

Без використання HTML-тегів браузер виведе невідформатований текст, без відступів, заголовків, абзаців тощо. Розглянемо структуру стандартної HTML-сторінки.

HTML-документ завжди починається з елемента `<!DOCTYPE>`, призначеного для того, щоб вказати браузеру, як слід інтерпретувати поточну вебсторінку. Зверніть увагу на те, що елемент `<!DOCTYPE>` пишеться великими літерами. Справа в тому, що це не є тег HTML, це інструкція браузеру про те, якою версією HTML написана сторінка.

`<!DOCTYPE html>` означає, що використовується HTML5.

Сторінка завжди починається з відкривального тега `<html>` та закінчується закривальним тегом `</html>` і складається з двох обов'язкових блоків — голови (`head`) і тіла (`body`), які записуються послідовно.

У блоці `<head></head>` зберігається службова інформація, призначена допомогти браузеру в роботі з даними. Тут розташовані мета-теги, які використовуються для зберігання інформації, призначеної для пошукових систем, а саме: опис сайту, ключові слова тощо. Одним із таких мета-тегів є визначення кодування HTML-сторінки. Це необхідно робити для того, щоб веббраузер міг правильно відображати текст, якщо кодування обрано неправильно — замість тексту будуть виведені ієрогліфи. Найбільш поширеним наразі є кодування UTF-8, тому в блоці `<head></head>` ми маємо написати:

| `<meta charset="utf-8">`

Інформація є зазвичай невидимою для пересічного користувача, крім тега `<title>`, в якому відображається назва сторінки сайта.

Увесь контент, який відображається на сторінці, розміщується між відкривальним і закривальним тегами `<body>`.

Зупинимось на тегах для роботи з текстом, який відображається у вигляді заголовків, абзаців і списків (рис. 2.2).

У HTML існують шість рівнів заголовків. Найвищим є заголовок первого рівня, найменшим — шостого. Позначаються вони відповідно `<h1>...<h6>`. Заголовки мають атрибут `align`, який визначає тип вирівнювання на сторінці та може набувати одне з чотирьох значень:

Назва	Значення
left	За лівим краєм (за замовчуванням)
right	За правим краєм
center	По центру
justify	За ширину (за лівим і правим краями)

Атрибути тега мають вигляд: назва атрибуту, потім знак рівності та значення атрибуту, записане в подвійних або одинарних лапках, наприклад:

| <h1 align='center'>

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <h1> Заголовок 1 рівня </h1>
    <h2 align='right'> Заголовок 2 рівня </h2>
    <h3 align='center'> Заголовок 3 рівня </h3>
    <h4 align='left'> Заголовок 4 рівня </h4>
    <h5 align='right'> Заголовок 5 рівня </h5>
    <h6 align='center'> Заголовок 6 рівня </h6>
</body>
</html>
```

Код HTML

Заголовок 1 рівня

Заголовок 2 рівня

Заголовок 3 рівня

Заголовок 4 рівня

Заголовок 5 рівня

Заголовок 6 рівня

Відображення
браузером

Рис. 2.2. Приклад запису заголовків та їх відображення браузером

Абзаци виділяються парним тегом `<p> </p>` (від англ. *paragraph*). Як і теги заголовків, тег `paragraph` має атрибут `align`.

Списки можуть бути впорядковані, тобто нумерованими, невпорядкованими (маркованими) та багаторівневими.

Кожен елемент списку виокремлюється парним тегом ``. Нумерований список позначається парним тегом `` (скор. *ordered list* — впорядкований список), маркований список — парним тегом `` (скор. *unordered list* — невпорядкований список) (рис. 2.3).

У HTML існує спеціальний тег-контейнер для реалізації навігації по сайту — `<nav>`. Це парний тег, який містить навігаційні посилання (рис. 2.4).

Для посилання на іншу сторінку використовують парний тег `<a>` (скор. від англ. *anchor* — якір). Тег має обов'язковий атрибут `href`, значенням якого є назва сторінки з розширенням `.html`.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <ol>
        <li> Елемент 1 </li>
        <li> Елемент 2 </li>
        <li> Елемент 3 </li>
        <li> Елемент 4 </li>
        <li> Елемент 5 </li>
    </ol>
</body>
</html>
```

Код HTML

- 1. Елемент 1
- 2. Елемент 2
- 3. Елемент 3
- 4. Елемент 4
- 5. Елемент 5

Відображення браузером

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <ul>
        <li> Елемент списку 1 </li>
        <li> Елемент списку 2 </li>
        <li> Елемент списку 3 </li>
        <li> Елемент списку 4 </li>
        <li> Елемент списку 5 </li>
    </ul>
</body>
</html>
```

Код HTML

- Елемент списку 1
- Елемент списку 2
- Елемент списку 3
- Елемент списку 4
- Елемент списку 5

Відображення браузером

Рис. 2.3. Приклади кодів нумерованого та маркованого списків та їх відображення браузером

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <nav>
        <a href="#">Головна</a>
        <a href="info.html">Інформація</a>
        <a href="gallery.html">Галерея</a>
        <a href="form.html">Зв'язатись з нами</a>
    </nav>
</body>
</html>
```

Код HTML

<u>Головна</u>	<u>Інформація</u>	<u>Галерея</u>	<u>Зв'язатись з нами</u>
----------------	-------------------	----------------	--------------------------

Відображення браузером

Рис. 2.4. Приклад запису навігаційного контейнера та його відображення браузером



Між відкривальним і закривальним тегами розміщується посилання на ресурс. Натиснувши на нього, користувач може переходити до потрібної сторінки.

З одного боку, HTML5 — це нова версія мови HTML, із новими елементами, атрибутами та новою поведінкою, з іншого — сукупність технологій, котра дозволяє створювати різноманітні сайти й вебастосунки.

Мова HTML постійно розвивається. Якщо ранні версії HTML поєднували функції розмітки та форматування контенту, тобто зміст (семантичний) та зовнішній вигляд цього змісту (презентаційний), то з появою багатосторінкових сайтів переважну частину функцій форматування було покладено на каскадні таблиці стилів (див. § 2.2).

У 1994 році засновано Консорціум Всесвітньої павутини — головну міжнародну організацію, що розробляє та впроваджує технологічні стандарти для Всесвітньої павутини.



Наразі Консорціум просуває версію HTML5, це остання версія стандарту HTML. HTML5 остаточно відмовляється від використання презентаційних тегів, натомість з'являється низка нових елементів і атрибутив, які відображають типову архітектуру сучасних вебсторінок. Приклад використання тегів HTML5 наведено на рис. 2.5.

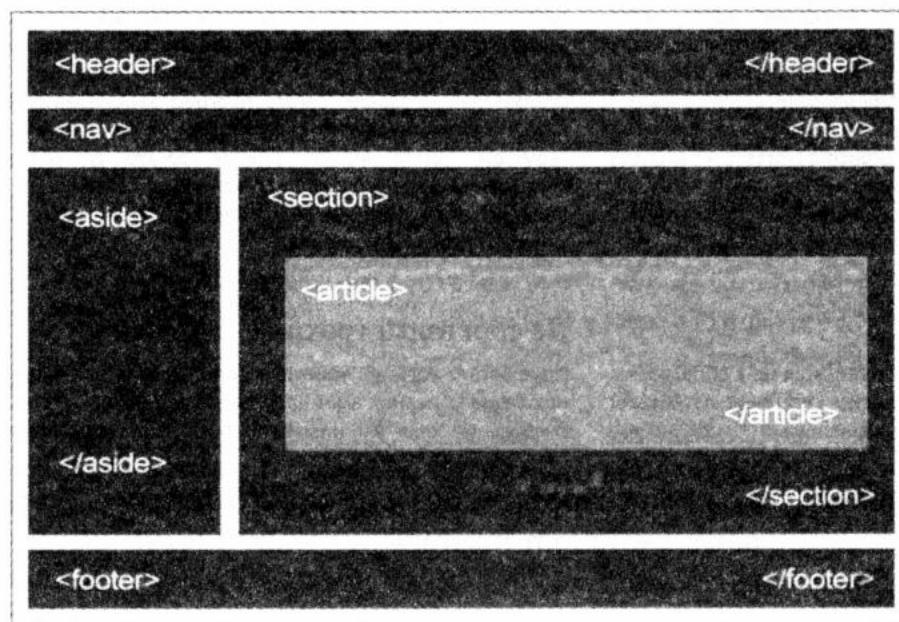


Рис. 2.5. Приклад розмітки сторінки з використанням тегів HTML5

Під час розроблення макета сайта завжди постає проблема заповнення його текстом. Зазвичай для надання макету закінченого вигляду дизайнери застосовують так звану «рибу» — великий текстовий масив беззмістовних слів із потрібною кількістю символів, абзаців і параграфів, який називається *Lorem Ipsum*.

Основна функція *Lorem Ipsum* полягає в зосередженні уваги на дизайні, а не на читанні вмісту. Ще одна причина, через яку використовують текст, — це заповнення сторінки для досягнення реального розподілу букв і пропусків у тексті, яке неможливе у випадку простого дублювання вислову «Тут написано ваш текст... Тут написано ваш текст...».

У більшість текстових і HTML-редакторів Lorem Ipsum входить як текст за замовчуванням. Існують і сайти, на яких можна згенерувати потрібну кількість абзаців. Одним із найвідоміших і найпопулярніших сайтів-генераторів є Llpsum generator (<https://uk.llpsum.com/>) (рис. 2.6).

Розширені мови: Azerbaijani, Shqip, العربية, Български, Català, 中文简体, Hrvatski, Česky, Dansk, Nederlands, English, Estonian, Filipino, Suomi, Français, Հայերեն, Deutsch, Ελληνικά, עברית, മലയാളം, Magyar, Indonesia, Italiano, Latvian, Lietuviškai, македонски, Melayu, Norsk, Polski, Português, Româna, Русский, Српски, Slovenčina, Slovenščina, Español, Svenska, ไทย, Türkçe, Українська, Tiếng Việt.

Lorem Ipsum

"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur adipisci velit..."
"Немає нікого, хто любив би самий біль, хто б шукав його чи хотів би його зазнавати тільки через те, що він - біль..."

Що таке Lorem Ipsum?

Lorem Ipsum - це текст-"риба", що використовується в друкарстві та дизайні. Lorem Ipsum є, фактично, стандартною "рибою" аж з XVI сторіччя, коли невідомий друкар взяв шрифтову гранку та склав на ній підбірку зразків шрифтів. "Риба" не тільки успішно пережила п'ять століть, але й прижилася в електронному верстуванні, залишаючись по суті незмінною. Вона популяризувалась в 60-их роках минулого сторіччя завдяки виданню зразків шрифтів Letraset, які містили уривки з Lorem Ipsum, і вдруге - нещодавно завдяки програмам комп'ютерного верстування на кшталт Aldus Pagemaker, які використовували різні версії Lorem Ipsum.

Чому ми ним користуємося?

Вже давно відомо, що читабельний зміст буде заважати зосередитись людині, яка оцінює композицію сторінки. Сенс використання Lorem Ipsum полягає в тому, що цей текст має більш-менш нормальнє розподілення літер на відміну від, наприклад, "Тут іде текст. Тут іде текст." Це робить текст схожим на оповідний. Багато програм верстування та веб-дизайну використовують Lorem Ipsum як зразок, і пошук за терміном "lorem ipsum" відкриває багато веб-сайтів, які знаходяться ще в зародковому стані. Різні версії Lorem Ipsum з'явилися за минулі роки, деякі випадково, деякі було створено зумисно (зокрема, жартівливі).

Звідки він походить?

На відміну від пошиrenoї думки Lorem Ipsum не є випадковим набором літер. Він походить з уривку класичної латинської літератури 45 року до н.е., тобто має більш як 2000-річну

Де собі взяти трохи?

Існує багато варіацій уривків з Lorem Ipsum, але більшість з них зазнала певних змін на кшталт жартівливих вставок або змішування слів, які навіть не виглядають правдоподібно.

Рис. 2.6. Головна сторінка генератора беззмістового тексту латиною Lorem Ipsum

Для більш глибокого вивчення технологій веброзробки радимо скористатися курсом «Основи веброзробки» освітньої платформи Prometheus. Викладач курсу — Світлана Шабаранська, Web-UI-девелопер у компанії N-iX.

Фонд BrainBasket (некомерційна організація, мета якої — сприяти розвитку освіти українців в сфері ІТ, заснована в 2014 році) та студія онлайн-освіти EdEra розробили онлайн-курс з основ веброзробки. Ведучим курсу є Сергій Денисов, викладач Київського національного університету ім. Тараса Шевченка.

Стане в пригоді вебсайт для вивчення вебтехнологій в інтернеті (<https://www.w3schools.com/>), який містить навчальні посібники для вивчення HTML, CSS, JavaScript та ін.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

- Що таке контент; HTML; гіпертекст?
- Як називають команди мови розмітки?
- Створіть класифікацію тегів HTML.
- Які невід'ємні компоненти структури сторінки?
- Порівняйте HTML4 і HTML5. Зробіть презентацію за результатами.
- Знайдіть в інтернеті відомості про Тіма Бернерс-Лі, зробіть коротке повідомлення.



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

- Увійдіть у редактор коду на комп'ютері. Створіть файл index.html.
- Наберіть послідовність команд, як наведено на рис. 2.7.
- Як текст використовуйте абзаци, згенеровані Lorem Ipsum.
- Збережіть файл.
- Перегляньте результат браузером.
- Відкрийте файл у редакторі коду. Приберіть список, залишивши посилання з тегами <a>. Збережіть модифікований файл.
- Перегляньте результат браузером.
- Проаналізуйте, як браузер відобразив змінений файл.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Моя перша сторінка</title>
    <meta charset="UTF-8">
  </head>
  <body>
    <header>
      <h1 align='center'>Головна сторінка</h1>
      <h2 align='right'>Слоган</h2>
    </header>
    <nav>
      <ul>
        <li><a href="#">Головна</a></li>
        <li><a href="#">Інформація</a></li>
        <li><a href="#">Контакти</a></li>
      </ul>
    </nav>
```

```
<main>
  <p>! ДОДАТИ ТЕКСТ LOREM</p>
  <p align='right'>! ДОДАТИ ТЕКСТ</p>
  <p align='center'>! ДОДАТИ ТЕКСТ</p>
  <p align='left'>! ДОДАТИ ТЕКСТ</p>
  <p align='justify'>! ДОДАТИ ТЕКСТ
  </p>
</main>
<footer>
  <p>Copyright Example 2020</p>
</footer>
</body>
</html>
```

Рис. 2.7. Команди до завдання 2



Дивіться презентацію «Мова гіпертекстової розмітки. Гіпертекстовий документ та його елементи».



Практична робота № 2

ТЕМА. Текстові елементи вебсторінки.

Гіперпосилання та списки на вебсторінках

ЗАВДАННЯ: створити сторінки універсального сайта, використовуючи для наповнення контенту генератор беззмістового тексту.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

1. Створіть теку site.
2. Відкрийте редактор коду. Створіть новий файл і збережіть його з іменем index.html у теці site.
3. Уведіть базову структуру HTML-сторінки (рис. 1).

```
<!DOCTYPE html>
<html lang="uk">
  <head>
    <title></title>
    <meta charset="UTF-8">
  </head>
  <body>
  </body>
```

Рис. 1. Базова структура HTML-сторінки

Примітка. 1-й рядок — це інструкція для браузера, з якою версією мови розмітки він працюватиме (ми орієнтуємося на HTML5), 2-й рядок визначає мову сайта (uk — українська), нарешті, 5-й рядок — кодування HTML-сторінки (його необхідно вказувати для правильного відображення тексту браузером). Найбільш поширенним кодуванням є UTF-8.

4. Уведіть назив сайту між відкривальним і закривальним тегами <title>, наприклад, My site. Установіть покажчик миші після відкривального тегу <body>.
5. Почніть заповнення контенту з тегу <header>. Уведіть наведений код (рис. 2).

```
<header>
  <h1>Main Header – Name of site</h1>
  <h2>Sub header – Topic of site</h2>
</header>
```

Рис. 2. Код



6. Створіть меню за допомогою маркованого списку `` (рис. 3):

```
<nav>
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="info.html">About</a></li>
    <li><a href="gallery.html">Gallery</a></li>
    <li><a href="form.html">Contacts</a></li>
  </ul>
</nav>
```

Рис. 3. Маркований список

7. Відкрийте генератор беззмістового тексту *Lorem Ipsum*. Згенеруйте два-три абзаци й додайте у створений файл, відокремте кожен абзац відкривальним і закривальним тегами `<p>`.
8. Додайте футер із наведеним кодом (рис. 4).

```
<footer>
  <p>Copyright 2020</p>
</footer>
```

Рис. 4. Код із футера

9. Збережіть файл.
10. Збережіть створений файл з іменем `info.html` у текці `site`. Додайте на нього 5–7 абзаців, згенерованих генератором беззмістового тексту.
11. Збережіть файл `index.html` з іменем `form.html` у текці `site`. Залиште один абзац згенерованого тексту.
12. Закрийте редактор коду. Увійдіть у теку `site`. Коли ви натиснете на `index.html`, сторінка відкриється браузером, вибраним на вашому комп’ютері за замовчуванням.
13. Перейдіть за допомогою команд меню на сторінки `About`, `Contacts`, `Home`.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Якого кольору тло й текст сайту?
2. Чи відрізняється розмір шрифту заголовків `h1`, `h2` і розмір шрифту абзаців?
3. Що відбувається при спробі перейти на сторінку `Gallery`? Чи можете ви пояснити причину такої реакції?

Зробіть висновок за результатами порівняння створеного сайта із будь-яким, відкритим у мережі.

2.2 Каскадні таблиці стилів

Поява величезної кількості сайтів у 1990-х роках спонукала до розробки нових тегів мови HTML, які відповідали за дизайн сайтів. Проте при зміні дизайну сайта в кожну його сторінку потрібно було вносити зміни, що значно ускладнювало роботу.

З розвитком Всесвітньої павутини з'явилося безліч ідей оформлення вебсторінки. Проте, на думку Тіма Бернерса-Лі, це не було надважливим завданням. Він розглядав веб насамперед як цифровий каталог наукових робіт (нагадаємо, для чого створювалась мова HTML). Замість одного стандарту в той час існувало кілька конкуруючих мов для макетування вебсторінок, створених різними розробниками, серед яких були Пей-Ян Вей, Марк Андріссен і Хокон Віум Лі.

Пей-Ян Вей створив графічний браузер ViolaWWW (рис. 2.8) в 1991 році. Він вмонтував власну мову стилів безпосередньо в браузер. Один із головних розробників браузера Netscape Navigator Марк Андріссен пішов іншим шляхом. Замість того, щоб створювати окрему мову, призначенну для стилізації вебсторінок, він розширив HTML, додавши в нього нестандартні теги, які можна було використовувати для цілей вебдизайну.

ЦІКАВІ ФАКТИ

Насправді існувало безліч альтернативних ідей: на кшталт RRP — мови стилів, яка характеризувалась стисливістю, або PSL96 — мови, яка підтримувала функції умовні вирази. Більш детально про перші спроби стилізації вебсторінок — в публікації Зака Блума, відомого дослідника історії інтернету (<https://eager.io/blog/the-languages-which-almost-were-css/>).

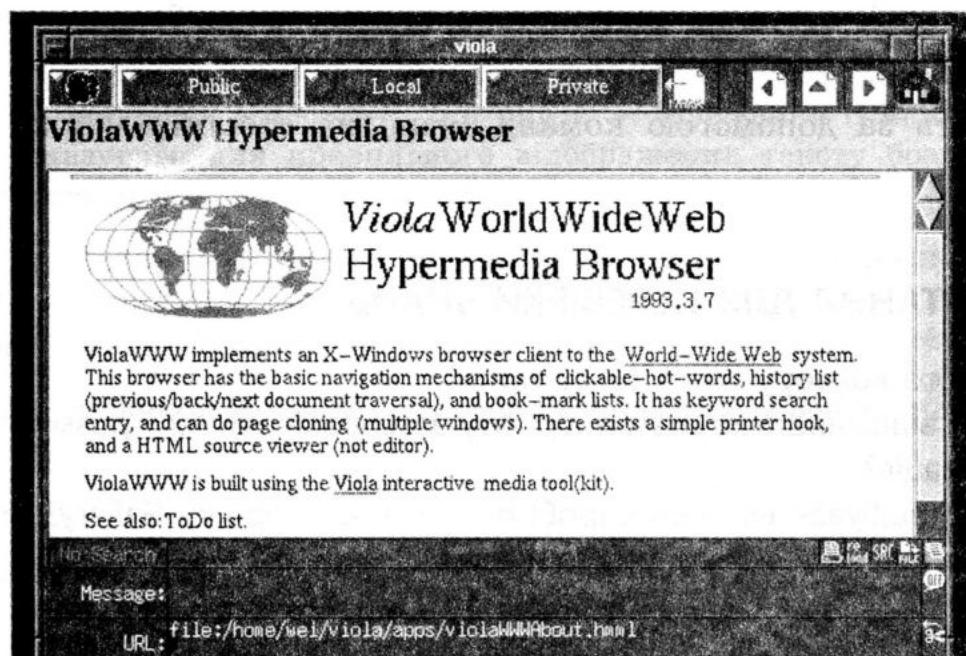


Рис. 2.8. Графічний браузер ViolaWWW



ЦІКАВІ ФАКТИ

У 1994 році норвезький учений Хокон Віум Лі запропонував концепцію відділення дизайну сайта від його вмісту за допомогою каскадних таблиць стилів.

Каскадні таблиці стилів (англ. *Cascading Style Sheets*, або скорочено CSS) — спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовою розмітки даних. Основна ідея CSS полягає в тому, щоб відокремити дизайн документа від його вмісту. CSS відповідає за оформлення і зовнішній вигляд HTML-коду, тоді як HTML — за зміст і логічну структуру документа.

Конструкція CSS, яка відповідає за зовнішній вигляд певного елемента HTML, називається **CSS-правилом**. Усі CSS-правила складаються із селектора та блоку оголошень.

Блок оголошень містить одне або кілька оголошень, розташованих у фігурних дужках. Усередині блоку оголошень знаходяться пари CSS-властивість — значення, розділені крапкою з комою (рис. 2.9).

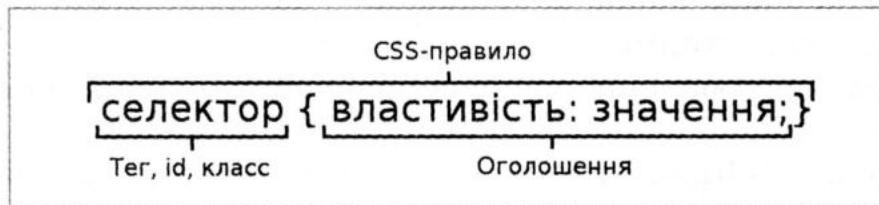


Рис. 2.9. Правило оголошення CSS

Правила форматування CSS

- ◆ Кожна властивість — в окремому рядку.
- ◆ Кожен селектор — в окремому рядку.

Порядок опису стилів

- ◆ Позиціонування (position, left/right/top/bottom, float, clear, z-index)
- ◆ Дісплей і блочне моделювання (display, width, height, margin, padding, ...)
- ◆ Колір (background)
- ◆ Текст (list-style-type, overflow, color font, ...)
- ◆ Інші

Кожне правило починається із селектора (покажчика), що вказує на ті HTML-елементи, до яких застосовується CSS-правило. Саме в блоці оголошень установлюються правила відображення вибраних елементів, визначаються їхні властивості — розмір, колір, відступи, поля, розташування на екрані (позиціонування) тощо. Селектор дозволяє звернутись до одного або кількох HTML-елементів.

Якщо необхідно визначити стиль таким чином, щоб один і той самий елемент у різних випадках відображався по-різному, то на допомогу

приходять класи. Клас описується у вигляді `.ім'я_класу {властивості}`. Для присвоєння класу заданому тегу використовується властивість `class = "ім'я_класу"`.

Ідентифікатори (селектори `id`) дуже подібні до класу, крім одного — ідентифікатор може мати одне-єдине унікальне ім'я в усьому документі. У файлі CSS ім'я вказується зі знаком решітки на початку, а до потрібного елемента додається атрибут `id = " "`.

Термін *cascading* (каскадні) у назві CSS вказує на можливість злиття різних таблиць стилів для створення єдиного визначення стилю окремого елемента (тегу) чи всього документа.

Каскадність CSS — це механізм, завдяки якому до елемента HTML-документа може застосовуватися більш ніж одне правило CSS.

Правила можуть виходити з різних джерел: із зовнішньої та внутрішньої таблиці стилів, від механізму наслідування, від батьківських елементів, від класів і `id`, від селектора тегу, від атрибута `style` тощо. Оскільки в цих випадках часто відбувається конфлікт стилів, була створена система пріоритетів: у кінцевому підсумку застосовується той стиль, який виходить від джерела з більш високим пріоритетом.

Таким чином, каскадність у CSS — це здатність стилювих правил накладатися одне на одне, перезаписуватися і змішуватися. Підсумковий стиль елемента, який видно в браузері, — це комбінація кількох послідовно застосованих стилів.

Розглянемо способи підключення CSS-коду до HTML-документа.

- Застосування **inline-стилів** (стилі, які підставляються безпосередньо в рядок).

Додавання CSS-правила в HTML-тег за допомогою атрибута `style`. Усередині атрибута `style` можна написати кілька CSS-оголошень, розділених крапкою з комою, фігурні дужки не використовуються. **Inline-стилі** змішують уміст документа і його дизайн, тому їх краще використовувати як виключення, у випадку, коли елемент зустрічається лише один раз у документі або на сайті, але вимагає особливого оформлення.

Розглянемо приклад 1.

ПРИКЛАД 1

```
<body>
    <header style="background-color: grey;
        width: 900px;
        height: 100px;
        text-align: center;
        padding-top: 30px;
        margin: 10px auto;">
        Заголовок сайта
    </header>
</body>
```



◆ **Застосування таблиць стилів документа (document style sheets).**

Називаються так тому, що розташовуються безпосередньо в HTML-документі й застосовуються лише до нього. Іноді їх називають *embedded style sheet* (убудований стиль). CSS-код зберігається в HTML-документі у тегу `<style>`, що теж розміщується в `<head>`. Зазвичай цей варіант використовується, коли існує лише одна проста HTML-сторінка й немає сенсу створювати додатковий файл. Розглянемо приклад 2.

ПРИКЛАД 2

```

<head>
<style type="text/css">
    header {
        background-color: grey;
        width: 900px;
        height: 100px;
        text-align: center;
        padding-top: 30px;
        margin: 10px auto;
    }
</style>
<body>
    <header>
        Заголовок сайта
    </header>
</body>

```

◆ **Застосування зовнішніх, або зв'язаних, стилів (external style sheets).**

Це найбільш поширений варіант. Він полягає у винесенні CSS-коду в окремий файл із розширенням .css та підключенням за допомогою тегу `<link>`, який знаходиться виключно всередині елемента `<head>`. Зустрівши в HTML-документі цей тег, браузер завантажить із сайту CSS-файл і застосує до документа стилі, що містяться в ньому. Розглянемо приклад 3.

ПРИКЛАД 3

```

<head>
<link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
    <header>
        Заголовок сайта
    </header>
</body>

```

```

header {
    background-color: grey;
    width: 900px;
    height: 100px;
    text-align: center;
    padding-top: 30px;
    margin: 10px auto;
}

```

Існує ще один спосіб підключення таблиць — за допомогою директиви `@import`. Цей спосіб дозволяє об'єднувати кілька таблиць стилів в одну.

Розглянемо приклад 4.

ПРИКЛАД 4

```
<head>
<style type="text/css">
    @import url("css/style.css")
</style>
<body>
    <header>
        Заголовок сайта
    </header>
</body>
```

Свого часу цей спосіб був досить популярний, проте наразі він втрачає свої позиції через те, що сторінка не завантажиться, доки браузер не завантажить файл CSS повністю. Це негативно позначається на швидкості завантаження сайта й, відповідно, гальмує роботу.

Розглянемо пріоритетність стилів.

- Якщо в таблиці є однакові селектори, то виконуватиметься той, який записаний останнім. Наприклад:

| p{color: grey;} p{color: pink;}

— колір абзацу буде рожевим.

- Якщо для одного елемента задано стиль і в зовнішній, і у внутрішній таблицях, то пріоритет віддається стилю в тій таблиці, яка знаходиться нижче в коді. Наприклад, спочатку в `<head>` підключили зовнішню таблицю, а потім за допомогою тегу `<style>` додали внутрішню таблицю. Браузер відобразить стилі внутрішньої таблиці.

- Пріоритетність за спаданням:

- ◆ inline-стилі,
- ◆ ідентифікатори,
- ◆ класи,
- ◆ теги.

Ознайомимося зі стислим описом модулів CSS.

Назва модуля	Опис
CSS-позиціонування	Дає змогу вказати, де з'явиться блок елемента, а вільне переміщення (floating) переміщує елементи до лівого або правого краю блока-контейнера, дозволяючи решті вмісту «обрікати» його



Закінчення таблиці

Назва модуля	Опис
CSS-текст	Керує перетворенням вихідного тексту у форматований і перенесенням рядків
CSS-шрифт	Вибір шрифту (font-family), його насиченість (font-weight), ширина (font-stretch), накреслення (font-style), розмір (font-size)
CSS-посилання	Відповідають за зовнішній вигляд гіпертекстових посилань HTML-документа. Оскільки посилання є основним способом навігації сайтом, то застосування CSS-стилів для оформлення покращить їх візуальне сприйняття
CSS-списки	Відповідають за оформлення списків. Зазвичай списки використовуються під час створення навігаційних панелей сайта. За допомогою стандартних CSS-властивостей можна змінити зовнішній вигляд маркера списку, додати зображення для маркера, а також змінити місце розташування маркера
CSS-тло	Додають тло для будь-якого HTML-елемента. Кожен елемент має шар тла, який може бути прозорим (за замовчуванням), мати кольорове заповнення та зображення
CSS-колір	Описує значення, які дозволяють визначати колір та непрозорість HTML-елементів, а також значення властивості color

Одиниці вимірювання в CSS

- піксель (px) — абсолютна одиниця вимірювання, оскільки не змінюється залежно від інших налаштувань;
- em — масштабована одиниця, яка вказує відношення до розміру шрифту (font-size) поточного елемента (якщо font-size у документі 12pt, тоді 1em = 12pt, 2em = 24pt і т. д.);
- vw і vh — дозволяють вказувати розміри відносно вікна користувача (1vw — це 0,01 ширини вікна, а 1vh — 0,01 висоти).



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке каскадні таблиці стилів? Назвіть причини їх появи.
2. Опишіть синтаксис CSS-правила.
3. Що таке селектор? Які бувають селектори?
4. Наведіть пріоритети виконання таблиць стилів.
5. Опишіть способи підключення стилів. Проаналізуйте, коли і який спосіб краще використовувати.
6. Виберіть із наведеної раніше таблиці CSS-модуль. Розробіть презентацію з докладним розбором та прикладами використання правил CSS.



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

1. Увійдіть у редактор коду, встановлений на вашому комп'ютері, і створіть файл style.css.
 2. Наберіть послідовність команд, наведену на рис. 2.10.
 3. Збережіть файл у тій самій текці, де ви зберегли файл index.html, створений на попередньому уроці.
 4. Відкрийте файл index.html у редакторі коду. Додайте у блок <head> таку команду:
- ```
<link rel="stylesheet" type="text/css" href="style.css">
```
5. Збережіть модифікований файл.
  6. Перегляньте результат браузером.
  7. Проаналізуйте, як браузер відобразив змінений файл.

```

body{
 font-family: "Segoe UI";
 background-color: #153e5c;
}
h1,h2{
 text-align: center;
 color: #cdd4ca;
}
a{
 color: #cdd4ca;
 text-decoration: none;
}
a:hover{
 color: #f2be54;
}
li{
 list-style: none;
 display: inline-block;
}
p{
 font-size: 150%;
 color: white;
}

```

Рис. 2.10. Команди до завдання 2



Дивіться презентацію «Каскадні таблиці стилів».





## Практична робота № 3

**ТЕМА.** Стильове оформлення сторінок із використанням CSS

**ЗАВДАННЯ:** відформатувати сторінки сайта, створеного в практичній роботі № 2, використовуючи каскадні таблиці стилів.

**ОБЛАДНАННЯ:** будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

### Хід роботи

*Під час роботи з комп'ютером дотримуйтесь правил безпеки.*

На початку роботи відкрийте вашу головну сторінку браузером, зробіть її скріншот.

1. У теці site створіть теку CSS.
2. Відкрийте редактор коду. Створіть файл з іменем style.css і збережіть у теці CSS.
3. Відкрийте браузер. Виберіть будь-який генератор кольорових палітр, наприклад <https://coolors.co>. Згенеруйте палітру (рис. 1).



Рис. 1. Приклад палітри

*Примітка.* У наданому вам коді замість «з обраної палітри!» додавайте код кольору. Не забувайте про знак «#» на початку шестицифрового коду. Для кольору тла й тексту варто вибирати контрастні відтінки.

4. Сформуйте стилі для кожного елемента сторінок сайта.
5. Почніть виконувати дії з головних елементів:
  - a) пропишіть стилі для body (рис. 2);
  - b) зробіть список-меню горизонтальним і зніміть значок маркера (рис. 3);

```
body{
 font-family: Arial, San-Serif;
 font-size: 100%;
 background-color: з обраної палітри;
}
```

```
ul{
 margin: 0;
 padding: 0;
}
ul li{
 list-style-type: none;
}
a{
 text-decoration: none;
}
```

Рис. 2. Стилі для body

Рис. 3. Горизонтальний список-меню

в) виберіть стилі для всіх елементів хедера (рис. 4);

```
header{
 padding: 50px 0;
 background-color: з обраної палітри;
}
h1,h2{
 color: з обраної палітри — контрастно до кольору тла;
 text-align: center;
}
```

Рис. 4.  
Елементи  
хедера

г) виберіть стилі для блоку навігації (рис. 5);

```
nav{
 background-color: з обраної палітри;
 border-color: з обраної палітри;
 min-height: 64px;
}
nav li{
 float: left;
 width: 100px;
}
nav a{
 display: block;
 color: з обраної палітри — контрастно до кольору тла;
 line-height: 4em;
 font-weight: bold;
 text-align: center;
}
nav li a:hover{
 color: з обраної палітри — контрастно до кольору зміненого тла;
 background-color: з обраної палітри — контрастно до кольору
активного елемента меню;
}
```

Рис. 5. Блок навігації

д) виберіть стилі для абзаців (рис. 6);

```
p{
 font-size: 16px;
 line-height: 1.3;
 margin-top: 0.1em;
 margin-bottom: 0;
 color: з обраної палітри — контрастно до кольору тла;
}
```

Рис. 6. Абзаци

*Примітка.* Якщо потрібно, щоб між абзацами були відступи, зі стилів слід вилучити рядки з використанням margin.



е) виберіть стилі для футера (рис. 7).

```
footer{
 background-color: з обраної палітри;
 color: з обраної палітри — контрастно до кольору тла;
 padding: 20px 0;
 margin-top: 30px;
}
```

Рис. 7.  
Футер

6. Збережіть файл style.css.
  7. Відкрийте всі файли HTML, створені в практичній роботі № 2.
  8. У кожному файлі в <head> додайте наведений далі рядок, прописуючи посилання на створений файл зі стилями:
- ```
<link rel="stylesheet" type="text/css" href="css/style.css">
```
9. Збережіть і закрійте файли, оновіть у браузері головну сторінку вашого сайта.
 10. Надішліть архів із роботою вчителю.

Зробіть висновок: як змінився сайт після додавання таблиць стилів; порівняйте отримані результати зі скріншотом; сформулюйте і запишіть зміни, які ви помітили.

2.3

Проектування та верстка вебсторінок

Процес створення сайта (вебпроєкту) умовно можна розподілити на кілька етапів (рис. 2.11). Розглянемо такі етапи, як загальне планування; розробка дизайну, планування макета вебсторінок.

Загальне планування складається зі створення ідеї, розробки структури проєкту, опрацювання макета проєкту.

Створення ідеї: необхідно вибрати тему проєкту (сайта сервісу) й відповідно до неї дібрати текстові та графічні матеріали.

Розробка структури проєкту: потрібно визначити кількість розділів сайту, класифікувати матеріал за розділами, приступити до формування навігаційного меню.

Опрацювання макета проєкту: потрібно скласти макет проєкту, використовуючи графічний редактор. У розділі растрової графіки вивчається

графічний редактор Gimp, який має потужний інструментарій для розробки макета сайта, створення банерів, кнопок і логотипів, опрацювання фотографій, створення gif-анімації. Можна схематично скласти макет проекту, використовуючи також папір і ручку.

Етапи створення сайту

Загальне планування сайта

Розробка дизайну сайта

Планування макета вебсторінок

Верстка вебсторінок

Програмування сайта

Розміщення сайта в інтернеті

Рис. 2.11. Порядок дій для створення сайта

Схематичний начерк дає розуміння того, як на сайті розташовуватимуться основні інформаційні блоки, графічні зображення тощо (рис. 2.12).

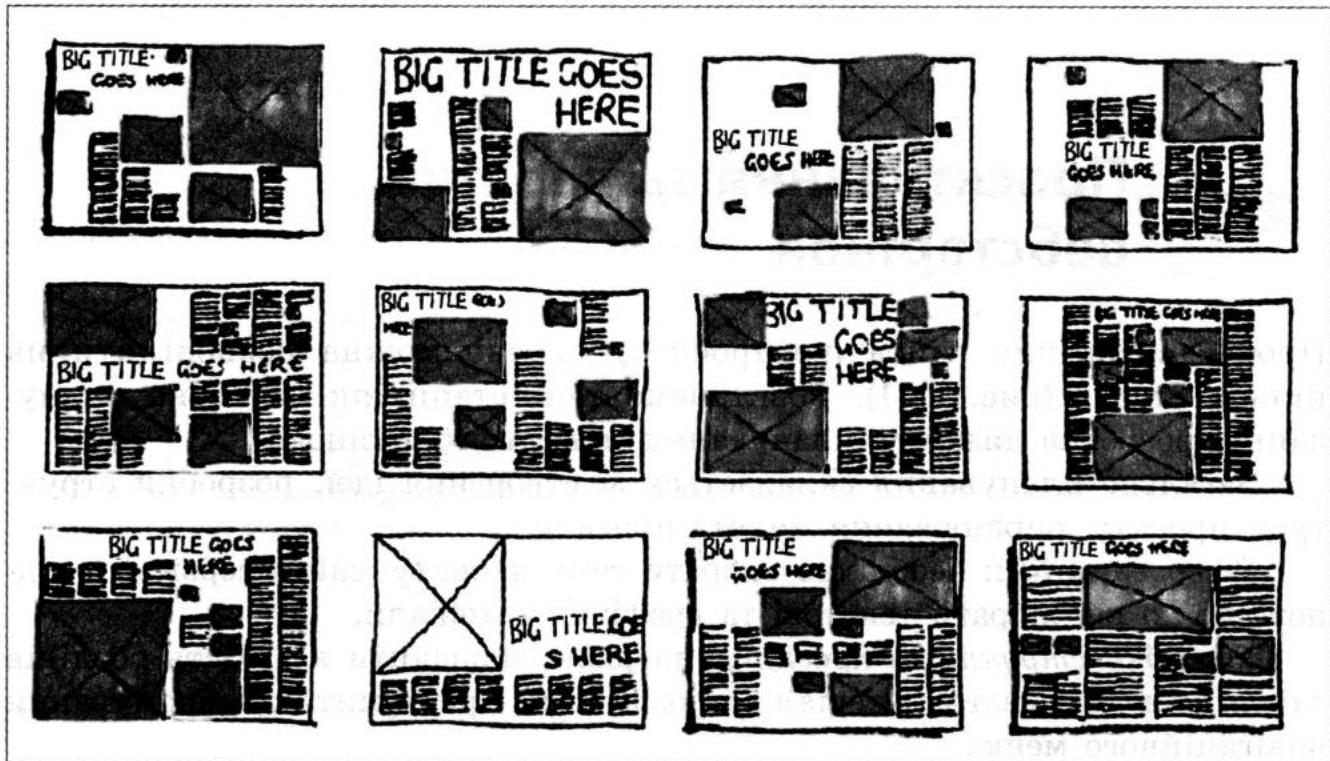


Рис. 2.12. Вибір макета проекту



Переважна більшість сучасних сайтів має блокову верстку. Розглянемо основні елементи вебсторінки сайта: блок (або контейнер — `div`), заголовок (хедер), у який входять назва сайта та логотип, навігація, контент, нижній колонитул (футер, підвал). Звернемося до історії розвитку верстки.

Вебсторінки спочатку були просто набором тегів. Елементи відображалися в тому порядку, в якому записувалися в HTML-коді. Щоб візуально розділити інформацію, використовувалися горизонтальні лінії або відступи.

Кроком уперед була так звана *таблична* верстка вебсторінки: елементи структурувалися за допомогою таблиці, інформація вставлялась у клітинку. Створити сайт за допомогою таблиці було досить просто — достатньо мати початкові знання HTML і CSS і використовувати мінімум правил CSS.

Таблична верстка була дуже популярною на початку 2000-х років. До сьогодні існує досить багато сайтів, які зверстано у вигляді таблиць.

Таблична верстка має низку недоліків: при складній структурі важко розібратися в коді, браузери відображають таблицю на екрані лише після повного завантаження, а складний дизайн з перекриттям елементів узагалі неможливо реалізувати. Таблична верстка неприйнятна для побудови адаптивних сайтів.

Наразі найпопулярнішим способом верстки є так званий *блоковий*. При цьому використовуються блоки, які або розташовують один під одним, або керують порядком їх відображення за допомогою позиціонування в CSS.

Контент є основною складовою вебсторінки. Він відіграє провідну роль у дизайні, тому займає більший простір і підкріплений графікою.

Розглянемо основні елементи вебсторінки сайта.

Елемент `div()` (від англ. *division* — розділ, блок) зазвичай виконує на вебсторінці роль контейнера (рис. 2.13).

Теги `div` мають такі характеристики:

- ◆ `div` — блочний елемент; якщо ширину не задано, блок займає всю ширину браузера;
- ◆ `div` — висота блоку; якщо висоту не задано, то блок дорівнює вмісту; порожній блок `div` має висоту 0 px, тому не відображається на сторінці;
- ◆ `div` не має оформлення; щоб його побачити, потрібно задати стилі в CSS;
- ◆ `div` — може містити будь-яке число вкладених елементів; можна вклсти інші блоки `div`, заголовки, таблиці, зображення та ін.

Елемент `<header>` (заголовок) є контейнером, у якому містяться назва сайта, логотип і навігаційна панель.

Логотип найчастіше розташовується у верхньому лівому кутку вебсторінки або посередині, залежно від ідеї, макета (рис. 2.14).

```
<div class="text">
    Lorem ipsum dolor sit amet, consectetur
    adipisicing elit, sed do eiusmod tempor
    incididunt ut labore et dolore magna aliqua.
    Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex
    ea commodo consequat. Duis aute irure dolor
    in reprehenderit in voluptate velit esse
    ...
</div>
```

a

```
.text{
    margin-top: 3vh;
    width: 70%;
    margin-left: auto;
    height: 16vh;
    overflow: hidden;
    line-height: 4vh;
    color: #303030;
    font-family: Verdana;
    font-size: 2vh;
    transition: 0.4s height;
}
```

б

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

в

Рис. 2.13. Приклад використання контейнера на вебсторінці: код HTML (а); правила CSS (б); відображення в браузері (в)

```
<a class='w3schools-logo notranslate'
    href='//www.w3schools.com'>w3schools<span
    class='dotcom'>.com</span></a>

.w3schools-logo {
    font-family: fontawesome;
    text-decoration: none;
    line-height: 1;
    font-size: 37px;
    letter-spacing: 3px;
    color: #555555;
    display: block;
    position: absolute;
    top: 17px;
}
w3schools-logo .dotcom
{color: #4CAF50}
```



Рис. 2.14. Приклад запису та відображення у браузері логотипа сайта



Навігаційна панель часто розташовується у верхній частині вебсторінки незалежно від того, вертикально або горизонтально розташовані елементи навігації сайту (рис. 2.15), і містить посилання на його основні розділи.

```
<nav class="menu">
  <li>
    <a href="#">Головна</a>
  </li>
  <li>
    <a href="#">Галерея</a>
  </li>
  <li>
    <a href="#">Контакти</a>
  </li>
</nav>
```

HTML-код

```
menu{
  display: inline-block;
  width: 100vw;
  height: 10vh;
  background-color: lightgreen;
  align-items: center;
  justify-content: center;
  margin-top: 35vh;
  padding-left: 18vw;
  padding-right: 18vw;
}
.menu li{
  list-style: none;
}
```

Правила CSS

[Головна](#)[Галерея](#)[Контакти](#)

Рис. 2.15. Навігаційна панель сайту

Нижній колонтитул (footer, підвід) розташовується внизу вебсторінки і містить інформацію про правовласників, контактні та юридичні дані, посилання на розділи сайту (найчастіше дублює основну навігацію), посилання на соціальні мережі, форму зворотного зв'язку та ін. (рис. 2.16).

```
<footer>
  ©All Right Reserved
  <br>
  Знайдіть нас у мережах:
  <a href="google.com">Google</a>
  <a href="youtube.com">YouTube</a>
</footer>
```

HTML-код

```
footer{
  background: #b6e283;
  width: 100%;
  padding: 1%;
```

Правила CSS

©All Rights Reserved
Знайдіть нас у мережах:
[Google](#)
[YouTube](#)

Рис. 2.16. Нижній колонтитул вебсторінки

Модульна сітка передбачає поділ вебсторінки на окремі колонки по вертикалі та вибудовування контенту. Дизайн макета зазвичай розробляється саме за цією сіткою (рис. 2.17).

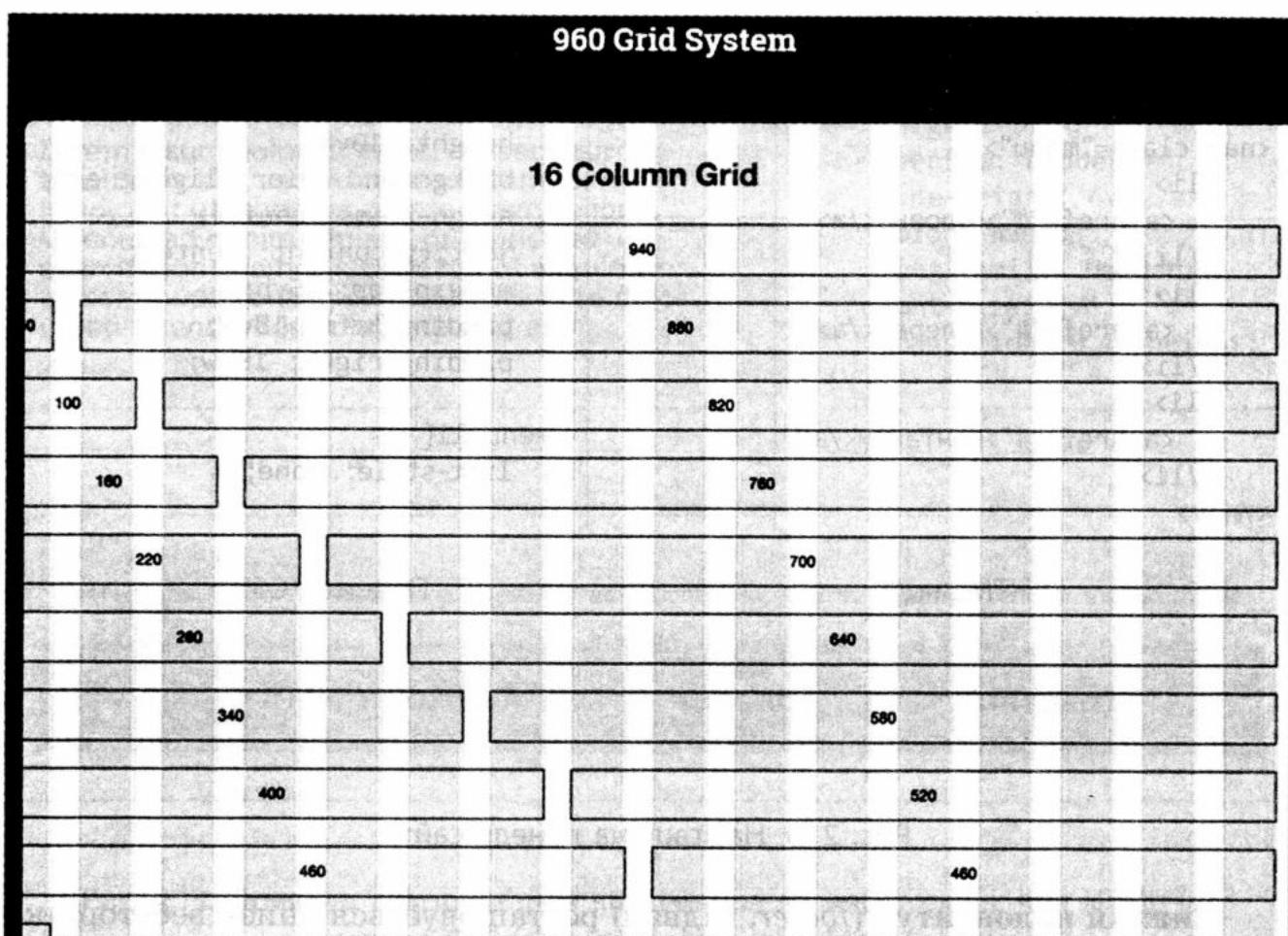


Рис. 2.17. Модульна сітка

Найбільш популярною є модульна сітка 960 Grid System (<http://960.gs>), яка максимально ділить сторінку на 12, 16 і 24 колонки. За ширину сітки становить максимум 960 пікселів. Такий вибір базується на тому, що на момент створення сітки більшість сучасних моніторів мали роздільність 1024×768 пікселів.

Створення макета на основі модульної сітки допоможе в подальшому прискорити процес верстання. Завдяки їй блоки контенту й елементи розташовуватимуться на певній відстані один від одного.

Можна сказати, що модульна сітка — це певна візуальна абстракція, візуальний розподіл сторінки на однакові за ширину стовпці з однаковими відступами між ними. Візуалізувати модель можна за допомогою напрямних або окремого шару, на якому будуть зображені ці стовпці (знову пригадаємо графічний редактор Gimp).

Серед усього різноманіття складання макета вебсторінки можна виділити різні типи навігації.

Розглянемо чотири найбільш поширені типи навігації вебсторінкою сайта (рис. 2.18):



- ◆ навігація в лівому стовпці (випадок а);
- ◆ навігація в правому стовпці (випадок б);
- ◆ горизонтальна навігація (випадок в);
- ◆ навігація Mobile First (випадок г).

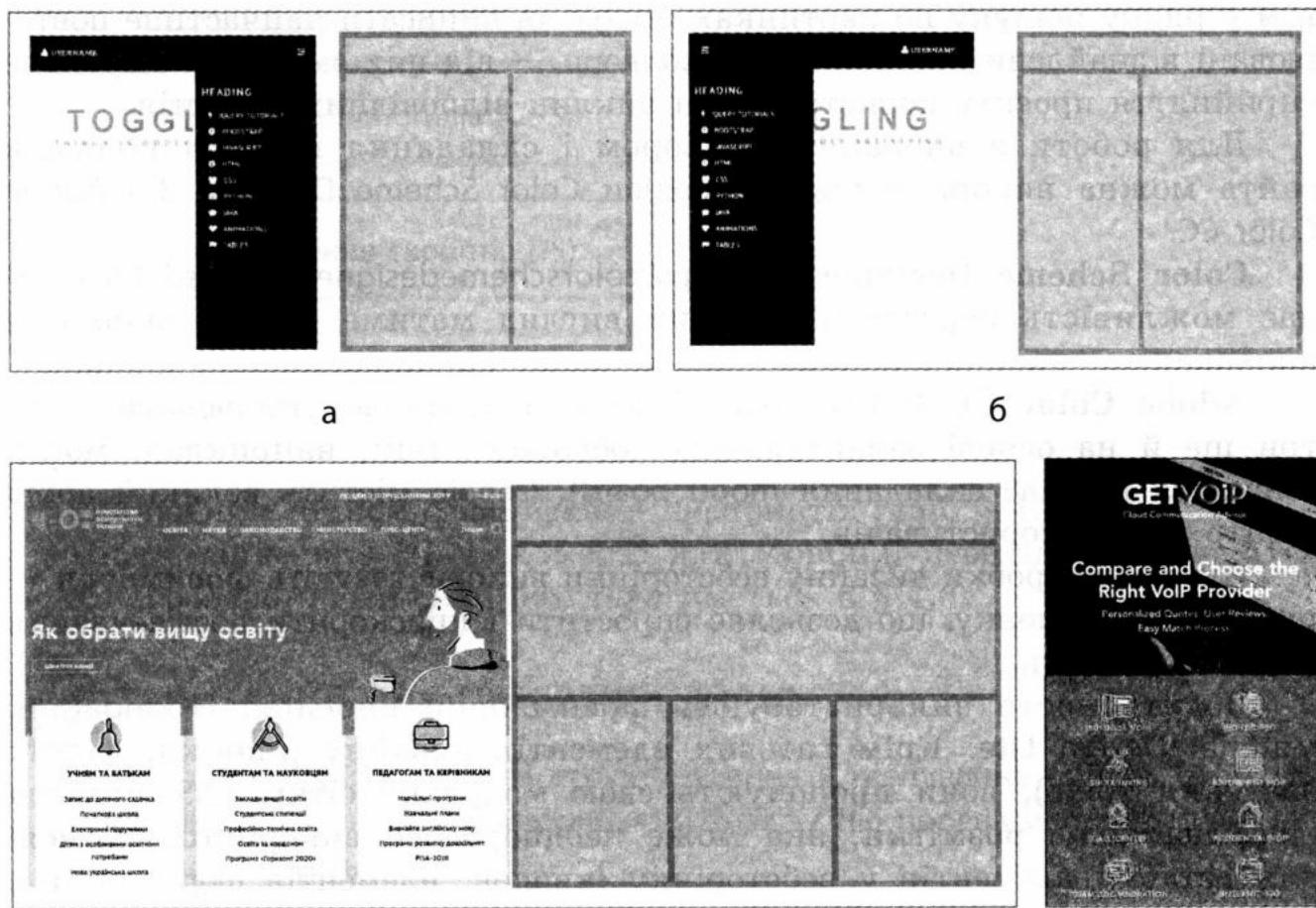


Рис. 2.18. Приклади сайту: з навігацією в лівому стовпці (а); з навігацією в правому стовпці (б); з горизонтальною навігацією (в); з технологією Mobile First (г)



Сайти з горизонтальною навігацією становлять більшість. Такий вибір пояснюється зручністю, адже залишається більше простору для контенту, яким наповнено сайт.

Понад 80 % користувачів інтернету використовують для доступу в мережу мобільні пристрої, тож правилом хорошого тону стає розробка не тільки десктопної, а й мобільної версії сайта.

З урахуванням тенденцій останніх років тип навігації Mobile First впевнено займає свою нішу. У разі його використання розробка макета сайта, дизайну й верстки починається з мобільної версії, а вже потім опрацьовуються макети для інших роздільностей: додаються блоки, банери, додаткові елементи дизайну тощо (більш детально цей підхід описано в § 2.4).

Одним зі способів визначення основного кольору в проєкті є складання mood board (у перекладі з англійської мови — дошка настрою). Під час створення дизайну макета варто розпочати роботу з визначенням колірної гами проєкту.

Порядок складання mood board такий: потрібно набрати кожен синонім у рядку пошуку по картинках Google та виписати найчастіше повторювані в знайдених зображеннях кольори — від них залежить візуальне сприйняття проєкту користувачем і виклик відповідних почуттів.

Для роботи з вибраним кольором і складання палітри кольорів сайта можна використовувати сервіси Color Scheme Designer 3 і Adobe Color CC.

Color Scheme Designer 3 (<http://colorschemedesigner.com/csd-3.5/>) надає можливість переглянути, який вигляд матиме сайт у вибраних кольорах.

Adobe Color CC (<https://color.adobe.com>) дозволяє створювати палітри ще й на основі завантажених зображень (які, наприклад, могли з'явитися під час складання mood board). Сервіс містить великий архів палітр інших користувачів.

Під час розробки дизайну вебсторінки використовують **фреймворк** — програмну оболонку, що дозволяє спростити і прискорити розв'язування типових завдань.

Досить часто використовують фреймворки Bootstrap, Foundation, Material Design Lite. Крім готових елементів дизайну (кнопки, форми введення тощо), вони пропонують свою модульну сітку, CSS-сніппети (частина коду, розмітки, яка може неодноразово використовуватися) для вставки елементів у вебсторінку (кнопок, елементів форм та ін.), і класи розмітки, а так само JS-скрипти для відповідних інтерактивних елементів.

На основі певного фреймворка можна знайти величезну кількість платних і безкоштовних тем і сторінок, а також розробити власні.

Під структурою проекту розуміють зберігання файлів проєкту в його директорії. окремі категорії файлів необхідно поміщати у свої папки: картинки, в папку images або img, css — у папку css, javascript — у папку js. У корені лежатимуть лише index.html і вебсторінки сайта або тільки index.html, а вебсторінки — в окремій папці pages.

Важливо дотримуватись подібних правил і під час іменування файлів проєкту.

Найчастіше застосовуються такі імена, як головна сторінка — index.html, стилі проєкту — styles.css, скрипти — scripts.js або app.js. Мінімізовані версії файлів мають префікс .min, назви картинок відображають те, що на них зображене.

Верстання вебсторінки здійснюється поетапно. Порядок дій під час верstanня наведено на рис. 2.19.

Як бачимо, за допомогою тегів HTML спочатку створюють скелет сторінки, орієнтуючись на структуру, складену ще на першому етапі. Далі визначають необхідні класи та переходять до написання CSS-стилів.

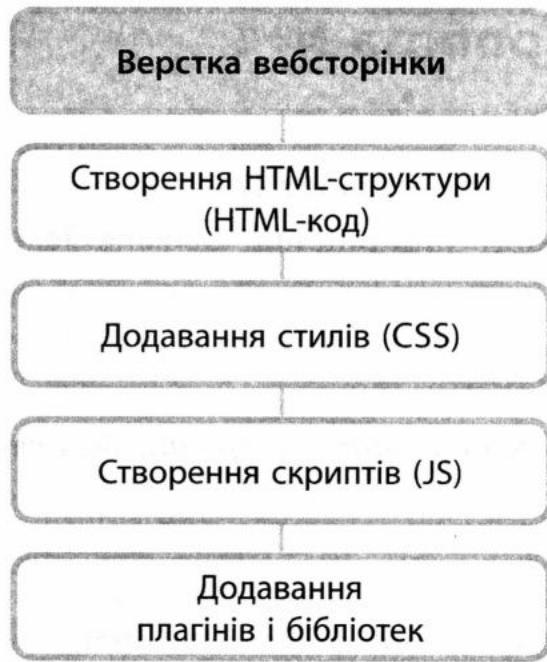


Рис. 2.19. Етапи верстання вебсторінки

Заключним етапом є написання JS-скриптів (див. § 4.1–4.2).

Після написання HTML, CSS і JS для сторінки необхідно перевірити, чи все зроблено правильно (розділиться в § 4.6)



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

- Сформулюйте правила структурування й іменування файлів.
- Які типи макетів вебсторінок ви знаєте? Які переваги та недоліки вони мають?
- Опишіть етапи створення сайта.
- Яке місце в дизайні сайта, на вашу думку, посідає колірна гама?
- Чим корисні фреймворки?
- Опишіть етапи верстки вебсторінки.



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

- Створіть теку для вашого майбутнього сайта з назвою Site.
- У створеній текі задайте ще відповідно теки images, css, js.
- Перенесіть у теку Site файл index.html, модифікований на минулому уроці.
- У теку css перенесіть файл style.css.
- У файлі index.html виправте посилання на файл style.css так, щоб ви могли знайти його у новоствореній текі. Для цього у записі `<link rel="stylesheet" type="text/css" href="style.css">` пропишіть шлях до файла наступним чином `href="css/style.css"`.
- Перегляньте отриманий результат у браузері.



Дивіться презентацію «Проектування та верстка вебсторінок».

Практична робота № 4

ТЕМА. Блокова модель CSS

ЗАВДАННЯ: навчитися використовувати блокову модель.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

Тренувальна вправа № 1

1. Створіть теку site-block.
2. Відкрийте редактор коду й створіть у теці файл index.html.
3. Запишіть код, наведений на рис. 1:

```
<!DOCTYPE html>
<html>
<head>
    <title>ONE BLOCK</title>
</head>
<body>
    <style type="text/css">
        .one{
            width: 70%;
            background: yellow;
            margin-top: 10%;
            margin-left: 15%;
        }
    </style>
    <div class="one">
        ДОДАЙТЕ ЗГЕНЕРОВАНИЙ РЯДОК "Lorem Ipsum"
    </div>
</body>
</html>
```

Рис. 1. Код для додавання

4. Збережіть файл і перегляньте результат.
5. Замініть згенерований рядок на один-два згенерованих абзаци Lorem Ipsum.
6. Проаналізуйте отриманий результат.

Тренувальна вправа № 2

1. Замініть код на поданий на рис. 2. Перегляньте отриманий результат.
2. Перемістіть згенерований абзац усередину тега <div class = «inner»>. Перегляньте отриманий результат.



```

<!DOCTYPE html>
<html>
<head>
    <title>INNER</title>
    <meta charset="utf-8">
</head>
<body>
    <style type="text/css">
        body{
            margin: 0;
        }
        .main{
            margin-top: 110px;
            width: 80%;
            height: 600px;
            margin-left: 10%;
            background: lightgreen;
        }
        .inner{
            width: 60%;
            height: 400px;
            background: yellow;
            margin-left: 20%;
        }
        .header{
            width: 100%;
            height: 100px;
            background-color: violet;
            position: fixed;
            top: 0;
        }
    </style>
    <div class="header"></div>
    <div class="main">
        <div class="inner"></div>
    </div>
    ДОДАЙТЕ ЗГЕНЕРОВАНИЙ РЯДОК "Lorem Ipsum"
</body>
</html>

```

Рис. 2. Код для заміни

3. Скопіюйте згенерований абзац усередину тега <div class="main">. Перегляньте отриманий результат
4. Проаналізуйте результати тренувальної вправи № 2.

Тренувальна вправа № 3

1. Замініть код на поданий на рис. 3.

```

<!DOCTYPE html>
<html>
<head>
    <title>Parent-child</title>
</head>
<body>
    <style type="text/css">
        body{
            margin: 0;
        }
        .parent{
            width: 80%;
            background: yellow;
            margin-left: 10%;
            position: relative;
        }
        .child{
            width: 50%;
            height: 200px;
            background: lightgreen;
            position: absolute;
            bottom: 20px;
            right: 20px;
            left: auto;
            top: auto;
            z-index: 9000;
        }
    </style>
    <div class="child"></div>
    <div class="parent">
        ДОДАЙТЕ ЗГЕНЕРОВАНИЙ РЯДОК "Lorem Ipsum"
    </div>
</body>
</html>

```

Рис. 3. Код на заміну

2. Збережіть і перегляньте результат.
3. Скопіюйте згенерований абзац усередину тега `<div class="child">`. Проаналізуйте отриманий результат.
4. Запишіть здійснений вами аналіз результатів трьох тренувальних вправ у текстовий документ і надішліть учителю на поштову скриньку.

Зробіть висновок за результатами роботи.

2.4

Адаптивна верстка



Книга Аарона Густафсона



Книги Ітана Маркотта та Люка Вроблевські

не були оптимізовані під екран великих розмірів, планшетів і Smart-TV є невід'ємною частиною сучасного життя, а кількість і різноманітність мобільних пристрій постійно збільшується (рис. 2.20).

Першою спробою підлаштування сайта під розміри екранів стала так звана *гумова верстка*. У разі її застосування й макету, й окремим його складовим надаються розміри не фіксовані (рис. 2.21, а), а подані у відсотках (рис. 2.21, б), тобто вміст сторінки розтягується. Проте коли, наприклад, той самий контент розтягується на 6-дюймовий екран смартфона та 42-дюймовий екран телевізора, це вже заважає. Тож необхідно встановити максимальний і мінімальний розміри (власність `max-width`).

Мобільні пристрої повсякчас урізноманітнюються й удосконалюються. Ознайомимося з ними докладніше.

Із появою мобільного інтернету до класичного вебсайту додався так званий *wap-сайт*. Перший, як відомо, призначений для перегляду на комп’ютері, другий — з мобільного телефону. Погодьтеся, це додало зручності. Адже веб сайти, маючи широкий функціонал і яскраве оформлення, дуже довго завантажувалися й забирали занадто багато трафіку. Натомість вап-сайти містили в основному текст і поодинокі зображення. Додамо, що на той час мобільні телефони були слабкі, а мобільний інтернет повільний і дуже дорогий.

Згодом широким попитом стали користуватися планшети. Це створило нові виклики фахівцям у галузі вебдизайну, адже фіксована верстка не дозволяла змінювати розмір роздільності сайта. Великі сайти переглядати з планшетів було незручно, а мобільні сайти



Наступним кроком стала поява медіа-запитів. Цьому сприяла пропозиція Марка ван ден Доббелстіна щодо додавання класів під час завантаження та оголошення кожному діапазону спеціальних стилів.

У 2006 році в інтернет-журналі A List Apart (<https://alistapart.com/>) вийшла стаття Доббелстіна, а через два роки з'явилися медіа-запити.

Медіа-запити (*media queries*) — це правила CSS, які дозволяють керувати стилями елементів залежно від значень технічних параметрів пристрійв.

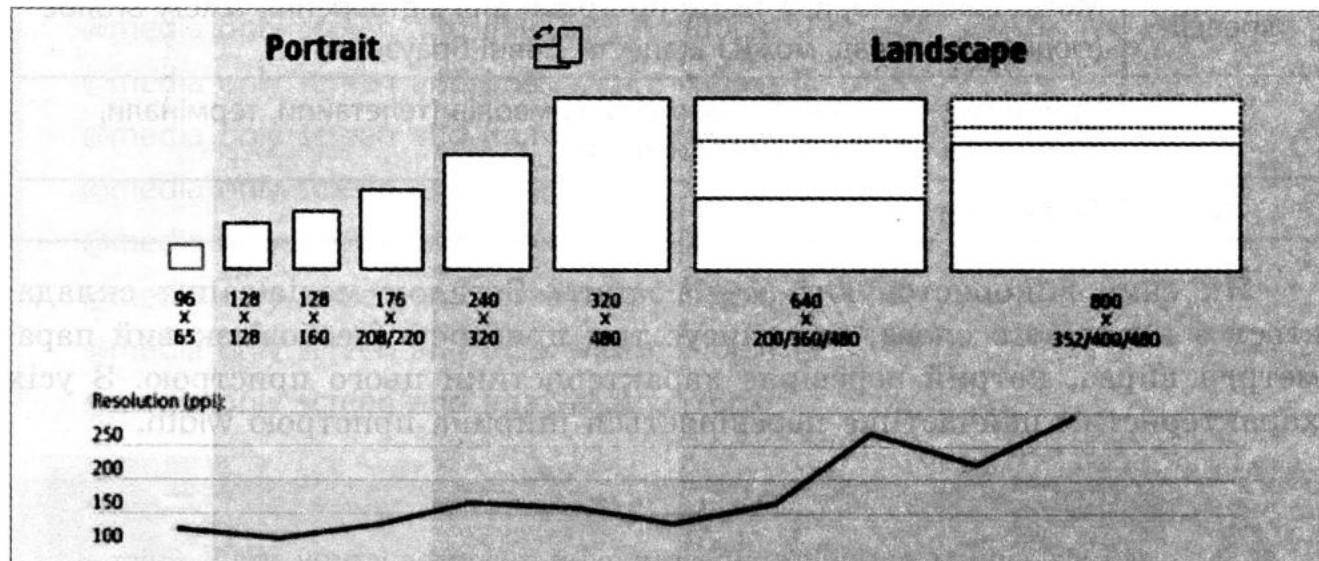


Рис. 2.20. Статистика різноманітних девайсів та їх роздільність

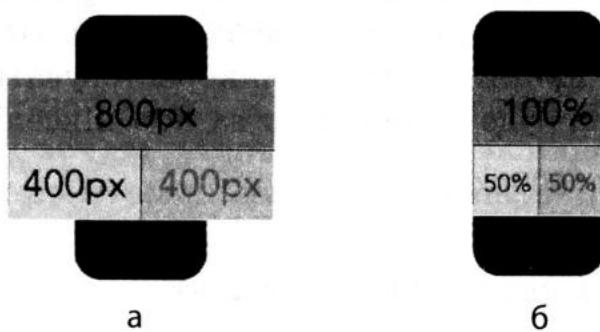


Рис. 2.21. Приклад відображення на смартфоні макета вебсторінки:
а — фіксована верстка;
б — гумова верстка

У 2001 році в HTML4 і CSS2 була введена підтримка апаратнозалежних таблиць стилів, що дозволило створювати стилі й таблиці стилів для певних типів пристроїв.

Таким чином, браузер застосовував таблицю стилів лише у випадку, коли активувався саме цей тип пристрою:

Тип носія	Опис медіа-запитів
all	Всі типи. Це значення використовується за замовчуванням
braille	Пристрої, засновані на системі Брайля й призначені для читання людьми з вадами зору
embossed	Принтери, що використовують для системи друку шрифт Брайля

Закінчення таблиці

Тип носія	Опис медіа-запитів
Handheld	Смартфони та аналогічні пристрої
print	Принтери та інші друкарські пристрої
projection	Проєктор
screen	Екран монітора
speech	Мовні синтезатори, а також програми для відтворення тексту вголос (сюди, наприклад, можна віднести мовні браузери)
tty	Пристрої з фіксованим розміром символів (телетайпи, термінали, пристрої з обмеженнями дисплея)
tv	Телевізори

Як саме використовують медіа-запит? Загалом медіа-запит складається з ключового слова, що описує тип пристрою (необов'язковий параметр) і вираз, котрий перевіряє характеристики цього пристрою. З усіх характеристик найчастіше перевіряється ширина пристрою `width`.



Медіа-запит є логічним виразом, який повертає істину або хибність.

Запит записують у кінці таблиці стилів (нагадаємо правила запису стилів: у стилів, які записані пізніше, вища пріоритетність). Наприклад, наведений далі запис означає, що стилі будуть використовуватися лише у випадку, коли ширина екрана буде не більшою за 600 пікселів:

```
@media screen and (max-width: 600px) {  
}
```

Розглянемо на прикладі використання медіа-запитів.

ПРИКЛАД

```
<style>  
    .block_left { width: 430px; }  
    @media (max-width: 1220px) {  
        .block_right { width: 380px; }  
    }  
    @media (max-width: 1120px) {  
        #block { width: 325px; }  
    }  
    @media (max-width: 680px) {  
        #block { width: 200px; }  
    }  
</style>
```



Наведемо список популярних медіа-запитів для стандартного набору розширення екранів.

```
@media only screen and (max-width: 1920px) /* CSS правила */
@media only screen and (max-width: 1680px) {}
@media only screen and (max-width: 1366px) {}
@media only screen and (max-width: 1280px) {}
@media only screen and (max-width: 1024px) {}
@media only screen and (max-width: 800px) {}
@media only screen and (max-width: 768px) {}
@media only screen and (max-width: 600px) {}
@media only screen and (max-width: 533px) {}
@media only screen and (max-width: 360px) {}
@media only screen and (max-width: 320px) {}
@media only screen and (max-width: 240px) {}
@media only screen and (max-width: 176px) {}
```



ЦІКАВІ ФАКТИ

Ідея розробки адаптивної верстки належить Аарону Густафсону. У своїй книзі «Адаптивний вебдизайн» він запропонував гнучко адаптувати сайти до можливостей пристрій і браузерів.

Адаптивні сайти з'явилися завдяки зростанню попиту тих користувачів, які хочуть завантажувати вебсторінки з ідеальною структурою на всіх своїх пристроях. Саме зараз адаптивний дизайн стає обов'язковим. Він значно спрощує регулярний серфінг в інтернеті незалежно від того, який пристрій вибрав користувач для доступу до сайтів, що його цікавлять.

Сьогодні прийнято вважати, що головне в адаптивній верстці — це прив'язка до конкретних пристрій із певною роздільністю екрана. Стилі перемикаються від одного брейкпоінта (контрольні точки, сталий термін у програмуванні, у випадку верстки — перемикання з одних умов на інші) до іншого, тобто насправді є фіксовані макети для кожного дебайсу.

Традиційно макет сайта спочатку розробляється для десктопної версії, а вже потім адаптується до мобільної. Люк Вроблевські у своїй книзі «Спочатку мобільні» запропонував піти від протилежного: розробити макет сайта для мобільної версії, а вже потім покращувати до десктопної. Вроблевські керувався тим, що ускладнювати просте легше, ніж спрощувати складне.

За Аароном Густафсоном, адаптивність — це особливий підхід до розробки сайта, який дозволяє вже наявним вебресурсам підлаштовуватися

під розміри екранів різних пристрій. Інакше кажучи, сторінка повинна автоматично підлаштовуватися під екран, змінювати розмір картинок тощо. Це дозволило усунути потребу в розробці дизайну для кожного нового типу пристрою.



ЦІКАВІ ФАКТИ

Концепцію чуйного дизайну у 2010 році запропонував та описав у своїй книзі *Responsive Web Design* Ітан Маркотт. Особливістю такого підходу стала плавна зміна сайту, зорієнтована на вміст, а не на конкретні пристрої.

Основні особливості адаптивного дизайну:

- застосування гнучкого макета на основі сітки (англ. *flexible, grid-based layout*);
- використання гнучких зображень (англ. *flexible images*);
- робота з медіа-запитами (англ. *media queries*);
- плавна перебудова блоків у разі зміни розміру екрана (наприклад, під час повертання планшета).

Якщо зображення гнучке, це означає, що його розмір змінюється залежно від розміру контейнера, в якому воно міститься (див. § 3.1).

Flexbox, Grid і багатоколонкова верстка (*Multi-column layout*) є адаптивними за замовчуванням (їх специфікації були написані у світі, де адаптивний дизайн і кросдевайсність вже стали реальністю). Це означає, що їм притаманні безліч функцій, які дозволяють легко створювати адаптивні сітки (рис. 2.22).

The screenshots illustrate the responsive design of Ian MacKott's website, 'The Baker Street INQUIRER'. The site features a header with the title 'The Baker Street INQUIRER' and a tagline 'Give me problems, give me work.' Below the header, there is a main article with a large image of two men in suits, followed by text and smaller images of people. The left sidebar contains links for 'WEBLOGUE', 'BASIC ISSUES', and 'ABOUT OUR PAPER'. The right sidebar has sections for 'victors & villains' and 'Illustrations by Arthur Conan Doyle, words by Sir Arthur Conan Doyle. What remains are by Ellen Harmon.' The bottom of the page shows a footer with social media icons and links.

Рис. 2.22. Сайт Ітана Маркотта



CSS flexbox (Flexible Box Layout Module) — модуль макета гнучкого контейнера, що являє собою спосіб компонування елементів. Технологія flexbox ставить на меті зробити шари гнучкими, а роботу з ними інтуїтивно зрозумілою.

Flexbox складається з гнучкого контейнера (*flex container*) і гнучких елементів (*flex items*). Останні можуть вибудовуватися в рядок або стовпець, а вільний простір розподіляється між ними різними способами.

Модуль flexbox дозволяє:

- ◆ розташовувати елементи в одному з чотирьох напрямків: зліва направо, справа наліво, зверху вниз або знизу вгору;
- ◆ перевизначати порядок відображення елементів;
- ◆ автоматично визначати розміри елементів так, щоб вони вписувалися в доступний простір;
- ◆ розв'язувати проблему з горизонтальним і вертикальним центруванням;
- ◆ переносити елементи всередині контейнера, не допускаючи його перевнення;
- ◆ створювати колонки однакової висоти.

Flex (від англ. *flex* — розтягувати) і *flex-inline* — це параметри для властивості елемента *display* батьківського HTML-елемента, що містить дочірні блоки. На відміну від інших параметрів цього елемента (*block* || *inline-block* || *table*) вони не задають новий тип відображення елемента.

Вони надають можливість керувати поведінкою дочірніх елементів, вкладених у контейнер-обгортку з цією властивістю, наприклад змінювати їх розмір і відстань між ними (рис. 2.23 та 2.24).

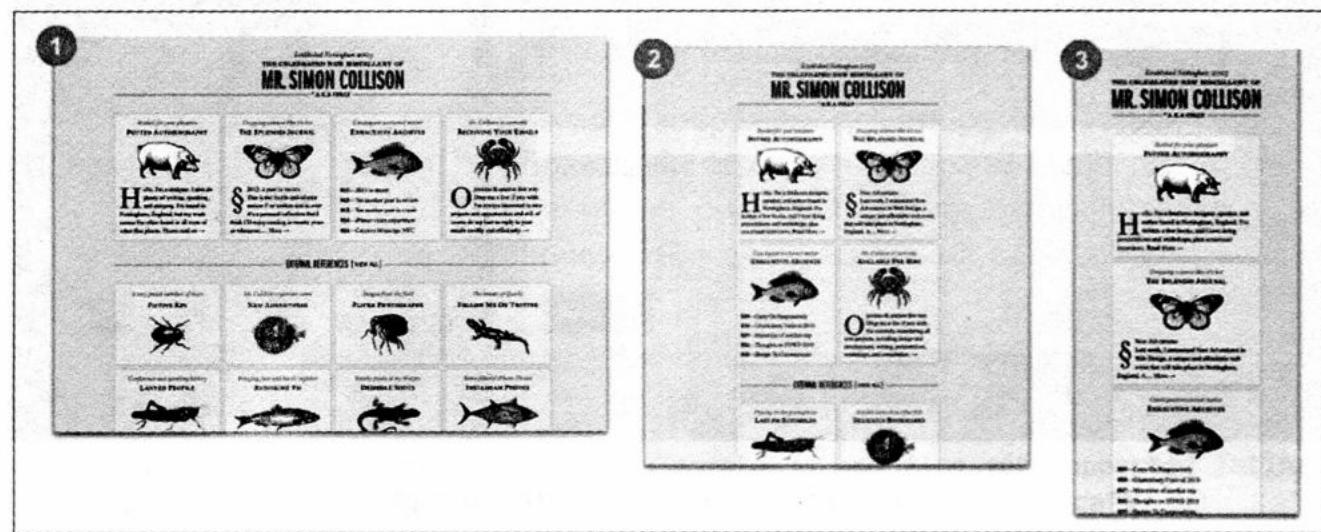


Рис. 2.23. Приклад адаптивної верстки на основі медіа-запитів

```
<div class="new-flex-container">
    <div class="new-flex-block">item1</div>
    <div class="new-flex-block">item2</div>
    <div class="new-flex-block">item3</div>
</div>
```

Рис. 2.24. Приклад оголошення flexbox

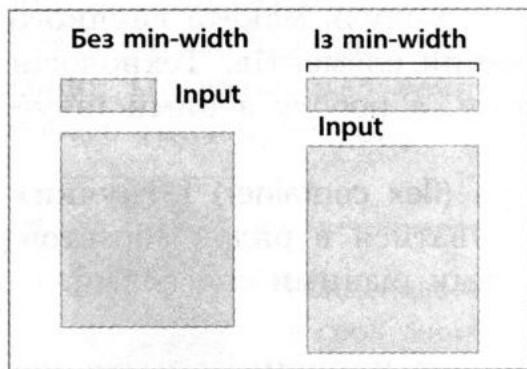


Рис. 2.25. Використання ширини

Flexbox дозволяє керувати елементами лише в одновимірному просторі, тому наступним кроком є розробка модуля grid, який дозволяє оперувати стовпцями та рядками. Більш детально з модулем flexbox можна ознайомитися за посиланням: <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Для тих, хто хоче досконало вивчити властивості flexbox, можна спробувати свої сили в грі Flexbox Froggy (рис. 2.26), де потрібно допомагати жабеняті Фротті та його друзям з написанням CSS-коду (<https://flexboxfroggy.com/#uk>).

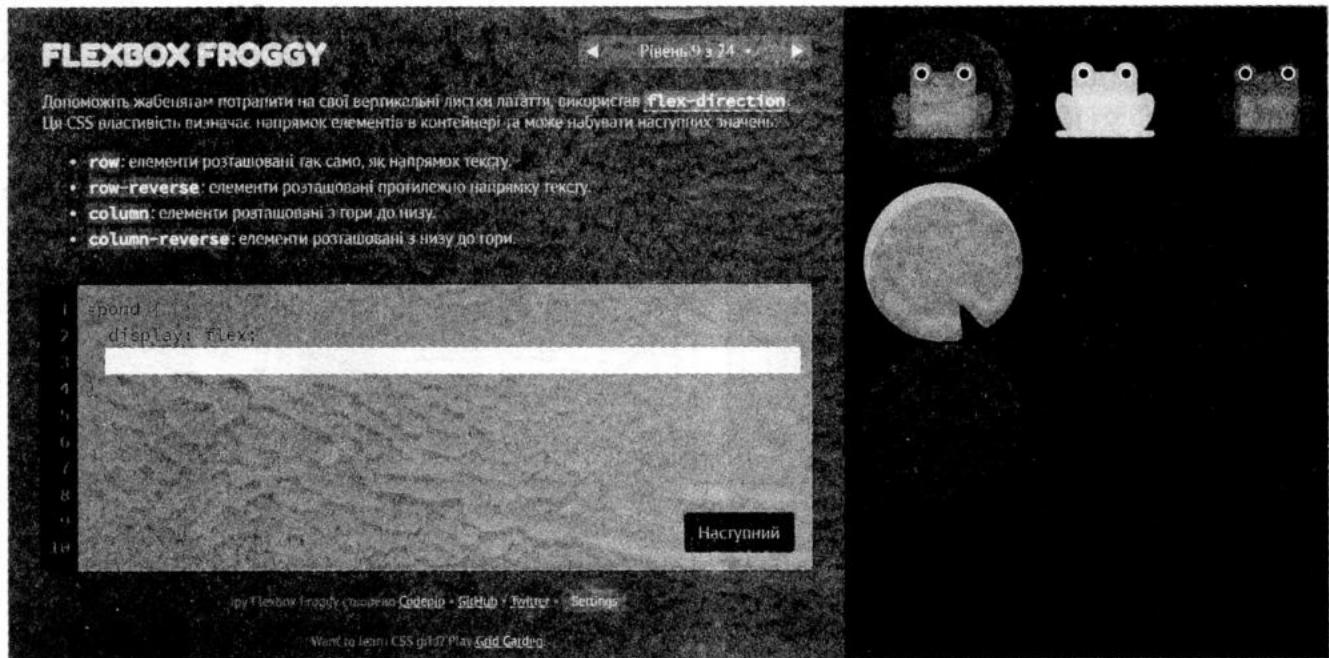


Рис. 2.26. Вікно гри Flexbox Froggy

Grid container (грід-контейнер) — це набір горизонтальних і вертикальних grid-ліній, що перетинаються. Ці лінії ділять простір на grid-області, де розташовуються grid-елементи. Усередині grid-контейнера є два набори grid-ліній: один визначає вісь стовпців, інший — вісь рядків.

Модуль grid також працює з елементом display (рис. 2.27).

Вивчити властивості модуля grid допоможе гра «Морквяні городи» за посиланням: <https://codepip.com/games/grid-garden/>



```
<div class="grid">
    <div class="box item1">item 1</div>
    <div class="box item2">item 2</div>
    <div class="box item3">item 3</div>
</div>

.grid{
    display: grid;
    grid-gap: 20px;
    grid-template: 100px auto 100px/1fr 80px 3fr 20%;
}
```

Рис. 2.27. Приклад використання модуля grid



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

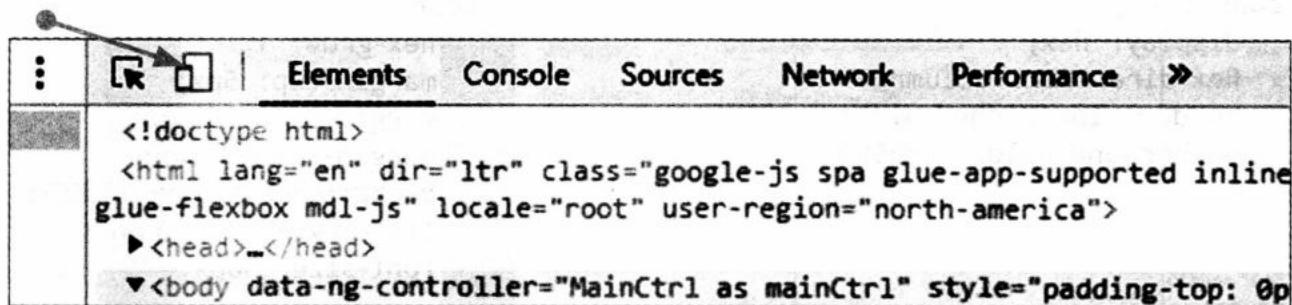
1. Поясніть необхідність розробки адаптивних сайтів.
2. Що таке медіа-запити? Чим обумовлена їх поява?
3. Яка сфера використання медіа-запиту braille?
4. Опишіть принципи адаптивного дизайну.
5. Які інструменти дозволяють розробляти адаптивні сайти?
6. Чому для адаптивної верстки краще обирати векторну графіку?



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

Примітка. Для розуміння роботи адаптивного шаблона відкрийте сторінку Google.

1. Зменшуйте розмір вікна браузера і слідкуйте за тим, як змінюються положення блоків, зникають зображення, як перетворюється меню.
2. Опишіть зміни у відображеній сторінці з режимом інспекції.
3. Розгляньте код, наведений на рис. 2.28. З'ясуйте, який механізм використовують розробники сайта для досягнення адаптивності.



```
<!doctype html>
<html lang="en" dir="ltr" class="google-js spa glue-app-supported inline-glue-flexbox mdl-js" locale="root" user-region="north-america">
  <head>...</head>
  <body data-ng-controller="MainCtrl as mainCtrl" style="padding-top: 0px">
```

Рис. 2.28. Код до завдання 3



Дивіться презентацію «Адаптивна верстка».

Практична робота № 5

ТЕМА. Використання модуля CSS Flexbox

ЗАВДАННЯ: створити адаптивну навігаційну панель засобами модуля CSS Flexbox.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

1. Візьміть за основу навігаційну панель, розроблену в практичній роботі № 2.
2. Відкрийте в редакторі коду HTML-файл index.html. Призначте елементу ul клас контейнер (рис. 1).

```
<nav>
  <ul class="container">
    <li><a href="index.html">Home</a></li>
    <li><a href="info.html">About</a></li>
    <li><a href="gallery.html">Gallery</a></li>
    <li><a href="form.html">Contacts</a></li>
  </ul>
</nav>
```

Рис. 1. Призначення класу

3. Відкрийте в редакторі коду CSS-файл style.css. Оголосіть створений клас із наведеними атрибутами (рис. 2).
4. Призначте елементам списку атрибути, як наведено на рис. 3. Замість запропонованих кольорів виберіть на свій розсуд колір тла та колір межі.

```
.container{
  display: flex;
  flex-direction: column;
  border: 1px dashed white;
  background-color: #ff5599;
}
```

```
.container li {
  flex-grow: 1;
  margin-top: 5px;
  margin-bottom: 2px;
  margin-right: 8px;
  background-color: #ff5555;
  list-style-type: none;
  font-size: 50px;
}
```

Рис. 2. Клас із наведеними атрибутами

Рис. 3. Атрибути

5. Перегляньте браузером отриманий результат. Зробіть скріншот і розмістіть його на слайді презентації з відповідним записом значення модуля Flexbox.

6. Замініть значення атрибута flex-direction на column-reverse.
Збережіть і оновіть сайт у браузері. Зробіть скріншот і розмістіть його на наступному слайді презентації.
7. Перегляньте сайт при зміні значень атрибута flex-direction на row-reverse і row відповідно.
Зробіть скріншоти, збережіть їх у презентації.
8. Додайте атрибут justify-content. Надайте значення center, flex-start, flex-end, space-around. Зробіть скріншоти для кожного значення атрибута.
9. Збільшіть висоту контейнера (height: 200px або 300px). Додайте атрибут align-items.
Перегляньте й зробіть скріншоти для значень center, flex-start, flex-end, stretch.
10. Зробіть висновок про природу значення stretch атрибута align-items і space-around атрибута justify-content.
11. Скористайтеся для перевірки браузером Google Chrome (сполучення клаівіш Ctrl + Shift + I або права кнопка миші).
12. Виберіть у контекстному меню команду Перевірити.
Відкриється панель інспектора коду.
13. Виберіть у рядку меню команду Панель перемикачів девайсів (Toggle device toolbar) (рис. 4, а).

Клацнувши цю команду, можна вибирати емуляцію девайса (рис. 4, б). Якщо девайса, який нас цікавить, у списку немає, то, клацнувши команду Edit, можна пошукати в додатковому меню (рис. 4, в).

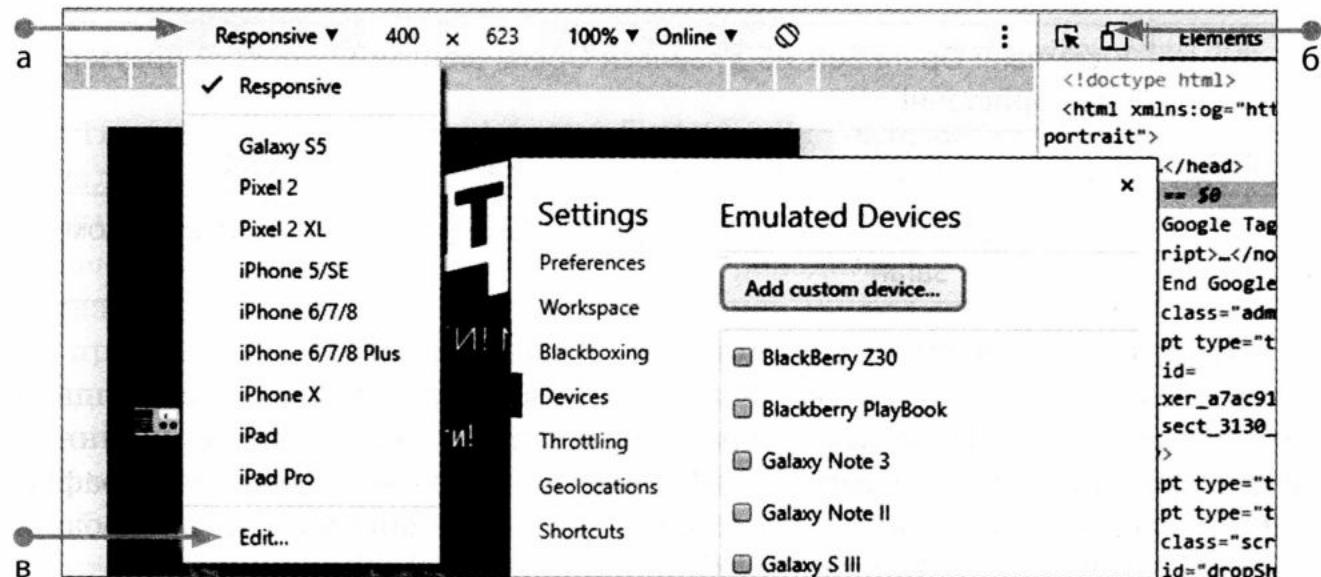


Рис. 4. Панель інспектора коду браузера Google Chrome:
а — панель перемикачів девайсів; б — вибір розміру екрана; в — додаткове меню

14. Проаналізуйте збережені скріншоти й запишіть отримані результати.

Зробіть висновки за результатами роботи.

2.5 Кросбраузерність

Кожен браузер має свою історію, свої версії, які, у свою чергу, розрізняються підтримкою Javascript, HTML і CSS. Хоча різні браузери в основному дотримуються загальних правил і стандартів, в деяких випадках алгоритми обробки HTML-кодів і каскадних таблиць CSS можуть бути різними. Це призводить до різного відображення одного і того самого елемента сайта в різних браузерах.

Як вже вказувалося раніше, найпопулярнішими браузерами сьогодні є Google Chrome, Safari, Mozilla Firefox, Opera, Internet Explorer. Ознайомимося з історією їх створення.

Рік	Опис
1994	З'явився один із перших вдалих браузерів Netscape Navigator, створений на основі первого браузера з графічною оболонкою NCSA Mosaic
1994–1996	Компанія Microsoft розробила браузер Internet Explorer, однак перші три версії широкого розповсюдження не отримали. Тривалий час браузери Netscape Navigator і Internet Explorer розвивалися паралельно, доки останній не захопив 95 % ринку. Далі в Internet Explorer застій (з 4.0 до 6.0 версії), а в Netscape Navigator, який був написаний на новому рушії Gecko, — відродження у версії 6.0. Оновлений Netscape Navigator не досяг колишніх вершин, та рушій Gecko у 2004 році послужив основою для створення сучасного браузера Mozilla Firefox одноіменної компанії
1996	У компанії Opera Soft AS з'явився Opera, швидкий і простий браузер у використанні
2003	Корпорація Apple випустила браузер Safari на рушії WebKit
2008	Корпорація Google випустила браузер Google Chrome на тому самому рушії, що й Safari

Розглянемо розвиток браузерів більш докладно.

Веббраузер Тіма Бернерса Лі World Wide Web дозволяв переглядати текстові сторінки. Перегляд зображень здійснювався в окремих вікнах і був з чорно-білим інтерфейсом. Перший браузер, який отримав графічний інтерфейс, тобто не тільки просто текст на чорному тлі, був розроблений у 1993 році і мав назву NCSA Mosaic (рис. 2.29).

Наступником Mosaic став браузер Netscape (рис. 2.30). Його розробники додавали в HTML нові теги, які робили зовнішній вигляд документа більш привабливим. Ці теги не були стандартизованими і працювали лише в Netscape. Та оскільки частка Netscape на той час становила понад 90 % від усіх наявних браузерів, це проблемою не було.



Рис. 2.29. Браузер NCSA Mosaic



Рис. 2.30. Браузер Netscape

Поява конкурента — Internet Explorer від Microsoft — практично розпочала відкриту війну між двома корпораціями за частку ринку, яка отримала назву «війна браузерів». Топ-менеджери Microsoft переманювали ключових співробітників Netscape та переходили до погроз, якщо це не вдавалося. Більш того, корпорація тиснула на великих клієнтів Netscape та провайдерів, погрожуючи, що перестане постачати їм програмне забезпечення, а для ІТ-бізнесу наприкінці 1990-х це означало крах (більш детально про війну браузерів можна дізнатись, переглянувши фільм *Download: The True Story of the Internet*). Ця війна призвела до серйозної проблеми, що полягала у відсутності єдиних стандартів відображення вебсторінок. Дійшло до того, що на сайтах вбудовувалися кнопки Best viewed in Netscape і Best viewed in Internet Explorer.

Найбільші відмінності виникали у підтримці JavaScript — мови сценаріїв, що додає інтерактивності документам. У результаті багато документів були «оптимізовані» для конкретного браузера й абсолютно не читалися в іншому.



Рис. 2.31. Браузер Mozilla

Mozilla — це внутрішнє ім'я браузера Netscape Navigator, що означає Mosaic Killer (вбивця Mosaic) (рис. 2.31). Назва здалася співробітникам фірми за надто зухвалою, пізніше так було названо нащадка Navigator — Mozilla Foundation. Браузер же отримав назву Phoenix на честь птаха фенікса, який згоряє, щоб відродитися з попелу. Згодом цю назву було змінено на Firebird (жар-птиця), а потім на Firefox (бо дві попередні назви вже використовувалися іншими розробниками).

Mozilla була не єдиним браузером, що кинув виклик Internet Explorer. На арену вийшла Opera — її перша версія, розроблена для корпоративного використання всередині норвезької телекомуникаційної компанії Telenor, з'явилася ще в далекому 1994 році. У 1996 році друга версія через тотальну війну між Netscape і Microsoft теж пройшла непоміченою, і лише третя версія почала завойовувати популярність.

У 2006 році на ринок браузерів зайшла ще й Apple зі своїм Safari, який базувався на рушії Webkit. У 2008 році світ побачив браузер від Google, розроблений на основі Chromium, який теж використовував рушії Webkit. Продукт, який перейняв кращі якості Firefox, Safari й Opera, захопив аудиторію (рис. 2.32).

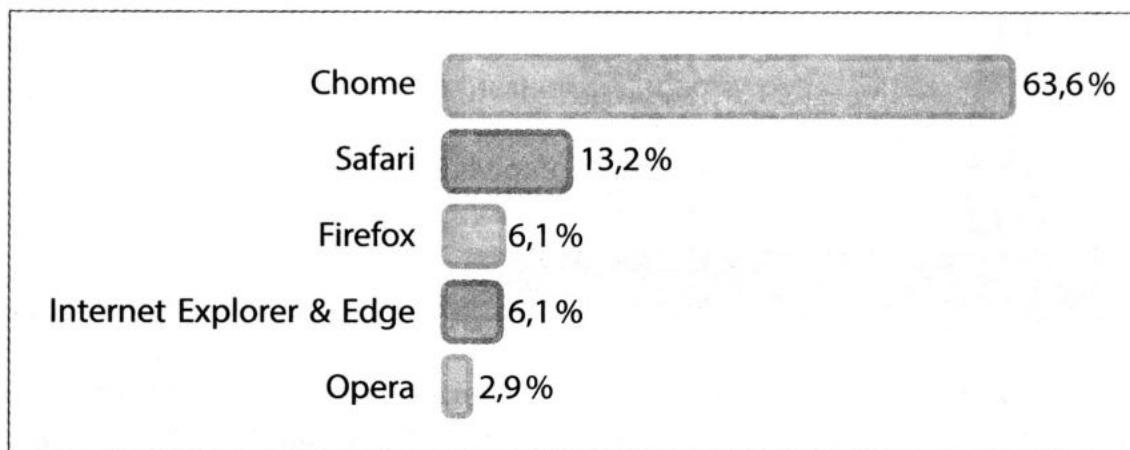


Рис. 2.32. Статистика використання браузерів

Сьогодні виробники браузерів почали активно займатися своєчасною підтримкою стандартів World Wide Web, HTML5 і CSS3, що, безсумнівно, позначилося на якості відображення вебсторіонок. Проте у деяких властивостях каскадних таблиць і трактуванні коду навіть зараз на різних рушіях зустрічаються суттєві розбіжності, які необхідно виправляти вручну.



Кросбраузерність — це правильна верстка сайта, за допомогою якої вебсторіонки однаково відображаються в різних браузерах. Реалізація відбувається за допомогою HTML і CSS, а також різноманітних хаків, в окремих випадках — JavaScript.



ЦІКАВІ ФАКТИ



Американський вебдизайнер Ерік Майєр запропонував CSS Reset у травні 2007 року. Майєр написав цілу низку популярних у світі веброзробки книжок, таких як *Cascading Style Sheets: The Definitive Guide*, *Eric Meyer on CSS*. Має власний сайт (<https://meyerweb.com>).

Для правильного відображення сайта одночасно в найбільш поширених браузерах, причому різноманітних версіях (від найбільш ранніх до найновіших), вебдизайнер обов'язково має дбати про кросбраузерність проекту сайта з першої секунди роботи над ним.

Найбільш поширений спосіб, який застосовується багатьма вебпрограмістами, це написання так званих хаків — наборів спеціальних селекторів або правил, які розуміє тільки якийсь певний браузер. Тобто якщо необхідно коректно відображати сайт, скажімо, у трьох браузерах, то потрібно написати по хаку для кожного браузера. Є ще один спосіб — просто використовувати при верстці HTML-коду ті елементи, які у всіх необхідних браузерах відображаються однаково.

CSS-хаки — уривки коду, що розуміються тільки одним певним браузером. Хаки — найбільш «брудний» спосіб виправлення помилок, що робить код неестетичним і недійсним, але робочим.

У кожного браузера є свої вбудовані експериментальні або нестандартні властивості, і щоб вони коректно працювали, використовують вендорні префікси. У назві недаремно використано слово «префікси», бо, як і в граматиці, вони є приставкою, тільки в даному випадку до властивості CSS.

Вендорні префікси:

- ◆ -webkit — для Google Chrome, Safari і iOs;
- ◆ -moz — для Mozilla;
- ◆ -o — для Opera;
- ◆ -ms — для Internet Explorer.

Вендорні префікси є ще однимrudиментом браузерних воєн, особливо браузерів WebKit, однак це більш «чистий» і чесний спосіб, ніж використання хаків.

Кожен браузер має власні властивості з вендорним префіксом. Так, наприклад, елемент border-radius у Mozilla Firefox представлений властивістю -moz-border-radius, а в Chrome і Safari — -webkit-border-radius. Такі властивості змінюють поведінку елемента тільки в певному браузері та ігноруються іншими платформами.

Таким чином, вендорні префікси — приставки до стилів CSS — мають змістовне навантаження лише для тих браузерів, до яких належать. Вони дозволяють браузеру сприймати нестандартні властивості, а також стилі, призначені для інших користувачьких клієнтів.

Можна скористатися плагіном Autoprefixer (<https://autoprefixer.github.io>) (рис. 2.33), який аналізує правила CSS, що існують на сайті, та додає необхідні вендорні префікси.

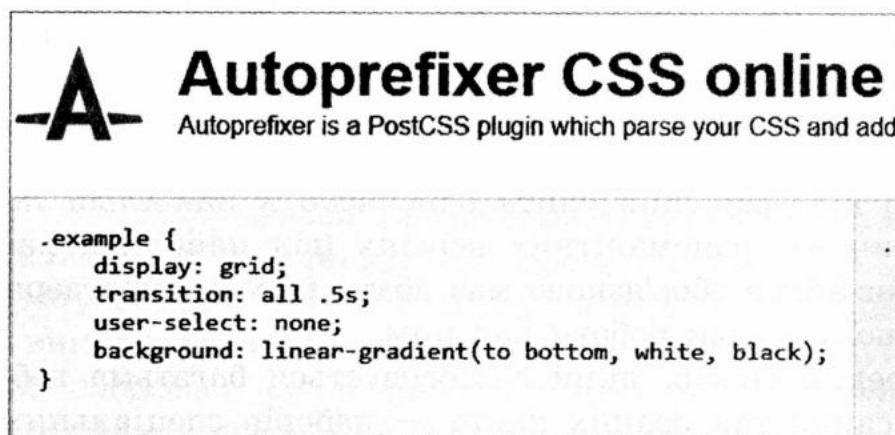


Рис. 2.33. Плагін створення вендерних префексів

Як відомо, кожен браузер за замовчуванням має певний набір базових стилів, які застосовує до сторінки. У різних браузерах ці правила трохи відрізняються. Щоб їх усунути та зробити за замовчуванням відображення сторінки у всіх браузерах однаковим, використовують спеціальні CSS-файли: `reset.css` або `normalize.css`.

Наразі вони є найпопулярнішими й повністю відповідають HTML5. Найпростішим способом використання файлів є завантаження їх із сайтів у теку CSS та підключення на сторінках перед власним файлом стилів.

Файл `reset.css` містить перелік усіх можливих HTML-тегів і скидає їх значення в нуль, тобто прибирає всі можливі відступи, робить шрифт однаковим у всіх тегах. Таким чином, всі заголовки й абзаци відображаються простим текстом, одним розміром і без відступів. Як наслідок, відбувається скидання стилів за замовчуванням у всіх браузерах. Спочатку на сторінці користувачу слід підключити файл `reset.css`, а потім — власний файл зі стилями `style.css`. У будь-якому браузері вся розмітка ґрунтуетиметься на тих стилях, які потрібно поставити в `style.css`.

Завантажити файл можна із сайта <https://cssreset.com/> (рис. 2.34).

`Normalize.css` — продукт глибокого дослідження відмінностей між початковими стилями браузера. Дослідження провів Ніколас Галахер, взявши за мету зберігати корисні налаштування; нормалізувати стилі для більшості HTML-елементів; коригувати помилки й основні невідповідності браузера; удосконалювати юзабіліті непомітними поліпшеннями; пояснювати код, використовуючи коментарі та детальну документацію. Наразі `normalize.css` використовується в Twitter Bootstrap, HTML5 Boilerplate, GOV.UK, Rdio, CSS Tricks і в багатьох інших фреймворках, інструментах і сайтах.

Скидати всі стилі не завжди доречно, саме тому існує ще один (дещо інший) інструмент — `normalize`. На відміну від попереднього інструменту `normalize.css`, він нормалізує стилі (тобто приводить до єдиного вигляду у всіх браузерах). Після його застосування базові стилі відображення заголовків, розмір шрифтів, відступи тощо уніфікуються і відображаються у всіх браузерах однаково.



The screenshot shows the homepage of CSS Reset. At the top, it says "2019's most popular CSS Reset scripts, all in one place". Below that is a note: "Click 'Get Code' to copy/paste the full or minified version, or check out the documentation." A list of scripts follows, each with a "Get The Code" button:

- Eric Meyer's "Reset CSS" 2.0 (107,633) Get The Code
- HTML5 Doctor CSS Reset (45,454) Get The Code
- Yahoo! (YUI 3) Reset CSS (13,205) Get The Code
- Universal Selector ** Reset (9,611) Get The Code
- Normalize.css 1.0 (9,224) Get The Code
- Free eBook: CSS Programming Cookbook (966) Free Download

At the bottom left, there are links for "What is a CSS Reset?" and "Which CSS Reset Should I Use?".

Рис. 2.34. Головна сторінка сайту CSS Reset

Завантажити файл можна із сайту <https://necolas.github.io/normalize.css>



Потрібно віддавати перевагу універсальним елементам — тим, які однаково працюють у більшості браузерів. Робота лише з ними зробить код коротким, чистим і зрозумілим.

Для досягнення правильного відображення сайта в різних браузерах слід використати вендорні префікси; підключити CSS-файл normalize.css або reset.css; намагатися використовувати елементи, що мають однакове відображення в усіх браузерах.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

- Що таке кросбраузерність?
- Які браузери ви знаєте?
- Що таке CSS-хаки, який їх недолік?
- Що таке вендорні префікси?
- Яким чином досягається кросбраузерність сайта?
- Знайдіть в інтернеті відомості про «війну браузерів» та підготуйте невеличку презентацію.



Дивіться презентацію «Кросбраузерна оптимізація сторінок сайту».



Практична робота № 6

ТЕМА. Використання спеціального CSS-файла для досягнення кросбаузерності

ЗАВДАННЯ: забезпечити однакове відображення сайта різними браузерами.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

1. Відкрийте сайт, створений на практичній роботі № 3, за допомогою всіх наявних на вашому комп'ютері браузерів.
2. Зробіть скріншоти головної сторінки сайта в кожному з цих браузерів.
3. В адресному рядку вашого браузера за замовчуванням наберіть команду:

```
https://necolas.github.io/normalize.css
```
4. Завантажте останню версію файла.
5. Збережіть файл у теці css як normalize.css.
6. Додайте посилання на цей файл у всіх HTML-сторінках вашого сайта за аналогією зі своїм файлом, але перед посиланням на style.css», як у наведеному зразку:

```
<link rel="stylesheet" type="text/css" href="css/normalize.css">
<link rel="stylesheet" type="text/css" href="css/style.css">
```
7. Збережіть сторінки.
8. Повторіть пункт 1 практичної роботи.
9. Порівняйте скріншоти з результатами вашої роботи.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Чи зміниться відображення сайта, якщо поміняти місцями лінки таблиць стилів розробленої вами та завантаженої з сайта вебсторінок?
2. Чим обумовлюється необхідність використання спеціального CSS-файла?

Зробіть висновок: як і чому змінилось (або не змінилось) відображення вашого сайта різними браузерами.

Розділ 3

ГРАФІКА ТА МУЛЬТИМЕДІА ДЛЯ ВЕБСЕРЕДОВИЩА

3.1 Графіка для вебсередовища

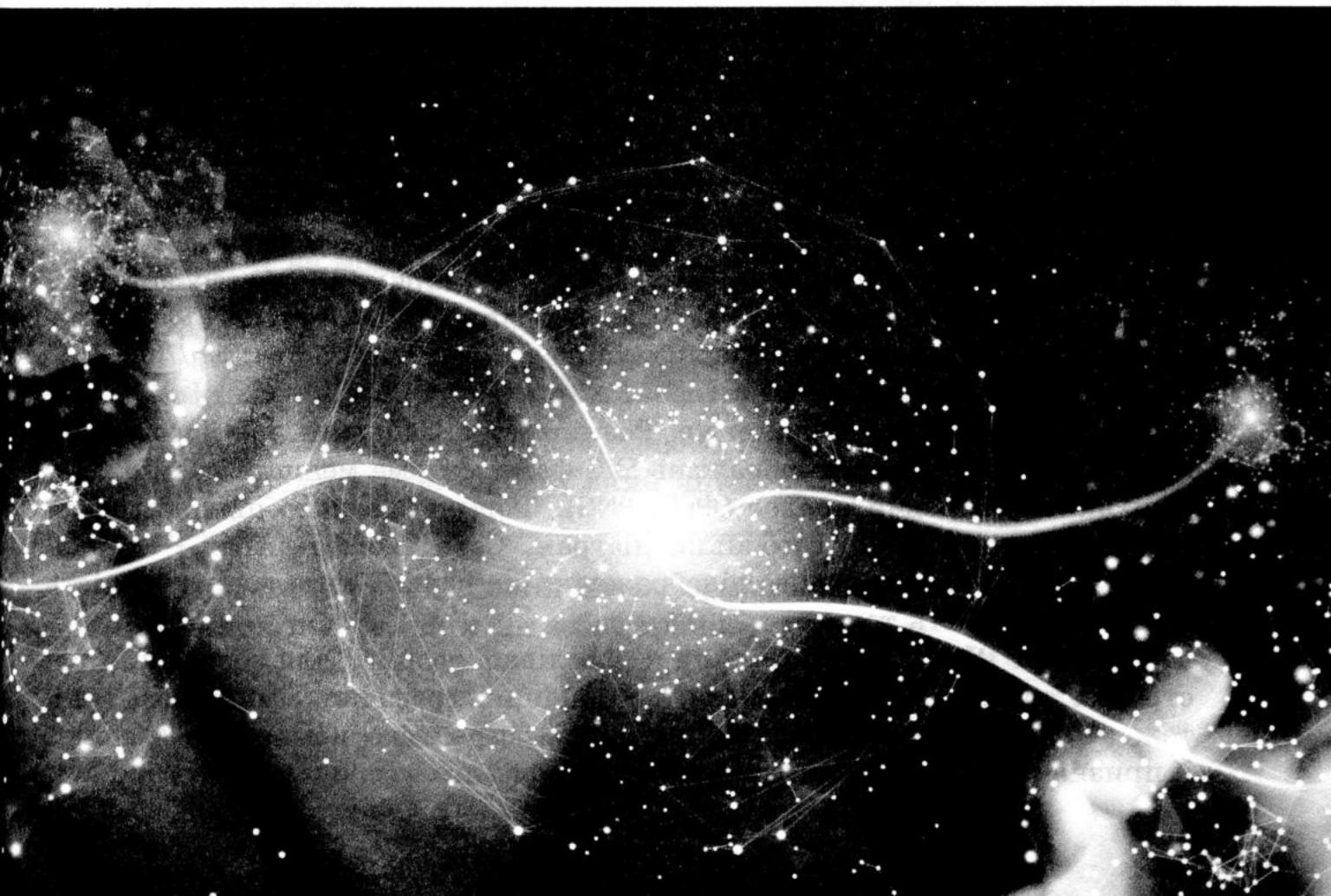
Практична робота №7

3.2 Анімаційні ефекти

Практична робота №8

3.3 Мультимедіа на вебсторінках

Практична робота №9



3.1

Графіка для вебсередовища

Фотографії, рисунки, фонові малюнки — все це візуальні елементи вебдизайну. Саме від них залежить зовнішній вигляд сайта, а також швидкість його завантаження. Загалом у вебдизайні використовують чотири основні формати графічних файлів, які вже знайомі вам із курсу 10 класу.

Формат JPEG (*Joint Photographic Experts Group*), або JPG, — це 16-бітовий формат растрових зображень, популярний для зберігання цифрових фотографій, які мають дрібні деталі та яскраві кольори. Завдяки тому, що зображення у форматі JPEG швидко завантажуються, його вибирає більшість вебдизайнерів для своїх сайтів. Проте JPEG не підтримує прозорість, і його не має сенсу використовувати для логотипів і піктограм.

Формат GIF (*Graphic Interchange Format*) — це формат, обмежений 256 кольорами, дуже ефективний для зберігання логотипів, піктограм, таблиць. Проте GIF абсолютно не підходить для цифрових фотографій із тисячами відтінків. На відміну від формату JPG, він підтримує прозорість зображення та дозволяє створювати різноманітну анімацію (див. § 3.2).

Формат PNG (*Portable Network Graphic*) — можна сказати, що це покращений JPEG, його зручно використовувати для простого й плоского графічного дизайну. PNG дозволяє працювати із прозорістю краще, ніж GIF, проте він не підтримує анімацію. Зазвичай цей формат використовують для публікації невеликих картинок, логотипів, іконок, діаграм, графічних елементів із прозорістю, фотографій без втрати якості.

Формат SVG (*Scalable Vector Graphics*) — найбільш поширений векторний формат, попит на який збільшується завдяки адаптивному дизайну у веброзробці. Все більше вебдизайнерів пристають до думки, що растрові зображення повинні використовуватися лише для фотографій. За свою суттю будь-який сайт — це інтерфейс, і всі інтерфейси мають бути векторними. Саме тому безумовну популярність набуває використання векторного формату SVG. Про особливості SVG та його використання можна дізнатися на сторінці <http://yoksel.github.io/about-svg/?full#8>.

Геометрична природа файлів SVG дозволяє легко адаптувати векторну графіку до потрібних параметрів. Розмір файла залежить від його складності, так, для простих зображень SVG-файл матиме менший розмір, ніж будь-який растровий аналог (JPEG, PNG). Формат ідеально підходить для зберігання піктограм, логотипів, діаграм. Серед його переваг — швидкість завантаження, чудове відображення на highDPI дисплеях, гнучкість і масштабованість тощо.

Формат BMP (*Bitmap Picture* — дослівно «формат для зберігання растрових зображень») — один із найстаріших графічних форматів, який теж можна використовувати у веб. BMP розроблено компанією Microsoft і призначено для зберігання великих файлів усередині ОС Windows.



Його особливість — не використовується жоден алгоритм стиснення, тому розмір файла зазвичай дуже великий.

Зазначимо, що ми говоримо насамперед про статичні зображення. Їх легко відобразити, використовуючи елемент ``. Тег `` використовують для розміщення фотографій, логотипів, графічних елементів інтерфейсу тощо.

Розглянемо його атрибути.

Елемент `` є стандартним тегом для додавання графічних елементів на вебсторінку. Цей тег вставляє зображення на сторінку в тому місці, де воно має з'явитися.

Тег `` є непарним.

- Обов'язковим атрибутом є `src` (англ. *source* — джерело), який містить шлях до зображення. Зазвичай зазначається або URL-адреса, або відносна адреса щодо місця розташування вебсторінки, яка містить посилання на зображення.

З огляду на стандарти розробки сайтів (див. § 2.3) рекомендується створювати окрему теку з назвою `images`, в якій зберігаються всі зображення, що містяться на вебсторінках сайта.

Тоді тег матиме такий вигляд:

```

```



Тег відображає зображення лише графічних форматів GIF, JPEG, PNG і SVG. В атрибуті обов'язково має вказуватися розширення.

- Іншим обов'язковим атрибутом є `alt` (*alternative*), у якому вказується альтернативний текст — опис зображення для тих випадків, коли користувачі не можуть побачити картинку.

Розглянемо приклад 1.

ПРИКЛАД 1

У випадку коли користувачі мають вади зору, використовується скрін-рідер (*screen-reader*), який читає описи зображень.

Атрибут `alt` має надати достатньо інформації користувачеві, щоб він склав уявлення про те, що є на зображенні:

```

```

Нагадаємо, що в стандарті HTML5 теги мають лише семантичний зміст, а функції форматування покладено на каскадні таблиці стилів.

Розглянемо CSS-властивості зображень. Розмір зображення задається двома параметрами: `width` — ширина зображення; `height` — висота зображення.

Якщо не задавати розміри зображення, то, по-перше, воно відобразиться на сторінці в реальному розмірі (рис. 3.1, *a*), по-друге, браузер потребуватиме часу на те, щоб дізнатися розміри і завантажити зображення. Лише після цього він повернеться до завантаження іншого вмісту документа, таким чином, виведення решти елементів затримається. Крім того, при маленькому розмірі зображення й довгому альтернативному тексті, ще до того як завантажиться графіка, тимчасово відбудеться зсув дизайну сайта. Адже довгий альтернативний текст буде займати стільки місця, скільки йому знадобиться.

Якщо зазначити розміри зображення (рис. 3.1, *в*), то браузер спочатку зарезервує місце під зображення, підготує макет документа, відобразить текст і лише потім завантажить зображення.

Слід пам'ятати, що зображення можуть бути квадратними або прямоугольними. Якщо ми використовуватимемо обидва параметри, браузер помістить зображення в прямоугольник виділеного розміру, навіть якщо його реальні ширини та висота більші (стисне) або менші (розтягне), як на рис. 3.1, *б*.

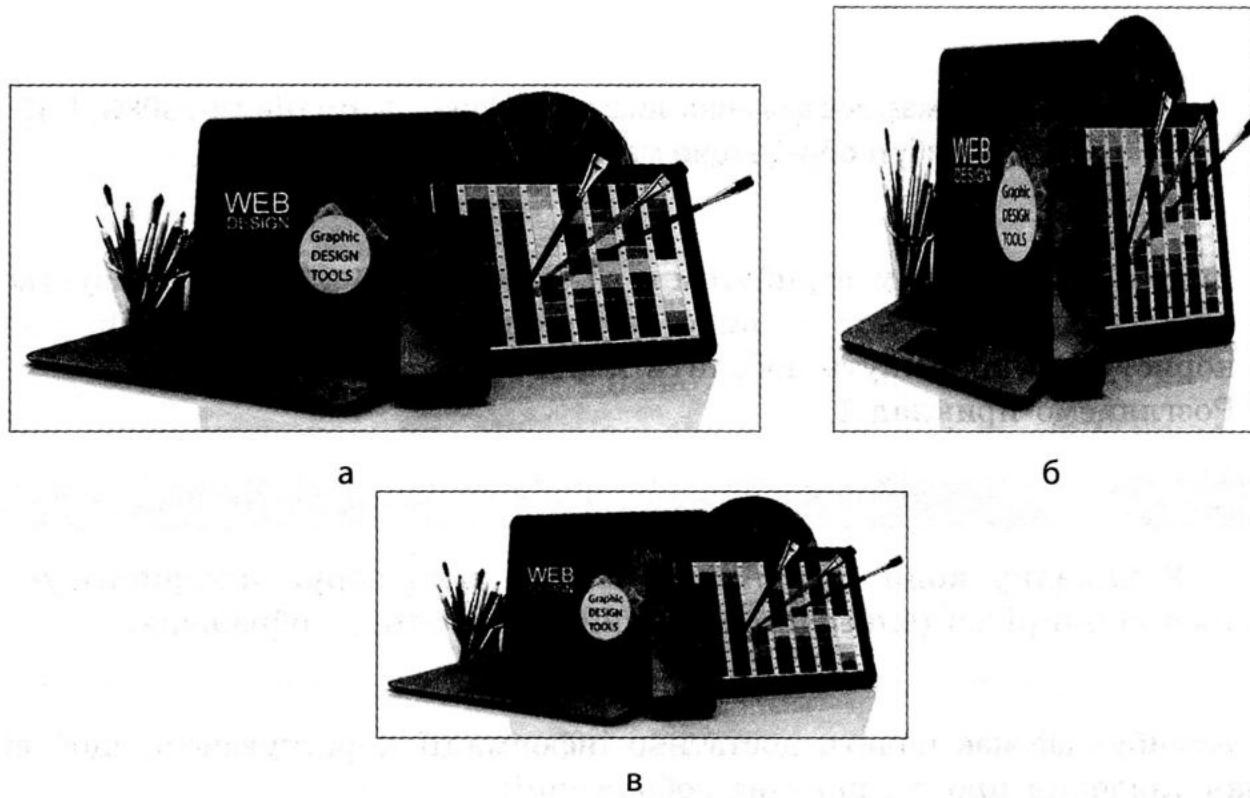


Рис. 3.1. Використання атрибутів розміру зображення: у реальному розмірі (*а*); вказані однакові ширина і висота, браузер відобразив прямоугольну картинку як квадратну, тобто стиснув її за ширину (*б*); вказана ширина, браузер автоматично обчислив висоту (*в*)



Тепер розглянемо приклад 2.

ПРИКЛАД 2

1. ``
2. ``
3. ``

Якщо ми задамо тільки один з атрибутів, то інший буде обчислюватися автоматично для збереження пропорцій малюнка.

Ширину й висоту зображення можна задавати як у пікселях (при цьому розмір картинки буде постійним незалежно від роздільності екрана), так і у відсотках — тоді розмір картинки залежатиме від роздільності екрана користувача.

Параметр `border` дозволяє створити рамку для зображення. Причому можна налаштовувати ширину, колір і стиль рамки за допомогою відповідних параметрів:

- ◆ `border-width`;
- ◆ `border-color`;
- ◆ `border-style`.

Додатково можна налаштовувати ці властивості дляожної сторони:

- ◆ `top` (верхня),
- ◆ `left` (ліва),
- ◆ `right` (права),
- ◆ `bottom` (нижня).

Наприклад,

`border-left-width`, `border-bottom-color`, `border-right-style`.

Параметри `padding` та `margin` задають відступи:

- ◆ `padding` — внутрішні відступи між зображенням та рамкою;
- ◆ `margin` — задає відступи зображення від усіх чотирьох країв.

Можна задавати відступ від конкретного краю (рис. 3.2).

Наприклад,

- ◆ `margin-top` — відступ від верхньої сторони;
- ◆ `margin-right` — відступ від правого боку;
- ◆ `margin-bottom` — відступ від нижньої сторони;
- ◆ `margin-left` — відступ від лівого боку.

Аналогічно можна задавати внутрішні відступи:

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

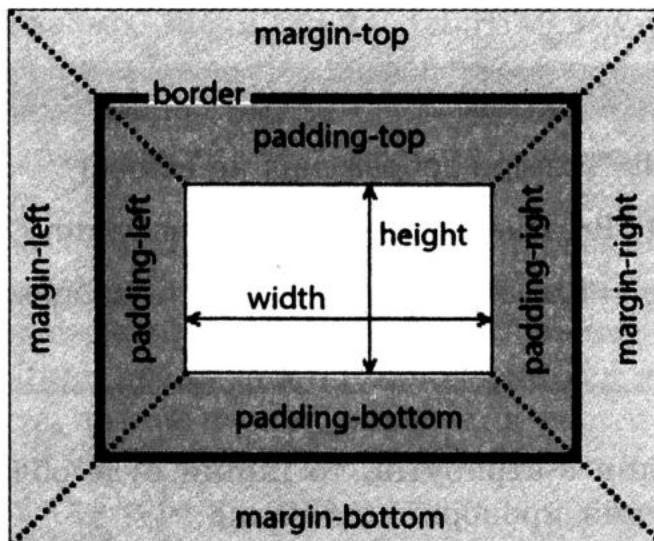


Рис. 3.2. Відступи в CSS

Якщо вказати `{margin: 50px;}`, ми задамо відступи в усіх напрямках: зверху, знизу, зліва і справа. Задати відступи для різних напрямків можна за допомогою скороченого оголошення.

Наприклад,

`margin: 10px 20px 50px 100px;`,

що еквівалентно записує

`margin-top: 10px;
margin-right: 20px;
margin-bottom: 50px;
margin-left: 100px;`



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Які графічні формати та якими командами мови розмітки відображаються?
2. Що таке screen-reader? Для чого він призначений?
3. Які обов'язкові атрибути має тег для відображення графічних об'єктів на вебсторінці?
4. Які стилі використовують для відображення графічного об'єкта на вебсторінці?
5. Обґрунтуйте використання атрибутів ширини та висоти зображення.
6. Який вид комп'ютерної графіки краще використовувати для фотографій?



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

1. Створіть сторінки, на яких відображатимуться графічні об'єкти, як на рис. 3.3.
2. Результат надішліть на електронну пошту вчителю.



Lore Ipsum



ФОТОГРАФІЯ

Lore ipsum dolor sit amet, consectetur adipiscing elit. Fusce id volutpat magna. Cras rhoncus, leo et sagittis sagittis, leo eti consectetur nulla, sit amet semper urna enim eget risus. Nullam quis auctor enim, pellentesque faucibus nunc. Maecenas laoreet lorem vitae velit aliquet rutrum. Nunc maximus volutpat mauris id ante aliquam. Quisque ac maximus diam, eget pretium leo. Nam consequat sed quam vitae dignissim.

Lore Ipsum



ФОТОГРАФІЯ

Lore ipsum dolor sit amet, consectetur adipiscing elit. Donec ac ornare ex. Nulla id sollicitudin metus, vel congue ex. Curabitur eu arcu leo. Integer ut elit metus. Phasellus consequat ac lacus vel tristique. Praesent ipsum augue, vestibulum id laoreet vel, imperdiet sed eros. Proin laicin, arcu consectetur sagittis malesuada, risus urna finibus justo, et consectetur justo leo non diam.

a

Lore Ipsum



ФОТОГРАФІЯ

Lore ipsum dolor sit amet, consectetur adipiscing elit. Fusce id volutpat magna. Cras rhoncus, leo et sagittis sagittis, leo eti consectetur nulla, sit amet semper urna enim eget risus. Nullam quis auctor enim, pellentesque faucibus nunc. Maecenas laoreet lorem vitae velit aliquet rutrum. Nunc maximus volutpat mauris id ante aliquam. Quisque ac maximus diam, eget pretium leo. Nam consequat sed quam vitae dignissim.

Lore Ipsum



ФОТОГРАФІЯ

Lore ipsum dolor sit amet, consectetur adipiscing elit. Donec ac ornare ex. Nulla id sollicitudin metus, vel congue ex. Curabitur eu arcu leo. Integer ut elit metus. Phasellus consequat ac lacus vel tristique. Praesent ipsum augue, vestibulum id laoreet vel, imperdiet sed eros. Proin laicin, arcu consectetur sagittis malesuada, risus urna finibus justo, et consectetur justo leo non diam.

b

Рис. 3.3. Сторінки до завдання 1



Дивіться презентацію «Графіка для вебсередовища».



Практична робота № 7

ТЕМА. Додавання статичних графічних об'єктів на сайт

ЗАВДАННЯ: створити сторінки сайту з графічними об'єктами.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

- Створіть у текці site теку images. Збережіть у ній будь-яку завантажену з інтернету фотографію з іменем 1.jpg.
- Завантажте у редактор коду файл info.html, створений у попередніх практичних роботах, і таблицю стилів style.css.
- Додайте збережену фотографію так, щоб вона обтікалася текстом зліва або справа.

4. У таблиці стилів створіть класи: .img, .left і .right, пропишіть їхні властивості (рис. 1).

```
.img{
    width: 200px;
}
.left{
    float: left;
}
.right{
    float: right;
}
```

Рис. 1. Властивості класів

5. У файлі info.html після другого й четвертого текстових абзаців поставте одинарний тег:

|

6. Після першого й третього текстових абзаців поставте тег:

|

7. Збережіть зміни у файлах і перегляньте результат.

8. Додайте на головну сторінку (index.html) три фотографії в одному рядку.

9. Під час виконання практичної роботи № 2 було створено три сторінки, а в меню подано чотири. Створіть четверту сторінку — gallery.html. На ній створіть галерею, де будуть відображатися фотографії.

Примітка. Існує багато способів створення галерей, ми розглянемо два з них.

10. Використайте 1-й спосіб — із уже відомих вам списків. У таблиці стилів пропишіть новий клас img-container із властивостями, як наведено на рис. 2.

11. Збережіть сторінку index.html як gallery.html. Після закриття контейнера класу <header> додайте код, наведений на рис. 3.

12. Збережіть змінені файли і перегляньте результат.

13. Використайте 2-й спосіб. Пропишіть у таблиці стилів новий клас. У файл стилів додайте контейнер inline-img із такими властивостями, як на рис. 4, а.

14. Додайте у файл із галереєю контейнер <div class="inline-img">.

15. Додайте в контейнер ваші 4 фотографії (рис. 4, б).

16. Закройте контейнер. Якщо потрібно створити декілька рядків із фотографіями, слід додати стільки контейнерів, скільки є рядків із фотографіями.



```
.img-container li{
    display: inline-block;
    width: 32%;
    padding: 10px;
}
.img-container{
    text-align: center;
    margin-top: 20px;
    width: 100%;
}
.img-container img{
    width: 80%;
    display: inline-block;
    transition: .25s;
}
.img-container ul li{
    list-style-type: none;
}
```

Рис. 2. Новий клас

```
<div id="main">
    <div class="img-container">
        <ul>
            <li></li>
            <li></li>
            <li></li>
            <li></li>
            <li></li>
            <li></li>
        </ul>
    </div>
</div>
```

Рис. 3. Код для додавання

```
inline-img{
    width: 250px;
    display: inline-block;
    border-radius: 15px;
    text-align: center;
}
```

Додаємо у файл *style.css*

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

Додаємо у файл *gallery.html*

a

б

Рис. 4. Властивості (a), додавання фотографій (б)

17. Перевірте галереї, створені різними способами, на адаптивність. Запишіть свої висновки й надішліть учителю.

Зробіть висновок за результатами роботи.

3.2 Анімаційні ефекти

Анімація є одним із трендів у дизайні вебінтерфейсів, вона вже давно стала невід'ємною частиною кожного сайта. Анімаційні ефекти можна застосовувати як до окремих об'єктів, так і до зображень.

Сучасні можливості анімації дозволяють зробити найрізноманітніші слайдери: на повний екран, із 3D-ефектами, адаптивні (які переглядаються з будь-яких мобільних пристрой) та ін.

Анімація — це і крихітні, ледь помітні індикатори завантаження, і сторінки, на яких вам наче показують фільм.

Важко назвати галузь вебдизайну, де б не використовувались анімаційні ефекти: від декоративних елементів, що просто прикрашають інтерфейс, до ефектів, які активно впливають на користувача.

Перший варіант (найпростіший), який ми розглянемо, — *анімовані об'єкти*. Це окремі зображення й об'єкти, які показуються користувачеві. Наприклад, курсор, що рухається, плаваюча кнопка «вгору», кнопки заклику до дії, зміна кольору тощо.

Другий варіант — *анімовані зображення* (наприклад, рекламні банери на сайті). Вони можуть вести на внутрішні сторінки, блог, перевідправляти відвідувача на інший сайт. Ще один приклад — слайд-шоу з фотографій.

Анімаційні ефекти можна створювати як виключно засобами каскадних таблиць стилів, так і за допомогою JavaScript.

CSS-анімація робить можливою анімацію переходів (*transitions*) з однієї конфігурації CSS-стилю до іншої. Анімація складається з двох компонентів, а саме: зі стилю, котрий описує CSS-анімацію, та набору ключових кадрів (*keyframes*), які задають початковий і кінцевий стани стилю анімації (також є можливість задання точок проміжного стану).

Розглянемо алгоритм створення CSS-анімації.

Крок 1	Починається з оголошення імені анімації в блоці <code>@keyframes</code> і визначення так званих кроків анімації, або ключових кадрів
Крок 2	Після оголошення відкривається фігурна дужка (у нашому прикладі виключно на CSS), у якій послідовно від 0 до 100 % прописуються властивості для кожного ключового кадру. Між цими значеннями можна вставляти скільки завгодно проміжних значень, наприклад, 50 %, 75 % або навіть 83 %
Крок 3	Наступною командою є використання CSS-властивості <code>animation</code>

Основні переваги CSS-анімації перед традиційними скриптовими техніками анімації:



- легка у використанні для простих анімацій та не потребує знання JavaScript;
- чудово функціонує навіть під час завантаження на систему (на відміну від анімації JavaScript).

Приклад простої анімації — плавне мерехтіння (зміна прозорості) — наведено на рис. 3.4.

```

622 .element{
623     animation: simpleAnimation 5s infinite;
624 }
625 @keyframes simpleAnimation {
626     0%{
627         opacity: 0;
628     }
629     25%{
630         opacity: 0.4;
631     }
632     50%{
633         opacity: 0.8;
634     }
635     75%{
636         opacity: 0.4;
637     }
638     100%{
639         opacity: 0;
640     }
641 }
642 }
```

Рис. 3.4. Код простої анімації

Розглянемо спочатку таблицю «Властивості animation», а потім — приклад.

Властивість `opacity` дозволяє зробити будь-який елемент вебсторінки частково або повністю прозорим. Вона змінює прозорість елементів, для яких встановлено фонове зображення або задано тло за допомогою кольору чи градієнта. Якщо елементи містять інші елементи, вони також змінять прозорість. Властивість `opacity` набуває значення в діапазоні від 0 (повністю прозорий) до 1 (непрозорий).

Таблиця. Властивості `animation`

Анімація	Опис
<code>animation-delay</code>	Змінює час затримки між моментом завантаження елемента та початком анімаційної послідовності
<code>animation-direction</code>	Визначає зміну напрямку анімації та його чергування залежно від кількості проходів анімації, може задавати повернення в початковий стан і починати прохід заново
<code>animation-duration</code>	Визначає тривалість циклу анімації
<code>animation-iteration-count</code>	Визначає кількість проходів (повторів) анімації; можна також обрати значення <code>infinite</code> для нескінченного повтору анімації

Закінчення таблиці

Анімація	Опис
animation-name	Задає ім'я для анімації @keyframes через at-правило, яке описує анімаційні ключові кадри
animation-play-state	Дозволяє призупиняти й відновлювати анімацію
animation-timing-function	Задає конфігурацію таймінгу анімації, інакше кажучи, як саме анімація робитиме прохід через ключові кадри, це можливо завдяки кривим прискорення
animation-fill-mode	Визначає, які значення будуть застосовані для анімації перед початком і після її закінчення

Тепер розглянемо приклад.

ПРИКЛАД

Зазвичай розробники не пишуть про всі ці властивості окремо, а використовують короткий запис такої структури:

- animation: (1. animation-name — назва)
- (2. animation-duration — тривалість)
- (3. animation-timing-function — динаміка руху)
- (4. animation-delay — пауза перед стартом)
- (5. animation-iteration-count — кількість виконань)
- (6. animation-direction — напрямок).

Animation-iteration-count може набувати такого значення: будь-яке додатне число, що вказує кількість повторень, або infinite — нескінченну кількість разів, як у нашому прикладі.

У нашему прикладі звичайна анімація 5 секунд нескінченна.

Крім того, існує не менше сотні різноманітних плагінів та бібліотек для створення анімацій, розглянемо декілька з них.

Animate.css — це фундаментальна бібліотека анімацій, сумісних з усіма браузерами та відповідних для безлічі завдань.

Animate.css містить все, від класичних підскакувань до останніх новинок і унікальних ефектів, і здатна задоволити потреби практично будь-якого проекту.

Anime.js — це вражаючий набір функцій, які дозволяють пов'язувати безліч анімацій, синхронізувати етапи, малювати лінії, змінювати форму об'єктів, створювати власні анімації тощо.

CSS-Animate — це простий майданчик для написання робочого коду для будь-якої анімації.

Достатньо задати ім'я, клас, властивості анімації і фрейму, і можна керувати часовою послідовністю і додавати маркери. Інакше кажучи, потрібно налаштувати все необхідне для створення стандартної анімації, заснованої на ключовому кадрі, як зображене на рис. 3.5.



The screenshot shows the main interface of the CSS Animate website. At the top, there's a navigation bar with links for Home, Quick help ?, About, Contact, Demos, and Login. Below the navigation is a large central area divided into several sections:

- Timeline:** A timeline bar at the top left with a playhead at 0.00s. It includes buttons for back, forward, and reset.
- Code:** A code editor window containing CSS code for an animation named "element-animation". The code uses the animation property with animationFrames linear 4s, setting iteration-count to 1 and transform-origin to 50% 50%.
- Keyframes:** A panel on the right showing animation properties like Duration (4 sec), Repeat (1 times, Infinite repeat), Timing function (linear), and Rotation point. It also lists frame properties for Left, Top, Rotation, Scale X, Scale Y, Skew X, Skew Y, and Opacity, all set to their default values.
- Other controls:** Buttons for Examples, Reset, and Copy code, along with checkboxes for Standard, Chrome/Safari, Opera, Firefox, and IE.
- Footer:** Includes a copyright notice: © 2013 - 2019 All Rights Reserved.

Рис. 3.5. Головна сторінка CSS-Animate



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Наведіть приклади анімації на вебсторінці.
2. Якими засобами можна створювати анімації?
3. Опишіть послідовність команд у CSS для створення анімації.
4. Перегляньте відео на YouTube. Спробуйте створити просту анімацію за інструкцією.
5. Дослідіть запропоновані плагіни, створіть засобами табличного процесора порівняльну таблицю характеристик.



Дивіться презентацію «Анімаційні ефекти».

Практична робота №8

ТЕМА. Використання анімаційних ефектів на сторінках сайта

ЗАВДАННЯ: створити найпростіші анімаційні ефекти на сайті.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

Примітка. Пригадаємо основні властивості блоку анімації CSS, наведені в таблиці.

Назва властивості	Опис	Яких значень може набувати
animation-name:	Задає ім'я для анімації @keyframes за правилом, яке описує анімаційні ключові кадри	у нашому випадку spin
animation-duration:	Визначає тривалість циклу анімації	Вказується в секундах, від'ємних значень не існує, в нашому випадку 4s (4 сек)
animation-timing-function:	Задає конфігурацію таймінгу анімації, інакше кажучи, визначає, як саме анімація робитиме прохід через ключові кадри	Linear — лінійна функція, анімація відбувається рівномірно протягом усього часу, без коливань у швидкості; ease — анімація починається повільно, швидко розганяється і сповільнюється в кінці; ease-in — анімація починається повільно й плавно прискорюється в кінці; ease-out — анімація починається швидко й плавно сповільнюється в кінці; ease-in-out — анімація повільно починається і повільно закінчується
animation-iteration-count	Визначає кількість повторів анімації	Вказується ціле число, infinite — нескінченне повторювання
animation-direction	Визначає зміну напрямку анімації та його чергування залежно від кількості проходів анімації	normal — всі повтори анімації відтворюються так, як зазначено, значення за замовчуванням; reverse — усі повтори анімації відтворюються в напрямку, зворотному тому, як вони були визначені; alternate — кожне непарне повторення циклу анімації відтворюється у визначеному напрямку, кожне парне повторення — у зворотному напрямку; alternate-reverse — кожне непарне повторення циклу анімації відтворюється у зворотному напрямку, кожне парне повторення — у визначеному напрямку

Тренувальна вправа

- Створіть теку page_training.
- Створіть у редакторі коду сторінку index.html, у якій оголосіть два блоки `<div>` з іменами класів `one` і `two` (рис. 1). Збережіть у теці `page_training`.

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8">
</head>
<body>
    <div class="two"></div>
</body>
</html>
```

Рис. 1. Блоки з іменами класів

- Створіть у тій самій теці `page_training` таблицю стилів `style.css` і запишіть властивості класів `one` і `two` (рис. 2). Збережіть і перегляньте результат.
- Додайте до таблиці опис класу `one` при наведенні миши (рис. 3). Збережіть і перегляньте результат. Запишіть зміни, які ви спостерігаєте, у зошит.

```
.one{
    height: 300px;
    width: 300px;
    background: #820202;
    transition-duration: .5s;
    transform: rotate(0);
}
.two{
    width: 100px;
    height: 100px;
    background: grey;
}
```

Рис. 2. Властивість класів

```
.one:hover{
    transform: rotate(90deg);
    transition-duration: 1s;
    background: blue;
    border-radius: 50%;
}
```

Рис. 3. Опис класу

- Внесення змін у сайт, розроблений під час попередніх практичних робіт
- Відкрийте редактором коду головну сторінку (`index.html`). У блоці хедера додайте блок `div` з іменем класу `square`:

```
<div class="square"></div>
```



2. Запишіть у таблиці стилів властивості цього класу й налаштуйте анімацію (рис. 4).

```
.square{  
    width: 90px;  
    height: 90px;  
    margin: 30px auto;  
    background-color: #123456;  
    animation-name: spin;  
    animation-duration: 4s;  
    animation-timing-function: linear;  
    animation-iteration-count: infinite;  
    animation-direction: alternate;  
}  
@keyframes spin{  
    0%{transform: rotate(0deg);}  
    25%{background-color: yellow;}  
    100%{transform: rotate(360deg);}  
}
```

Рис. 4. Властивості класу

3. Збережіть зміни. Відкрийте сторінку браузером.
4. Спробуйте внести свої корективи в запропоновану анімацію: поміняти `animation-direction`, `animation-iteration-count` тощо. Як це відобразилося на відтворенні анімації? Результат запишіть у зошит.
- Налаштування анімації фотографій у галереї для збільшення при наведенні миши
1. Відкрийте в редакторі коду файл `style.css`. Додайте в клас `inline-img img` код (рис. 5) і створіть новий псевдоклас (рис. 6).

```
transform: scale(0.95);  
transition: .3s;  
max-width: 35%;
```

Рис. 5. Код для додавання

```
inline-img img:hover{  
    transform: scale(1);  
}
```

Рис. 6. Команда псевдокласу

2. Збережіть зміни. Відкрийте в браузері файл і перегляньте результат.

Поміркуйте, що необхідно змінити у таблиці стилів для збільшення розміру фотографій на інших сторінках.

Зробіть висновок: для яких елементів можна використовувати анімаційні ефекти.

3.3

Мультимедіа на вебсторінках

Технології опрацювання мультимедіа зараз є одним із найперспективніших і найпопулярніших напрямків. Мета — створення продукту, який передає інформацію шляхом упровадження та використання нових технологій, набору зображень, текстів і даних, що супроводжуються звуком, відео, анімацією й іншими візуальними ефектами.



Мультимедіа — комп'ютеризована технологія, що об'єднує роботу зі всіма джерелами даних, засіб подання різних видів інформації у цифровому вигляді.

Ознайомимося з тим, які об'єкти належать до мультимедіа (рис. 3.6). У попередніх параграфах розглядалося, як вставляти графічні й текстові об'єкти у вебсторінку. Розглянемо, як працювати з аудіо- та відеооб'єктами.



Рис 3.6. Класифікація об'єктів мультимедіа

Слід зазначити, що всі дані, надіслані мережею, позначаються певними назвами, які однозначно вказують їх тип, — так званими MIME (*Multipurpose Internet Mail Extensions*, багатоцільові розширення пошти інтернету).

Тип MIME надає даним та сама програма, що їх надсилає, наприклад, вебсервер. А браузер (тобто програма, яка їх отримує) визначає за типом MIME, чи підтримувати цей тип даних, і якщо так, що саме з ними робити.

Тепер розглянемо приклад.

ПРИКЛАД

- ◆ Вебсторінка має тип MIME text/html.
- ◆ Графічне зображення формату GIF має тип MIME image/gif.
- ◆ Свої типи MIME мають і мультимедійні файли.

Звукові файли мають розширення .wav, .au, .aif та ін. Фільми у форматі QuickTime мають розширення .qt або .mov, відео від Microsoft (*Microsoft Video for Windows*) — .avi (*Audio Video Interface*). Відеофайли (.mpg або .mpeg) у форматі mpeg зазвичай мають великий розмір, забезпечують високу якість відео, формат mpeg-4 дуже часто використовується при програванні відеофайлів online.

Сучасні браузери працюють із кількома форматами мультимедійних файлів із десятків наявних сьогодні. Не слід забувати, що різні браузери підтримують різні формати файлів.

У таблиці наведено типи MIME-форматів мультимедійних файлів, які підтримуються браузерами:

Формат файлів	Тип MIME	Браузери, якими підтримується
MPEG4	video/mp4	Chrome, Safari (не підтримується Firefox, Opera)
OGG, OGA, OGV	audio/ogg (для аудіофайлів) video/ogg (для відеофайлів)	Chrome, Firefox, Opera
MP4	video/mp4	Chrome, Safari
WEBM	video/webm	Chrome, Firefox, Opera

Відео у форматі avi на сайті засобами HTML5 не відтворюється. Його слід конвертувати у формати, які підтримуються браузерами. Можна скористатися онлайн-конверторами на кшталт VIDEO-CONVERTER (<https://convert-video-online.com/>).

HTML5-відео — стандарт для розміщення мультимедійних файлів у мережі з оригінальним програмним інтерфейсом без залучення додаткових модулів.

За допомогою елемента `<video>` з'явилася можливість додавати відео-вміст на вебсторінки, а також стилізувати зовнішній вигляд відеоплеєра за допомогою CSS-стилів.

Для відображення на сторінці відеозапису необхідно використовувати тег `<video>`, що включає тег `<source>`, який має обов'язковий атрибут `src`, котрий визначає адресу відео.

Атрибут	Опис
<code>autoplay</code>	Автоматичне відтворення відеофайла відразу після завантаження сторінки
<code>controls</code>	Вказівка браузеру, що потрібно відобразити базові елементи управління відтворенням (відтворення, пауза, гучність)
<code>height</code>	Задання висоти вікна для відображення відеоданих, можливі значення: px або %
<code>loop</code>	Циклічне відтворення відеофайла



Закінчення таблиці

Атрибут	Опис
muted	Вимикання звуку під час відтворення відеозапису
poster	URL-файл зображення, яке відображатиметься під час завантаження відеофайла або доти, поки користувач не натисне на кнопку PLAY. Якщо атрибут не задано, буде відображатися перший кадр відеофайла
preload	Атрибут, який відповідає за попереднє завантаження відеоконтенту. Не є обов'язковим, деякі браузери ігнорують його
auto-	Браузер завантажує відеофайл повністю, щоб він був доступний, коли почнеться відтворення
metadata	Браузер завантажує першу невелику частину відеофайла, щоб визначити його основні характеристики
none-	Відсутність автоматичного завантаження відеофайла
src	Містить абсолютну або відносну URL-адресу відеофайла
width	Задає ширину вікна для відображення відеоданих, можливі значення: px або %

Згадаємо, що перед використанням HTML5-елементів необхідно покроково зробити такі дії.

Крок 1	Указати правильний доктайп: <!DOCTYPE html>
Крок 2	<p>У стилях CSS позначаємо тип елемента HTML <video> як блочний новий video</p> <pre>{ display: block; }</pre>
Крок 3	<p>Для досягнення кросбраузерності доцільно перераховувати в <source> усі формати, починаючи з більш пріоритетного, та зазначати тип MIME для кожного відеофайла.</p> <p>З огляду на стандарти розробки сайтів (для дотримання правил створення структури сайта) варто в теці site створити додаткову теку video (audio), де розмістити відконвертовані відео- чи аудіофайли.</p> <p>В елементі <video> краще використовувати атрибут controls, який відповідає за відображення елементів керування плеєром.</p> <pre><video controls> <source src="video/movie.mp4" type="video/mp4"> <source src="video/movie.webm" type="video/webm"> <source src="video/movie.ogv" type="video/ogg"> </video></pre>

До появи стандарту HTML5-відео використовувався елемент `<embed>`, який визначає контейнер для зовнішнього застосування або інтерактивного вмісту (іншими словами, плагіна).

Більшість браузерів протягом довгого часу підтримували цей елемент. Проте даний тег не був включений у специфікацію HTML4, його додали в специфікацію HTML5.

Тег `<embed>` використовується досить рідко. Його призначення — вбудовування об'єктів на вебсторінку, наприклад флеш-ролики або флеш-ігри.

?

ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Дайте означення мультимедіа.
2. Наведіть приклади об'єктів мультимедіа.
3. Що таке формат MIME?
4. Поясніть необхідність конвертації відеофайлів.
5. Які теги для яких форматів використовують?

✓

ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

1. Додайте на першу сторінку завдання із § 3.1 два відеофайли (можна однакові), щоб вони відображалися, як наведено на рис. 3.7. Результат надішліть учителю.

Підказка: використовуйте контейнер `<div>` зі стилем `{display: inline-block;}`.

2. Додайте на другу сторінку завдання із § 3.1 після графічного об'єкта аудіофайл. Результат надішліть учителю.
3. Використовуючи тег `<embed>`, додайте будь-який флеш-ролик на третю сторінку завдання із § 3.1 (як допомогу перегляньте відео https://www.youtube.com/watch?v=i_l4X2K6PbE).

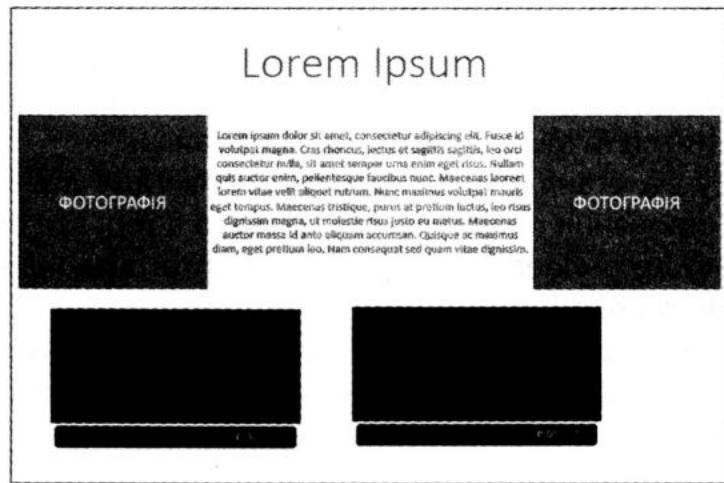


Рис. 3.7. Відображення відеофайлів



Дивіться презентацію «Мультимедія на вебсторінках».



Практична робота №9

ТЕМА. Додавання мультимедійних елементів на сторінку сайта

ЗАВДАННЯ: додати на сторінку сайта відеофрагменти з можливістю відтворення.

ОБЛАДНАННЯ: будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп'ютером дотримуйтесь правил безпеки.

- Створіть у текці site теку video. Додайте в неї вибране відео з назвою movie із розширенням .mp4.
- Відкрийте в редакторі коду файл gallery.htm.
- Для вставлення відео скористайтесь контейнером <div>.
- Створіть два класи: multimedia і multimedia-video.
- Контейнер <div class="multimedia"> буде містити всі контейнери класу multimedia-video.
- Уведіть у відкритому файлі перед оголошенням блока футера код, наведений на рис. 1:

```
<div class="multimedia"></div>
```

Рис. 1. Розміщення на одному рядку

- Розмістіть три відеофрагменти на одному рядку у файлі. Для цього введіть заданий код тричі (рис. 2):

```
<div class="multimedia">
    <video controls>
        <source src="video/movie.mp4" type="video/mp4">
    </video>
</div>
```

Рис. 2. Форматування

- Закрийте зовнішній <div>.
- Відкрийте редактором коду таблицю стилів і додайте форматування створених класів multimedia і multimedia-video (рис. 3).

```
.multimedia{  
    overflow: auto;  
}  
.multimedia-video{  
    display: inline-block;  
    float: left;  
    padding-left: 5px;  
    padding-right: 5px;  
    width: 33.33%;  
    box-sizing: border-box;  
}
```

Рис. 3. Форматування

10. Збережіть зміни та відкрийте файл у браузері.
11. Внесіть необхідні зміни на сторінку та в таблицю стилів для відображення двох і чотирьох відеофрагментів.
12. Перегляньте результат браузером.
13. Запишіть, що саме необхідно змінити при зміні кількості фрагментів.

Зробіть висновок: які дії необхідно виконати, щоб отримати можливість відтворення відеофрагментів на сайті та для зміни кількості відеофрагментів.

Розділ 4

ВЕБПРОГРАМУВАННЯ

4.1 Об'єктна модель документа

4.2 Вебпрограмування та інтерактивні сторінки

Практична робота № 10

4.3 Хостинг сайта

Практична робота № 11

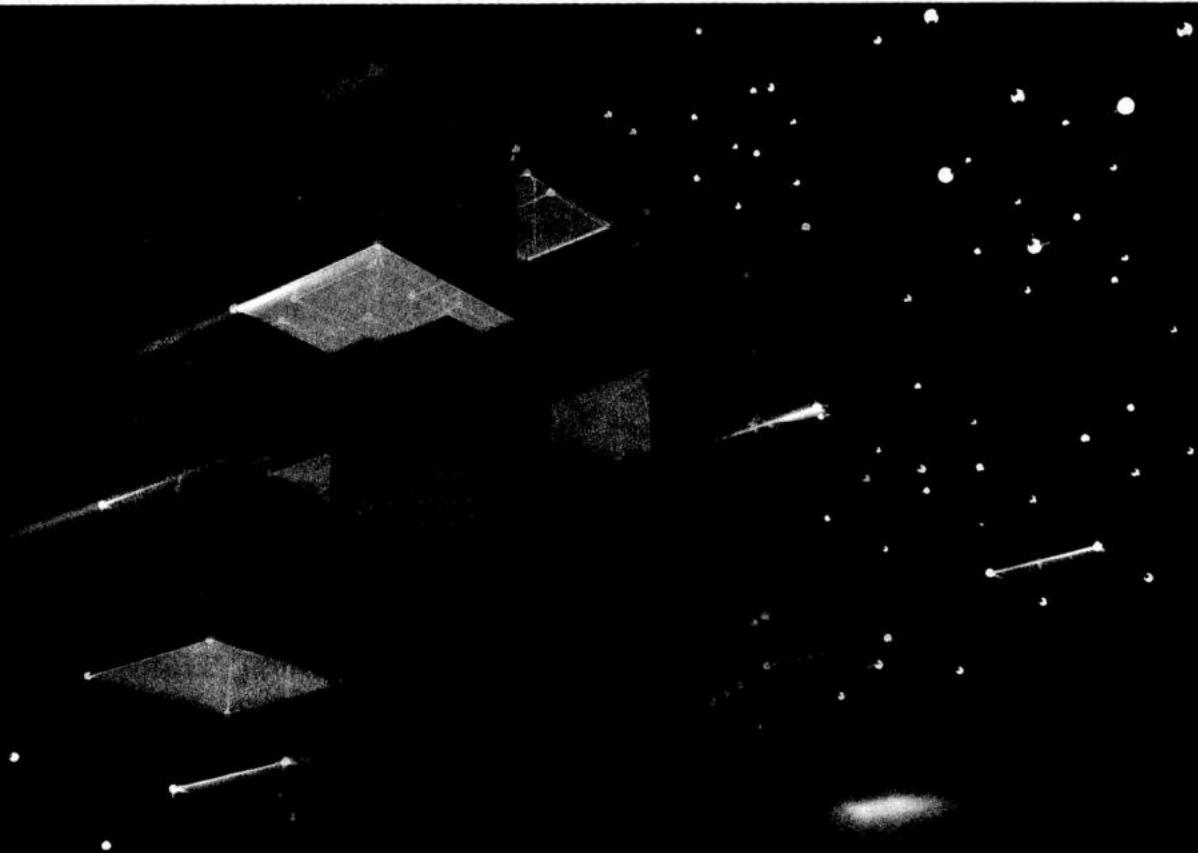
4.4 Вебсервер та бази даних

4.5 Взаємодія «клієнт — сервер»

4.6 Валідація сайта та збереження даних форм

Практична робота № 12

4.7 Прикладний програмний інтерфейс



4.1 Об'єктна модель документа

Об'єктна модель документа (англ. *Document Object Model*, DOM) — це програмний інтерфейс (API) для HTML.

DOM є стандартом, запропонованим вебконсорціумом W3C (*World Wide Web Consortium* — Консорціум Всесвітньої павутини), і регламентує спосіб подання вмісту документа (зокрема вебсторінки) у вигляді набору об'єктів.

DOM надає структуроване уявлення про документ та уможливлює доступ до цієї структури програмам, які можуть змінювати вміст, стиль і структуру документа. Подання DOM складається із структурованої групи вузлів і об'єктів, які мають властивості і методи. Власне, DOM з'єднує вебсторінку з мовами опису сценаріїв або мовами програмування.

Вебсторінка — це документ, який може бути поданий як у вікні браузера, так і в самому HTML-коді. У будь-якому випадку це один і той самий документ. DOM надає інший спосіб подання, зберігання й керування цього документа. Він повністю підтримує об'єктно-орієнтоване уявлення вебсторінки, роблячи можливим її зміну за допомогою мови опису сценаріїв на кшталт JavaScript.

Усі властивості, методи і події, доступні для керування й створення нових сторінок, організовані у вигляді об'єктів. DOM подає HTML-теги у вигляді об'єктів із властивостями і методами. У кожного HTML-тега (об'єкта) на HTML-сторінці, завдяки DOM, є своя унікальна адреса. Отримуючи доступ за цією адресою, JavaScript може керувати HTML-тегом.

Відкриваючи HTML-сторінку, браузер створює на основі її тегів структуру DOM, де кожен HTML-тег постає у вигляді об'єкта зі своєю унікальною адресою. Після аналізу структурованого документа будується його подання у вигляді дерева (рис. 4.1, 4.2).

```
<html>
<head>
    <title>My title</title>
</head>

<body>
    <h1>My header</h1>
    <a href="#">My link</a>
</body>
</html>
```

Рис. 4.1. DOM

Дерево в моделі DOM складається із множини зв'язних вузлів (Node) різних типів. Все, що є в HTML, знаходиться і в DOM. Навіть директива `<!DOCTYPE ...>`, яку ми ставимо на початку HTML, теж є DOM-вузлом і знаходиться в дереві DOM безпосередньо перед `<html>`.

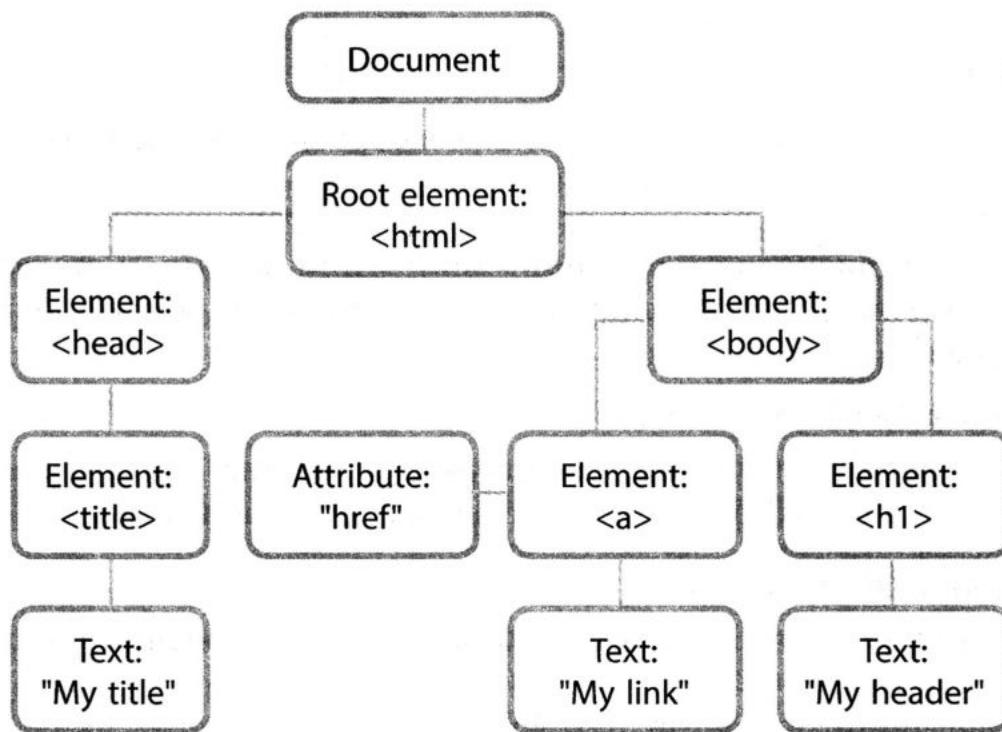


Рис. 4.2. Дерево вузлів DOM

DOM не є мовою програмування, але без нього JavaScript не мав би жодної моделі або уявлення про вебсторінку, HTML-документ, його елементи. Спочатку JavaScript і DOM були тісно пов'язані, але згодом розвинулися в різні сутності. Вміст сторінки зберігається в DOM і може бути доступним і змінюватися з використанням JavaScript.



Таким чином, можна сказати, що DOM — це вебтехнологія, що дозволяє керувати HTML-тегами сторінки через мову JavaScript.

Зазвичай розрізняють вузли декількох типів.

Вузол	Опис
Документ (Document)	Корінь дерева, представляє цілий документ, інакше кажучи, точка входу в DOM
Фрагмент документа (DocumentFragment)	Вузол, який є коренем піддерева основного документа
Елемент (Element)	Представляє окремий елемент HTML, можна сказати, що це основні будівельні блоки. Атрибут (Attr) представляє атрибут елемента
Текст (Text)	Представляє текстові дані, які містяться в елементі або атрибуті

Вузол	Опис
Коментарі	Іноді можна включити інформацію, яка не буде показана, проте є доступною з JS

Стандартом визначаються деякі інші типи вузлів у моделі документа. Вузли деяких типів можуть мати гілки, інші можуть бути лише листям дерева.

Створення DOM-елементів:

- ◆ `document.createElement(tag);` — створює новий тег.
- ◆ `document.createTextNode(text);` — створює текстовий вузол.

Додавання DOM-елемента:

- ◆ `parentElem.appendChild(elem);` — додає елемент у кінець дочірніх елементів;
- ◆ `parentElem.insertBefore(elem, nextSibling);` — вказує, перед яким із дочірніх елементів додати новий вузол.

Видалення DOM-елементів:

- ◆ `parentElem.removeChild(elem)` — видаляє конкретний елемент зі списку дітей батьківського елемента;
- ◆ `parentElem.replaceChild(newElem, elem)` — видаляє вказаний елемент і заміняє його новим.

DOM — стандарт, що розвивається і поділяється на три рівні. Перший рівень є першою версією стандарту і поки що єдиною закінченою. Він складається з двох розділів: перший є ядром і визначає принципи маніпуляції зі структурою документа (генерація і навігація), а другий присвячений поданням у DOM елементів HTML, що визначаються одноіменними тегами. Другий і третій рівні описують модель подій, доповнюють таблиці стилів, проходи по структурі.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке об'єктна модель документа?
2. Яка організація і чому запропонувала стандарт DOM?
3. Як можна подати вебсторінку?
4. У чому перевага використання DOM?
5. Назвіть типи вузлів.
6. Як саме можна працювати з вебдокументом завдяки DOM?



Дивіться презентацію «Об'єктна модель документа (DOM)».

4.2

Вебпрограмування та інтерактивні сторінки

Історія створення JavaScript починається в 1995 році, у самий розпал війни Netscape і Microsoft (див. про «війну браузерів» у § 2.5). Це був час, коли анімації, взаємодія з користувачами та інші види інтерактивності мали стати невід'ємною частиною інтернету майбутнього.

Веб зажадав легкої скриптової мови (мови сценаріїв), спроможної працювати з DOM (див. § 4.1), який ще не було стандартизовано.

**ЦІКАВІ ФАКТИ**

Її створив американський програміст Брэндан Айк, який брав участь у створенні американської компанії Mozilla, що розробляє браузер Firefox. Мова програмування JavaScript, як пригадує її розробник, була створена за 10 днів. Вона подавалась як скриптова для виконання невеликих клієнтських завдань у браузері.

Синтаксис JavaScript навмисно було розроблено максимально подібним до Java, яка на той момент дуже активно використовувалася, а динамічну типізацію запозичено від не менш популярної мови Perl. Функції в JavaScript — це просто ще один тип об'єкта. Ними можна оперувати, як і будь-якими іншими елементами. Цією особливістю JavaScript зобов'язаний Scheme. Наслідування реалізовано через прототипи, як у мові Self.



Вебпрограмування — галузь веброзробки і різновид дизайну, в за- вдання яких входить проєктування користувальників вебінтерфейсів для сайтів або вебзастосунків.

JavaScript — мова програмування, що дозволяє реалізувати низку складних рішень у вебдокументах. Вона допомагає зробити сторінки сайта більш інтерактивними, обробляє дії користувачів сайта. Це об'єктно-орієнтована клієнтська мова, яка підтримується застосунками, що працюють з дизайном сайта.

Разом з HTML і CSS, JavaScript — третій важливий блок, на основі якого будується більшість стандартних вебінтерфейсів.

Функції JavaScript дозволяють:

- ◆ зберігати дані в змінних;
- ◆ активувати частину коду згідно з певними сценаріями, що реалізуються на сторінці сайта;
- ◆ створювати контент, який оновлюється автоматично;
- ◆ керувати мультимедійними можливостями (працювати з відео, анимувати зображення).

JavaScript обробляється у вебзастосунках на стороні клієнта, тобто у браузері. Завдяки цьому вона може виконуватися на будь-якій операційній системі, а вебінтерфейси, що працюють на його основі, є кросплатформними.

Переваги JavaScript:

- ◆ підтримує всі браузери;
- ◆ перевіряє реєстраційні форми на помилки ще до відправлення на сервер;
- ◆ створює яскраві й інтерактивні сторінки сайта;
- ◆ може здійснювати різного типу обчислення.

За даними дослідження сайту спільноти програмістів України Dou.ua (<https://dou.ua/>), проведеного у лютому 2020 року, найпопулярнішою мовою програмування в Україні є JavaScript (рис. 4.3.). За версією найбільш впливового сайта Stackoverflow (<https://insights.stackoverflow.com/survey/2019#technology>), JavaScript впевнено посідає перше місце у світовому рейтингу мов програмування, скриптових мов та мов розмітки (рис. 4.4.).

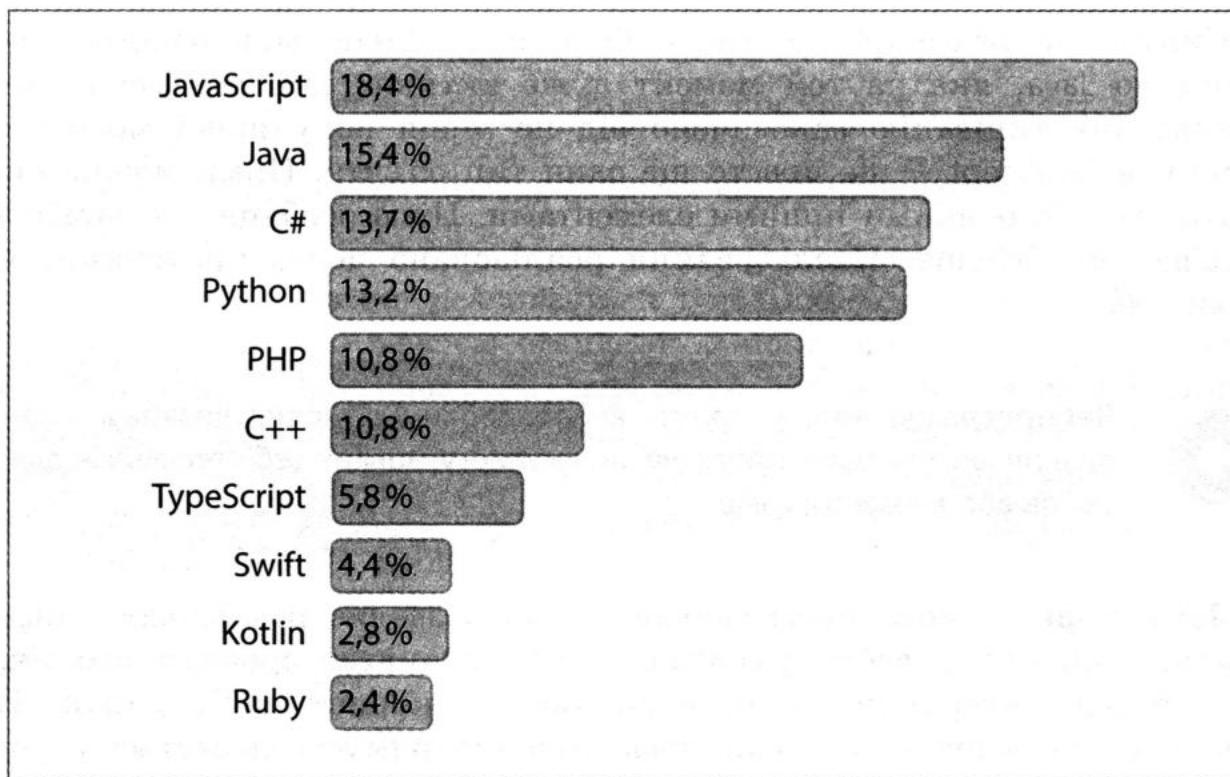


Рис. 4.3. Найпопулярніші мови програмування в Україні

Така популярність не є випадковою, адже значна кількість технологій, без яких неможливо уявити сучасний вебпростір, базується на використанні JavaScript. Серед них такі.

AJAX (англ. *Asynchronous Java Script And XML* — асинхронний JavaScript (мова програмування) і XML (мова розмітки вебсторінок)) — дає змогу створювати набагато зручніші вебінтерфейси користувача на тих сторінках сайтів, де необхідна активна взаємодія.

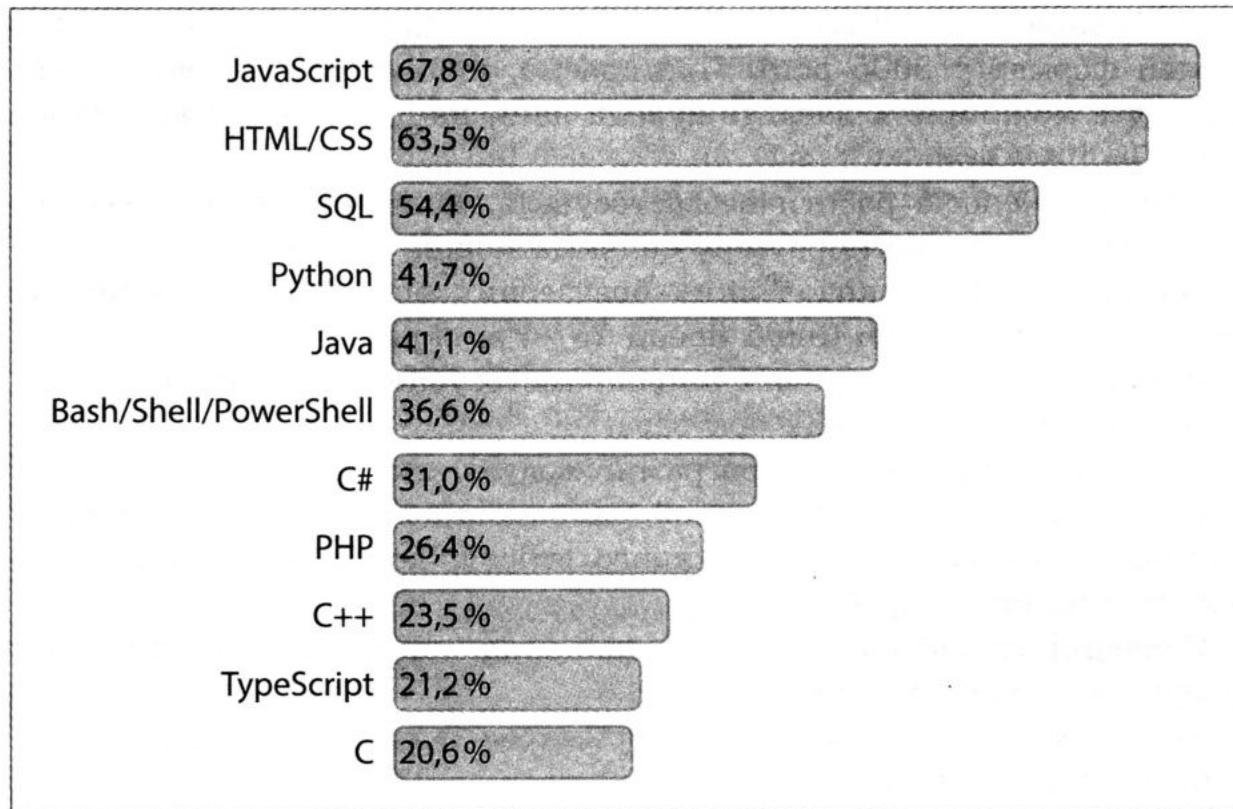


Рис. 4.4. Світовий рейтинг мов програмування

Поки сервер обробляє запит, користувач може переглядати вміст сайта. Браузер лише дозавантажує потрібні йому дані. Рівень використання AJAX значно підвищився після того, як компанія Google почала активно використовувати цю технологію у створенні своїх сайтів, таких як Gmail, Google Maps і Google Suggest.

Прикладами застосування технології AJAX є відображення контенту, що періодично оновлюється (інтерактивні карти); створення анімації й графічних об'єктів у форматі 2D/3D.

Поява формату **JSON** (англ. *JavaScript Object Notation* — запис об'єктів JavaScript) стала наступним важливим кроком. JSON був розроблений у 2001 році Дугласом Крокфордом. Цей легкий формат, який використовується для обміну даними, заснований на підмножині мови JavaScript (спосіб створення об'єктів у JavaScript), яка використовувала можливості звичайного браузера та дозволяла веброзробникам створювати вебзастосунки з постійним двостороннім зв'язком із вебсервером.

ЦІКАВІ ФАКТИ



Дуглас Крокфорд (фото зліва) — американський програміст, творець текстового формату обміну даними JSON.

Джеймс Гаррет (фото справа) — основоположник науки «інформаційна архітектура», засновник компанії Adaptive Path, яка займається консультаціями зі створення сайтів, дружніх до користувачів.

Найкраще JSON працює разом із AJAX. Джеймс Гаррет запропонував цей формат у 2005 році. Пригадаємо, що суттєвою перевагою цього формату є можливість довантажувати дані, не перевантажуючи сторінку (про JSON і AJAX дів. у § 4.7).

Comet — спосіб роботи вебзастосунків, коли під час HTTP-з'єднання сервер відправляє дані браузеру без додаткових запитів.

Браузерні ОС — код деяких браузерних операційних систем, який складається переважно (іноді понад 75 %) зі скриптів.

Закладки — JavaScript має широке застосування в роботі програм, що розміщаються в закладках браузера.

Браузерні скрипти — програмні модулі, які пишуться цією мовою і дають дуже багато можливостей (автозаповнення форм, зміна формату сторінок, приховання небажаного змісту, додавання інтерактивних елементів на сторінках).

Серверні застосунки — фрагменти коду, які виконуються на стороні сервера, де використовується Java 6.

Мобільні застосунки — JavaScript може бути корисною в цьому популярному напрямку.

Віджети — на мові JavaScript пишуть різні міні-програми, які використовують в робочому просторі і є дуже зручними.

Прикладне програмне забезпечення — об'єктно-орієнтована мова JavaScript використовується для створення окремих програм, у тому числі нескладних ігор. Наразі JavaScript є однією з найбільш популярних клієнтських мов.

Почувши словосполучення «інтерактивна сторінка», багато хто асоціює його з різними flash-ефектами, що активуються залежно від місця розташування курсору миші відвідувачів сайта. Проте цей стереотип дещо неправильний. Для реалізації інтерактивних «властивостей» сайтів застосовуються спеціальні програмні коди — **серверні скрипти**. Саме вони обробляють дані, отримані від відвідувачів сайта, і формують відповідну HTML-сторінку.

Для написання серверних скриптів застосовуються **серверні мови вебпрограмування**, такі як PHP, Perl, ASP.NET. Виконується серверний скрипт на стороні сервера: відвідувач не бачить вихідного програмного коду виконуваного скрипта, а отримує тільки готову відповідь.

Створити інтерактивну сторінку — означає створити сторінку, що вміє «спілкуватися» зі своїми відвідувачами. Проста статична сторінка доступна лише для перегляду. Для того щоб зв'язатися з адміністрацією сайта або зробити замовлення, відвідувачеві необхідно зателефонувати за контактним телефонним номером, написати електронного листа на e-mail або відправити факс. Під інтерактивним слід розуміти сайт, контент якого формується «на льоту». Поняття «інтерактивний» можна віднести до всіх сайтів, що мають форми надсилання повідомлень, онлайн-анкети та опитування, реєстраційні форми, лічильники відвідувань, форми для онлайн-замовлень та інші подібні елементи. Інтерактивність сторінки



насамперед забезпечується за допомогою HTML-форм: реєстраційних, надсилання повідомень, онлайн-замовлень тощо.

HTML-форми — це елементи управління для збирання інформації від відвідувачів вебсайта. Такі форми є набором текстових полів, кнопок, списків та інших елементів управління, які активізуються клацанням миші. Завдання форми полягає в передаванні даних користувача віддаленому серверу.

Для отримання й опрацювання таких даних використовуються мови вебпрограмування: PHP або Perl.

У HTML5 розроблено дуже потужний інструментарій для створення форм будь-якої складності. Далі ми розглянемо лише створення найпростіших форм.



Базою для форми є парний тег `<form>` — контейнер, який утримує всі елементи керування форми, — поля, що групуються залежно від призначення форми. Загалом можна сказати, що кожна форма є набором логічно пов'язаних елементів.

Серед атрибутів тегу є два обов'язкові: `action` і `method`.

`Action` містить URL-адресу, яка визначає, куди буде надіслано дані форми.

`Method` визначає спосіб відправлення даних із форми. Він може набувати двох значень: "get" (за замовчуванням надсилає дані на сервер через адресний рядок, тобто їх видно в адресному рядку браузера) та "post" (приховує запит). Зазвичай використовується "post", оскільки він дозволяє передавати великі обсяги даних.

Елемент `<input>` є одинарним (непарним) тегом і створює більшість полів форми. Атрибути елемента відрізняються залежно від типу поля, для створення якого використовується цей елемент, тому першим обов'язковим його атрибутом є `<type>`.

Найчастіше використовувані значення атрибута `type` наведено в таблиці.

Назва	Опис
Text	Створює у формі текстові поля, виводячи однорядкове текстове поле для введення тексту
Submit	Створює стандартну кнопку, що активізується клацанням миші, збирає інформацію з форми й надсилає серверу для опрацювання
Reset	Кнопка для повернення даних форми в початкове значення
Placeholder	Містить текст, який відображається в полі введення до заповнення (найчастіше це підказка), тобто в текстовому полі відображатиметься текст підказки, який зникне, щойно буде введено певний текст

Закінчення таблиці

Назва	Опис
E-mail	Браузери, що підтримують цей атрибут, очікують, що користувач уведе дані, відповідні синтаксису адреси електронної пошти
Password	Створює у формі текстові поля, при цьому символи, що вводяться користувачем, замінюються на зірочки (або інші значки, встановлені браузером)

Слід наголосити, що `type="email"` і `type="password"` забезпечують специфічну семантику для введення, визначаючи, яку інформацію повинно містити поле.

Другий атрибут, який треба зазначати, це `<name>` — ім'я поля, необхідне для правильного опрацювання даних на сервері. Зазвичай воно має бути унікальним, принаймні в межах форми. Для імені поля використовують латиницю.

Для оформлення форми незайвим буде робити підписи до полів. Для цього існує спеціальний тег — парний тег `<label>`. Його завдання — створення логічного зв'язку між текстом і полем введення. Крім того, якщо клацнути на текст такого підпису, то курсор переміститься у відповідне поле.

Найпростіший спосіб створити підпис — це просто обернути текст підпису і тег поля тегом `<label>` таким чином:

```
<label>
Введіть ім'я <input type="text" name="username">
</label>
```

Більш детально ознайомитися з можливостями HTML-форм можна за посиланнями: <https://css.in.ua/html/tag/form>, <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form>.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

- Що таке вебпрограмування?
- Яка мова програмування є базовою у веброзробці?
- Наведіть приклади використання JavaScript.
- Опишіть призначення форм. З яких елементів складається форма?
- Як досягається інтерактивність сторінок?
- Розгляньте такі елементи форми: текстові поля введення (`<textarea>`), розкривні списки (`<select>`), кнопки (`<button>`), пропорці (`<input type="checkbox">`), перемикачі (`<input type="radio">`). Інформацію про ці елементи оформіть у вигляді презентації.



ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

- Додайте до вашого сайта HTML-сторінку з назвою form.html і створіть на ній форму такого вигляду, як наведено на рис. 4.5.

Створити новий обліковий запис

Логін

Логін

Електронна скринька

Електронна скринька

Пароль

Пароль

Відправити

Рис. 4.5. Приклад форми



Дивіться презентацію «Вебпрограмування та інтерактивні сторінки».

Практична робота № 10

ТЕМА. Створення інтерактивної сторінки за допомогою форми

ЗАВДАННЯ: використати елементи форми для розробки інтерактивної сторінки.

ОБЛАДНАННЯ: комп’ютер із доступом до мережі інтернет, будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп’ютером дотримуйтесь правил безпеки.

- Відкрийте ваш сайт. Додайте форму, яка дозволяє увійти зареєстрованому користувачу.

Зазвичай такі форми розташовуються безпосередньо у блочі навігації на головній сторінці сайта й мають три поля:

- ◆ поле для логіну <input type="text">;
- ◆ поле для паролю <input type="password">;
- ◆ кнопка надсилання <input type="submit">.



2. Відкрийте в редакторі коду файл index.html.
3. Пропишіть у блоці <nav> після блоку наведений код (рис. 1).

```
<form action="#">
    <input type="text" placeholder="Login" required>
    <input type="password" placeholder="Password" required>
    <input type="submit" value="Sign In">
</form>
```

Рис. 1.
Код для
додавання

Примітка. Елемент `action` містить адресу, яка визначає, куди буде надіслано інформацію форми. В нашому випадку ми просто ставимо знак «#». Елемент `placeholder` містить текст-підказку, який зникатиме, щойно користувач буде вводити текст. Атрибут `required` означає, що поле обов'язкове для заповнення. Якщо воно залишиться незаповненим, то браузер виведе повідомлення про те, що поле порожнє, і форму надіслано не буде.

4. Відкрийте в редакторі коду файл style.css.
5. Додайте форматування форми (рис. 2).

```
form{
    padding: 15px 0;
    float: right;
}
```

Рис. 2. Форматування форми

6. Перевірте коректність введення коду в браузері.
7. Створіть форму, що дозволяє користувачеві надсилати коментарі. Для цього відкріте в редакторі коду файл form.html.
8. Пропишіть у блоці <body> код, як наведено на рис. 3.

```
<form action="#" method="post" name="comment" id="comment">
    <p><input type="text" name="author" id="author" value="" size="25"/>
        Ім'я
    </p>
    <p><input type="text" name="email" id="email" value="" size="25"/>
        Пошта
    </p>
    <p><textarea name="comment" id="comment" cols="48" rows="8"></textarea>
    </p>
    <p><input type="submit" name="submit" id="submit" value="Надіслати"/>
    </p>
</form>
```

Рис. 3. Код для блока <body>

9. Додайте в таблицю стилів відповідне форматування для створеної форми.
Зробіть висновок: які можливості для інтерактивності надає використання форм.



4.3 Хостинг сайту

Сайт створюється насамперед для того, щоб його відвідували користувачі. Для цього якимось чином їм потрібно надати доступ до нього. Звісно, ми можемо розмістити наші вебсторінки на власному комп'ютері й використовувати його як сервер. Проте це має бути потужний комп'ютер із безперешкодним доступом до нього в режимі «нон-стоп» (тобто 24 години на добу і 7 днів на тиждень), а також виділений канал передавання даних, який надає можливість одночасного звернення до серверу великої кількості осіб. Двоє з п'яти користувачів інтернету відмовляються від повільного завантаження вебсайта (під повільним розуміється сайт, який завантажується більш ніж 3 с). 1,5 млрд долларів щорічно втрачаються в економіці США через повільне завантаження вебсайтів, а затримка на 1 с знижує коефіцієнт переходів на 7 %. Оскільки описане для власного комп'ютера не завжди можливо, а швидкість завантаження відіграє важливу роль, логічно звернутися до організацій, які спеціалізуються на наданні хостингу.

Поняття хостингу включає широкий спектр послуг із використанням різного апаратного та програмного забезпечення. Зазвичай під цим поняттям, як мінімум, мають на увазі послугу розміщення файлів сайта на сервері, на якому запущене програмне забезпечення, необхідне для обробки запитів до цих файлів (вебсервер).

Вебхостинг — це послуга, що дозволяє приватним особам, підприємствам і організаціям розміщувати в інтернеті вебсайт або вебсторінку. Послуги хостингу надають технології та підтримку, необхідні для перегляду вебсайта або вебсторінки в інтернеті.

Вебсайти розміщаються або зберігаються на спеціальних комп'ютерах, які називають серверами, що обслуговуються чи належать службі вебхостингу або передані в оренду сторонніми службами, які експлуатують «ферми серверів», або дата-центрі.



Хостинг (англ. *hosting*) — послуга, яка включає надання дискового простору, підключення до мережі та інших ресурсів для розміщення фізичної інформації на сервері, що постійно перебуває в мережі (наприклад, інтернету).

Послуги хостингу можуть надаватися у пакеті з іншими інформаційними послугами, такими як реєстрація доменного імені, створення сайта, надання додаткового програмного забезпечення тощо.

Зазвичай до послуг хостингу вже входить надання місця для поштової кореспонденції, баз даних, DNS-файлового сховища тощо, а також підтримка функціонування відповідних сервісів. Однак вони можуть надаватися й окремо. А власне послуга може бути обмежена розміщенням

поштової кореспонденції та відповідного ПЗ (поштовий хостинг), клієнтських файлів (файловий хостинг), виключно відеофайлів (відеохостинг) або інших файлів певного типу та за певних умов.

Провайдерами хостингу можуть виступати як компанії, що спеціалізуються на цих послугах (хостери), так і великі провайдери інформаційних послуг, які спеціалізуються на інших послугах (такі як Google, Microsoft, Yahoo та ін.).

10 найбільших хостингових компаній становлять 24 % ринку вебхостингу, тобто кожен четвертий вебсайт користується їх послугами.

Розрізняють такі види хостингу:

- ◆ віртуальний;
- ◆ віртуальний виділений сервер;
- ◆ виділений сервер;
- ◆ хмарний хостинг;
- ◆ вебхостинг;
- ◆ керований WordPress-хостинг.

Розглянемо види хостингу більш детально (рис. 4.6).

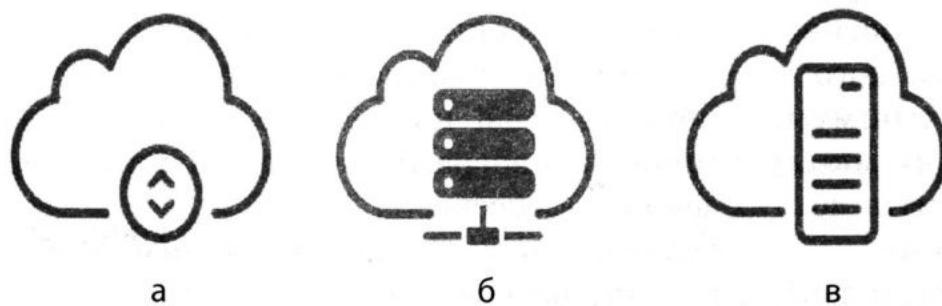


Рис. 4.6. Деякі види хостингу: віртуальний (а); віртуальний виділений сервер (б); виділений сервер (в)

Віртуальний хостинг (*virtual hosting*, або *shared hosting*) — користувачеві надається частина місця на диску для розміщення вебсайтів. При цьому середовище виконання вебсервісів єдине для багатьох користувачів, а апаратні й програмні ресурси розподілені між усіма користувачами на одному сервері, де може розміщуватись від 50 до 1000 користувачів.

Перевагами віртуального хостингу є відносно низькі ціни та набір послуг, що є адекватним для функціонування невеликого та оптимізованого сайта. Недоліками віртуального хостингу можна вважати те, що через розподілення ресурсів сервера між багатьма користувачами надмірне споживання цих ресурсів одним сайтом може вплинути на роботу інших.

Віртуальний виділений сервер (VPS, або VDS) — послуга, в рамках якої користувачеві надається так званий віртуальний виділений сервер. Спосіб керування операційною системою здебільшого відповідає управлінню фізичним виділеним сервером, зокрема, права адміністратора, root-доступ, власні IP-адреси, порти, правила фільтрування і таблиці маршрутизації.



Віртуальний виділений сервер надає більше можливостей і привілеїв, а часто і більше ресурсів, ніж віртуальний хостинг. Водночас він є і більш дорогим.

Виділений сервер (dedicated server) — сервер надається повністю і використовується для реалізації нестандартних завдань (сервісів), розміщення «важких» вебпроектів, які не можуть співіснувати на одному сервері з іншими проектами і вимагають для себе всі ресурси.

Виділений сервер вважається найбільш ефективним і водночас найдорожчим серед усіх видів хостингу. Управління ним потребує від користувача найбільше технічних знань і навичок.

Колокація (collocation) — надання місця в дата-центрі провайдера для обладнання клієнта (зазвичай шляхом монтажу в стійці) і підключення його до інтернету. Надає можливість користуватися інфраструктурою дата-центру (системами охолодження повітря, пожежної безпеки тощо).

Колокацією користуються переважно досвідчені клієнти, що мають достатньо навичок для підключення й адміністрування власних серверів, або компанії, які самі надають інформаційні послуги.

Хмарний хостинг (cloud hosting, або cloud storage) — послуга з розміщення файлів користувача, за якої дані зберігаються на багатьох серверах, що розподілені в мережі (рис. 4.7). Файли зберігаються у так званій «хмарі», що фізично складається із серверів, які можуть знаходитися далеко один від одного, але з точки зору користувача працюють як один потужний віртуальний сервер. Перевагами хмарного хостингу є можливість колективної роботи з даними та відсутність прив'язки до ресурсів одного окремого сервера.

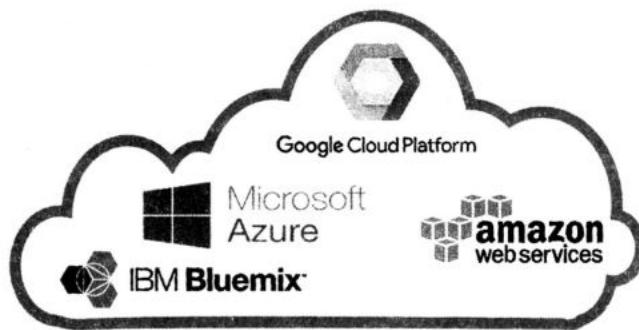


Рис. 4.7. Логотипи платформ хмарного хостингу

За наведеними даними (рис. 4.8), безперечним лідером є Amazon WEB Services, скорочено AWS (<https://aws.amazon.com/>), із величезним набором інструментів. Платформа була пionером у цій галузі й завоювала чималий ринок. Можливості Amazon вже сьогодні не мають собі рівних і продовжують зростати в геометричній прогресії.

Microsoft Azure (<https://azure.microsoft.com/en-us/>) — близький конкурент AWS із надзвичайно розвиненою інфраструктурою. Система існує з 2010 року й розвивається швидкими темпами. Наразі Microsoft Azure являє собою багатогранну складну систему, яка забезпечує підтримку багатьох видів послуг, мовних програм і фреймворків.

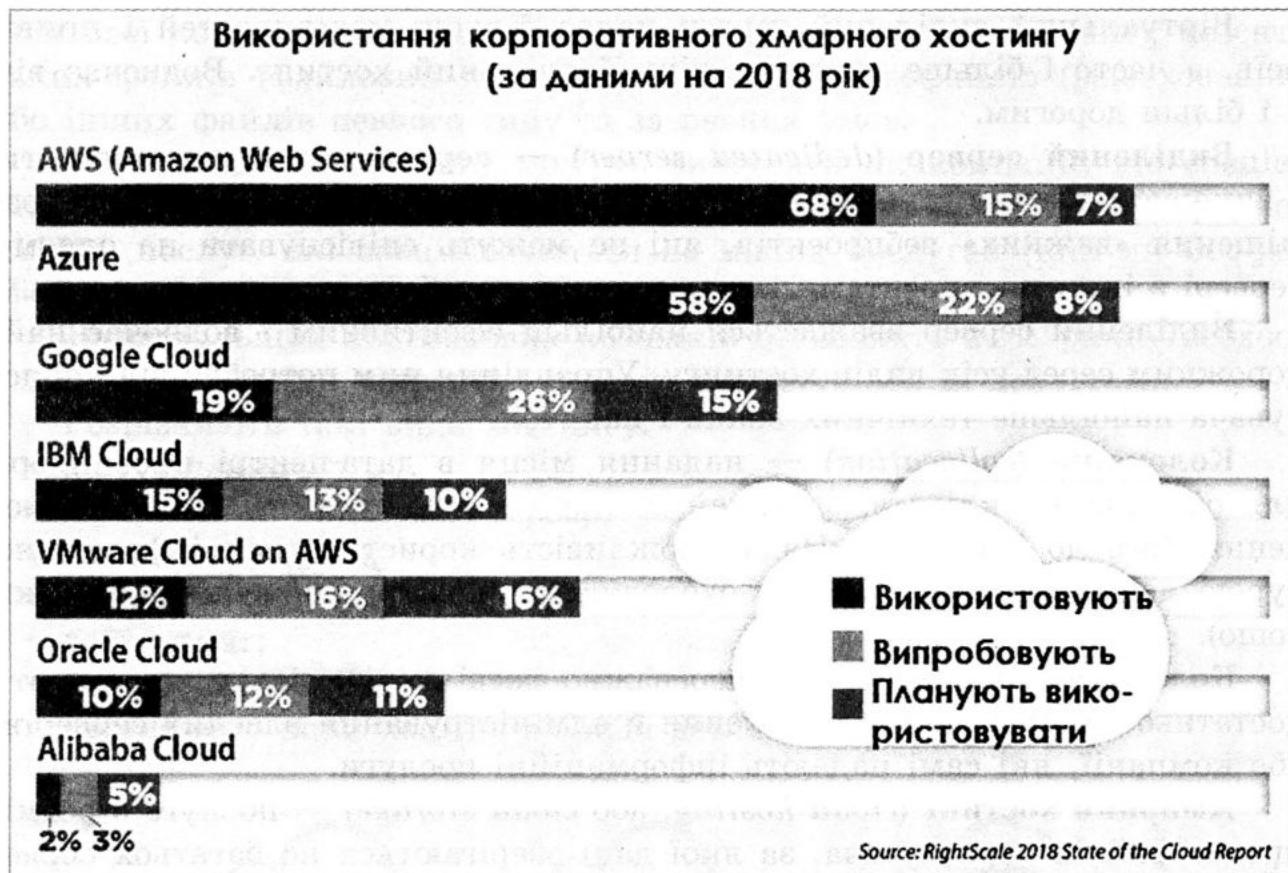


Рис. 4.8. Рейтинг платформ хмарного хостингу

Третє місце посідає Google Cloud (<https://cloud.google.com/>), який з'явився на ринку хмарного хостингу пізніше, проте його провідна роль у галузі штучного інтелекту, машинного навчання й аналітики даних надає значні переваги.

Реселлер хостинг (*reseller hosting*) — хостинг із послугою перепродажу. Користувачеві надається можливість розподіляти дисковий простір і ресурси свого віртуального хостингу або сервера з метою розміщення на ньому сайтів третіх осіб, що можуть бути його клієнтами. Пакет послуг такого хостингу зазвичай включає спеціальне програмне забезпечення для управління клієнтською базою, ресурсами, що надані клієнтам, тощо.

Зазвичай компанія, що надає безкоштовний хостинг, заробляє шляхом показу реклами на сторінках, розміщених на ньому. Такий хостинг, як правило, повільніший від платного, надає тільки базові послуги, недостовірний. Безкоштовний хостинг часто надається для того, щоб через якийсь час перевести користувача на платну основу шляхом погіршення умов користування.

Безкоштовний хостинг використовують приватні особи для своїх домашніх сторінок на початковому етапі їх розвитку. Громадські організації можуть використовувати як платний хостинг, так і безкоштовний. Комерційні організації практично завжди користуються послугами платного хостингу.



Вебхостинг є одним із типів інтернет-хостингу, який дозволяє окремим особам та організаціям зробити свій сайт доступним через World Wide Web. Сайти компаній, які надають місце на сервері, що належать або орендовані для використання клієнтами, а також інтернет-з'єднання, як правило, розташовані у центрі обробки даних.

Багато інтернет-провайдерів (ISP) пропонують цю послугу безкоштовно для абонентів. Приватні особи та організації можуть також отримати вебсторінку хостингу від альтернативних постачальників послуг. Особистий вебсайт-хостинг, як правило, безкоштовний або недорогий. Бізнес-вебсайт-хостинг часто платний.

Насамкінець, **WordPress-хостинг** — це різновид загального хостінгу, який спеціально створений для розміщення сайтів на WordPress. При цьому сервер налаштований на найбільш оптимальний режим роботи саме з CMS WordPress. На сайті відразу є заздалегідь встановлені плагіни для кешування, безпеки тощо. Завдяки оптимізованій конфігурації сайт швидше завантажується, також існують додаткові функції, пов'язані з WordPress, такі як додаткові теми WordPress, конструктори сторінок простих перегортань і спеціальних інструментів розробки. Прикладом такого хостингу є перший спеціалізований хостинг в Україні (<https://wphost.me/>).

Який би тип хостингу не вибрали користувачі, їм необхідна панель керування вебхостингом, що дозволяє:

- ◆ керувати файлами;
- ◆ здійснювати статистику трафіку;
- ◆ адмініструвати бази даних;
- ◆ зберігати облікові записи електронної пошти та конфігурацію;
- ◆ керувати протоколом FTP;
- ◆ здійснювати переадресацію;
- ◆ автоматизувати виконання задач, які повторюються (наприклад, за допомогою популярної утиліти Cron);
- ◆ формувати DNS.

Існує ціла низка панелей керування вебхостингом (Parallels Plesk Panel, DirectAdmin, Webmin тощо). Деякі хостингові компанії використовують один тип панелі керування хостингом, тоді як інші дозволяють обирати один із чотирьох варіантів. Сьогодні більшість панелей керування хостингом також дозволяють одним натисканням кнопки установлювати популярні сценарії, такі як WordPress.

cPanel є найпопулярнішою панеллю керування хостингом онлайн (рис. 4.9) завдяки легкому налаштуванню.

cPanel розроблена на початку 1996 року для серверів Linux. Проте компанія пропонує панель керування хостингом під назвою Enkompass, створену й для серверів Windows. Крім того, існує додаткова панель керування, яка дає змогу адмініструвати декілька вебсайтів і налаштувати кожен аспект сервера.

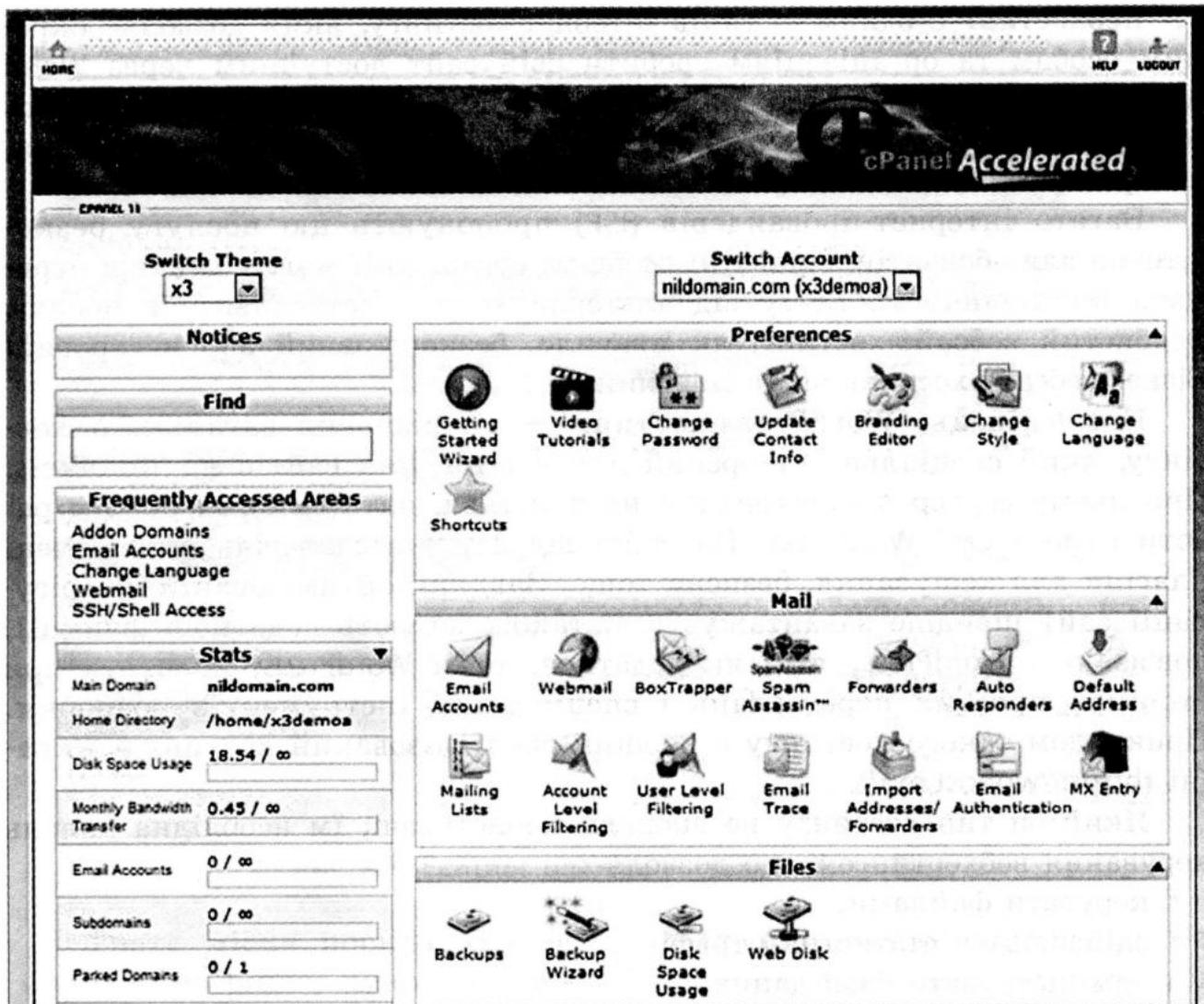


Рис. 4.9. Приклад найбільш поширеної панелі керування вебхостингом cPanel



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке хостинг?
2. Назвіть види хостингів.
3. Опишіть різницю між віртуальним виділеним сервером; виділеним сервером.
4. Пригадайте, що таке SSL.
5. Знайдіть в інтернеті відомості про українських інтернет-провайдерів. Порівняйте вартість їхніх послуг. Результати подайте у вигляді інфографіки.
6. Які, на вашу думку, недоліки має безкоштовний хостинг?



Дивіться презентацію «Хостинг сайту».





Практична робота № 11

ТЕМА. Хостинг сайту

ЗАВДАННЯ: зареєструвати розроблений на попередніх практичних роботах сайт у хостинг-провайдера.

ОБЛАДНАННЯ: комп’ютер із доступом до мережі інтернет, будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп’ютером дотримуйтесь правил безпеки.

Примітка. Практичну роботу зорієнтовано на використання безкоштовного хостингу. Оскільки хостинг-провайдери непередбачувано змінюють тарифи, автор не гарантує актуальність запропонованого хостингу. Наразі використовується хостинг <https://www.ho.ua/uk/>.

Заповнення форми заяви

- Уведіть в адресний рядок браузера url-адресу <https://www.ho.ua/uk/>. Завантажиться головна сторінка сайта (рис. 1).

VPS-ХОСТИНГ SSD ВІРТУАЛЬНИЙ СЕРВЕР	VPS-ХОСТИНГ HDD ВІРТУАЛЬНИЙ СЕРВЕР	БЕЗКОШТОВНИЙ ЛЕНДІНГ	ЗВІЧАЙНИЙ ХОСТИНГ
50 ГБ диск SSD 2000 МБ пам'яти CPU 2000 Мгц Безлімітний трафік 1 реальна IP адреса. Техпідтримка по e-mail. Безкоштовний тест.	100 ГБ диск HDD 2000 МБ пам'яти CPU 1000 Мгц Безлімітний трафік 1 реальна IP адреса. Техпідтримка по e-mail. Безкоштовний тест.	1 ГБ диск HDD 1 сайт до 10 доменів Безлімітний трафік Без реклами. PHP, MySQL, CGI, SSL, Perl, Python, crontab, htaccess	15 ГБ диск SSD 2 сайта до 20 доменів Безлімітний трафік SSL, PHP, MySQL, CGI, SSL, Perl, Python, crontab, htaccess, phpMyAdmin
Детально От 125 грн / місяць Замовити	Детально Від 115 грн / місяць Замовити	Детально Обмеження навантаження на базу даних; Безкоштовно Замовити	Детально Від 63 грн / місяць Заказать

Рис. 1. Головна сторінка сайта

- Ознайомтеся з інструкцією користувача (рис. 2).
- Виберіть безкоштовний лендінг — відкриється форма (рис. 3).

При замовленні хостингу вам необхідно заповнити наступні поля реєстраційної форми:

- **Ваше ім'я та назва організації** – буде використовуватися для рахунку.
- **Ваш контактний e-mail** – адреса електронної пошти, на яку буде відправлено ключ активації хостингу і через яку буде здійснюватися підтримка хостингу. Ця адреса повинна бути рабоча, тому що на неї будуть відправлятися всі інформаційні та технічні повідомлення.
- **Бажаний логін** – ім'я (маленькі латинські літери, цифри і знак тире) довжиною до 16 символів, яке буде використовуватися для доступу по FTP. Це також частина адреси вашого майбутнього акаунта хостингу (наприклад, LOGIN.ho.ua, де LOGIN – це вказаний вами логін). Це також і ім'я можливої бази даних MySQL. Врахуйте, що бажаний логін може бути вже кимось зайнятий і вам доведеться його уточнити. Поряд з логіном вам також пропонується на вибір 2 варіанти доменних імен для сайту (*.ho.ua и *.houa.org).
- **Ваші існуючі домени** – Якщо у вас є домени (зареєстровані у будь-якого реєстратора доменних імен), які підтримуються на NS-серверах і які ви бажаєте налаштувати для майбутнього хостингу – вкажіть їх в даному полі форми.
- **Країна** – це країна, де ви мешкаєте.

Після заповнення реєстраційної форми натисніть кнопку **Замовити**. Якщо форма заповнена вірно та вказаний логін в нашій базі відсутній, ви побачите повідомлення про те, що на ваш e-mail відправлено листа з ключем (посилання) активації хостингу. Вам необхідно **уважно** прочитати цей лист, перш ніж переходити по вказаному в ньому посиланню.

Якщо ж при реєстрації виникли деякі помилки (неточність заповнення форми, бажаний логін зайнятий тощо) – вам буде повідомлено про це та запропоновано їх відправити.

Рис. 2. Інструкція користувача

Замовлення хостингу

Ваше ім'я:

Ваша контактна e-mail:

Назва організації:

Бажаний логін: .

Ваші існуючі домени:

Країна:

З правилами ознайомлений та згоден (правила)

Замовити хостинг

Рис. 3. Форма для безкоштовного лендінгу



4. Заповніть форму.
5. На вказану вами електронну пошту буде надіслано листа, що містить відомості про активацію хостингу.

Примітка. Ви маєте відправити SMS-підтвердження, вартість якого залежить від вашого тарифного плану в оператора мобільного зв'язку. SMS приймаються лише з реальних мобільних телефонів.

Активація хостингу

Приблизно за 10 хв після надіслання SMS хостинг буде активовано. Ви отримаєте листа з налаштуваннями та паролем.

1. Клацніть панель керування (рис. 4).

Рис. 4. Панель керування

2. Уведіть отриманий поштою логін і пароль, натисніть Вхід.
- Ви отримаєте доступ до панелі керування своїм доменом (рис. 5).
3. Виберіть кнопку Управління файлами і перейдіть безпосередньо до каталогів (рис. 6).

Примітка. Усі сторінки слід розміщувати у каталозі `htdocs/`. На початку там знаходяться зразки документів і скриптів, роботу яких можна побачити.

У каталозі `cgi-bin/` сайта міститься інтерпретатор `php`, необхідний для роботи `php`, у каталозі `logs` — файли протоколу роботи сайта. Розмір цих файлів враховується у загальній квоті. Для подальшої роботи користуються меню, розташованим над вмістом каталогу (рис. 7).

4. Створіть каталог `site` у каталозі `htdocs/`.
5. Завантажте файли у створений каталог.
6. Об'єднайтесь в малі групи (по двоє-троє осіб) і перегляньте активовані на хостингу сайти.

Статистика	Перегляд статистики
Управління файлами	Зміна вмісту сайту
Завдання (cron)	Управління завданнями
Тема	Тематика сайту
Категорія	Категорія сайту
Домени	Управління доменами
База даних	Управління базою даних
Журнальні файли	Управління журнальними файлами
Пароль	Зміна пароля
Контакти	Зміна контактної інформації
Продовження	Продовження хостингу
Оплата	Отримання квитанції для оплати хостингу
Повернутися	Увійти з іншим ім'ям

Рис. 5. Панель керування доменом

 ЗМІСТ КАТАЛОГА ~/ 					
ЗМІСТ КАТАЛОГА ~/					
<input type="checkbox"/>	Ім'я 	Режим доступу	Розмір	Час модифікації	Опції
<input type="checkbox"/>	 cgi-bin	rxwxr-xr-x	755	-	2012-07-16 16:19:08  
<input type="checkbox"/>	 htdocs	rxwxr-xr-x	755	-	2012-07-16 16:25:53  
	з виділеними виконати:	--виберіть дію--			
	Дискова квота:	зайнято: 52 кб; вільно: 1048524 кб			
 ДІЇ 					

Рис. 6. Каталоги

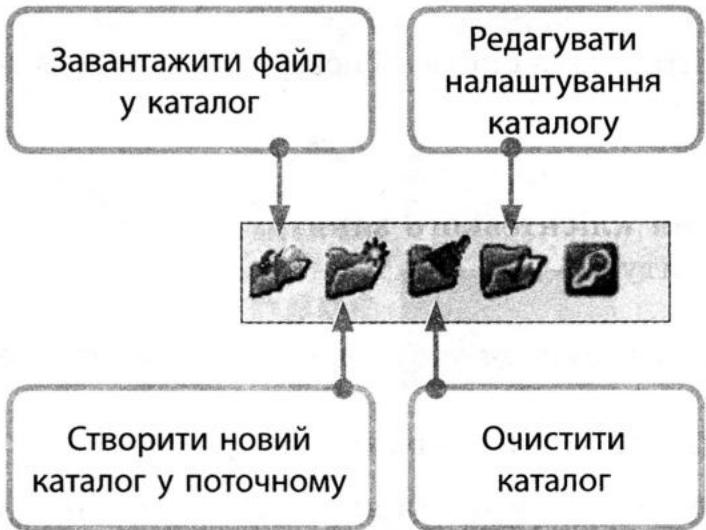


Рис. 7. Меню для роботи

Зробіть висновок за результатами роботи.

4.4

Вебсервер та бази даних

Пригадаємо, що розробка зі сторони браузера називається front-end, а розробка на стороні сервера — back-end. Наразі мова піде про розробку функціоналу на стороні сервера. Серверну сторону вебсайта найчастіше пишуть такими мовами програмування, як Python, PHP, Java, C#, Ruby, JavaScript.

Щоб завантажити вебсторінку, браузер надсилає запит до вебсервера (рис. 4.10), який приступає до пошуку запитуваного файла у своєму власному просторі пам'яті. Знайшовши файл, сервер його читає, опрацьовує, як йому потрібно, і повертає до браузера. Отже, вебсервер повинен містити файли вебсайта, а саме всі HTML-документи і пов'язані з ними ресурси, включаючи зображення, CSS-стилі, JavaScript-файли, шрифти й відео. Вебсервер забезпечує підтримку HTTP.

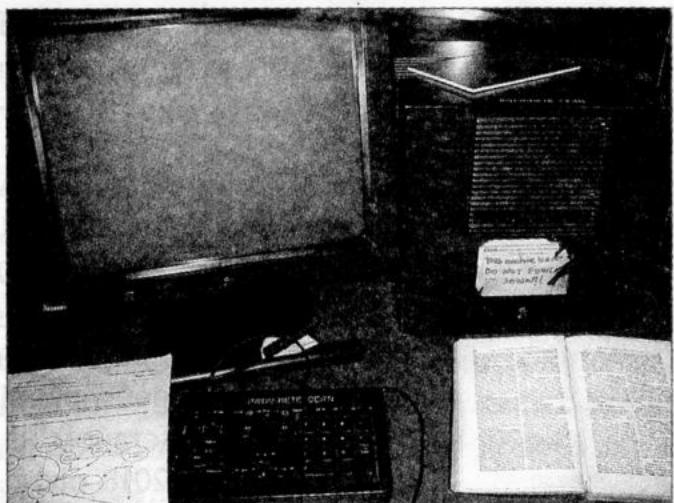


Рис. 4.10. Перший у світі вебсервер, робоча станція NeXT Computer з Ethernet, 1990 рік. На етикетці написано: «Ця машина є сервером. НЕ ВИМИКАТИ!!»



Вебсервер — це програма, яка створює і повертає відповіді на запити вебресурсів клієнтам.

Порядок дій опрацювання клієнтського запиту:

- 1) синтаксичний аналіз запиту;
- 2) перевірка повноважень;
- 3) зв'язування URL у запиті з ресурсом у файловій системі сервера;
- 4) побудова відповіді;
- 5) повернення відповіді клієнту, який звернувся із запитом.

Сервер може генерувати повідомлення-відповідь у різний спосіб. У найпростішому випадку він лише витягує файл, асоційований із URL, і повертає вміст клієнту. В інших випадках сервер може викликати сценарій, який зв'язується з іншими серверами або базами даних для побудови повідомлення-відповіді.

Моніторингові компанії приводять порівняльну статистику, зіставляючи кількість наявних сайтів для різних вебсерверів. І слід зазначити, що у всіх подібних дослідженнях фігурують лише три назви: Apache, IIS (сервер компанії Microsoft) і Nginx (рис. 4.11). Саме на цих трьох китах тримається основна частина мережевого світу. Безумовним лідером понад 20 років залишається Apache.

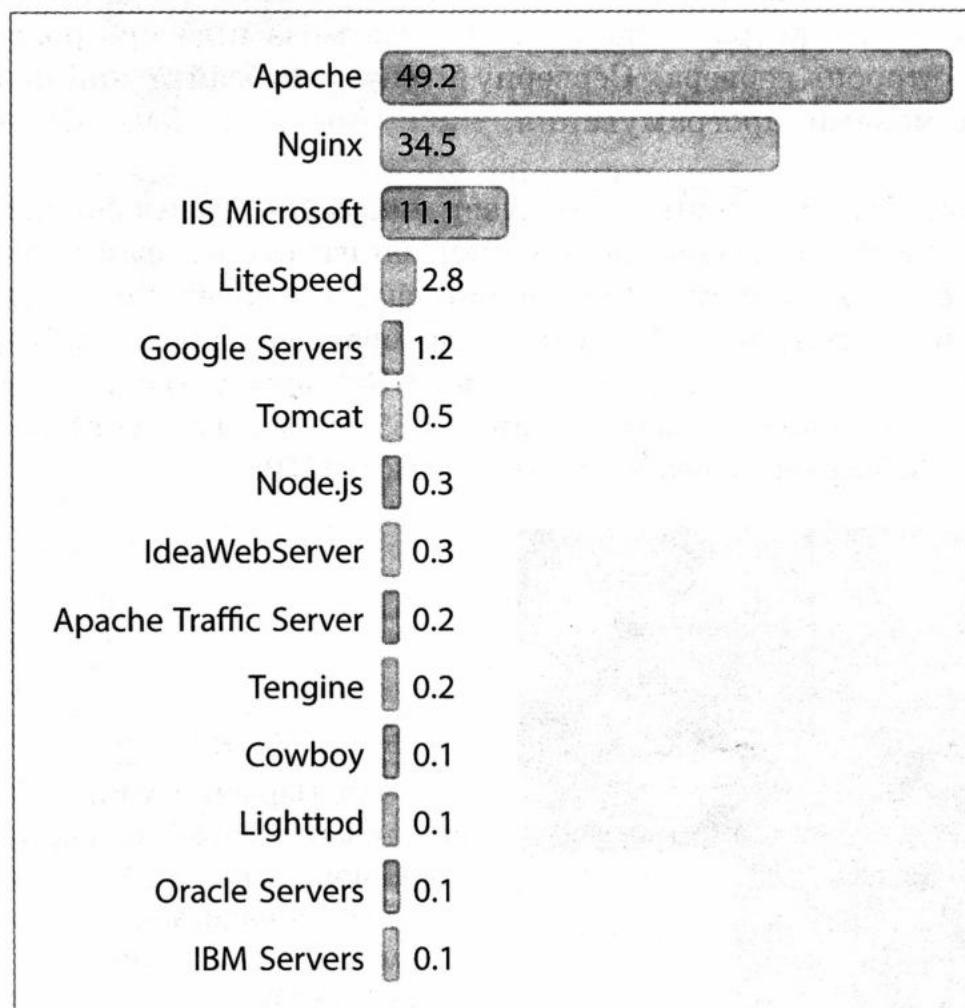


Рис. 4.11.
Рейтинг
популярності
вебсерверів
(станом на
2018 рік)

Розглянемо найпопулярніші вебсервери (рис. 4.12).

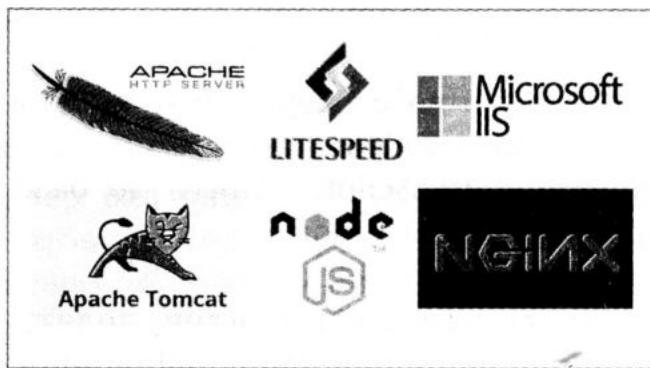


Рис. 4.12. Логотипи найпопулярніших вебсерверів

- ♦ **Apache** — кросплатформний вебсервер, він добре працює як на Unix-, так і на Windows-серверах. Сервер і клієнт взаємодіють за протоколом HTTP, і Apache відповідальний за безпечне з'єднання між двома машинами. Вебсервер працює з різними мовами програмування (PHP, Python, Perl та ін.) за допомогою спеціального модуля Apache (mod_php, mod_python, mod_perl та ін.).

Якщо отримано запит із розширенням .php, сторінки обробляються PHP (*Hypertext Preprocessor*, або *Personal Home Page*). PHP є мовою сценаріїв, що не залежить від платформи, сценарії вбудовуються в HTML-код. Інтерпретатор PHP виконується під управлінням вебсервера, інтерпретатор здійснює синтаксичний аналіз і обробку файлів. PHP-файл може містити дані, надіслані користувачем у HTML-формах.

У PHP є гнучкі засоби підтримки роботи з базами даних. Замість того щоб вбудовувати інформацію в HTML-файл, деякі програми можуть генерувати весь ресурс. У цьому випадку URL може розв'язувати різні завдання, такі як доступ до інформації в базі даних або створення відповіді, зміст якої залежить від клієнта, що звертається із запитом.

- ♦ **Nginx** — вебсервер, перший реліз якого відбувся в 2004 році. Наразі він набув значної популярності. Nginx було створено для розв'язування так званої «проблеми c10k» — проблеми 10 тисяч з'єднань. Це означає, що вебсервери, які використовують потоки, не можуть обробляти запити користувачів більше ніж із 10 000 підключень одночасно.

Вебсервер Nginx має подійно-орієнтовану архітектуру, тобто він обробляє кожен вхідний запит в єдиному потоці. Він використовується сайтами з великою кількістю показів, такими як Netflix, Pinterest і Airbnb.

- ♦ **Apache Tomcat** — ще один HTTP-сервер, проте він обробляє додатки Java замість статичних сайтів. Google Web Server, IBM HTTP Server (IHS), Oracle HTTP Server (OHS) — всі ці сервери базуються на Apache і доопрацьовані відповідно компаніями Google, IBM та Oracle.

Сервер Apache Tomcat, опрацьовуючи HTTP-запит, виконує:

- 1) перетворення URL-запиту в шлях до файла у файловій системі сервера;
- 2) визначення, чи має запит дозвіл на доступ до файла;
- 3) ідентифікацію;

- 4) виклик обробника для створення відповіді;
- 5) передачу відповіді клієнту;
- 6) створення запису про запит у журналі.

♦ **NODE.JS** — також кросплатформний вебсервер. Він працює з одним потоком, і його перевага в тому, що і клієнтська частина, і серверна частина написані однією мовою — JavaScript. Запити до бази даних можна робити, також використовуючи цю мову.

Ознайомимося з цікавими сторінками історії.

У 1994 році співробітник Національного центру додатків для суперкомп'ютерів в Університеті Іллінойсу США (NCSA) Роб Маккул виклав у загальне користування перший вебсервер — NCSA HTTP daemon. Сервер набув поширення, а влітку 1994 року Маккул залишив університет, і розробки припинилися.

Невелика група зацікавлених вебмайстрів почала спільну роботу над продуктом. Спілкуючись у дискусійному листі електронною поштою, вони розробляли нововведення на базі NCSA Server 1.3. Саме вони й заснували Apache Group, яка розробила першу версію Apache-сервера. Сталося це у квітні 1995 року, коли з'явився перший офіційний публічний реліз Apache 0.6.2. Назва Apache була пов'язана з широким використанням розробниками технології «програмних латок» (*patches*) і є скороченням від *a patchy server*.

Робота над сервером не припинялася. Після численних випробувань 1 грудня 1995 року з'явилася версія 1.0, стійка і надійна.

Протягом усіх цих років і до сьогодні Apache залишається абсолютно безкоштовним. За даними NetCraft, Apache на сьогоднішній момент встановлено на 67 % всіх серверів світу.

Ми вже говорили, що у вебсерверів є досить розвинені засоби підтримки роботи з базами даних. Пригадаємо, що бази даних поділяються на реляційні та нереляційні. До найпопулярніших реляційних належать MySQL, PostgreSQL, Oracle, до найпопулярніших нереляційних — MongoDB, CouchDB, HBase, Redis, Cassandra.

Щоб взаємодіяти напряму із **реляційною базою**, потрібно опанувати ази мови запитів баз даних — SQL. Проте з часом виникла потреба у складних вебсайтах, що повинні бути швидкими при великих об'ємах даних, у вебчатах, а можливостей реляційних баз даних уже бракувало.

Були розроблені так звані **нереляційні (NoSQL) бази даних**. Вони діляться на дві підгрупи: перша розв'язує питання швидкодії під час роботи з великими об'ємами даних, а друга дозволяє будувати ефективні «живі» вебзастосунки (такі, як чати і новини в соцмережах).

Звернемо увагу на те, що поняття вебсервера несе два змістових навантаження. З одного боку, це сервер, на якому розміщено вебсайт з усіма файлами, з яких він складається, а з іншого — так називається програма, що опрацьовує запити клієнтів, які стосуються вебресурсів.

Існує багато серверів застосунків. Причому деякі з них орієнтується на певні категорії вебсайтів, такі як блоги, вікі-сторінки або



інтернет-магазини. Деякі, наприклад системи управління контентом, є більш універсальними.

Сервер застосунків — сервер, на якому створені застосунки, що використовують вашу базу даних, вебсервіс і т. д. Він буде розміщувати бізнес-рівень (загорнутий вебслужбами), заплановані завдання, служби Windows та ін.

Сервер баз даних — матиме одну або кілька баз даних, таких як Oracle, SqlServer, MySql та ін.

Програмне забезпечення вебсервера, застосунків і баз даних може працювати на одному фізичному сервері або розподілятися по кількох фізичних машинах. На більшості великих сайтів є кілька машин; більшість «споживчих» хостингових пакетів запускаються в одному вікні.

Логічно поділ виглядає таким чином. Вебсервер обробляє запити HTTP (S) і передає їх «обробникам». У них є вбудовані обробники для запитів файлів — HTML-сторінки, зображення, CSS, JavaScript і т. д.

Якщо необхідно обробити динамічну сторінку, сервер передає її спеціальній програмі, яка і формує остаточну сторінку. Така програма називається **сервером застосунків**. Вона виконує читання коду, що знаходиться на сторінці, формує остаточну сторінку відповідно до прочитаних кодів, а потім вилучає його зі сторінки.

У результаті всіх цих операцій виходить статична сторінка, що передається вебсервером, який, у свою чергу, надсилає її клієнтському браузеру. Всі сторінки, які отримує браузер, містять тільки HTML-код.

Сервер застосунків надає можливість використовувати бази даних.

Розглянемо приклад 1.

ПРИКЛАД 1

Динамічна сторінка може містити програмні інструкції для сервера застосунків, при виконанні яких серверу необхідно отримати певні дані з бази даних і розмістити їх у HTML-код сторінки.

Програмна інструкція, призначена для отримання даних із бази даних, називається **запитом до бази даних**. Він складається з критеріїв пошуку, записаних за допомогою мови запитів SQL. Текст SQL-запиту розташовується в сценаріях сторінок на стороні сервера або в тегах.



Сервер застосунків не може безпосередньо отримати дані з бази даних, оскільки вони використовують специфічні формати зберігання даних. Для підключення до бази даних сервер застосунків використовує посередника — драйвер бази даних.

Драйвер бази даних являє собою програмний модуль, за допомогою якого встановлюється взаємодія між сервером застосунків і базою даних.

Після того як драйвер установить з'єднання, виконується запит до бази даних, у результаті чого формується набір записів, який повертається серверу застосунків, і він використовує отримані дані для формування сторінки.

Для різних завдань використовуються різні зв'язки вебсервера (Nginx або Apache) і бази даних, такі як `php_fpm` або `spawn-fcgi`. Проте зазвичай обирається PHP + MySQL.

Розглянемо роботу такої зв'язки на прикладі 2. Це буде серверна підтримка розробленої нами форми реєстрації (див. завдання для самостійного виконання до § 4.2).

ПРИКЛАД 2

Відбувається надсилання форми з вебсторінки у вигляді сценарію JavaScript. JS AJAX отримує дані й надсилає їх на HTML-сервер, сервер отримує ці дані, а також певний файл PHP. Файл PHP установлює з'єднання з базою даних і виконує потрібні дії. Після виконання операції база даних повертає результат. PHP надсилає отримані дані назад до AJAX, де залежно від результату остаточно з'ясовуються наступні дії (якщо вже існує такий логін або набраний пароль не відповідає заявленому, то повертається повідомлення про невідповідність і прохання повторити спробу; при підтверджені облікового запису відкривається сторінка профілю).

Програмне забезпечення баз даних може працювати на тому самому фізичному комп'ютері, що й вебсервер. Але зазвичай це перше, що потрібно для розміщення на окремому фізичному обладнанні, коли сайт повинен масштабуватися.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Назвіть завдання серверної сторони.
2. Чим відрізняються back-end і front-end?
3. Які мови програмування використовують back-end-програмісти?
4. Наведіть приклади використання серверів.
5. Опишіть послідовність роботи серверів.
6. Пригадайте розвиток реляційних баз даних або знайдіть відомості в інтернеті.



Дивіться презентацію «Вебсервер та бази даних. Взаємодія клієнт-сервер».





4.5

Взаємодія «клієнт — сервер»

Клієнт і сервер взаємодіють один з одним у мережі інтернет або в будь-якій іншій комп'ютерній мережі за допомогою різних мережевих протоколів, наприклад IP, HTTP, FTP та ін. Протоколів, як ви знаєте, досить багато, і кожен із них дозволяє надавати певну послугу.

Розглянемо приклад.

ПРИКЛАД

За допомогою протоколу HTTP браузер надсилає спеціальне HTTP-повідомлення, в якому зазначається, яку інформацію та в якому вигляді він має отримати від сервера. Отримавши його, сервер у відповідь надсилає подібне за структурою повідомлення (або кілька), у якому міститься вся потрібна інформація, зазвичай це HTML-документ.

Повідомлення, які надсилають клієнти, отримали назву **HTTP-запити**. Вони містять певні методи, які вказують серверу, як саме обробляти повідомлення. Повідомлення, які надсилає сервер, отримали назву **HTTP-відповіді**. Крім корисної інформації, вони містять спеціальні коди стану, що дозволяють браузеру дізнатись, як сервер зрозумів його запит. Взаємодію клієнтів та серверів відображенено на рис. 4.13.



Рис. 4.13. Взаємодія «клієнт — сервер»

Користувач вводить ім'я і пароль на початку сеансу роботи із сервером. Сервер перевіряє ім'я і пароль і зберігає інформацію про користувача на цей сеанс. У певний момент користувач або сервер завершує сеанс. Для подальшого доступу користувачеві необхідно ініціювати новий сеанс, що вимагає повторного введення імені та пароля.

Вебсервер може обмежувати доступ користувачів до певних ресурсів. Керування доступом потребує поєднання автентифікації та авторизації.

Автентифікація (перевірка справжності) визначає користувача, який ініціював запит, а під час перевірки повноважень з'ясовується, чи може

користувач мати доступ до певного ресурсу. Більшість систем «клієнт — сервер» автентифікують користувача, запитуючи в нього ім'я і пароль. У цьому випадку на сервері розташовано файл, який містить імена і паролі всіх зареєстрованих користувачів. Причому з метою захисту інформації паролі можуть зберігатися в зашифрованому вигляді.

Процес автентифікації дає можливість серверу ідентифікувати користувача, який звернувся з HTTP-запитом. Для керування доступом до вебресурсів сервер повинен реалізувати стратегію **авторизації** (перевірки повноважень). Йому необхідно мати ефективний спосіб визначення, які автентифіковані користувачі можуть мати доступ до певного ресурсу.

Подібні стратегії поведінки є складовою частиною налаштування сервера й зазвичай мають вигляд списків керування доступом із переліком користувачів, яким дозволено або заборонено доступ до ресурсу. Як саме створюються списки керування доступом і як зберігаються імена користувачів і паролі, залежить від програмного забезпечення сервера. Ресурси, що вимагають автентифікації, можуть бути розміщені в окремому каталогі з .php. Це прийнято для сторінок, які обробляються PHP (*Hypertext Preprocessor*).

Замість того щоб вбудовувати інформацію в HTML-файл, деякі програми можуть генерувати весь ресурс. У цьому випадку URL у HTTP-запиті відповідає програмі, а не документу. Програма може реалізувати різні завдання, такі як доступ до інформації в базах даних або створення відповіді, зміст якої залежить від клієнта, що звертається із запитом.

Веббраузери взаємодіють із вебсерверами за допомогою протоколу передавання гіпертексту (HTTP). Коли ви клацаете посилання на сторінці, заповнююте форму або запускаєте пошук, браузер надсилає на сервер HTTP-запит.

HTTP-запит включає:

- ◆ шлях, який визначає цільові сервер і ресурс (наприклад, файл, певна точка даних на сервері, що запускається, сервіс та ін.);
- ◆ метод, який визначає необхідну дію (наприклад, отримати файл, зберегти або відновити деякі дані тощо);
- ◆ методи передавання даних — GET (отримати певний ресурс, наприклад HTML-файл, що містить інформацію про товар або список товарів); POST (створити новий ресурс, наприклад нову статтю на Вікіпедії, додати новий контакт у базу даних);
- ◆ HEAD — отримати метадані про певний ресурс без отримання змісту, як робить GET. Можна використовувати запит HEAD, щоб дізнатися, коли в останній раз ресурс оновлювався, і тільки потім використовувати запит GET (більш «витратний»), щоб завантажити ресурс, який було змінено;
- ◆ PUT — оновити існуючий ресурс (або створити новий, якщо такого не існує);
- ◆ DELETE — вилучити вказаний ресурс.



Раніше вже наголошувалося на відмінностях роботи зі статичним і динамічним сайтами. Розглянемо взаємодію «клієнт — сервер» на прикладі статичного сайта (рис. 4.14).

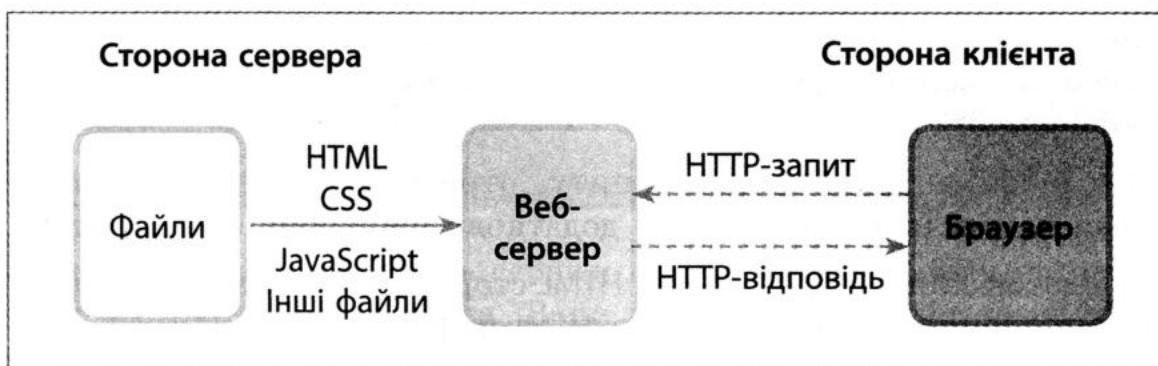


Рис. 4.14. Взаємодія «клієнт — сервер» на прикладі статичного сайта

Коли користувач хоче перейти на сторінку, браузер надсилає HTTP-запит GET із зазначенням URL-адреси його сторінки HTML. Сервер витягає запитуваний документ зі своєї файлової системи і повертає відповідь HTTP, що містить документ, і код стану HTTP «200 OK» (із зазначенням успіху). Сервер може повернути інший код стану, наприклад «404 Not Found», якщо файл відсутній на сервері, або «301 Moved Permanently», якщо файл існує, але був перенаправлений в інше місце.

Серверу для статичного сайта буде потрібно лише обробляти запити GET, тому що він не зберігає жодних даних, які модифікуються. Він також не змінює свої відповіді на основі даних HTTP-запиту.

Розглянемо порядок дій для роботи з динамічним сайтом (рис. 4.15).

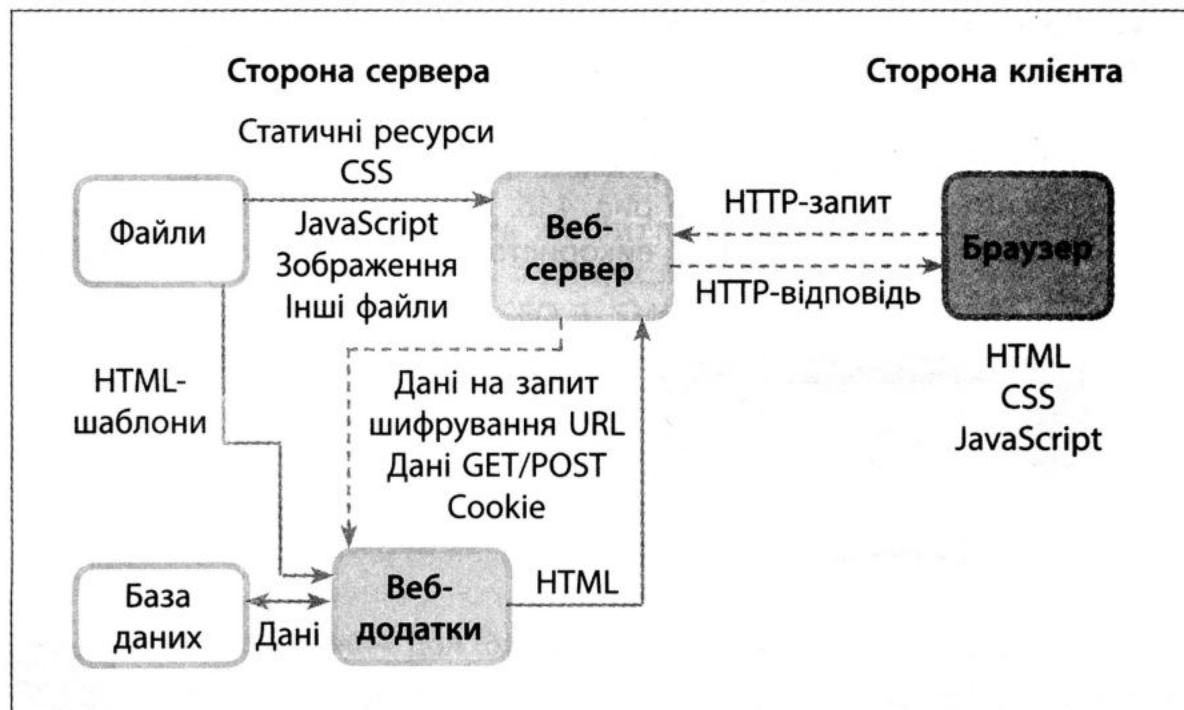


Рис. 4.15. Взаємодія «клієнт — сервер» на прикладі динамічного сайта

1. Веббраузер створює HTTP-запит GET на сервер з використанням базової URL-адреси ресурсу. Запит GET використовується, тому що запит — це лише вибірка даних (не зміна даних).
2. Вебсервер виявляє, що запит є «динамічним», і пересилає його у вебдодаток для обробки (вебсервер визначає, як обробляти різні URL-адреси на основі правил зіставлення шаблонів, визначених у його конфігурації).
3. Вебдодаток визначає мету запиту, отримує необхідну інформацію з бази даних (використовуючи додаткові «внутрішні» параметри).
4. Вебдодаток динамічно створює HTML-сторінку, поміщаючи дані (з бази даних) у наповнювачі всередині HTML-шаблону.
5. Вебдодаток повертає згенерований HTML у веббраузер (через вебсервер) разом із кодом стану HTTP 200 («успіх»). Якщо щось перешкоджає поверненню HTML, він поверне інший код, наприклад «404», щоб вказати, що <такого> посилання не існує.
6. Веббраузер починає обробляти повернутий HTML, надіславши окремі запити, щоб отримати інші файли CSS або JavaScript, на які посилається.
7. Вебсервер завантажує файли з файлової системи й повертає безпосередньо в браузер.

Операція з оновлення запису в базі даних оброблятиметься аналогічно. За винятком того, що, як будь-яке оновлення, HTTP-запит із браузера повинен бути закодований як запит POST.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке автентифікація та авторизація? У чому їх різниця?
2. Як називають повідомлення, сформовані клієнтом; сервером?
3. У чому різниця між методами GET і POST?
4. Опишіть механізм автентифікації користувача на сервері.
5. Поясніть повідомлення на рис. 4.16. Коли воно може з'явитися?
6. Які протоколи (крім HTTP) використовуються під час взаємодії «клієнт — сервер»?



Рис. 4.16. Код «404»



Дивіться презентацію «Вебсервер та бази даних. Взаємодія клієнт-сервер».





4.6

Валідація сайта і збереження даних форм



Валідація сайта — перевірка синтаксичних помилок, вкладеності тегів та інших критеріїв.

Зазвичай **валідатори** — це сервіси для перевірки сайтів на наявність помилок у структурі документа. Вони перевіряють HTML-код на відповідність певному стандарту, який зазначено на самому початку будь-якої HTML-сторінки першим рядком.

Валідація сайта дозволяє стежити за правильним відображенням сайта в різних браузерах. Розглянемо приклад 1.

ПРИКЛАД 1

Якщо не закрити тег або припуститися помилки в коді, у подальшому одна й та сама сторінка може відображатися в різних браузерах по-різному.

Крім того, пошукові системи віддають перевагу сайтам із валідним HTML-кодом, роботи пошукових систем розпізнають HTML-код за тими самими параметрами, що й браузери. Розглянемо приклад 2.

ПРИКЛАД 2

Якщо у вашому коді містяться помилки, наприклад незакриті теги, биті посилання або поламана структура, все це може вплинути на індексацію сайта пошуковими роботами. Як наслідок, грубі помилки в коді можуть знищити позиції ресурсу у видачі. Це означає, що в подальшому валідація сайта зіграє дуже важливу роль у просуванні сайта в SEO і SMM. Більше того, від валідації синтаксису навіть залежить коректність відображення всіх елементів сайта.



Якщо ви отримаєте повідомлення про те, що у вашому коді трапляються помилки чи навіть критичні попередження, то потрібно ще раз ретельно переглянути код, перевірити валідацію і провести сканування заново. Такий цикл слід повторювати, доки перевірка не дасть позитивного результату.

Потрібно також звернути увагу на кросбраузерність сайта. В ідеалі сайт повинен однаково відображатися у всіх браузерах. У нашому випадку цю опцію гарантує відповідність нормам W3C.

Більша частина проблем із кросбраузерністю виникає саме через помилки відображення оформлення елементів, заданих за допомогою CSS, оскільки для розв'язання проблеми коректного виведення дизайну сайта в усіх браузерах доводиться застосовувати не зовсім «валідні» (на думку W3C) способи:

- ◆ **коментарі** (коли в коментах (найчастіше для IE) прописується альтернативне значення властивості, невидиме для інших браузерів);
- ◆ **хаки** (маються на увазі спеціальні властивості CSS, що дозволяють розв'язати проблему некоректного відображення в одному з браузерів);
- ◆ за допомогою JavaScript (зміна стильової властивості елемента через об'єктну модель документа).

Застосування цих способів (особливо хаків) може негативно впливати на валідність усього сайта, разом із тим дозволяє повністю розв'язувати проблему кросбраузерності.

Ознайомимося з найбільш поширеними помилками, яких припускаються в коді, на прикладах.

- ◆ **Тег не закритий** — переважна більшість тегів у HTML є парними, тобто складаються з відкривального та закривального тегів. Часто в ході верстання чи написання скрипту забувають дописувати закривальний тег. А це може привести до неправильного відображення усього сайта.
- ◆ **Порушення вкладеності елементів** — ця проблема виникає в разі блокового верстання, коли не дотримуються ієрархічної вкладеності всіх блоків `<div>`, або інших парних елементів, якщо частина тега відкрита в одному шарі, а закривається в іншому. При цьому може статися порушення структури дизайну сайта. Наприклад:

Неправильно	Правильно
<code><p>Lorem</p></code>	<code><p>Lorem</p></code>

- ◆ **Використання атрибута style** — у сучасному сайтомбудуванні загальноприйнятою нормою є використання для оформлення елементів дизайну каскадних таблиць стилів (CSS). Застарілий атрибут style вживається вкрай рідко: в разі його використання обсяги HTML-коду можуть збільшуватися в рази. Це не лише сповільнює швидкість завантаження усього сайта, а й ускладнює його розуміння і подальше редагування.
- ◆ **Використання вкладеного CSS** — впровадження каскадних таблиць стилів всередину HTML. Це також збільшує обсяг коду й ускладнює його налагодження. Найкраще стильові описи зберігати в окремих файлах.



- ♦ **Missing DOCTYPE.** Валідатор найчастіше видає це повідомлення, якщо вміст елемента <!DOCTYPE> і сам елемент записані в нижньому регистрі. Необхідно пам'ятати, що цей елемент є залежним від регістру тегом. Наприклад:

Неправильно	Правильно
<!doctypehtmlpublic"-//w3c//dtdxhtml1.0 strict//en" "http://www.w3.org/tr/xhtml1/dtd/xhtml1-strict.dtd">	<!DOCTYPE html PUBLIC"-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

- ♦ **Thereisnosuchelement** — такого елемента не існує.

Така помилка виникає, якщо один або декілька тегів записано у верхньому регистрі. Наприклад:

Неправильно	Правильно
<P>Lorem</P>	<p>Lorem</p>

- ♦ **requiredattribute "alt" notspecified** — обов'язковий атрибут "alt" не вказано.

Як ми вже зазначали, з багатьох причин бажано використовувати атрибут "alt". Наприклад:

Неправильно	Правильно

- ♦ **Missing " "** — відсутні лапки.

Така помилка виникає, якщо значення одного або декількох атрибутів записані не в лапках. Наприклад:

Неправильно	Правильно
	

Найпоширеніший спосіб перевірити вихідний код сайта на валідацію — це онлайн-сервіси, оскільки вони постійно стежать за нововведеннями та модифікують свої алгоритми.

Посилання на онлайн-сервіси (рис. 4.17) ви можете знайти на головній сторінці офіційного сайта www.w3.org.

Форми також мають правила валідації. Наприклад, можна задати обов'язкові для заповнення поля форми або вказати, що в певні поля потрібно вводити дані тільки певного типу (наприклад, тільки букви або тільки цифри; введення email-адреси; введення URL-адреси тощо).



Рис. 4.17. Посилання для перевірки на сайті консорціуму W3C

Валідація (від латин. *validus* — сильний) — визнання чогось законним, дійсним, підтвердження на основі надання об'єктивних доказів того, що вимоги, призначені для конкретного використання або застосування, точно і в повному обсязі зумовлені, а мету досягнуто (означення зі стандарту ISO 9000).

Правила валідації забезпечують правильність заповнення форми від відувачем сайта. Після заповнення форми дані користувача потрапляють у спеціальний скрипт на сервері — **обробник форми**. У ньому вони можуть бути збережені в базі даних або надіслані електронною поштою.

Якщо користувач заповнив вебформу без помилок, то після надсилання даних на сервер відображається сторінка успішного заповнення форми або відбувається переадресація на заздалегідь задану сторінку.

Якщо ж користувач заповнює форму для опитування або голосування, то після успішного заповнення форми відразу ж відображається сторінка із загальними результатами голосування.

?

ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

- Що таке валідація сайта?
- Як валідність коду впливає на індексацію сайта?
- Наведіть приклади найбільш поширених помилок.
- Яка організація встановлює правила валідності коду?
- Наведіть приклади онлайн-сервісів перевірки на валідність.

✓

ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ

- Перевірте за допомогою будь-якого онлайн-сервісу сторінки, створені на попередніх уроках, на валідність.
- Проаналізуйте отриманий результат.



Дивіться презентацію «Валідація та збереження даних форм».



Практична робота № 12

ТЕМА. Перевірка сайта на валідність

ЗАВДАННЯ: перевірити правильність написання HTML- і CSS-коду за допомогою валідатора Консорціуму Всесвітньої Павутини.

ОБЛАДНАННЯ: комп’ютер із доступом до мережі інтернет, будь-який редактор коду (наприклад, Sublime Text, Atom, Notepad++).

Хід роботи

Під час роботи з комп’ютером дотримуйтесь правил безпеки.

Перевірка правильності HTML-коду

- Перейдіть за адресою validator.w3.org

Відкриється сторінка, на якій є три вкладки. На вкладці Validate by URI ви можете перевірити валідність сайта, розміщеного в інтернеті, на Validate by File Upload — завантажити файл із комп’ютера, на Validate by Direct Input — додати вміст файла безпосередньо у форму для введення (рис. 1).

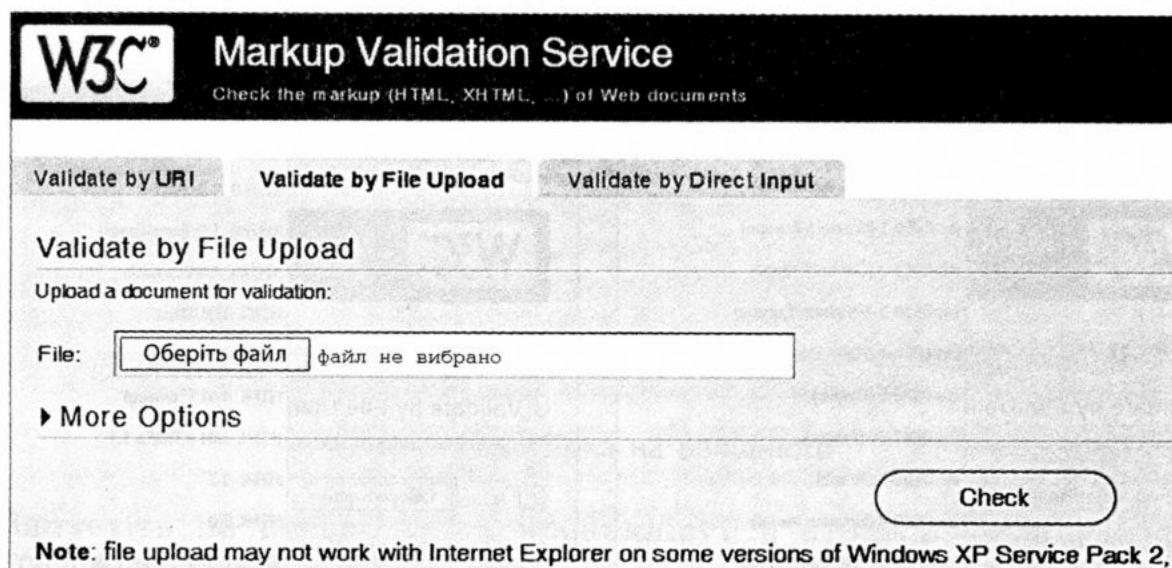


Рис. 1. Головна сторінка валідатора W3C

- Скористайтеся другим способом: завантажте файл із комп’ютера. Для цього виберіть вкладку Validate by File Upload і натисніть команду Виберіть файл — назва вибраного файла відобразиться у вікні (рис. 2).
- Додатково налаштуйте опції перевірки.
- Визначте кодування (рис. 3.) і тип документа (рис. 4). Можна вибрати визначати автоматично (detect automatically).
- Запустіть на перевірку за допомогою команди Check.

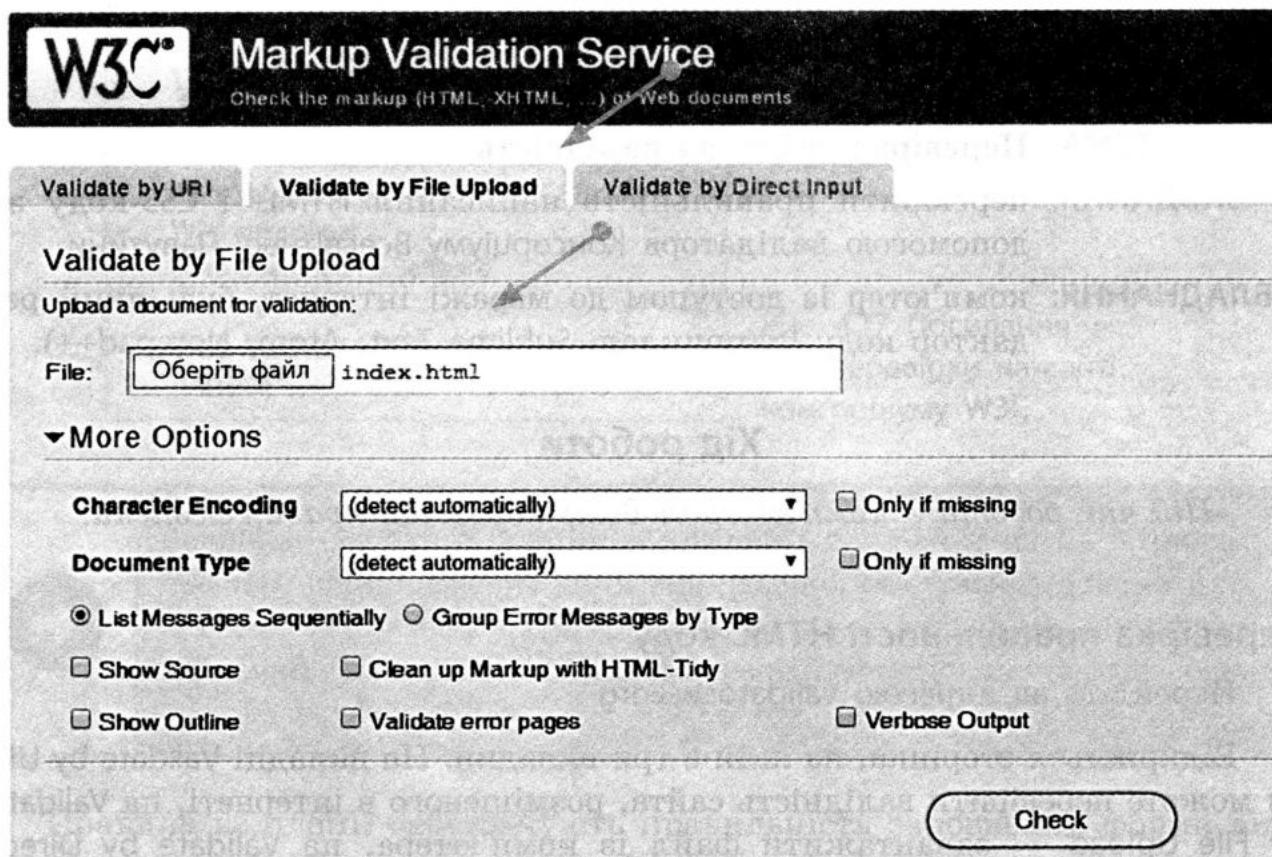


Рис. 2. Вкладка Validate by File Upload

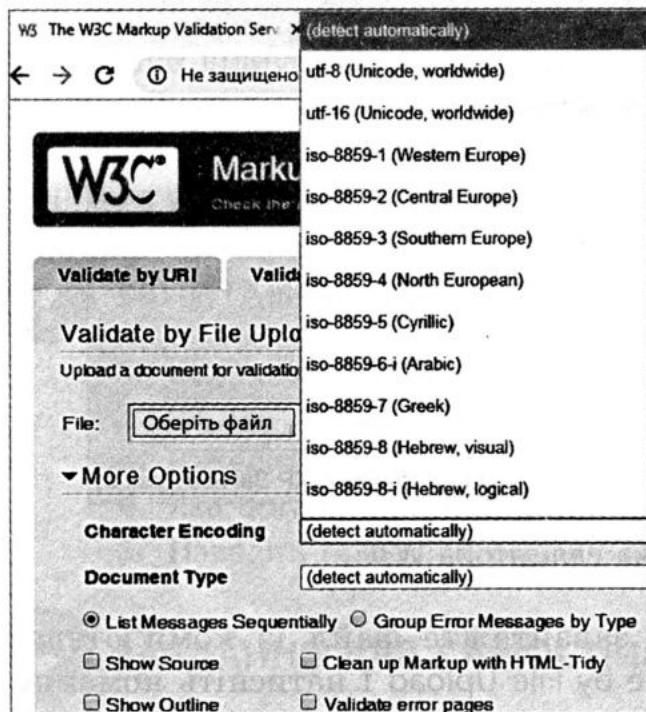


Рис. 3. Кодування

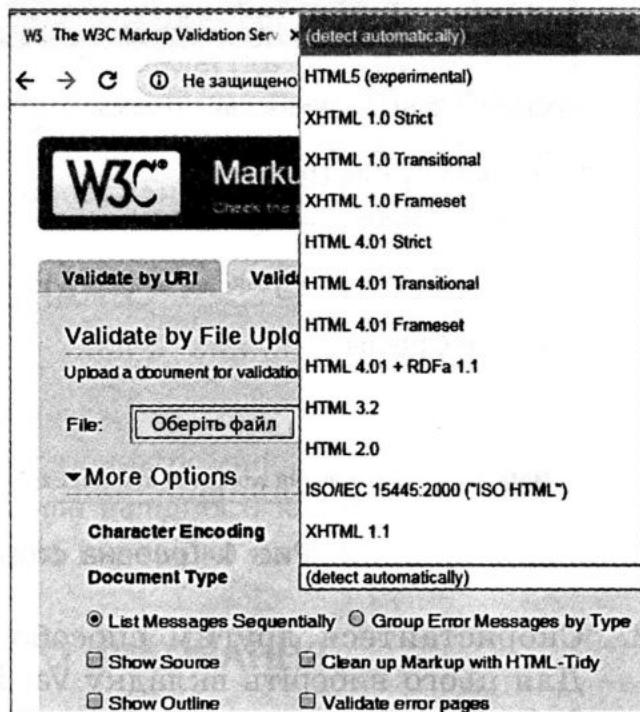


Рис. 4. Тип документа

- Проаналізуйте отримані попередження та помилки. Після виправлення знову застосуйте перевірку, повторюючи ці дії до появи повідомлення, аналогічного наведеному на рис. 5.

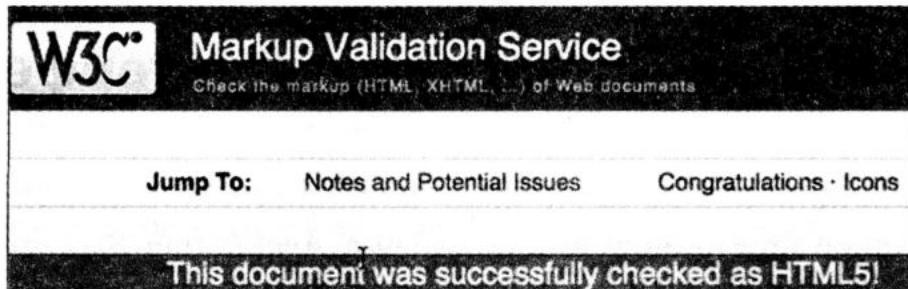


Рис. 5.
Повідомлення

Перевірка правильності CSS-коду

- Для перевірки на валідність CSS-коду перейдіть за посиланням <https://jigsaw.w3.org/css-validator/>.
- Сервіс аналогічно validator.w3.org надає три можливості перевірки коду.
- Виберіть другий спосіб (рис. 6).

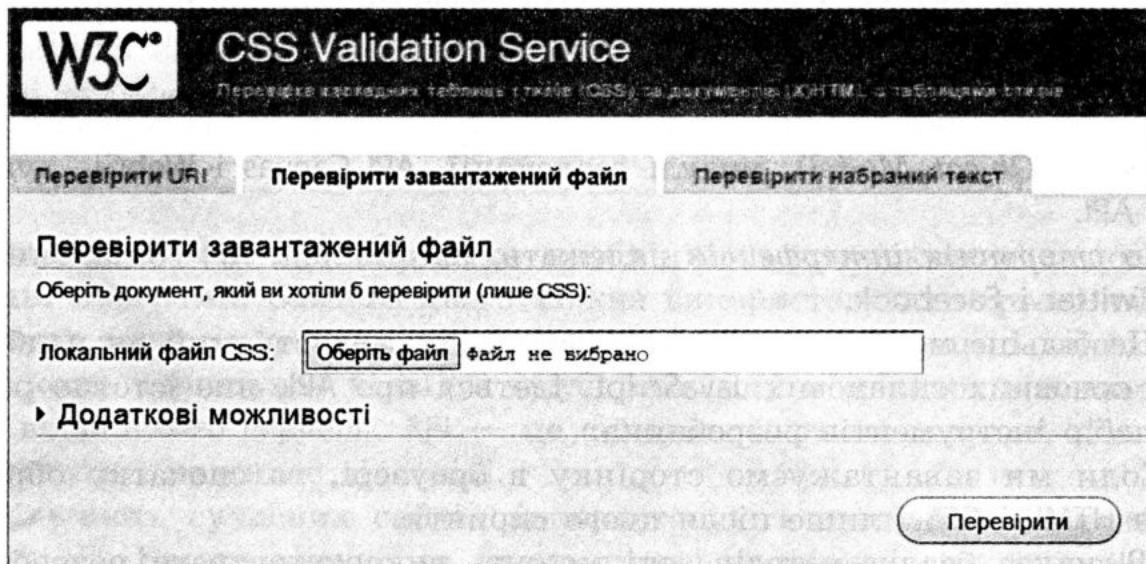


Рис. 6. Перевірка на валідність

- Запустіть на перевірку та проаналізуйте отримані повідомлення. При появі помилки виправте її й знову активізуйте перевірку. Продовжуйте повторювати дії до появи повідомлення про відсутність помилок.
- Об'єднайтесь у малі групи (по дві-три особи). Обговоріть, яких однакових помилок припустилися всі члени вашої групи. Чи були помилки, виправлення яких викликає труднощі? Поміркуйте, чи можливо уникнути помилок і створити сторінку, яка пройде перевірку на валідацію з першого разу? Обґрунтуйте свою думку. Результати обговорення надайте вчителю у вигляді текстового файла.

Зробіть висновок за результатами роботи.

4.7

Прикладний програмний інтерфейс



Прикладний програмний інтерфейс (англ. *Application Programming Interface*, скорочено API) — це сукупність засобів і правил, які уможливлюють взаємодію між окремими складниками програмного забезпечення або між програмним і апаратним забезпеченням.

У галузі веброзробки поняття прикладного програмного інтерфейсу (API) охоплює низку засобів програмного коду (методи, події та посилання). Розробник може використовувати їх у власних застосунках задля взаємодії з програмним або апаратним оточенням — із додатками вебпереглядача, іншим програмним чи апаратним забезпеченням комп’ютера користувача або навіть сторонніми сайтами чи службами.

API — це готові модулі коду, які допомагають програмісту в реалізації деяких складних завдань. Зазвичай такі «заготовки» діляться на браузерні і API третіх розробників.

До *браузерних API-інтерфейсів* належать: API-інтерфейс DOM (*Document Object Model*), модулі геолокації, API Canvas і WebGL, аудіо та відео API.

До *сторонніх інтерфейсів* належать, наприклад, API соціальних мереж Twitter і Facebook.

Ще більше можливостей надає функціонал, доступний як надбудова щодо основних складових JavaScript. Ідеться про API, яке істотно розширяє набір інструментів розробника.

Коли ми завантажуємо сторінку в браузері, то спочатку обробляються HTML і CSS і лише після цього скрипти.

API надає безліч методів, які можуть використовувати розробники. Розробнику не обов’язково знати, як працює система всередині, він просто може використовувати певний функціонал у своєму застосунку.

API-інтерфейси дозволяють розробникам заощадити час, скориставшись вже готовим функціоналом. Це допомагає зменшити кількість виробленого коду, створювати деяку послідовність для різних застосунків на тій самій платформі. API-інтерфейси можуть контролювати доступ до апаратних і програмних ресурсів.

Програміст має змогу скористатися API для отримання доступу до функціоналу сторонньої програми. API робить можливим роботу ресурсів, які використовують потенціал і потужність іншого сайта або програми.

Як відомо, на онлайн-сервіси або платформи можна увійти через власні облікові записи в соціальних мережах. Саме це і є використанням API, коли сервіси або застосунки використовують бази даних соціальних мереж. Сервіс може отримувати інформацію про користувача і використовувати її у своїх цілях.



Розглянемо приклади 1–3.

ПРИКЛАД 1

Amazon пропонує користувачеві книжки, перелік яких ґрунтуються на виборі книжок його друзями у Facebook.

ПРИКЛАД 2

Сервіс IFFTT дозволяє пов'язувати акаунти користувача в різних онлайн-сервісах і програмах так, що дія в одній програмі викликає дію в іншій.

ПРИКЛАД 3

Відео, що сподобалися вам у YouTube, можуть автоматично з'являтися на вашому сайті або у ваших соціальних мережах. Це можливо саме завдяки API — коли одна програма використовує дані та інформацію іншої програми. До речі, вбудовання YouTube-відео на свій сайт також можливе завдяки API-сервісу YouTube.

Використання API скорочує необхідність створювати самостійно складні програми. Замість цього можна використовувати готові частини існуючих ресурсів, у яких є доступ до потрібної інформації і даних. Щоразу, коли користувач відвідує якусь сторінку в мережі, він взаємодіє з API віддаленого сервера. API — це складова частина сервера, яка отримує запити і надсилає відповіді.

Більшість сучасних сайтів використовують принаймні кілька сторонніх API. Багато задач вже мають готові рішення, запропоновані сторонніми розробниками, будь то бібліотека чи послуга. Часто простіше і надійніше вдатися саме до вже готового рішення.

Як вже було сказано, API — це насамперед інтерфейс, який дозволяє розробникам використовувати готові блоки для побудови програми. У випадку з розробкою мобільних додатків у ролі API може виступати бібліотека для роботи з розумним будинком — всі нюанси реалізовані в бібліотеці, і ви лише звертаєтесь до цього API у своєму коді.

Щодо вебдодатків API може передавати інформацію у відмінному від стандартного HTML форматі, завдяки чому ним зручно користуватися під час написання власних програм.

Сторонні загальнодоступні API найчастіше віддають дані в одному з двох форматів: XML або JSON.

XML (англ. *Extensible Markup Language* — розширення мова розмітки) — запропонований World Wide Web Consortium (W3C) стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між

різними застосунками. XML-документ складається із текстових знаків і придатний до читання людиною.

JSON (англ. *JavaScript Object Notation*) — запис об'єктів JavaScript, текстовий формат обміну даними між комп'ютерами. Він базується на тексті й може бути прочитаний людиною. Формат дозволяє описувати об'єкти та інші структури даних. Цей формат головним чином використовується для передавання структурованої інформації через мережу.

Формат JSON знайшов своє головне призначення у написанні вебпрограм, а саме при використанні технології AJAX.

JSON досить лаконічний і простий у читанні, дозволяє складні структури в атрибутах, займає менше місця і прямо інтерпретується за допомогою JavaScript в об'єкти, на відміну від XML. Сервіси, що надають доступ до даних у XML-форматі, поступово від нього відмовляються.

JSON, що використовується в AJAX, виступає як заміна XML (використовується в AJAX) під час асинхронного передавання структурованої інформації між клієнтом і сервером.

Розглянемо приклад 4.

ПРИКЛАД 4

Візьмемо API Twitter. Інтерфейс цього сервісу може вам видати інформацію про твіти користувача, його читачів і тих, хто його читає, і тощо. Це лише частина можливостей, які будь-хто може втілити, використовуючи API стороннього сервісу або створюючи власний.

На основі API будується такі речі, як карти 2GIS, всілякі мобільні й десктопні клієнти для Twitter.

Всі їх функції стали можливими саме завдяки тому, що відповідні сервіси мають якісні й детально задокументовані API.

API бувають публічними й приватними.

Публічні API випускаються такими компаніями, як Slack і Shopify, щоб розробники використовували їх на своїх платформах. Компанії діляться набором вступних параметрів, а розробники використовують їх, щоб досягти певного результату. Це досить легко, оскільки документація перебуває у вільному доступі.

Приватні API використовуються всередині компанії. Якщо в компанії багато програмних продуктів, то приватне API використовується, щоб програми взаємодіяли між собою. Компоненти API можуть змінюватися за бажанням компанії, тоді як зміни в публічному API можуть викликати протест.

Фреймворк (*framework* — конструкція, структура) — програмне середовище спеціального призначення, своєрідний каркас, який використовується для того, щоб істотно полегшити процес об'єднання певних компонентів під час створення програм.



Фреймворк — це основа, що дозволяє додавати компоненти залежно від потреб; база, на якій можна сформувати програму будь-якого призначення досить швидко і без особливих труднощів.

Якщо порівнювати динамічну бібліотеку (DLL), яка відрізняється велими обмеженим функціоналом, і фреймворк, що вважається основою програм, можна виділити суттєву перевагу фреймворків.

Саме фреймворк є сполучною ланкою, яка об'єднує всі використовувані програмні компоненти. Всередині фреймворка часто є необхідні тематичні бібліотеки.

Наведемо характеристики кількох фреймворків.

- ♦ **Semantic UI** (<https://semantic-ui.com>) — використовується для створення переносимих інтерфейсів. Цей досить молодий фреймворк постійно розвивається. У ньому можна знайти величезну кількість кнопок та інших елементів, необхідних для роботи, — зображення, іконки, написи.
- ♦ **Foundation** (<https://foundation.zurb.com>) — фреймворк, що є одним з досить популярних у сегменті front-end-фреймворків. Останні версії відрізняються поліпшеним функціоналом для сучасних мобільних пристройів. Завдяки семантичному підходу є можливість використання SCSS, написання більш чистого коду в HTML. Він ідеальний для ситуації, коли потрібне швидке прототипування.

Дуже популярним і затребуваним є фреймворк Bootstrap (<https://getbootstrap.com>). Його презентували ще на початку 2011 року. Його головна перевага — адаптивність (адаптивна верстка). Він дозволяє створювати проекти зі стильним дизайном — проект буде автоматично підлаштовуватися, враховуючи розмір екрана комп’ютера або мобільного пристрою користувача, що переглядає сайт.

До переваг Bootstrap належить велика кількість стилів, шаблонів, посторінковий дизайн — це істотно полегшує створення сайта. У Bootstrap практично відсутні недоліки. Це не тільки HTML/CSS-фреймворк, у Bootstrap також включені плагіни й готові стилі JS/Jquery.

У §4.2 ми розглядали HTML-елемент `<form>`, наголошуючи на тому, що його опрацювання сервером здійснюється за допомогою мов програмування PHP або Perl. В HTML5 з’являється ще один елемент — Canvas (в англомовній версії — Canvas API). Його специфікація визначає універсальний JavaScript API, який дозволяє створювати растрові зображення та анімацію за допомогою JavaScript.

У перекладі з англійської canvas означає «полотно». За замовчуванням елемент `<canvas>` створює прямокутну область, причому у нього немає ані власного контенту, ані рамок. Основними атрибутами є `width` і `height` (ширина та висота відповідно), якщо вони не оголошені, то набувають значень `300px` та `150px`.

Для розміщення елемента на сторінці HTML досить вказати:

```
<canvas width="600" height="250" id="drawingcanvas"></ canvas>
```

Після цього елементом `<canvas>` можна маніпулювати як завгодно: розміщувати в ньому текст, малювати графічні елементи і лінії, виконувати заливку, додавати анімацію. Все це робиться за допомогою команд JavaScript. Спочатку зображення створюється програмно, а потім результат виводиться візуально.

Важливо пам'ятати: розміри полотна завжди слід встановлювати за допомогою атрибутів `width` і `height` елемента `<canvas>`, а не за допомогою властивостей `width` і `height` таблиці стилів. Інакше можуть виникнути проблеми з відображенням малюнків. Ще раз наголосимо, що полотно відображається як порожній прямокутник без рамки, тобто його не видно взагалі. Щоб зробити полотно видимим, за допомогою таблиці стилів йому можна надати кольорове тло або рамку, наприклад:

```
canvas { border: 2px dashed grey;}
```

Тепер необхідно скористатись командами мови JavaScript. Щоб малювати на полотні, треба спочатку отримати об'єкт за допомогою методу `document.getElementById`. Потім застосовують метод `getContext()`, вказавши, що потрібен двовимірний контекст малювання. Таким чином, будь-яка робота з елементом `<canvas>` починається з двох команд:

```
var canvas=document.getElementById("drawingCanvas");
var context=canvas.getContext("2d");
```

Для створення малюнків необхідно знати особливості системи координат елемента `<canvas>`.

Координати лівої верхньої точки полотна, яка називається вихідною точкою, $(0, 0)$. Значення абсциси (тобто x -координати) збільшується при переміщенні вправо від вихідної точки, а значення ординати (тобто y -координати) — при переміщенні вниз. Таким чином, для полотна розміром в 500×300 пікселів координати кінцевої точки (правого нижнього кута) полотна будуть $(500, 300)$ (рис. 4.18).

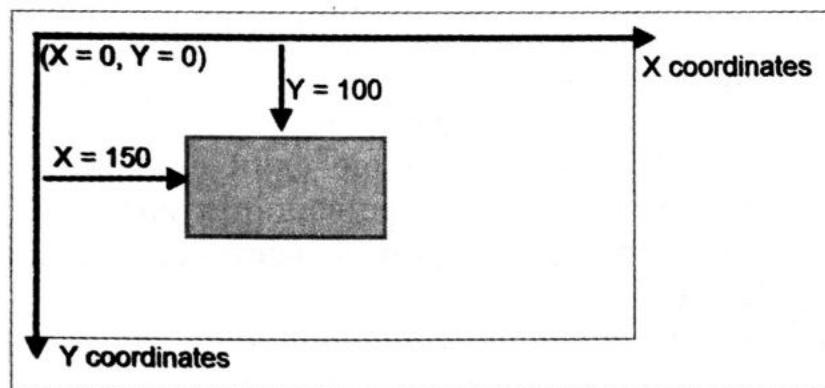


Рис. 4.18. Координати полотна `canvas`

Розглянемо принцип малювання за допомогою елемента `<canvas>` на прикладі найпростішої фігури — лінії.

Вказуємо початкову точку лінії за допомогою методу `moveTo(x1,y1)`. Метод `lineTo(x2,y2)` задає кінцеву точку лінії, а метод `stroke()` власне малює лінію.

Наприклад:

```
context.moveTo(20,20);
context.lineTo(300,50);
context.stroke();
```

Перед викликом методу `stroke()` можна встановити три властивості контексту малювання: `lineWidth`, `strokeStyle` та `lineCap`.

Властивість `lineWidth` визначає товщину лінії в пікселях.

Властивість `strokeStyle` визначає колір ліній. Її значення може бути у вигляді назви кольору HTML, коду кольору HTML або CSS-функції `rgb()`, у вигляді градієнтного зафарбування або візерунка з зображень.

`lineCap` вказує тип кінців лінії:

- ◆ `butt` (за замовчуванням, кінці лінії прямокутної форми);
- ◆ `round` (округлі кінці);
- ◆ `square` (прямокутні, але з подовженням на кожному кінці на половину значення товщини лінії).

Наприклад (рис. 4.19):

```
context.beginPath();
context.lineWidth=10;
context.strokeStyle="rgb(16,155,252)";
context.moveTo(10,120);
context.lineTo(400,120);
context.lineCap="round";
context.stroke();
```

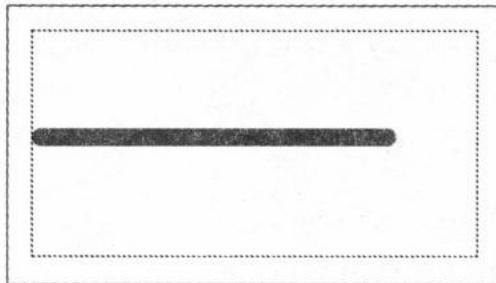


Рис. 4.19. Приклад лінії, побудованої за допомогою елемента <canvas>

Метод `beginPath()` починає новий, окремий шлях малюнка. Його слід викликати, тільки коли треба почати новий шлях, наприклад при зміні параметрів лінії або нової фігури.

Щоб відокремити лінії одну від одної, дляожної необхідно створювати новий шлях, тобто викликати метод `beginPath()`. Цей підхід дозволяє присвоїти кожній лінії свій колір, а також товщину і тип закінчення. Важливість шляхів також полягає в тому, що вони дозволяють заповнювати кольором фігури.

Спочатку використовуємо вже відомий нам алгоритм для малювання лініями фігури. Коли необхідно зафарбувати внутрішню область цієї фігури, потрібно закрити поточний шлях за допомогою методу `closePath()`, вибрати колір заливки, встановивши значення властивості `fillStyle`, а потім викликати метод `fill()`, щоб виконати власне заливку.

Розглянемо приклад створення трикутника (рис. 4.20):

```
context.moveTo(250,50);
context.lineTo(50,250);
context.lineTo(450,250);
context.closePath();

// Заливка
context.fillStyle="#109bfc";
context.fill();

// Контур
context.lineWidth=10;
context.strokeStyle="orange";
context.stroke();
```

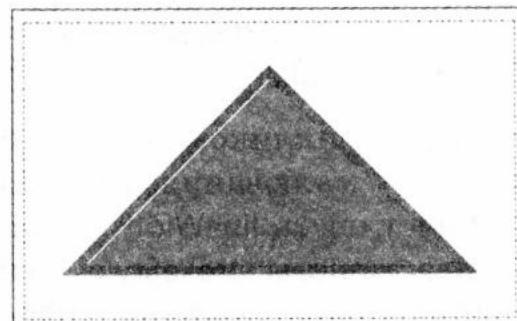


Рис. 4.20. Приклад малювання трикутника

5 кроків для роботи з елементом <canvas>

1. Створити елемент полотна: дати йому ідентифікатор, ширину, висоту (HTML).
2. Додати базові стилі — центрувати полотно, додати колір тла тощо (CSS).
3. У JavaScript отримати елемент полотна, використовуючи id.
4. Обрати контекст.
5. Використати контекст для малювання.

Це лише елементарні прийоми малювання за допомогою елемента <canvas>. Більш детально про роботу з ним можна дізнатись за посиланням: https://developer.mozilla.org/uk/docs/Web/API/Canvas_API/Tutorial.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Наведіть означення API.
2. Запропонуйте власні приклади застосування API.
3. Наведіть переваги прикладного програмного забезпечення.
4. Що таке фреймворк? Наведіть приклади.
5. Знайдіть в інтернеті відомості про використання JSON, підготуйте повідомлення.
6. Як можна пов'язати інтерактивність сторінок та використання API?

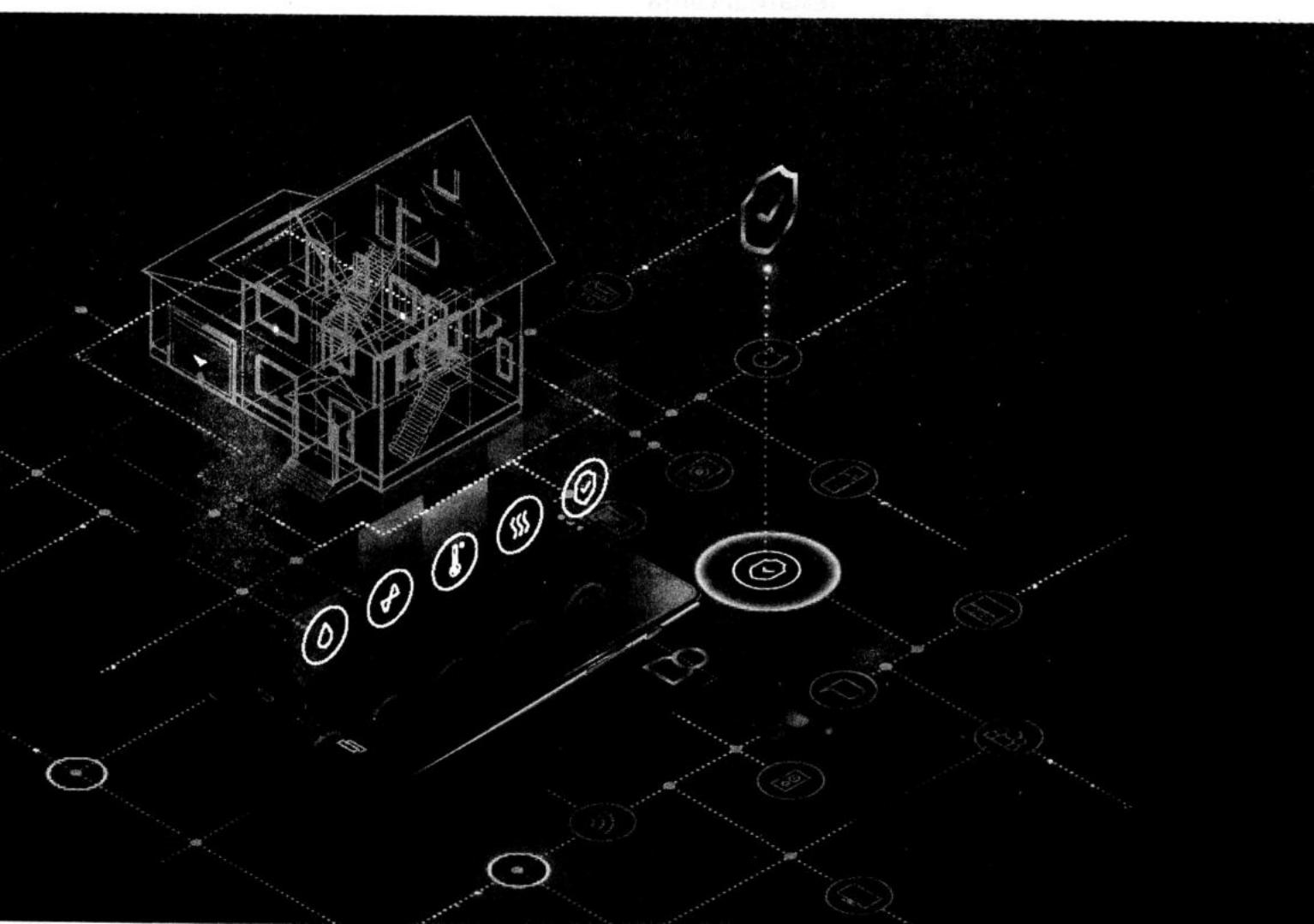


Дивіться презентацію «Прикладний програмний інтерфейс».

Розділ 5

ОСНОВИ ДИЗАЙНУ ТА ПРОСУВАННЯ ВЕБСАЙТА

- 5.1** Правила ергономічного розміщення відомостей на вебсторінці
- 5.2** Пошукова оптимізація та просування вебсайтів



5.1

Правила ергономічного розміщення відомостей на вебсторінці

Основна мета створення сайта — його спрямованість на користувача, тому зручність використання, зрозумілість і простота є досить важливими аспектами розробки сайта.

Потрібно створити умови, щоб користувач мав змогу інтуїтивно пов'язувати дії, які йому необхідно виконати на вебсторінці.



Ергономіка (від грец. *ergon* — робота, *potos* — закон) — наука, яка вивчає робочі процеси з метою створення оптимальних умов праці, що сприяє підвищенню її продуктивності.

Ергономічний сайт має бути добре пристосований для зручної та безпечної роботи користувача. Щоб цього досягти, потрібно користуватися певними правилами (рис. 5.1).

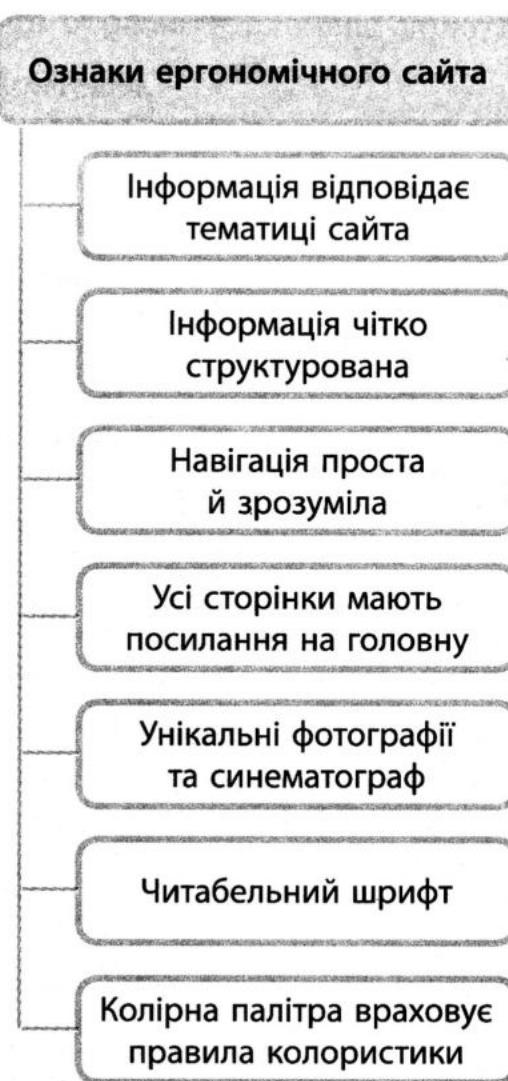


Рис. 5.1. Правила створення ергономічного сайта



Юзабіліті (англ. *usability* — зручність і простота використання) — це підхід, покликаний зробити сайт простим у користуванні, таким, що не потребує додаткового навчання користувача, тобто має орієнтований на нього інтерфейс.

Деякі загальні цілі юзабіліті:

- ◆ подавати інформацію в зрозумілій і стислій формі;
- ◆ створювати користувачам можливість робити вибір найочевиднішим шляхом;
- ◆ усувати будь-яку двозначність щодо наслідків дій (наприклад, кнопка Видалити/Покупка);
- ◆ розміщувати важливі елементи у відповідній ділянці на вебсторінці або вебзаєсунку.

Розглянемо основні критерії ергономічного розміщення відомостей під час створення сайта:

Критерій	Склад
Лаконічність	Простота викладення
	Неперевантаженість
Чіткість	Ясність
	Структурованість
	Розташування
	Видимість адреси
	Однорідність структури
Швидкість	Час завантаження
	Оптимізовані зображення
Взаємодія	Гіпертекстові посилання
	Сегментація інформації
	Сприяння взаємодії
Адаптивність	Можливість змінити розмір шрифту
Доступність	Доступ до всього
	Взаємодія
	Принцип прозорості
	Підпис

Закінчення таблиці

Критерій	Склад
Доступність	Вибір кольору
	Правильне використання стилів
	Контраст
	Можливість змінити розмір шрифту

Важливо пам'ятати, що вся першочергова інформація повинна бути зліва. Читач знайомиться з нею зліва направо, тому потрібно ретельно все продумати. Таким чином, створення сайта з урахуванням ергономіки може бути визначено як здатність ефективно реагувати на потреби користувачів і забезпечувати їм комфорт під час перегляду сторінки.

Колірна палітра є одним із найважливіших елементів сайта. Існує безліч онлайн-сервісів, які допомагають дібрати палітру кольорів сайта, що відповідає правилам колористики (рис. 5.2 і 5.3).



Рис. 5.2. Логотипи деяких онлайн-сервісів



Рис. 5.3. Палітри in color balance



Найпопулярніші онлайн-сервіси можна знайти за посиланнями:

- ◆ ColorExplorer (<http://colorexplorer.com/>);
- ◆ cOLORotate (<http://mobile.colorotate.org/>);
- ◆ ColorBlender (<http://www.colorblender.com/>);
- ◆ ColorWizard (<http://www.colorsontheweb.com/Color-Tools/Color-Wizard>).



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке юзабіліті; ергономіка? Що, на вашу думку, їх відрізняє?
2. Наведіть правила створення ергономічного сайта.
3. Які онлайн-сервіси добору палітри ви знаєте?
4. Чому важливо правильно добирати кольори?
5. Виберіть сайт і проаналізуйте його з точки зору юзабіліті.
6. Проаналізуйте тренди поточного року та оцініть їх із точки зору ергономіки.

5.2

Пошукова оптимізація та просування вебсайтів

Будь-який сайт створюється для того, щоб його відвідували якомога більше користувачів.

SEO (від англ. *Search Engine Optimization* — пошукова оптимізація) — маркетингове поняття, що охоплює цілий комплекс заходів. Це процес коригування HTML-коду, структури та текстового наповнення (контенту) сайта; контроль зовнішніх чинників на відповідність вимогам алгоритму пошукових систем.

Пошукова оптимізація може бути постійним джерелом збільшення кількості відвідувачів, адже 90 % користувачів знаходять нові сайти через пошукові системи. 55 % онлайн-покупок здійснюються на сайтах, які знайдено через пошукові системи.

Мета SEO — підвищення рейтингу сайта в пошукових системах (Google, Bing, Yahoo) та залучення трафіку.

Завдання SEO — підняти вебресурс у топ за результатами пошукової видачі за конкурентними запитами користувачів.

Складові SEO наведено на рис. 5.4. Що вищою є позиція сайта в результатах пошуку, то більшою є ймовірність переходу відвідувача на нього з пошукових систем, бо зазвичай люди ідуть за першими посиланнями.



Рис. 5.4. Складові SEO

Роботи з позиціонування сайта в пошукових системах — це один із найважливіших заходів щодо залучення цільової аудиторії. Особу, яка проводить роботу з оптимізації вебсайтів, називають *оптимізатором* (SEO-Manager).

Основною пошуковою системою у світі нині є Google. Популярність Google у більшості країн становить понад 60 %, в окремих європейських країнах — понад 90 %. Алгоритм розрахунку авторитетності, який використовує Google, це PageRank. Він застосовується до бази документів, пов’язаних гіперпосиланнями, — призначає кожному документу чисельне значення, що характеризує його «авторитетність» (що більше посилань на сторінку, то важливішою вона є).



Пошукова система — повністю автоматизований механізм, який глибоко сканує всі задані сервери (відкриті для сканування) і збирає індекс-інформацію про те, що і де (на якій вебсторінці) виявлено.

Зібрана інформація вноситься до бази даних пошукової системи, де алгоритм із ранжування реалізується у два етапи. Спочатку по сайтах «проходить» так званий «швидкобот» та індексує їх для того, щоб додавати новини на видачу пошукових систем, а потім (здебільшого протягом доби) — основний бот, який уже повністю індексує статтю.

Українські розробники створили такі сервіси для пошуку інформації:

- ◆ Sova.com.ua
- ◆ Gala.Net
- ◆ Online.ua
- ◆ MetaPing
- ◆ Аванпорт
- ◆ Мета
- ◆ UAport
- ◆ Атлас UA
- ◆ I.ua
- ◆ Ukr.net
- ◆ Meta.ua
- ◆ Poshukach.com
- ◆ Shukalka.com.ua

Оптимізація та просування сайтів у пошукових системах (рис. 5.5) мають забезпечити використання ключових слів у контексті, заголовках і мета-даних.

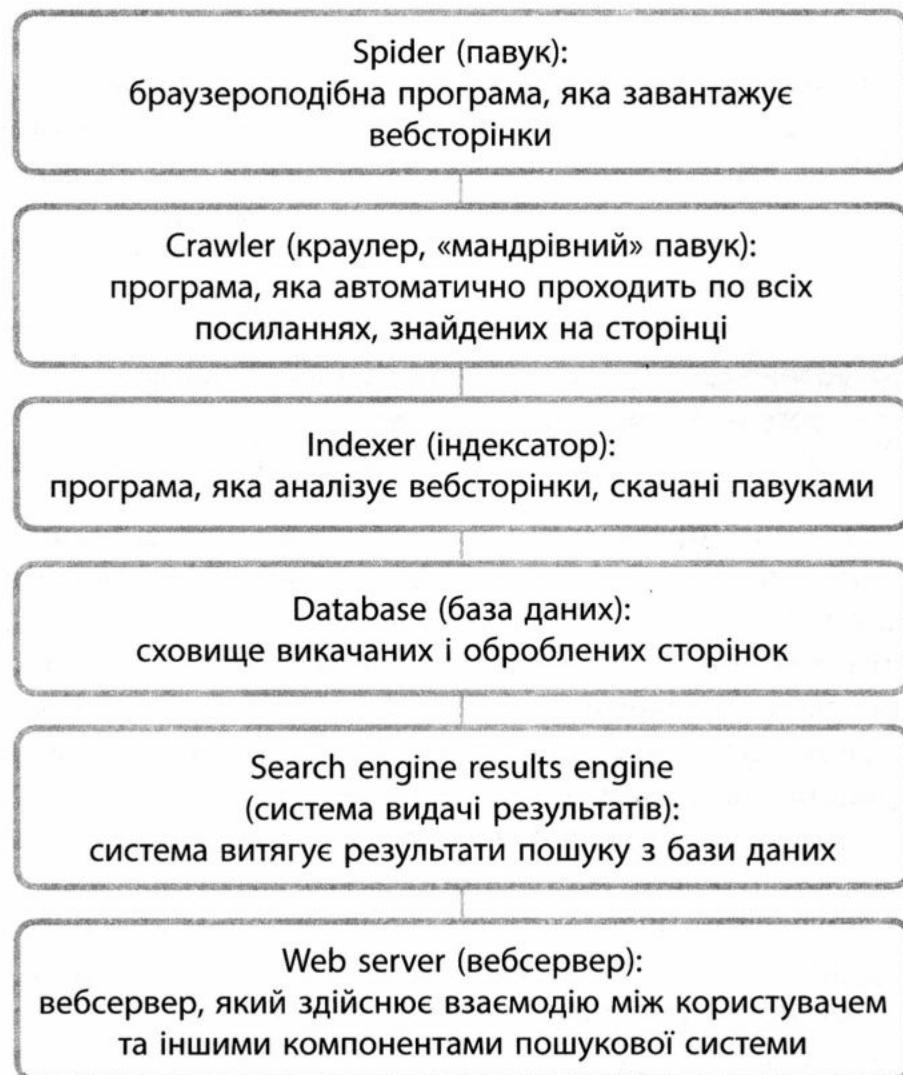


Рис. 5.5. Робота пошукової системи



Пошуковий робот (*webcrawler*, *bot*, *webrobots*, *webspider* — бот, павук) — це спеціальна програма, що є складовою частиною пошукової системи та призначена для перебiranня сторінок інтернету з метою занесення інформації про них у базу даних пошукової системи.

За введеними словами пошукова система шукає сторінки, зіставляє їх і ранжує, визначає релевантність конкретної сторінки за введеним запитом (тобто наскільки повно документ відповідає критеріям, зазначенним у запиті користувача).

Пошукова оптимізація передбачає такі етапи.

Етап

1

Збирання семантичного ядра. Під час добирання всіх слів вибираються ключові словоформи, що додаватимуть відвідувачів на сайт. Водночас здійснюється аналіз ринку сайтів, які наявні в топ-видачі. Збирання ключових слів для написання внутрішніх текстів можна виконати за допомогою як платних, так і безкоштовних програм.

Закінчення таблиці

Етап

2

Написання на основі семантичного ядра контенту, що відповідає запитам користувачів і розв'язує проблеми відвідувачів сайта. Водночас кількість ключових слів у співвідношенні до загальної кількості тексту не має перевищувати 3–5 %.

 **Просування сайта** — комплекс заходів щодо збільшення відвідуваності вебресурсу цільовими відвідувачами.

Метою будь-якого просування є збільшення співвідношення відвідувачів сайта, які вчинили очікувану дію, до всіх відвідувачів (подається у відсотковому вираженні).

Аналіз дій користувачів на сайті здійснюється за допомогою спеціального програмного забезпечення. Для Google — це Google Analytics. За даними досліджень, на другу сторінку переходят не більше ніж 85 % користувачів, далі другої — не більше від 10 %.

Власник вебресурсу (комерційна компанія, державна організація, соціальна мережа, ігровий майданчик, клуб за інтересами тощо) намагається, щоб його сайт було «видно» користувачам інтернету, аби мати змогу зібрати якомога більше відвідувачів. Таким чином він прагне забезпечити собі потрапляння на перші сторінки (а ліпше — на перші рядки, у топ) пошукової видачі. Розв'язання цього завдання забезпечує процес SEO-просування.

Існує низка прийомів, які дозволяють маніпулювати пошуковою системою. Вони розрізняються за своєю коректністю й легальністю. У зв'язку з цим виникло три класи пошукової оптимізації сайтів: чорна, сіра та біла. Ознайомимося з їхнім призначенням.

Оптимізація	Призначення
Чорна	Набір прийомів, які характеризуються явною некоректністю. Багато з них заборонені, деякі створені для введення пошукової системи в оману
Сіра	Не заборонені, але потенційно некоректні прийоми. Якщо чорну або сіру оптимізацію буде виявлено пошуковою системою, до сайта будуть застосовані штрафні санкції чи навіть бан
Біла	Використовує легальні професійні методи, що доповнюють один одного і дають стабільний результат. Спрямована на те, щоб і відвідувачі, і пошукові машини ставили сайту високу оцінку

Існує кілька розповсюджених методів просування сайта.



До наймолодших і перспективних методів із використанням можливостей, що відкриваються соціальним мережам, належать **SMO (Social Media Optimization)**, або **SMM (Social Media Marketing)**.

Методи SMO (SMM) просування сайта засновані на самостійній передачі даних від одного користувача до іншого за допомогою сервісів соціальних мереж (рис. 5.6).



Рис. 5.6. Способи SMO (SMM) просування сайта

Ще одним методом просування сайта є так званий **вірусний маркетинг** — тут головними розповсюджувачами інформації є самі користувачі. Один користувач передає інформацію з посиланням на ресурс знайомим, кожен із яких — кільком своїм. Таким чином, спрацьовує ефект «снігової лавини», що приносить результат у вигляді тисяч, а часом і сотень тисяч унікальних відвідувачів.

Основним інструментом вірусного маркетингу є створення цікавого, несподіваного, креативного контенту, який гарантовано зможе зацікавити максимальну кількість користувачів таким чином, аби змусити їх поділитися інформацією з іншими.

Яскравим прикладом вірусного маркетингу є розміщення цікавих відеороликів на YouTube і подібних до нього сервісах, де як джерело відео вказується ресурс, що просувається.



ЗАПИТАННЯ ДЛЯ ПЕРЕВІРКИ ЗНАНЬ

1. Що таке SEO?
2. Назвіть завдання SEO.
3. Що таке пошуковий робот? Наведіть схему роботи робота Google.
4. Назвіть класи оптимізації сайтів.
5. Що таке SMM?
6. Які ви знаєте способи просування сайта? Опишить їх.

Література

1. Інформатика : навчальна програма вибірково-обов'язкового предмету для учнів 10–11 класів загальноосвіт. навч. закл. (рівень стандарту). — Міністерство освіти і науки України, 2018. — 62 с.
2. Інформатика для загальноосвітніх навчальних закладів з поглибленим вивченням інформатики : підруч. для 9 кл. загальноосвіт. навч. закл. / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. — Харків : Вид-во «Ранок», 2017. — 240 с.
3. Інформатика (рівень стандарту) : підруч. для 10 (11) кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. — Харків : Вид-во «Ранок», 2018. — 160 с.
4. Інформатика (профільний рівень) : підруч. для 10 кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. — Харків : Вид-во «Ранок», 2019. — 256 с.
5. Інформатика (профільний рівень) : підруч. для 11 кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. — Харків : Вид-во «Ранок», 2019. — 256 с.

Зміст

Передмова	3
РОЗДІЛ 1. Напрямки та інструменти вебдизайну	
1.1 Основні тренди у вебдизайні	6
1.2 Види сайтів та цільова аудиторія	12
1.3 Інформаційна структура сайта	22
<i>Практична робота №1</i>	26
1.4 Інструменти веброзробника	29
РОЗДІЛ 2. Проектування та верстка вебсторінок	
2.1 Мова гіпертекстової розмітки.....	34
<i>Практична робота №2</i>	42
2.2 Каскадні таблиці стилів	44
<i>Практична робота №3</i>	51
2.3 Проектування та верстка вебсторінок.....	53
<i>Практична робота №4</i>	62
2.4 Адаптивна верстка	64
<i>Практична робота №5</i>	72
2.5 Кросбраузерність.....	74
<i>Практична робота №6</i>	80
РОЗДІЛ 3. Графіка та мультимедіа для вебсередовища	
3.1 Графіка для вебсередовища.....	82
<i>Практична робота №7</i>	87
3.2 Анімаційні ефекти.....	90
<i>Практична робота №8</i>	93
3.3 Мультимедіа на вебсторінках	97
<i>Практична робота №9</i>	101
РОЗДІЛ 4. Вебпрограмування	
4.1 Об'єктна модель документа.....	104
4.2 Вебпрограмування та інтерактивні сторінки	107
<i>Практична робота №10</i>	113
4.3 Хостинг сайта	115
<i>Практична робота №11</i>	121
4.4 Вебсервер та бази даних	125
4.5 Взаємодія «клієнт—сервер»	131
4.6 Валідація сайта і збереження даних форм	135
<i>Практична робота №12</i>	139
4.7 Прикладний програмний інтерфейс	142
РОЗДІЛ 5. Основи дизайну та просування вебсайта	
5.1 Правила ергономічного розміщення відомостей на вебсторінці.....	150
5.2 Пошукова оптимізація та просування вебсайтів.....	153
Література	158

**Навчальне видання
РЕЧИЧ Наталія Василівна**

**ІНФОРМАТИКА
ВЕБТЕХНОЛОГІЙ
вибірковий модуль
для учнів 10–11 класів
рівень стандарту**

Редактор Л. А. Каюда
Верстка І. І. Пікальової

Регіональні представництва
видавництва «Ранок»:

З питань придбання продукції
видавництва «Ранок» звертатися за тел.:
у Харкові – (057) 727-70-80;
Києві – (067) 449-89-65, (093) 177-05-04;
Вінниці – (067) 534-51-62;
Дніпрі – (056) 785-01-74, (067) 635-19-85;

«Книга поштою»: вул. Котельниківська, 5, Харків, 61051.

Тел. (057) 727-70-90, (067) 546-53-73.

E-mail: pochta@ranok.com.ua

www.ranok.com.ua

ТИ901874У. Підписано до друку 10.07.2020.

Формат 70×100/16. Папір офсетний.

Гарнітура Шкільна. Друк офсетний.

Ум. друк. арк. 13.

ТОВ Видавництво «Ранок»,

вул. Кібальчича, 27, к. 135, Харків, 61071.

Свідоцтво суб'єкта видавничої справи

ДК № 5215 від 22.09.2016.

Для листів: вул. Космічна, 21а, Харків, 61145.

E-mail: office@ranok.com.ua

Тел. (057) 719-48-65,

тел./факс (057) 719-58-67.

Київ – тел. (044) 229-84-01,

e-mail: office.kyiv@ranok.com.ua,

Львів – тел. (067) 269-00-61,

e-mail: office.lviv@ranok.com.ua

Житомирі – (067) 122-63-60;

Львові – (032) 244-14-36, (067) 340-36-60;

Миколаєві та Одесі – (067) 551-10-79;

Черкасах – (0472) 51-22-51;

Чернігові – (067) 440-88-93.

E-mail: commerce@ranok.com.ua



ВІДЕОУРОКИ

**Віртуальна школа
РАНОК**

Папір, на якому надрукована ця книга:



безпечний для здоров'я
та повністю
переробляється



з оптимальною білизною,
рекомендованою
офтальмологами



відбілювався
без хлору,
без діоксиду титану

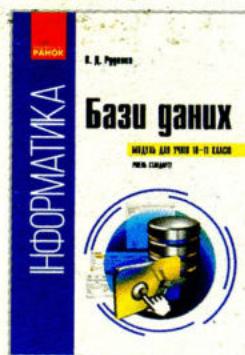
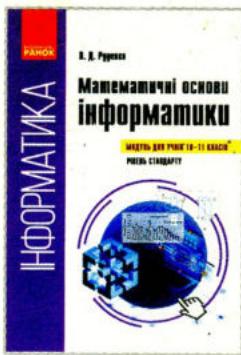
Разом обираємо про екологію та здоров'я

**ВИДАВНИЦТВО
РАНОК**

НАВЧАЛЬНИЙ ПОСІБНИК МІСТИТЬ:

- огляд основних трендів у вебдизайні
- опис базових тегів мови HTML та селекторів CSS
- алгоритми проєктування та створення сайту
- початкові відомості з вебпрограмування
- приклади застосування базових правил ергономіки сайту
- запитання для перевірки знань
- завдання для самостійного виконання
- практичні роботи

ПОСІБНИКИ НА ПІДТРИМКУ ВИБІРКОВИХ МОДУЛІВ:



навчально-методична література
УСІ КНИГИ ТУТ!

🛒 ranok.com.ua
⬇️ e-ranok.com.ua
✉️ pochta@ranok.com.ua
📞 (057) 727-70-90



Інтернет-підтримка
interactive.ranok.com.ua



ISBN 978-617-09-6223-2
9 786170 962232