О. Г. Трофименко, О. Б. Козін, О. В. Задерейко, О. Є. Плачінда

ВЕБ-ТЕХНОЛОГІЇ ТА ВЕБ-ДИЗАЙН

навчальний посібник

Одеса
Фенікс
2019

УДК 004.738.5 (076)
В 26

Рекомендовано Вченою Радою
Національного університету «Одеська юридична академія»,
протокол № 4 від 27 грудня 2018 р.

Рецензенти:
Казаков А. І. – доктор технічних наук, доцент, завідувач кафедри інформаційних технологій проектування в електроніці Одеського національного політехнічного університету;
Єгошина Г. А. – кандидат технічних наук, доцент кафедри інформаційних технологій Одеської національної академії зв'язку ім. О. С. Попова.

В 26 **Трофименко О. Г.**
Веб-технології та веб-дизайн : навч. посібник / О. Г. Трофименко, О. Б. Козін, О. В. Задерейко, О. Є. Плачінда. – Одеса : Фенікс, 2019. – 284 с.

ISBN 978-966-928-394-8

Навчальний посібник присвячено теоретичним і практичним аспектам веб-технології та веб-дизайну. Наведено опис основних засобів HTML та CSS для проектування, макетування й редагування веб-сайтів. Містить численну кількість прикладів працездатного HTML та CSS коду з демонстрацією структурування та форматування тексту й табличних даних, вбудовування зображень на веб-сторінці, розміщення медіаконтенту, створення форм, засобів позиціонування об'єктів тощо. Детально розглянуто специфіку застосування сучасних онлайн веб-конструкторів сайтів (uCoz, WordPress). Приділено увагу етапам розроблення сайту, засобам формування адаптивного веб-дизайну веб-сторінок, різним видам верстки веб-сторінок, вимогам до якості контенту при інформаційному наповненні сайту, сучасним тенденціям та стильовим рішенням у веб-дизайні. Сформульовані вимоги, підходи й специфіка аналізу та оптимізації роботи сайтів як важливого етапу у підтримці веб-сайту.

Призначений для студентів технічних спеціальностей і може бути корисним для широкого кола читачів, які бажають навчатися створювати, аналізувати та оптимізувати веб-сайти.

УДК 004.738.5 (076)

ISBN 978-966-928-394-8

Зміст

Передмова

Веб-технології стрімко розвиваються і на сьогодні створювані веб-сайти отримують все більше нових можливостей, стають більш зручними для користувачів. Саме тому уміння розробляти привабливі, функціональні, зручні веб-сайти наразі стають важливими складовими інформаційної культури фахівця у галузі інформаційних технологій, адже від того, як він зможе реалізувати у світовому інформаційному інтернет-просторі той чи інший проект, пов'язаний із професійною діяльністю, багато в чому залежить успішність її кар'єри.

Посібник може використовуватись як джерело прикладів та вправ для розглядання. Автори намагались показати специфіку сучасних засобів проектування та можливостей створення, аналізу та оптимізації власного веб-сайту.

Навчальний посібник складається із **семи розділів**.

На початку, у **розділі 1** докладно описані засоби мови HTML. Саме знання та використання засобів HTML 5 дозволяє прискорити завантаження сторінок сайту, додає нові можливості на сайт. Веб-програмування – це завжди пошук оптимальних рішень для виконання завдань, поставлених перед сайтами, які розробляються. Засоби HTML 5 сприяють знаходженню таких рішень, а тому професійні веб-програмісти активно застосовують можливості даних технологій при створенні сайтів.

Розділ 2 присвячений вивченню засобів стильового форматування вмісту веб-сторінок сайту – CSS. Каскадні таблиці стилів відповідають за зовнішнє подання (форматування) матеріалу на веб-сторінці. Саме CSS 3 дозволяє значно розширити можливості верстки сайтів без застосування сторонніх технологій. Привабливий дизайн сайтів із використанням мінімально необхідного коду – ось результат, який можна досягти за допомогою CSS 3.

HTML 5 і CSS 3 є певними фундаментальними засобами в розвитку технологій веб-програмування. Для їхнього засвоєння у посібнику наведено значну кількість прикладів працездатного HTML та CSS коду з демонстрацією структурування та форматування тексту і табличних даних, вбудовування зображень на веб-сторінці, розміщення медіаконтенту, створення форм, засобів позиціонування об'єктів тощо. Автори сподіваються, що численна кількість прикладів програмного коду спричинить у читача бажання не лише реалізувати їх на комп'ютері та перевірити їхню працездатність, а й змінити та удосконалити цей код.

У **розділі 3** наведено матеріал щодо застосування сучасних онлайн веб-конструкторів сайтів, за допомогою яких можна створити сайт без володіння спеціальних знань. Детально покроково розглянуто можливості загальнодоступного онлайн-конструктора сайтів з україномовним інтерфейсом uCoz та безкоштовного потужного конструктора сайтів WordPress, за допомогою якого можна створювати сайти «з нуля» й керувати їхнім вмістом без технічних навиків і знань. При цьому створення дизайну, заповнення та редагування контенту

відбувається безпосередньо в браузері комп'ютера за допомогою різноманітних шаблонів галереї і дизайнерських інструментів. Також у цьому розділі розтлумачено специфіку будови доменних імен та IP-адреси веб-сайтів.

Створюючи проект сайту, треба добре продумати його загальну структуру, зміст інформації та посилання. У **розділі 4** приділено увагу етапам розроблення сайту: проектуванню інтерфейсу майбутнього ресурсу і складанню технічного завдання, розробленню дизайну сайту, верстці сторінок сайту відповідно до сучасних вимог, тестуванню, перенесенню на хостинг (розміщенню сайту в інтернеті) та адмініструванню розробленого ресурсу. Окремо висвітлено важливі характеристики якості сайту: призначення сайту, ознаки якості сайту з погляду користувачів і з погляду професіоналів, стиль подачі матеріалу, критерії якості бізнес-сайтів тощо. Особливу увагу приділено вимогам до якості контенту при інформаційному наповненні сайту.

Розділ 5 присвячено способам та засобам розмітки веб-документів. Докладно на прикладах розглянуто різні види верстки веб-сторінок. Нині для оптимального відображення та взаємодії сайту з користувачем здебільшого на практиці вдаються до використання засобів адаптивного веб-дизайну веб-сторінок. У розділі на численних прикладах висвітлено переваги та специфіку застосування флексбоксів як відносно нового засобу розмітки для впорядкування елементів на сторінці з метою гнучкої адаптивності сторінки під різні розміри екранів для різних пристроїв. Адже саме модуль Flexible Box з жовтня 2017 року рекомендується корпорацією W3C як засіб оптимального компонування у дизайні інтерфейсу користувача.

У **розділі 6** йдеться про елементи та принципи веб-дизайну, які визначають правила взаємодії всіх елементів веб-сайту. Завдяки дотриманню цих загальноприйнятих принципів веб-дизайну можна створити вдалий успішний дизайн в результаті. Окремо зібрано та проаналізовано поширені помилки дизайнерів-початківців, на які варто звернути увагу для забезпечення зручної подачі інформації користувачеві та задоволення естетичного смаку аудиторії сайту. Значну увагу приділено сучасним тенденціям та стильовим рішенням у веб-дизайні. Використання у веб-дизайні асиметричних макетів і «ламаної» розмітки, контрастів, індивідуальних ілюстрацій, великих заголовків, які перекривають графічний контент, і поєднання різних стилів та напрямків веб-дизайну проілюстровано численними прикладами діючих сучасних сайтів. Показано, як при оформленні веб-сайтів великими зображеннями сучасні дизайнери додають ретро-патерни, кольори і типографіку. Продемонстровано використання 2D-оточення у тривимірному дизайні, статичних ілюстрацій з ненав'язливими ефектами, брутального веб-дизайну, продуманої анімації та ефектів UI, анімованої SVG-графіки тощо. Показано як візуальні контрасти, ізометрія, градієнти 2.0, Duotone та оверлей поверх зображень дозволяють змінювати веб-дизайн, адже нині, прагнучи до ясного розмежування, дизайнери тонко поєднують вишукані штрихи з об'ємними формами. У розділі детально розглянуто різноманітні стильові рішення сучасного веб-дизайну з прикладами їхнього застосування у композиції

сучасних сайтів. Також сформульовані вимоги та підходи до аналізу веб-дизайну.

Розділ 7 висвітлює складові та специфіку аналізу роботи сайту як важливого етапу у підтримці веб-сайту. SEO-аналіз, юзабіліті-аналіз, аналіз відвідуваності та технічний аналіз веб-сайту потрібно проводити на регулярній основі. Саме регулярний аналіз дозволяє вчасно помітити проблеми або перебої в роботі сайту і внести потрібні правки. У розділі розглянуто алгоритм виконання SEO-аудиту для визначення готовності сайту до виконання активних дій з його розкручування. Охарактеризовано різні методи SEO-оптимізації сайту як комплексу заходів, направлених на підняття позицій сайту в результатах видачі пошукових систем за певними запитами (ключовими фразами). Наведено приклади безкоштовних онлайн-засобів перевірки сайтів для виявлення «слабких місць», які до того ж формують пропозиції щодо усунення виявлених проблем, забезпечення високого рівня юзабіліті сайту та його ефективності. Проаналізовано сучасні доступні інструменти для веб-аналітики. Охарактеризовано півсотні різних сучасних плагінів, які інтегруються в різні браузери чи онлайн-конструктори і покликані спростити процес розроблення і налагодження веб-сторінок.

Щоб забезпечити ефективніше сприйняття матеріалу та формування практичних навиків, наприкінці кожного розділу розміщені запитання для самоконтролю засвоєних знань.

Автори сподіваються, що запропонований до Вашої уваги посібник стане корисним при вивченні веб-технологій, і будуть щиро вдячні за доброзичливу та конструктивну критику й відгуки.

E-mail: egt@ukr.net.

Розділ 1
Засоби HTML для розроблення сайтів

1.1. Предмет і задачі дисципліни

Майбутні фахівці, мають не лише користуватися інтернетом, а й використовувати його як знаряддя праці у своїй роботі.

ТОП затребувані професії в інтернеті:

– веб-програміст (Web developer);
– верстальник веб-сторінок (HTML-верстальник, HTML coder);
– веб-дизайнер (Web designer);
– розробник мобільних застосунків;
– відео-монтажер (Video editor);
– адміністратор соціальних мереж;
– спеціаліст із контекстної реклами;
– блоґер (Blogger).

Охарактеризуємо деякі з них.

HTML-верстальник – це фахівець, який виконує верстку веб-сторінок. Саме він створює HTML-шаблон для веб-сайту з використанням знань HTML-коду і всіх особливостей стилю і графічного оформлення. Функціональні обов'язки HTML-верстальника полягають у реалізації концепції та ідеї сайту, розроблених веб-дизайнером, у вигляді HTML-коду. Рівень середньої зарплатні у цій професії за даними 2018 року в Одеській області – 17 000 грн (≈ $630).

Нині існує численна кількість комп'ютерних програм, які автоматизують працю верстальника: різні текстові редактори з підсвічуванням коду та візуальні редактори (Notepad++, Coffecup, Brackets, Notetab чи BlueGriffon), front-end фреймворки (набори фрагментів коду і бібліотек класів для прискореної розроблення макета сайту шляхом прототипування – ZurbFoundation-4 тощо). Фреймворки дозволяють писати великі фрагменти коду в наочному режимі, тобто результат кожного етапу роботи можна спостерігати в окремому вікні. Але професійний HTML-верстальник цими програмами не користується. Він повинен вміти використовувати кодування HTML вручну, без допомоги візуальних редакторів, щоб забезпечити максимальну коректність коду в мінімальній вазі.

HTML-верстальник має знати каскадні стильові таблиці CSS, володіти JavaScript і базовими навиками веб-програмування мовами PHP, Perl або Java, а також основними графічними редакторами Photoshop, Fireworks, Gimp. Досвідчений верстальник може створити невеликий сайт, використовуючи будь-який текстовий редактор з мінімальною кількістю засобів й інструментів.

Успішна робота HTML-верстальника будується на трьох «китах»: якісний продукт, принцип юзабіліті, адаптивний дизайн. *Якісний код* має бути добре

структурованим і мати правильне семантичне оформлення відповідно до правил SEO-оптимізації. *Принцип юзабіліті* – це простота в розробленні інтерфейсу. Навігація по сайту не повинна змушувати відвідувачів напружено думати. *Простота інтерфейсу* – запорука успіху проекту. Адаптивний дизайн зробить сайт доброзичливим до мобільних пристроїв.

При роботі над великими проектами робота HTML-верстальника зводиться до створення тільки макета сайту. Різні модулі та елементи в нього встановлюють програмісти вузької спеціалізації. Але на невеликих проектах HTML-верстальнику доводиться виконувати роботу з кодом від початку до кінця.

Плюси і мінуси професії:

– плюси:
  - ✓ можливість самостійного освоєння професії;
  - ✓ високий попит на ринку праці;
  - ✓ потреба постійного вдосконалення й оновлення знань;
  - ✓ можливість працювати віддалено;
  - ✓ нечіткі межі між роботою веб-дизайнера, веб-програміста і банермейкера дають можливість працювати в суміжних галузях;
– мінуси:
  - ✓ присутня частка рутинності й одноманітності;
  - ✓ довготривале сидіння за комп'ютером.

Місцями роботи можуть бути: інтернет-компанії, фірми з розроблення сайтів, дизайн-студії, організації зі своїми інтернет-проектами, фрілансерська робота.

Важливі якості HTML-верстальника: уважність, зосередженість, здатність концентруватися, терпіння у виявленні власних помилок, посидючість, акуратність, прагнення працювати на кінцевий результат.

Здебільшого HTML-верстальники освоюють професію самостійно, але є і спеціалізовані курси. Основна вимога роботодавців: досвід роботи, підкріплений портфоліо. Треба знати: HTML (вручну), CSS, JavaScript, Photoshop, PHP, MySQL, XML з XSL.

Веб-дизайнер – це свого роду художник з технічним ухилом або ж програміст із почуттям естетики. По суті, веб-дизайнери створюють сам зовнішній вигляд майбутнього сайту, визначають, де саме будуть перебувати різні елементи, оформляють їх і вирішують, як взагалі буде відбуватися взаємодія користувача із сайтом. Хороші веб-дизайнери цілком здатні заробляти понад $900 – $1500 на місяць.

Суміжними спеціальностями веб-дизайнера є: інформаційний дизайнер (займається «наведенням порядку» на сайті з метою більш зручного сприйняття інформації), дизайнер інтерфейсів (забезпечення зручності навігації по сайту), ілюстратор, графічний дизайнер (створення логотипів, фірмового стилю) і дизайнер поліграфії (буклети, календарі, візитки тощо).

Як технар такий фахівець утримує у себе в голові безліч технічних вимог до дизайну сайту, а як творець він шукає нестандартні рішення для складних проблем.

Блоґер займається тим, що створює різні тексти, фото і відеоматеріали для свого авторського проекту – сайту в мережі. При досягненні певної популярності такий блоґ вже можна монетизувати за допомогою тієї ж контекстної реклами, партнерських програм, різних платних оглядів тощо. Діапазон заробітків широкий – від $100 до $1000 на місяць. Оскільки блоґери часто самі займаються просуванням своїх сайтів, то до суміжних онлайн-професій можна віднести фахівця з просування сайтів і веб-верстальника. Поширено вважають, що популярна і затребувана сьогодні інтернет-професія блоґера є однією з найпростіших, проте це помилкова думка. Так, у мережі можна найти сотні найрізноманітніших авторських блоґів, але лише одиниці з них «виходять в топ», домагаються популярності і визнання своєї аудиторії.

Для створення інтерфейсу сайтів використовуються HTML, CSS і JavaScript:

– HTML – відповідає за структуру і вміст веб-сторінки;
– CSS – за зовнішній вигляд сторінок сайту;
– JavaScript – використовується для визначення поведінки сторінок.

1.2. Історія появи мови HTML

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

1989 року в Женеві Бернерс-Лі[1] запропонував впровадити на базі Internet гіпертекстову[2] систему документів, яку назвав World Wide Web (WWW) або Всесвітня павутина.

Наприкінці 1990 року він розробив мову розмітки HTML і написав браузер та серверне програмне забезпечення для запропонованої системи.

Наприкінці 1991 року Тімоті Джон Бернерс-Лі опублікував в інтернеті перший загальнодоступний опис мови HTML, відомий як документ «HTML-теґи» (HTML Tags). У ньому були описані 20 елементів первісної, відносно простої схеми розмітки HTML. 13 із цих елементів і досі існують у HTML4.

1993 року Спеціальна комісія інтернет-розробок (IETF) офіційно опублікувала першу специфікацію HTML: «HyperText Markup Language (HTML)».

Рік	1991	1993	1995	1997	1999	2000	2014
Версія	HTML	HTML+	HTML 2.0	HTML 3.2	HTML 4.01	XHTML	HTML 5

[1] Тімоті Джон Бернерс-Лі 1994 року заснував і нині очолює консорціум W3C (англ. World Wide Web Consortium), який розробляє і впроваджує технологічні стандарти для Всесвітньої павутини.

[2] Термін гіпертекст був введений Т. Нельсоном у 1965 р. для опису документів, які подаються комп'ютером і виражають нелінійну структуру ідей, на противагу лінійній структурі традиційних книг, фільмів і мови. Завдяки нелінійній структурі і зв'язкам, можна читати матеріал у будь-якому порядку. Найпростіший приклад гіпертексту «доінтернетовської епохи» – це будь-який словник чи енциклопедія, де кожна стаття має посилання до інших статей цього ж словника. У результаті читати такий текст можна по-різному: від однієї статті до іншої, у міру потреби, ігноруючи гіпертекстові посилання; читати статті одну за одною, справляючись із відсиланнями; переходити від одного відсилання до іншого тощо.

Новації версій HTML:

- HTML (без номера версії, 1992) – найперша версія, орієнтована лише на текст;
- HTML (без номера, 30 квітня 1993 року): до тексту додано атрибути, які визначають курсивне або жирне накреслення літер, та зображення;
- HTML+ (листопад 1993 року): заплановані доповнення, які потрапили до наступних версій, але не були відокремлені як HTML+;
- HTML 2.0 (1995) визначено стандартом з підтримкою форм (офіційної специфікації HTML 1.0 не існує);
- HTML 3.0 (1995);
- HTML 3.2 (14 січня 1997 року): додані численні можливості: таблиці, обтікання текстом зображень, інтеграція аплетів;
- HTML 4.0 (18 грудня 1997 року): додані таблиці стилів, скрипти і фрейми. 24 квітня 1998 року випущено виправлену версію цього стандарту. Відбулася певна «чистка» стандарту. Деякі елементи були позначені як застарілі і не рекомендовані (англ. deprecated). Так, тег <font> був позначений як застарілий (замість нього рекомендується використовувати таблиці стилів CSS);
- HTML 4.01 (24 грудня 1999 року) містить численні дрібні виправлення версії HTML 4.0;
- HTML 5 (5 квітня 2008): запроваджено новий словник, побудований на основі HTML 4.01 та XHTML 1.0. Також перероблено і розширено пов'язану з HTML специфікацію DOM;
- HTML 5.1 почав розроблятися 17 грудня 2012 року;
- HTML 5.2 (14 грудня 2017 року).

1.3. Засоби створення веб-сторінок

Веб-сайт – це сукупність багатьох веб-сторінок, доступних через інтернет, об'єднаних загальною темою і пов'язаних між собою гіпертекстовими та іншими навігаційними посиланнями. Простіше – це певне місце в інтернеті, де можна розмістити будь-яку інформацію, зробивши її доступною з будь-якої точки світу.

Веб-сторінки зберігаються як окремі файли з розширенням htm або html. Одна веб-сторінка займає один або декілька файлів.

Для зберігання файлів, що утворюють веб-сайт, на диску серверного комп'ютера виділяється спеціальна папка, яка називається кореневою папкою веб-сайту. Усі файли, складові веб-сайту, повинні зберігатися в кореневій папці або в папках, вкладених в неї.

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

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

HTML-документ переважно є простим текстовим файлом, який містить тільки текст. Тому для створення веб-сторінки можна використовувати будь-який текстовий редактор, наприклад, Блокнот (Notepad) або спеціалізовані HTML-редактори.

Усі HTML-редактори можна умовно класифікувати за такими категоріями:

– WYSIWYG HTML-редактори коду та редактори HTML-тегів;
– спеціалізовані програми та онлайн-сервіси.

WYSIWYG HTML-редактори коду, за допомогою яких можна створювати веб-сторінки без знання мов програмування, що випливає з назви: What You See Is What You Get (що бачиш, те й отримуєш)[3]. Перевагою таких програм є відсутність потреби заглиблення в процес побудови сторінки, однак це саме є і їхнім недоліком. Редактори цього типу часто формують об'ємні HTML-коди, в результаті чого документ виходить неймовірно громіздким, а час його завантаження завеликим;

Редактори HTML-тегів дозволяють формувати і змінювати HTML-код його веб-сторінок, в результаті HTML-документ чого буде набагато компактнішим, порівняно з результатами роботи редакторів першого типу. Однак для комфортної роботи й отримання прийнятного результату треба знати мови веб-програмування на достатньо високому рівні.

Професіонали використовують здебільшого (проте не завжди) **спеціалізовані редактори-HTML**, наприклад:

– ***Notepad++*** – мабуть найкращий безкоштовний HTML-редактор для Windows, не придатний для Mac чи Linux (для користувачів Mac альтернативою є Brackets). Скачати можна на офіційному сайті https://notepad-plus-plus.org. Величезні можливості Notepad++ забезпечуються сторонніми плагінами, наприклад: PreviewHTML, HTML tag plugin для підсвічування тегів, Tidy2 для відступів тощо. Ключові особливості Notepad++: табличний інтерфейс для одночасної роботи з декількома файлами; легке згортання і підсвічування синтаксису; вертикальні таблиці; багатомовність (вибір мови інтерфейсу: англійська, українська, французька, іспанська, китайська чи то 80 інших мов) тощо;
– ***Google Web Designer*** (скачати можна за посиланням https://www.google.com/webdesigner/index.html);
– ***Microsoft Office SharePoint Designer*** замінила застарілу Microsoft Office FrontPage. У склад Microsoft Office не входить. З 2009 року Microsoft SharePoint Designer розповсюджується безкоштовно на офіційному сайті Microsoft Office, скачати можна за посиланням https://www.microsoft.com/ru-RU/download/details.aspx?id=35491. Є російськомовна версія;

[3] Выбираем HTML редактор – путеводитель для новичков. URL: http://www.internet-technologies.ru/articles/vybiraem-html-redaktor-putevoditel-dlya-novichkov.html (дата звернення 12.07.2018).

- ***Brackets*** – це популярний, безкоштовний, надійний і легкий у використанні інструмент для веб-розробки і редагування коду HTML в Mac, Windows і Linux. Він має розширення, які поліпшують його функціональність: Emmet прискорює написання коду CSS і HTML; Beutify форматує файли HTML, CSS і JavaScript; W3C validation перевіряє HTML-код на валідність. Зроблені в коді CSS або HTML змінення відображаються у браузері в режимі реального часу;
- ***Coffecup*** – якісний HTML-редактор, який має як безкоштовну, так і комерційну версію. У безкоштовної немає деяких функцій, однак вона має такі функції: налаштовувані панелі інструментів, завершення коду для елементів, атрибутів і селекторів, готові до використання теми і шаблони, підтримка drag and place для зображень, опція попереднього перегляду, підсвічування синтаксису, тезаурус для пошуку альтернативи для слів і багато іншого. Платна версія не надто дорога. Вона має кілька додаткових функцій, наприклад: валідацію коду HTML і CSS, бібліотеку тегів, чистильник коду, динамічну перевірку правопису. Coffecup на ринку з 1996 року і використовується фрілансерами, у стартапах, дрібному бізнесі, а також веб-розробниками з великих компаній. Це прекрасний інструмент для створення сайтів, веб-сторінок, розсилок, заміток, відформатованих в HTML, контенту для соціальних медіа. Завантажити безкоштовну версію Coffecup можна тут – https://www.coffeecup.com/free-editor;
- ***Notetab –*** редактор для HTML і CSS від компанії Fookes software, яка останні 20 років займається інструментами для прискорення процесу розроблення. Популярність NoteTab підтверджується тим, що його використовують в NASA, VISA, CIA Hewlett Packard, MIT (Massachusetts Institute of Technology). NoteTab має три версії: Light, Standard і Pro. Light-версія розповсюджується безкоштовно для індивідуального використання. Вона не має всіх функцій Pro-версії, однак підтримує бібліотеки HTML5 та CSS3, Bootstrap (вільний набір інструментів для створення сайтів і веб-застосунків), автозаповнення HTML, об'єднання файлів у проекти, підтримку HTML Tidy, HTML to text, а також має багато тем оформлення. NoteTab може запускатися прямо з флешки і не вимагає установки на вашу машину. Для блоґерів і людей, що займаються наповненням сайтів, цей редактор підтримує підрахунок слів і SEO-статистику. Познайомитися з NoteTab ближче можна на офіційному сайті https://www.notetab.com;
- ***Eclipse*** є популярним інтегрованим середовищем розроблення сайтів із відкритим вихідним кодом. Це безкоштовний «важкоатлет» серед HTML-редакторів, який можна з успіхом використовувати, якщо встановити (http://www.eclipse.org) і налаштувати (може зайняти деякий час) цього «звіра». Дозволяє створювати складні сайти з базами даних, об'єднаними з іншими джерелами даних тощо. Потрібно буде встано-

влення додаткових плагінів. Цей редактор підтримує Java, JavaScript, PHP, Ruby, Android і багатьох інших мов програмування;

– ***BlueGriffon*** **–** це потужний редактор, який можна завантажити на сайті http://www.bluegriffon.org і встановити на Windows, Linux Ubuntu і OS X на Mac. BlueGriffon успадкував більшу частину своїх можливостей від Netscape, Composer, Nvu і Mozilla. BlueGriffon побудований на Gecko, движку Mozilla для виведення веб-сторінок. BlueGriffon має три варіації, перша з яких безкоштовна, друга поширюється по базовій ліцензії, а найпотужніша – за ліцензією EPUB. Безкоштовна версія теж багато чого може. У неї є чорна і світла теми оформлення, підтримка аудіо, відео та форм з HTML5, функції редагування CSS3, 3D і 2D трансформації, створення SVG, технологію WYSIWYG, google fonts менеджер, менеджер шрифтів font squirrel, підтримку формату маркдаун (полегшена мова розмітки), інтерфейс на 20 мовах, зокрема російській;
– ***Microsoft Expression Web*** та інші.

Попередньо такі редактори треба інсталювати.

Онлайн HTML-редактори для створення веб-сторінок мають суттєві переваги, оскільки не потребують скачування, встановлення, до того ж вони безкоштовні:

– ***Htmledit*** (http://htmledit.squarefree.com) – простий онлайн HTML-редактор, який можна використовувати для швидкого чорнового кодування;
– ***Xhtml.ru*** (http://xhtml.ru/instr/html_editor) – єдиною функцією цього та двох попередніх редакторів є написання HTML-коду. Перевагами xhtml.ru є наявність у сервісі http://xhtml.ru/online-tools: таблиці кольорів та їхніх числових шістнадцяткових значень (http://xhtml.ru/online-tools/webcolors), генератора кольорових схем підбору шрифтів тощо;
– ***Htmlinstant*** **(**http://www.htmlinstant.com) – онлайн веб-редактор, який, крім звичайного рукописного введення коду, пропонує скористатися невеликою панеллю інструментів, містить все найнеобхідніше, наприклад: маркірований і нумерований списки, вставлення зображення, посилання, виділення тексту та інше;
– ***HTML-online.com*** (https://html-online.com/editor) – надійний візуальний редактор, який дозволяє створювати веб-сторінки і водночас відстежувати за змінами коду в сусідньому вікні. Є функції чищення HTML-коду, приведення табличних елементів до div, опції пошуку і заміни. Перевагою цього редактора є можливість конвертації Word в HTML, завдяки чому можна копіювати контент із файлів Microsoft Word і автоматично застосовувати до нього HTML-розмітку. Також є підтримка Google docs, PDF, Excel, PowerPoint і багатьох інших видів документів;

- ***HTML editor*** (https://htmleditor.tools/html-editor) складається з двох полів, де можна водночас переглядати й отримувати доступ до візуального та вихідного кодів. Змінювати код можна як в WYSIWYG-редакторі ліворуч, так і синтаксично у вікні праворуч.
- ***JSFiddle*** (http://jsfiddle.net) – одне з найпопулярніших середовищ веб-розробки (свого роду пісочниця для веб-програмістів), що дозволяє редагувати і запускати код («фіддл»), написаний на HTML, JavaScript і CSS. Робоче поле jsFiddle поділене на чотири частини, розміри яких можна змінювати. Окремо в одній частині пишуть HTML-код, окремо в другій CSS-код, окремо JavaSript. Результат відображається в четвертій частині (робочій області). Є підсвічування синтаксису. Можна вибрати світлу або темну тему. Є можливість використовувати бібліотеки для JavaScript, наприклад, jQuery. Кнопка *Tidy* на верхній панелі дозволяє вирівняти усі рядки коду. Популярним використанням jsFiddle є вставлення фрагмента коду у блоґи, можливість ділитися кодом через соціальні мережі і спільна робота над кодом. JsFiddle фактично є саме JavaScript пісочницею, яка, на відміну від WYSIWYG, доволі функціональна.
- ***HTML-CSS-JS.com*** (http://html-css-js.com/css або http://html-css-js.com/html) дозволяє збільшити ефективність кодування та забезпечує гарну користуцьку роботу за допомогою безкоштовної онлайн-колекції інструментів.

Отже, нині мова розмітки HTML використовується повсюдно. Є багато HTML-редакторів, які створені давно, але при цьому йдуть в ногу з часом і цілком придатні для веб-розробки. Проте є й інші, яким вже не вистачає функціоналу і продуктивності. З'являються нові редактори, такі як Brackets та Atom, які забезпечують ефективне редагування коду.

Більшість редакторів мають свої особливості, проте слугують одній меті – створенню веб-сторінок. Вони збільшують швидкість розроблення і допомагають впорядковувати код, при цьому враховують можливість його масштабування.

Незалежно від потреби створювати лише блоґи і статті, добре відформатовані для читання їх браузерами, або прагнення побудувати повністю функціональний веб-сайт, використовуючи HTML і CSS, вибір правильного редактора значно підвищить продуктивність.[4] Професійні розробники веб-сторінок витрачають значну кількість часу на вибір редактора й інструментів, які найкращим чином відповідатимуть їхнім потребам і сприятимуть зростанню продуктивності.

Доцільно записати свої ключові потреби, придивитися до різних наявних варіантів редакторів, скоротити список до пари редакторів, спробувати кожен і визначити, який сподобається більше.

[4] 10 лучших бесплатных HTML-редакторов. URL: https://techrocks.ru/2017/09/28/10-best-free-html-editors/ (дата звернення 12.07.2018).

1.4. Елементи HTML і структура HTML-документа

1.4.1. Використовувана термінологія

Коли веб-сторінка відкривається в браузері, він переглядає її HTML-код, знаходить спеціальні символи, теґи та використовує їх для змінення вигляду тексту, вставлення зображень, створення посилань на інші веб-сторінки тощо.

Дескриптор (**теґ**) – основний елемент кодування, прийнятий у стандарті HTML. Теґи визначають межі дії елементів і відокремлюють елементи один від одного. Позначення теґів записується у кутових дужках: <теґ>.

Існує два типи HTML-теґів:

1) **парні теґи** (контейнери) – дескрипторні пари, які складаються з початкових (відкривних, стартових) і завершальних (кінцевих, закривних) дескрипторів. Завершальний теґ відрізняється від стартового прямим слешем перед текстом всередині кутових дужок. Наприклад (рис. 1.1): <html> і </html>, <head> і </head>, <title> і </title>, <body> і </body>, і , і тощо. Контейнери призначені для зберігання деякої інформації, приміром, тексту або інших HTML-дескрипторів. Вміст HTML-елемента розміщується поміж стартовим і завершальним теґами (рис. 1.1).

2) **одинарні теґи** – складаються тільки з початкових дескрипторів й не має вмісту. Він виконує самостійне завдання, не пов'язане з конкретним текстом, наприклад, теґ розриву рядка
 (рис. 1.1), теґ горизонтальної лінії <hr /> тощо.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title> Назва документа</title>
  </head>
  <body>
    <p><b>Напівжирний</b> <br/>текст.</p>
  </body>
</html>
```

Рисунок 1.1 – Використання парних та одинарного теґів

Теґи нечутливі до регістру, тому записи <P> і еквівалентні.

Три обов'язкові теґи:

<html> – контейнер для всіх HTML-елементів сторінки, зокрема з теґами <head> і <body>. Як правило, теґ <html> йде у документі відразу після доктайпу. Теґ <html> повідомляє браузеру, що цей файл є HTML-документом. Теґ <html> ще називають кореневим теґом для HTML-документа;

<head> призначений для зберігання елементів, мета яких – допомогти браузеру у роботі з даними. Вміст цього теґу не відображається напряму;

<body> призначений для зберігання вмісту веб-сторінки (контенту) у вікні браузера.

В останні версії HTML (після 4.01) в основну структуру HTML-документа було додано опис **<!DOCTYPE>**, перед теґом <html>. Доктайп не є теґом і потрібен браузерам для правильної інтерпретації подання веб-сторінки, адже HTML існує в декількох версіях, крім того є XHTML[5]. Доктайп вказує на тип поточного документа, для якого була визначена HTML-сторінка, щоб браузер «не плутався» і розумів, згідно з яким стандартом відображати веб-сторінку. Існує декілька видів <!DOCTYPE>, вони розрізняються залежно від версії HTML, на яку орієнтовані.

1.4.2. Загальноприйнята структура HTML-документа

У 5-й версії HTML **базова структура HTML-документа** буде такою:

```
<!DOCTYPE HTML>
<html>
 <head>
   <!-- Цей розділ призначений для заголовка сторінки title та технічної
   інформації. До речі, це коментар.
   Зверніть увагу на синтаксис коментаря.-->
   <meta charset="utf-8">
   <title> Назва документа</title>
 </head>
 <body>
    <!--А тут треба розміщувати все, що бажано побачити на сторінці. -->
 </body>
</html>
```

Як тут показано, певний текст (здебільшого пояснювальний) можна приховати від показу в браузері, зробивши його коментарем. Коментарі починаються теґом <!-- і закінчуються теґом -->:

```
<!-- Коментарі-->
```

Отже, відповідно до структури веб-сторінки її розроблення полягає у наповненні двох основних розділів: заголовка (<head>) і тіла документа (<body>).

Розділ заголовка <head> може містити текст і теґи, але вміст цього розділу не показується безпосередньо на сторінці.

Тіло документа <body> призначене для розміщення змістовної частини.

Ієрархічна структура HTML-документа нагадує велике сімейне древо, з батьками, братами, дітьми, предками і нащадками (рис. 1.2).

Дерево документа HTML (document tree) – це схема побудови HTML-документа, яка показує зв'язки між різними елементами сторінки: порядок слідування і вкладеність елементів. Ця схема допомагає орієнтуватися у цій, на перший погляд, хаотичному накопиченню HTML-теґів.

[5] XHTML (EXtensible HyperText Markup Language – розширена мова розмітки гіпертексту) схожий на HTML, але відрізняється синтаксисом.

Наведемо приклад доволі простого HTML-коду:

```
<html>
  <head>
        <title>Заголовок сторінки</title>
        <meta charset="utf-8">
  </head>
  <body>
        <div class="mainWrap">
              <h1>Основний заголовок</h1>
              <p>Абзац тексту</p>
              <ul>
                    <li>пункт 1</li>
                    <li>пункт 2</li>
              </ul>
        </div>
        <div class="sideBar">
              <h2>Другий заголовок</h2>
              <p>Текст</p>
        </div>
  </body>
</html>
```

Цей HTML-код досвідчене око веб-розробника розбирає по поличках, відповідно до ієрархії теґів, враховуючи всі рівні вкладеності і взаємозв'язку, і сприймає як чітку ієрархічну структуру у вигляді дерева (рис. 1.2).

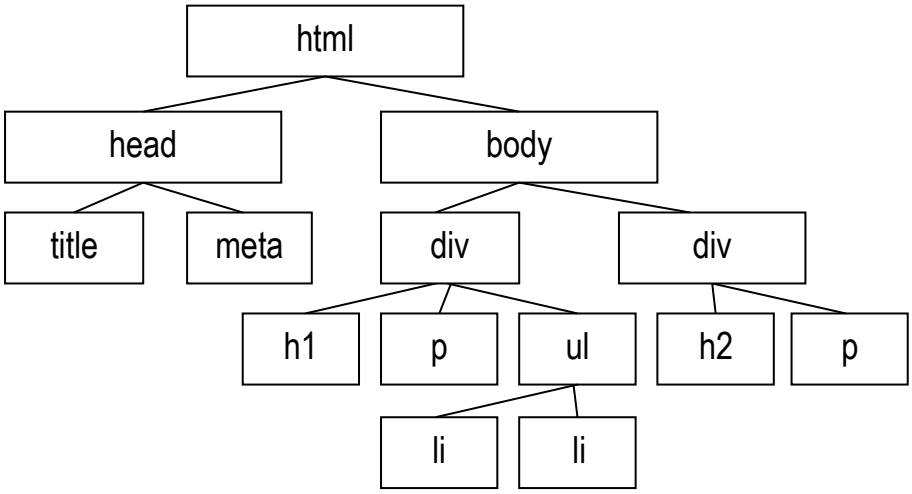


Рисунок 1.2 – Дерево документа HTML

Отже, текст на веб-сторінці може складатися з різних за змістом та формою елементів: заголовків (теґи від <h1> до <h6>), абзаців (теґ <p>), списків (ul, ol, li тощо), таблиць (теґи <table>, <tr>, <td> тощо), вбудованих зображень (теґ <img>), гіперпосилань (теґ <a>) тощо.

Веб-розробнику дерево документа допомагає при написанні CSS-правил.

1.4.3. Терміни успадкування

Для механізму успадкува́ння (наслідування) використовують такі **терміни**[6]:

- **предок** (ancestor) – базовий елемент, який містить в собі інші елементи. Наприклад: <html> – предок для всіх, а <head> – лише для <title> (рис. 1.2);
- **потомок** (descendant) – породжений елемент, який розміщено всередині якогось елемента. Наприклад: <title> – потомок <head>, а <head> – <html>;
- **батько** (parent) – елемент, який стоїть вище на один рівень відносно того елемента, який розглядається. Наприклад: <html> – батько для <head> і <body>, <body> – для , а – для <h1>, <p>, <ul> тощо (рис. 1.2);
- **дитина** (child) – елемент, який стоїть на один рівень нижче елемента, який зараз розглядається. Наприклад: <h1>, <p>, <ul> є дочірніми тільки відносно , але не до <html>, а <body> – дочірній до <html> (рис. 1.2);
- **братський елемент** (siblings) – елемент, розміщений на одному рівні поруч з елементом, який розглядається. Наприклад: <h1>, <p>, <ul> – братські, також братськими є <head> і <body>, а ось <head> і <title> мають різних батьків і не є братськими (рис. 1.2).

Позаяк одночасно можна використовувати будь-яке розумне поєднання теґів, треба пам'ятати про їхню **вкладеність**:

```
<!DOCTYPE HTML >
<html>
 <head>
  <meta charset="utf-8">
  <title>Вкладення теґів</title>
 </head>
 <body>
  <p><b><i>Напівжирний курсивний текст</i></b></p>
</body>
</html>
```

У цій структурі <html> є предком для будь-якого іншого елемента і є батьком для <head> і <body>. <head> і <body> є братніми один для одного. В свою чергу, <body> є батьком для <p>, який в ньому міститься. З іншого боку всі теґи <head>, <meta>, <title>, <body>, <p>, <b>, <i> – це все нащадки <html>. У цьому прикладі текст записано всередині контейнера <i>, який встановлює курсивне накреслення. Він, у свою чергу, розміщується всередині контейнера <b>, що задає жирне накреслення тексту.

Аналогія із сімейним деревом також простежується при проходженні декількох шарів вкладень в HTML:

- нащадок елемента X – це будь-який елемент всередині X;
- дитина – це просто прямий нащадок;

[6] Conformance: Requirements and Recommendations. URL: https://www.w3.org/TR/CSS21/conform.html (дата звернення 3.11.2018).

− предком елемента є будь-який елемент назовні його;
− батько – це лише перший прямий предок.

Один контейнер має розташовуватись всередині іншого і не перетинатися. Помилковою буде така комбінація теґів:

```
<p><i><b>Напівжирний курсивний текст</i></b></p>
```

У цьому прикладі порушена вкладеність теґів один в одного. Хоча браузер і відобразить приклад коректно, самостійно «здогадавшись», чого від нього хочуть, подібних помилок треба уникати, оскільки вони призводять до уповільнення роботи сторінки і до неправильної демонстрації сторінки в більшості випадків.[7]

1.4.4. Атрибути теґів

Деякі теґи можуть мати **атрибути**, які надають додаткову інформацію HTML-елементам. Атрибути записують у відкривному (стартовому) теґу елемента з дотриманням правил синтаксису: назва атрибута="значення".

Наприклад:

```
<img src="cat.jpg" alt="Кіт" width="100" height="140">
```

Стандарти рекомендують використовувати лапки для визначення значень атрибута. Відсутність лапок може призвести до неправильної інтерпретації коду.

Глобальні атрибути – це атрибути, які можна використовувати в усіх HTML-елементах.

Атрибут	Опис
class	Визначає одну чи декілька назв класів для HTML-елемента (з'явився як наслідок потреби визначення стилів)
contenteditable	Визначає, чи може вміст HTML-елемента бути змінюваними, чи ні
draggable	Визначає, чи може вміст елемента бути пересунутим
dropzone	Визначає, як і куди можна пересунути чи скопіювати HTML-елемент
hidden	Визначає, можна чи ні показувати вміст HTML-елемента
id	Визначає ідентифікатор для HTML-елемента (найчастіше використовується JS)
style	Визначає inline стилі
title	Визначає додаткову інформацію для HTML-елемента

Існують інші глобальні атрибути[8], які визначають запуск подій і ставлять у відповідність скрипт.

[7] Огурцов В.В., Гриньов Д.В., Щербаков О.В. Основи веб та веб-дизайн, програмування на боці клієнта: лабораторний практикум з навчальної дисципліни "Веб-технології та веб-дизайн" для студентів напряму підготовки 6.050101 "Комп'ютерні науки". Харків: ХНЕУ ім. С. Кузнеця, 2015. 208 с.

[8] HTML Event Attributes. URL: http://www.w3schools.com/tags/ref_eventattributes.asp (дата звернення 12.07.2018).

1.4.5. Елемент CER для подання спеціальних символів

Крім основного елемента HTML – теґу, є елемент HTML **CER** (Character Entity Reference, посилання на наявні символи), призначені для подання спеціальних символів у документі HTML, які можуть бути некоректно розпізнані браузером.

Символ	Іменна заміна	Числовий код	Опис
			нерозривний пробіл
/	&frasl;	&#8260;	коса риска
‘	&apos;	&#39;	одинарна лапка
"	&quot;	&#34;	подвійна лапка
«	&laquo;	&#171;	ліва кутова лапка
»	&raquo;	&#187;	права кутова лапка
&	&amp;	&#38;	амперсанд
<	&lt;	&#60;	менше
>	&gt;	&#62;	більше
±	&plusmn;	&#177;	плюс-мінус
€	&euro;	&#8364;	євро
–	&ndash;	&#8211;	тире
—	&mdash;	&#8212;	довге тире
®	&reg;	&#174;	правова охорона торгової марки
©	&copy;	&#169;	охорона авторського права
№		&#8470;	номер
́		&#769;	Знак наголосу (ставиться безпосередньо після літери, над якою має з’явитися)

Так, браузери не коректно розпізнають символи "<" (менше) та ">" (більше), оскільки вони використовуються у теґах. За потреби виведення таких символів у вікні веб-сторінки слід скористатись CER – < та >. Тобто, якщо вказати у тексті HTML рядок <BODY>, то на екрані у тексті буде виведено: <BODY>.

Крім того, CER використовують для виведення символів одинарних і подвійних лапок, нерозривного пробілу, тире та довге тире, "&" (амперсанд), "©" (копірайт), "®" (правова охорона торгової марки) тощо.

На відміну від найменувань теґів HTML, найменування CER чутливі до регістру символів. Також найменування CER можуть задаватися не тільки поіменно, а й за допомогою тризначкових кодів символів у вигляді &#nnn;. Далі в таблиці наведено найбільш часто використовувані CER і відповідні їм числові коди.

1.5. Елементи із заголовка документа <head>

Теґи та тексти, що містяться у цьому розділі, не відображуються на веб-сторінці. Цей розділ призначений для службової інформації. Усі веб-браузери підтримують цей елемент.

Теґ <head> є контейнером для елементів заголовка документа. До 5-ї версії було обов'язковим його використання, а в HTML5 цей теґ може бути відсутнім.

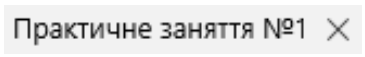
Базовий елемент <head> може містити такі теґи:

1) заголовок поточної сторінки (<title>);
2) базова адреса (<base>);
3) метатеґи (<meta>);
4) стилі форматування (<style>);
5) зв'язок (<link>);
6) сценарії JavaScript (<script>).

Порядок теґів у заголовку документа принципового значення не має.

1) **Заголовок сторінки (теґ <title>)** використовується для відображення рядка тексту на вкладці вікна браузера. Такий рядок повідомляє користувачеві назву сайту, яку додає розробник, наприклад:

```
<head>
    <title>Практичне заняття №1 </title>
</head>
```

Результат: 

Теґ <title> виконує ще й опосередковані задачі[9]:

– по тексту заголовка користувач отримує додаткову інформацію, що це за сайт, на який він зайшов, і як називається поточна сторінка. Не варто думати, що достатньо у документі вказати логотип сайту і проігнорувати заголовок, адже відвідувач може згорнути вікно. У згорнутому вигляді заголовок також відображається на кнопках панелі задач, тому можна легко орієнтуватися, з яким сайтом у цей момент працювати, а не перегортати їх по черзі;

– більшість браузерів підтримують можливість збереження веб-сторінки на локальний комп'ютер. У цьому разі ім'я збереженого файлу збігається з назвою заголовка документа. Якщо у тексті заголовка є символи неприпустимі для імені файлу (\ /: *? "<> |), вони будуть проігноровані або автоматично замінені іншими дозволеними символами;

– при збереженні у розділі браузера «Вибране» адреса поточної сторінки з її заголовком поміщається у список бажаних посилань. Оскільки цей список зазвичай зберігається у вигляді окремих файлів, до їхніх імен також працює вже вищеописане правило;

– у результатах пошуку за ключовими словами пошукові системи використовують заголовок сторінки як посилання на цей документ. Цікаво написа-

[9] Теґ <title> // Справочник по HTML. URL: http://htmlbook.ru/html/title (дата звернення 12.07.2018).

ний заголовок, що містить ключові слова, приверне більше уваги відвідувачів і збільшить шанси на те, що сайт відвідає більше людей.

2) **Теґ <base>** інструктує браузер про повну базову адресу поточного документа. Цей теґ призначений для документів, в яких використовується відносна адреса і ці документи можуть переноситись в іншу папку або навіть на інший комп'ютер без втрати зв'язку[10]. Браузер шукає теґ <base>, визначає повну адресу документа і коректно завантажує його.

Наприклад, якщо адресу вказано як <base href="http://it.inf.od.ua/">, то при додаванні зображень достатньо скористатись відносною адресою <img src="images/ris.gif">. При цьому повна адреса до зображення буде такою: http://it.inf.od.ua/images/ris.gif, що дозволить браузеру завжди найти графічний файл, незалежно від того, де міститься поточна веб-сторінка. Також можна застосовувати ієрархічну систему адреси з двома крапками. Так, якщо зображення додається як <img src="../images/ris.gif">, то повний шлях до файлу буде http://it.inf.od.ua/images/ris.gif.

Друге застосування теґу <base> – задавання цільового вікна для всіх посилань на поточну сторінку.

3) **Метатеґи (<meta>)** (англ. meta tags) використовуються для зберігання інформації призначеної для браузерів і пошукових систем. Наприклад, механізми пошукових систем звертаються до метатеґів для отримання опису сайту, ключових слів та інших даних. Дозволяється використовувати більш ніж один теґ, всі вони розміщуються в контейнері <head>. Зазвичай атрибути будь-якого метатеґу зводяться до пари "атрибут = значення", яка визначається атрибутами **content**, **name**, **charset** або **http-equiv**.

Значення атрибутів Description (опис поточного документа) і Keywords (ключові слова) призначені спеціально для пошукових серверів:

```
<meta name="Description" content="Сайт про HTML">
```

Більшість пошукових серверів відображують вміст поля **Description** при виведенні результатів пошуку. Якщо цього теґу нема на сторінці, то пошуковий движок просто перерахує перші слова, які зустрічаються на сторінці.

```
<meta name="Keywords" content="HTML, META, метатеґ">
```

Поле **Keywords** призначене для опису ключових слів, що зустрічаються на сторінці. Але в результаті дії користувачів, які бажають потрапити у верхні рядки пошукових систем будь-якими засобами, наразі Keywords дискредитований. Тому більшість пошукових систем просто пропускають цей параметр.

Щоб автоматично завантажувати новий документ через визначений проміжок часу, використовується атрибут **http-equiv**="refresh" (завантажити інший документ у поточне вікно браузера):

```
<meta http-equiv="refresh" content="5; URL=http://www.htmlbook.ru">
```

[10] Теґ <base> // Справочник по HTML. URL: http://htmlbook.ru/html/base (дата звернення 12.07.2018).

Щоб повідомити браузеру, в якому кодуванні подані символи веб-сторінки, треба задати атрибут **charset**:

<meta http-equiv="Content-Type" content="text/html; charset=ім'я кодування">

У стандарті HTML5 цей рядок може виглядати так:

<meta charset="utf-8">

Для операційної системи Windows у кирилиці аргумент charset звичайно має значення utf-8 чи windows-1251. Якщо вказівка на кодування відсутня, браузер намагається сам визначити, який тип символів використовується в документі, і вибирає потрібне кодування автоматично. Якщо браузер неправильно вгадає кодування, то замість тексту відображатимуться ієрогліфи, приміром, з в'єтнамським кодуванням замість кирилиці. З цієї причини треба завжди задавати кодування.

4) **Тег <style>** застосовується для визначення стилів елементів веб-сторінки. Стилі зберігають набір елементів форматування, який застосовується до тексту документа, щоб швидко змінити його зовнішній вигляд. Можна задавати більш ніж один тег <style>.

Синтаксис[11]: <style type="text/css"> ... </style>

Атрибути:

– media визначає пристрій виведення, для якого призначена таблиця стилів. Для кожного пристрою – від смартфона до принтера можна визначити свій власний стиль, який би враховував специфіку пристрою і підганяв під нього вигляд веб-сторінки[12];
– type повідомляє браузеру, який синтаксис використовувати, щоб правильно інтерпретувати стилі.

Закривний тег </style> обов'язковий.

Приклад:

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Тег STYLE</title>
  <style type="text/css">
    h1 {
     font-size: 120%;       color: #333366;
     font-family: Verdana, Arial, Helvetica, sans-serif;
    }
  </style>
  <style media="screen">
   .window {
     background: #333;
```

[11] Тег <style> // Справочник по HTML. URL: http://htmlbook.ru/html/style (дата звернення 12.07.2018).
[12] Атрибут media // Справочник по HTML. URL: http://htmlbook.ru/html/style/media (дата звернення 12.07.2018).

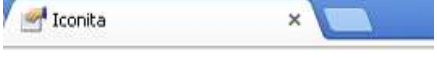
```
      border: 1px solid red;
      font-size: 90%;       color: #fc0;
      padding: 10px;
    }
  </style>
  <style media="print">
    .window {
     border: 1px solid black;
     font-family: Arial;
     font-size: 90%;
     font-weight: bold;
     color: black;
     padding: 10px
    }
  </style>
 </head>
 <body>
   <h1>Hello, world!</h1>
 </body>
</html>
```

Результат прикладу:

Тут колір тексту і накреслення шрифту в тегу <h1> змінились.

5) **Тег <link>** встановлює зв'язок із зовнішнім документом на зразок файлу зі стилями або з шрифтами. На відміну від тегу <a>, тег <link> розміщується завжди всередині контейнера <head> і не створює посилання. Закривний тег не потрібний.

```
  <title>Iconita</title>
  <link rel="stylesheet" type="text/css" href="/styles/htmlbook.css">
  <link rel="alternate" type="application/rss+xml"
  <link rel="shortcut icon" href="http://htmlbook.ru/favicon.ico">
```

Тут підключається зовнішній файл зі стилями за допомогою атрибута rel="stylesheet" тегу <link>, вказується RSS-документ поточного сайту (link rel="alternate" type="application/rss+xml") й задається іконка сайту в адресному рядку браузера (link rel="shortcut icon"). 

Атрибути:

– charset – кодування пов'язаного документа, наприклад: "charset=utf-8";
– href – місцерозташування пов'язуваного файлу;
– media – визначення пристрою, для якого треба застосувати стильове оформлення, наприклад:
<link media="all|braille|handheld|print|screen|speech|projection|tty|tv">
– all – всі пристрої; braille – пристрої, засновані на системі Брайля, призначені для сліпих людей; handheld – смартфони, пристрої з малою шириною екрана; print – принтери; screen – екрани монітора; speech – мовні синтезатори, а також програми для відтворення тексту вголос і мовні браузери; projection – проектори; tty – телетайпи, термінали, портативні пристрої з обмеженими можливостями екрана (для них не повинні використовуватися пікселі як одиниці виміру); tv – телевізор;
– rel – визначення співвідношення між поточним документом і файлом, на який робиться посилання, щоб браузер знав, як використовувати долучений документ. Можливі значення: stylesheet (файл зберігає таблицю стилів css), alternate (альтернативний тип, використовується, приміром, для зазначення посилання на файл у форматі XML для опису стрічки новин, анонсів статей), icon (іконка);
– sizes – визначення розміру іконок;
– type – MIME[13]-тип даних файлу, який підключається. Для файлів таблиць пов'язаних стилів застосовується тип text/css.

Крім цих атрибутів, для тегу <link> доступні універсальні атрибути[14].

6) **Тег <script>** призначений для опису скриптів і може містити посилання на програму або її частину тексту певною мовою. Скриптом традиційно називають програму, яка впроваджується в тіло веб-сторінки та виконує на ній певні дії. Поширеною мовою програмування для написання скриптів є JavaScript. Скрипти можуть розташовуватися у зовнішньому файлі і зв'язуватися з будь-яким HTML-документом. Такий підхід дозволяє використовувати одні й ті самі загальні функції на багатьох веб-сторінках і прискорює їхнє завантаження, оскільки зовнішній файл кешується при першому завантаженні, і скрипт викликається швидше при наступних викликах.

Тег <script> може розташовуватися у заголовку або тілі HTML-документа в необмеженій кількості. Здебільшого розташування скрипту ніяк не позначається на роботу програми. Однак скрипти, які повинні виконуватися в першу чергу, переважно розміщують у заголовку документа.[15]

Приклад перший:

---

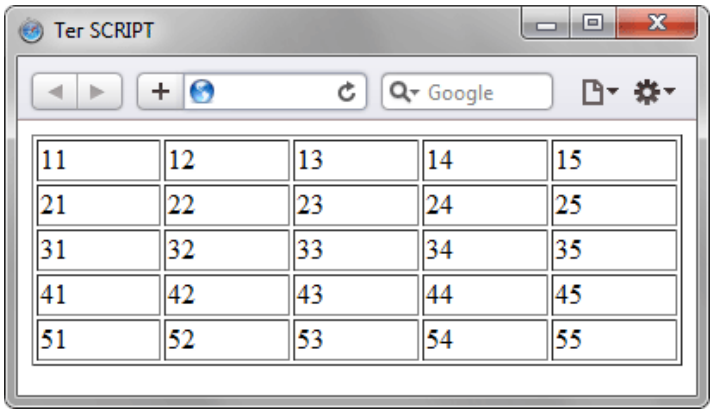
[13] MIME (Multipurpose Internet Mail Extension, багатоцільові розширення пошти інтернету) – специфікація для передавання мережею файлів різного типу: зображень, музики, текстів, відео, архівів тощо. Зазначення MIME-типу використовується в HTML зазвичай при передаванні даних форм і вставленні на сторінку різних об'єктів.

[14] Универсальные атрибуты // Справочник по HTML. URL: http://htmlbook.ru/html/attr/common (дата звернення 12.07.2018).

[15] Тег <script> // Справочник по HTML. URL: http://htmlbook.ru/html/script (дата звернення 12.07.2018).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/strict.dtd">
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Ter SCRIPT</title>
 </head>
 <body>
  <script type="text/javascript">
    document.write ('<table width="100%" border="1">');
    for (i=1; i<6; i++) {
      document.writeln("<tr>");
      for (j=1; j<6; j++) document.write("<td>" + i + j + "<\/td>");
      document.writeln("<\/tr>");
    }
    document.write ("<\/table> ");
  </script>
 </body>
</html>
```

Результат даного прикладу роботи скрипту:



11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

У цьому прикладі засобами скрипту виводиться таблиця з п’яти рядків і стовпців, яка заповнюється числами.

Приклад другий:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Ter SCRIPT</title>
  <script>
  function popup() {
   document.getElementById('welcome').innerHTML = Ласкаво просимо!';
  }
  </script>
```

```
 </head>
 <body onload="popup()">
   <div id="welcome"></div>
 </body>
</html>
```

В HTML5 атрибут type можна пропустити, він є необов'язковим і набуває значення text/javascript, коли не заданий явно. У попередніх версіях HTML атрибут type необхідний.

1.6. Визначення вмісту веб-сторінки – теґ <body>

Тіло документа призначене для відображення даних на веб-сторінці, зокрема, у тілі розміщується текст, зображення, посилання, таблиці, списки тощо.

1.6.1. Заголовки

Для заголовків в HTML є шість теґів від <h1> до <h6>. Вони призначені для створення заголовків до розділів та підрозділів і показують їхню відносну важливість. Так, зазвичай текст всередині теґу <h1> відображується в жирному накресленні розміром 24 пункти. Вміст теґу <h2> вже має розмір 18 пунктів, а <h3> – 14 пунктів.

Код HTML	Приклад
Заголовок	**Заголовок**
Заголовок	**Заголовок**
Заголовок	**Заголовок**
Заголовок	**Заголовок**
Заголовок	**Заголовок**
Заголовок	**Заголовок**

Пошукові системи дуже «полюбляють» заголовки. Тим самим заголовки підвищують цінність тексту на веб-сторінці. З цієї причини краще використовувати теґи <h1> ... <h6>, незважаючи на те, що за допомогою стилів будь-який текст можна задати великого розміру і, тим самим, зробити його як заголовок.

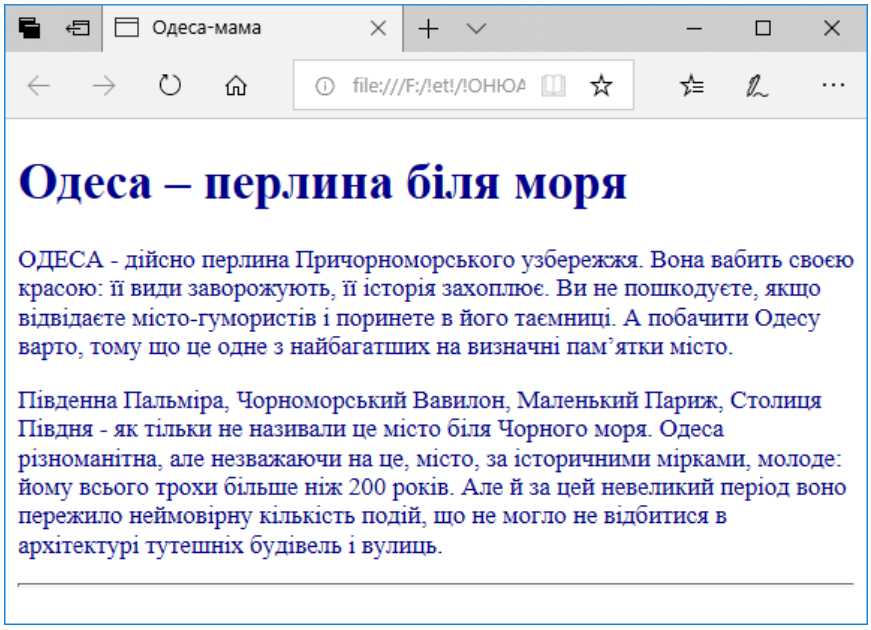
1.6.2. Елементи розміщення тексту на веб-сторінці

**Теґ ** визначає абзац в HTML. Цей теґ парний. Якщо закривного теґу немає, вважається, що кінець абзацу збігається з початком наступного блокового елемента.

Приклад:

```
<!DOCTYPE html>
<html>
```

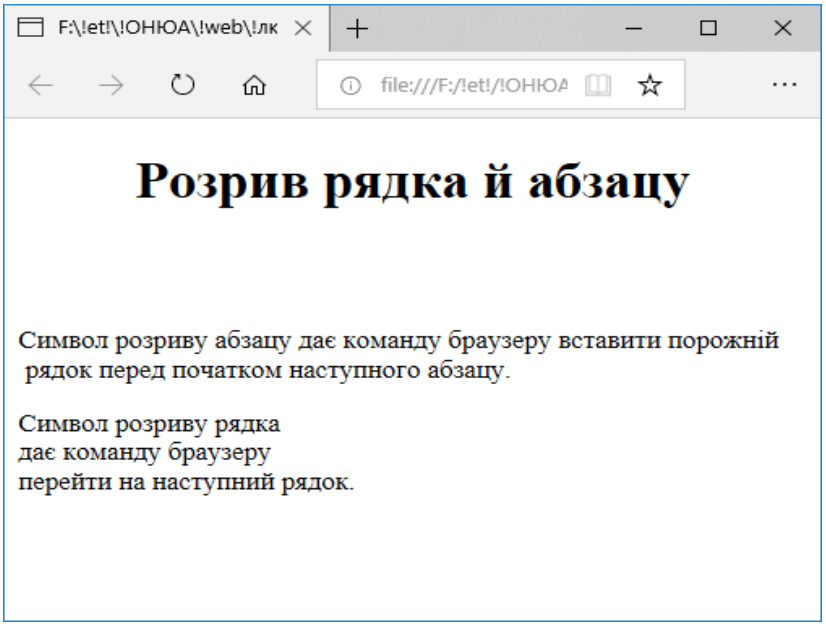
```
  <head>
    <title>Одеса-мама</title>
  </head>
  <body text="darkblue">
    <h1>Одеса – перлина біля моря</h1>
    <p>ОДЕСА – дійсно перлина Причорноморського узбережжя. Вона вабить своєю
       красою: її види заворожують, її історія захоплює. Ви не пошкодуєте, якщо відві-
       даєте місто-гумористів і поринете в його таємниці. А побачити Одесу варто, тому
       що це одне з найбагатших на визначні пам'ятки місто.</p>
    <p>Південна Пальміра, Чорноморський Вавилон, Маленький Париж, Столиця Півдня
       – як тільки не називали це місто біля Чорного моря. Одеса різноманітна, але не-
       зважаючи на це, місто, за історичними мірками, молоде: йому всього трохи більше,
       ніж 200 років. Але й за цей невеликий період воно пережило неймовірну кількість
       подій, що не могло не відбитися в архітектурі тутешніх будівель і вулиць.</p><hr />
  </body>
</html>
```



**Тег
** (або <br>) формує розрив рядка. Цей тег є одинарним.

Приклад:

```
<!DOCTYPE html>
<html>
<body>
   <h1 align=center>Розрив рядка й абзацу</h1>
   <br /><br /><br />
    <p>Символ розриву абзацу дає команду браузеру вставити порожній рядок перед по-
       чатком наступного абзацу. </p>
   <p>Символ розриву рядка<br /> дає команду браузеру<br />перейти на наступний ря-
       док.</p>
</body>
</html>
```



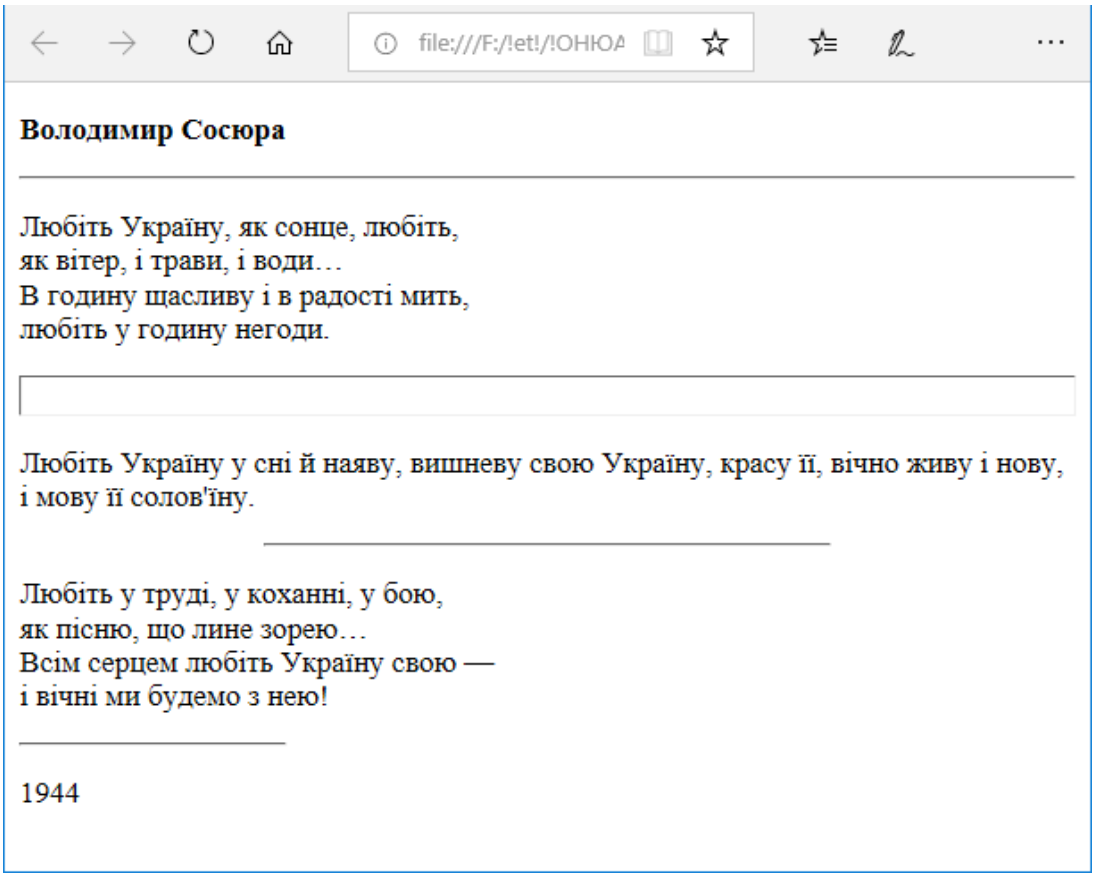
Тег <hr> (або **<hr />**) малює горизонтальну лінію, вигляд якої залежить від використовуваних параметрів, а також браузера. Тег <hr> є блоковим елементом, лінія завжди починається з нового рядка, а після неї всі елементи виводяться з нового рядка.

Атрибути:

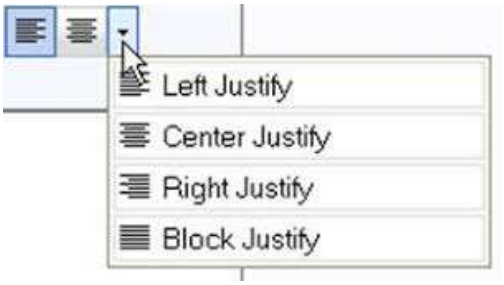
align – вирівнювання лінії; color – колір лінії;
size – товщина лінії; width – ширина лінії.

Приклад:

```
<body>
  <p><b>Володимир Сосюра</b></p>
<hr>
  <p>Любіть Україну, як сонце, любіть,<br>як вітер, і трави, і води…
      <br>  В годину щасливу і в радості мить,<br>любіть у годину негоди.
 </p>
   <hr size=20>
   <p>Любіть Україну у сні й наяву,
       вишневу свою Україну,
       красу її, вічно живу і нову,
       і мову її солов'їну.</p>
   <hr width="300">
   <p> Любіть у труді, у коханні, у бою,<br>
      як пісню, що лине зорею… <br>
      Всім серцем любіть Україну свою – <br>
      і вічні ми будемо з нею!
   </p>
   <hr width="25%" align=left>
   <p>1944</p>
</body>
```



1.6.3. Вирівнювання тексту



Вирівнювання тексту визначає його зовнішній вигляд, орієнтацію країв абзацу та може виконуватися за лівим або правим краєм, по центру або за шириною.

Вирівнювання реалізується теґом <align>. Теґ може мати одне з чотирьох значень:

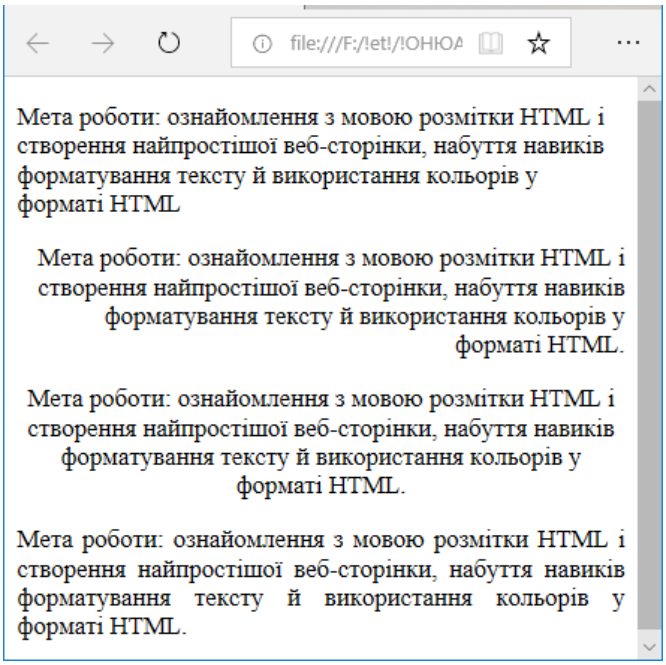
align="left | center | right | justify"

Найпоширеніший варіант – вирівнювання за лівим краєм, який задається усталено. Вирівнювання по центру використовується здебільшого у заголовках і короткому змісті. Варто зважати на те, що при використанні вирівнювання за шириною в тексті між словами можуть з'явитися великі інтервали, що не гарно виглядає на екрані при перегляданні веб-сторінки.

Приклад:

```
<!DOCTYPE html>
<html>
<body>
  <p align=left>Мета роботи: ознайомлення з мовою розмітки HTML і створення найпростішої веб-сторінки, набуття навиків форматування тексту й використання кольорів у форматі HTML</p>
```

```
<p align=right>Мета роботи: ознайомлення з мовою розмітки HTML і створення найпростішої веб-сторінки, набуття навиків форматування тексту й використання кольорів у форматі HTML.</p>
<p align=center>Мета роботи: ознайомлення з мовою розмітки HTML і створення найпростішої веб-сторінки, набуття навиків форматування тексту й використання кольорів у форматі HTML.</p>
<p align=justify>Мета роботи: ознайомлення з мовою розмітки HTML і створення найпростішої веб-сторінки, набуття навиків форматування тексту й використання кольорів у форматі HTML.</p>
</body>
</html>
```



Якщо замість теґу <p> використовувати теґ , то на початку і наприкінці параграфа не з’являться вертикальні відступи.

```
<div align="center">Текст</div>
```

Параметр align достатньо універсальний і може застосовуватися не тільки до основного тексту, а й до заголовків, наприклад, <h1>.

Преформатований текст. Для виведення преформатованого тексту призначений теґ <pre>. Це парний теґ, який зберігає авторське форматування тексту, наприклад, взятого з якогось іншого документа. При цьому контейнер <pre> дозволяє зберегти форматування тексту таким, яким воно було при копіюванні, оскільки символи переходу на новий рядок він інтерпретує як розриви рядка. Пробіли всередині тексту інтерпретуються в точній відповідності до їхнього розташування у текстовому редакторі. Дескриптори абзаців і заголовків, що випадково потрапили до контейнера <pre>, не інтерпретуються. Але в контейнері <pre> можна використати дескриптори стилю й дескриптор <a> для створення прив’язок (посилань).

Синтаксис: <pre>Текст</pre>

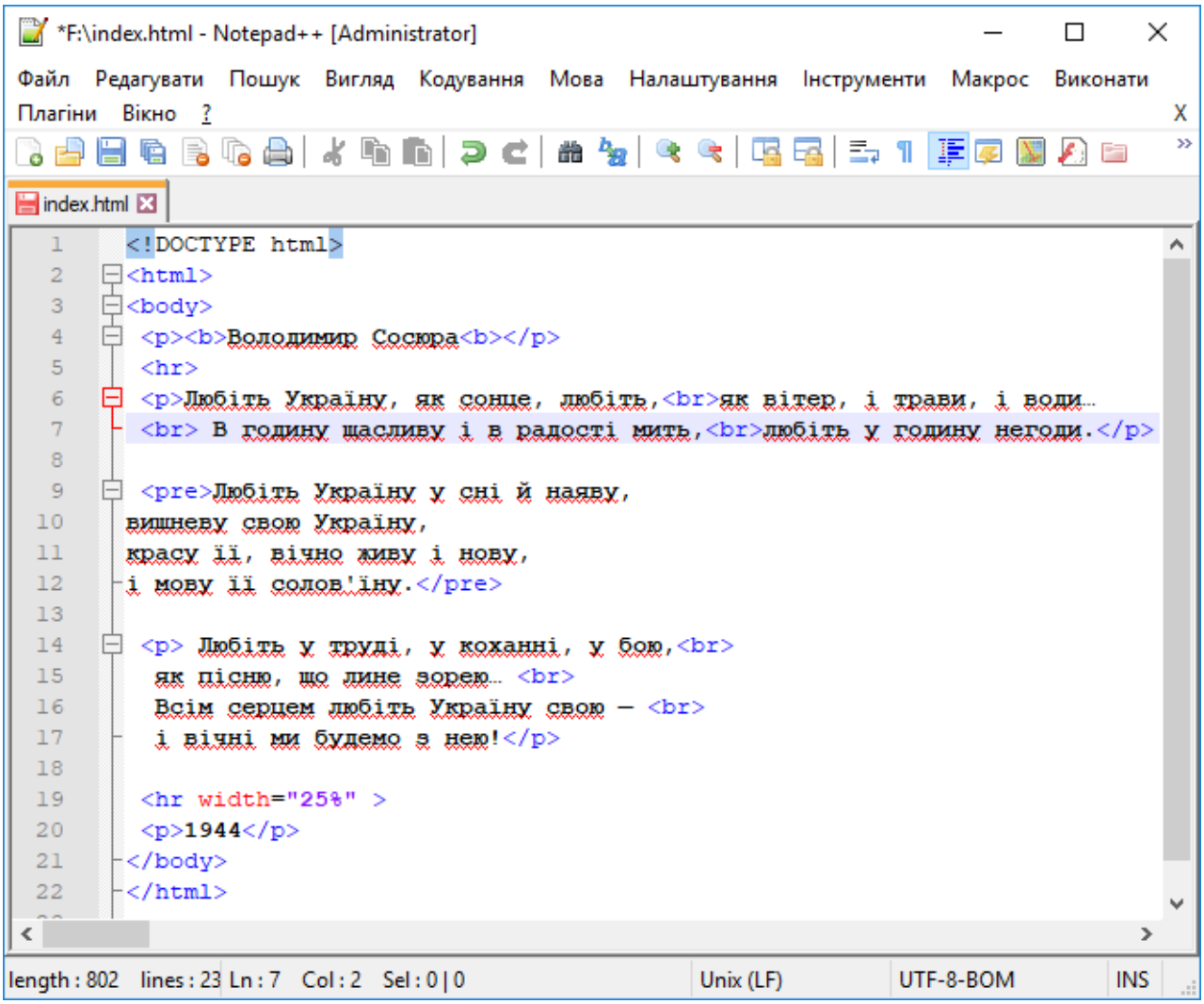
Текст в елементі <pre> виводитиметься шрифтом з фіксованою шириною зі збереженням кількості пробілів і переходів на новий рядок.

Теґ підтримується усіма браузерами.

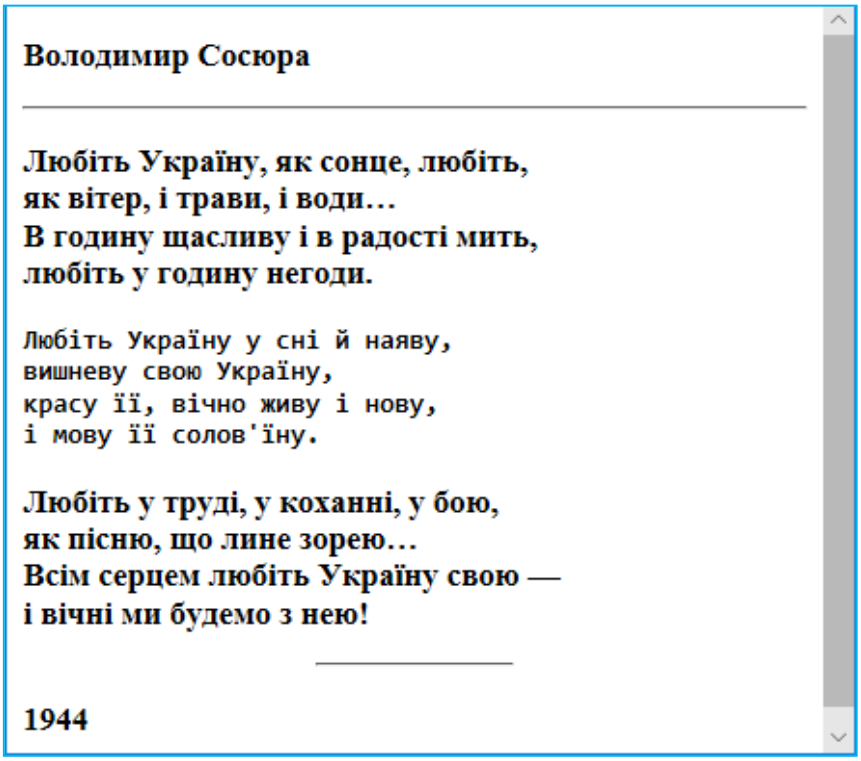
Атрибут width задає максимальну кількість символів в одному рядку.

Атрибут не підтримується в HTML5, оскільки рекомендується використовувати стилі.

Приклад використання теґу <pre> в HTML-редакторі Notepad++.



Результат при перегляданні у браузері:



1.6.4. Форматування тексту

Форматування тексту – це засоби змінення його вигляду: накреслення, колір і розмір шрифту, вирівнювання, відступи та інші параметри абзаців. У табл. 1.1 наведено основні теґи для змінення оформлення тексту.

Використання фізичних і логічних дескрипторів стилів

В HTML передбачено кілька способів форматування тексту. Це явне (або абсолютне) форматування засобами **фізичних стилів**; неявне (або відносне) форматування за допомогою **логічних стилів**; змінення накреслення, кольору і розміру шрифту теґом **<font>**.

Текст, виділений фізичним стилем, в усіх браузерах відображається однаково. Логічні стилі у різних браузерах відображаються по-різному. Дескриптори (теґи) фізичних стилів наведені у табл. 1.1.

Таблиця 1.1

Дескриптори фізичних стилів

Дескриптор	Стиль
<b>	Напівжирний шрифт
<i>	Курсив
<tt>	Моноширинний шрифт
<u>	Підкреслення
<sub>	Підрядковий текст (нижній індекс)
<sup>	Надрядковий текст (верхній індекс)
<strike>	Перекреслення

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

Таблиця 1.2

Дескриптори логічних стилів

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

Слід зазначити, що теґи <b> та , <i> та не є тотожними. Теґ <b> є теґом фізичної розмітки, який задає жирний текст, а теґ – теґом логічної розмітки, який визначає важливість поміченого тексту. Такий поділ теґів на

логічне та фізичне форматування спочатку призначався для того, щоб зробити HTML універсальним, зокрема незалежним від пристрою виведення інформації. Теоретично, якщо скористатися, наприклад, мовним браузером, текст, оформлений за допомогою теґів і , буде подано по-різному. Однак нині у популярних браузерах результат використання цих теґів рівнозначне.

Теґи <big> і <small> можна повторювати кілька разів поспіль, тим самим збільшуючи або зменшуючи текст до потрібних розмірів.

Таблиця 1.3

Приклади використання теґів форматування тексту

Код HTML	Опис	Приклад
Текст	Жирне накреслення тексту	**Текст**
Текст	Курсив тексту	*Текст*
e=mc²	Верхній індекс	$e=mc^2$
H ₂O	Нижній індекс	H_2O
<pre>Текст</pre>	Блок тексту моноширинним шрифтом	Текст
Текст	Дрібний шрифт (на 1 менший звичайного)	Текст
<big> Текст </big>	Великий шрифт (на 1 більший звичайного)	Текст
<u> Текст</u>	Підкреслений текст	Текст
<ins> Текст</ins>	Підкреслений текст	Текст
<del> Текст</del>	Закреслений текст	~~Текст~~
<strike> Текст</strike>	Закреслений текст	~~Текст~~
<mark> Текст</mark>	Виділення маркером жовтого кольору	Текст

Приклад застосування теґів із таблиць 1.1 та 1.2:

```
<!DOCTYPE html>
<html>
  <body>
    <p>Відомий сучасний журналіст <b>Матвій Ганапольський</b> стверджує: <i> «Все починається з віри в свої сили. Особливо в <u>журналістиці</u>».</i></p>
    <p>Американський журналіст і видавець, творець всесвітньо відомих журналів Time <small>(1923)</small>, Fortune <small>(1930)</small>, Life <small>(1936) </small> та інших видань <strong>Генрі Люс</strong> сказав: <em>«Я став <ins>журналістом</ins>, щоб максимально близько підійти до <mark>серця світу</mark>». </em></p>
  </body>
</html>
```

Результат:

Відомий сучасний журналіст **Матвій Ганапольський** стверджує: *«Все починається з віри в свої сили. Особливо в журналістиці».*

Американський журналіст і видавець, творець всесвітньо відомих журналів Time (1923), Fortune (1930), Life (1936) та інших видань **Генрі Люс** сказав: *«Я став журналістом, щоб максимально близько підійти до серця світу».*

Приклад застосування теґів <del> та <ins>:

```
<!DOCTYPE html>
<html>
  <body>
    <p>Мене надихають <del>кактуси</del> <ins>рози</ins>!</p>
  </body>
</html>
```

Результат:

Мене надихають ~~кактуси~~ рози!

Тег <font> дозволяє визначити розмір і колір шрифту.

Атрибут size задає абсолютний розмір шрифту (він може набувати значення від 1 до 7) або відносний, стосовно розміру шрифту, використовуваного в основній частині сторінки (він може набувати значень від -4 до +4):

```
<font size=значення>
```

Для змінення розміру основного шрифту документа використовується елемент basefont:

```
<basefont size=значення>
```

Використання кольорів для змінення тексту і фону сторінки

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

```
<font color=#rrggbb>текст</font>
```

або

```
<font color=red>текст</font>
```

Запис #rrggbb складається з трьох двознакових шістнадцяткових чисел, які визначають співвідношення червоного, зеленого і блакитного кольорів (табл. 1.4), наприклад:

```
<font color=#a045ff>текст</font>
```

Таблиця 1.4.

Кольори в HTML

Колір	Шіснадцятковий код кольору	Колір у системі RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

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

Цей атрибут має формат:

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

Цитати

В HTML існує декілька теґів для позначення цитат:

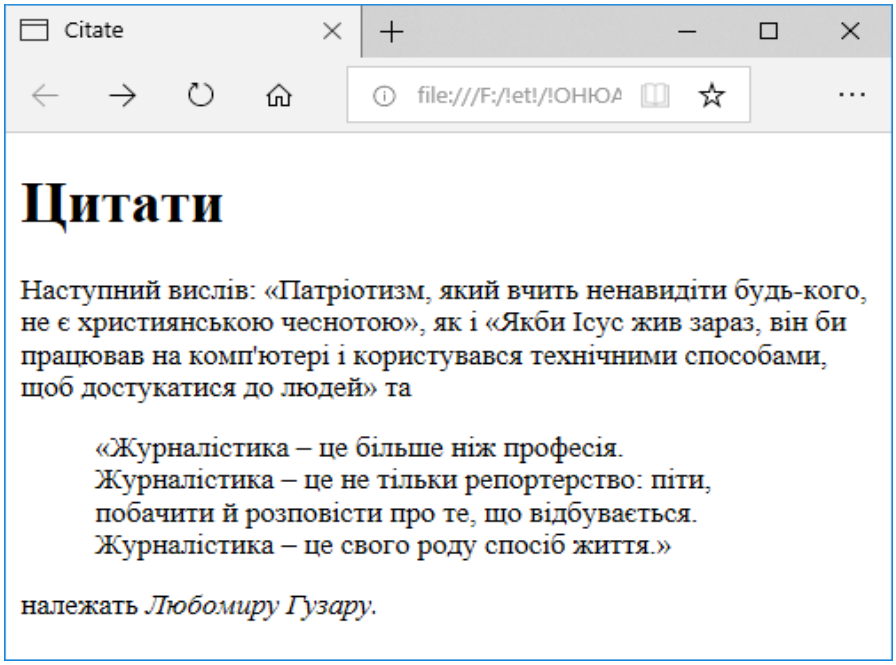
– <blockquote> призначений для довгих цитат, які можуть складатися з декількох абзаців. Теґ виділяє цитату як окремий блок тексту з відступами;
– <q> призначений для коротких цитат у тексті речення. Текст всередині цього теґу автоматично обрамляється лапками;
– <cite> використовують для виділення джерела цитати, назви твору або автора цитати.

Усі ці теґи є парними.

Приклад:

```
<!DOCTYPE html>
<html>
<head>
   <title> Quotes </title>
   <meta charset="UTF-8" />
</head>
<body>
   <h1>Цитати</h1>
   <p>Наступний вислів: <q>Патріотизм, який вчить ненавидіти будь-кого, не є християнською чеснотою</q>, як і <q>Якби Ісус жив зараз, він би працював на комп'ютері і користувався технічними способами, щоб достукатися до людей</q> та <blockquote>
   Журналістика – це більше ніж професія. Журналістика – це не тільки репортерство: піти, побачити й розповісти про те, що відбувається. Журналістика – це свого роду спосіб життя.</blockquote>
   належать
   <cite>
   Любомиру Гузару.
   </cite>
   </p>
</body>
</html>
```

Результат:



Адреса й абревіатура

HTML-тег <address> визначає контактну інформацію автора документа або статті. Вміст елемента <address> зазвичай виводиться italic-ом. Більшість браузерів виводить адресу з нового рядка.

Синтаксис: <address>Вміст</address>

Тег <abbr> визначає абревіатуру.

Синтаксис: <abbr title="опис">Абревіатура</abbr>

Приклад:

```
<body>
  <p>Факультет журналістики в  <abbr title="Національний університет Одеська
   юридична академія"> НУ «ОЮА»</abbr></p>
  <address>Адреса: м. Одеса<br />вул. Академічна, 7</address>
</body>
```

Результат:

Факультет журналістики в НУ «ОЮА»

Адреса: м. Одеса
вул. Академічна, 7

Гіпертекст і посилання

Посилання є майже на кожній веб-сторінці для організації переходів всередині сторінки або для переходу на інші веб-сторінки. Посиланням / гіперпосиланням може бути слово, група слів або зображення. Позначене місце переходу називають якорем (anchor).

Зазвичай в усіх браузерах посилання мають колір:

– синій, коли вони не були відвіданими;
– фіолетовий, якщо їх відвідали;
– червоний – для активних посилань.

Визначити посилання в HTML можна **тегом <a>** (anchor).

Атрибут href у тегу <a> задає URL-адресу посилання:

```
<a href="URL"> Текстове посилання </a>
```

Можливі значення для "URL":

Абсолютна URL-адреса, яка визначає зовнішній ресурс іншого веб-сайту:

```
<a href="http://omr.gov.ua/ua/symbolics/">Деталі дивись тут</a>
```

Відносна URL-адреса, яка визначає файл цього самого сайту:

```
<a href="герб.jpg">Герб міста</a>
```

Посилання на елемент тої самої сторінки, заданий ідентифікатором (якорем): href="#top"

Інші протоколи: ftp://, mailto:, file: etc.

```
<a href="mailto: incognito@gmail.com>email</a>
```

Скрипт: href="javascript:alert('Hello');"

Якщо не вписати **атрибут href** у тег <a>, то посилання не буде доступним.

Приклад:

```
<!DOCTYPE html>
<html>
  <head><title>Інформація про Одесу</title></head>
  <body>
    <h1>Одеса</h1>    <hr />
    <a href="герб.jpg">Герб міста</a>    <br />
    <a href="http://omr.gov.ua/ua/symbolics/">
            Деталі дивись тут</a>
 </body>
</html>
```

Результат:

Коли треба створити зв'язок між розділами одного документа (наприклад, коли документ – книга, підручник або курс), можна створити зміст для кожного розділу цього документа.

Щоб створити внутрішні посилання, використовується **атрибут name**, який задає ключове слово, що однозначно ідентифікує необхідний параграф.

Цей атрибут підтримується більшістю браузерів.

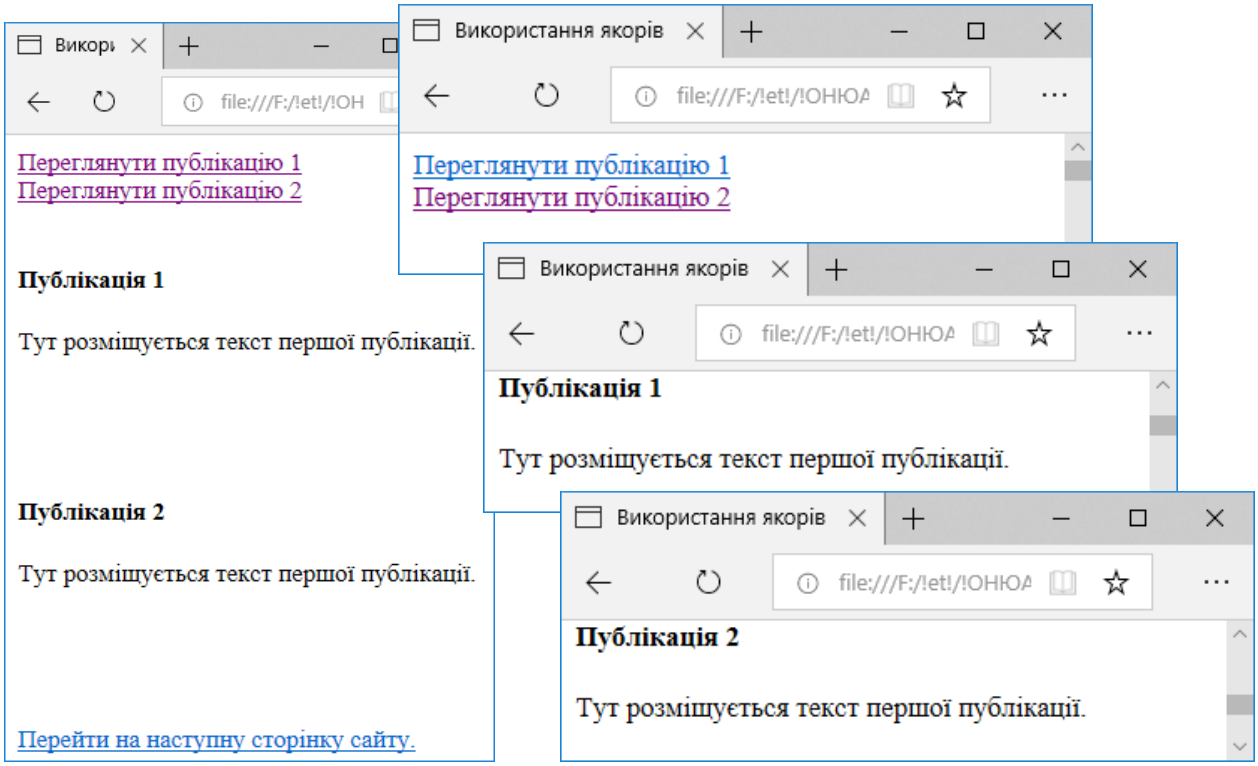
Синтаксис: Параграф

Цей атрибут не використовується більше в HTML5 – він був змінений на атрибут ID**:** Параграф

Посилання задається: Текст посилання

Приклад використання атрибута ID тегу <a>:

```
<html>
<head>
  <title>Використання якорів</title>
</head>
<body>
   <a href="#p1">Переглянути публікацію 1</a>
   <br />
   <a href="#p2">Переглянути публікацію 2</a>
   <br /><br />
   <h4><a ID="p1">Публікація 1</a></h4>
   <p>Тут розміщується текст першої публікації.</p>
   <br /> <br /> <br />
   <h4><a ID="p2">Публікація 2</a></h4>
   <p>Тут розміщується текст першої публікації.</p>
   <br /><br /><br />
   <a href="next.html">Перейти на наступну сторінку сайту.</a>
</body>
</html>
```



Атрибут target тегу <a>. Усталено при натисканні посилання ресурс відкривається у тій самій вкладці. Щоб задати відкривання у новій вкладці чи то вікні, треба скористатися атрибутом target. Цей тег підтримується більшістю веб-браузерів.

Синтаксис:

<a target="_blank|_self|_parent|_top|framename">

Значення	Опис
_blank	Відкриває документ у новій вкладці браузера
_self	Запускає ресурс на поточній вкладці, задається усталено
_parent	Відкриває сторінку у батьківському фреймі (вікні). Якщо фреймів немає, то це значення працює як _self
_top	Скасовує усі фрейми і завантажує сторінку у повному вікні, поверх решти, на увесь екран. Якщо фреймів немає, то працює як _self

До речі, **тег <body> може мати декілька атрибутів**, які вже не підтримуються версією HTML5:

Атрибут	Опис
alink	Колір активного посилання
link	Колір посилання, яке не активувалось
vlink	Колір посилання, яке вже було активоване
background	Фонове зображення сторінки
bgcolor	Фоновий колір сторінки
text	Колір тексту веб-сторінки

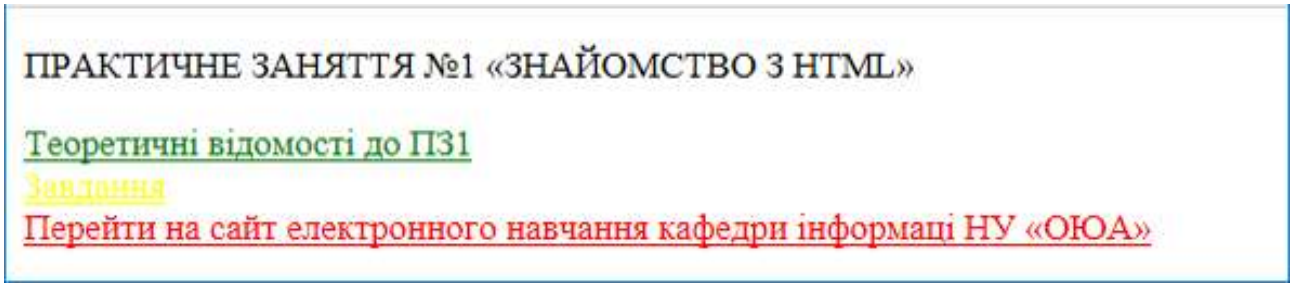
Синтаксис атрибутів alink, link, vlink:

<body alink=«назва_кольору | шіснадцятковий_код_кольору |номер_rgb»>
<body link= «назва_кольору | шіснадцятковий_код_кольору |номер_rgb»>
<body vlink= «назва_кольору | шіснадцятковий_код_кольору |номер_rgb»>

Значення	Опис
назва_кольору	Колір посилання із зазначенням назви кольору: aqua, black, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, white, yellow тощо
шістнадцятко-вий_код_кольору	Колір посилання зазначається шіснадцятковим кодом: "#00FF00", "#ff0000". Можна задати один з 65 536 кольорів
номер_rgb	Колір посилання здається кодом rgb: "rgb(255,0,0)"

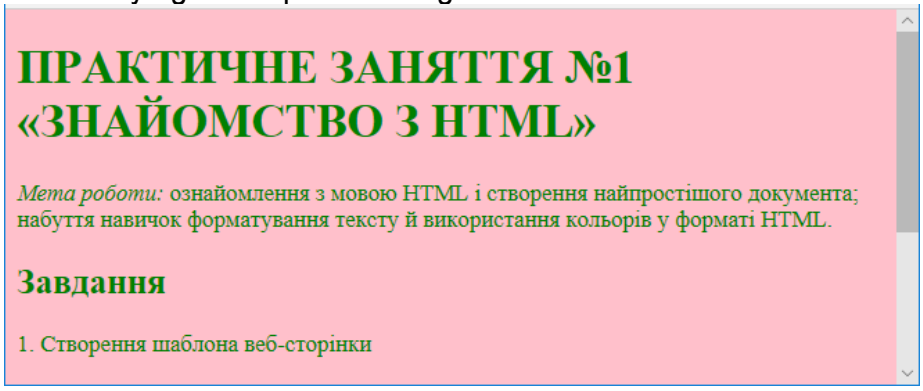
Приклад використання alink, link, vlink

```
<body alink="green" link="yellow" vlink="red">
   <p>ПРАКТИЧНЕ ЗАНЯТТЯ №1 «ЗНАЙОМСТВО З HTML»</p>
   <a href="http://it.inf.od.ua/mod/page/view.php?id=1494">Теоретичні відомості до ПЗ1</a>
   <br />
   <a href="http://it.inf.od.ua/mod/page/view.php?id=1493">Завдання</a>
   <br />
   <a href="http://it.inf.od.ua/">Перейти на сайт електронного навчання кафедри інформа-
   ційних технологій НУ «ОЮА»</a>
</body>
```



Приклад використання атрибутів bgcolor і text:

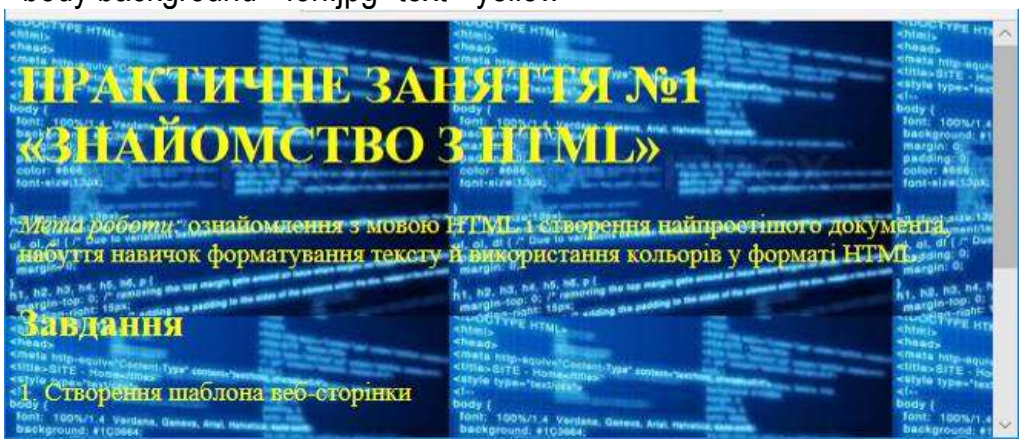
```
<body bgcolor="pink" text="green">
```



Інші приклади:

<body bgcolor=#000000> – чорний колір;

<body bgcolor=#ffffff> – білий колір;
<body bgcolor=#ff0000> – червоний колір;
<body bgcolor=#702499> – фіолетовий колір.
Приклад використання атрибутів background і text:
<body background="fon.jpg" text="yellow">



1.6.5. Засоби HTML для роботи з таблицями

Створення таблиць

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

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

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

Створення таблиці в HTML вимагає певної структури з наявністю обов’язкових теґів:

<table> – відкрити таблицю;

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

Закривний тег для <table> – обов'язковий.

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

Наприклад:

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

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

Можна вдосконалити структуру (ієрархію) цієї таблиці, задавши головну (thead), основну (tbody) і нижню частини (tfoot) таблиці в такий спосіб[16]:

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

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

[16] Таблицы в HTML. URL: https://webref.ru/course/html-content/tables (дата звернення 25.09.2018).

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

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

При використанні теґів <thead>, <tbody> і <tfoot> треба дотримуватись правил:

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

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

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

Крім об'єднання групи рядків через <thead>, є ще один спосіб групування клітинок – по колонках за допомогою елементів **<col>** і **<colgroup>**. Кожна колонка (стовпець) таблиці зіставляється зі своїм елементом <col>, який пишеться після

відкривного тегу <table>. Частину колонок можна об'єднати за допомогою атрибута span, його значенням є число поєднуваних колонок. Елементи <col> припустимо розміщувати в один або кілька <colgroup>, об'єднуючи тим самим колонки.

Елементи <col> і <colgroup> ніяк не впливають на вигляд таблиці та її відображення у браузері і застосовуються тільки зі стилями, як буде показано далі у прикладі[17]:

```
<section>
 <style>
  table {
    border-collapse: collapse; /* Прибрати подвійні межі */
    border: solid thick;
  }
  colgroup, tbody { border: solid medium;   }
  td {
      border: solid thin;
      height: 1.4em; width: 1.4em;
      text-align: center;
      padding: 0; /* Поля в клітинках */
  }
 </style>
 <h1>Today's Sudoku</h1>
 <table>
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <tbody>
   <tr> <td> 1 <td>   <td> 3 <td> 6 <td>   <td> 4 <td> 7 <td>   <td> 9
   <tr> <td>   <td> 2 <td>   <td>   <td> 9 <td>   <td>   <td> 1 <td>
   <tr> <td> 7 <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td> 6
  <tbody>
   <tr> <td> 2 <td>   <td> 4 <td>   <td> 3 <td>   <td> 9 <td>   <td> 8
   <tr> <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td>   <td>
   <tr> <td> 5 <td>   <td>   <td> 9 <td>   <td> 7 <td>   <td>   <td> 1
  <tbody>
   <tr> <td> 6 <td>   <td>   <td>   <td> 5 <td>   <td>   <td>   <td> 2
   <tr> <td>   <td>   <td>   <td>   <td>   <td> 7 <td>   <td>   <td>
   <tr> <td> 9 <td>   <td>   <td> 8 <td>   <td> 2 <td>   <td>   <td> 5
 </table>
</section>
```

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

[17] The table element // HTML5: Edition for Web Authors. URL: https://www.w3.org/TR/2011/WD-html5-author-20110809/the-table-element.html (дата звернення 25.09.2018).

Today's Sudoku

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

Атрибути для форматування таблиці

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

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

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

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

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

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

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

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

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

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

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

rules – повідомляє браузеру, як відображати межі між клітинками (в CSS аналог – border). Значеннями можуть бути: all (лінія навколо кожної клітинки таблиці); groups (лінія між групами, які утворені теґами <thead>, <tfoot>, <tbody>, <colgroup> або <col>); cols (лінія між колонками); none (без меж, усталено); rows (межа між рядками, створеними через тег <tr>);

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

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

Приклад можливого використання цих атрибутів:

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

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

Особливості структурування таблиць

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

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

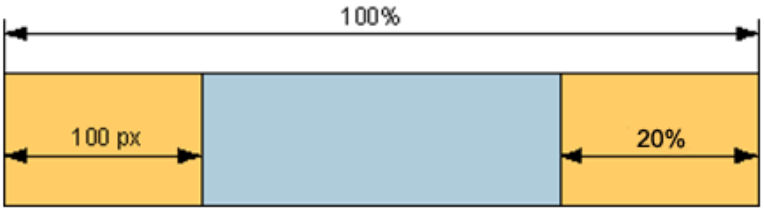
Розміри таблиці спочатку не задають, вони обчислюються на основі вмісту клітинок. Так, загальна ширина визначається автоматично як сумарна ширина вмісту усіх клітинок плюс ширина меж між клітинками, полів навколо вмісту та відстані між клітинками.

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

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

Поєднання відсотків і пікселів при задаванні ширини

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



Приклад програмного коду:

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

Розглянемо засоби створення і форматування таблиці під час створеня веб-сторінки зі статистично-довідковими даними про Одесу:

Довідково-статистичні відомості про Одесу

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

Повернутись на головну сторінку

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title> Таблиця </title>
      <meta name="keywords" content=«довідково-статистичні відомості про Одесу, дані
            довідкові про Одесу, статистичні дані про Одесу»>
      <style type="text/css">
      table { border: 2px solid gray;                  /* стиль для таблиці */
              border-collapse: collapse;
              background-color: #F0F0FF;               /*світло-блакитний*/
             }
      td    { border: 1px solid  gray;                 /* стиль для клітинки */
              padding: 5px;
             }
      h1    { text-align: center;
              font-size: 28px;  font-family: inherit;
              color: #618ad2;
             }
      </style>
   </head>
   <body background="photo/fon.gif">
   <h1>Довідково-статистичні відомості про Одесу</h1>
   <hr size=3 width="50%" align=center>
   <table >
      <tr>
         <td>Населення</td>       <td>1 010 912 (01.05.2017)</td>
         <td rowspan="2" width="30%">Густота населення<br>6224 осіб/км²</td>
      </tr>
      <tr>     <td>Площа</td>     <td>162.42 км²</td>          </tr>
      <tr>
         <td>Координати</td>     <td> 46°29'08" пн. ш.</td> <td> 30°44'36" сх. д.</td>
      </tr>
      <tr>
         <td>Засновано</td>       <td colspan="2">XV століття</td>
      </tr>
      <tr>
         <td>Міста-побратими</td>
         <td colspan="2">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя, Єреван,
         Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель, Нікосія, Оулу,
         Пірей, Реґенсбург, Сеґед, Спліт, Стамбул, Хайфа, Циндао</td>
      </tr>
      <tr>
         <td>Поділ міста</td>       <td colspan="2">4 райони</td>
      </tr>
```

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

Можна для таблиці на веб-сторінці задати обтіканням текстом, наприклад, розмістивши її у правому верхньому куті поруч з текстом:

В такому разі для розміщення таблиці поруч з текстом треба використовувати структурний тег (або тег), для якого за допомогою атрибута <id> (унікальний ідентифікатор елемента по всьому документу) задати спеціально створений стиль розміщення з параметрами обтікання (float: right). Тобто треба розмістити тег <table> разом з усім його вмістом усередину тегу :

```
    <body>
        <div id="placetabl">
            <table>
            . . . . . .
            </table>
        </div>
            . . . . . .
```

Крім того, до стилів треба додати стиль ідентифікатора та дещо змінити стиль клітинки td, задавши їй поля (відступи у 5 пікселів).

```
td {
   border: 1px solid  gray;   padding: 5px;
}
#placetabl {
   float: right;
   width: 30%;
   padding-left: 15px; padding-top: 40px;
}
```

1.6.6. Розміщення зображень на веб-сторінці

Вбудовування зображень тегом <img>

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

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

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

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

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

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

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

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

Крім того, для зазначення розмірів зображення можуть бути використані властивості width і height в CSS. Коли водночас використовуються властивості CSS й атрибути HTML, то властивості CSS матимуть пріоритет над атрибутами HTML.

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

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

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

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

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

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

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

sizes – задає розміри зображення для різних макетів сторінки;

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

Тег <picture>

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

– для екранів Retina планшетів iPad можна показувати картинку більшого розміру;

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

Синтаксис теґу <picture> є таким:

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

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

Наприклад:

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

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

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

Формати зображень

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

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

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

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

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

Позиціонування (position) зображень

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

Щоб задати позиціонування елемента, при зазначення властивості **position** використовують властивості:[18] top (зверху), bottom (знизу), left (зліва) та right (справа). Наприклад, розмістити картинку у верхньому куті з відступами згори і справа у 30px можна в такий спосіб:

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

Види позиціонування:

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

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

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

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

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

fixed – фіксоване позиціонування закріплює елемент у певному місці екрана. Тим самим таке позиціонування дозволяє розмістити елемент у будь-якій точці відносно вікна браузера. При прокручуванні фіксовано позиціонований елемент не прокручується і не змінює своє розташування. Фіксовано позиціоновані елементи можуть перекривати інші елементи.

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

inherit – елемент успадковує значення від батьківського елемента.

Рядкове позиціонування зображень

Елемент <img> є усталено рядково-блоковим. Додавання зображення на сторінку без будь-яких стилів позиціонуватиме його на тому самому рядку, що і

[18] Для усіх видів позиціонування, крім static.

вміст навколо зображення. Крім того, зміниться висота рядка, в якому з'явиться зображення, щоб відповідати висоті зображення, і це може створити великі вертикальні зазори у рядку.

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

Блокове позиціонування зображень

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

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

Позиціонування зображень ліворуч або праворуч (обтікання зображення)

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

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

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

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

Доволі часто при блоковому розміщенні декількох зображень на сторінці виникає потреба перевірки того, що вони не перевищують ширину відведених колонок (блоків). Для цього можна в окремо створеному класі з ім'ям effigy задати їхню максимальну ширину, як 100% від ширини блока. Це дозволить усім зображенням підлаштовуватись під ширину відведених колонок. Крім того,

можна трохи закруглити кути зображень і застосувати до них нижній відступ (властивість margin), наприклад, у 22 пікселі:

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

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

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

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Зображення</title>
  <style>
   figcaption {
    text-align: center;
   }
   .left {
    float: left;                  /* вирівнювання за лівим краєм */
    margin: 0 1em 1em 0;    /* відступ справа і знизу */
   }
   .right {
    float: right;                 /* вирівнювання за правим краєм */
    margin: 0 0 1em 1em;    /* відступ знизу і зліва */
   }
  </style>
 </head>
 <body>
  <figure class="left">
   <img src="images/fig.jpg" width="200" alt=" " >
   <figcaption>Підпис знизу</figcaption>
  </figure>
  <p>Текст</p>
 </body>
</html>
```

Вирівнювання зображення відносно базової лінії тексту

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

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

Синтаксис властивості **vertical-align**:

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

Значеннями властивості vertical-align можуть бути:

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

bottom – вирівнює основу елемента по нижній частині елемента;

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

sub – нижній індекс без змінення розміру шрифту;

super – верхній індекс без змінення розміру шрифту;

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

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

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

inherit – успадковує значення батьківського елемента.

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

Приклад:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>vertical-align</title>
 </head>
<body>
```

```
  <div style="font-size: 2em">
  T<span style="vertical-align: sub">E</span>X та L
  <span style="vertical-align: 5px;  font-size: 0.8em">A</span>T
  <span style="vertical-align: sub">E</span>X
 </div>
</body>
</html>
```

Трансформація зображень

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

```
transform: функція (параметри);
```

Наведемо приклад програмного коду для змінення масштабу фотографій про видатні місця Одеси при наведенні на них вказівника миші такого вигляду:


```
<html>
  <head>
    <meta charset="utf-8">
    <title> Фото Одеси</title>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta name="keywords" content="Одеса, видатні місця Одеси">
    <meta name="description" content="Видатні місця Одеси">
  </head>
  <body background="photo/fon.gif">
   <div class="thumbs">
```

```
      <h1 aligment="center">Видатні місця Одеси</h1>
      <img src="photo/img1.jpg" alt=" "  class="thumb">
      <img src="photo/img2.jpg" alt=" "  class="thumb">
      <img src="photo/img3.jpg" alt=" "  class="thumb">
      <img src="photo/img4.jpg" alt=" "  class="thumb">
      <img src="photo/img5.jpg" alt=" "  class="thumb">
      <img src="photo/img6.jpg" alt=" "  class="thumb">
      <img src="photo/img7.jpg" alt=" "  class="thumb">
      <img src="photo/img8.jpg" alt=" "  class="thumb">
      <img src="photo/img9.jpg" alt=" "  class="thumb">
      <img src="photo/img10.jpg" alt=" " class="thumb">
      <img src="photo/img11.jpg" alt=" " class="thumb">
      <img src="photo/img12.jpg" alt=" " class="thumb">
      <p><a href="index.html">Повернутись на головну сторінку сайту</a></p>
    </div>
  </body>
</html>
```

Також треба створити файл стилів **style.css**, в якому записати таке:

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

При наведенні вказівника миші на будь-яку фотографію вона трохи (на 20%) збільшиться в розмірах, а решта фотографій, навпаки, зменшиться на 10%.

Для такого ефекту використовується псевдоклас :hover для класу thumbs і функція scale() для зменшення усіх зображень на 10% (аргумент 0.9). Для подальшого збільшення вибраної фотографії треба задіяти для неї :hover з аргументом 1.2. Правило !important застосовується для підвищення пріоритету стильового правила, оскільки зменшення зазвичай має вищий пріоритет.[19]

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

1.6.7. Розміщення медіаконтенту на веб-сторінці

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

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

Формати відеофайлів

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

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

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

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

Формати аудіофайлів

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

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

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

Технології та засоби відтворення мультимедіа

Для використання звукових та мультимедійних можливостей комп’ютера треба потурбуватися про те, щоб він був обладнаний звуковою картою та колонками. Щодо самого процесу відтворення, то він має низку особливостей. Файли більшості форматів починають відтворюватись лише після завершення їхньо-

го завантаження. Є й інший спосіб передавання та відтворення файлів мультимедіа – у режимі реального часу. Таку технологію називають **потоковою**. Інформацію можна отримувати безпосередньо від джерела даних, зокрема з відеокамери або файлу на сервері. Дані відтворюються в міру їхнього надходження, копія на твердому диску комп'ютера користувача не створюється. Передавання потокових мультимедійних даних схоже на трансляцію телевізійних та радіопередач: користувач може приймати одну передачу, потім переключитися на інший канал або взагалі припинити прийом.

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

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

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

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

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

Наразі поширеними є такі **потокові технології**:

– *RealAudio/Video* – це технологія потокового передавання аудіо- та відеоінформації, розроблена компанією Progressive Networks. Для відтворення даних потрібен додатковий програмний модуль RealPlayer. Файли, призначені для оброблення засобами *RealAudio/Video*, мають розширення *.ra, .ram, .rm, .rmm, .rmd*;

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

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

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

Додавання аудіо

HTML5 пропонує швидкий і простий спосіб додавати аудіофайли на сайт через елемент <audio>. Як і тег <img>, для елемента <audio> треба задати URL

джерела, вказаного в атрибуті src. При цьому, на відміну від теґу <img>, теґ <audio> потребує відкривний і закривний теґи.

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

Атрибути <audio>

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Додавання відео

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

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

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

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

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

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

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

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

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

Додаткові варіанти використання мультимедіа на веб-сторінках

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

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

Атрибут loop може набувати значень:

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

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

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

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

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

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

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

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

4) Розглянемо приклад розміщення звукового файлу audio.wav за допомогою тегу **<embed>**, який дає змогу розмістити на веб-сторінці спеціальну панель

програвача мультимедійних файлів. Для цього використовують парний тег <embed src=. . .> </embed>, наприклад:

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

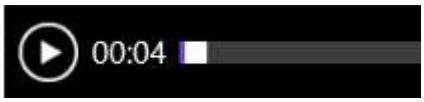
Файл audio.wav у цьому прикладі має бути збережений у поточній папці (тій самій, що й HTML-документ).

Тег <embed> може мати такі атрибути:

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

Приклади використання тегу <embed>:

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



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

Наприклад:

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

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

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

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

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

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

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

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

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

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

Додавання вбудованих фреймів

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

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

Елемент <iframe> усталено містить декілька стилів, зокрема: border, width і height. Ці стилі можна регулювати за допомогою HTML-атрибутів frameborder, width і height або за допомогою CSS-властивостей border, width і height.

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

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

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

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

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

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

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

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

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

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

Не всякий тип медіаконтенту має бути розміщений в елементі <figure> або мати <figcaption>, а лише той, який є незалежним і становить одну групу.

Як приклад створимо веб-сторінку з піснями та відеовізитівкою про Одесу. Для цього у папці поряд з іншими HTML-файлами свого сайту треба створити папку *audio*, в яку записати заздалегідь завантажені файли з піснями та відео.

Текст HTML-коду:

```
<html>
  <head>
      <meta charset="utf-8">
      <title>Медіаконтент про Одесу</title>
      <meta name="keywords"
      content="пісні про Одесу, відео про Одесу">
      <meta name="description"
      content="пісні та відео про Одесу">
  </head>
  <body background="photo/fon.gif">
    <div class="thumbs">
      <h1 aligment="center">Пісні про Одесу</h1>
        <figure>
            <audio src="audio/Odessa moy gorod rodnoy.mp3" controls></audio>
            <figcaption>Одесса – мой город родной</figcaption>
             <audio src="audio/у Черного моря.mp3" controls></audio>
            <figcaption>У черного моря</figcaption>
            <audio src="audio/ax, Одесса жемчужина у моря.mp3" controls></audio>
            <figcaption>Ах, Одесса жемчужина у моря</figcaption>
       </figure>
      <br>
      <h1 aligment="center">Відео про Одесу</h1>
```

```
      <object data="audio/aerovizitkaOdessa.mp4" type="video/mp4" standby="Йде заванта-
       ження. Зачекайте.">Візитівка Одеси </object>
      <figcaption>Візитівка Одеси</figcaption>
      <p><a href="index.html">Повернутись на головну сторінку сайту</a></p>
    </div>
  </body>
</html>
```

1.6.8. Створення форм на веб-сторінці

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

Засоби створення форми на сторінці

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

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

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

Теґи створення текстових полів і текстових областей на формі

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

1) Текстове поле <input>

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

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

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

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

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

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

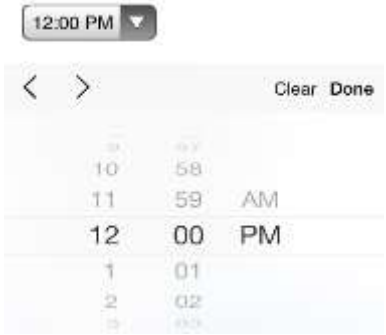
Наступні приклади <input> показують використання деяких з цих значень атрибута type з HTML5, а на рисунках показано, як ці унікальні значення можуть виглядати в iOS. Зверніть увагу, що різні значення забезпечують різні елементи керування, всі вони роблять простішим процес збору даних від користувачів.

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

Розглянемо на прикладі атрибута type для iOS7 різні значення елемента <input>

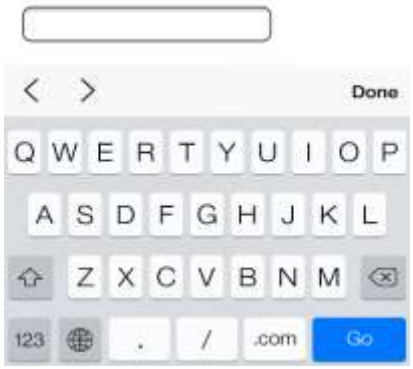
– значення date:

– значення time:



– значення email:

– значення url:



– значення number:

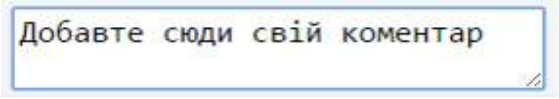
– значення tel:

2) Тектсовий елемент <textarea>

Ще одним елементом, використовуваним для збору текстових даних, є тег <textarea>. Він відрізняється від <input> тим, що може збирати великі уривки тексту

в декілька рядків. Елемент <textarea> також містить початковий і кінцевий теґи, які можуть обгортати простий текст. Оскільки <textarea> бере тільки один тип значення, атрибута type тут немає, але атрибут name, як і раніше, використовується.

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



Поля множинного вибору

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

1) Перемикачі

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

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

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

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

2) Прапорці

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

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

3) Випадні списки

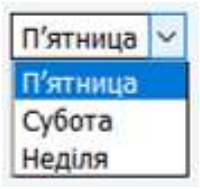
Випадні списки надають користувачам можливість вибору потрібного значення зі списку можливих варіантів. Для створення списку застосовують елементи <select> та <option>. Елемент <select> обгортає всі пункти меню, а кожен пункт меню розмічений за допомогою елемента <option>.

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

Кожен елемент <option> обгортає текст (який бачить користувач) окремого пункту у списку.

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

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

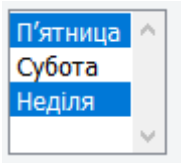


4) Список для множинного вибору

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

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

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



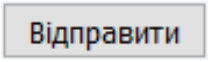
Кнопки на формі для відправлення даних

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

1) Елемент <input>

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

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



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

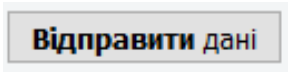
2) Елемент <button>

Елемент <button> виконує те саме, що й елемент <input> зі значенням submit в атрибуті type. Однак він містить відкривний і закривний теґи, які можуть обгорта-

ти інші елементи. Усталено елемент <button> діє зі значенням submit для атрибута type, а тому атрибут type разом зі значенням можна не вказувати.

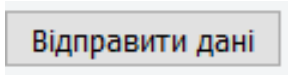
Текст на кнопці записується між відкривним і закривним теґами <button>:

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



або

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



Інші поля форми

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

1) Приховане поле

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

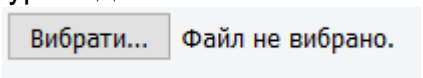
Щоб створити приховане поле, слід задати в атрибуті type значення hidden. Додатково приховане поле має відповідні значення атрибутів name і value:

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

2) Поле для завантаження файлу

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

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



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

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

Організація елементів форми

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

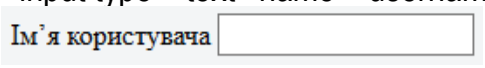
Елементи <label>, <fieldset> та <legend> допомогають організовувати форми і скеровувати користувачів правильно їх заповнювати.

1) Елемент <label>

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

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

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



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

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

2) Елемент <fieldset>

<fieldset> групує поля форми в організовані розділи подібно <section> або іншим структурним елементам. Проте <fieldset> є блоковим елементом, який обгортає пов'язані елементи, зокрема у <form>, для їхньої кращої організації. Елемент <fieldset> усталено має межі контуру, які можуть бути змінені засобами CSS.

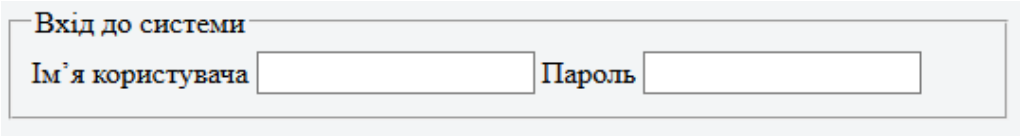
```
<fieldset>
  <label>
    Ім'я користувача
    <input type="text" name="username">
  </label>
  <label>    Пароль
    <input type="text" name="password">
  </label>
</fieldset>
```

3) Елемент <legend>

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

```
<fieldset>
  <legend>Вхід до системи</legend>
  <label>
```

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



Деякі поширені атрибути налаштування елементів на формі

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

1) **Атрибут disabled**

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

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

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

2) **Атрибут placeholder**

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

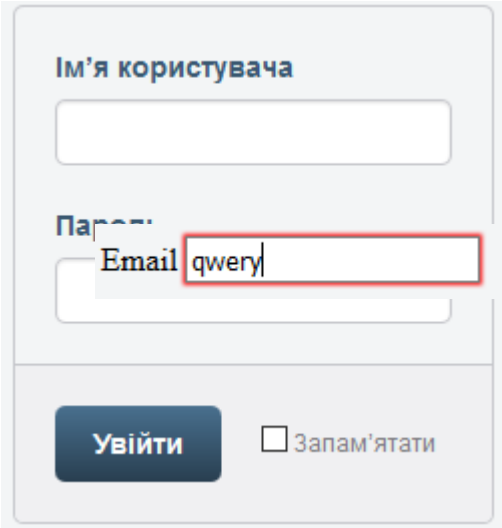
```
<label>
 Email
 <input type="email" name="email-address" placeholder="name@domain.com">
</label>
```

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

3) **Атрибут required**

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

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



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

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

Додаткові атрибути

Також елементи форм можуть використовувати додаткові атрибути:

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

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

Приклад 1.1.

HTML:

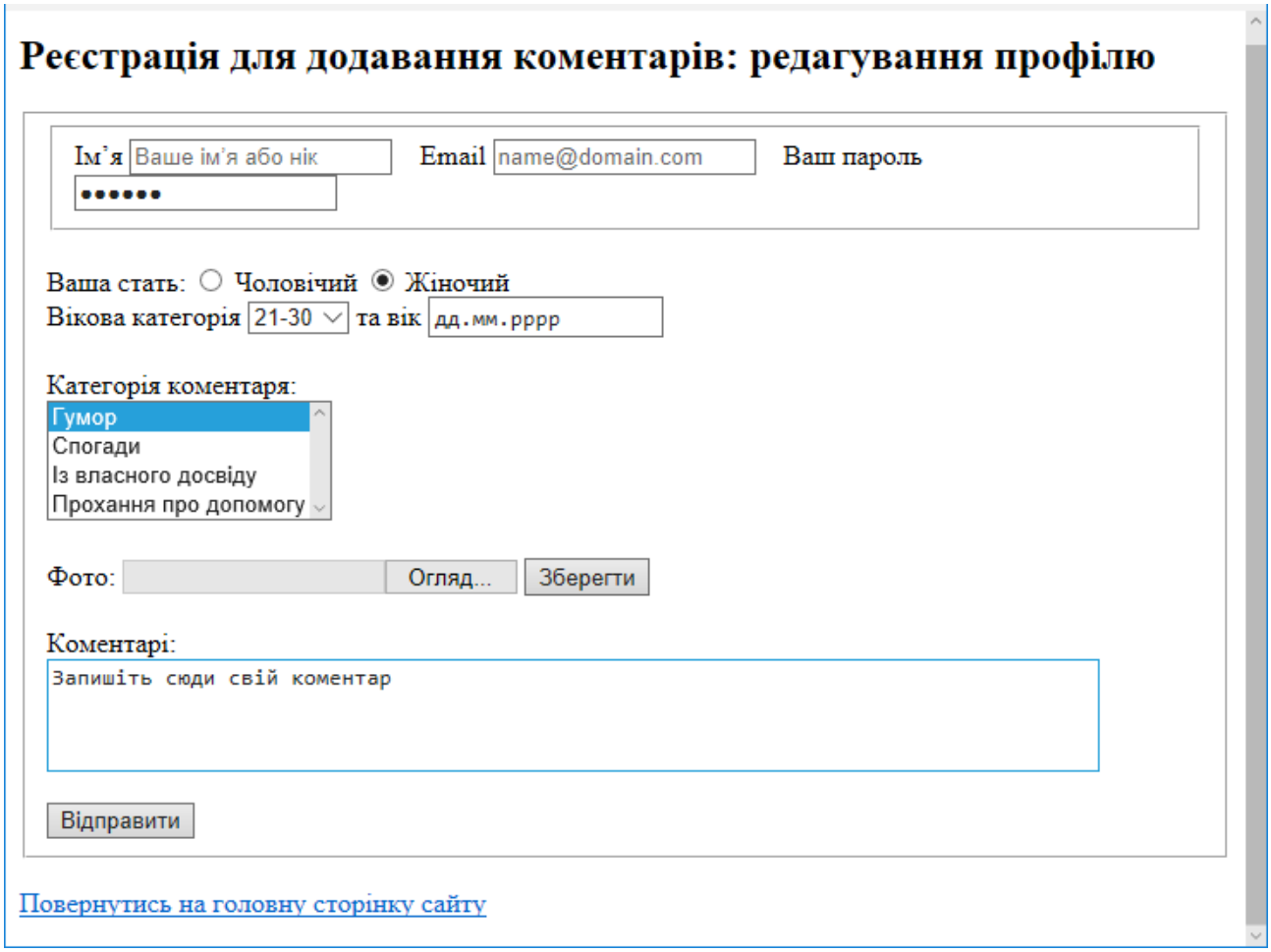
```
<form>
  <fieldset class="account-info">
    <label>
      Ім'я користувача
      <input type="text" name="username">
    </label>
    <label>     Пароль
      <input type="password" name="password">
    </label>
  </fieldset>
  <fieldset class="account-action">
    <input class="btn" type="submit"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

CSS:

```
*,
*:before,
*:after {  box-sizing: border-box;}
form {
  border: 1px solid #c6c7cc;
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  overflow: hidden;
  width: 240px;
}
fieldset {  border: 0;  margin: 0;  padding: 0; }
input {
  border-radius: 5px;
  font: 14px/1.4 "Helvetica Neue", Helvetica, Arial, sans-serif;
  margin: 0;
}
.account-info {  padding: 20px 20px 0 20px;}
.account-info label {
  color: #395870;
  display: block;
  font-weight: bold;
  margin-bottom: 20px;
}
.account-info input {
  background: #fff;
  border: 1px solid #c6c7cc;
  box-shadow: inset 0 1px 1px rgba(0, 0, 0, .1);
  color: #636466;
  padding: 6px;  margin-top: 6px;
  width: 100%;
}
.account-action {
  background: #f0f0f2;
  border-top: 1px solid #c6c7cc;
  padding: 20px;
}
.account-action .btn {
  background: linear-gradient(#49708f, #293f50);
  border: 0;  color: #fff;
  cursor: pointer;
  font-weight: bold;
  float: left;
  padding: 8px 16px;
}
```

```
.account-action label {
  color: #7c7c80;
  font-size: 12px;
  float: left;
  margin: 10px 0 0 20px;
}
```

Приклад 1.2.



HTML:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Профіль</title>
    <style>
       textarea {
            width: 90%;    height: 60px /* Ширина поля у відсотках  і висота у пікселях */
        }
    </style>
  </head>
```

```
<body>
 <h1>Реєстрація для додавання коментарів: редагування профілю</h1>
 <form action="#" method="post">
  <fieldset>
    <fieldset>
     <label>  Ім’я
        <input type="text" name="name" placeholder="Ваше ім’я або нік" required>
     </label>
     <label>     Email
       <input type="email" name="email" placeholder="name@domain.com" required>
     </label>
    <label for="password-field">   Ваш пароль</label>
      <input type="password" name="password" id="password-field" value="123456">
    </fieldset>
    <br>      Ваша стать:
    <label>
      <input type="radio" name="question-one" value="m">        Чоловічий
    </label>
    <label>
      <input type="radio" name="question-one" value="w" checked>      Жіночий
    </label>
    <br>  Вікова категорія
   <select name="age">
      <option value="до 15">до 15</option>
      <option value="16-20">16-20</option>
      <option value="21-30" selected>21-30</option>
      <option value="31-40">31-40</option>
      <option value="41-50">41-50</option>
      <option value="51-60">51-60</option>
      <option value="61-70">61-70</option>
    </select>
    та вік      <input type="date" name="birthday">
    <br><br>     Категорія коментаря:   <br>
    <select name="tehn" multiple size="4">
      <option selected> Гумор</option>
      <option > Спогади</option>
      <option> Із власного досвіду</option>
      <option> Прохання про допомогу</option>
   </select>
   <br><br>   Фото:         <input type="file" value="Choose File">
   <input type="submit" value="Зберегти">        <br> <br>
  <label>  Коментарі:<br>
    <textarea name="comment">Запишіть сюди свій коментар   </textarea>
  </label>         <br><br>
  <input type="submit" name="submit" value="Відправити">
```

```
    </fieldset>
  </form>
  <p><a href="index.html">Повернутись на головну сторінку сайту</a></p>
 </body>
</html>
```

1.7. Основні теґи мови HTML

Розглянемо в алфавітному порядку основні теґи HTML (навіть ті, з якими вже познайомились раніше).

1) Теґ <a>

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

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

2) Теґ <b>

Парний теґ <b> (подібно теґу) задає тексту напівжирне накреслення: **текст**. Цей теґ є теґом фізичної розмітки.

3) Теґ <basefont>

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

4) Теґи <big> та <blockquote>

Теґ <big> дозволяє збільшити шрифт тексту порівняно із сусіднім текстом, а за допомогою теґу <blockquote> можна створити в документі блок цитати.

**5) Теґ
**

Теґ
 створює новий рядок, тобто переносить текст на наступний рядок без формування нового абзацу. Цей теґ має один атрибут – clear, призначений для запобігання переносу слів тексту.

6) Теґ <body>

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

- alink, link і vlink – дозволяють задати колір гіперпосилань;
- background – задає файл графічного фону;
- bgcolor – змінює колір фону веб-сторінки;
- bgproperties – задає режим переміщення графічного фону при прокручуванні веб-сторінки;

– bottommargin – дозволяє редагувати висоту нижнього поля;
– leftmargin – дозволяє редагувати ширину лівого поля;
– marginheight – дозволяє редагувати висоту верхнього і нижнього полів;
– marginwidth – дозволяє редагувати ширину лівого і правого полів;
– rightmargin – дозволяє редагувати ширину правого поля;
– text – змінення кольору тексту;
– topmargin – дозволяє редагувати висоту верхнього поля.

7) Теґ <caption>

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

**8) Теґ **

Теґ призначений для центрування елементів веб-сторінки

9) Теґ <col>

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

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

10) Теґ <colgroup>

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

11) Теґ <dd>

Теґ <dd> дозволяє визначити елемент списку.

**12) Теґ **

Блоковий теґ формує на веб-сторінці розділи з різними стилями.

13) Теґ <font>

Теґ <font> змінює параметри шрифту тексту за допомогою таких атрибутів:
– color – змінює колір тексту;
– face – змінює шрифт тексту;
– size – змінює розмір шрифту.

14) Теґ <form>

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

– accept-charset – визначає тип кодування тексту даних;
– action – задає URL-адресу, куди пересилати дані форми;

– enctype – задає формат кодування даних форми до відправлення;
– method – вибирає спосіб (get або post) пересилання даних форми у мережі.

15) Теґ <frame>

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

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

HTML 5 не підтримує теґ <frame>.

16) Теґ <frameset>

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

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

HTML 5 не підтримує теґ <frameset>.

17) Теґи заголовків

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

18) Теґ <head>

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

19) Теґ <hr>

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

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

20) Теґ <html>

Структурний теґ <html> призначений для ідентифікації веб-сторінки. Програмний код будь-якої веб-сторінки має починатися теґом <html> і завершуватись теґом </html>.

21) Теґ <i>

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

22) Теґ <iframe>

Теґ <iframe> дозволяє створити вбудований фрейм для відтворення різних незалежних сторінок-документів. Підтримується HTML5. Цей теґ є блоковим елементом. Вигляд блока визначається стилем, який можна винести у зовнішній css-файл, а для теґу визначити атрибути class або id з ім'ям селектора.

23) Теґ <img>

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

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

**24) Теґ **

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

25) Теґ <link>

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

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

26) Теґ <marquee>

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

– behavior – тип руху рядка[20];
– bgcolor – колір фону рухомого рядка;
– direction – напрям руху рядка;
– height, width – висота і ширина рухомого рядка;
– hspace, vspace – поля навколо рядка;
– loop – кількість циклів переміщення рухомого рядка;
– scrollamount – крок одночасного переміщення руху рухомого рядка;
– scrolldelay – затримка між окремими тактами переміщення тексту рядка;
– truespeed – мінімальне значення інтервалу затримки scrolldelay.

27) Тег <meta>

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

28) Тег <nobr>

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

29) Тег <noframes>

Тег <noframes> відображатиме альтернативний текст у разі неможливості показу фрейму.

30) Тег <noscript>

За допомогою тегу <noscript> відображатиметься альтернативний текст у разі неможливості виконання сценарію JavaScript.

**31) Тег **

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

32) Тег <p>

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

33) Тег <script>

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

34) Тег <small>

Тег <small> зменшує розмір шрифту на одну умовну одиницю[21] порівняно з основним текстом.

35) Тег <strike>

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

**36) Тег **

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

[20] Атрибут behavior. URL: http://htmlbook.ru/html/marquee/behavior (дата звернення 28.09.2018).
[21] В HTML розмір шрифту вимірюється в умовних одиницях від 1 до 7. Середній розмір тексту, використовуваного усталено, – 3.

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

37) Теґ <style>

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

38) Теґ <sub>

Теґ <sub> перетворює текст на нижній індекс.

39) Теґ <sup>

Теґ <sup> перетворює текст на верхній індекс.

40) Теґ <table>

Теґ <table> призначений для створення таблиць. Він має такі атрибути:

- align – задає обтікання таблиці текстом;
- background – визначає графічний фон таблиці;
- bgcolor – задає колір фону таблиці;
- border – дозволяє задати потрібну товщину ліній рамки;
- bordercolor – задає колір ліній рамки;
- cellpadding – задає відступи між вмістом клітинки та її межами;
- cellspacing – дозволяє задати відступи між клітинками таблиці;
- frame – дозволяє задати набір ліній рамки таблиці;
- height, width – задають висоту і ширину таблиці;
- rules – задає набір внутрішніх ліній таблиці.

41) Теґ <tbody>

За допомогою теґу <tbody> ідентифікується група рядків таблиці. Теґ може використовуватись з такими атрибутами.

- align – спосіб вирівнювання даних по горизонталі;
- bgcolor – колір фону;
- valign – спосіб вирівнювання даних по вертикалі.

42) Теґи <td> і <th>

Теґи <td> і <th> дозволяють створити клітинку даних і клітинку заголовка таблиці відповідно. Вони можуть використовуватись з атрибутами:

- align – спосіб вирівнювання даних по горизонталі;
- background – графічний фон клітинок;
- bgcolor – колір фону клітинок;
- colspan – об’єднання сусідніх клітинок рядка таблиці;
- height – висота клітинки;
- width – ширина клітинки;
- nowrap – заборона переносу слів всередині клітинки на новий рядок;
- rowspan – об’єднання сусідніх клітинок одного стовпця;
- valign – спосіб вирівнювання даних по вертикалі.

43) Теґ <tfoot>

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

44) Теґ <title>

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

45) Теґ <tr>

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

46) Теґ <tt>

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

47) Теґ <u>

Теґ <u> задає підкреслений текст: текст.

**48) Теґ **

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

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

Контрольні запитання

1. Що таке веб-сайт?
2. Чим по суті є веб-сторінка? Як її створити та зберегти?
3. Що таке HTML?
4. Які категорії HTML-редакторів існують?
5. Що таке WYSIWYG в HTML-редакторі?
6. Назвати приклади сучасних онлайн HTML-редакторів.
7. Якою є структура HTML-документа?
8. Що таке теґ? Які теґи призначені для задавання структури документа?
9. Яка структура та формат запису парних теґів HTML? Навести приклади.
10. Що таке параметр теґу? Який формат запису параметра теґу HTML? Навести приклади.
11. Яке призначення CER як елемента HTML?
12. Перелічити теґи для подання текстової інформації і надати їхній опис.
13. Як подаються гіперпосилання в HTML-документі? Навести приклад теґів створення внутрішніх і зовнішніх посилань.

14. У чому схожість та відмінність призначення теґів <p> та
?
    а) вони тотожні;
    б) <p> формує новий абзац тексту з параметрами відступів між абзацами, а
 формує новий рядок тексту (без відступів);
    в)
 формує новий абзац з параметрами відступів, а <p> формує новий рядок (без відступів);
    г)
 є закривним теґом до <p>.
15. Яке призначення метатеґів? Навести приклади метатеґів.
16. Яке призначення теґу <style>?
    а) записується у розділі head і задає внутрішні стилі;
    б) визначає одну чи декілька назв класів для HTML-елемента;
    в) для вставлення зображень;
    г) визначає, чи може вміст HTML-елемента бути змінюваними, чи ні.
17. Як можна задати вирівнювання абзацу у веб-документі?
18. Які варіанти вирівнювання тексту абзацу можливі?
19. Яке вирівнювання тексту абзацу задається усталено?
    а) за шириною;
    б) по центру;
    в) за лівим краєм;
    г) немає вирівнювання.
20. Чим відрізняються засоби фізичних стилів форматування тексту від логічних стилів?
21. Чим схожі і відрізняються дії теґів <b> та , <i> та ?
22. Яке призначення теґу <mark>?
    а) щоб оцінити роботу розробника сторінки;
    б) для формування гіперпосилань;
    в) для вставлення приміток;
    г) коли треба виділити маркером жовтого кольору текст.
23. Коли виникає потреба у преформатованому тексті, який формується теґом <pre>?
    а) коли треба скинути будь-яке форматування тексту;
    б) при формуванні гіперпосилань;
    в) для вставлення зображень;
    г) коли треба зберегти авторське форматування тексту, взятого з якогось іншого документа.
24. Яке призначення атрибута id?
    а) для формування гіперпосилань;
    б) визначає ідентифікатор для HTML-елемента;
    в) визначає, чи може вміст елемента бути пересунутим;
    г) визначає, чи може вміст HTML-елемента бути змінюваними, чи ні.
25. Охарактеризувати теґи, які використовуються для позначення заголовків у HTML?

Розділ 2
Каскадні таблиці стилів CSS

2.1. Призначення і рівні CSS

2.1.1. Поняття CSS

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

CSS (від англ. Cascading Style Sheets) – спеціальна мова для стильового форматування вмісту веб-сторінок сайту. Можна сказати, що CSS – це спосіб додаткового форматування стандартних теґів HTML.

Термін «каскадні таблиці стилів» і концепцію використання стилів запропонував Хокон Віум Лі[1] 1994 року.

Розберемо складові слова поняття «каскадна таблиця стилів»:

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

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

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

Основні переваги CSS:

– керування дизайном будь-якої кількості HTML-документів з однієї таблиці стилів;

[1] Хокон Віум Лі (род. 27 липня 1965, Норвегія) – вчений, фахівець у галузі інформатики, який 2004 року запропонував каскадні таблиці стилів (CSS). 2006 року в Університеті Осло захистив дисертацію на тему «Каскадні таблиці стилів». З 1999 року працює головним інженером норвезької компанії Opera Software, яка займається розробкою браузера Opera.

– більш точний дизайн сторінок в усіх браузерах;
– поділ документа на дві складові: структуру (вміст веб-сторінки) і дизайн, завдяки чому вихідний код стає чистим і легко читається;
– розширені можливості порівняно зі звичайним HTML.

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

2.1.2. Стандарти (рівні) CSS

У 1990-х роках різні підходи щодо розроблення веб-сторінок, а відтак і плутанина, зумовили потребу стандартизувати Web, створити загальні правила, за якими програмісти і веб-дизайнери проектуватимуть сайти. Так з'явились мови HTML 4.01 і XHTML та стандарт CSS. Крім того, самі браузери тоді містили власні стилі, а відтак одна й та сама сторінка по-різному могла виводитись на екран у різних браузерах.

На відміну від багатьох наявних на той момент мов стилю, CSS використовує успадкування від батька до нащадка, тому розробник може визначити різні стилі, грунтуючись на вже визначених раніше стилях.

У середині 1990-х Консорціум Всесвітньої павутини (W3C[2]) став проявляти інтерес до CSS, і в грудні 1996 року була видана рекомендація CSS1.

На сьогодні існують три рекомендовані W3C стандарти (рівні) CSS, які визначаються наявністю завершеної редакції структури. Це:

– CSS1 – перший рівень структури стильових шаблонів був опублікований W3C 1996 року[3] і відкоригований 1999[4] і 2008[5] роках;

– CSS2 – другий рівень стильових конструкцій. Був рекомендований W3C 1998 року, переглянутий 2008 року. 2011 року вийшла версія CSS 2.1[6] і 2016 року – версія CSS 2.2[7]). Нині робота над CSS2 ще триває;

– CSS3 – третій рівень стильового оформлення веб-сторінок почав розроблятися 2002 року[8], остання рекомендація – 21 червня 2018 року.[9]

Починаючи з 2011 року ведуться чорнові розробки четвертого рівня.[10,11]

Розглянемо коротко специфіку кожного зі стандартів.

---

[2] W3C (англ. World Wide Web Consortium, W3C) – консорціум, який розробляє і впроваджує технологічні стандарти для Всесвітньої павутини. Засновником і главою консорціуму є сер Тімоті Джон Бернерс-Лі, автор безлічі розробок в галузі інформаційних технологій.

[3] Cascading Style Sheets, level 1.. URL: https://www.w3.org/TR/CSS1 (дата звернення 18.07.2018).

[4] Cascading Style Sheets, level 1. URL: https://www.w3.org/TR/1999/REC-CSS1-19990111 (дата звернення 18.07.2018).

[5] Cascading Style Sheets, level 1. URL: https://www.w3.org/TR/2008/REC-CSS1-20080411 (дата звернення 18.07.2018).

[6] Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification. URL: https://www.w3.org/TR/2011/REC-CSS2-20110607 (дата звернення 18.07.2018).

[7] Cascading Style Sheets Level 2 Revision 2 (CSS 2.2) Specification. URL: https://www.w3.org/TR/CSS22 (дата звернення 18.07.2018).

[8] CSS3 module: Basic User Interface. URL: https://www.w3.org/TR/2002/WD-css3-ui-20020802 (дата звернення 18.07.2018).

[9] CSS Basic User Interface Module Level 3 (CSS3 UI). URL: https://www.w3.org/TR/2018/REC-css-ui-3-20180621 (дата звернення 18.07.2018).

[10] Selectors Level 4. URL: https://www.w3.org/TR/selectors-4 (дата звернення 18.07.2018).

[11] CSS Cascading and Inheritance Level 4. URL: https://drafts.csswg.org/css-cascade (дата звернення 18.07.2018).

Можливості, рекомендовані **CSS1**:

– задавання параметрів шрифтів: гарнітура, розмір шрифту, стиль (звичайний, курсивний або напівжирний);
– задавання кольору тексту, фону, рамок та інших елементів сторінки;
– атрибути тексту (міжсимвольний інтервал, відстань між словами, висота рядка, тобто відступи між рядками);
– вирівнювання для тексту, зображень, таблиць та інших елементів;
– властивості блоків (висота, ширина, внутрішні (padding) і зовнішні (margin) відступи і рамки);
– обмежені засоби позиціонування елементів float і clear.

Кожний наступний стандарт виправляє помилки, може видалити (скасувати) певні частини специфікації і вносить певні новації.

Додаткові функціональності **CSS2**:

– блокова верстка: відносне, абсолютне і фіксоване позиціонування елементів на сторінці без табличної верстки;
– можливість задавати різні стилі для різних типів носіїв (монітор, принтер, телефон);
– звукові таблиці стилів визначають голос, гучність для звукових носіїв;
– розширений механізм селекторів;
– покажчики тощо.

CSS3 – наймасштабніша редакція, порівняно з CSS1 та CSS2. Головною особливістю CSS3 є можливість створювати анімовані елементи без використання JS, підтримка лінійних і радіальних градієнтів, тіней, згладжування і багато іншого. Переважно використовується як засіб опису та оформлення зовнішнього вигляду веб-сторінок, написаних засобами мов розмітки HTML і XHTML. Саме третій рівень CSS3 позиціонується розробниками як єдина система подання стилів веб-сторінки, заснована на використанні спеціальних модулів.

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

Правило складається із селектора і блока оголошення стилів, укладеного у фігурні дужки:

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

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

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

Тут задано червоний колір шрифту розміром 14 пікселів для заголовка h1.

2.2. Способи визначення таблиць стилів

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

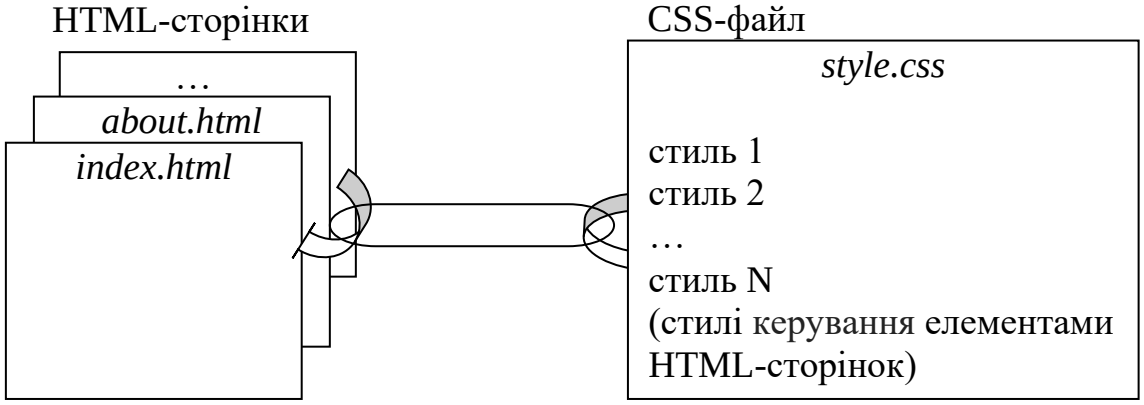
Визначити таблиці стилів (стильового шаблона) можна чотирма способами.

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

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

Тип посилання на таблицю стилів — *Тип даних файлу* — *Місце розташування файлу стилів*

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



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

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

Наприклад:

```
<html>
  <head>
    <title>Підключення CSS до HTML</title>
    <meta charset="utf-8">
```

```
    <style type="text/css">
      h1 { color: red;  }
    </style>
  </head>
  <body>
     <h1>Заголовок червоного кольору 1</h1>
     <h1>Заголовок червоного кольору 2</h1>
     <h1>Заголовок червоного кольору 3</h1>
  </body>
</html>
```

Тепер усі заголовки h1 на сторінці будуть червоного кольору.

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

3) **Вбудовані стилі** (входять у теґові конструкції, inline-стиль): будь-який окремий HTML-елемент можна відформатувати засобами CSS. Для цього треба задати певне правило реалізації того чи іншого теґу, наприклад:

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

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

Також можна створити стиль для одного конкретного заголовка h1:

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

Окремому HTML-елементу можна задати певний клас стильового шаблона:

```
<table>
   <tr>
      <td class="header"> № з/п </ td> <td class="header"> Студент</ td>
   </tr>
   <tr>
      <td class="text"> 1 </ td> <td class="text"> Антоненко Олександр</ td>
   </tr>
   <tr>
      <td class="text"> 2 </ td> <td class="text"> Гулько Наталія</ td>
   </tr>
</table>
```

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

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

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

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

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

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

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

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

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

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

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

Щоб уникнути суперечливих ситуацій, пов'язаних зі спробою одному й тому самому елементу задати суперечливий стиль, треба розставити пріоритет (вагу). Порядок зростання пріоритетів такий:

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

Отже, inline-стиль має найвищий пріоритет.

Розглянемо на прикладі поєднання різних способів визначення стильового шаблона CSS:

```
<html>
  <head>
```

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

2.3. Запис шаблона CSS

Розглянемо способи визначення стильових шаблонів CSS.

2.3.1. Групування та успадкування

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

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

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

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

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

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

2.3.2. Види селекторів

Селекторами CSS можуть виступати:

1) Селектори теґів HTML використовуються для визначення стилів конкретних елементів. Селектором виступає сам HTML-теґ, наприклад:

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

Тут увесь текст у межах розділу body буде помаранчевим, а для заголовка, крім розміру шрифту, задано синій колір для шрифту і зелений – для фону. Нагадаємо, колір можна задавати константними назвами (red, blue, green, black тощо) або числовими значеннями палітри у формі від #000 (чорний) до #FFF (білий), або від #000000 (чорний) до #FFFFFF (білий). Числові значення дозволяють гнучко задавати відтінки тої чи іншої складової кольору: перша цифра у тризнаковому форматі (дві цифри у шестизнаковому) задає червону складову кольору від мінімального 0 (00) до максимального F[12] (FF[13]) значення, друга – зелену складову, а третя – синю. До речі, потужним безкоштовним сервісом для добору кольорової палітри є https://color.adobe.com/create/color-wheel від компанії Adobe.

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

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

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

Для застосування цього класу в HTML-файлі треба прописати його у будь-якому теґу:

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

або

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

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

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

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

При зазначенні класів стильового шаблона треба уважно стежити за тим, чи підтримує HTML-теґ тип, зазначений у визначенні. Наприклад, запис:

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

---

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

[13] FF – значення 255 у шістнадцятковому форматі.

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

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

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

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

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

Ще один приклад ID-селектора:

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

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

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

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

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

Підіб'ємо підсумки:

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

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

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

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

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

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

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

2.3.3. Псевдокласи

Псевдоклас в CSS – це ключове слово, додане до селектора, яке визначає його особливий стан. Наприклад, :hover може бути використано для змінення кольору кнопки при наведенні вказівника миші на неї.

```
div:hover { background-color: #F89B4D; }
```

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

Розглянемо функціональність псевдокласів на прикладі гіперпосилань. Згідно зі специфікаціями HTML і CSS гіперпосилання може перебувати у чотирьох станах: невідвідуване посилання (link), відвідуване посилання (visited), активне посилання (active) і посилання при наведенні вказівника миші (hover). Перші три стани (link, visited, active) зазвичай прописуються у теґу <body> HTML-документа (рівень CSS1). Четвертий стан (hover) стосується рівня CSS2 і змінює колір посилання при наведенні на нього вказівника миші.

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

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

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

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

Отже, псевдокласи надають можливість стилізувати елемент, залежно від зовнішніх факторів, як от історія відвідувань (наприклад, :visited), стан вмісту (на кшталт :checked у деяких елементів форми) або позиції вказівника миші (наприклад, :hover визначає, чи перебуває вказівник миші над елементом).

2.4. Специфіка використання CSS

Передусім, варто зазначити, що при визначенні стильових таблиць далеко не завжди властивості стандартного HTML-теґу відповідають опису шаблона сти-

лю. Наприклад, в HTML для жирного накреслення використовується теґ-контейнер <b> (або), а в CSS – конструкція font-weight: bold. Для виділення тексту підкресленням в HTML передбачений теґ <u>, а в CSS використовується запис text-decoration: underline тощо.

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

2.4.1. CSS для форматування тексту

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

У CSS розрізняють абсолютні і відносні одиниці виміру властивостей елементів (табл. 2.1).

Таблиця 2.1

Одиниці виміру CSS

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

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

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

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

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

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

«%» (percents) – схожий на «em», за винятком кількох принципових відмінностей. По-перше, поточний font-size дорівнює 100% (тобто 12pt = 100%).

По-друге, при використанні «%» текст стає повністю масштабованим для мобільних пристроїв і зручності користувача (accessibility).

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

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

Таблиця 2.2

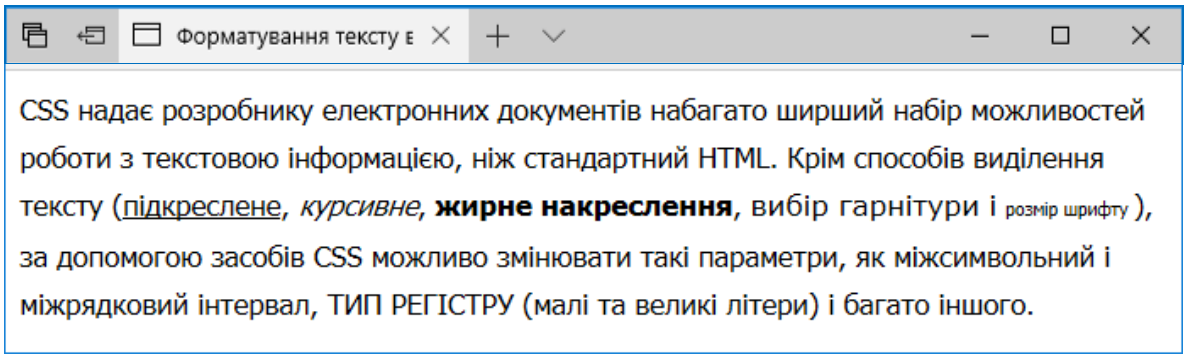
Властивості форматування тексту в CSS

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

Розглянемо на прикладі застосування цих властивостей:

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Форматування тексту в CSS</title>
    <style type="text/css">
      .text { font-family: Tahoma; color: 1003366;  /* визначення класу text */
              line-height: 7mm; font-size: 12pt;
             }
      #kursiv { font-style: italic; }                    /* визначення ID-селектора kursiv */
      span.font { font-size: 10px; }
      .color { background-color: #cecece; }
    </style>
  </head>
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
    <font class="text"> CSS надає розробнику електронних документів набагато ширший
        набір можливостей роботи з текстовою інформацією, ніж стандартний HTML.
        Крім способів виділення тексту (<font style = "text-decoration:
        underline;">підкреслене</font>, <font id = "kursiv"> курсивне</font>,
```

```
        <font style = "font-weight: bold;"> жирне накреслення</font>,
        <font style = "font-family: verdana;"> вибір гарнітури </font> і <span class = "font">
        розмір шрифту </span>), за допомогою засобів CSS можливо <font class =
        "color"> змінювати такі параметри</font>, як <font style = "letter-spacing: 3px;">
        міжсимвольний </font> і міжрядковий інтервал, <font style = "text-transform:
        uppercase;"> тип регістру </font> (малі та великі літери) і багато іншого.
    </font>
  </body>
</html>
```



2.4.2. Структурне форматування

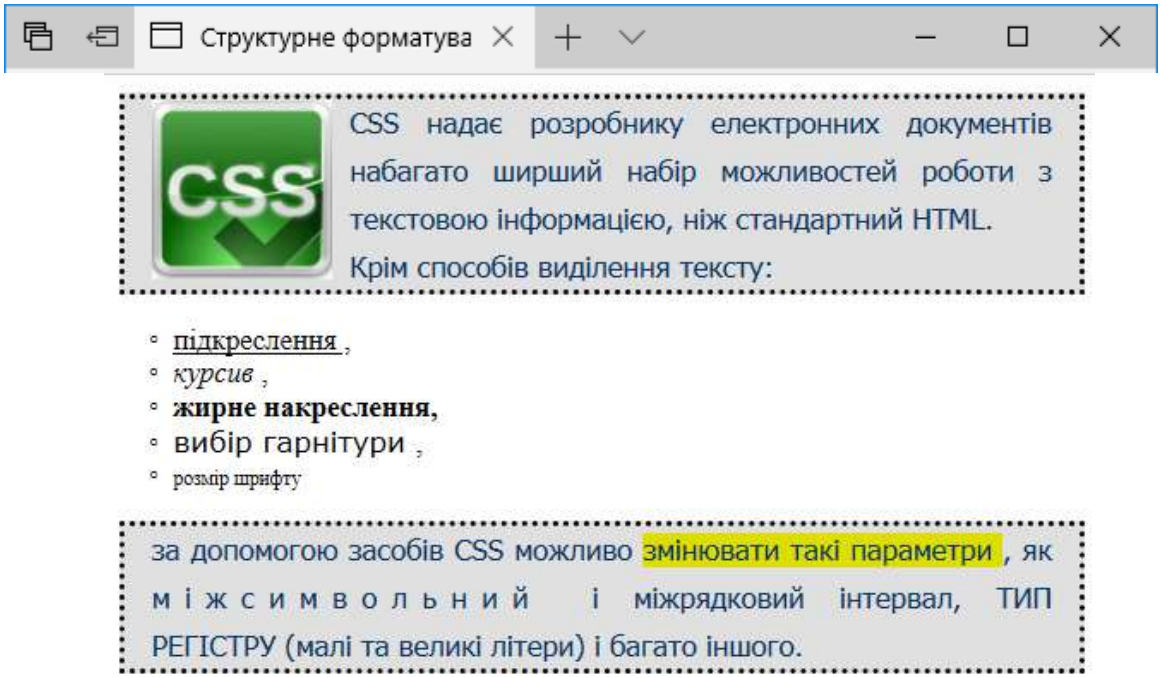
Засоби CSS помітно розширили функціональність форматування структурних елементів веб-сторінки: p, div та ін. У табл. 2.3 наведено найпоширеніші властивості структурного форматування в CSS.

Таблиця 2.3

Властивості структурного форматування в CSS

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

Розглянемо на прикладі властивості структурного форматування в CSS:




```
<html>
  <head>
    <meta charset="utf-8">
    <title>Структурне форматування в CSS</title>
    <style type="text/css">
     .text { font-family: Tahoma; font-size: 12pt;
             color: #003366; line-height: 7mm;
             margin: 10px; padding-left: 15px; padding-right: 15px;
             border-color: black; border-style: dotted;
             background-color: #e0e0e0;   width: 500px; text-align: justify;
           }
     #kursiv { font-style: italic; }
     span.font { font-size: 11px; }
     .color { background-color: #dede00; }
     img { float: left; width: 100px; height: 100px;  margin-right: 10px;}
     li { list-style-type: circle; }
   </style>
  </head>
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
    <p class="text">  <img src="picture.jpg"> CSS надає розробнику електронних документів
       набагато ширший набір можливостей роботи з текстовою інформацією, ніж стандарт-
       ний HTML.<br>Крім способів виділення тексту:
    </p>
    <ul>
       <li><font style="text-decoration: underline;">підкреслення </font>,
       <li><font id="kursiv"> курсив </font>,
       <li><font style="font-weight: bold;"> жирне накреслення, </font>
```

```
      <li><font style="font-family: verdana;"> вибір гарнітури </font>,
      <li><span class="font"> розмір шрифту </span>
    </ul>
    <p class="text"> за допомогою засобів CSS можливо <font class="color"> змінювати
      такі параметри </font>, як <font style="letter-spacing: 8;"> міжсимвольний </font> і
      міжрядковий інтервал, <font style="text-transform: uppercase;"> тип регістру </font>
      (малі та великі літери) і багато іншого. </p>
  </body>
</html>
```

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

2.5. Засоби позиціонування об’єктів

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

2.5.1. Позиціонування засобами float

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

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

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

Найчастіше при позиціонуванні для float задають значення left і right:

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

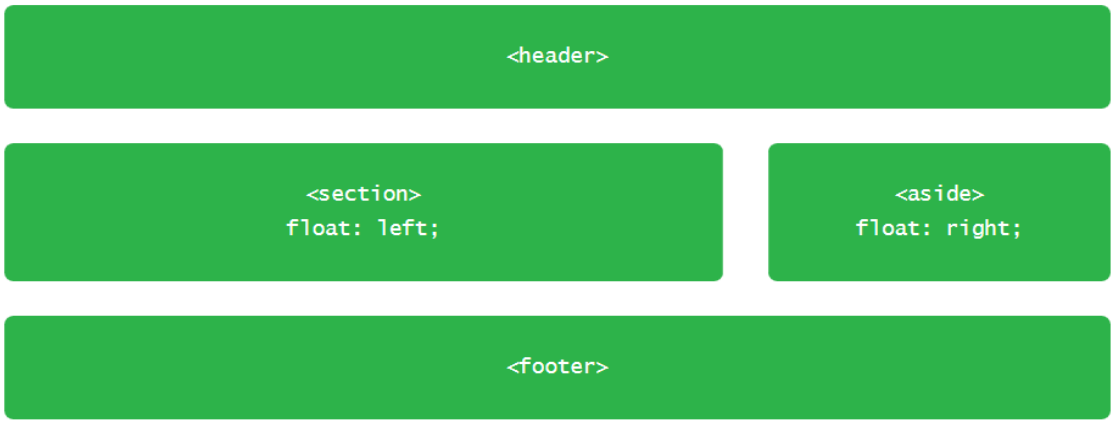
Одночасне застосування float до декількох елементів дає змогу створити макет з елементами, розміщеними поряд або один напроти одного.

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

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

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

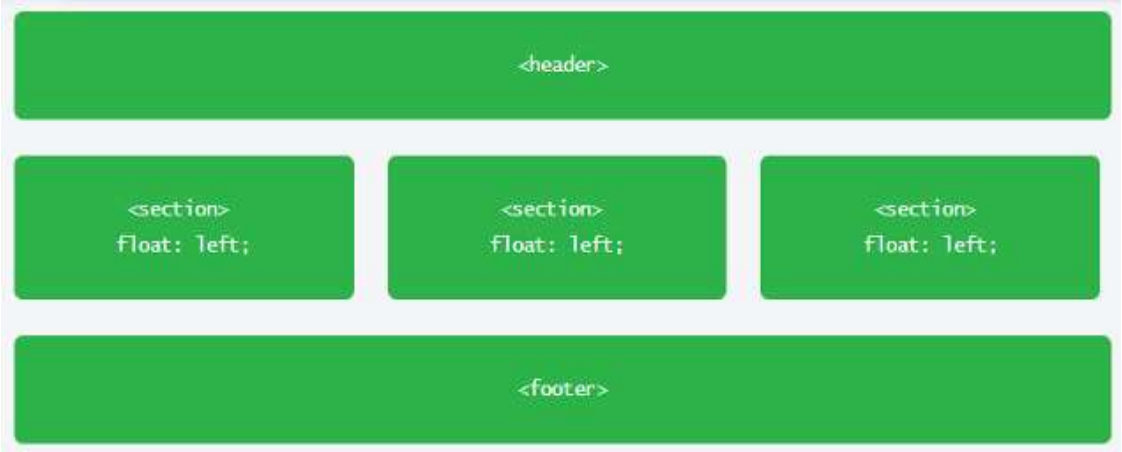
```
section { float: left;   margin: 0 1.5%;   width: 63%; }
aside { float: right;  margin: 0 1.5%;  width: 30%;  }
```



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

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

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



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

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

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

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

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

Докладіше про це можна прочитати, скориставшись посиланням https://learn.shayhowe.com/html-css/positioning-content/.

2.5.2. Позиціонування засобами position

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

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

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

Є два типи візуального позиціонування елементів: абсолютне і відносне.

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

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

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

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

HTML-код:

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

CSS-правила:

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

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

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

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

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

HTML-код:

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

CSS-правила:

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

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

Ці особливості можна використати для створення тіні до заголовка:

HTML-код:

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

CSS-правила:

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

2.5.3. Позиціонування через inline-block

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

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

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

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

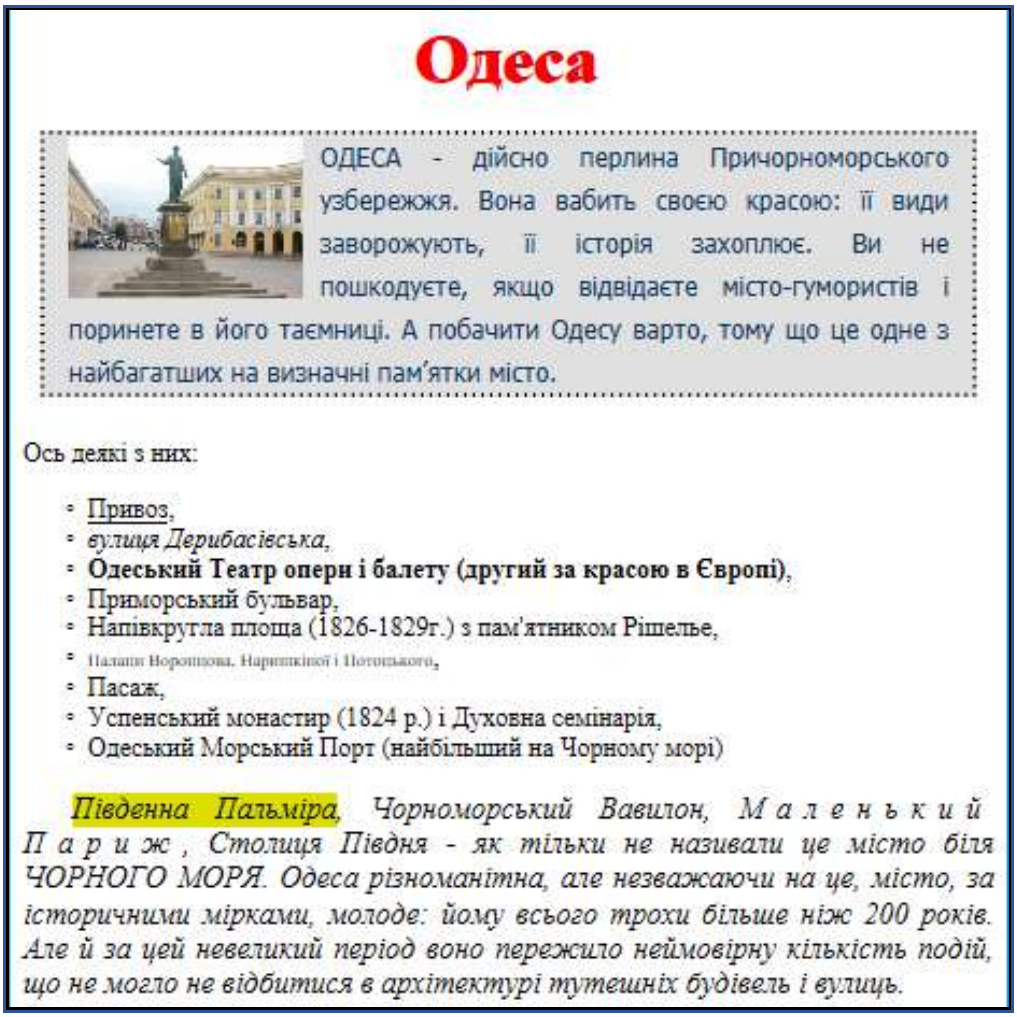
У CSS-правилах треба властивість float для трьох елементів <section> замінити на display: inline-block, залишаючи властивості margin і width тими, що були раніше. Як результат, наш CSS буде виглядати так:

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

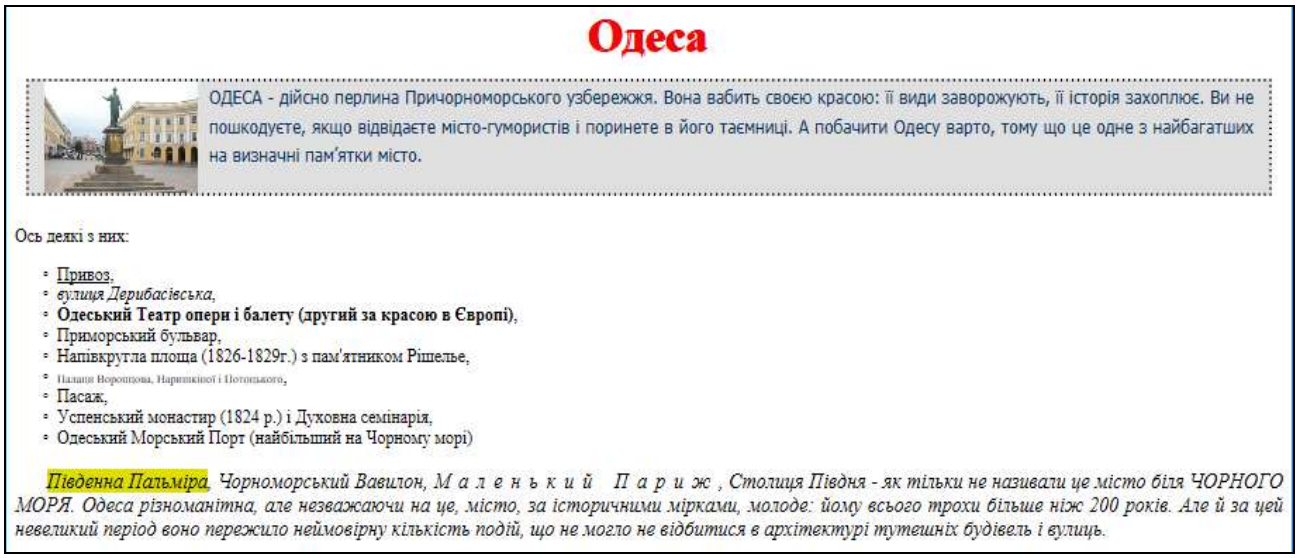
На жаль, при цьому останній елемент <section> виштовхнеться на новий рядок, оскільки загальна ширина стане занадто великою (розмір кожного окремого простору додається до ширини зі значенням горизонтального margin усіх елементів у рядку). Щоб відобразити всі елементи <section> в одному рядку, треба видалити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

Розглянемо приклади форматування вмісту веб-сторінки засобами CSS.

Приклад 2.1. Створимо веб-сторінку такого вигляду:



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




```
<html>
  <head>
```

```
    <meta charset="utf-8">
    <title>Структурне форматування в CSS</title>
    <style type="text/css">
     .text-block { display: inline-block;
                   font-family: Tahoma;
                   font-size: 12pt;
                   margin: 10px;
                   color: #003366;
                   line-height: 7mm;
                   padding-left: 15px;
                   padding-right: 15px;
                   border-color: black;
                   border-style: dotted;
                   background-color: #e0e0e0;
                   text-align: justify;
                   }
     .text {
                   font-family: "Times New Roman";
                   text-indent: 30;                    /*Відступ першого рядка абзацу*/
                   font-size: 14pt;
                   font-style: italic;
                   text-align: justify;
                }
     #kursiv { font-style: italic; }
     span.font { font-size: 11px; }
     .color { background-color: #dede00; }
     .img-head { float: left;   height: 100;  margin-right: 10px; }
     li { list-style-type: circle; }
     .offset   {  right: 2px;   position: relative;
                  color: red;   font-style: bold;
                  top: -48;
                }
     h1,.offset { font-size: 32pt;  text-align: center;   }
     header { height: 5%; }
    </style>
  </head>
  <body bgcolor="#ffffff" text="black" link="#00ff00" alink="#00ff00" vlink="blue">
   <header>
    <h1><font color="dark-grey"> Одеса</font><!-- --><div class="offset">Одеса</div></h1>
   </header>
    <p class="text-block"> <img src=«Odessa.jpg» class="img-head"> ОДЕСА – дійсно пер-
       лина Причорноморського узбережжя. Вона вабить своєю красою: її види заворожу-
       ють, її історія захоплює. Ви не пошкодуєте, якщо відвідаєте місто-гумористів і пори-
```

```
        нете в його таємниці. А побачити Одесу варто, тому що це одне з найбагатших на
        визначні пам'ятки місто.</p>
    <p>Ось деякі з них:</p>
    <ul>
      <li><font style="text-decoration: underline; ">Привоз</font>,</li>
      <li><font id="kursiv">вулиця Дерибасівська</font>,</li>
      <li><font style="font-weight: bold;">Одеський Театр опери і балету (другий за красою в
        Європі)  </font>,</li>
      <li>Приморський бульвар,</li>
      <li>Напівкругла площа (1826-1829г.) з пам'ятником Рішелье,</li>
      <li><span class="font">Палаци Воронцова, Наришкіної і Потоцького</span>,</li>
      <li><font style="font-family: verdana; ">Пасаж</font>,</li>
      <li>Успенський монастир (1824 р.) і Духовна семінарія,</li>
      <li>Одеський Морський Порт (найбільший на Чорному морі).</li>
    </ul>
    <p class="text"> <font class="color">Південна Пальміра</font>, Чорноморський Вавилон,
        <font style="letter-spacing: 8;">Маленький Париж</font>, Столиця Півдня – як тільки
        не називали це місто біля <font style=" text-transform: uppercase;">Чорного мо-
        ря</font>. Одеса різноманітна, але незважаючи на це, місто, за історичними мірка-
        ми, молоде: йому всього трохи більше, ніж 200 років. Але й за цей невеликий період
        воно пережило неймовірну кількість подій, що не могло не відбитися в архітектурі
        тутешніх будівель і вулиць.
    </p>
  </body>
</html>
```

Приклад 2.2. Створимо на веб-сторінці чотири зображення та спозиціонуємо їх у такий спосіб:

Додамо в HTML-код такий фрагмент у розділ body:

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

А до CSS-правил додамо:

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

Приклад 2.3. Створимо на веб-сторінці гіперпосилання на відкриття у новому вікні іншої веб-сторінки. Доповнимо файл style.css описом псевдокласів гіперпосилань, які при наведенні вказівника миші на це гіперпосилання змінюватимуть його колір, розмір тексту, колір фону тощо.

Додамо в HTML-код такий фрагмент у розділ body:

```
<p><a href="lb2.html" target="_blank">Перейти на веб-сторінку №2 </a></p>
```

А до файлу CSS-правил додамо опис псевдокласів гіперпосилань:

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

Контрольні запитання

1) Як розшифровується і що означає CSS?
2) Які переваги використання CSS?
3) Які особливості CSS3?
4) Що таке стиль?
   а) метод перетворення текстових документів на HTML;
   б) набір правил форматування елементів веб-сторінки;
   в) спосіб зменшення HTML-коду веб-сторінки;
   г) мова розмітки гіпертекстових документів.
5) Що таке правило стилю? Описати різні компоненти і синтаксис.
6) Яке призначення селектора?
7) Припустимо, що є набір правил стилю. Що треба зробити, щоб перетворити їх на зовнішню таблицю стилів?
8) У чому перевага об’єднання селекторів у групу?
9) Який атрибут використовується для визначення внутрішнього стилю?
   а) class; б) style; в) font; г) link.
10) Який недолік використання внутрішнього стилю?
11) Який недолік використання вбудованих стилів (inline)?

12) Який варіант задавання кольору НЕ спрацює?
а) color: #000; б) color: #aaa; в) color: #hhh; г) color: #aaaaaa.
13) Якщо межі сторінки не примикають до країв вікна браузера, куди треба додати властивість margin: 0?
а) doctype; б) html; в) head; г) body.
14) Який CSS-код написано правильно?
а) {border: 1px solid #ccc;} ;
б) div {border: 1px solid #ccc;} ;
в) {border: 1px solid #hhh;} ;
г) div {border: 1px solid #hhh;}.
15) Коли імена ідентифікаторів і класів можна називати однаково?
а) ніколи;
б) якщо зустрічаються у коді лише раз;
в) завжди;
г) якщо використовуються для різних елементів.
16) Який CSS-код написано правильно?
а) p { color: #ccc;} ;
б) p: {border: 1px solid #ccc;} ;
в) <p> { color: #ccc;} ;
г) p { color: cccc;}.
17) Як правильно вставляти коментарі в CSS-код?
а) /* Мій коментар */; в) # Мій коментар;
б) // Мій коментар; г) # Мій коментар #.
18) Який CSS-код треба задати, щоб колір відвіданих і невідвіданих посилань був одним і тим самим?
а) a:link, a:visited {color: yellow;} ;
б) a:active, a:visited {color: yellow;} ;
в) a:link {color: yellow;} ;
г) a:link, a:active {color: yellow;}.
19) Яку помилку припущено у такому описі шрифту:
font-family: Arial, Times New Roman, Helvetica, sans-serif;?
а) замість font-family треба використовувати властивість font;
б) треба поставити лапки для Times New Roman;
в) не можна вказувати більше 3-х різних шрифтів;
г) шрифту Helvetica не існує.
20) В якому з варіантів припущено явну помилку?
а) p span { font-size: 150%;} ; в) p {font-size: 150%;} ;
б) p span#text (font-size: 150%;} ; г) p text (font-size: 150%;}.
21) Яке призначення псевдокласів у CSS?
22) У чому специфіка використання одиниці виміру «em» у веб?
23) Чому використання «em» стає все популярнішим у веб-документах?
24) Коли доцільно задавати розмір об’єктів в одиницях «px»?
25) Чому дорівнює 1px? У чому проблема використання «px» у веб?

Розділ 3
Онлайн веб-конструктори сайтів

3.1. Доменні імена та IP-адреси веб-сайтів

Кожному комп'ютеру при підключенні до інтернету надається власний унікальний номер, так звана IP-адреса. У кожного веб-ресурсу також є своя IP-адреса.

На сьогодні **IP-адреса** – це група з чотирьох чисел від 0 до 255, яка записується у вигляді 123.45.67.89. Так, IP-адресою офіційного сайту провайдера ООО «Укрнет» є 212.42.76.253, а компанії «Google Україна» – 172.217.18.67.

Числову комбінацію 172.217.18.67 важко запам'ятати, однак, якщо її написати в адресному рядку, то браузер відкриє саме сайт www.google.com.ua, адже доменне ім'я цього сайту www.google.com.ua відповідає IP-адресі 172.217.18.67.

Наявність доменного імені, замість числового еквівалента, дозволяє звертатися до комп'ютера за ім'ям, що ідентифікує власника IP-адреси.

Доменне ім'я (англ. domain name) – унікальний ідентифікатор у вигляді поєднання символів латинського алфавіту, який надається певній IP-адресі для ідентифікації сайту серед безлічі інших (двох однакових імен бути не може). Крім літер латиниці, у доменному імені можуть використовуватись цифри від 1 до 9 та символ дефісу «-», проте дефіс не може бути першим та/чи останнім символом в імені. Довжина імені домену може бути від 2 до 63 символів, інколи до 127 символів. Доменне ім'я виконує функцію унікального імені в інтернеті і по суті є простішим і зручнішим варіантом запису IP-адреси сайту.

Донедавна в іменах доменів використовувались лише літери латиниці, проте останнім часом все більше доменних зон підтримують кирилицю в імені домену, наприклад: коп.укр, вікна.укр, новий-місяць.укр, затишок.укр, азбукауспіху.укр тощо. Вони називаються IDN (Internationalized Domain Names)[1]. Вибираючи ім'я варто пам'ятати, що змішувати символи не можна, у назві домену можуть бути символи лише латиниці або лише кирилиці.

Цікаво, що перший зареєстрований у мережі домен – це Symbolics.com. Американська фірма Symbolics Inc. оформила його в 1985 році, він і зараз працює.

Для того щоб конкретній цифровій IP-адресі поставити у відповідність символьне доменне ім'я сайту, є спеціальні сервери DNS[2].

DNS-сервер – програма, яка здійснює перетворення доменного імені на цифрову IP-адресу та навпаки. У пам'яті цих серверів зберігаються величезні таблиці, в яких кожному доменному імені ставиться у відповідність IP-адреса.

Звертаємо увагу на те, що одній IP-адресі можуть відповідати декілька доменних імен (наприклад, при розміщенні на комп'ютері кількох WWW-серверів різних організацій). І навпаки, одній IP-адресі можуть відповідати

[1] Як правильно підібрати домен. URL: https://freehost.com.ua/ukr/domain/domainsel (дата звернення 27.09.2018).
[2] **DNS** (Domain Name Service) – служба доменних імен.

кілька доменних імен. Здебільшого це практикується на популярних інтернет-вузлах, що дозволяє за допомогою спеціальних рішень розподілити навантаження з одного комп'ютера на декілька або спрямувати користувача зі схожих за накресленням чи то «тлумаченням» доменних імен.

IP-адреса, яка відповідає даному доменному імені, може змінитися. Наприклад, організація переїжджає або змінює інтернет-провайдера. Збереження «за собою» доменного імені дозволяє не турбуватися, що за подібних обставин доведеться нести витрати на «розкрутку» нового імені.

Вся інформація про доменні імена зберігається в центральній базі даних DNS, яка розміщена на кількох потужних комп'ютерах, розкиданих по всьому світу. У цій базі зберігається інформація про дату реєстрації, фізичного або юридичного власника доменного імені, а також шлях до так званого сервера імен – NAMESERVER, де міститься інформація, на яку вказує доменне ім'я.

«Корпорація з керування доменними іменами та IP-адресами» («Internet Corporation for Assigned Names and Numbers», скорочено **ICANN**) – міжнародна некомерційна організація, створена 18 вересня 1998 року за участю уряду США для врегулювання питань, пов'язаних з доменними іменами, IP-адресами та іншими аспектами функціонування інтернету. ICANN займається акредитацією реєстраторів, а також наглядає за виконанням ними своїх функцій.

. (крапка) – домен нульового рівня. Зараз ICANN забезпечує підтримку та керування всім адресним простором DNS в інтернеті, крім доменів першого рівня обмеженого користування, які безпосередньо управляються американськими державними організаціями.

Єдиний інтернет-каталог, який визначає основу DNS, перебуває в державній організації SRI International-Menlo Park, CA, US (Менло-Парк, Каліфорнія, США).

Щоразу, коли користувач вводить у браузері доменне ім'я, служба DNS визначає, якій саме IP-адресі відповідає це ім'я і який саме ресурс треба надати. Тому доменне ім'я сайту часто називають **доменною адресою сайту** або **доменним ім'ям сервера**. По суті, це одне й те саме. Кожне доменне ім'я складається з декількох складових частин, розділених крапками, – це **домени різних рівнів**. Здебільшого кількість рівнів доменів обмежується двома-трьома. Довге доменне ім'я і багато рівнів домену незручні у використанні. Крайнє праве поле називають **доменом першого (верхнього) рівня**, лівіше слідують імена доменів нижчого рівня. Наприклад, в доменному імені inf.od.ua доменом першого рівня є ua, od – домен другого рівня, inf – домен третього рівня.

Отже, доменне ім'я або літерна адреса комп'ютера може бути:

– доменним ім'ям першого рівня (*first level domain*);

– доменним ім'ям другого рівня (*second level domain*);

– доменним ім'ям третього рівня (*third level domain*) тощо.

Домени першого рівня призначаються для кожної країни, а також для різних типів організацій. Імена цих доменів повинні відповідати міжнародному стандарту ISO 3166. Для позначення країн використовуються двобуквені абревіатури, наприклад: ua (Україна), us (США), it (Італія), fr (Франція).

На сайті https://www.nic.ru/cgi/na.cgi?step=n_a.all_world наведено понад 200 різних географічних доменних імен першого рівня.

Домени верхнього рівня ще називають доменними зонами. Доменне ім'я першого рівня має вигляд: www.ua (українська зона інтернету). Доменне ім'я другого рівня пишеться так – wikipedia.org. У домені wikipedia.org частина wikipedia є доменом другого рівня. Доменне ім'я третього рівня складається з домену другого рівня, до якого ліворуч додано піддомен, наприклад: cat.dog.ua.

Організаційні доменні імена першого рівня в США		**Географічні доменні імена першого рівня**	
biz	businesses firms (комерційні)	ua	Ukraine (Україна)
com	commercial (комерційні)	uk	United Kingdom (Великобританія)
edu	US educational (освіта)	de	Germany (Німеччина)
gov	US goverment (уряд)	fr	France (Франція)
int	international (міжнародні)	ru	Russia (Росія)
mil	US military (військові США)	tv	Tuvalu (Тувалу)
nato	NATO (НАТО)	cc	Cocos Islands (Кокосові острови)
org	non-profit organization (некомерційні організації)	us	Сполучені Штати Америки
net	network (мережеві послуги)	ws	Western Samoa (Західна Самоа)

Українські домени:

.ua – національний український домен першого (верхнього) рівня, призначений для організацій та зареєстрованих торгових марок. Реєструється тільки на основі свідоцтва на знак для товарів і послуг;

.gov.ua – урядові організації;

.com.ua – комерційний домен для підприємств, які працюють в Україні;

.net.ua – для мереж та організацій зв'язку на території України;

.org.ua – некомерційні організації на території України;

.biz.ua – для фізичних та юридичних осіб, які ведуть підприємницьку діяльність на території України;

.edu.ua – освітні організації на території України;

.in.ua – для приватних осіб;

.co.ua – призначений для будь-яких реєстрацій без обмежень.

Сайт або веб-сайт (від англ. website – місце, майданчик в інтернеті) – сукупність веб-сторінок, доступних в інтернет-мережі, які об'єднані як за змістом, так і навігаційно. Апаратні сервери для зберігання веб-сайтів називаються **веб-серверами**. Сама послуга зберігання називається **веб-хостингом**[3]. Фізично сайт може розміщуватися як на одному, так і на кількох серверах. Зараз сервери для зберігання тільки одного сайту називаються **виділеними** (англ. dedicated).

[3] **Хостинг** (від англ. hosting) – послуга надання ресурсів для розміщення інформації на сервер, що постійно перебуває у мережі (зазвичай інтернет).

Перший у світі сайт info.cern.ch з'явився 1990 року. Його створив Тім Бернерс-Лі[4] – батько сучасного інтернету. Автор опублікував на своєму сайті опис нової технології WWW (World Wide Web), основаної на протоколі передачі даних HTTP, системі адресації URI і мові розмітки HTML.

Сторінки сайтів – це набір текстових файлів, розмічених мовою HTML. Сторінки сайтів можуть бути простим статичним набором файлів (**статичні**) або створюватися спеціальною комп'ютерною програмою на сервері (**динамічні**). Зазвичай сторінки взаємопов'язані між собою. І відвідувач сайту сам вирішує, в якій послідовності їх переглядати. В інтернеті перегляд сторінок сайтів здійснюється через спеціальні програми – **браузери**.

Здебільшого в інтернеті одному веб-сайту відповідає одне доменне ім'я. Саме за доменними іменами сайти ідентифікуються в глобальній мережі. Проте можливі інші варіанти: один сайт на декількох доменах або кілька сайтів під одним доменом. Зазвичай кілька доменів використовують великі сайти (веб-портали), щоб логічно відокремити різні види послуг (mail.google.com, news.google.com, maps.google.com). Нерідкі випадки виділення окремих доменів для різних країн або мов. Наприклад, google.ru, google.de і google.com.ua логічно є сайтом Google на різних мовах, але технічно – це різні сайти.

3.2. Конструктори сайтів

Конструктори сайтів – це онлайн-сервіси, за допомогою яких будь-хто може створити сайт, не володіючи при цьому спеціальними знаннями. Створення дизайну, заповнення та редагування контенту відбувається безпосередньо в браузері комп'ютера за допомогою різноманітних шаблонів галереї і дизайнерських інструментів, а тексти, зображення та інші модулі можна пересувати по сайту за допомогою миші.

Зазвичай сайт-конструктор не вимагає скачування або встановлення спеціального програмного забезпечення, потрібні тільки комп'ютер, браузер та доступ до інтернету.

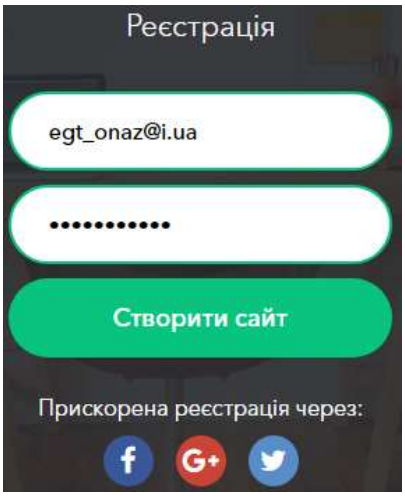
Більшість сучасних потужних сайт-конструкторів з якісними, сучасними дизайнерськими шаблонами є платними[5] (наприклад: Instapage (instapage.com), Wix (ru.wix.com)) проте абонентська плата є помірною і становить в еквіваленті від $5 до $25, залежно від функціоналу вибраного пакету. Деякі онлайн-конструктори сайтів є безкоштовними (наприклад: україномовний uCoz (ucoz.ua), Website Builder (websitebuilder.com), SiteBuilder (sitebuilder.com)), Sitey (sitey.com), і переважно працювати з ними можна відразу ж після невеликої реєстрації. Інші ж можуть надавати основну частину функціоналу безкоштовно, а низку додаткових послуг і можливостей для сайту – на платній основі, наприклад: Weebly (weebly.com), Jimdo (ru.jimdo.com), Sitelio (sitelio.com). Також є онлайн-

[4] Цікаво, що 2004 року королева Великої Британії Єлизавета II посвятила Тіма Бернерс-Лі у лицарі за його винахід мови HTML та мережі WWW.

[5] Станом на 2018 рік.

конструктори сайтів із безкоштовним функціоналом конструкторів лише на певний термін, наприклад: Shopify (shopify.com) надає безкоштовний функціонал на 14 днів, а надалі вимагає оплати для можливості подальшої роботи.

Як приклад розглянемо сайт-конструктор uCoz на сервері www.ucoz.ua. Для реєстрації в системі uCoz треба зайти на сторінку http://www.ucoz.ua/register і вибрати один із доступних способів реєстрації.



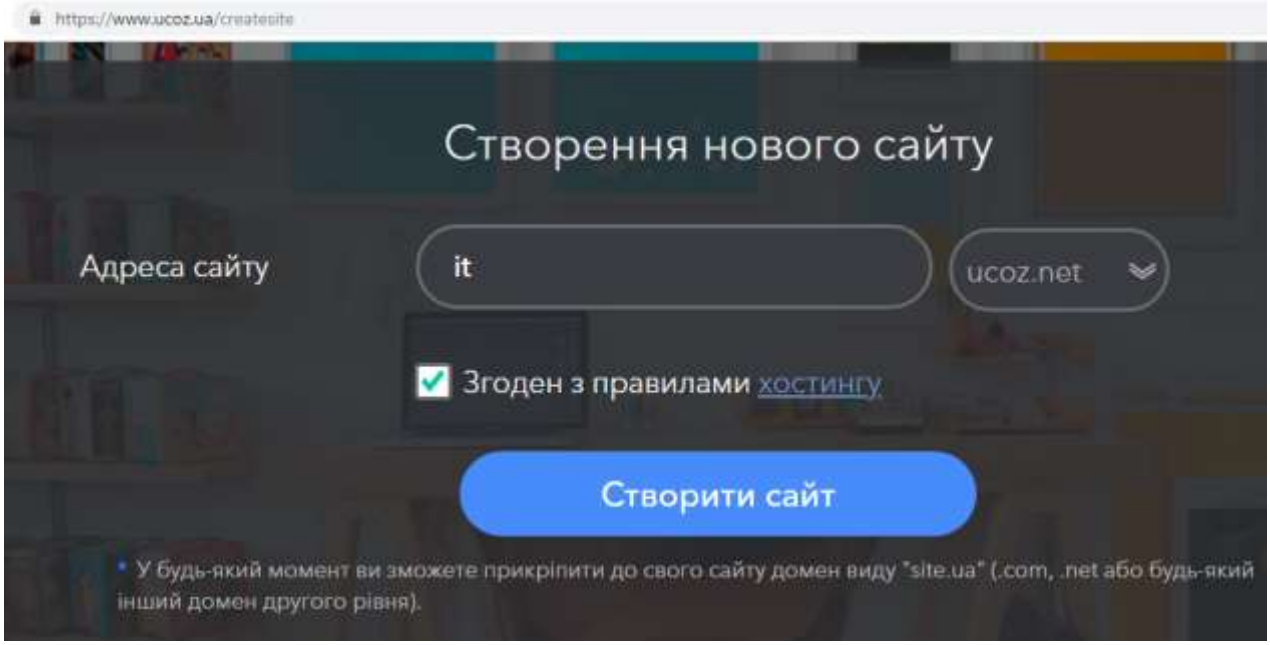
При першому знайомстві з uCoz, коли uID аккаунта ще немає, треба вибрати реєстрацію через email та задати пароль.

Крім того, користувачі соціальних мереж можуть скористатися прискореною реєстрацією, для цього треба натиснути на іконку відповідної соціальної мережі, в якій вони мають свій аккаунт, і слідувати вказівкам.

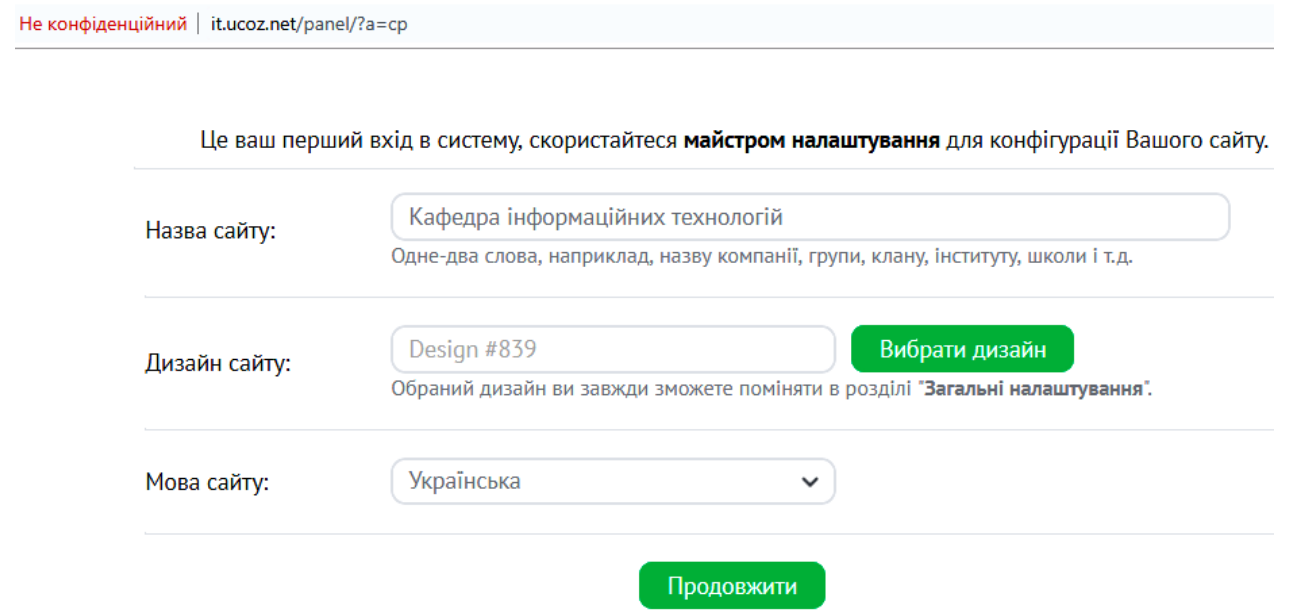
3.3. Засоби онлайн-конструктора сайтів uCoz

3.3.1. Створення дизайну та структури сайту

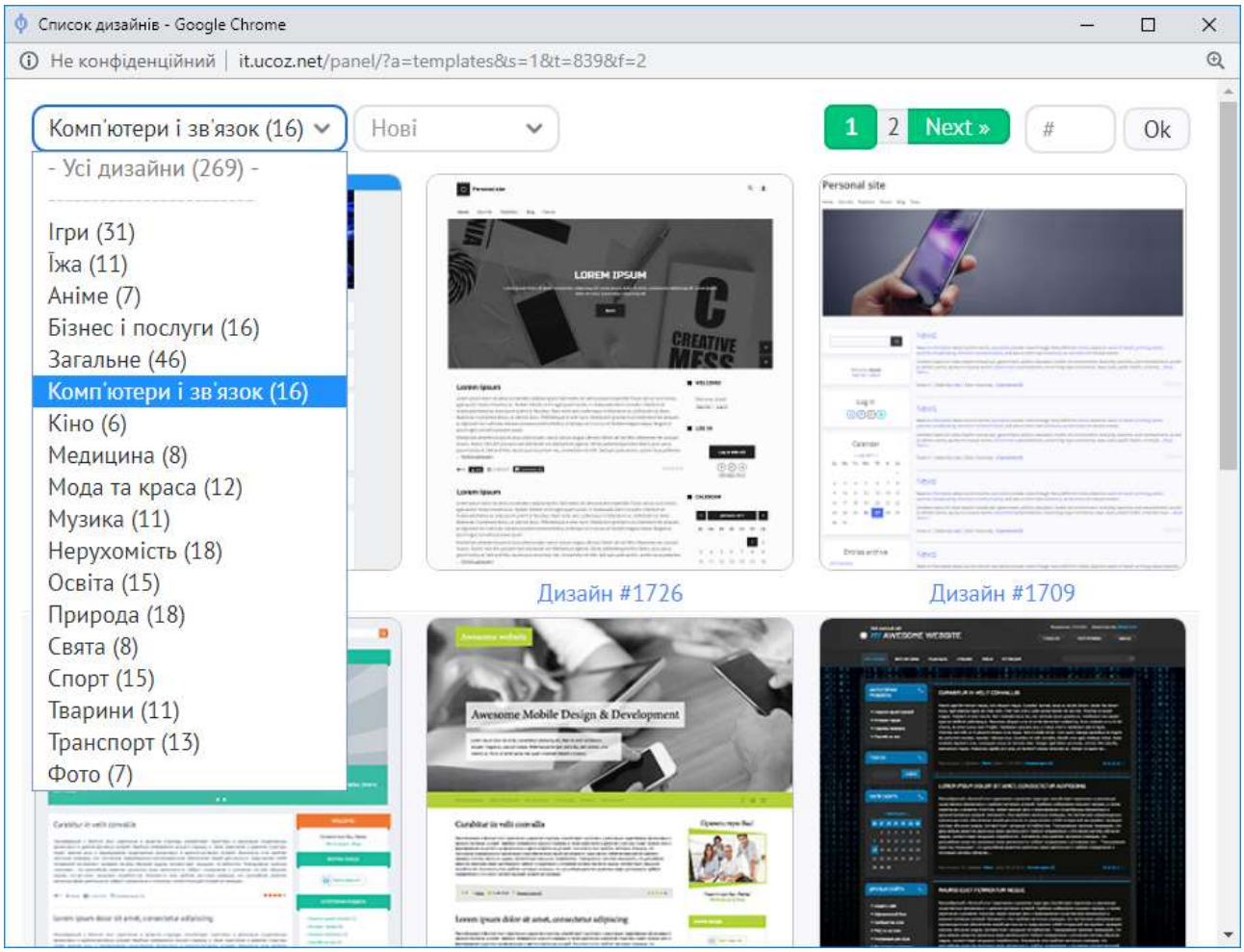
Після авторизації на uCoz можна приступити до створення сайту і вибрати для нього доменне ім’я. Саме за цим доменним ім’ям відвідувачі бачитимуть цей сайт в інтернеті. Для придуманого вільним чином імені сайту треба вибрати доменну зону (ucoz.com, ucoz.net чи ін.) і натиснути кнопку *Створити сайт.*



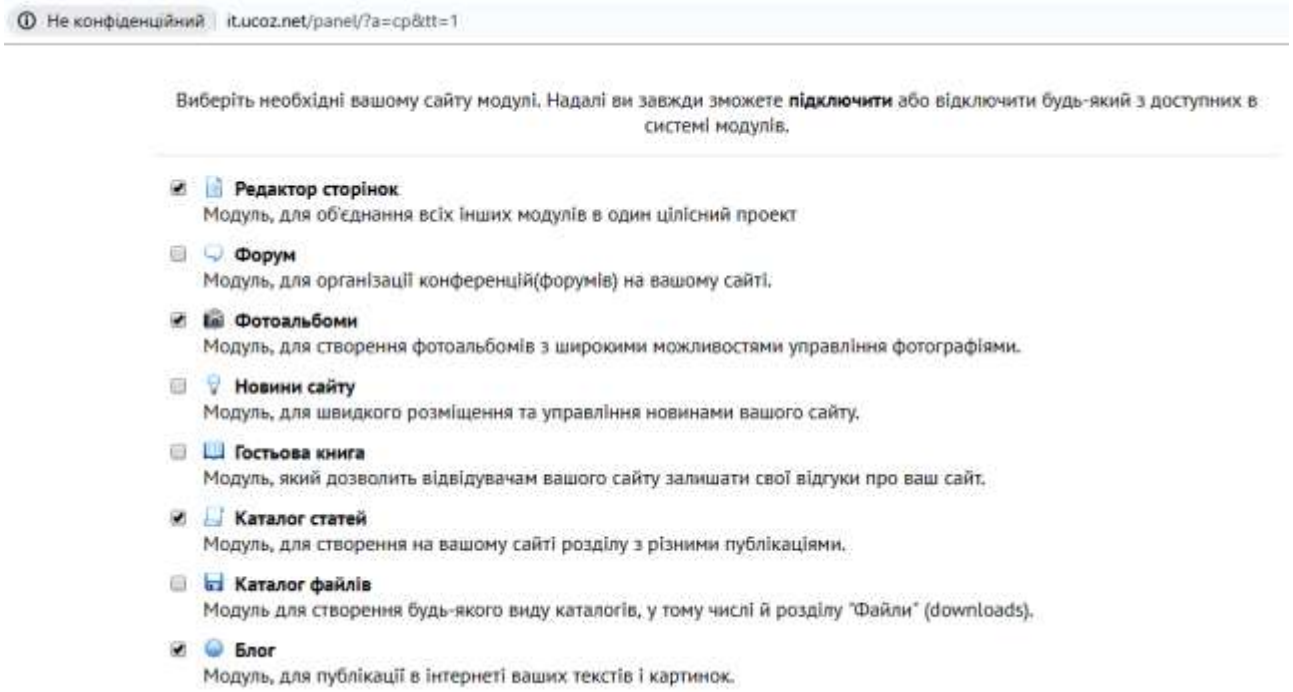
У наступному вікні треба ввести назву сайту, вибрати доречний дизайн відповідно до тематики майбутнього сайту та вибрати зі списку мову сайту.



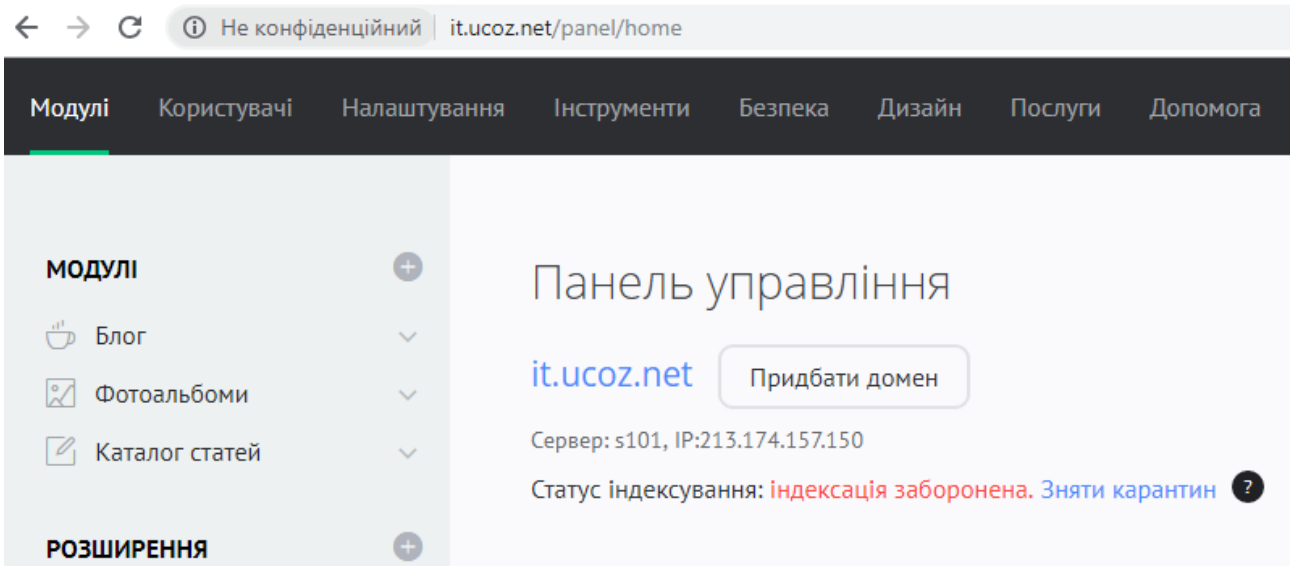
Дизайн на цьому етапі можна вибрати, скориставшись кнопкою *Вибрати дизайн*. При цьому з'явиться вікно зі списком дизайнів. Вибір уподобаного дизайну здійснюється простим клацанням по ньому. Після цього система повернеться до вищенаведеного вікна, де залишиться натиснути кнопку *Продовжити*.



Далі буде запропоновано вибрати модулі для створюваного сайту. Треба поставити галочки навпроти потрібних модулів. Згодом можна буде відключити чи підключити будь-який модуль системи uCoz.



Після натискання кнопки *Продовжити*, розміщеної наприкінці переліку модулів, відбудеться перехід до панелі управління сайтом. Саме тут можна налаштувати всі використовувані модулі та вибрати або змінити дизайн сайту.

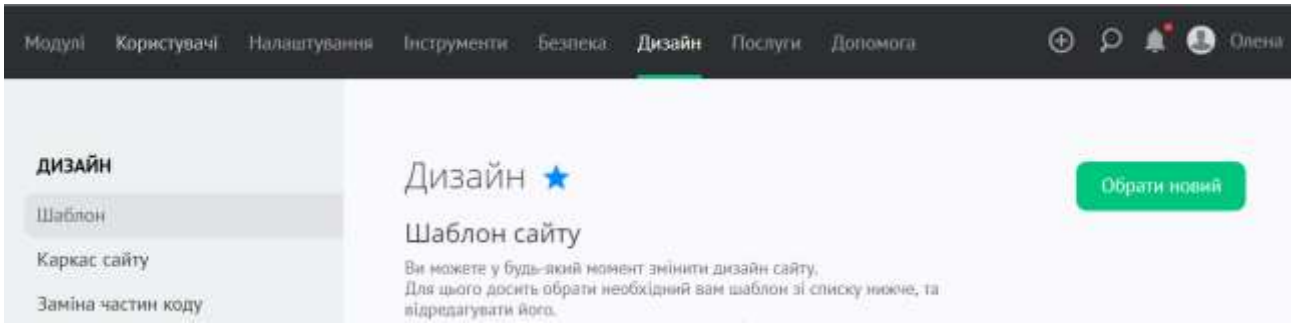


Засобами панелі управління можна:

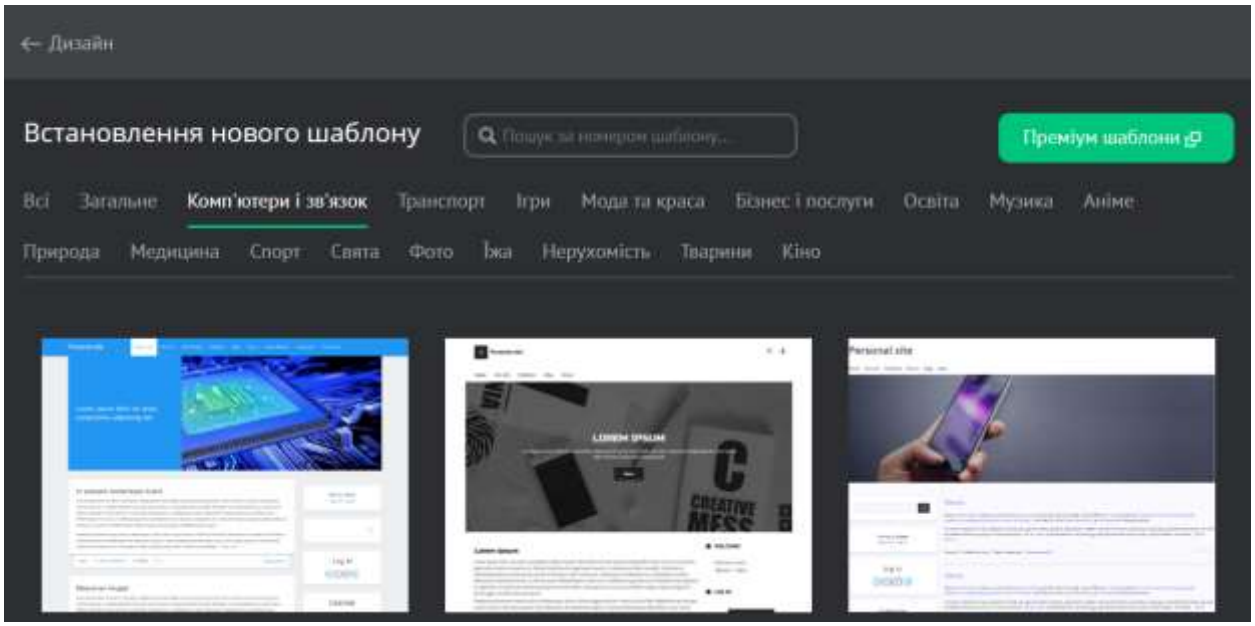
- додавати і видаляти модулі сайту;
- керувати основними налаштуваннями модулів;
- змінювати дизайн сайту;
- активувати додаткові можливості;
- створювати сторінки / категорії / інформери;

– робити резервні копії сайту;
– керувати налаштуваннями домену;
– змінювати права для груп користувачів та багато іншого.

Змінити раніше вибраний шаблон на інший можна, якщо скористатись командою *Дизайн* та натиснувши кнопку *Обрати новий*.

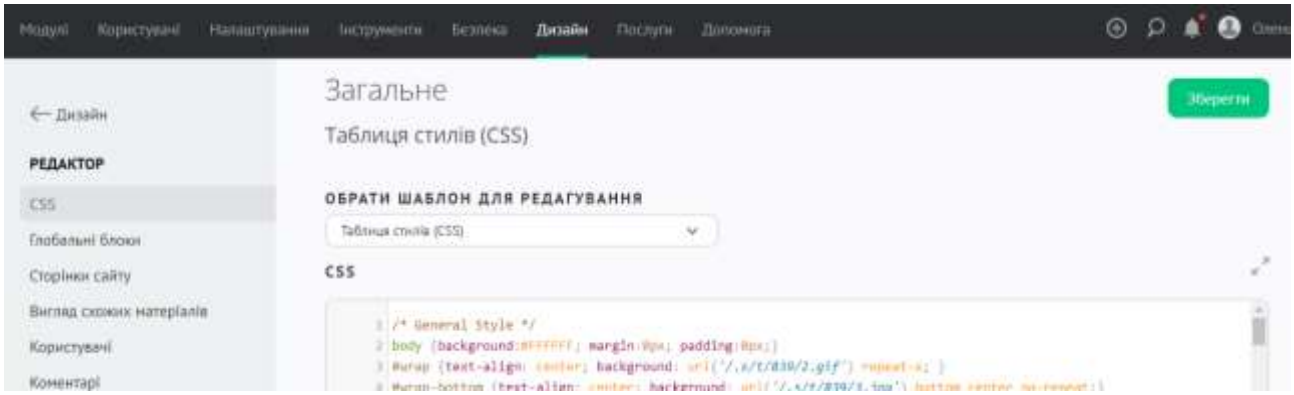


При цьому з’явиться вікно, де всі шаблони згруповані за тематичними рубриками.

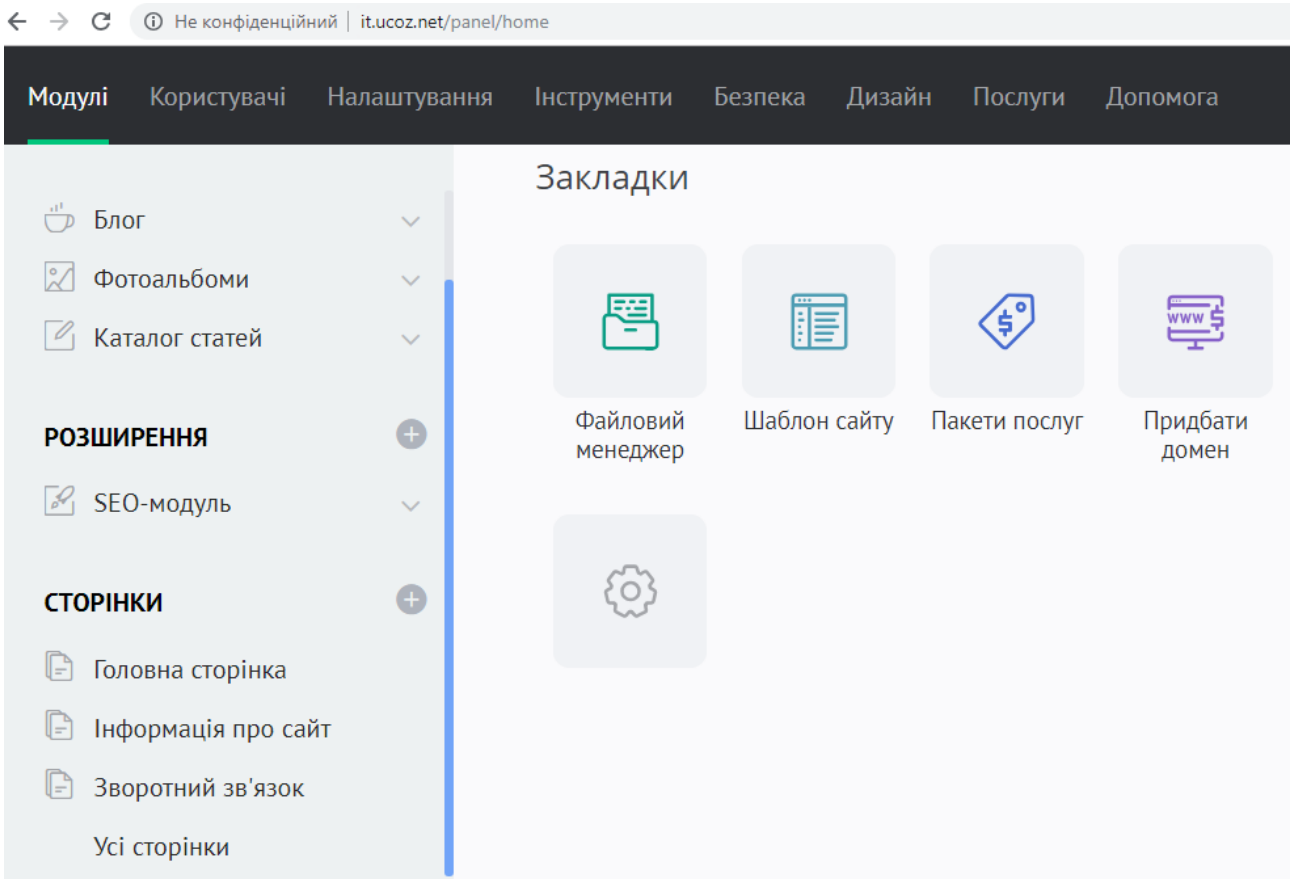


Подальше натискання кнопки *Встановити* на уподобаному шаблоні призведе до змінення стилю створюваного веб-сайту.

Щоб відкоригувати вибраний дизайн сайту, треба у меню справа скористатись командою *Редактор*. Це дозволить гнучко відкоригувати CSS-стилі та HTML-код кожного з блоків на веб-сторінці.



Зайти знову до панелі управління можна за адресою, сформованою з назви сайту на через косу риску /admin, наприклад: http://it.ucoz.net/admin.



Докладний опис інструментів панелі управління описано за посиланням https://www.ucoz.ru/help/start/panel-upravlenija.

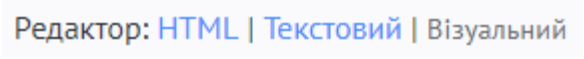
3.3.2. Робота над сайтом у сайт-конструкторі Ucoz

Після того як сайт створено, його можна відразу налаштувати та приступити до наповнення оригінальним авторським вмістом, при цьому публікація сайту відбуватиметься автоматично. У будь-який момент наповнення чи редагування сайту можна переглянути його вигляд, натиснувши на посилання адреси Вашого сайту. При цьому сайт відкриється у новій вкладці браузера.

Для змінення загальних даних треба зайти до панелі управління у розділ *Налаштування / Основні*. Далі можна змінити назву сайту, вибрати мову тощо. Опції *Дата і час* дозволять налаштувати місцевий час, вибрати формати дати і часу.

Для додавання нової веб-сторінки треба спочатку зайти на сайт в ролі адміністратора. Потім скористатися командою *Модулі* і на лівій панелі праворуч від команди *Сторінки* натиснути на кружечок з плюсом , після чого заповнити поле *Назва сторінки*, внести вміст сторінки і за потреби додати зображення.

Для заповнення вмісту сторінки Ucoz пропонує три режими редагування, які можна вибрати під вікном редагування вмісту сторінки:



– *HTML* – конструктор, який дозволяє працювати безпосередньо зі сторінками сайтів у вікні браузера;

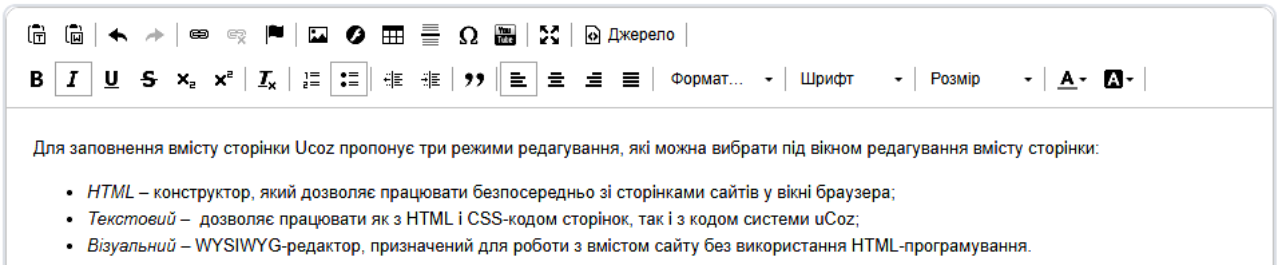
```
<p><span style="tab-stops:42.55pt"><span style="text-autospace:none">Для заповнення вмісту сторінки Ucoz пропонує три режими редагування, які можна вибрати під вікном редагування вмісту сторінки: </span></span></p>

<ul>
    <li><span style="tab-stops:49.65pt"><span style="text-autospace:none"><i>HTML</i> – конструктор, який дозволяє працювати безпосередньо зі сторінками сайтів у вікні браузера;</span></span></li>
    <li><span style="tab-stops:49.65pt"><span style="text-autospace:none"><i>Текстовий</i> –  дозволяє працювати як з HTML і CSS-кодом сторінок, так і з кодом системи uCoz;</span></span></li>
    <li><span style="tab-stops:49.65pt"><span style="text-autospace:none"><i>Візуальний</i> – WYSIWYG-редактор, призначений для роботи з вмістом сайту без використання HTML-програмування.</span></span></li>
</ul>
```

– *Текстовий* – дозволяє працювати як з HTML і CSS-кодом сторінок, так і з кодом системи uCoz;

```
<p><span style="tab-stops:42.55pt"><span style="text-autospace:none">Для заповнення вмісту сторінки Ucoz пропонує три режими редагування, які можна вибрати під вікном редагування вмісту сторінки: </span></span></p>
<ul>
    <li><span style="tab-stops:49.65pt"><span style="text-autospace:none"><i>HTML</i> – конструктор, який дозволяє працювати безпосередньо зі сторінками сайтів у вікні браузера;</span></span></li>
    <li><span style="tab-stops:49.65pt"><span style="text-autospace:none"><i>Текстовий</i> –  дозволяє працювати як з HTML і CSS-кодом сторінок, так і з кодом системи uCoz;</span></span></li>
```

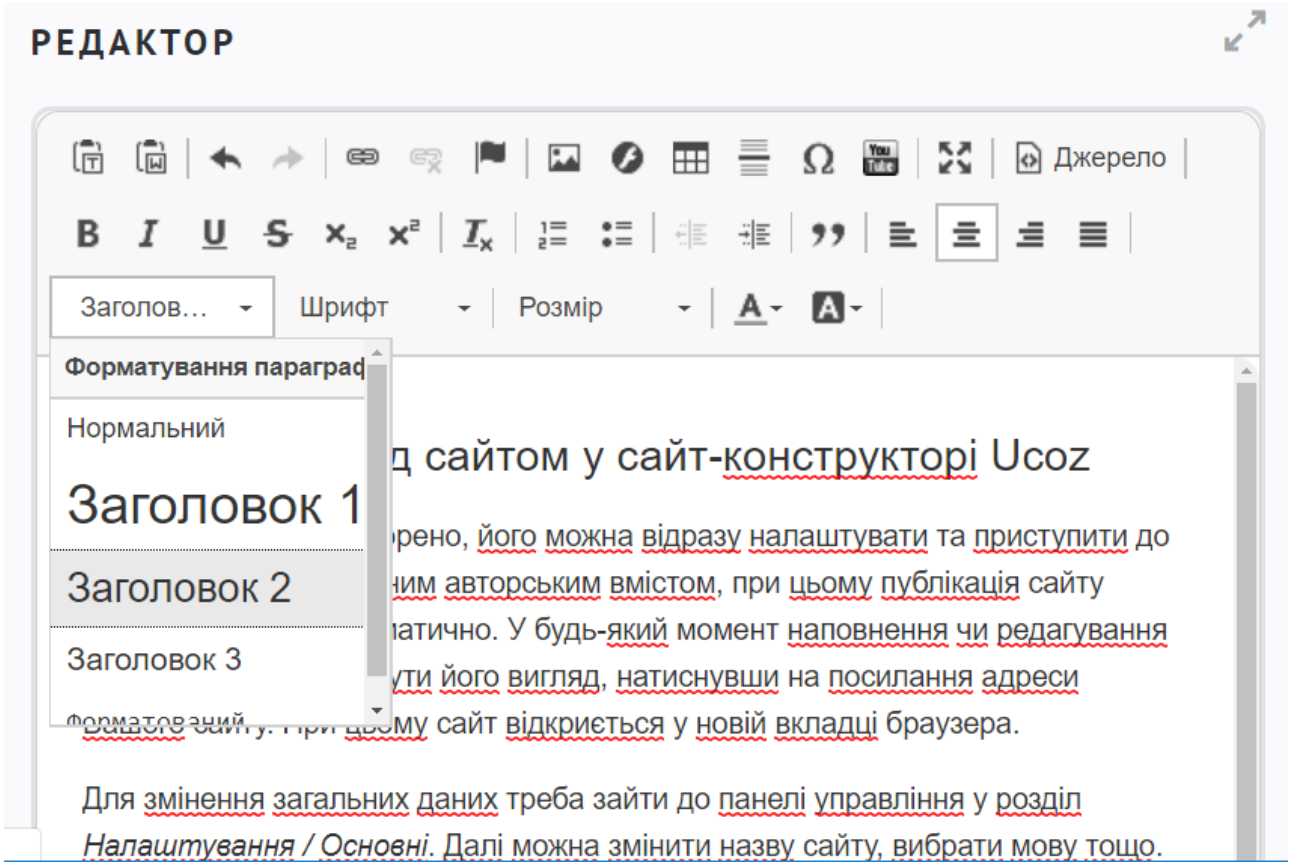
– *Візуальний* – WYSIWYG-редактор, призначений для роботи з вмістом сайту без використання HTML-програмування.



Використовуваний у системі uCoz WYSIWYG-редактор відображає вміст у форматі, схожому з кінцевим виглядом. Основне завдання редактора полягає у максимальному спрощенні процесу роботи із сайтом для людей, які не мають знань з HTML-програмування.

Текст можна не тільки вписувати на кшталт текстового редактора, його можна вставляти, копіюючи з інших джерел. Якщо джерелом є файл Word з форматованим відповідним чином текстом, списками, таблицями тощо, то для збереження параметрів форматування доцільно скористатись іконкою на панелі інструментів візуального редактора, після чого вставити скопійований фрагмент.

Візуальний редактор має доволі широкий спектр засобів форматування. Так, для основного тексту доцільно використовувати нормальний стиль, також передбачено три рівні заголовків. На панелі інструментів є іконки для вирівнювання абзаців, різних накреслень шрифту тощо.



Іконка Ω *Спеціальний символ* дозволяє вставити ріноманітні спеціальні символи, яких немає на клавіаиурі, наприклад: © або §.

Сформувати гіперпосилання з виокремленого тексту дозволить іконка *Вставити / Редагувати посилання*.

Іконка *Вставити / Редагувати якір* призначена для створення і редагування якорів. Якір – це закладка з унікальним ім'ям на певному місці вебсторінці, призначена для створення переходу до неї за посиланням. Якоря зручно застосовувати у документах великих розмірів, щоб можна було швидко переходити до потрібного розділу. Послідовність створення якоря може бути такою:

– поставити курсор на початок абзацу, на який треба організувати швидке переміщення;
– клацнути іконку ;
– ввести ім'я якоря та натиснути *ОК*, після чого на початку абзацу з'явиться червоний прапорець (якір);
– перейти на те місце, де треба організувати гіперпосилання для швидкого переходу на якір, виокремити відповідний текст і натиснути іконку *Вставити / Редагувати посилання*;
– у полі *Тип посилання* вибрати *Якір на цю сторінку* і вибрати потрібний якір.

У будь-який час можна переглянути HTML-код сторінки, клацнувши замість режиму візуального редактора на *HTML*:

Після цього вікно перетвориться на HTML-редактор з тегами.

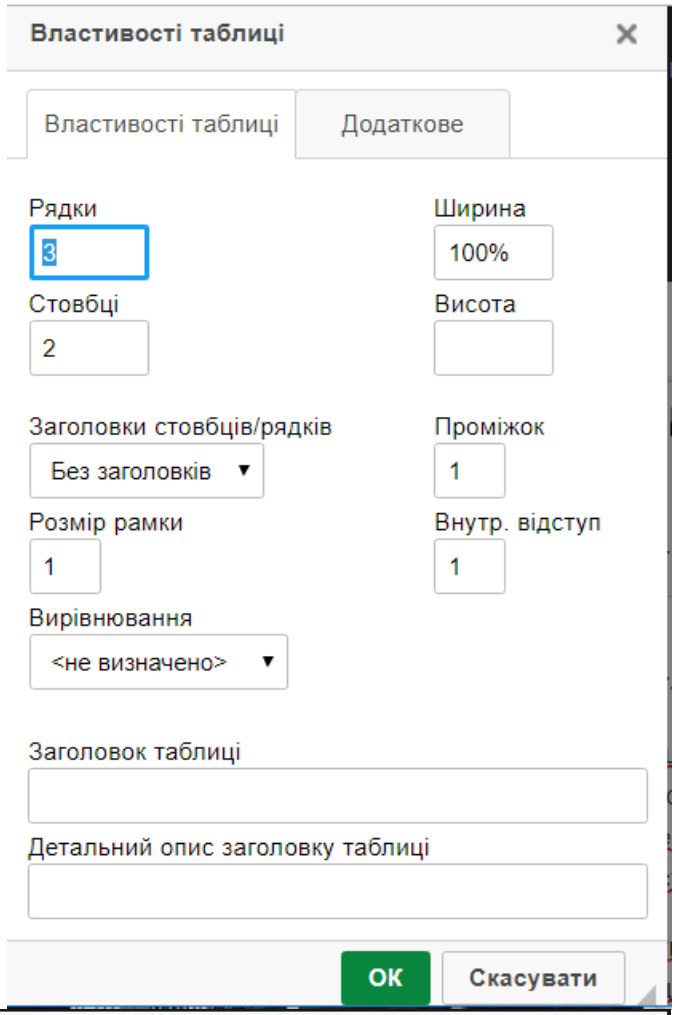
Робота з таблицями

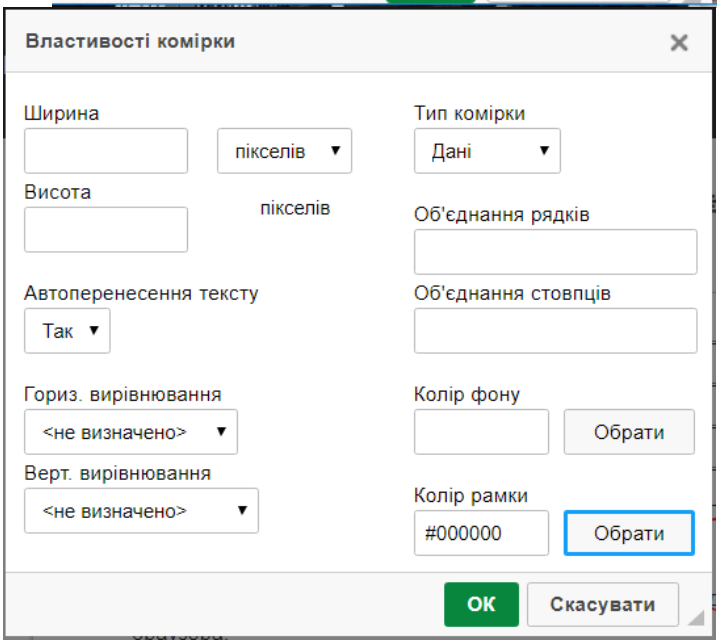
Створити таблицю на вебсторінці можна як копіюванням з файлу Word, якщо такий є, так і у візуальному редакторі за допомогою іконки  *Таблиця.* Після натискання цієї іконки з'явиться діалогове вікно, в якому можна задати розміри (кількість рядків і стовпців) та інші параметри створюваної таблиці (відступи у комірках, вирівнювання тексту у них, колір та розміри меж, заголовки тощо).

Створену у такий спосіб порожню таблицю треба заповнити даними.

Редагувати таблицю (додавати / видаляти / об'єднувати комірки, колонки, рядки тощо) можна, клацнувши по таблиці правою кнопкою миші та вибравши відповідну команду з контекстного меню.

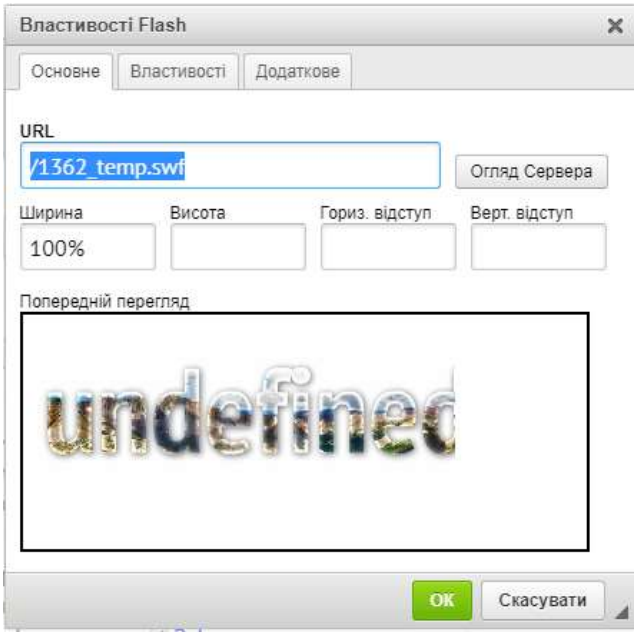
Для того щоб змінити властивості комірки або всієї таблиці (змінити фон, колір меж тощо), слід виокремити потрібні комірки або всю таблицю, клацнути на них правою кнопкою миші та вибрати команду *Комірки / Властивості комірки*. Діалогове вікно *Властивості комірки* дозволить задати детальні параметри: ширину і висоту комірок у пікселях або відсотках, колір рамки (усталено він білий), тип комірок (дані або заголовок), ознаку автоперенесення тексту, вирівнювання тощо.





Робота з Flash

Додати flash-елемент на веб-сторінку дозволить іконка *Flash*. Натискання на неї спричиняє появу діалогового вікна *Властивості Flash* з трьома вкладками: *Основне*, *Властивості* і *Додаткове*. Натискання кнопки *Огляд сервера* дозволить вибрати та завантажити *swf*-файл flash-елемента. Додатково можна задати розміри та інші параметри елемента: масштабованість, вирівнювання, межі, режим відтворення тощо. У полі *Попередній перегляд* можна побачити сам flash-елемент.

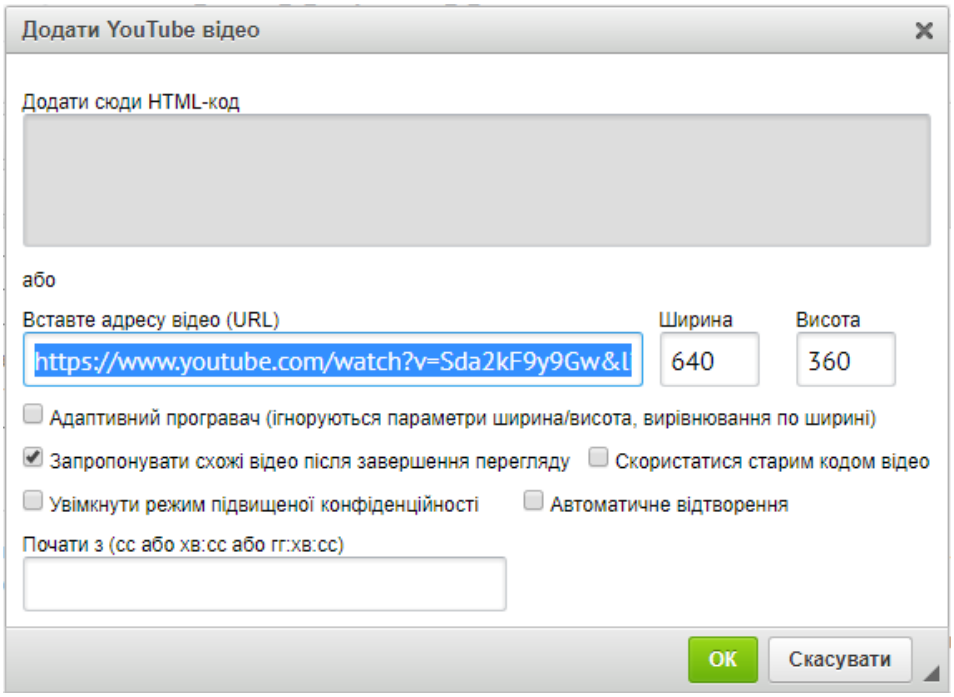


Вставлення відео з YouTube

Вставити відео із сайту YouTube у візуальний редактор можна двома способами:

1) через вставлення HTML-коду;
2) посиланням на відео.

Після натискання у візуальному редакторі на іконку *Додати YouTube відео* з'явиться вікно, в якому можна вставити URL-адресу відео та задати різноманітні опції його відтворення: показувати схожі відео після перегляду; встановити режим підвищеної конфіденційності; автозапуск; ширину і висоту у пікселях тощо.



3.4. Створення веб-блоґу засобами WordPress

3.4.1. Використовувана термінологія та засоби

Блоґ – це особливий тип веб-ресурсу, який характеризується як публічний онлайн-журнал, основним вмістом якого є систематичні записи, що містять в собі текст, зображення, відео, особисті фотографії.

Створення блоґів є дуже і дуже популярним, хоча ще 10 років тому ніхто не чув і нічого не знав про слово «блоґ», яке прийшло до нас із Заходу. Нині особисті блоґи ведуть багато популярних журналістів, артистів, музикантів, спортсменів, політиків. Засобами блоґів вони спілкуються зі своїми шанувальниками, розповідають про події з особистого життя, діляться своїми думками тощо.

За останні роки ведення блоґу стало популярним і в простого люду. Люди заводять блоґи і діляться розповідями про своє особисте життя, про подорожі, про свою роботу, про свої захоплення або навчають чогось інших. А деякі, крім ведення свого онлайн-журналу, ще й заробляють гроші на ньому.

Існують численні інтернет-сервіси для створення сайтів як таких, так і блоґів як різновиду сайту. Охарактеризуємо засоби WordPress для створення блоґів.

WordPress – безкоштовний, загальнодоступний конструктор сайтів, засобами якого можна створювати сайти «з нуля» і керувати їхнім вмістом без технічних навиків і знань. WordPress є системою керування вмістом сайту з відкритим вихідним кодом, яка написана на PHP. Для зберігання даних використовує реляційну базу даних MySQL. Сфера застосування – від блоґів до доволі складних новинних ресурсів і інтернет-магазинів. Вбудована система тем і плагінів разом із вдалою архітектурою дозволяє конструювати проекти широкої функціональної складності. WordPress іноді називають «серцем» або «движком» сайту.

Для використання WordPress слід завантажити із сайту https://uk.wordpress.org/releases одну з версій WordPress (наприклад, файл wordpress-4.9.8-uk.zip).

Далі потрібно систему WordPress встановити на сервер. Як приклад можна скористатися послугами сервера www.ho.ua, який надає послуги як платного, так і безкоштовного хостингу.

Для зручності розуміння процес створення блоґу на WordPress можна умовно розбити на кілька кроків:

1) вибір теми, якій буде присвячений блоґ;
2) замовлення тематичного домену;
3) вибір надійного хостингу;
4) створення бази даних;
5) встановлення WordPress на хостинг.

Виконавши ці прості 5 дій, можна створити свій особистий блоґ. Залишиться лише зробити базові налаштування, встановити потрібні плагіни, створити сторінки і наповнити своє творіння корисним і якісним контентом.

3.4.2. Типова анатомія блоґу

Вміст блоґу можна зробити на кшталт панелі або стрічки, на якій у хронологічному порядку згідно з датами публікації один за одним йтимуть дописи блоґера, так звані пости.

Оскільки з часом у блоґу накопичуватиметься багато постів (зазвичай вони займають кілька веб-сторінок), то найновіший пост розміщують вгорі першої сторінки, при цьому давніші переміщуються донизу. Наприклад, на першій сторінці розміщуватимуться пости цього тижня, друга сторінка присвячена постам за минулий тиждень, третя сторінка ще давнішим і так далі. Зазвичай сторінки блоґу містять посилання на архів блоґу, тобто на попередні пости, згруповані за місяцями і роками. Отже, навігація по блоґу в хронологічному порядку є доволі легкою.

Окрім того, у багатьох системах блоґування постам можна призначати категорії. Ці категорії відбивають тематику постів, як наприклад: «подорожі», «поетика», «сімейні справи» тощо. Тоді відвідувачі блоґу, які цікавляться думками блоґера щодо подорожей, можуть за посиланням на цю категорію перейти до всіх наявних постів автора, присвячених саме цьому.

Типово окремий пост має свій заголовок, дату публікації, власне зміст, який складається з гіпертексту (думки автора, цитати тощо), посилань на інші пости, сайти та блоґи в інтернеті, інколи зображення та відео. Також пост містить коментарі до нього, залишені відвідувачами, та просту веб-форму, за допомогою якої вони долучають ці коментарі.[6]

3.4.3. Вибір тематики блоґу

Перш ніж робити свій особистий блоґ, потрібно визначитися з його тематикою, оскільки саме від цього залежатиме успішне майбутнє сайту. Надамо декілька теоретичних порад щодо вибору тематики блоґу.

Тема вибирається один раз і назавжди, тому потрібно серйозно і відповідально поставитися до цього питання. Для цього спочатку блоґеру доцільно відповісти на декілька питань[7]:

- Яке у вас улюблене захоплення?
- В якій справі Ви є експертом?
- Чим би Ви щиро хотіли ділитися з іншими людьми?
- Що часто просять у Вас зробити друзі?
- Про що 70% життєвого часу Ви розмовляєте з усіма людьми?
- Що би Ви робили просто так, якщо Вам не потрібно було працювати заради забезпечення свого життя?

Після відповідей на всі ці питання можна рухатися далі. Якщо у блоґу буде висока відвідуваність і кілька тисяч постійних читачів, найімовірніше, виникне

[6] Блоґ. URL: https://uk.wikipedia.org/wiki/Блог (дата звернення 28.06.2018).

[7] Маслов И. Как создать блог на WordPress? Пошаговая инструкция от вебмастера! / Иван Маслов. URL: http://ivan-maslov.ru/kak-sdelat-sajt/kak-sozdat-blog-na-wordpress.html (дата звернення 12.07.2018).

змога його монетизувати. Тому обов'язково потрібно з'ясувати: чи користується попитом вибрана тема блоґу в інших людей? Це найважливіший момент, оскільки, якщо, наприклад, вибрати тему «розведення орангутангів у домашній оселі», то створювати блоґ на цю тему просто безглуздо. Людей, які цікавитимуться цим, просто не існує.

Аналіз популярності вибраної теми краще робити декількома способами:

– пошукати в інтернеті форуми на вибрану тему (їхня присутність свідчить про популярність та актуальність);
– пошукати подібні тематичні групи у соціальних мережах, наприклад: відкрити Facebook або Twitter, у пошуку ввести ключові слова вибраної теми, оцінити кількість людей, які цікавляться цим напрямом, і виходячи з цього приймати рішення про звуження або розширення теми;
– пошукати подібні тематичні канали на YouTube;
– зробити аналіз популярних запитів для визначення ключових слів за різними показниками ефективності (кількість запитів, вартість за клік тощо) за допомогою спеціалізованих на цьому онлайн-сервісів:
  – Google Trends (https://trends.google.com/trends/explore?geo=UA);
  – Serpstat (https://serpstat.com/uk/);
  – Buzzsumo (http://buzzsumo.com/);
  – SemRush.com (https://ru.semrush.com/);
  – Keyword Tool (http://keywordtool.io/ru).

Ці або інші подібні сервіси допоможуть віднайти топові публікації за заданими ключовими словами, визначити запити, які найчастіше задають користувачі інтернету з метою подальшого використання відповідних тем для написання статей блоґу. Крім того, налаштування підписки в **Google Alerts** (https://www.google.com.ua/alerts) з оповіщеннями на теми і запити, які стосуються вибраної сфери, дозволить регулярно спостерігати за публікаціями конкурентів та тематичними порталами. Використання при цьому ефективного досвіду конкурентів сприятиме оптимізації веб-сторінок та виходу на топові позиції в результатах пошуку.

Наприклад, для аналізу ключових запитів засобами сервісу **Google Trends** треба перейти за адресою https://trends.google.com/trends/explore?geo=UA (або https://trends.google.com/trends/hottrends#pn=p35) і ввести у полі назву вибраної тематики. Також тут можна вибрати регіон та період часу, після чого натиснути кнопку пошуку. До речі, цей сервіс абсолютно безкоштовний, але потребує реєстрації. Сервіс є доволі зручним, оскільки він дозволяє порівнювати водночас популярність декількох тем.

Наведемо приклад аналізу вибраних нами тем популярних запитів. Так, серед запитів «політичний блоґ», «журналістське розслідування», «інфографіка» лідером є саме останній із запитів. Проте, і цей запит суттєво відстає від популярності запитів: «котики», «субсидії», «спорт», «HTML», «погода». Причому тут ці

запити, як і попередні, розташовані саме за зростанням їхньої популярності. Це дозволить підібрати вдалу тематику блоґу.

3.4.4. Замовлення тематичного домену

Після того як тему майбутнього інтернет-проекту вибрано, можна перейти до наступного етапу і підібрати доменне ім'я. Вибір домену для свого блоґу – заняття цікаве і творче. Наведемо декілька порад щодо вибору доменного імені.

Як і зазначалось у підрозд. 3.1, в імені домену можуть бути літери, цифри і символ дефісу.

Визначитися з доменною зоною для реєстрації імені так само важливо, як і підібрати домен, оскільки вибір домену в певній зоні дозволяє однозначно вказати приналежність сайту.

З точки зору маркетингу доменне ім'я треба вибирати або за назвою компанії, або за напрямом діяльності. Якщо назва складається з декількох слів, їх можна написати разом або з'єднати через дефіс. Якщо бюджет дозволяє, краще зареєструвати доменне ім'я і з тире, і без нього. Це дозволить убезпечити себе від сквотерів та конкурентів.

Домен має бути: бажано коротким, легко запам'ятовуватись і наочно характеризувати тематику блоґу чи сайту. Не рекомендовано назву домену складати з більше ніж трьох слів. Оскільки деякі пошукові системи при пошуку позиціонують краще сайти у назві яких є слова з пошуку, можна підібрати домен з найважливіших ключових слів.

Гарне доменне ім'я легко вимовляється і його важко неправильно написати. Це має особливе значення, якщо для просування сайту планується використовувати рекламу на радіо чи сарафанний маркетинг. Гарне доменне ім'я важко сплутати з іншим. При виборі домену треба переконатися в тому, що немає схожого імені у конкурентів.

При виборі домену також треба переконатися, що назва не порушує авторські права третіх осіб, оскільки це може призвести до вимоги з боку власника торгового знаку видалити домен або передати ім'я йому.

Правильно вибраний домен це ще не все. Якщо бюджет дозволяє, доречно зареєструвати схожі доменні імена для того, щоб надалі цього не зробили конкуренти або сквотери. Наприклад, зареєструвавши домен domain.com, варто також задуматися про придбання домену domain.com.ua, а якщо Ваша компанія працює в Одесі, то і domain.odessa.ua, domain.od.ua та domain.odessa.net.ua.

До речі, можна використовувати різні імена для різних видів реклами, а також для окремих видів продукції. Наприклад, спеціально для реклами на телебаченні і радіо можна використовувати кириличні імена, а для просування в інтернет – звичайні латинські. Надалі всі імена можна без особливих зусиль закріпити за одним сайтом або навіть розділом сайту.

Вибір хостингу – не менш важливий і серйозний пункт при створенні блоґу на WordPress. Фахівці радять: ні в якому разі не публікувати свій веб-

ресурс на безкоштовному хостингу. Це стосується блоґів (і сайтів), які планується використовувати як довгострокові проекти. Для того щоб надалі не виникало зайвих проблем, варто розміщувати свій блоґ відразу на хорошому і перевіреному хості.

3.4.5. Створення бази даних на сайті

Для роботи блоґу у системі WordPress потрібна база даних MySQL. Спочатку розберемося з тим, що таке база даних сайту і навіщо вона потрібна.

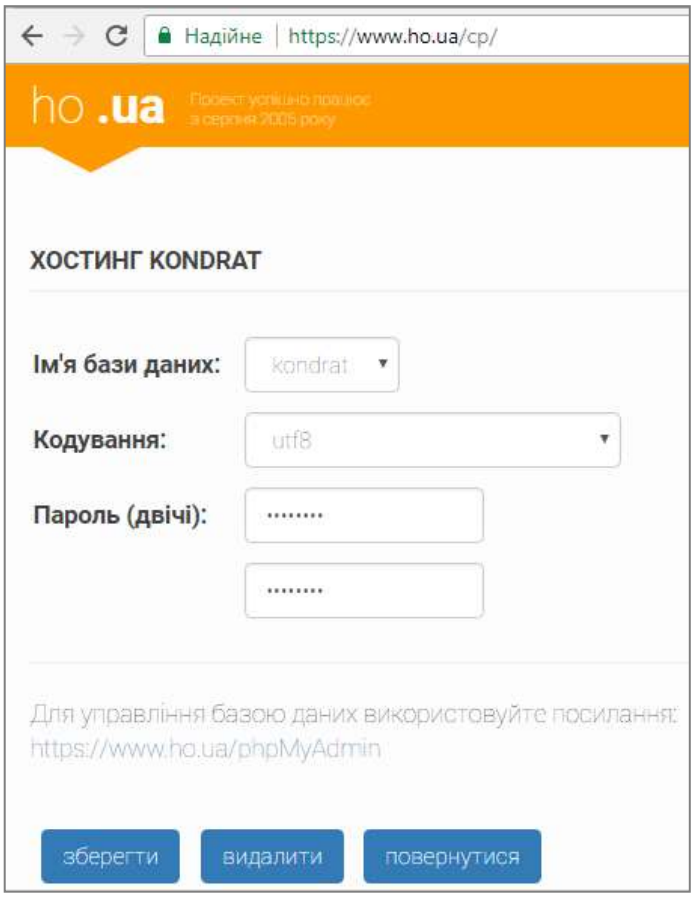
База даних сайту – це місце, де фіксується і зберігається вся опублікована інформація на сайті (паролі, тексти, сторінки, коментарі тощо). Отже, WordPress – це серце сайту, а ось база даних – це, абстрактно кажучи, мізки сайту.

Оскільки попередні пункти носили здебільшого теоретичний характер, зараз детально по пунктах розглянемо порядок створення бази даних на сайті.

1) Треба увійти на свій сайт із правами адміністратора сайту, увівши його URL-адресу. Оскільки ми скористалися послугами сервера www.ho.ua, то для цього у нашому прикладі треба ввести в адресному рядку https://www.ho.ua/cp/, далі ввести логін і пароль (див. лист, який Вам надіслали, у нашому прикладі логін *lozindre*).

2) Зайти до розділу *Управління базою даних*, натиснувши кнопку *База даних*.

3) Далі відкриється вікно, в якому ввести придумане ім’я бази даних, задати кодування utf8, двічі ввести придуманий пароль і натиснути кнопку *Зберегти* для створення бази даних на своєму сайті. На інших хостах створення бази даних виглядає подібним чином.



4) Якщо параметри бази даних Вас задовольняють, натиснути кнопку *ОК*, а якщо ні – натиснути гіперпосилання поряд з *Управління базою*. Оскільки змінити опції можна буде і на подальших кроках, зупиняємось на цьому і натискаємо кнопку *Зберегти*. Після цього буде сформовано вікно на кшталт показаного праворуч.

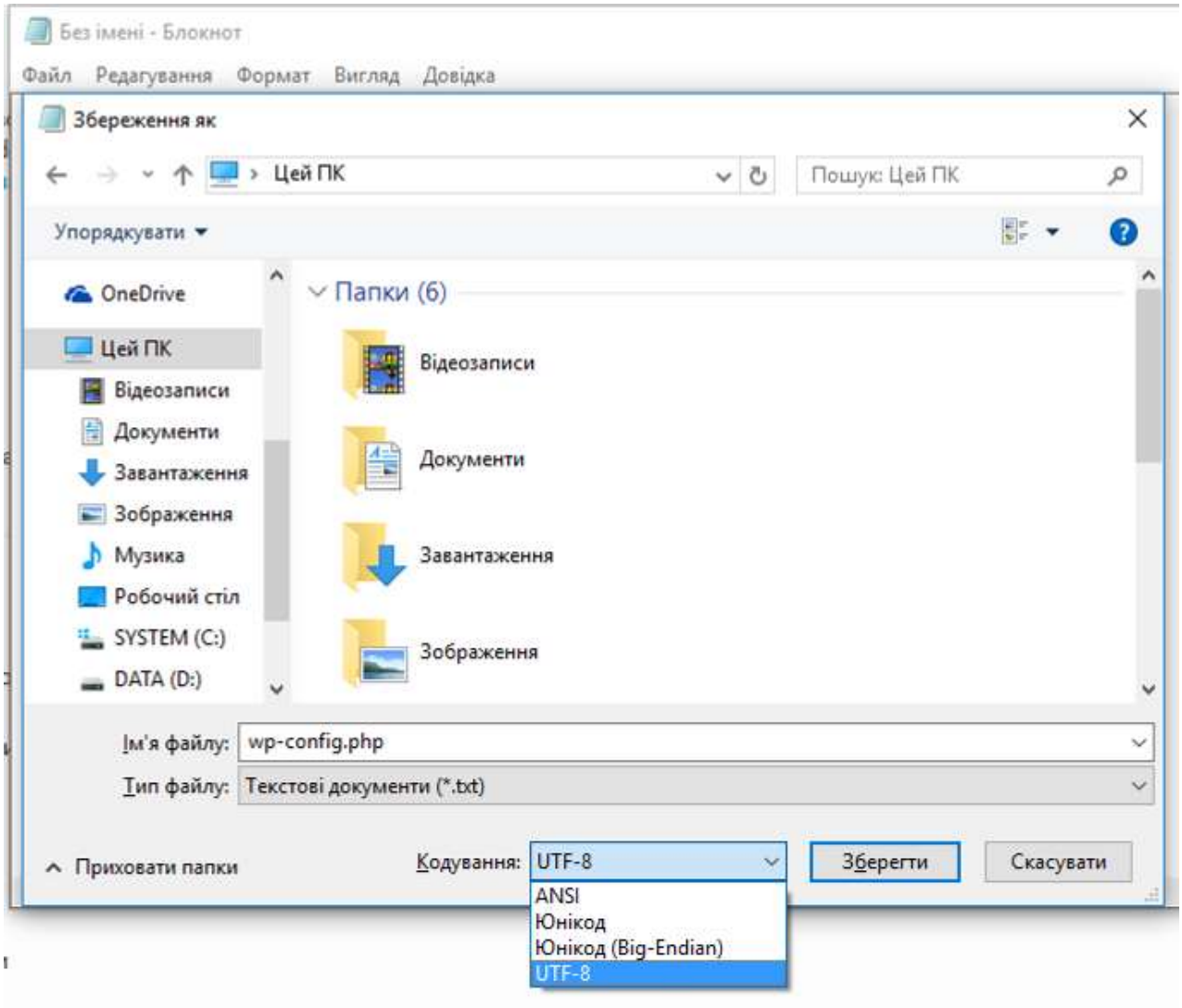
На цьому базу даних успішно створено. Далі, залежно від сервера, можливо треба буде зачекати (до 10 хвилин) допоки база даних підключиться.

3.4.6. Встановлення WordPress на хостинг

Для встановлення WordPress на хостинг спочатку на сайті https://uk.wordpress.org/download/ треба скачати одну з версій WordPress (наприклад, файл wordpress-4.9.8-uk.zip). Далі послідовність дій може бути такою:

1) Розпакувати архів wordpress-4.9.8-uk.zip. Результат – папка з назвою *wordpress* з вкладеними файлами і папками.

2) Найти у папці *wordpress* файл wp-config-sample.php і відкрити його у будь-якому редакторі, щоб редагувати код. Наприклад, можна редагувати цей файл у текстовому редакторі Блокнот, але після редагування треба задати кодування (при зберіганні файлу) UTF-8.



Зараз відредагуємо файл в такий спосіб:

а) Найти рядок:

define(‘DB_NAME’, ‘database_name_here’)

і замість ‘database_name_here’ вставити ім’я своєї бази даних (у нашому прикладі ‘lozindre’). Вийде так:

define(‘DB_NAME’, ‘lozindre’);

б) Найти рядок:

define('DB_USER', 'username_here');

і замість 'username_here' вставити ім'я користувача ('lozindre'). Вийде:

define('DB_USER', 'lozindre');

в) Найти рядок:

define('DB_PASSWORD', 'password_here');

і замість 'password_here' вставити свій пароль, наприклад:

define('DB_PASSWORD', 'GEWfeMUPrl');

г) Найти рядок:

define('DB_HOST', 'localhost');

і замість 'localhost' вставити сервер баз даних 'db2.ho.ua'. Вийде:

define('DB_HOST', 'db2.ho.ua');

д) Зберегти цей файл як **wp-config.php**. Нагадуємо, що треба вибрати кодування UTF-8.

Далі завантажити цей дистрибутив на сервер та виконати встановлення.

3) Для цього спочатку треба заархівувати всі вкладені файли і папки всередині папки *wordpress*. У нашому прикладі ми створили архів з ім'ям wordpress.zip (суттєвим є використання саме *zip*-формату архіву).

4) Далі треба повернутися на сторінку https://www.ho.ua/cp/, де натиснути кнопку «*Управління файлами*» *Зміна вмісту сайту.* – Відбудеться перехід до файлового менеджера.


Ласкаво просимо в **Файл-менеджер** . Якщо у вас виникнуть питання, пов'язані з використанням **Файл-менеджера** , ви можете скористатися довідкою .

Увага: для повноцінної роботи в **Файл-менеджері** в вашому браузері повинна бути включена підтримка **JavaScript** . В іншому випадку, деякі критичні операції будуть виконуватися без підтвердження з вашого боку.

ЗМІСТ КАТАЛОГА ~ /

	ім'я	Режим доступу	Розмір	час модифікації	Опції
	cgi-bin	rwxr-xr-x 755	-	2012-07-16 16:26:39	
	htdocs	rwxr-xr-x 755	-	2017-08-07 23:10:18	
	із зазначеними виконати:	-- виберіть дію --			
	Дискова квота:	**зайнято 272 012 кб; квота 1 048 568 кб; вільно: 776 556 кб**			

5) Увійти до папки *htdocs*, натиснувши на гіперпосилання *htdocs*. Виокремити та видалити усі файли. Підтвердити видалення у відповідному повідомленні. Подібні дії Ви виконували під час самостійних робіт № 2 і № 5.

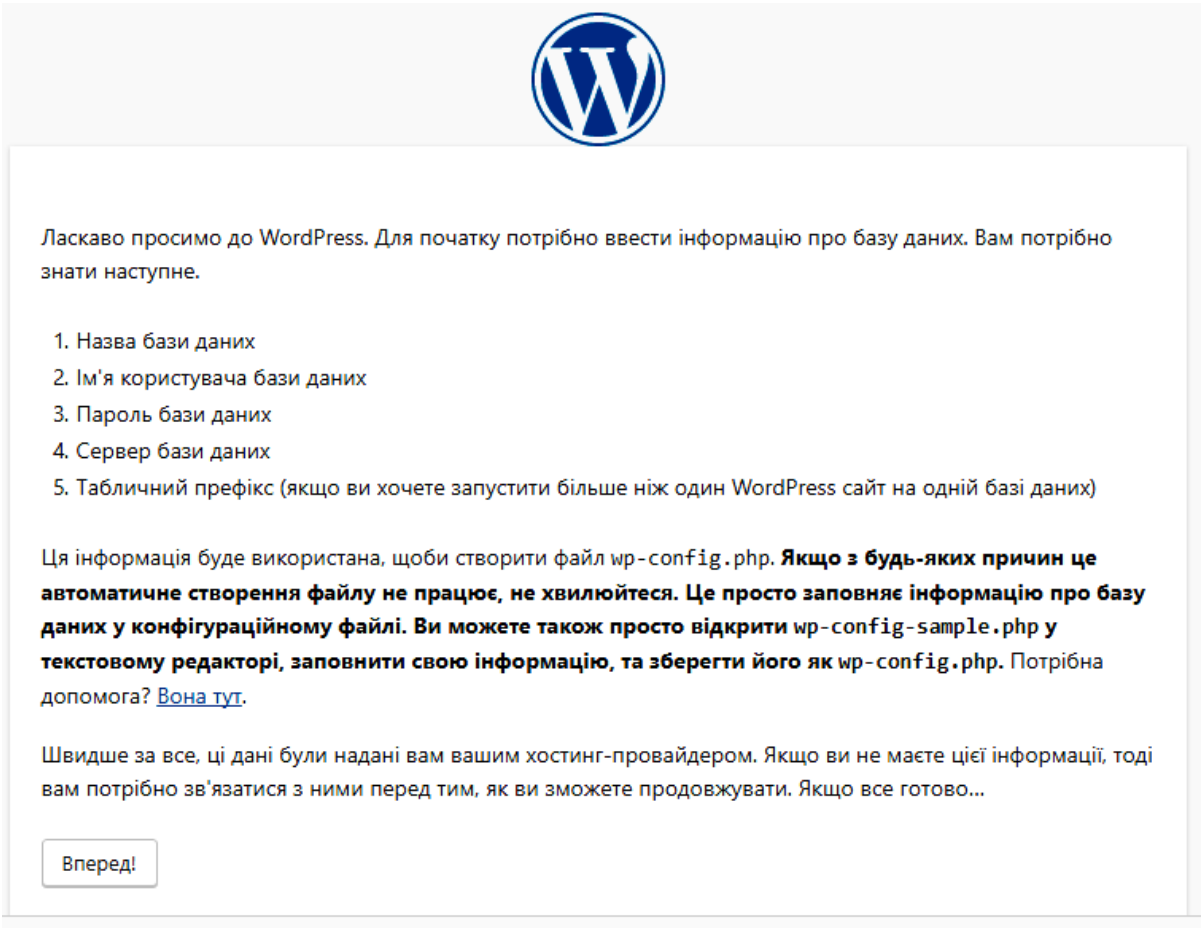
6) Натиснути на значок  *Завантажити файл у каталог* (підказка з назвою значка з'явиться при наведенні вказівника миші на значок лівої нижньої папки).

7) Клацнути кнопку *Вибрати файл* та найти файл архіву wordpress.zip. Далі вибраний файл завантажити, натиснувши кнопку *Завантажити.*

8) Щоб розпакувати архів, треба праворуч файлу wordpress.zip клацнути мишею папку жовтого кольору  *Розпакувати архів у поточний каталог*.

9) Тепер залишилося закрити вікно файл-менеджера, а в новій вкладці браузера набрати адресу свого сайту (у нашому прикладі lozindre.ho.ua) і натиснути *Enter*. Якщо з’явиться порожнє вікно, як при запуску у п. 5, треба оновити сторінку, натиснувши клавішу *F5*.

Натиснути кнопку *Вперед*.



Ласкаво просимо до WordPress. Для початку потрібно ввести інформацію про базу даних. Вам потрібно знати наступне.

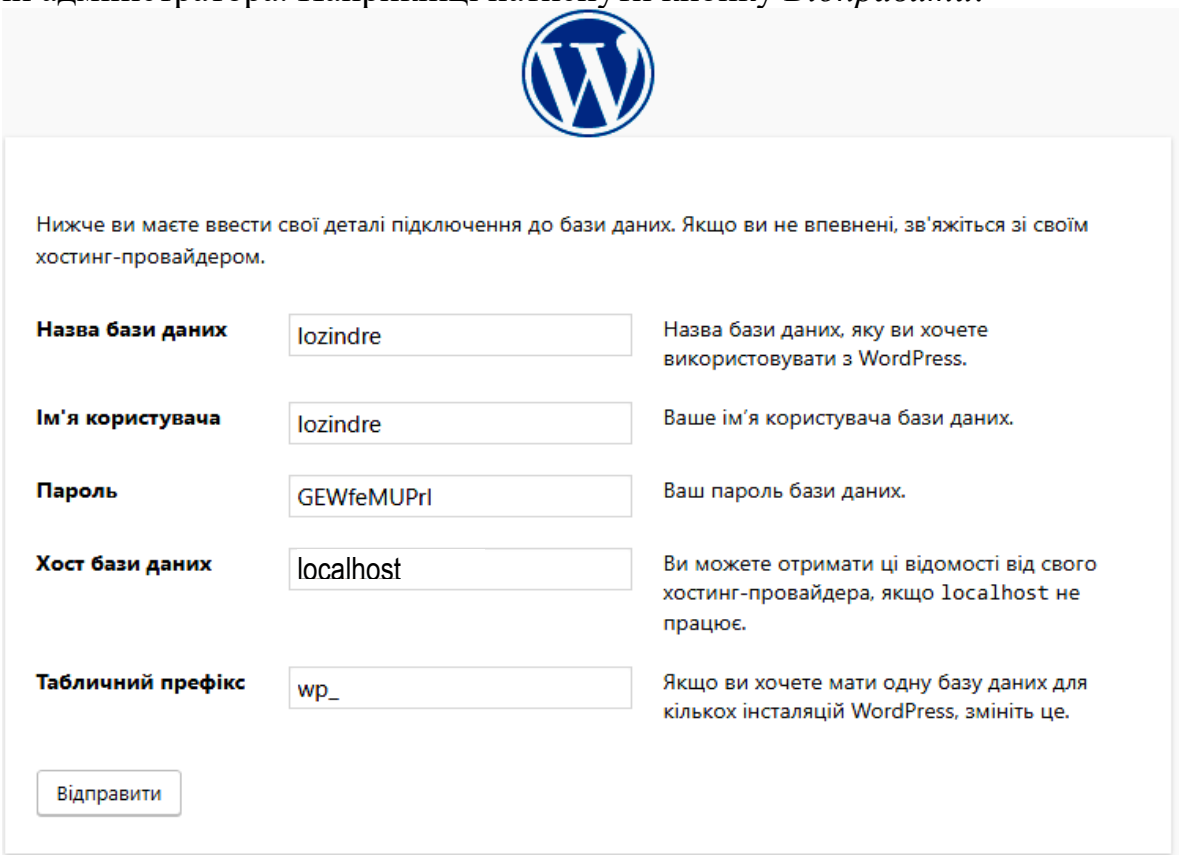
1. Назва бази даних
2. Ім'я користувача бази даних
3. Пароль бази даних
4. Сервер бази даних
5. Табличний префікс (якщо ви хочете запустити більше ніж один WordPress сайт на одній базі даних)

Ця інформація буде використана, щоби створити файл `wp-config.php`. **Якщо з будь-яких причин це автоматичне створення файлу не працює, не хвилюйтеся. Це просто заповняє інформацію про базу даних у конфігураційному файлі. Ви можете також просто відкрити `wp-config-sample.php` у текстовому редакторі, заповнити свою інформацію, та зберегти його як `wp-config.php`.** Потрібна допомога? Вона тут.

Швидше за все, ці дані були надані вам вашим хостинг-провайдером. Якщо ви не маєте цієї інформації, тоді вам потрібно зв'язатися з ними перед тим, як ви зможете продовжувати. Якщо все готово...

Вперед!

10) Далі треба заповнити поля з назвою майбутнього сайту та іншими даними адміністратора. Наприкінці натиснути кнопку *Відправити*.

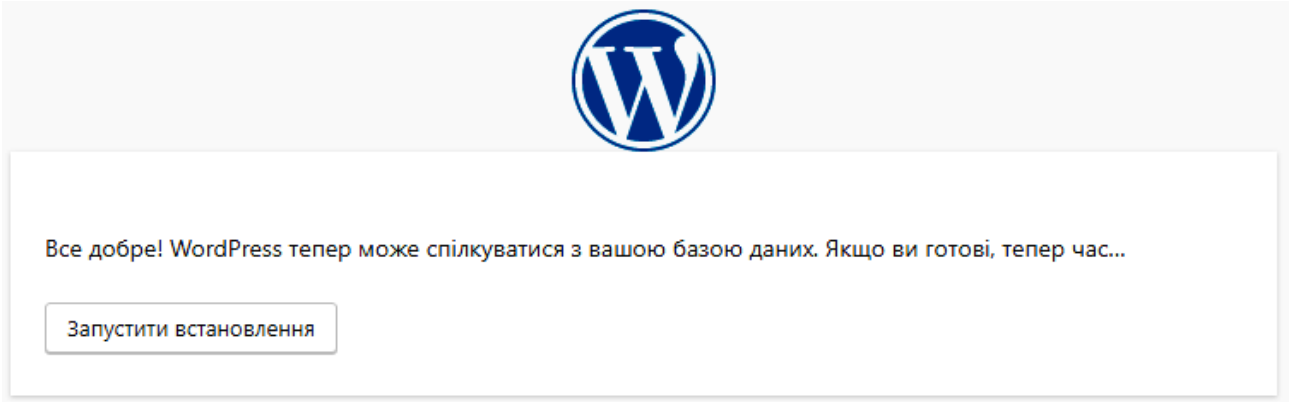


Нижче ви маєте ввести свої деталі підключення до бази даних. Якщо ви не впевнені, зв'яжіться зі своїм хостинг-провайдером.

Назва бази даних	lozindre	Назва бази даних, яку ви хочете використовувати з WordPress.
Ім'я користувача	lozindre	Ваше ім'я користувача бази даних.
Пароль	GEWfeMUPrl	Ваш пароль бази даних.
Хост бази даних	localhost	Ви можете отримати ці відомості від свого хостинг-провайдера, якщо `localhost` не працює.
Табличний префікс	wp_	Якщо ви хочете мати одну базу даних для кількох інсталяцій WordPress, змініть це.

Відправити

Після цього з'явиться вікно, в якому натиснути кнопку *Запустити встановлення*.



Далі заповнити поля для реєстрації (назву сайту, ім'я користувача, пароль, email тощо) та натиснути кнопку *Встановити WordPress.*

Оце і все, *WordPress* встановлено. Можна увійти, зазначивши email та пароль. Далі можна приступити до дизайну та наповнення створеного блоґу. А поки що доречно найти в інтернеті подібні сайти, щоб створити собі уявлення про можливі варіанти їхнього конструювання.

3.4.7. Опис інтерфейсу WordPress

На платформі WordPress визначено п'ять типів ролей, доступних для користувачів[8]:

- адміністратор (administrator) з усіма можливими повноваженнями;
- редактор (editor), який володіє правами адміністратора, за винятком повноважень для внесення змін у конфігурацію веб-сервера;
- автор (author), який створює і публікує власні матеріали (пости);
- учасники (contributor) можуть створювати власні записи, але не мають права публікувати їх самостійно;
- передплатник (subscriber) може тільки читати записи блоґу і залишати коментарі.

Не рекомендується використовувати обліковий запис адміністратора для щоденної роботи, щоб уникнути компрометації системи. Варто створити дублюючий обліковий запис з менш широкими правами і використовувати його.

При вході в систему WordPress першою сторінкою є адміністративна *Майстерня*, на яку виводиться інформація про стан блоґу – кількість коментарів, оновлення, новини WordPress. З цієї майстерні можна швидко перейти до інших розділів адміністрування інтерфейсу веб-сайту (рис. 3.1).

Під номером 1 на рис. 3.1 розташоване посилання *Параметри екрану*, яке дозволяє використовувати метабокси.

[8] Воропай А. Создание Web-сайта на базе WordPress CMS. URL: https://www.ibm.com/developerworks/ru/library/os-wordpress/index.html (дата звернення 29.09.2018).

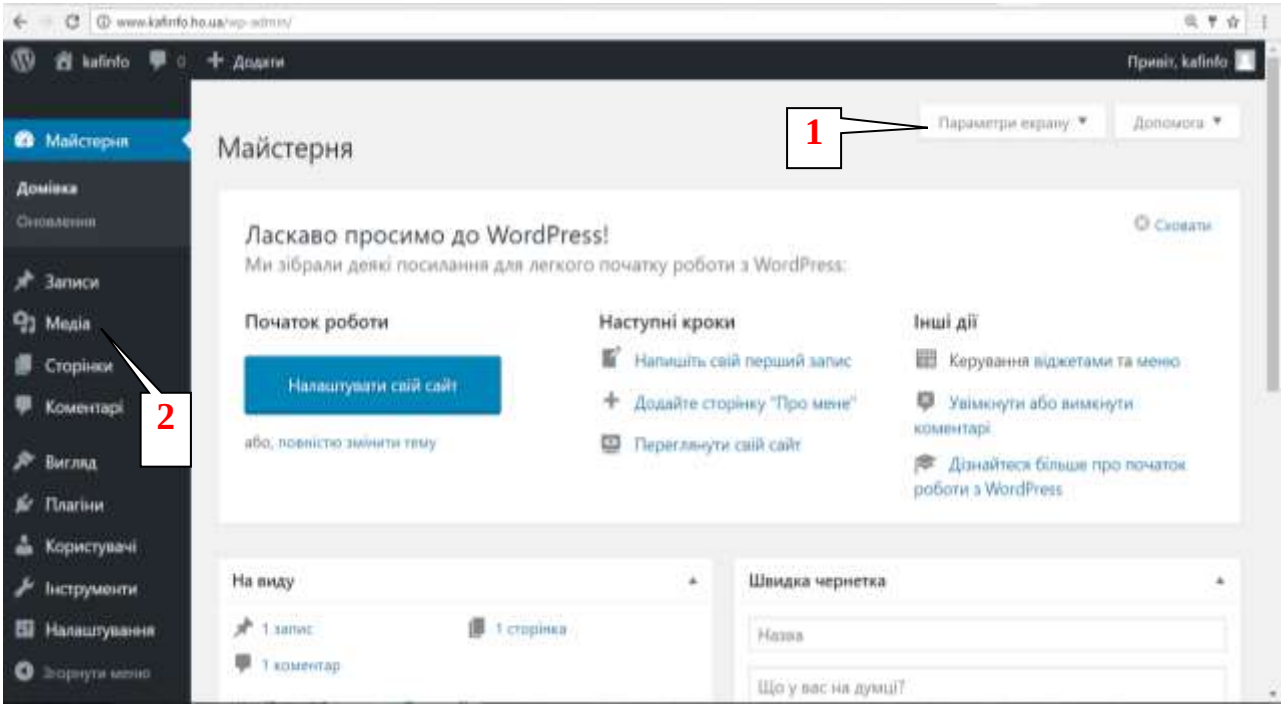
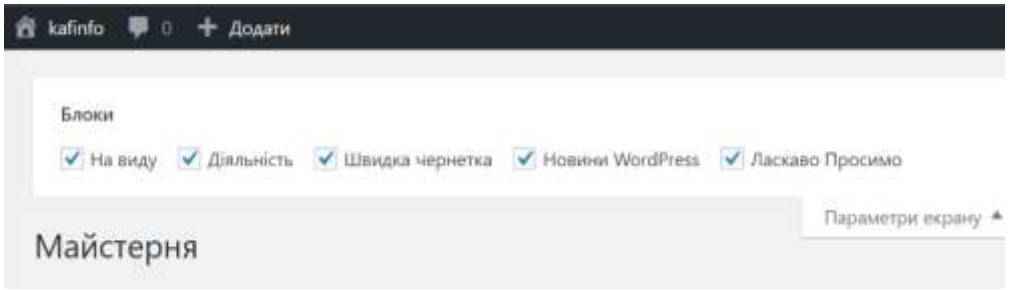
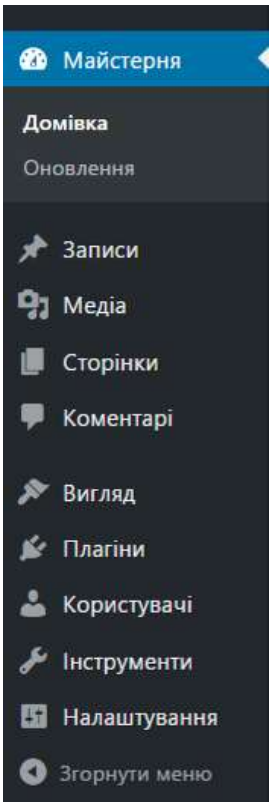


Рисунок 3.1 – Вигляд вікна *Майстерні* сайту

Після клацання по кнопці *Параметри екрана* відкриється зверху додаткова панель для керування метабоксами. Метабокси (модулі) можна міняти місцями перетягуванням мишею. Нові метабокси можна додавати через плагіни або файл functions.php у темі сайту.





Панель навігації (номер 2 на рис. 3.1) використовується для швидкого доступу до найчастіше використовуваних дій в адміністративному інтерфейсі. На панелі навігації розміщено розділи для адміністрування інтерфейсу, а саме:

– *Записи* (Posts);
– *Медіа* (Media);
– *Сторінки* (Pages);
– *Коментарі* (Comments);
– *Вигляд* (Appearance);
– *Плагіни* (Plugins);
– *Користувачі* (Users);
– *Інструменти* (Tools);
– *Налаштування* (Settings).

Кожний розділ має кілька підрозділів, назви яких можна побачити при наведенні мишею на відповідний розділ. Приміром, для створення нового запису треба виконати команду *Записи / Додати*. Після цього з'явиться вікно з такими розділами (рис. 3.2):

1. ***Заголовок*** – поле для введення заголовка статті;
2. ***Візуально*** – переключення до режиму візуального редактора для створення статей без знань HTML;
3. ***Текст*** – переключення до режиму введення даних у вигляді HTML-коду;
4. ***Оприлюднити*** – публікує запис або зберігає як чернетку:
   - ***Статус*** – дозволяє вибрати параметри *Чернетка*, *До огляду*;
   - ***Видимість*** – можна вибрати рівень видимості: *Приватно*, *Захищено паролем*, *Публічно*;
   - ***Оприлюднити негайно*** – вибирається дата публікації;
5. ***Формат*** – можна використовувати для виведення різних типів матеріалу;
6. ***Категорії*** (під розділом *Формат*) – можна вибрати наявну рубрику або додати нову;
7. ***Мітки*** (під розділом *Категорії*) **–** ключові слова, що стосуються статті.

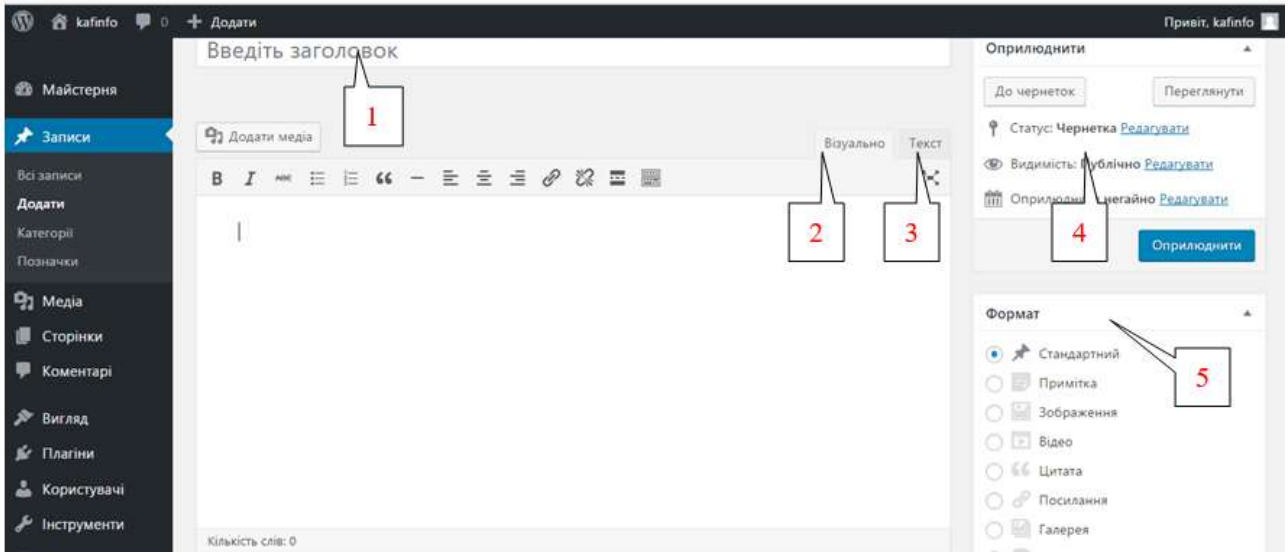
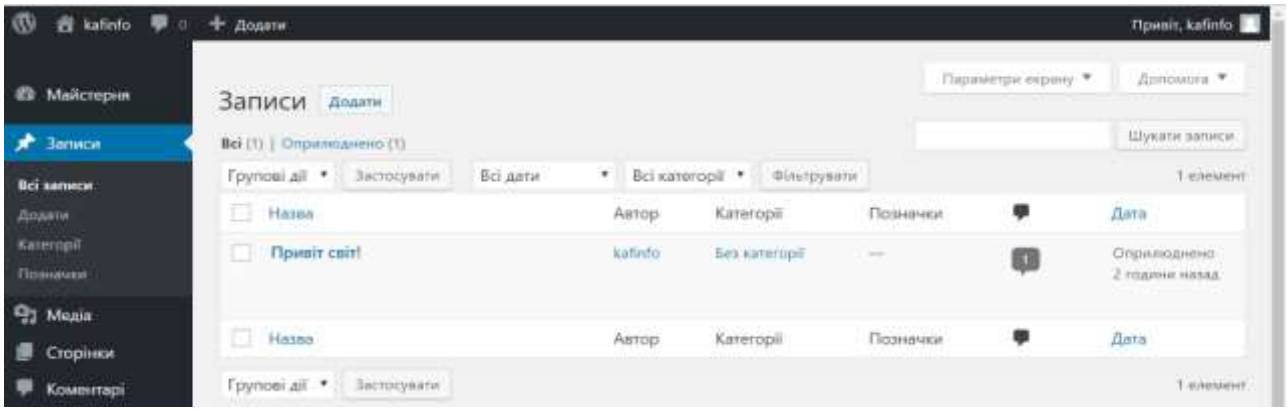


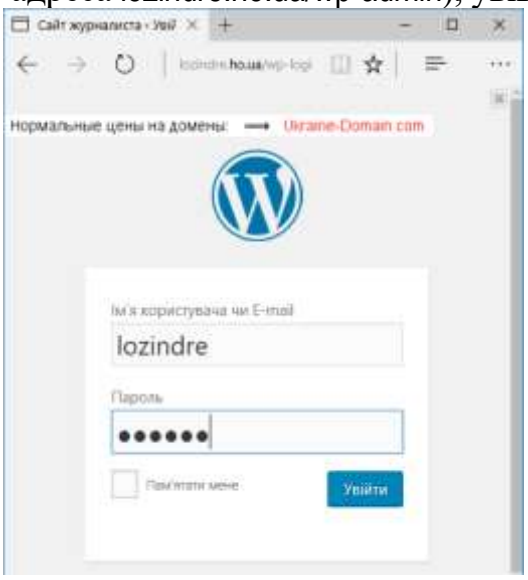
Рисунок 3.2 – Інтерфейс для створення нового запису

Після публікації запису можна зайти у ліве меню *Записи / Всі записи* і перевірити статус нового запису.

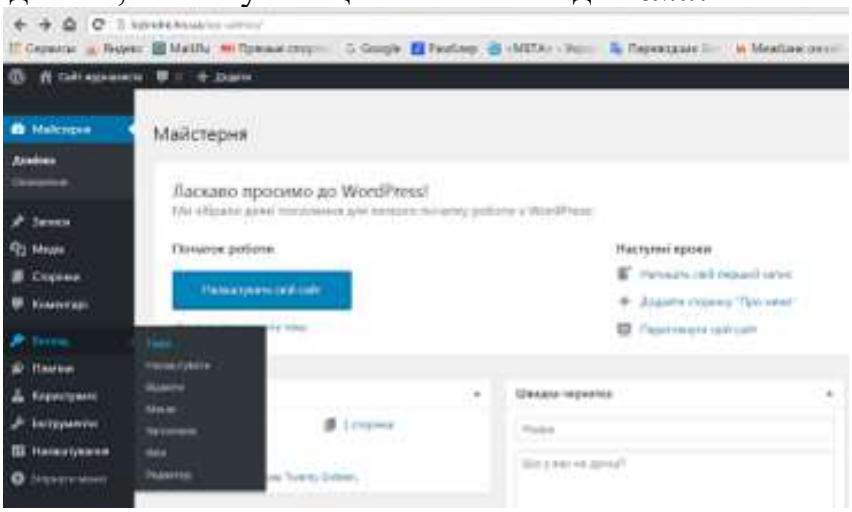


3.4.8. Налаштування теми та структури сайту на WordPress

Для налаштування теми та структури сайту на WordPress треба зайти на свій сайт із правами адміністратора (до URL-адреси свого сайту дописується /wp-admin/, у нашому прикладі URL-адреса lozindre.ho.ua/wp-admin), увівши логін і пароль:



З'явиться *Майстерня* для оформлення сайту (вона завжди з'являється при вході на сайт із правами адміністратора). При наведенні на пункт меню *Вигляд* відкриється підменю, в якому нас цікавить команда *Теми*:



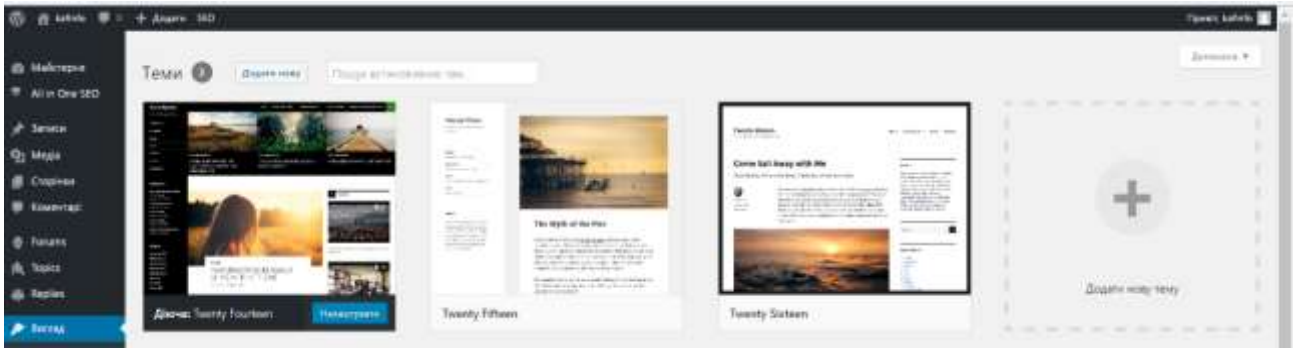
Команда *Теми* (або *Шаблони WordPress*) – це готові макети для популярної системи керування контентом, використовуваної для ведення блогів, новинних сайтів і проектів для електронної комерції. Ці шаблони можуть використовуватись для створення веб-проекту «з нуля» або модернізації вже наявного.

Під **темою** в WordPress розуміються файли (або шаблони), які дозволяють змінювати графічний інтерфейс і стиль відображення вмісту сайту без внесення будь-яких змін у програмний код. Тема складається з файлів шаблонів, зображень (**.jpg*, **.gif*), каскадних таблиць стилів (**.css*) і файлів із PHP-кодом (**.php*).

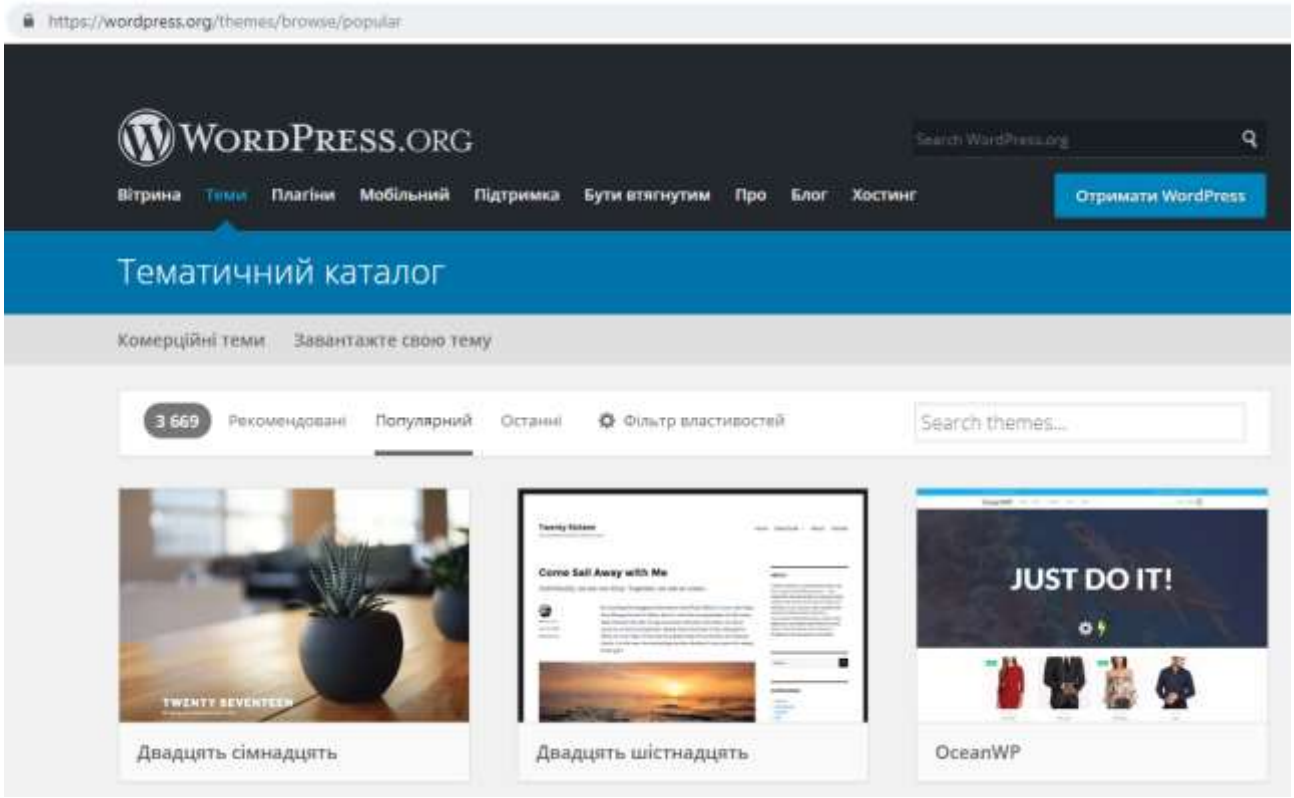
Вдало вибрана для блогу тема (шаблон оформлення) WordPress дозволить залучити трафік та аудиторію, надати привабливості і, навіть, набути престижності. Шаблони влаштовані так, що дозволяють за потреби змінювати структу-

ру або повністю оформлення сайту в будь-який час, за настроєм або відповідно до тематики ресурсу. Цей процес не займе багато часу.

Зокрема, командою *Теми* можна викликати заздалегідь заготовлені теми для WordPress, за допомогою яких можна змінити дизайн сайту.



У мережі існують численні варіації тем для сайтів, які без зусиль можна скачати і встановити на власний сайт, форум, блог. На сьогодні тільки на офіційному сайті WordPress у папці *Theme Directory* (https://wordpress.org/themes/browse/popular) налічується понад 3600 тем для вільного скачування. Після додавання нових тем у певну папку на сервері, вони з'являться у розділі *Теми*, як показано на рисунку:



Файли кожної нової теми містяться в окремій папці з відповідною назвою. Щоб тему можна було використовувати, вона повинна мати певний набір файлів:

– style.css – головний файл таблиці стилів;
– index.php – головний файл шаблонів;
– comments.php – шаблон коментарів;

– home.php – шаблон головної сторінки.

Для встановлення теми достатньо просто скопіювати її файли у папку **themes** або скористатися адміністративним інтерфейсом WordPress.

Побачити свій сайт у тому вигляді, як його бачитимуть відвідувачі, можна відповідною командою у верхньому лівому куті екрана. Зазначимо, під час вибору дизайну сайту доцільно періодично переглядати вигляд сайту на іншій вкладці браузера, оновлюючи її після того чи іншого змінення вигляду сайту.

При наведенні миші на назву сайту відкриватиметься підменю, за допомогою якого можна знову повернутися до *Майстерні* для подальшого редагування сайту.

Крім встановлення WordPress на певний хостинг, зареєструватися та отримати безкоштовний хостинг можна і на сайті https://wordpress.com. Для цього треба зайти на цей сайт і натиснути кнопку *Приступайте* (*Get Started*). Станом на осінь 2018 року сервіс дозволяє вибрати одну з 19 мов інтерфейсу. На жаль, української серед них немає, але можна вибрати російську.

Далі пропонується 5 кроків для вибору спрямованості сайту або блоґу:

1) оскільки у прикладі розглядається створення блоґу, натиснути кнопку *Начать с блога*;
2) вибрати одну із популярних тем (згодом після реєстрації можна буде вибрати іншу тему);
3) підібрати домене ім’я, тобто IP-адресу майбутнього сайту, або вказати адресу свого зареєстрованого хостингу, якщо така є;
4) вибрати тарифний план (у рамках нашої дисципліни – безкоштовний);
5) вказати власне ім’я, адресу поштової скриньки та придумати пароль.

Далі на Вашу поштову скриньку прийде лист із посиланням для підтвредження реєстрації.

Після цього можна приступити до налаштування відображення блоґу та його наповнення. Наприклад, можна вибрати тему, створити меню певної структури, зайти в *Настройки* і задати ім’я. До речі, якщо натиснути на кнопку *Настроить* поряд із командою меню *Темы* у групі *Персонализация*, можна задати докладно різні налаштування, зокрема і власний дизайн.

3.4.9. Додавання форуму на сайт Wordpress

Для створення форуму потрібен плагін *bbPress 2.0*[9], який можна встановити через меню *Плагіни* в адміністративному інтерфейсі *Майстерні*. Як результат на адміністративній панелі навігації з’являться нові розділи, як показано на рисунку праворуч.

Для створення нового форуму треба клацнути посилання *New Forum* та пройти через серію екранів майстра створення форуму.

[9] Воропай А. Создание Web-сайта на базе WordPress CMS. URL: https://www.ibm.com/developerworks/ru/library/os-wordpress/index.html (дата звернення 29.09.2018).

Приклади вигляду різних форумів можна побачити за адресами:
– http://odesskiy.com/odesskiy-forum/;
– www.ukrcenter.com/Форум;
– http://forum.maidan.org.ua/.

Отже, платформа WordPress заслужено є однією з найпопулярніших і поширених CMS[10]. Так, стандартний блоґ можна створити підключенням всього декількох модулів, а зручність навігаційного меню і легкість налаштування надає змогу адаптувати сайт під конкретні завдання. На цей час для WordPress існує понад 52 тисяч модулів, крім того веб-сайти, розроблені на цій платформі, займають високі позиції в провідних пошукових системах.

WordPress можна назвати ідеальною платформою для початківців вебмайстрів, а до її недоліків можна віднести тільки складну структуру бази даних і великий обсяг пам'яті, потрібний для роботи даної CMS. Однак у сучасних умовах ці проблеми легко розв'язні.

Розглянуті в роботі прийоми роботи з Wordpress можуть допомогти при створенні власних сайтів для блоґів чи інших цілей, і, крім того, слугуватимуть відправною точкою для внесення змін у стандартний функціонал WordPress.

3.4.10. Використання плагінів

Отже, WordPress – зручна платформа для створення веб-сайтів, яка користується популярністю серед розробників. Але, на жаль, її базова функціональність обмежена і може запропонувати розробнику тільки стандартний набір опцій. Тому для розширення можливостей WordPress використовуються спеціальні модулі (плагіни), які дозволяють модифікувати базову систему для створення сайтів, що відповідатимуть специфічним вимогам того чи іншого проекту.

Плагін[11] для WordPress – це невелика програма, яка встановлюється в систему у вигляді окремого блока і допомагає розширити її можливості. Простіше кажучи, це доповнення до базової програми, метою якого є поліпшення функціональності веб-сайту або впровадження додаткових можливостей. Основною перевагою плагінів є те, що при їх впровадженні не потрібно вносити зміни в базову програму. Крім того, при оновленні базової системи плагіни не чіпають і вони функціонуватимуть в колишньому режимі.

Модулі є корисним доповненням до WordPress, оскільки вони не тільки використовуються для наповнення сайту інформацією, а й дозволяють розширити можливості основної системи. Більшість розробників не можуть обійтися тільки стандартними функціями WordPress, тому існує безліч плагінів для вирішення найрізноманітніших завдань. Проте не варто забувати про той факт, що

[10] CMS (від англ. *Content Management System* – система керування контентом) – це програмне забезпечення, яке дозволяє користувачам розміщувати або змінювати вже розміщену на сайті інформацію без залучення розробників сайту. Іноді CMS попросту називають «движок» сайту.

[11] **Плагін** (від англ. *plug-in* – підключати) – незалежно компільований програмний модуль, динамічно долучений до основної програми і призначений для розширення і/або використання її можливостей. Просто кажучи, плагін – це доповнення (розширення можливостей) для будь-якої програми на Вашому комп'ютері або движку сайту в інтернеті.

надмірне навантаження системи різними плагінами може серйозно позначитися на її продуктивності. Саме тому треба намагатися вибирати найбільш оптимальні реалізації плагінів, які відповідатимуть заданим вимогам, але не будуть надавати негативного впливу на роботу системи в цілому.

Щоб встановити плагін, його спочатку треба скачати. Наприклад, можна скачати плагін *wptouch* (архівний *zip*-файл) з офіційного сайту WordPress (https://wordpress.org/plugins/wptouch/).

Після цього треба зайти у *Майстерню* (нагадуємо, *Майстерня* завжди відкривається при вході з правами адміністратора, у нашому прикладі, при введенні URL-адреси – lozindre.ho.ua/wp-admin/).

Виконати команду *Плагіни* / *Додати* і клацнути кнопку *Завантажити плагін*. Далі клацнути по меню *Плагіни* й активувати цей плагін за допомогою кнопки *Активувати*. Вибрати файл плагіна wptouch.4.2.5.zip за допомогою кнопки *Вибрати файл* і встановити його за допомогою кнопки *Встановити*.

Контрольні запитання

1) Що таке IP-адреса? Яка форма її запису?
2) Що називають доменним ім’ям? Які правила його запису?
3) Які існують рівні доменних імен? Навести приклади доменних імен різних рівнів.
4) На які групи поділяються доменні імена першого рівня?
5) Які Ви знаєте українські домени другого рівня?
6) Які поради є щодо вибору доменного імені?

7) Що називається конструктором сайтів? Які конструктори сайтів Ви знаєте?
8) Які інструменти зазвичай пропонуються у конструкторах для роботи із сайтом?
9) Назвати основні етапи створення сайту за допомогою конструктора сайтів uCoz.
10) Що таке WYSIWYG-редактор?
11) Як змінити пункти меню сайту за допомогою конструктора сайтів uCoz?
12) Як активувати або видалити модуль в uCoz?
13) Що таке конструктор шаблонів в uCoz і як ним користуватися?
14) Що таке віджети, як їх підключити в uCoz?
15) Як налаштувати права доступу на сайті uCoz?
16) Які інструменти розкрутки інтегровані в uCoz?
17) Яке призначення WordPress?
18) Що таке хостинг?
19) Як вибрати надійний хостинг?
20) Що таке блоґ?
21) Якою є послідовність дій для створення блоґу на WordPress?
22) Чому до вибору теми блоґу треба ставитися серйозно і відповідально? Які кроки треба виконати, щоб правильно вибрати тему блоґу?
23) Для чого в WordPress потрібна база даних MySQL?
24) Який порядок створення бази даних у WordPress?
25) Як встановити WordPress на сервер?

Розділ 4
Послідовність дій і специфіка створення сайту

4.1. Послідовність дій при створенні сайту

Створення веб-сайту починається зі створення інформаційної моделі сайту. Будь-яку веб-сторінку можна оцінити за двома параметрами: вміст та зовнішній вигляд. Проте спочатку потрібно вирішити, яку інформацію на ній розміщувати. Варто детально проаналізувати, скільки та якої інформації потрібно подати на веб-сторінці. Створюючи проект сайту, треба добре продумати його загальну структуру, зміст інформації та посилання.

Етапи створення сайту:

1) **Проектування інтерфейсу майбутнього ресурсу і складання технічного завдання (ТЗ)**. Спочатку визначається тематика і призначення майбутнього сайту, завдання, які ставляться перед ресурсом. Саме на цьому етапі малюється докладна карта сайту, де вказуються всі розділи та формат інформації, що надається на їхніх сторінках, а також деталі навігації по сайту.

При розробленні структури сайту варто визначитися з необхідною кількістю сторінок та встановити зв'язки між ними. Розрізняють лінійну, ієрархічну та довільну структури сайтів[1] (рис. 4.1).

Лінійну структуру веб-сайту доцільно використовувати у разі послідовного подання інформації, наприклад, про товари та послуги або матеріали навчального посібника[2]. Перегляд таких сайтів здійснюється послідовно: від початкової (головної) до останньої сторінки. Кожна сторінка має посилання тільки на одну, наступну сторінку сайту. Інколи для зручності навігації по сайту до сторінки також додається посилання на попередню сторінку.

При **ієрархічній структурі** створюється одна сторінка (головна), яка не має попередніх, решта сторінок мають лише одну попередню сторінку (рис. 4.1). При ієрархічній структурі кожна сторінка може містити посилання на довільну кількість сторінок сайту. Така структура найкраще підходить для сайтів, що містять різну за тематикою інформацію: каталогів, зібрань статей з різних тем або добірок.[3]

---

[1] Етапи створення веб-сайтів. URL: http://edufuture.biz/index.php?title=Етапи_створення_веб-сайтів (дата звернення 28.08.2018).

[2] Етапи створення веб-сторінок. URL: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2013/127/127.html (дата звернення 25.08.2018).

[3] Етапи створення веб-сайтів. URL: http://alextexnok.blogspot.com/p/blog-page_85.html (дата звернення 28.08.2018).

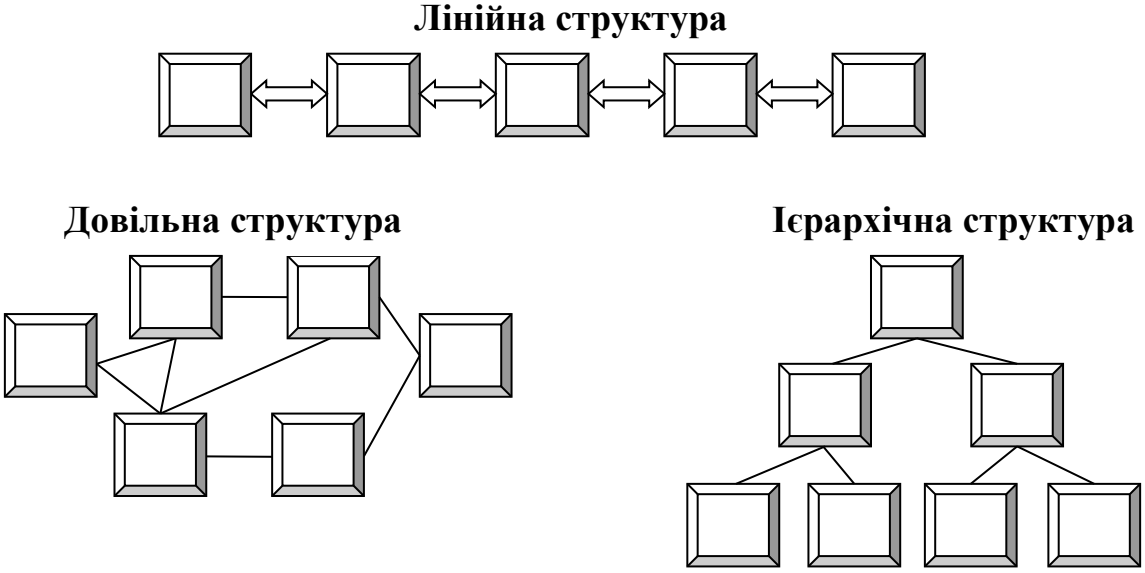


Рисунок 4.1 – Структури веб-сайтів

Найчастіше для створення сайтів використовують **довільну структуру**. При такій структурі сайту його сторінки пов'язані між собою довільним чином (рис. 4.1). У сайтах довільної структури можна виділити фрагменти, які є лінійними або ієрархічними. Прикладом довільної побудови структури сайту є інтернет-енциклопедія Вікіпедія[4].

На етапі розроблення треба також здійснити добір матеріалів і вибрати програмні засоби, за допомогою яких будуть розроблятися веб-сторінки.

Грамотно складене техзавдання дозволяє уникнути непорозуміння між замовником і виконавцем, що також економить час, який витрачається на роботу. Тому ТЗ – документ, без якого просто не обійтися. ТЗ – це основа порядку в роботі над проектом і головний орієнтир. Тільки після виконання всіх вимог, зазначених у ТЗ, проект може вважатися закінченим.

На етапі проектування інтерфейсу доречно використовувати зразки уподобаних сайтів. Для цього можна знайти щонайменше п'ять сайтів зі схожою структурою. Якщо метою є зробити спортивний сайт, то можна брати за приклад новинний. Якщо це landing page, доречно заглянути на land-book.com.

2) Розроблення дизайну сайту[5]. Як і розроблення сайту в цілому, робота над його дизайном також є поетапною. І першим відбувається створення дизайну головної сторінки ресурсу, який починається з розроблення концепції. Вона

[4] Цей проект стартував у 2001 р. Нині статті до цієї енциклопедії пишуться 250 мовами народів світу. Характерною особливістю Вікіпедії є те, що її статті відкриті для доповнення і змін будь-яким користувачем. У Вікіпедії на цей час розміщено більше 6 млн. статей. Орієнтовна кількість авторів та редакторів – 50 тис. В україномовному розділі енциклопедії налічується 124 тис. статей, працюють 15 тис. авторів та редакторів. Вікіпедія перебуває в десятці найбільш відвідуваних веб-ресурсів світу за кількістю запитів сторінок щомісяця. Так, кожен 200-й запит в Інтернеті направляється у Вікіпедію. Вікіпедія складається з окремих розділів. Працюючи з матеріалами цієї енциклопедії, ви можете здійснювати різноманітні переходи та самостійно визначати послідовність перегляду окремих сторінок.

[5] Етапи розробки сайту. URL: https://websait.uz.ua/website-development/etapy-rozrobky-sajtu/ (дата звернення 28.08.2018).

створюється на основі спроектованого у ТЗ інтерфейсу. Проектування головної сторінки сайту розпочинається з того, що за допомогою ручки на папері роблять декілька варіантів розташування елементів[6]. Саме тому, перед цим доречно переглянули якомога більше подібних сайтів і вибрати уподобані варіанти дизайну чи то елементи дизайну, які якнайкраще виконуватимуть головну задачу. Далі відбувається доопрацювання, в ході якого промальовуються способи виділення меню, спливаючі вікна, випадні списки.

При створенні загального малюнка головної сторінки та необхідних графічних елементів не треба відразу зациклюватися на кольорах. Можна спочатку зробити його чорно-білим, а потім додати акцентний колір для посилань і кнопок. Варто звертати увагу на розміри елементів, стиль і розмір шрифтів, колірну гаму тексту і фону, оскільки погано зверстаний текст кидається в очі.

Після доопрацювань відбувається затвердження концепції дизайну головної сторінки, і дизайнери приступають до створення концепцій внутрішніх сторінок, які також проходять етапи узгодження, доопрацювання та затвердження. При створенні дизайну окремих сторінок відповідно до структури треба продумати вигляд і зручність гіперпосилань для переходів (навігації) як в межах сторінки чи між сторінками сайту, так і на зовнішні ресурси за потреби.

Зазвичай на головній сторінці сайту розміщують логотип компанії – торговий знак, за яким би її ідентифікували. Якщо такого логотипу ще не існує, на цьому етапі доречно розробити його. Природно, що логотип розробляється ще до початку відтворення концепції дизайну головної сторінки. Це відбувається тому, що без урахування корпоративної символіки не можна створити по-справжньому оригінальний і цілісний дизайн.

Дизайн – це талант. Якщо Ви маєте намір стати дизайнером, будьте готові, що перші роботи будуть жахливі. Доопрацьовуйте їх. Дивіться приклади і думайте, чому в інших вийшло, а у Вас – ні. Виправляйте. Знову практикуйтеся і коли-небудь Ви поглянете на свою роботу і скажете: «Це круто»[7].

Підсумок етапу – затверджені макети дизайну всіх сторінок сайту здебільшого у форматі PSD, тобто реалізовані у Photoshop.

3) Верстка сторінок сайту. Верстка має здійснюватися відповідно до сучасних вимог, після чого пройти перевірку на валідність та сумісність з усіма сучасними браузерами і коректність відображення на моніторах різних розмірів.

У верстці доречно використовувати засоби, які дозволяють зробити код компактним, щоб збільшити швидкість завантаження веб-сторінок. На етапі верстки на основі ТЗ і макетів дизайну сторінок сайту здійснюється програмування функціоналу, тобто народжується майже готовий веб-ресурс. Саме зараз здійснюється скриптування елементів дизайну, якщо воно передбачене, веб-

---

[6] Етапи створення веб-сайтів. URL: http://edufuture.biz/index.php?title=Етапи_створення_веб-сайтів (дата звернення 28.08.2018).

[7] Сидоренко И. Как стать дизайнером. Главная ошибка и необходимые навыки. *Личный опыт сотрудников Mail.Ru Group, Badoo, Trood*. URL: https://designpub.ru как-стать-дизайнером-главная-ошибка-и-необходимые-навыки-1b6da258b96c (дата обращения 3.11.2018.)

програмісти створюють і тестують необхідні для сайту компоненти і модулі, оптимізують створений дизайнерами шаблон сайту під різні браузери і розміри (роздільні здатності) моніторів, а верстальники підготовляють й оптимізують для пошукових систем контент сайту.

Результатом етапу є готовий сайт у вигляді зверстаних html-файлів усіх сторінок сайту.

4) Тестування. Після того як роботи зі створення сайту на сервері розробки завершаться, розпочинається тестування, перевірка та редагування веб-сайту.

На цьому етапі потрібно перевірити:

– чи правильно працюють усі гіперпосилання;
– чи зручною є навігація сайтом;
– чи відкриваються при відкритті сторінок графічні зображення;
– чи зручно розташовані для сприйняття матеріали на сторінках тощо.

У разі потреби, слід внести зміни у наповнення або структуру сайту, змінити гіперпосилання. Перевіряється відповідність сайту описаному в ТЗ функціоналу, коректність відображення верстки у всіх підтримуваних браузерах на моніторах різних розмірів і відповідність сайту внутрішнім вимогам якості. Повторне тестування сайту проходить після його перенесення на хостинг.

5) Перенесення на хостинг (розміщення сайту в інтернеті).

Спочатку треба визначити, де буде розміщено створений сайт:

– на власному сервері установи;
– на сервері вашого провайдера;
– на сервері організації, яка спеціалізується у наданні послуг розміщення сайтів користувачам інтернету;
– на сервері, який надає послуги вільного і безкоштовного розміщення сайтів.

На цьому етапі створення сайту перевіряється доступність бажаного доменного імені. Корисно дослідити історію домену (розшукати його попередніх власників) і перевірити наявність штрафних санкцій до домену з боку пошукових систем. Залежно від найдених результатів або приступають до реєстрації доменного імені або продумують інший кращий варіант написання імені. Залежно від обсягу створюваного сайту та його типу, що визначає подальші перспективи росту ресурсу, вибирається тарифний план на хостинг.

Заключним етапом розроблення сайту є його розміщення на хостингу.

Після розміщення сайту в інтернеті потрібно здійснювати його підтримку, щоб сайт не втрачав своєї популярності. Ця підтримка полягає в періодичному оновленні та доповненні наявних матеріалів, створенні нових цікавих сторінок тощо.

6) Адміністрування ресурсу. На цьому етапі вже є функціонуючий вебсайт, але про нього мало хто знає. Він усього лише один з мільярдів, хоча розроблено його правильно та зі смаком. Для підняття рейтингу й залучення відвідувачів до створеного сайту необхідно періодично оновлювати вміст, реєструвати сайт у пошукових системах і каталогах, давати оголошення на дошках

оголошень і залишати записи в гостьових книгах, форумах і соціальних мережах. Усі ці дії називаються «розкрутка й просування сайтів».

Розкрутка сайту перетворить інтернет-ресурс зі звичайного атрибута компанії на ефективний маркетинговий інструмент. Однак розкрутка сайту – це складний технологічний процес, який залежить і потребує врахування безлічі нюансів, зокрема перманентного дослідження постійно мінливих пошукових алгоритмів[8].

Просування сайту включає пошукову оптимізацію сайту як джерело залучення зацікавлених відвідувачів на сайт із пошукових систем і найбільш дієвий у наш час інструмент рекламної кампанії в інтернеті.

Оптимізація сайту – складова частина просування й розкручування сайту. Мета оптимізації сайту – досягти високої релевантності сторінок сайту за важливими для даного бізнесу ключовими запитами у пошукових системах. Тобто досягти високої відвідуваності і, як наслідок, досягти комерційного успіху сайту.

Розкрутка й просування сайтів має за мету підвищення позиції сайту у видачі пошукових систем, ріст відвідуваності сайту за запитами, зробити бренд впізнаваним, для комерційних компаній підвищення рівнів продажів, збільшення доходів, покращення конкурентоздатності.

Згодом на сайті можуть з'явитися перебої в роботі, сайт зникне з перших сторінок пошукової видачі, він може морально застаріти або його можуть зламати. Саме тому важливо не лише періодично оновлювати і додавати контент (копірайтінг), щоб нагадати про себе пошуковим системам, а й на регулярній основі проводити роботу щодо оптимізації сайту: усувати вразливі місця захисту, робити резервне копіювання даних, періодично змінювати паролі, слідкувати за обслуговуванням бази даних, оновлювати модулі і компоненти, враховувати змінювані алгоритми ранжування у пошукових системах, вдосконалювати веб-дизайн з урахуванням новітніх тенденцій тощо.

Отже, існування сайту та його повноцінна робота неможливі без регулярних заходів щодо його підтримки, а саме: наповнення сайту новим контентом, редагування наявних матеріалів та своєчасного оновлення необхідних компонентів і модулів, захисту від мережевих атак тощо. Увесь комплекс подібних робіт – це ніщо інше, як **адміністрування сайту**[9].

Адміністрування сайту має здійснюватися на постійній основі, адже відвідувачам цікавий «живий» ресурс, що містить актуальну інформацію та виконує всі заявлені функції. До **завдань адміністратора сайту** може входити його інформаційна та технічна підтримка, а також роботи з модернізації, які не потребують кардинальних змін дизайну або функціональності сайту.

Переважно адміністрування сайту виконується безпосередньо силами його власника. Для цієї мети або наймається спеціальний співробітник або ж на-

[8] Розкрутка сайтів та їх просування. URL: http://webstudio2u.net/ua/promotion.html (дата звернення 28.08.2018).
[9] Адміністрування сайту. URL: http://webstudio2u.net/ua/studio-web/673-administrirovanie-saita.html (дата звернення 28.08.2018).

вчається якийсь з працівників. При цьому, якщо сайт виконаний на CMS (система управління контентом), то адміністрування сайту зазвичай не вимагає знань у сфері веб-дизайну та веб-програмування.

Втім, навіть використання CMS на сайті не дає підстави вважати, що адмініструванням можна знехтувати або ж покласти його на плечі одного із співробітників компанії-власника сайту як додаток до основних обов'язків. Насправді, в адмініструванні сайтів, як і в будь-якій іншій справі, **потрібен ретельно продуманий підхід**.

4.2. Інформаційне наповнення сайту

Увесь контент охороняється законом про авторське право, оскільки він є продуктом інтелектуальної праці і має своїх авторів і власників. Окрім якості контенту, одним з важливих критеріїв є його **доступність**. Особливу важливість для користувача має **актуальність** контенту, його значущість на цей час і достовірність наданих даних, а також відповідність контенту щодо поставлених цілей.

Унікальний контент[10] (ексклюзивний контент) – це інформація, яка не має аналогів на ресурсах схожої тематики або розміщена на веб-сайті з дозволу правовласника, така, що є результатом інтелектуальної праці та охороняється законом «Про авторське право і суміжні права»[11]. Найчастіше цей термін застосовують до текстового наповнення сайтів (текстовий контент).

Унікальні статті, що написані для конкретного ресурсу, розміщуються на ньому і є першоджерелом, будь-який передрук допустимий лише з дозволу законного власника і за його умовами. Грамотне, якісно виконане і цікаве наповнення сайту здатне значно допомогти компанії та її сайту – підвищити відвідуваність, популярність, прибуток.

Один зі способів отримати контент для сайту – **рерайт** (ReWrite) готових статей. Рерайт – це перерозподіл домінанти тексту, змінення форми без зміни змісту, так і розбавлення контенту необхідними ключовими фразами, за якими розробник подає сайт. При рерайті, так само, як і при копіпасті, постає питання законності й етики. Щоб не переступати межі етики, варто обов'язково посилатися на джерело цих ідей.

Правила і поради щодо якості контенту[12]:

– текст має бути написаний цікаво, привертати увагу й затримувати відвідувача;

---

[10] Інформаційне наповнення сайту. URL: https://uk.wikipedia.org/wiki/Інформаційне_наповнення_сайту (дата звернення 27.08.2018).

[11] Про авторське право і суміжні права: Закон України. *Відомості Верховної Ради України (ВВР)*. 1994. № 13. Ст. 64.

[12] Інформаційне наповнення сайту. URL: https://uk.wikipedia.org/wiki/Інформаційне_наповнення_сайту (дата звернення 27.08.2018).

– у тексті не повинно бути орфографічних помилок, друкарських недоречностей, використання різних елементів у межах одного тексту (наприклад, різних термінів для одного й того самого поняття), оскільки це може заплутати читача;

– текст має бути унікальним для пошукових систем. Брати чужий текст і ставити на свій сайт, змінивши лише назву компанії, неправильно. За це Ваш сайт може бути вилучено з баз пошукових систем;

– слід оптимізувати текст під пошукові системи – додати (органічно вписати) достатньо багато ключових слів і виразів;

– текст має бути «читабельним» для відвідувачів сайту. Часто власники сайту так «оптимізують» текст, що жоден відвідувач не в силах затриматись на сторінці вже після першого речення;

– текст має виконувати задану для нього мету: інформувати, переконувати, продавати тощо;

– текст треба розміщувати на сторінці так, щоб відвідувач не плутався у великому масиві тексту без єдиного виділеного рядка і не шукав довго те зображення, яке описується в тексті.

4.3. Характеристики якості сайту

Виділяють такі важливі характеристики якості сайту[13]:

1) **Призначення сайту.** Оцінювання цього критерію допомагає дати відповідь на питання: «Чи виконує ресурс поставлені завдання?»

Сайт має насамперед повною мірою виконувати необхідні функції, для яких призначений цей проект. Насправді, не важко проаналізувати сайт за ознакою «потенційна функціональність сайту»: спочатку треба визначити призначення проекту, далі, виходячи з цього, визначити міру його функціональності і скласти перелік необхідних та достатніх критеріїв оцінки з потрібною ієрархією.

З точки зору професійного підходу до методики оцінювання сайту «призначення сайту» є найважливішим критерієм. Решта факторів (тематичне наповнення, художнє оформлення) порівняно з ним є якщо не другорядними, то хоча б повинні розглядатися тільки після оцінки призначення проекту. Для цілей власника не має значення ані повнота досліджень висвітленої теми, ані динамічність чи інтерактивність сайту, ані естетичні критерії аналізу сайту, якщо ресурс не виконує свою функцію, нехай навіть службового сателіту, який є фундаментом для основного проекту.

Отже, спочатку треба визначити ціль створення ресурсу, а вже потім стане зрозумілим, наскільки глибоко треба характеризувати решту оцінювальних факторів.

---

[13] Огірко І., Паславська І., Пілат О. Інформаційна система оцінювання якості електронних видань. *Вісник Львівського університету. Серія економічна*. 2013. Вип. 49. С. 391–398.

2) **Загальні ознаки якості з погляду користувача**: зовнішній вигляд, простота навігації на сайті, зручність системної конструкції сайту, зручність інтерфейсу, своєчасне оновлення інформації, дотримання авторських прав на використання інформації.

3) **Стиль подачі матеріалу**: розмір шрифту, ширина рядка, приємний для очей міжрядковий інтервал, колірний контраст, доступність копіювання, масштабування, цитування та пересилання інформації.

Попередньо налаштований браузером **розмір шрифту** для довгих текстів переважно невеликий, його важко сприймати, браузери виставляють його за умовчанням. При створенні браузера передбачалося, що він буде показувати текст саме цього розміру. Коли користувач читає такий текст, він не має бажання натискати на кнопки збільшення або зменшення шрифту, не хоче змінювати свої налаштування, він просто хоче читати, хоче щоб текст був налаштований максимально близько до його уподобань. Набагато складніше зробити гарний макет сайту з великим розміром шрифту, але ця складність допоможе зробити простішим, доступнішим і зрозумілішим сайт для користувача. Наповнити сайт інформацією нескладно, а от зробити його простим і зручним у користуванні важче. Спочатку викликатиме велике здивування користувача завеликий попередньо встановлений у браузерах розмір шрифту, але з часом користувач вже не захоче читати текст менший 100% або 1 em.

Ширина рядка набору повинна бути пропорційна до розміру шрифту. Занадто широкі рядки стомлюють очі і викликають несприятливий психологічний ефект. Занадто вузька колонка також дратує, оскільки порушує безперервність читання. Зайвий частий рух очима для зміни рядка відштовхує читача від читання.

Наявність «повітря» навколо тексту знижує напругу, завдяки чому стає набагато легше сконцентруватися на суті тексту. Не потрібно намагатися заповнити вікно браузера цілком. Те, що порожній простір виглядає краще, – не побічний ефект, а логічний наслідок функціонального дизайну.

Не менш важливим є питання **ширини колонки**. Варто переконатися, що ширина рядка не надто велика, а також, що залишилося достатньо порожнього місця ліворуч і праворуч від колонки тексту. Це допоможе очам легше перестрибувати з кінця одного рядка на початок іншого. Добре, коли рядок тексту є масштабованої ширини, тоді не треба налаштовувати ані розмір вікна браузера, ані розмір шрифту.

Звідси можемо вивести таке головне правило – 10–15 слів на рядок. При гумовій верстці з розміром шрифту 100% стандартна рекомендація для більшості варіантів роздільної здатності екрана – колонка шириною 50% від ширини вікна браузера.

Фахівці з читання вважають, що щільно стиснуті по вертикалі рядки зменшують швидкість читання, оскільки верхній і нижній рядки схоплюються оком одночасно. Це саме стосується і випадків, коли інтервал занадто великий. До речі, стандартний міжрядковий інтервал замалий. Якщо збільшити міжрядкову

відстань, то текст читатиметься набагато легше. Звичайна рекомендація – інтервал 140% від стандартного.

Щодо **колірного контрасту**, то неприпустимими є поєднання сірого тексту на світло-сірому фоні, на жовтому чи білому фоні, жовтий чи зелений текст на червоному тлі і таке інше.

Ще одне правило – ніякого тексту картинками. Користувачі бажають мати можливість шукати у тексті, копіювати текст та зберігати його, виділяти текст під час читання. Текст у вигляді картинки може виглядати гарно, але візуальна краса у мережі – не головне. Головне – це взаємодія та інформація, при цьому інформація має бути легкою для читання, використання, масштабування, цитування і пересилання.

4) **Критерії якості бізнес-сайтів**:

– середня кількість відвідувачів сайту на день;
– середня кількість сторінок сайту, які переглянули відвідувачі;
– середня вартість залучення одного відвідувача;
– середня кількість реальних покупців сайту у відсотковому співвідношенні до загальної кількості відвідувачів;
– термін окупності сайту;
– рентабельність сайту;
– час роботи сайту у штатному режимі без потреби його редизайну;
– середній час, який відвідувач провів на сайті (якщо до 1 хв – поганий результат, якщо понад п’ять – дуже добрий).

5) **Якість сайту з погляду професіоналів**. При оцінюванні концепції викладення інформації на ресурсі з професійної сторони сайт розглядають за різними критеріями сайтобудування.

5.1. **Код сторінки** перевіряють на:

– наявність помилок у коді HTML;
– працездатність посилань;
– відповідність HTML-коду сучасним стандартам мови;
– недоречність використання тегів у тексті сторінки;
– перевантаженість коду;
– відповідність таблиць стилів CSS сучасним стандартам;
– відповідність XML-елементів сучасним стандартам мови.

В результаті перевірки критерію «код сторінки» складаються поради експертів: як правильно оптимізувати сторінки на сайті власника за виявленими слабкими місцями сайту.

5.2. При оцінюванні **дизайну** перевіряють:

– якість навігації на сайті;
– зайві елементи у дизайні сторінок сайту;
– коректність відображення сторінки у різних браузерах.

Наприкінці перевірки зазначаються загальні зауваження та поради щодо покращення юзабіліті проекту та оптимізації графіки на сайті.

5.3. Для оцінювання критерію «**розкрутка (популяризація) сайту**» виявляються та аналізуються:

– статистичні показники сайту;
– видимість сайту в пошукових інтернет-системах (індексація сторінок, коректність файлу robots.txt);
– основні теґи сторінки (заголовок сторінки, теґи <h1>–<h6>, , <b>, <i>, <p> тощо), оптимізація яких впливає на ранжування сайту у пошуку;
– кількість і якість посилань (перевіряється якість наявних на сторінці посилань, формуються поради щодо оптимізації внутрішніх посилань для пошукових систем);
– густота ключових слів на сторінці, якість оптимізації сторінки за ключовими словами з формуванням порад щодо покращення текстів на сайті та перспективи сторінки бути популярною за цими ключовими словами, відповідно до виявленого рівня оптимізації сайту;
– ефективність розміщення банерної та контекстної реклами, формуються поради щодо оптимального налаштування;
– відповідність критеріям хорошого сайту різних пошукових систем;
– проблеми індексації сайту (склеювання сторінок, причини банів, коректність robots.txt, файлів Sitemaps, метатеґів, HTTP-заголовків, редиректів 301 та 302, помилок сервера 404, 505 тощо);
– сайти-конкуренти.

Варто пам’ятати, що «ринкові» показники сайту (позиція у пошуковій видачі, дані про ранжування та ступінь цитування, відвідуваність ресурсу) здебільшого необ’єктивні. Проте такий стан справ не дивує, оскільки нині ці характеристики цілковито залежать від майстерності оптимізаторів.

5.4. Для критерію «**технічна сторона**» перевіряються:

– якість місця розміщення сайту;
– швидкість завантаження веб-сторінок;
– доступність сервера.

На практиці часто аналіз сайту поширюється на комплекс критеріїв оцінки, до повного переліку яких входять і суб’єктивні, й об’єктивні. Аналіз сайту, який здійснюють за суб’єктивними факторами, не можна вважати хоча б напівоб’єктивним.

Наведемо приклад. Візьмемо деякий український ресурс, при цьому обов’язково виберемо гідний – тобто оригінальний, пізнавальний, грамотний і цікавий. Здійснимо повномасштабну оптимізацію, досягнемо його рейтингової видачі у перших рядках пошукової системи. Природно, що аналіз сайту дозволить нам поставити високу оцінку даному проекту. Пізніше зробимо гарний переклад його контенту на англійську мову та опублікуємо в англомовній зоні. Результат зовсім непередбачуваний, бо навіть, якщо тема нашого гіпотетичного проекту винятково актуальна, а контент пізнавальний і бездоганний, ймовірно знадобиться знову провести повну (або часткову) оптимізацію сайту, цього разу щодо кон’юнктури «місцевого» ринку.

Використання у методиці аналізу сайту суб'єктивних оцінювальних факторів робить суб'єктивним і сам результат. Причому, цей результат буде актуальним лише до появи більш оптимізованих мережевих ресурсів, які, до речі, зовсім не обов'язково кращі в площині інформаційної та художньої цінності. Щодо об'єктивності аналізу сайту, можна зробити висновок, що результати рейтингової видачі не є абсолютним показником, який дає змогу визначити цінність викладених матеріалів на сайті. А отже, ранжування ресурсів у сучасних пошукових системах настільки суб'єктивне, наскільки суб'єктивні критерії оцінювання. Враховуючи наявні методи та засоби, які використовують пошукові системи, об'єктивно здійснити аналіз сайту практично неможливо. Результат видачі сам по собі є суб'єктивним, оскільки залежить не від фактичної якості матеріалів проекту, а саме від майстерності його оптимізатора.

Набирають популярності ресурси, за допомогою яких можна перевірити якість свого сайту. Наприклад, на **http://cys.ru**[14] можна безкоштовно і за кілька секунд отримати оцінку свого сайту. Ось деякі спостереження. Сайт відомого дизайнера Джеффрі Зельдмана (www.zeldman.com) забрав 632.5 балів із тисячі можливих й отримав таку оцінку: «*Прекрасні скрипти! Відвідувачі дуже люблять, коли сайти наповнені скриптами! Середній рівень технічного виконання. Наявні недоліки в коді. Без фантазії, шаблонно забезпечена функціональність сторінки. Нормальний дизайн. Погана сумісність з різними браузерами, застосування нестандартних прийомів, які потребують плагінів або конкретних версій браузерів*». До сайту іншого визнаного дизайнера Дейва Ши (www.csszengarden.com) кібераналітик поставився лояльніше – 712 балів. Коментарі такі самі з додаванням речення: «Повна відсутність графіки – ознака лаконічності чи скупості»[15]. Вочевидь кібераналітик розрізняє табличну і блокову верстки, бачить скрипти, але зображення він бачить лише за наявності на сторінці теґу <img>. Фонових зображень він не бачить, але це не дивно, оскільки відповідно до стандарту XHTML1.0 Strict інформувати про це має таблиця стилів, а не HTML-код. Виникає питання: як власники цього ресурсу можуть пояснити принцип роботи кібераналітика?

Оцінку якості сайту можна здійснити лише на базі тих показників, які можна перевірити за визначеними критеріями. Для здійснення об'єктивної оцінки сайту коректно використовувати не розмиті фактори (наприклад, художній образ, графічне оформлення), а стійкі критерії аналізу якості: його інформативність, якість тематичного змісту (контенту), його структуру, навігацію та компонування, ілюстрованість – все те, що в решті решт для конкретного проекту визначається узагальнюючим терміном **«інформаційна та художня цінність»**.

Насамперед, наведена інформація про засоби і способи оцінки сайту адресована замовникам веб-проектів та авторам технічних умов (технічних за-

[14] Cybernetic analytic system. URL: http://cys.ru (дата звернення 28.06.2018).

[15] Огірко І., Паславська І., Пілат О. Інформаційна система оцінювання якості електронних видань. *Вісник Львівського університету. Серія економічна*. 2013. Вип. 49. С. 391–398.

вдань), веб-майстрам та дизайнерам, а також творцям тематичного і спеціалізованого контенту для веб-проектів.

Крім того, розглядаючи веб-проект як об'єкт для критики, треба повністю ігнорувати такі його відносні фактори як: рейтинг, цитування іншими, відвідуваність та авторитетність творця. Підхід до оцінювання сайту має бути цілком однаковим як для проекту, створеного професійним веб-дизайнером, так і для ресурсу, автором якого є недосвідчений початківець. Створення веб-проектів – це творчість.

Отже, розглянувши різні методи оцінювання сайтів[16], можемо сформулювати головну аксіому методології оцінки сайту так: повноцінний мережевий ресурс можна побудувати винятково за тими критеріями, за якими в результаті його будуть оцінювати.

Контрольні запитання

1) Перелічити етапи створення сайту.
2) Які розрізняють структури сайтів?
3) Коли доречно застосовувати лінійну структуру сайту?
4) У чому полягає специфіка ієрархічної структури сайтів?
5) Яка структура сайтів є найпоширенішою?
6) Що таке технічне завдання на сайт?
7) Що виконують на етапі розроблення дизайну сайту?
8) Що формується в результаті розроблення дизайну сайту?
9) Що перевіряють на етапі тестування сайту?
10) Навіщо виконувати адміністрування ресурсу?
11) Яка мета розкрутки сайту?
12) Які завдання адміністратора сайту?
13) Що включає просування сайту?
14) Яка мета оптимізації сайту?
15) У чому специфіка унікальності контенту?
16) Що таке рерайт готових статей?
17) Назвати важливі характеристики якості сайту.
18) Які вимоги до ширини рядка набору на веб-сторінці?
19) Навіщо потрібно навколо тексту залишати порожній простір, якщо це так?
20) Навіщо виникає потреба у створенні колонок тексту на веб-сторінці?
21) Які вимоги до ширини колонок на веб-сторінці?
22) Які поєднання кольорів є неприпустимими за правилами колірного контрасту?
23) Назвати критерії якості бізнес-сайтів.
24) За якими критеріями перевіряють код сторінки?
25) Що перевіряють при оцінюванні дизайну сайту?

[16] Огірко І., Паславська І., Пілат О. Інформаційна система оцінювання якості електронних видань. *Вісник Львівського університету. Серія економічна*. 2013. Вип. 49. С. 391–398.

Розділ 5
РОЗМІТКА ВЕБ-ДОКУМЕНТА

5.1. Види верстки

Нагадаємо:

– HTML – Hypertext Markup Language, мова гіпертекстової розмітки, призначена для створення логічної структури і наповнення веб-сторінок;

– CSS – Cascading Style Sheets, каскадні таблиці стилів, які формують візуальне подання вмісту веб-сторінки, тобто фізичну розмітку.

Саме тому можна умовно виділити логічну і візуальну структури веб-документа. Створення розмітки веб-сторінки (англ. Page layout) можна сміливо відносити до мистецтва: тут немає точних правил і єдиної відповіді на будь-яке питання, зате практично завжди є кілька варіантів вирішення певної проблеми. За принципами використання засобів розмітки HTML розрізняють: **логічну розмітку**, яка вказує на місце і функцію елементів у логічній структурі документа, і **візуальну (фізичну, презентаційну**), яка вказує як саме (в якому вигляді) подати елемент користувачу.

Наприклад, курсивний текст можна отримати як за допомогою теґу <i>, так і за допомогою теґу . У першому випадку курсив задається явно, а в другому на текст робиться логічний наголос, який зазвичай відображається курсивом. Інакше кажучи, при першому підході орієнтуються на зовнішній вигляд, а в другому – на логічне призначення. Специфікою другого підходу є незалежність верстки від використовуваного типу пристроїв і дизайну веб-сторінок. Якщо дотримуватися логічної розмітки, то можна використовувати один і той самий варіант верстки для екрана, друку і мобільних пристроїв, регулюючи зовнішній вигляд за допомогою окремих файлів CSS-стилів.[1]

Верстка веб-сторінки (англ. page-proof*)* – це процес формування веб-сторінки у вигляді HTML/CSS коду із попередньо створеного макета дизайну сайту, заздалегідь намальованого за допомогою графічних редакторів. Верстка сайту є кінцевим етапом розроблення сайту, створення структури сайту, яка визначатиме вигляд сайту в різних браузерах. Верстка сайту полягає у з'єднанні всіх його функціональних елементів і компонентів в єдине ціле.

Верстка веб-сторінок відрізняється від поліграфічної тим, що треба враховувати різницю відображення елементів у різних браузерах і різницю в розмірах робочого простору пристроїв. Процес складний і має творчу основу, жоден зі способів не є канонічним і прийнятим як основа. Усі підходи до верстки мають як переваги, так і недоліки. Хоча робота верстальника прихована від

[1] Вёрстка веб-страниц. URL: http://templates.hol.es (дата обращения 26.07.2018).

очей, саме вона забезпечує безперебійність при роботі на різних пристроях, а також швидкість завантаження кожної сторінки сайту.

Процес формування веб-документа можливий засобами веб-конструктора, WYSIWYG-редактора або через самостійне написання HTML-коду. Зазвичай верстальник отримує від дизайнера затверджений дизайн-макет сторінки, аналізує отриманий макет, розбиває його на горизонтальні лінії (смуги) – «поверхи». Далі, кожен «поверх» аналізується окремо і розбивається на прямокутні блоки – колонки. Далі відбувається рекурсивний процес верстки цих окремих рядків, а в них стовпців.

а **б** **в**

Рисунок 5.1 – Типові веб-макети[2]:
а) одноколонковий; б) двоколонковий; в) триколонковий

Після верстки сторінка перевіряється на крос-платформовість, при цьому перевіряється:[3]

– Чи однаково відображається сторінка у різних браузерах і на різних операційних системах?
– Чи відбувається критичне зміщення блоків, якщо змінювати (зменшувати / збільшувати) розмір шрифту в налаштуваннях браузера?
– Чи відбувається критичний зсув блоків, якщо відключити показ зображень у браузері?
– Чи суттєво впливає на цілісність сторінки роздільна здатність монітора?

Після цього критичні виправлення вносяться в документ, і перевірка повторюється із самого початку.

Одним з важливих обмежень у верстці є шрифти, оскільки гарнітурні набори у різних операційних системах і різних браузерах відрізняються. Варто зважати на те, що вибір гарнітури не обмежений нічим, але не знайшовши вказаного набору шрифтів браузер використовуватиме стандартні параметри.

Найпоширеніші види верстки:

1) таблична; 2) фреймова; 3) блокова; 4) семантична.

[2] Вёрстка веб-страниц. URL: https://seo-wikipedia.org/verstka-web-stranic (дата обращения 30.07.2018).
[3] Мержевич В. Основы верстки. URL: http://htmlbook.ru/content/osnovy-verstki (дата обращения 26.07.2018).

У кожного виду дизайну є свої переваги і недоліки, а вибір залежить від розв'язуваної задачі. При цьому може використовуватися і змішаний дизайн – деякі стовпці табличного дизайну задати у відсотках, а інші – у пікселях.

Таблична і фреймова верстки мають фіксовану (статичну) ширину і при зміненні розмірів екрана ведуть себе некоректно або просто масштабуються, якщо розміри були задані відсотками, і при невеликих розмірах екрана на телефонах текст стає замалим для читання. Це було однією з причин того, що нині в HTML5 ці види верстки вже не підтримуються.

Блокова і семантична види верстки є адаптивними, вони еластично підлаштовуються під розмір екрана і при цьому може відбуватися переміщення блоків з одного місця на інше та заміна блоків при певній роздільній здатності. Адаптивна верстка прийшла на зміну ідеї створення спеціальних мобільних версій сайту, які розміщують на окремих піддоменах (наприклад, m.wikipedia.org). При адаптивній верстці сайт відображатиметься при різних розмірах екрана так, як це найбільш зручно користувачеві, проте це вимагає ретельного опрацювання кількох макетів для різних розмірів екранів.

5.1.1. Таблична верстка

Таблична верстка характеризується тим, що модульна сітка (розмітка структури) створюється комірками (рядками і стовпцями) таблиці.

Позиціонування елементів сторінки здійснюється через розміщення у комірках таблиці.

В HTML виведення таблиці реалізується тегом <table>. Вкладений тег <tr> формує рядок таблиці. У нього вкладається тег <td>, який формує комірку (стовпець).

```
<table>
  <tr>
    <td> Комірка 1 </td>
    <td> Комірка 2 </td>
    <td> Комірка 3 </td>
  </tr>
  <tr>
    <td> Комірка 4 </td>
    <td> Комірка 5 </td>
    <td> Комірка 6 </td>
  </tr>
</table>
```

Комірка 1	Комірка 2	Комірка 3
Комірка 4	Комірка 5	Комірка 6

Кількість стовпців у кожному рядку має бути однаковою, але комірки

можна об’єднувати по вертикалі і горизонталі за допомогою атрибутів rowspan="число" і colspan="число" тегу <td>:

```
<table>
 <tr>
  <td rowspan="2"> Комірки 1 і 4 </td>
  <td colspan="2"> Комірки 2 і 3 </td>
 </tr>
 <tr>
  <td> Комірка 5 </td>
  <td> Комірка 6 </td>
 </tr>
</table>
```

Комірки 1 і 4	Комірки 2 і 3	
	Комірка 5	Комірка 6

В атрибутах комірки (<td></td>) або у відповідному селекторі CSS можна задавати: вирівнювання вмісту по горизонталі (align="") і вертикалі (valign=""), ширину (width="") і висоту (height="") комірки. Усталено комірки мають однакову ширину і висоту, тобто залежать одна від одної. У комірки можна вкладати нові таблиці та інші блокові елементи. Можна задавати різні властивості комірок (відступи, межі, поля) і вкладених елементів засобами CSS.

Приклад табличної верстки:

```
<html>
 <body>
  <table style="width: 500px; border:0px;">
   <tr>
    <td colspan="2" style="background-color:#85C2FF;">
     <h1 style="color: black;">Верхня частина сайту</h1>
    </td>
   </tr>
   <tr style="vertical-align: top;">
    <td style="background-color:#C2EBFF; width:100px; text-align:top;">
     <b>Меню</b><br> пункт 1<br>  пункт 2
     <br> пункт 3
    </td>
    <td style="background-color:#fff;
               height:200px; width:400px;
               text-align:top;">
     Місце для контенту</td>
   </tr>
   <tr>
    <td colspan="2" style="background-color:
        #85C2FF; text-align:center;">
       Copyright © 2018 www.it.inf.od.ua</td>
   </tr>
  </table>
 </body>
</html>
```

Перевагою табличної розмітки є простота і кросбраузерність, оскільки табличні теґи підтримуються усіма браузерами. До *недоліків* можна віднести надлишковий, захаращений, складний для розуміння код, недостатню гнучкість і можливість виникнення проблем відображення таких сторінок, оскільки уся веб-сторінка не буде відображена допоки не буде завантажений останній закривний теґ таблиці,[4] що є критичним при обриві зв'язку і повільному з'єднанні. Крім того, є проблема виникнення помилок верстки при множинній вкладеності таблиць, що характерно для досягнення певних ефектів на веб-сторінці. Зростання кількості таблиць збільшує розмір документів і знижує швидкість завантаження файлів. Крім того, використання таблиць для цілей оформлення не відповідає концепції семантичної верстки, яка закликає використовувати елементи (теґи) тільки відповідно до їхнього змісту, семантичного значення.[5]

5.1.2. Фреймова верстка

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

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

Теґ <frameset> визначає структуру фреймів на веб-сторінці і по суті є контейнером для фреймів. Особливістю веб-документа з фреймами є те, що в HTML-коді немає теґу-контейнера <body>, а теґ <frameset> йде відразу після розділу <head>, тобто теґ <frameset> замінює собою елемент <body> на веб-сторінці. Крім того, структурний HTML-документ (той, який визначає структуру фреймів) не може містити ані теґів форматування, ані будь-яких HTML-елементів.

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

[4] Браузери навмисно розцінюють таблицю як один об'єкт, а тому використання таблиці в якості каркаса для розміщення елементів веб-сторінки призводить до затримки виведення вмісту. Треба враховувати також і зростаючий обсяг веб-сторінок при активному використанні таблиць, особливо в разі їх вкладеності одна в одну. Все це призводить до того, що таблична верстка викликає непотрібні затримки виведення інформації у браузері.

[5] Вёрстка с помощью таблиц. URL: http://htmlbook.ru/samlayout/verstka-s-pomoshchyu-tablits (дата обращения 26.07.2018).

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

Наприклад, тег

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

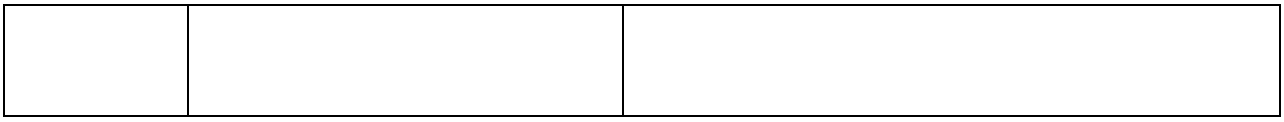
утворить два горизонтальних фрейми, один з яких (верхній) займатиме 30% робочої області вікна браузера, а другий (нижній) – 70% (загальна сума завжди має складати 100%).

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

Розглянемо запис значення атрибута у відносних одиницях:

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

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



Також можна для параметрів rows та cols задавати змішані значення:

```
<frameset rows="50, 50%, *, 3*">
```

Така структура має чотири горизонтальні фрейми: перший фіксований у 50 пікселів, другий – 50% від розміру вікна браузера, а два останніх фрейми поділять решту місця у співвідношенні 1/4 та 3/4.

Обов’язкового порядку для запису змішаних значень не існує, однак рекомендується найперше вказувати фіксовані значення (пікселі), тоді відсотки, а вже потім відносні одиниці.

Тег **<frame>** визначає властивості окремого фрейму у складі фреймової структури. Цей елемент повинен розташовуватися в контейнері <frameset>, а кількість тегів <frame> має відповідати кількості організованих у <frameset> областей. У кожну з таких областей завантажується самостійна веб-сторінка, яка визначається атрибутом src. Хоча обов’язкових атрибутів у тегу <frame> немає, рекомендується задавати кожному фрейму його ім’я через атрибут name. Це є важливим, якщо треба за посиланням з одного фрейму завантажити документ в інший. Наприклад:

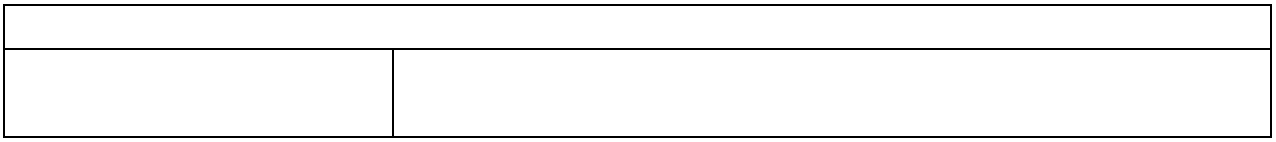
```
<frame src="frames/menu.html" marginwidth="5" marginheight="3">
```

Тут задається шлях до файлу, який завантажуватиметься у фрейм, та задаються відступи: горизонтальний – 5 пікселів, вертикальний – 3 пікселі. Задавати відступи

у фреймах слід обережно, пам'ятаючи про те, що такі самі (чи ще більші) відступи можуть бути задані і в HTML-коді документа вибраного фрейму (параметри leftmargin, rightmargin, topmargin, bottommargin, marginwidth та marginheight у тегу <body>).

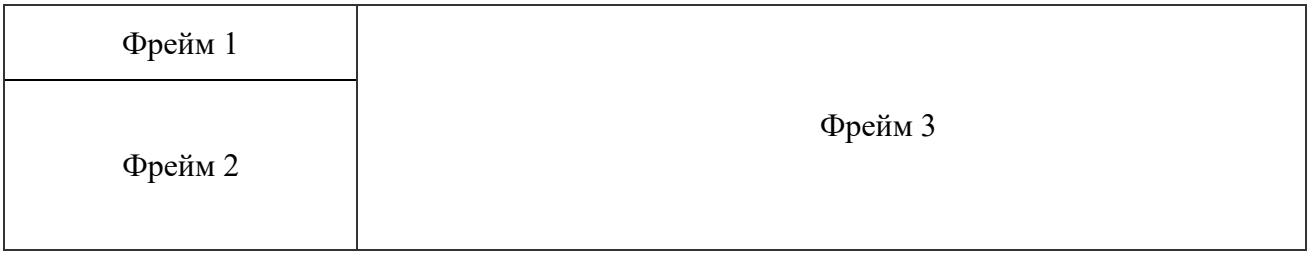
Отже, треба пам'ятати, що властивості документа, завантаженого у вікно фреймової структури, визначаються в HTML-коді саме документа, а не в межах конструкцій <frameset> або <frame>.

Приклад створення веб-сторінки з такою фреймовою структурою:



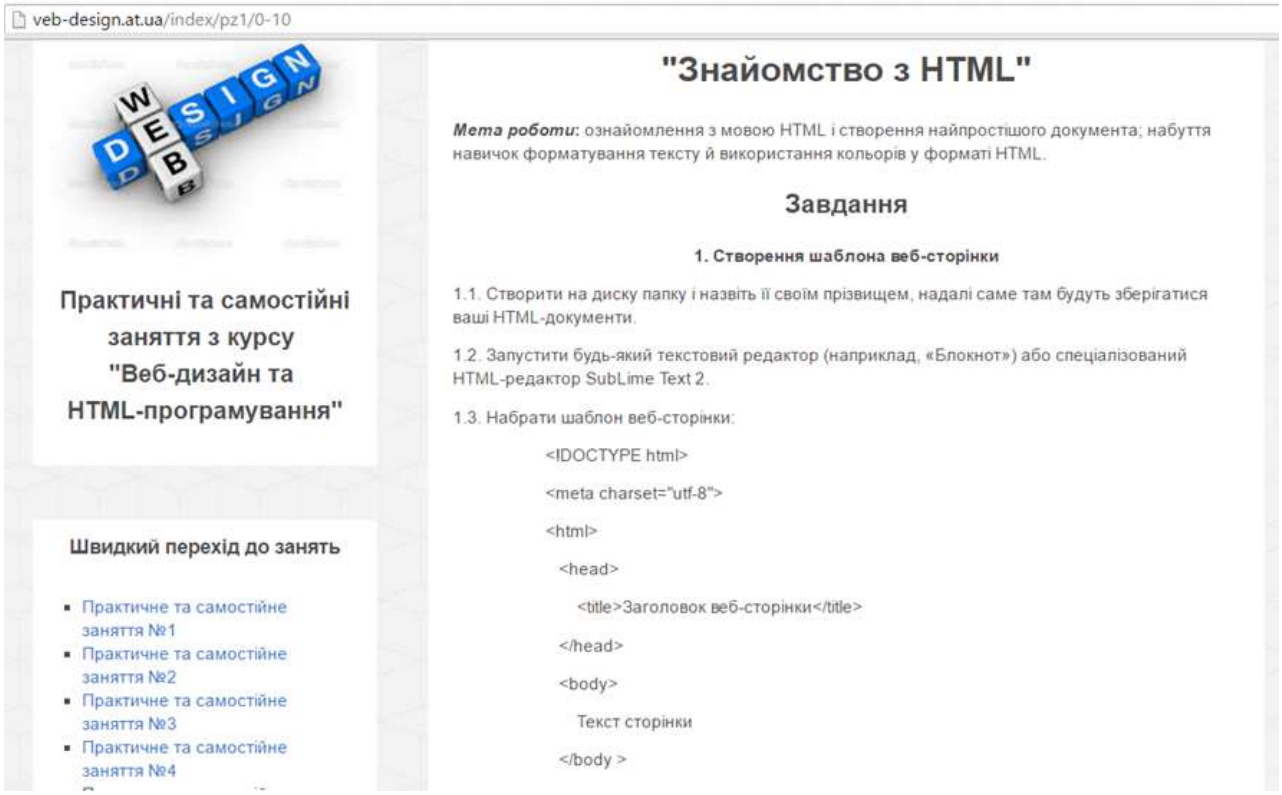
```
<!DOCTYPE html>
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Teґ FRAME</title>
  </head>
  <frameset rows="80,* " cols="*"> /*заголовок сайту */
    <frame src="nazva.html" name="topFrame" scrolling=no" noresize>
    <frameset cols="30%,*"> /*основна частина з двох фреймів (стовпців) */
      <frame src="spisok.html" name="leftFrame" scrolling="no" noresize>
      <frame src="lb1.html" name="mainFrame">
    </frameset>
  </frameset>
</html>
```

Приклад створення веб-сторінки з такою фреймовою структурою:

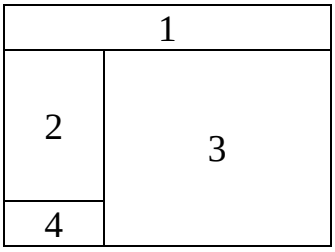



```
 <frameset cols="80,*">
   <frameset rows="*,2* " >
     <frame src="frame1.html" name="Фрейм 1">
     <frame src="frame2.html" name="Фрейм 2">
   </frameset>
   <frame src="frame3.html" name="Фрейм 3">
</frameset>
```

Приклад можливого наповнення сайту з такою структурою:



Як приклад створимо сайт із фреймовою структурою: 1 – назва сайту; 2 – навігаційна панель зі списком назв веб-сторінок; 3 – основна частина, в яку завантажуватиметься вміст створених нами на попередніх заняттях веб-сторінок про Одесу; 4 – дані про автора.



Після заповнення фреймів даними він набуде вигляду:

Для цього треба створити HTML-документ з ім'ям index.html як головний файл з фреймовою структурою, а також три HTML-файли для заповнення фреймів 1, 2 та 4: nazva.html (назва сайту), navig.html (навігаційна панель), author.html (дані про автора).

Лістинг файлу index.html:

```
<html>
  <head>
     <meta charset="utf-8">
     <title> Одеса найкраще місто в світі </title>
     <link rel="stylesheet"  type="text/css"  href="style.css">
     <meta name="keywords" content="Одеса найкраще місто в світі">
     <meta name="description" content="Одеса найкраще місто в світі">
  </head>
  <frameset rows="130,*"  cols="*"  >       /* заголовок сайту */
    <frame src="nazva.html"  name="topFrame"  scrolling="no"  noresize>
    <frameset cols="20%,*">                 /* основна частина з двох фреймів (стовпців) */
      <frameset rows="90%,*"  >
        <frame src="navig.html"  name="leftFrame" scrolling="no" noresize >
        <frame src="author.html" name="leftFrame" scrolling="no" noresize>
      </frameset>
      <frame src="lb2.html" name="mainFrame">
    </frameset>
  </frameset>
</html>
```

Лістинг верхнього фрейму nazva.html:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Сайт про Одесу </title>
    <style type="text/css">
       h1 { text-align: center;
            font-size: 300%;
            font-family: fantasy;
            color: #0000ff;
          }
     .gradient {    /* градієнтна заливка */
                 background: -webkit-linear-gradient(top, #00c 0%,#fff 50%,#00f 100%,#fff 50%);
         }
    </style>
  </head>
  <body background="photo/fon.gif">
    <h1 class="gradient"><img src="photo/ship.gif" height="70px">
```

```
        ОДЕСА – найкраще місто в світі</h1>
  </body>
</html>
```

Лістинг фрейму navig.html:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Панель  навігації</title>
     <style type="text/css">
      p { font-size: 120%;
          margin: 20px;
          font-family: sans-serif;
          color: #0000ff;
        }
      .gradient { background: -webkit-linear-gradient(left, #55c 0%,#fff 50%,#77f 100%,#fff 50%); }
     </style>
  </head>
  <body class="gradient">
   <p> <a href=''lb2.html'' target=''mainFrame''>Перлина у Чорного моря</a></p>
   <p> <a href=''lb3.html'' target=''mainFrame''>Довідково-статистичні відомості про Одесу </a></p>
   <p> <a href=''lb4.html'' target=''mainFrame''>Видатні місця Одеси</a></p>
  </body>
</html>
```

Лістинг фрейму author.html:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Сайт про Одесу </title>
  </head>
  <body background="photo/fon.gif">
    <p>  &#1693айченко Маргарита</p>    /* &#169 – код символу авторського знака */
  </body>
</html>
```

При натисканні інших посилань на навігаційній панелі у третьому фреймі завантажуватиметься відповідний html-файл:

Розглянемо основні переваги та недоліки застосування фреймових структур. Почнемо, як водиться, з **позитивних сторін**:

– фрейми дозволяють економити на обсязі даних, які передаються користувачеві, оскільки після активізації посилання змінюється тільки один з них;

– фрейми помітно полегшують навігацію по електронних документах завдяки можливості переходу з інших посилань у межах інтернет-ресурсу;

– можливість роботи відразу з декількома інформаційними блоками у межах одного вікна дозволяє економити час;

– використання правил опису фреймових структур дозволяє розробнику HTML-документів як завгодно варіювати розміри полів фреймів, що надає широкий спектр можливостей для візуалізації просторових об’єктів;

– сторінки поділені на фрейми мають менше коду, внаслідок відсутності повторюваних частин, що не перезавантажуються.

А тепер декілька **недоліків** фреймів, через які **фреймова верстка не підтримується в HTML5**:

– фрейми погано індексуються пошуковими системами, оскільки на сторінках з вмістом немає посилань на інші сторінки сайту і навпаки, на навігаційній сторінці немає ніякого вмісту, що призводить до індексування не батьківського фрейму, як потрібно, а одного з його складових;

– пошукові системи погано працюють з фреймовою структурою, оскільки на сторінках, які містять контент, зазвичай немає посилань на інші документи. Перехід з пошукової сторінки відбувається на одну сторінку без завантаження інших фреймів;

– велике число фреймів вимагає для браузера виділення дещо більшої пам’яті, ніж звичайно;

– внутрішні сторінки не можна додати в закладки, оскільки браузер не показує змінення в адресному рядку, відображаючи завжди тільки адресу сайту;

– фрейми приховують адресу сторінки, на якій перебуває відвідувач, і завжди показують тільки адресу сайту. З цієї причини вподобану сторінку неможливо помістити у розділ браузера *Вибране*;

– деякі браузери при спробі перейти назад до попереднього документа, який щойно був відображений, повертаються на початок веб-сайту. Те саме відбувається, якщо спробувати відновити сторінку з фреймовою структурою;

– сумісність фреймової верстки між браузерами суперечлива. Одні й ті самі параметри інтерпретуються браузерами по-різному.

5.1.3. Блокова верстка

Розмітка сторінки у блоковій верстці реалізується універсальним блоковим елементом , який утворює прямокутний гнучкий блок (контейнер, шар). Блоки є структурними елементами, які можна розміщувати на веб-сторінці шляхом накладення їх один на одний з точністю до пікселя. У можна вкладати рядкові і блокові елементи, зокрема . Положення блока задається двома координатами щодо будь-якого кута вікна браузера, батьківського елемента або документа. Таблиці використовуються тільки для виведення табличних даних.

Перед блоковою верстkою сторінок сайту бажано розробити їхні каркасні структури для відображення на різних пристроях з різними розмірами екрана: кількість колонок, розташування основного контенту, вміст хедера і футера тощо (рис. 5.2). Після визначення розташування елементів веб-сторінки можна буде швидше створити HTML-каркас, який стилізувати через CSS.

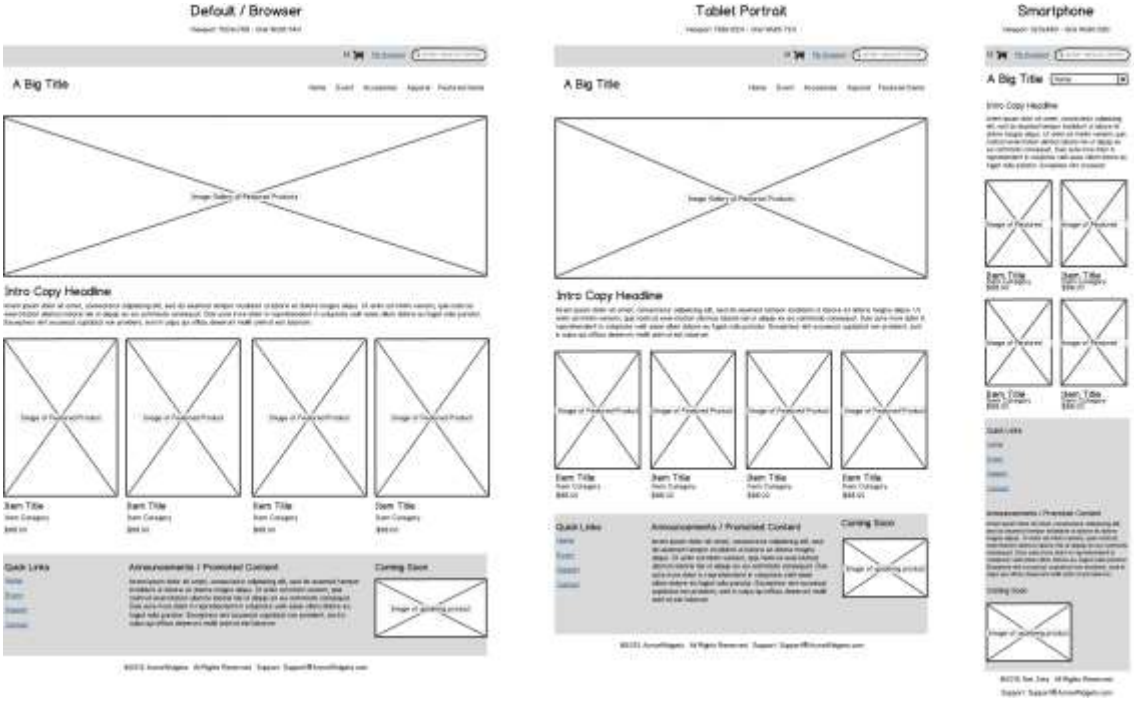
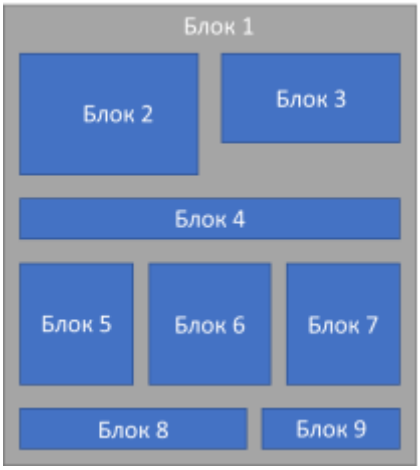


Рисунок 5.2 – Каркаси вигляду веб-сторінки на різних пристоях[6]

[6] Создание разметки: основные правила. URL: https://idg.net.ua/blog/uchebnik-css/razmetka-css/osnovnye-pravila (дата обращения 30.07.2018).



Блокова верстка має компактний код, при цьому HTML-код містить тільки теґи розмітки і логічного форматування, а все візуальне оформлення виноситься у CSS. Блокам присвоюється клас зі стилів CSS. Використання стильових таблиць дозволяє нескладними методами утворити компактний і ефективний код. У CSS задається позиціонування блока, вирівнювання відносно інших елементів, поля, відступи, межі, фон, вирівнювання і стилі вмісту. Можна виводити блоки з потоку документа засобами позиціонування в CSS. Скрипти дозволяють змінювати параметри блока динамічно. Це дозволяє створювати на сторінці різні ефекти: випадне меню, динамічні банери, рухомі вікна тощо.

Приклад простої блокової верстки:

```
<html>
 <body>
   <div id="container" style="width:500px">
    <div id="header" style="background-color:#A3FFC2;">
      <h1 style="margin-bottom:0;">Верхня частина сайту </h1>
    </div>
    <div id="menu" style="background-color:#FFFFB8; height:200px; width:100px; float:left;">
     <b>Меню</b><br>
     Пункт 1<br> Пункт 2<br> Пункт 3</div>
    <div id="content" style="background-color: #fff; height:200px; width:400px; float:left;">
     Місце для контенту
    </div>
    <div id="footer" style="background-color:#A3FFC2; clear:both; text-align:center;">
     Copyright © 2018 www.it.inf.od.ua
    </div>
  </div>
</body>
</html>
```

Верхня частина сайту

Меню пункт 1 пункт 2 пункт 3	Місце для контенту

Copyright © 2018 www.it.inf.od.ua

Блоки, порівняно з таблицями, відображаються швидше, що досягається за рахунок компактного коду і того, що відображення вмісту шару відбувається в міру його завантаження. Щоправда, це може призвести до «стрибків» елементів

сторінки при їх підвантаженні.[7]

На жаль, браузери можуть дещо по-різному відображати певні елементи. Використання блокової верстки потребує знань і навиків.

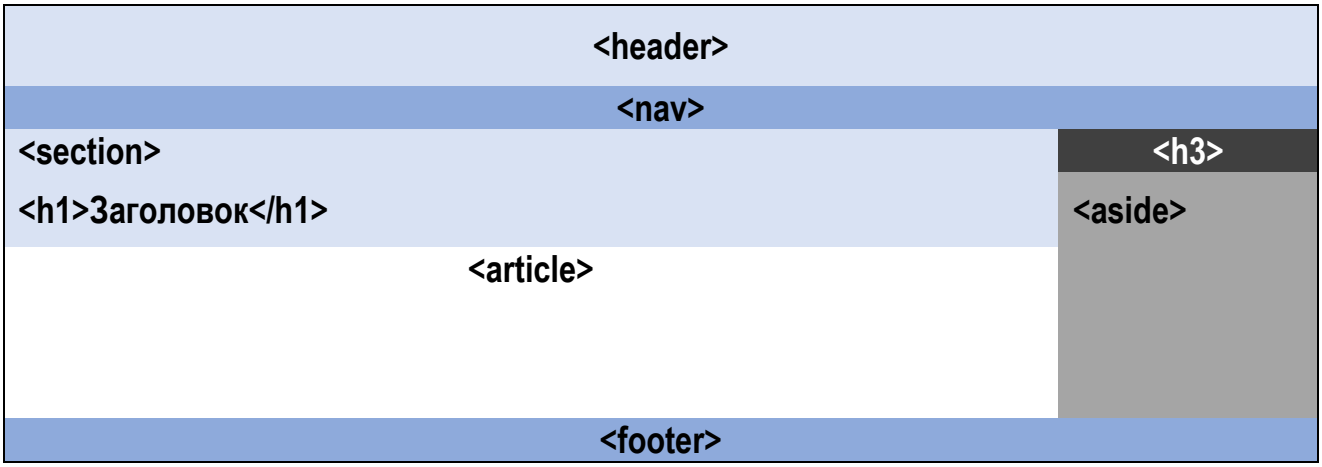
5.1.4. Семантична верстка

Семантичною називають верстку, що використовує для структурування html-документів теґи, які поділяють код на логічні блоки (явно показують їхню роль і зміст у структурі веб-сторінки).

Цей вид верстки став доступний з появою HTML5. Його нові елементи відіграють значну роль у створенні явної логічної структури документа.

Семантична верстка відрізняється від блокової лише елементами, використовуваними для структуризації сторінки, але на відміну від семантичних теґів, які використовуються у семантичній верстці, теґ , що є основним використовуваним елементом у блокової верстці, є лише контейнером для угруповання контенту.

Наведемо приклад можливого макета веб-сторінки із семантичною версткою:



Розглянемо елементи, які були використані в макеті. Кожний розділ може мати свої власні теґи <header> і <footer>, які матимуть свої власні заголовки.

Теґ <section> визначає розділ сторінки, в якому розміщується логічно пов'язана інформація, яка зазвичай має свій заголовок.

Теґ <header> визначає верхню частину сторінки або елемента. Він може містити заголовки, але це не є обов'язковим. Є лише одне обмеження використання <header>: він не повинен містити сам себе й елемент <footer>.

Теґ <nav> призначений для створення меню навігації на веб-сторінці. Він з'явився тільки в HTML5. <nav> призначений для розміщення гіперпосилань на інші сторінки та інші частини документа.

[7] Блочная вёрстка. URL: http://htmlbook.ru/samlayout/blochnaya-verstka (дата обращения 26.07.2018).

Теґ <article> використовується для поділу розділу сторінки на логічні блоки; блок має бути пов'язаний із загальним розділом, але мати свій власний сенс і бути окремою логічною одиницею сторінки.

Теґ <aside> створює невелику область на сторінці, збоку від загального вмісту. <aside> потрібен для винесення інформації в окремий блок, його вміст зазвичай не стосується основної теми сторінки, а лише надає додаткову інформацію до якоїсь її частини.

Теґ <footer> визначає нижній колонтитул сторінки або елемента, зазвичай містить інформацію про розділ, як от: авторські дані, посилання на відповідні документи тощо.

Хоча нові HTML-елементи можна використовувати, вони не завжди розпізнаються старими версіями браузерів. У старих браузерах нові HTML-теґи, які використовуються для семантичної розмітки, не підтримуються належним чином і можуть бути відображені як рядкові елементи. Для усунення цієї проблеми треба явно зазначати, що вони є блоковими елементами:

```
<style>
  section, header, nav, article, aside, footer { display: block; }
</style>
```

У нових версіях браузерів цей код не матиме ніякого впливу на відображення елементів на веб-сторінці, але при цьому він буде запасним варіантом правильного відображення елементів у старих версіях браузерів, які, як показує практика, можуть використовуватися користувачами ще досить довго.[8]

На жаль, такий CSS-код не допоможе браузеру Internet Explorer різних версій коректно відображати семантичні елементи. Для коректного оброблення нових семантичних елементів у всіх версіях IE доведеться вдатися до допомоги JavaScript. Для цього доречно додати такий код в елемент <head>:

```
<script>
  document.createElement("section");
  document.createElement("header");
  document.createElement("nav");
  document.createElement("article");
  document.createElement("aside");
  document.createElement("footer");
</script>
```

Несемантичне використання HTML плутає користувачів і агентів користувача[9], які намагаються розпізнати структуру вмісту сторінки, а це є проблемою доступу.

---

[8] HTML: Семантическая верстка. URL: https://puzzleweb.ru/html/16_vidi_verstki3.php (дата обращения 26.07.2018).

[9] **Агент користувача** – це програмне забезпечення (програмний агент), який діє від імені користувача. Одне з поширених термінів означає, що веб-браузер повідомляє дані про сам браузер, використовувану роздільну здатність (розміри екрана) та операційну систему. Це дозволяє веб-сайту налаштовувати вміст під можливості конкретного пристрою, але при цьому породжує проблеми конфіденційності.

Як було зазначено, блокова і семантична види верстки є адаптивними («гумовими», змінюваними), вони еластично підлаштовуються під розмір екрана так, як це найбільш зручно користувачеві, і при цьому може відбуватися переміщення блоків з одного місця на інше та заміна блоків при певній роздільній здатності. В той час, як таблична і фреймова верстки є «жорсткими», оскільки для всіх елементів веб-сторінки задають фіксовані розміри, незалежно від розміру екрана користувача і встановленої роздільної здатності. Саме тому в HTML5 таблична і фреймова верстки вже не підтримуються, а рекомендованими є адаптивний веб-дизайн (блокова і семантична види верстки) сайтів.

5.2. Концепція адаптивного веб-дизайну

Адаптивний веб-дизайн (англ. Adaptive Web Design, AWD) – дизайн веб-сторінок, який забезпечує оптимальне відображення та взаємодію сайту з користувачем незалежно від роздільної здатності та формату пристрою, з якого здійснюється перегляд сторінки.

Метою адаптивного веб-дизайну є практичне відображення інформації та зручна навігація на всіх пристроях із доступом до інтернету (від стаціонарних ПК до мобільних телефонів). За технологією адаптивного веб-дизайну не потрібно створювати окремі версії веб-сайту. Один сайт може працювати на всьому спектрі пристроїв.[10] Для цього заготовлюється кілька стильових правил або файлів під різний діапазон роздільної здатності екранів, вибір правил відбувається через скрипти або CSS3, які і визначають потрібну для цього інформацію про користувача.[11]

Переваги адаптивного веб-дизайну очевидні, а недоліками є те, що це найскладніший тип з усіх макетів, адже, по суті, замість одного потрібно зробити кілька макетів зі своєю графікою і CSS, а також прописати механізм визначення роздільної здатності монітора або ширини вікна браузера. Тому за рахунок універсальності макет складно перевіряти на різні умови, які можливі у користувачів.

А втім, популярність адаптивного веб-дизайну зростає з кожним днем, оскільки кількість мобільного трафіка є переважною в обсягах всього інтернет-трафіка.[12] Ця тенденція настільки поширена, що Google з 21 квітня 2015 року запустив у своїй пошуковій системі алгоритм оцінки сайту на відповідність принципам «дружного» до мобільних пристроїв інтерфейсу.[13] Від цього показника залежить те, як високо сторінка буде подана у результатах мобільного по-

[10] Marcotte E. Responsive Web design. URL: http://www.alistapart.com/articles/responsive-web-design (accessed 26.07.2018).

[11] Типовые макеты. URL: http://htmlbook.ru/samlayout/tipovye-makety (дата обращения 27.07.2018).

[12] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper. – January 30, 2015. URL: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html (accessed 26.07.2018).

[13] Official Google Webmaster Central Blog: Rolling out the mobile-friendly update. URL: https://webmasters.googleblog.com/2015/04/rolling-out-mobile-friendly-update.html (accessed 26.07.2018).

шуку, а отже, ця оцінка частково діє як штраф для сайтів, які не відповідають стандартам інтерфейсу для мобільних пристроїв.

Нині адаптивний веб-дизайн є «золотим перетином» і найкращим рішенням сучасного сайтобудування. Суть його полягає у створенні декількох варіацій розмітки з корекцією ширини контейнера залежно від розмірів екрана. Наприклад, для настільних комп'ютерів можна встановити ширину сторінки в 960 пікселів, для планшетів – ширину цієї самої сторінки в 700 пікселів, а для всіх смартфонів взагалі поставити непостійну ширину у відсотках.[14] В такий спосіб можна перетворити, наприклад, великий триколонковий макет на один вузький стовпчик, якщо сайт переглядається з невеликого екрана. Стандарно при адаптивному компонуванні розглядають шість ширин екрана: 320, 480, 760, 960, 1200 і 1600.[15]

У рамках адаптивного дизайну існує поняття респонзивного (чутливого, чуйного, озивного, чулого, пристосовчого)[16] дизайну (англ. Responsive web-design, RWD). Це новий підхід,[17] заснований на CSS3, з більш глибоким рівнем специфікації для кожного пристрою у таблиці стилів за рахунок покращеного використання @media запитів CSS, використання пропорційно змінюваних сіток, гнучких зображень та CSS3 медіа. Від початку розробниками створюється єдина версія застосунка, яка буде динамічно змінюватися, залежно від розмірів вікна браузера.[18]

Основні принципи[19] чуйного дизайну:

– розташування всіх елементів у рамках модульної сітки;

– гнучка структура сторінки – «гумовий макет» (англ. fluid grid),[20] пропорції і розміри елементів якого задаються у відсотках;

– усі елементи верстки та медіа-файли (зокрема зображення) є гнучкими (англ. flexible), розміри яких залежать від розміру екрана. При зменшенні ширини такої сторінки весь контент плавно стискатиметься, структурні елементи зменшуватимуться один відносно одного. Так, наприклад, якщо веб-сайт мав триколонкову структуру, то на вузькому екрані він матиме дві або одну колонку контенту;

– робота з медіа запитами (англ. media queries) – модулем CSS3, який дозволяє задавати різні стилі (або навіть таблиці стилів) для виведення різних те-

---

[14] Ширина веб-страницы. *Современный учебник CSS*. URL: https://idg.net.ua/blog/uchebnik-css/razmetka-css/shirina-web-stranitsy (дата обращения 27.07.2018).

[15] Designing for 10000 Screens: 4 Layout Tips for Responsive Web Design. URL: http://blog.venturepact.com/designing-for-10000-screens-4-layout-tips-for-responsive-web-design (accessed 27.07.2018).

[16] Перекладаємо слово респонсивний. URL: https://slovotvir.org.ua/words/responsyvnyi (дата звернення 26.07.2018).

[17] 9 основних принципів чуйного веб-дизайну. URL: http://it-ua.info/news/2014/11/14/9-osnovnih-principv-chuynogo-veb-dizaynu.html (дата звернення 27.07.2018).

[18] Респонсивный или адаптивный: какой дизайн выбрать? URL: https://umbrellait.com/ru/responsive-vs-adaptive-web-design (дата обращения 27.07.2018).

[19] Адаптивный и отзывчивый веб-дизайн. URL: https://itkeys.org/responsive-and-adaptive-design/ (дата обращения 27.07.2018).

[20] Responsive или Adaptive Web Design. URL: https://raskrutka.com.ua/blog/responsive-ili-adaptive-web-design/ (accessed 27.07.2018).

матичних блоків у різному порядку і різних пропорціях, залежно від роздільної здатності екрана, його розмірів та інших характеристик.

По суті, респонзивний – це один з різновидів[21] адаптивного веб-дизайну, який є лише «гумовим» макетом, що трансформується залежно від ширини екрана.[22] Розрізняють стандартний (standard), респонзивний (responsive), масштабований (scaled) і гнучко масштабований (flux & zoom) макети.[23]

5.3. Розмітка флексбоксами

5.3.1. Правила розмітки флексбоксами

Флексбокс або **CSS3 Flexible Box Layout** (від англ. flexbox – гнучкий блок) – це відносно новий засіб розмітки, створений для впорядкування елементів на сторінці з метою гнучкої адаптивності сторінки під різні розміри екранів для різних пристроїв. Модуль Flexible Box з жовтня 2017 року рекомендується корпорацією W3C як засіб оптимального компонування у дизайні користувацького інтерфейсу.[24]

У моделі розмітки флексбоксами гнучкий контейнер і його елементи можуть бути викладені у будь-якому напрямку і можуть змінювати свої розміри: рости, щоб заповнити невикористаний простір, або зменшуватись, щоб уникнути переповнення. Можна вибирати потрібний порядок елементів на сторінці та впорядкувати їх, легко керувати горизонтальним і вертикальним вирівнюванням вмісту за допомогою однієї властивості і навіть додавати будь-яку кількість стовпців у розмітку сторінки свого сайту.

Головною концепцією флексбоксів є можливість змінення висоти та / або ширини його елементів, щоб найкраще заповнити простір будь-якого екрана. Така адаптація важлива при зміненні орієнтації (з вертикальної на горизонтальну чи навпаки) та розмірів екрана.

На сьогодні флексбокси підтримується практично усіма сучасними браузерами, зокрема на Android та iOS.

Основні переваги флексбоксів:

1. Створювані флексблоки (елементи) є «гумовими», тобто вони можуть стискатися і розтягуватися за заданими правилами, займаючи потрібний простір.

2. Вирівнювання по вертикалі і горизонталі щодо базової лінії тексту працює бездоганно.

---

[21] Adaptive vs. Responsive Design. URL: https://www.interaction-design.org/literature/article/adaptive-vs-responsive-design (accessed 27.07.2018).

[22] Разница между адаптивным (Adaptive) и отзывчивым (Responsive) дизайном. URL: https://www.templatemonster.com/help/ru/difference-between-adaptive-design-and-responsive-design.html (дата обращения 27.07.2018).

[23] Adaptive web design. URL: https://www.jdvorak.eu/adaptive-web-design/ (accessed 27.07.2018).

[24] CSS Flexible Box Layout Module Level 1 // W3C Candidate Recommendation. URL: https://www.w3.org/TR/css-flexbox-1 (accessed 26.07.2018).

3. Розташування елементів в HTML-коді не має вирішального значення. Його можна змінювати засобами CSS. Це суттєво для деяких аспектів респонзивної (чуйної) верстки.

4. Елементи можуть автоматично вибудовуватися у кілька рядів чи то стовпців, займаючи весь наданий простір.

5. Багато мов у світі використовують написання справа наліво rtl (right-to-left), на відміну від звичного нам ltr (left-to-right). Flexbox адаптований і для цього. У ньому є поняття початку і кінця, а не «справа» чи «зліва». Тобто у браузерах з локаллю rtl всі елементи автоматично розташовуватимуться у реверсному порядку.

6. Синтаксис CSS-правил щодо флексбоксів доволі простий.

Пошук нових підходів[25] щодо оптимального розроблення макетів вебсторінок триває. Є певні баги флексбоксів, наприклад, деякі елементи HTML не можуть бути гнучкими контейнерами (<fieldset> і <button> не працюють як контейнери з display: flex декларацією[26]), але для кожного з багів є обхідні шляхи вирішення.

Основна термінологія та правила розмітки флексбоксами:

– Контейнер стає флексбоксом, якщо для нього задати властивість display зі значенням flex[27].

– Контейнер (flexbox) вміщує в собі елементи (flex items) (рис. 5.3).

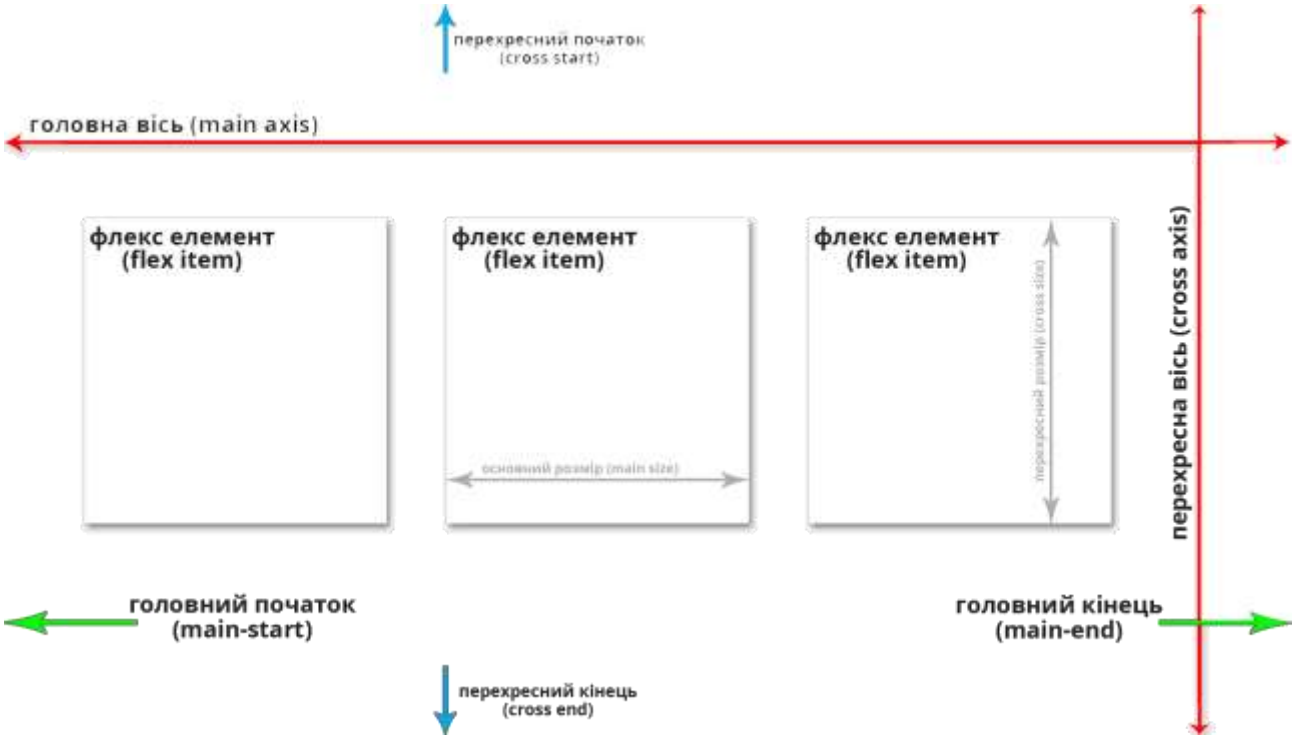


Рисунок 5.3 – Графічний опис важливих концепцій для flexbox у CSS3[28]

[25] Why Flexboxes Aren't Good for Page Layout. URL: https://www.xanthir.com/blog/b4580 (accessed 26.07.2018).
[26] Flexbugs. URL: https://github.com/philipwalton/flexbugs#flexbug-9 (accessed 26.07.2018).
[27] Властивість flex задає гнучкість боку і вміщує в собі значення трьох властивостей: flex-grow, flex-shrink та flex-basis.
[28] Використання Flexbox в CSS3 для адаптивного дизайну. URL: https://sebweo.com/vikoristannya-flexbox-v-css3-dlya-adaptivnogo-dizajnu (дата звернення 26.07.2018).

– Головна вісь (main axis) – це вісь, уздовж якої вирівнюються елементи (може бути як горизонтальною, так і вертикальною). Властивість flex-direction встановлює main axis. Властивість justify-content визначає, як елементи розташовані вздовж головної осі на поточній лінії.

– Перехресна вісь (cross axis) (рис. 5.3) – вісь, перпендикулярна до головної осі. Властивість align-items визначає правила розташування елементів уздовж cross axis на поточній лінії. Властивість align-self задає вирівнювання елементів уздовж cross axis і змінює значення, задані в align-items.

– Головний початок / головний кінець (main start / main end**)** і перехресний початок / перехресний кінець (cross start / cross end**)** (рис. 5.3) – це сторони контейнера, що визначають початок і закінчення потоку елементів. Елементи (flex items) розміщуються у контейнері по головній і перехресній осях у заданому напрямі: зліва-направо (усталено), справа-наліво тощо, починаючи з боку головного початку (main-start) і до main-end.

– Перехресний початок і кінець (cross start / end): флекс-лінії заповнюються items (елементами) і розміщуються у контейнері, починаючи з боку перехресного початку контейнера і до перехресного кінця.

– Властивість основного розміру (main size) – ширина або висота елемента, залежно від вибору головної осі, яка і є основним розміром елемента.

– Властивість перехресного розміру (cross size) – висота або ширина флекс-елемента, залежно від вибору поперечної осі.

– Еквівалентами для розмірів (height і width) елементів є main size та cross size, які відповідно стосуються main axis і cross axis контейнера.

– Елементи можуть бути розміщені на одній чи кількох лініях відповідно до flex-wrap властивості, яка контролює напрямок перехресних ліній і напрямок, в якому складаються нові лінії.

Як приклад створимо всередині body HTML-файлу батьківський контейнер (флексбокс) класу main з трьома елементами класу blok з невеликою інформацією, і блоком з основним контентом (article) під ними:

```
<div class="main">
  <div class="blok"><p>Блок інформації №1</p></div>
  <div class="blok"><p>Блок інформації №2</p></div>
  <div class="blok"><p>Блок інформації №3</p></div>
  <article><p>Тут основний контент.</p></article>
</div>
```

Налаштування флексбоксу та його елементів задається в CSS (або в CSS-файлі, або в тегу <style></style> у розділі head).

Щоб створити flexbox, потрібно визначити для контейнера властивість display зі значенням flex. Крім того, щоб параметри flex спрацювали, треба задати розміри контейнера, наприклад, окремо ширину і висоту: width: 75%; height:

450px; або хоча б один максимально допустимий розмір max-width: 800px (тоді висота залежатиме від наповнення контентом).

Наведемо вигляд початкового налаштування стилів контейнера main та його трьох елементів класу blok:

```
.main {  /* головний контейнер, в якому розміщуватимуться елементи */
    display: flex;              /* задати цей блок гнучким – флексбоксом */
    width: 75%;  height: 450px;
}
.blok {
   order: 1;
    flex: 1 1 30%;
}
article {
   order: 2;
   flex: 1 1 auto;
}
```

Щоб розібратися з використаними тут та іншими флекс-властивостями, розглянемо їх докладніше.

5.3.2. Основні властивості flex-контейнера

Властивість display зі значенням flex або inline-flex задає батьківський контейнер гнучким.

```
.flex-container {
    display: -webkit-flex;      /* браузерний префікс */
    display: flex;              /* відображати контейнер як блоковий елемент */
}
.flex-container {
    display: -webkit-inline-flex;  /*  браузерний префікс */
    display: inline-flex;          /*  відображати контейнер як рядковий елемент */
}
```

Після встановлення наведених значень властивості display кожен дочірній елемент автоматично стає flex-елементом, шикуючись в ряд (уздовж головної осі) колонками однакової висоти по висоті блока-контейнера. При цьому блокові і дочірні елементи поводяться однаково, тобто ширина блоків дорівнює ширині їхнього вмісту з урахуванням внутрішніх полів і відступів елемента.

На сьогодні підтримка флексбоксів браузерами складає 86,3% без використання префіксів і 97,7% з ними[29]. Для підтримки флексбоксів у старих версіях популярних браузерів, зокрема, Internet Explorer (IE) і Safari, треба задати браузерні префікси[30]:

```
.main {
  display:-webkit-box;          /* OLD – iOS 6-, Safari 3.1-6 */
```

[29] CSS Flexible Box Layout Module. URL: https://caniuse.com/#feat=flexbox (accessed 26.07.2018).

[30] Using Flexbox: Mixing Old and New for the Best Browser Support. URL: https://css-tricks.com/using-flexbox (accessed 26.07.2018).

```
  display:-webkit-flex;          /* NEW – Chrome */
  display: -moz-box;             /* OLD – Firefox 19- (buggy but mostly works) */
  display:-ms-flexbox;           /* TWEENER – IE 10 */
  display: flex;                 /* NEW, Spec – Opera 12.1, Firefox 20+ */
 }
 .main-content {
  -webkit-box-ordinal-group: 1;  /* OLD – iOS 6-, Safari 3.1-6 */
  -moz-box-ordinal-group: 1;     /* OLD – Firefox 19- */
  -ms-flex-order: 1;             /* TWEENER – IE 10 */
  -webkit-order: 1;              /* NEW – Chrome */
  order: 1;                      /* NEW, Spec – Opera 12.1, Firefox 20+ */
 }
```

З цими та іншими браузерними префіксами докладніше можна ознайомитись за посиланнями: https://css-tricks.com/using-flexbox та http://blogo-daru.ru/2017/04/09/prefiksy-dlya-flexbox-css-krosbrauzernyj-flexbox/. Сучасні тенденції вказують на те, що потреба використання браузерних префіксів скоро мине.

Властивість justify-content визначає, як елементи розташовуватимуться уздовж головної осі на поточній лінії. Можливі значення цієї властивості:

- flex-start (усталено) – елементи притиснуті до початку головної осі;
- flex-end – елементи притиснуті до кінця головної осі;
- center – елементи розташовуватимуться по центру;
- space-between рівномірно розподіляє блоки на проміжку від початку до кінця;
- space-around рівномірно розподіляє блоки на проміжку, але утворює рівномірні відступи між блоками і по піввідступу з країв.

Властивість align-items визначає вирівнювання по вертикальній осі. Можливі значення цієї властивості:

- flex-start – блоки притискатимуться до початку поперечної осі (догори);
- flex-end – блоки притискатимуться до кінця поперечної осі (донизу);
- flex-center – блоки по центру поперечної осі;
- strench (усталено) – блоки розтягуватимуться, займаючи все доступне місце вздовж поперечної осі, при цьому все ж таки враховуватимуться min-width / max-width, якщо вони задані;
- baseline – блоки вирівнюватимуться за базовою лінією.

Властивість flex-flow поєднує в собі дві властивості flex-direction і flex-wrap. По суті, flex-flow дозволяє однією властивістю описати напрямок головної і багаторядковість поперечної осі. Усталено – flex-flow: row nowrap.

Властивість flex-wrap зі значенням wrap дозволяє перенесення елементів на новий ряд, якщо вони не поміщаються. Значення wrap-reverse для цієї властивості задає зворотній порядок. Усталено flex-wrap має значення nowrap і при цьому блоки розташовуються в одну лінію зліва направо.

Властивість flex-direction задає напрям головної осі. Можливі значення:

- row (значення за умовчанням) – розміщення елементів в ряд зліва направо (в rtl справа наліво);

– row-reverse – розміщення елементів в ряд справа наліво (в rtl зліва направо);
– column змінює осі і формує відображення елементів у стовпчик зверху донизу уздовж поперечної осі;
– column-reverse – розміщення елементів знизу догори (зворотній порядок з притисканням донизу).

CSS властивості flex-direction, justify-content, align-items мають бути задані для flex-контейнера, а не для його дочірніх елементів. Наприклад:

```
.main {
  display: flex;
  flex-flow: row wrap;
  width: 75%;   height: 450px;
}
```

Наведений стиль main сформує відображення всіх елементів у контейнері в один ряд. Перенесення елементів по рядах контейнера задає властивість flex-flow зі значенням row wrap.

5.3.3. Основні властивості flex-елементів

Властивість flex з трьома значеннями (наприклад, flex: 1 1 30% чи flex: 1 1 auto) поєднує в собі відразу три властивості flex-grow, flex-shrink і flex-basis. Для розуміння розглянемо їх докладніше:

– flex-grow – визначає, яку частину вільного простору може зайняти контейнер щодо інших контейнерів. Це може бути тільки додатне число;
– flex-shrink – властивість, яка визначає фактор стискання елемента. Значення flex-shrink (додатне число) визначає співвідношення заповнення контейнера, коли ширина елементів є ширшою, ніж сам контейнер;
– flex-basis – початковий розмір елемента (у пікселях або відсотках).

Так, властивість flex: 1 1 auto дозволить елементу стискатися і збільшуватися, підлаштовуючись під будь-який розмір екрана. Значення властивості flex: none або flex: 0 0 auto вимкне гнучкість розміру і задасть розмір елемента фіксованим і нерегульованим для користувача при будь-якому розмірі екрана.

Властивість flex-grow визначає те, наскільки окремий елемент може бути більше сусідніх елементів, якщо це необхідно. Ця властивість показує, як захоплювати вільний простір елемента. Ціле додатне значення (від 0 – за умовчанням і більше) показує пропорцію, в якій ділити вільний простір (якщо він є) контейнера між елементами.

Якщо всі елементи всередині контейнера мають flex-grow:1, то вони будуть однакового розміру. Якщо один з них має flex-grow:2, то він буде удвічі більше, ніж решта. Якщо всі елементи всередині контейнера мають flex-grow:3, то вони будуть однакового розміру. Якщо один з них має flex-grow:12, то він буде в 4 рази більше, ніж решта.[31]

[31] Що таке Flexbox? Опис всіх css властивостей, основні принципи, переваги та недоліки. URL: http://html5.by/blog/flexbox/ (дата звернення 26.07.2018).

Тобто абсолютне значення flex-grow не визначає точну ширину. Воно визначає його ступінь «жадібності» щодо інших елементів такого самого рівня.

Властивість flex-shrink – антипод flex-grow. Визначає, наскільки елемент зменшуватиметься щодо сусідніх елементів усередині контейнера в разі нестачі вільного місця. Усталено має значення 1. Значення 0 змусить блок розтягуватись на всю ширину контейнера.

Властивість flex-basis – базовий розмір елемента по головній осі. Може задаватися у будь-яких одиницях виміру довжини (px, em, %, ...) або auto (усталено) – фіксований розмір, залежно від значень властивостей width і height. Наприклад: flex-basis: 300px.

Властивість order повідомляє браузеру номери флекс-елементів у порядку їхнього відображення. Усталено дорівнює 0. До речі, для order можна задавати і від'ємні значення, наприклад, якщо потрібно додати елемент перед першим. Значення order не ставить абсолютну позицію елемента в послідовності, воно визначає вагу позиції елемента.

Властивість align-self задає вирівнювання окремо взятого елемента уздовж поперечній осі (cross axis) і перевизначає властивість align-items. Доступні значення align-self – ті самі 5 варіантів, що і для align-items: flex-start, flex-end, center, baseline, stretch (усталене значення).

Властивість margin: auto центрує елементи флексбоксу і по горизонталі, і по вертикалі.

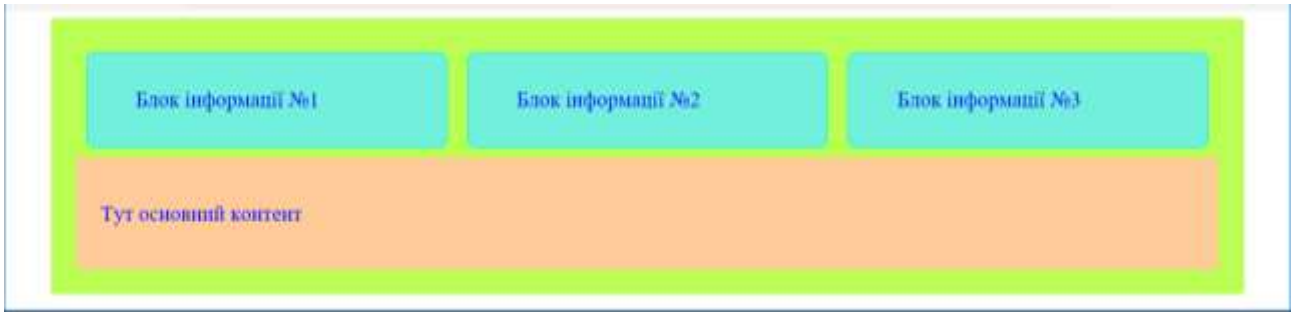
Для прикладу контейнера класу main з трьома елементами класу blok і блоком (article) під ними створимо стилі з використанням розглянутих властивостей та різними кольорами для наочності:

```
.main {
  display: flex;
  flex-flow: row wrap;
  max-width: 90%;              /* максимальна ширина блока */
  margin: auto;
  padding: 2%;                 /* відступи навколо елемента */
  background-color: #bf5;
  border-radius: 5px;          /* радіус заокруглень блока*/
}
.blok {
  order: 1;
  flex: 1 1 150px ;
  border-radius: 10px;
  border: 1px solid #0fc;      /* колір межі */
  background: #72efdd;         /* колір фону */
  padding: 3%;  margin: 10px ;
}
article {
  order: 2;
  flex: 1 1 100%;
```

```
  background: #fc9;
  padding: 10px;
}
p {
  color: #00f;
  font-size: 22px;
  padding: 15px;
}
```

До речі, для адаптивності дизайну краще використовувати ширину основного контейнера з використанням відсотків. Якщо задати жорсткі значення у пікселях, для малих екранів адаптивність не спрацює.

Далі наведено скріншоти такої веб-сторінки при зміненні розмірів вікна браузера.

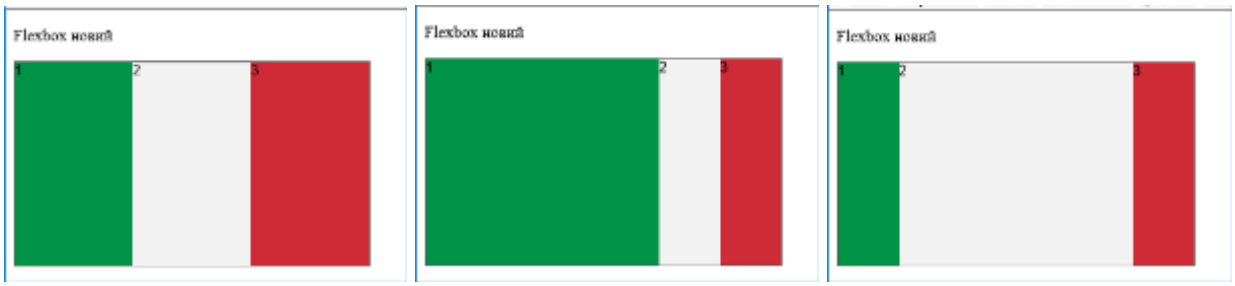


Як приклад довідника та джерела прикладів верстки на флексбоксаах пропонуємо скористатись ресурсом https://css-tricks.com.[32]

Приклад 5.1. Створимо гнучкі елементи, динамічно змінювані при наведенні на них миші.[33]

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <style>
   .flex {  /* basic styling */
     width: 350px;   height: 200px;
     border: 1px solid #555;
     font: 14px Arial;
     display: -webkit-flex;                    /* flexbox setup */
     -webkit-flex-direction: row;
     display: flex;
     flex-direction: row;
   }
   .flex > div {
    -webkit-flex: 1 1 auto;
     flex: 1 1 auto;
     width: 30px;
     -webkit-transition: width 0.7s ease-out;
     transition: width 0.7s ease-out;        /* Швидке змінення ширини за 0,7 секунди */
   }
   /* Кольори */
   .flex > div:nth-child(1) { background : #009246; }
   .flex > div:nth-child(2) { background : #F1F2F1; }
   .flex > div:nth-child(3) { background : #CE2B37; }
   .flex > div:hover   {         /* При наведенні миші */
       width: 200px;             /* ширина блока збільшиться до 200 пікселів */
   }
  </style>
 </head>
 <body>
  <p> Новий flexbox </p>
  <div class="flex">
   <div>1</div>    <div>2</div>    <div>3</div>
  </div>
 </body>
</html>
```

[32] A Complete Guide to Flexbox. URL: https://css-tricks.com/snippets/css/a-guide-to-flexbox (accessed 26.07.2018).
[33] Використання CSS flexible-боксів. URL: https://developer.mozilla.org/uk/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes (дата звернення 26.07.2018).



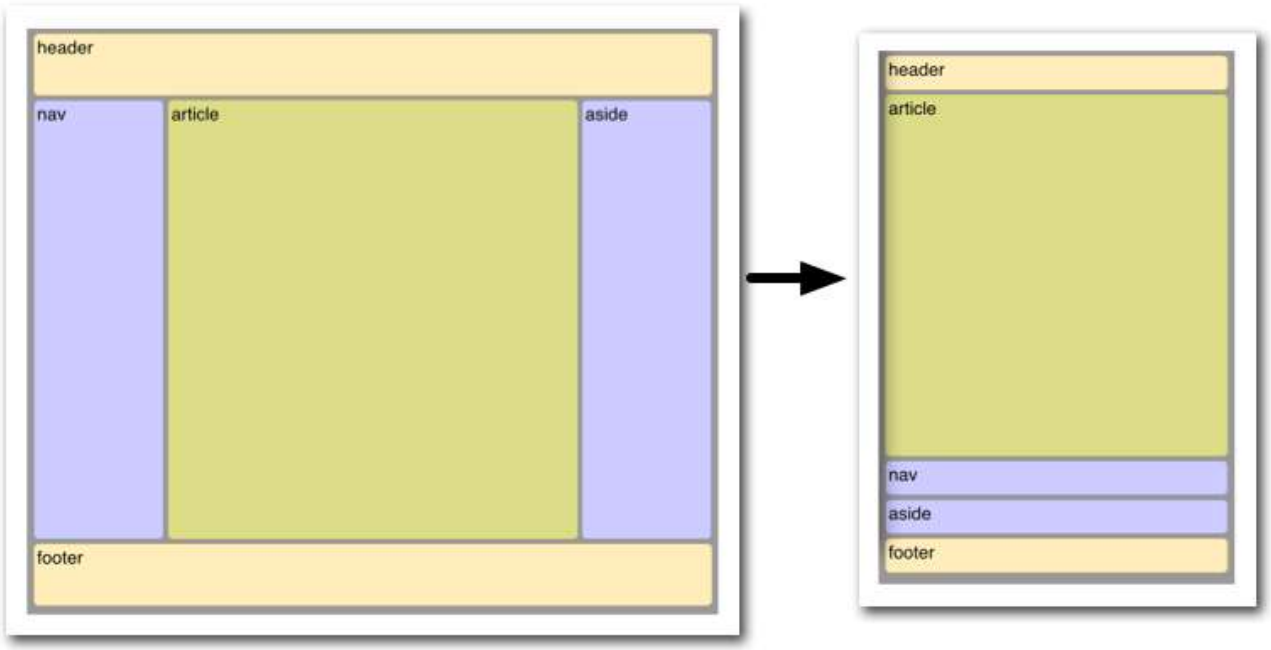
Приклад 5.2. Продемонструємо на прикладі,[34] як flexbox дозволяє динамічно змінювати макет для різних розмірів екрана, тим самим оптимізувати вигляд веб-сторінки для вікна смартфона та монітора стаціонарного ПК.

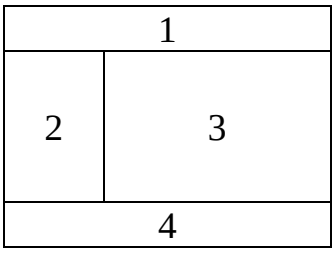
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <style>
          body {
            font: 24px Helvetica;
            background: #999999;
       }
      #main {
            min-height: 800px;
            margin: 0px;
            padding: 0px;
            display: -webkit-flex;
            display:    flex;
            -webkit-flex-flow: row;
            flex-flow: row;
   }
  #main > article {
          margin: 4px;
          padding: 5px;
          border: 1px solid #cccc33;
          border-radius: 7pt;
          background: #dddd88;
          -webkit-flex: 3 1 60%;       flex: 3 1 60%;
          -webkit-order: 2;              order: 2;
   }
  #main > nav {
          margin: 4px;              padding: 5px;
          border: 1px solid #8888bb;
          border-radius: 7pt;
```

[34] Використання CSS flexible-боксів. URL: https://developer.mozilla.org/uk/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes (дата звернення 26.07.2018).

```
        background: #ccccff;
        -webkit-flex: 1 6 20%        flex: 1 6 20%;
        -webkit-order: 1;            order: 1;
 }
#main > aside {
        margin: 4px;
        padding: 5px;
        border: 1px solid #8888bb;
        border-radius: 7pt;
        background: #ccccff;
        -webkit-flex: 1 6 20%;       flex: 1 6 20%;
        -webkit-order: 3;            order: 3;
 }
header, footer {
        display: block;
        margin: 4px;
        padding: 5px;
        min-height: 100px;
        border: 1px solid #eebb55;
        border-radius: 7pt;
        background: #ffeebb;
 }
@media all and (max-width: 640px) {
         #main, #page {
        -webkit-flex-flow: column;   flex-direction: column;
   }
   #main > article, #main > nav, #main > aside {
        -webkit-order: 0;     order: 0;
   }
   #main > nav, #main > aside, header, footer {
        min-height: 50px;
        max-height: 50px;
   }
 }
 </style>
</head>
<body>
 <header>header</header>
 <div id='main'>
   <article>article</article>
   <nav>nav</nav>
   <aside>aside</aside>
 </div>
```

```
  <footer>footer</footer>
  </body>
</html>
```





Приклад 5.3. Створити сайт із такою структурою: 1 – назва сайту; 2 – навігаційна панель зі списком назв веб-сторінок; 3 – основна частина, яка змінюватиметься, залежно від вибраної веб-сторінки сайту; 4 – дані про автора. Застосувати адаптивну верстку за допомогою флексбоксів.

Для цього треба створити HTML-файл з ім'ям *index.html* як головну веб-сторінку сайту. На цій веб-сторінці засобами флексбоксів треба утворити задану структуру, а стилі цих флексбоксів описати у файлі ***style.scc***. Готову веб-сторінку треба скопіювати чотири рази і зберегти копії з іменами: ***table.html***, ***pictures.html***, ***media.html*** та ***form.html***. Далі змінити в цих файлах вміст найбільшого флексбоксу на відповідні фрагменти HTML-файлів веб-сторінок. У навігаційні панелі цих файлів відкоригувати посилання. За потреби внести коригування стилів.

Наведемо можливий вміст усіх файлів веб-сторінок сайту.

Лістинг файлу *style.scc*

```
body {
        margin: 10px;          padding: 0px;
        font:1.3em Arial, Tahoma, "Trebuchet MS", sans-serif;
}
.main { /*флексбокс, контейнер*/
        border: 1px solid #ccc;
        padding: 10px;
        border-radius: 3px;
        display:-webkit-box;        display:-webkit-flex;
```

```
        display:-ms-flexbox;          display: flex;
        max-width:1600px;             min-height: 800px;
        margin: 20px auto;
        line-height: 140%;
        -webkit-flex-flow: row;       flex-flow: row;
       -webkit-order: 0;              order: 0;
}
.fl1 {
        padding: 5px 10px;
        background: #72efdd;
        border-radius: 5px;
        margin: 10px;
        font-size: 1em;
        flex: 1 3 15%;
        order:1;
        min-width:15%;
 }
.fl2 {
        border-radius: 5px;
        margin: 10px;
        border:1px solid #00d;
        background: #9feeff;
        font-size: 1em;
        flex: 5 1 900; order:2;
        padding:10px 3% 10px 3%;
}
.fl3 {
        font-size: 14px; display: block;
}
@media all and (max-width: 800px) {
   .main  {
            -webkit-flex-flow: column;        flex-direction: column;
      } }

h1 {
        text-align: center;
        font-size: 200%;
        font-family: fantasy;
        color: #0000ff;
}
h2 {    padding:10px;
        line-height: 120%;
        font-family: fantasy;
        font-size: 180%;
        color: #333;
```

```
        background-color: #fff;
}
.gradient {        /*градієнтна заливка*/
        background: -webkit-linear-gradient(top, #00c 0%,#fff 50%,#00f 100%,#fff 50%);
}
.note {
        font-size: 90%;
        color: white;
        margin: 10px 0;          padding: 10px; /*відступи*/
        border-left-width: 10px;
        background-color: #999999;
        width: 50%;
}
img {
        height: 200px;
        margin-left: 10px;
        vertical-align: middle;
}
.imgship {
        height: 70px;
}
table {
        border: 2px solid gray;
        border-collapse: collapse;
        background-color: #F0F0FF; /*світло-блакитний*/
}
td {    /*стиль для клітинки таблиці*/
        border: 1px solid  gray;
        padding: 5px;
}
#tabl {
         float: right;               width: 30%;
         padding-left: 15px;     padding-top: 40px;
}
```

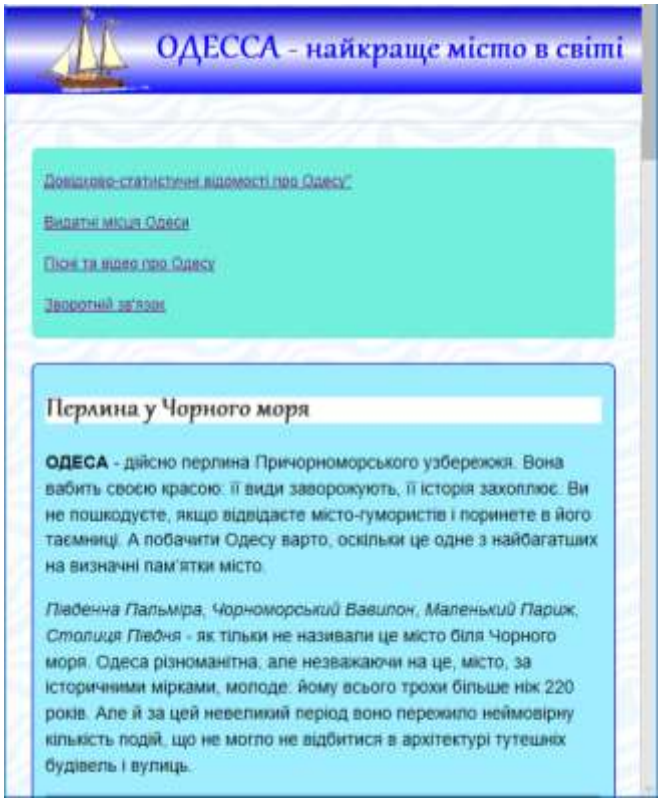
Лістинг файлу *index.html*

```
<!DOCTYPE html>
<html>
  <head>
        <title>Flexbox на практиці</title>
        <meta charset="utf-8">
        <link rel="stylesheet" type="text/css" href="style.css">
        <meta name="keywords" content="Одеса найкраще місто в світі">
        <meta name="description" content="Одеса найкраще місто в світі">
  </head>
  <body background="Photo/fon.gif" >
```

```
<h1 class="gradient">
<img src="Photo/ship.gif"  class="imgship">
ОДЕССА – найкраще місто в світі
</h1>
<div class="main">
 <div class="fl1">
   <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
   Одесу"</a></p>
   <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
   <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
   <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
 </div>
 <div class="fl2">
   <h2>Перлина у Чорного моря</h2>
   <p><strong>ОДЕСА</strong> – дійсно перлина Причорноморського узбережжя.
   Вона вабить своєю красою: її види заворожують, її історія захоплює. Ви не пошко-
   дуєте, якщо відвідаєте місто-гумористів і поринете в його таємниці. А побачити
   Одесу варто, оскільки це одне з найбагатших на визначні пам'ятки місто.</p>
   <p><em>Південна Пальміра</em>, <em>Чорноморський Вавилон</em>,
   <em>Маленький Париж</em>, <em>Столиця Півдня</em> – як тільки не називали
   це місто біля Чорного моря. Одеса різноманітна, але незважаючи на це, місто, за
   історичними мірками, молоде: йому всього трохи більше, ніж 220 років. Але й за
   цей невеликий період воно пережило неймовірну кількість подій, що не могло не
   відбитися в архітектурі тутешніх будівель і вулиць.</p>
   <hr><h2>Дещо з історії міста</h2>
   <p>Будівництво міста і морського порту почалося 22 травня 1794 року за указом
   Катерини Другої, що повинно було значно покращити торгівельні зв'язки з Євро-
   пою. Почалося будівництво під керівництвом віце-адмірала Хосе де Рибаса –
   першого градоначальника Одеси, за планом, що склав полковник-інженер Франц
   Деволан.</p>
   <p>Тутешні землі можуть розповісти про скіфів, кіммерійців, сарматів і греків.
   Кожна культура залишила свій слід в історії міста. Місцевість, на якій нині роз-
   ташована Одеса, була заселена ще до початку Великого переселення народів,
   вона належала і Золотій Орді, і Великому князівству Литовському.</p>
   <p>Відзначилася Одеса і в Другій світовій війні. <strong>Одеса – місто-
   герой</strong>. Оборона міста трималася <strong>73 дні</strong>. Щоб дізнати-
   ся більше про цей період варто відвідати Меморіал героїчної оборони Одеси
   411-ї берегової батареї. Цей меморіальний комплекс, присвячений героїчній
   обороні міста під час Великої Вітчизняної війни. Комплекс має в своєму складі
   музей, виставку військової техніки під відкритим небом, батарею берегової обо-
   рони, а також великий засаджений дубами парк.</p>
   <hr> <h2>Сьогодення одеситів</h2>
   <p>Сьогодні місто розвивається як курортний і промисловий центр. Це одне з
   найулюбленіших туристами місць в Україні, і в цьому немає нічого дивного. Тут
```

```
        Ви можете ніжитися в променях теплого південного сонця і купатися в ласкаво-
        му Чорному морі. Кількість пам'яток не злічити на пальцях однієї руки:</p>
        <ul>
            <li><a href="http://odessa-life.od.ua/article/5449-sekrety-odesskogo-rynka-privoz">
            Привоз </a>, </li>
            <li><a href="http://www.odessaguide.net/sights_deribasovskayastreet.ru.html">
            вулиця Дерибасівська</a>,</li>
            <li>Одеський театр опери і балету (другий за красою в Європі),</li>
            <li>Приморський бульвар,</li>
            <li>Напівкругла площа (1826-1829) з пам'ятником Рішелье,</li>
            <li>палаци Воронцова, Наришкіної і Потоцького,</li>
            <li>Пасаж,</li>
            <li>Успенський монастир (1824) і Духовна семінарія,</li>
            <li>Одеський морський порт (найбільший на Чорному морі),</li>
        </ul>
        <p>– і це далеко не весь список. Для маленьких і непосидючих відкриває свої
        двері Одеський дельфінарій.</p>
        <p>Все, що Ви можете побачити в Одесі. Є тут навіть «восьме чудо світу» –
        <em>Потьомкінські сходи</em>, своєрідний вхід у місто з боку моря.  2004 року
        було проведено голосування, серед найкрасивіших сходів, і в Європі Потьомкінські
        увійшли до десятки кращих, посівши 6-те місце, також у цьому списку присутні
        сходи Монмартр у Парижі та Сходи Храму Афіни на о. Родос.</p>
        <p>В Одесі існує величезна система катакомб, вони не тільки за своїм хитросп-
        летінням, але й за протяжністю (≈ 3 000 км) – одні з найбільших у світі.</p>
        <p class="note"> Для порівняння: довжина Римських катакомб становить 300 км,
        Паризьких – 500 км.</p>
        <p>Якщо Ви втомилися від споглядання пам'ятників, то ласкаво запрошуємо до
        Одеського Ботанічного саду, в якому Ви не лише відпочините душею і тілом, а й
        побачите фактично перший парк степової частини України. За двохсотлітню іс-
        торію саду з усього світу у ньому назбиралася величезна колекція найрізнома-
        нітніших рослин.</p>
        <p>Одеса – це місто веселощів та гумору. Відвідавши його лише раз, Вам
        обов'язково захочеться побачити його ще. Адже Одеса – це не зовсім місто – це
        <sup> ☺ </sup>посмішка Бога.<sub font-color="red">☼</sub></p>
        <img src="photo/img1.jpg" alt="Одеса – це не місто, а ...">
        <img src="photo/img0.jpg"> <img src="photo/img00.jpg">
        <img src="photo/img000.jpg">
        <p class="note"> Використовувались матеріали статті
        <a href="http://angatravel.com.ua/ua/ah-odesa-perlina-bilya-morya.html">
        «АХ, ОДЕСА! ПЕРЛИНА БІЛЯ МОРЯ»</a> </p>
      </div>
    </div>
    <div class="fl3"><p > &#169 Зайченко Маргарита</p></div>
  </body>
</html>
```

При зміненні розміру екрана браузера вміст веб-сторінки гнучко зміниться:

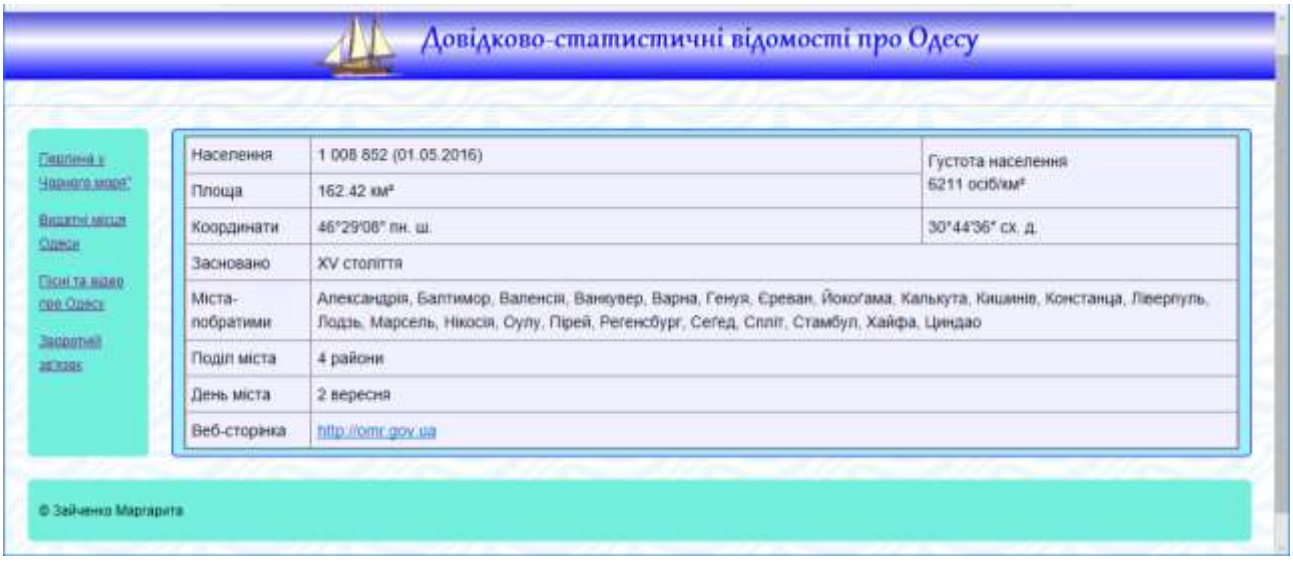


Лістинг файлу *table.html*

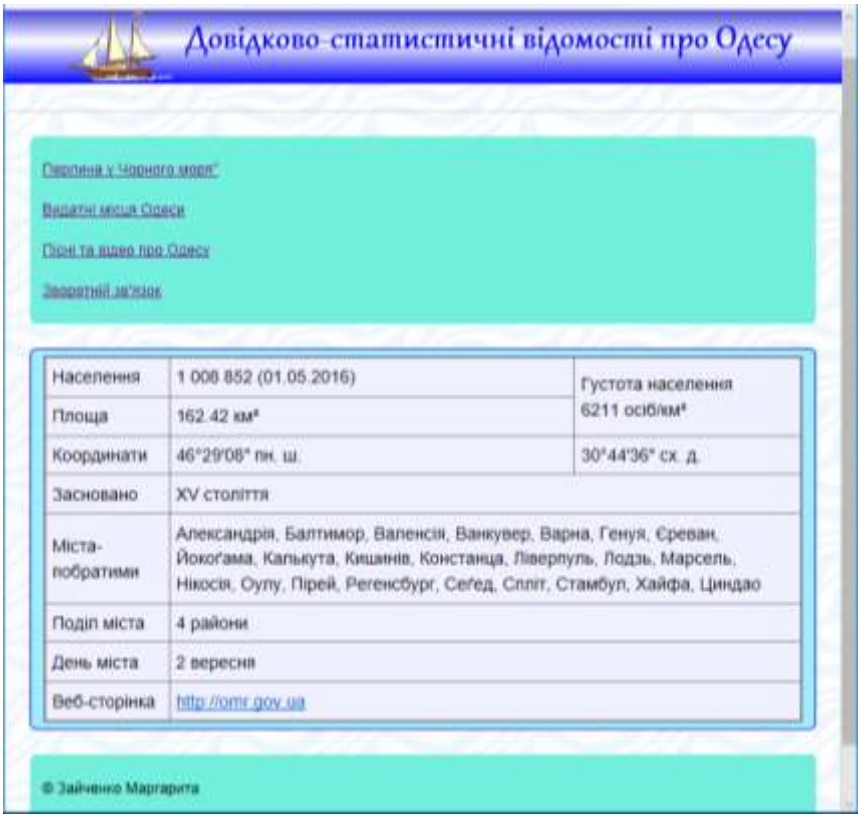
```
<!DOCTYPE html>
<html>
  <head>
        <title>Flexbox на практиці</title>
```

```
    <meta charset="utf-8">
  <link rel="stylesheet" type="text/css" href="style.css">
  <meta name="keywords" content="Одеса найкраще місто в світі">
  <meta name="description" content="Одеса найкраще місто в світі">
</head>
<body background="Photo/fon.gif">
    <h1 class="gradient">  <img src="Photo/ship.gif"  class="imgship">
      Довідково-статистичні відомості про Одесу
    </h1>
    <div class="main">
     <div class="fl1">
      <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
      <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
      <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
      <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
     </div>
     <div class="fl2">
      <table>
`        <tr>  <td>Населення</th>   <td>1 008 852 (01.05.2016)</th>
              <td rowspan="2" width="30%">Густота населення<br>6211 осіб/км²</td>
        </tr>
        <tr>   <td>Площа</td>  <td>162.42 км²</td>
        </tr>
        <tr>   <td>Координати</td>
               <td> 46°29'08" пн. ш.</td>  <td> 30°44'36" сх. д.</td>
        </tr>
        <tr>
              <td>Засновано</td>   <td colspan="2">XV століття</td>
        </tr>
        <tr>
             <td>Міста-побратими</td>
      <td colspan="2">Александрія, Балтимор, Валенсія, Ванкувер, Варна, Генуя,
      Єреван, Йокоґама, Калькута, Кишинів, Констанца, Ліверпуль, Лодзь, Марсель,
      Нікосія, Оулу, Пірей, Регенсбург, Сеґед, Спліт, Стамбул, Хайфа, Циндао</td>
        </tr>
        <tr>  <td>Поділ міста</td>   <td colspan="2">4 райони</td>
        </tr>
        <tr>  <td>День міста </td>   <td colspan="2">2 вересня</td>
        </tr>
        <tr>  <td>Веб-сторінка</td>
          <td colspan="2"><a href="http://omr.gov.ua">http://omr.gov.ua</a></td>
        </tr>
      </table>
     </div>
    </div>
```

```
        <div class="fl3">
          <p> &#169 Зайченко Маргарита</p>
        </div>
  </body>
</html>
```



При зміненні розміру екрана браузера вміст веб-сторінки гнучко зміниться:



Лістинг файлу *pictures.html*:

```
<!DOCTYPE html>
<html>
  <head>
        <title>Flexbox на практиці</title>
```

```
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta name="keywords" content="Одеса найкраще місто в світі">
    <meta name="description" content="Одеса найкраще місто в світі">
    <style>
      .thumbs {    display: inline-block;    }
      .thumbs:hover .thumb {  transform: scale(0.9);  /* Зменшити всі фотографії */
      }
     .thumb { background: #27f;       /* Колір рамки */
              padding: 10px;          /* Товщина рамки навколо картинки */
              transition: 1s;         /* Час переходу */
      }
     .thumb:hover { transform: scale(1.2) !important; }  /* Збільшити фотографію */
      img { height: 220px;
            margin-left: 10px;
      }
    </style>
</head>
<body background="Photo/fon.gif" >
    <h1 class="gradient"> <img src="Photo/ship.gif"  class="imgship">
      Видатні місця Одеси      </h1>
    <div class="main">
      <div class="fl1">
        <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
        <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості
        про Одесу</a></p>
        <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
        <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
      </div>
      <div class="fl2">
        <div class="thumbs">
            <img src="Photo/img_lb5_1.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_2.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_3.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_4.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_5.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_6.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_7.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_8.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_9.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_10.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_11.jpg" alt="" class="thumb">
            <img src="Photo/img_lb5_12.jpg" alt="" class="thumb">
        </div>
      </div>
```

```
        <div class="fl3">     <p> &#169 Зайченко Маргарита</p></div>     </div>
  </body>
</html>
```

Лістинг файлу *media.html*:

```
<!DOCTYPE html>
<html>
  <head>
        <title>Flexbox на практиці</title>
        <meta charset="utf-8">
        <link rel="stylesheet" type="text/css" href="style.css">
        <meta name="keywords" content="Одеса найкраще місто в світі">
        <meta name="description" content="Одеса найкраще місто в світі">
  </head>
  <body background="Photo/fon.gif" >
        <h1 class="gradient"> <img src="Photo/ship.gif"  class="imgship">
          Пісні та відеовізитівка про Одесу
        </h1>
        <div class="main">
          <div class="fl1">
            <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
          <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
          Одесу</a></p>
            <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
            <p> <a href="form.html" target="mainFrame">Зворотній зв'язок</a></p>
          </div>
          <div class="fl2">
            <div class="thumbs">
              <h2">Пісні про Одесу</h2>
              <figure>
                    <audio src="Audio/Odessa moy gorod rodnoy.mp3" controls></audio>
```

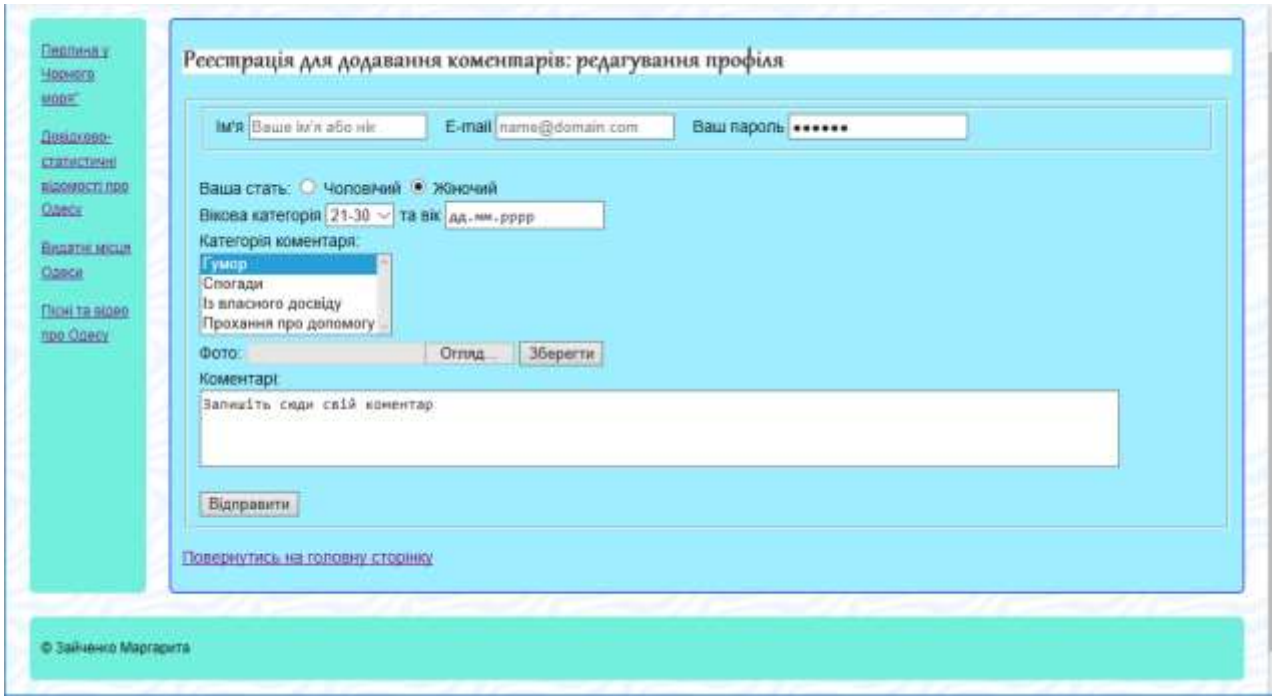
```
                    <figcaption>Одесса - мой город родной</figcaption>
                    <audio src="Audio/у Черного моря.mp3" controls></audio>
                    <figcaption>У черного моря</figcaption>
                    <audio src="Audio/ах, Одесса жемчужина у моря.mp3" controls>
                    </audio>
                    <figcaption>Ах, Одесса жемчужина у моря</figcaption>
            </figure>
            <br> <h2>Відео про Одесу</h2>
          <object data="Audio/aerovizitkaOdessa.mp4" type="video/mp4"
          standby="Йде завантаження. Зачекайте.">Візитівка Одеси </object>
            <figcaption>Візитівка Одеси</figcaption>
          </div>
        </div>
  </div>
  <div class="fl3"><p > &#169 Зайченко Маргарита</p></div>
</body>
</html>
```

Лістинг файлу *form.html*:

```
<!DOCTYPE html>
 <head>
  <meta charset="utf-8">
  <title>Реєстрація для додавання коментарів: редагування профіля</title>
  <link rel="stylesheet" type="text/css" href="style.css">
  <style>
     textarea {  width: 90%;  height: 60px;   }
  </style>
 </head>
```

```
<body background="Photo/fon.gif">
  <div class="main">
     <div class="fl1">
        <p> <a href="index.html" target="mainFrame">Перлина у Чорного моря"</a></p>
        <p> <a href="table.html" target="mainFrame">Довідково-статистичні відомості про
        Одесу</a></p>
        <p> <a href="pictures.html" target="mainFrame">Видатні місця Одеси</a></p>
        <p> <a href="media.html" target="mainFrame">Пісні та відео про Одесу</a></p>
     </div>
     <div class="fl2">
          <h2>Реєстрація для додавання коментарів: редагування профіля</h2>
          <form action="#" method="post">
          <fieldset>
            <fieldset>
           <label>  Ім'я
              <input type="text" name="name" placeholder="Ваше ім'я або нік" required>
          </label>
                <label>     E-mail
                <input type="email" name="email" placeholder="name@domain.com" required>
                </label>
                <label for="password-field">   Ваш пароль</label>
                <input type="password" name="password" id="password-field" value="123456">
           </fieldset> <br> Ваша стать:
           <label>
                 <input type="radio" name="question-one" value="m">  Чоловічий
           </label>
           <label>
                 <input type="radio" name="question-one" value="w" checked>  Жіночий
            </label>
            <br> Вікова категорія
            <select name="age">
                 <option value="до 15">до 15</option>
                 <option value="16-20">16-20</option>
                 <option value="21-30" selected>21-30</option>
                 <option value="31-40">31-40</option>
                 <option value="41-50">41-50</option>
                 <option value="51-60">51-60</option>
                 <option value="61-70">61-70</option>
            </select>       та вік <input type="date" name="birthday">
            <br><br>Категорія коментаря:<br>
            <select name="tehn" multiple size="4">
                 <option selected> Гумор</option>
                 <option> Спогади</option>
                 <option> Із власного досвіду</option>
                 <option> Прохання про допомогу</option>
```

```
            </select>
            <br><br> Фото: <input type="file" value="Choose File">
            <input type="submit" value="Зберегти">   <br> <br>
            <label>  Коментарі:<br>
                 <textarea name="comment">Запишіть сюди свій коментар  </textarea>
            </label>    <br><br>
            <input type="submit" name="submit" value="Відправити">
          </fieldset>
        </form>
        <p><a href="index.html">Повернутись на головну сторінку </a> </p>
    </div>
    </div>
     <div class="fl3">
        <p> &#169 Зайченко Маргарита</p>
  </div>
 </body>
</html>
```



Контрольні запитання

1) Чим логічна розмітка веб-документа відрізняється від фізичної?
2) Що таке верстка сайту?
3) Чим верстка веб-сторінок відрізняється від поліграфічної?
4) Назвати найпоширеніші види верстки.
5) Охарактеризувати табличну верстку. Які її переваги і недоліки?
6) Охарактеризувати фреймову верстку. Які її переваги і недоліки?
7) Охарактеризувати блокову верстку. Які її переваги і недоліки?
8) У чому специфіка адаптивної верстки?
9) Що таке адаптивний веб-дизайн?
10) Яка мета адаптивного веб-дизайну?
11) Яка головна концепція flexbox?
12) Коли доречно застосовувати flexbox?
13) Які переваги використання flexbox для розмітки сайтів?
14) Що треба зробити, щоб задати контейнер флексбоксом?
15) Для чого задавати браузерні префікси для флексбоксів?
16) Назвати властивість, яка визначає, як елементи розташовуватимуться уздовж головної осі на поточній лінії.
17) Яке значення треба задати для властивості justify-content, щоб елементи рівномірно розподілялись на проміжку від початку до кінця?
18) Яке значення треба задати для властивості justify-content, щоб елементи рівномірно розподілялись на проміжку й утворювали рівномірні відступи між блоками та по піввідступу з країв?
19) Яка властивість визначає, як елементи розташовуватимуться уздовж поперечної осі? Яке її значення центруватиме блоки по поперечній осі?
20) Яка властивість дозволяє одною властивістю описати напрямок головної і багаторядковість поперечної осі? Записати цю властивість зі значеннями, які зададуть розміщення блоків у стовпець без переносів.
21) Що саме задаватиме властивість flex-direction: column-reverse?
22) Що саме задаватиме властивість flex: 1 1 auto?
23) Яке призначення властивості order? Навести приклад.
24) Назвати flex-властивості, які треба задавати тільки для контейнера.
25) Назвати flex-властивості, які треба задавати тільки для елементів.

Розділ 6
Веб-дизайн

6.1. Елементи веб-дизайну

Кожна веб-сторінка містить певний набір елементів, які є обов'язковими. Їхня кількість і різноманітність можуть варіюватися залежно від тематики сайту. Компонування цих елементів та їхній вигляд є головним завданням веб-дизайнера.

Елементи веб-дизайну[1] – це ті абстрактні матеріали, з якими доводиться працювати дизайнерові. Основними елементами дизайну є: простір, лінія, фігура, колір, текстура, шрифт, форма, світлотінь, розмір.

Усі елементи веб-дизайну є «будівельними матеріалами», від правильного використання яких залежить міць конструкції дизайну. Розглянемо докладніше основні елементи дизайну.

Простір – це своєрідний фундамент композиції, на якому ґрунтується вся структура елементів веб-дизайну сайту. Він може бути двовимірним, у вигляді плоскої картинки, або тривимірним, у форматі об'ємного зображення.

Лінія – найважливіший (після фундаменту (простору)) елемент для веб-дизайну, який бере участь у створенні цілісної картини. Будуються лінії на поверхні за допомогою послідовно з'єднаних точок. Завдяки їм задається загальний обрис і контури веб-дизайну, які допомагають людському оку комфортно і структуровано сприймати інформацію. Лінії можуть бути прямого, кривого, хвилястого, горизонтального, вертикального, паралельного або пунктирного виду. Утворити ескіз майбутнього проекту за допомогою ліній можна не лише в графічних редакторах, а й на аркуші паперу, як часто і роблять веб-дизайнери.

Фігура – елемент веб-дизайну, що утворюється шляхом перетину декількох ліній між собою. Фігури прийнято розділяти на два види[2]: органічні (обриси машин, будинків, меблів та інших предметів, створінь і явищ, які зустрічаються в житті) та геометричні (прямокутник, ромб, коло, трикутник, квадрат тощо).

Колір – важливий елемент веб-дизайну сайту, адже колірне сприйняття безпосередньо впливає на психіку людини і навіть формує настрій відвідувача. Завдяки кольорам можна передати емоції і задати для користувача визначений ритм думки. Наприклад, перебуваючи на сайті з переважним темно-синім або чорним кольором, людина мимоволі зосереджується і стає серйозніше, а світло-зелений, помаранчевий або жовтий кольори налаштовують на посмішку і спокій. Завдяки яскравим відтінкам можна привернути увагу відвідувача або побу-

[1] Веб-дизайн. URL: http://іт.словник.укр/index.php/Веб-дизайн# (дата звернення 17.09.2018).

[2] Основные элементы веб-дизайна. Как благодаря им создать эффективный сайт. URL: https://webformyself.com/osnovnye-elementy-veb-dizajna-kak-blagodarya-im-sozdat-effektivnyj-sajt (дата обращения 18.09.2018).

дувати змістовний ланцюжок на сайті, за яким користувач буде слідувати, щоб знайти потрібну йому інформацію.

Текстура при оформленні веб-сайту дозволяє задати вигляд поверхні об'єкта. Вона може мати гладку, м'яку або шорстку форму. Іноді на сайтах фон та інші елементи задають текстурою дерева, шкіри, заліза тощо. Це доволі корисний інструмент, проте, тут головне використовувати його з розумом.

Форма відрізняється від фігури тим, що є тривимірним об'єктом. До її складу входять такі параметри, як ширина, глибина і висота. Створюється форма завдяки з'єднанню декількох фігур воєдино і додаванню тіней.

Світлотінь відповідає за темні і світлі області певного об'єкта. Цей елемент подається у вигляді зовнішнього джерела світла, за рахунок якого накладаються відблиски і різні тіні на певний об'єкт. Світлотінь допомагає надати зображенню візуальну глибину.

Розмір. Даний параметр стосується кожного окремого елемента. Розмір блоків та інших об'єктів визначається замовником під час складання технічного завдання або ж самим виконавцем ще до безпосереднього старту проектування веб-дизайну. Розмір продумується в першу чергу, щоб мати хоч якесь уявлення про майбутнє зображенні. Визначається він в залежності від інших елементів на сайті або щодо всього веб-дизайну в цілому.

Дотримання грамотного використання основних елементів веб-дизайну допоможе досягти гарних результатів у його проектуванні.

6.2. Принципи веб-дизайну

Принципи веб-дизайну визначають правила взаємодії всіх елементів. Основними принципами веб-дизайну є: акцент, контраст, баланс, точність, стиль, зручність сприйняття, напрямок уваги, пропорції, масштаб, ритм, єдність.

Чим більше уваги приділяти цим принципам, тим вдалішим в результаті вийде дизайн. Недосвідчені розробники рідко дотримуються загальноприйнятих принципів веб-дизайну, чи то через незнання, чи то через впевненість, що вони розумніші інших, а тому їхні результати іноді жахають.[3] Враховуючи специфіку кожного з цих принципів, під час створення графічного макета треба сконцентруватися на потрібних аспектах майбутнього проекту. Основні принципи веб-дизайну – це не чарівні формули, які сприяють вирішенню завдання за лічені секунди, а скоріше орієнтири, за якими доречно слідувати на шляху створення успішного дизайну.

Акцент. Кожен досвідчений майстер веб-дизайну знає про важливість акцентування уваги на ключових елементах сайту. Якщо графічне акцентування буде розпорошено по всіх частинах сторінки, то можна навіть не сподіватися на успішний результат (особливо, якщо йдеться про комерційні проекти). В першу чергу,

[3] Основные принципы веб-дизайна. URL: https://webformyself.com/osnovnye-principy-veb-dizajna (дата обращения 18.09.2018).

треба визначити ключові елементи на сайті і розставити їх пріоритетність. Менш важливі деталі доречно залишити в тіні або навіть позбутися їх, а елементи з високим пріоритетом ненав'язливо виділити на загальному тлі. Задля грамотної реалізації подібного процесу треба провести глибокий аналіз свого проекту та його цільової аудиторії. Знайшовши однозначну відповідь на питання щодо того, навіщо відвідувачі заходять на сайт, можна буде акцентувати увагу на певній зоні (наприклад, каталозі товарів або блоці новин). Здійснюється це не лише за допомогою яскравих графічних акцентів, а й, наприклад, шляхом розроблення зручного і нестандартного навігаційного блока, на який хотілося б натиснути.

Контраст. При розробленні веб-дизайну важливо витримувати загальний контраст – візуальну диференціацію декількох елементів. Фірмовий стиль, звичайно ж, має велике значення, проте не треба перетворювати сторінку на суцільний, монотонний, малопривабливий масив даних. Щоб відвідувачі затримувалися довше, основні елементи на сайті треба чітко позначити. Існують численні способи для реалізації контрастування. Зробити це можна за допомогою змінення розміру шрифту, додавання незвичайних кольорів, заміни розташування елементів тощо.

Баланс. Одним з ключових принципів веб-дизайну є балансування. Візуальне навантаження, яке створює групування елементів у дизайні, повинне бути рівномірно розподілене в рамках веб-сторінки. Веб-сайт не має виглядати порожнім, однак, перевантаження графікою ще більше погіршить ситуацію.

Баланс буває двох видів: симетричний і асиметричний[4]. **Симетричний баланс** у дизайні досягається, коли ліва і права половини дизайну щодо тієї чи іншої осі є ніби дзеркальними копіями один одного і несуть ідентичне візуальне навантаження. Здебільшого це характерно для сайтів, де логотип і верхнє меню візуально розташовуються по центру. **Асиметричний баланс** досягається, коли візуальне навантаження в рамках веб-сторінки рівномірно розподіляється по тій чи іншій осі, проте, окремі елементи двох складових дизайну не є дзеркально однаковими.

Балансування є одним із тонких аспектів дизайну, до якого багато хто вдається інстинктивно. Чи не виглядає дизайн нібито перевернутим з ніг на голову? Чи не здається він неврівноваженим? Саме такі питання треба задавати собі, щоб зрозуміти, чи є недоліки в балансі.

Точність. При розробленні веб-дизайну треба намагатися не допускати будь-якого роду неточності. Розмір ідентичних блоків має збігатися піксель у піксель, усі елементи треба якомога ближче підганяти під межі сторінки та максимально ефективно організувати вільний простір. Додаткова перевірка і доопрацювання дрібних деталей у цьому сенсі є ознакою професіоналізму. У новачків найбільш проблемним місцем, в якому виникає безліч неточностей, є розроблення фонового зображення для сайту. Тут не треба забувати, що ідеальне відображення на одному екрані ще не означає, що картинка буде так само ви-

[4] Основные принципы веб-дизайна и их характеристики. URL: http://www.designonstop.com/webdesign/article/osnovnye-principy-veb-dizajna-i-ix-xarakteristiki.htm (дата обращения 18.09.2018).

глядати і на інших. Треба враховувати різні роздільні здатності, браузери, формати моніторів тощо.

Стиль. Фірмовий візуальний стиль дизайну передбачає використання на всіх елементах сайту однакового графічного почерку. Повинна вимальовуватися чітка кольорова палітра, відмінні лінії, обриси і збалансований набір шрифтів. В ідеалі варто розробити єдину стилізацію і для зображень на сайті. Ясно промальований логічний зв'язок елементів полегшить процес пошуку необхідної інформації для користувача. Чітка структура сайту і приємна для очей стилізація – це невід'ємна частина будь-якого проекту високого рівня.

Зручність сприйняття. Сприйняття є своєрідною стежкою, якою користувач пересувається по сайту при візуальному ознайомленні з елементами дизайну.[5] Якщо веб-дизайнер не розставив певних акцентів і не налагодив візуальний контраст елементів сторінки, то відвідувач просто покине сайт, не бажаючи довго розбиратися в тому, що тут і як працює. Все повинно бути максимально простим і зрозумілим. За долі секунди людина має зрозуміти, як отримати те, за чим вона прийшла.

При обдумуванні того, як забезпечити зручність сприйняття, треба пам'ятати про природний порядок речей. Згідно з дослідженнями, люди схильні оглядати ті чи інші речі в передбачуваній манері. Зазвичай людина ковзає поглядом зліва направо і зверху вниз. Саме тому сайти, які нав'язують користувачам вивчення вмісту справа наліво, здебільшого відштовхують відвідувачів. У тому, щоб йти проти природного порядку речей, немає нічого неправильного; варто лише зважати на можливі наслідки такого рішення. Кожен сайт забезпечує для користувача певне сприйняття, яке може бути як зручним, так і незручним. Але як зрозуміти, що сайт не буде забезпечувати зручність сприйняття свого вмісту? Напевно перевагу матимуть ті сайти, які забезпечують плавний, комфортний і максимально природний процес сприйняття.

Сайт може мати складну структуру, але повинен забезпечувати природний і зручний підхід до сприйняття свого вмісту, щоб не доводилося стрибати поглядом по веб-сторінці. Сайт, який забезпечує зручне сприйняття, сприяє тому, щоб у людини виникло бажання ще раз пройтися поглядом по тому, що він вже подивився, тобто підтримувати у нього інтерес.

6.3. Історія веб-дизайну

Темний вік (1989). Витоки веб-дизайну цілком можна охарактеризувати як чорно-білі, оскільки екрани були монохромні. Дизайн на той час створювали шляхом використання символів псевдографіки і табуляції.

Таблиці (1995). Виникнення браузерів, які могли демонструвати кольорові малюнки, стало першим кроком у веб-дизайні. Найбільш доступним способом для структурування інформації стала концепція таблиць в HTML. Все

[5] Макнейл П. Веб-дизайн. Идеи. Секреты. Советы. Питер, 2000. 272 с.

пішло з книги Д. Сігела «Створення приголомшливих сайтів».[6] Програмісти почали вкладати таблиці в таблиці, намагаючись поєднати статистичні і динамічні елементи. Цей спосіб довгий час був провідним при розробленні веб-дизайну.[7] Таблиці надавали безвідмовні можливості вирівнювати дані по вертикалі, задавати параметри у пікселях або у відсотках. Недоліком такого підходу є складність та доволі крихкі багаторазово вкладені табличні структури.

JavaScript (1995). Відповіддю на табличне обмеження HTML став JavaScript, який дозволив створювати спливаючі вікна, динамічно змінювати розташування елементів. JS (або JavaScript) – це дуже потужний елемент в умілих руках, проте він завантажується окремо поверх HTML-матеріалу. Сьогодні JS не використовується там, де можна задіяти CSS. Але як і раніше, він залишається важливим інструментом у front-end (jQuery) і back-end (Node. Js) розробці.

Flash (1996). Технологія Flash дозволила зняти обмеження у веб-дизайні, надала небачену досі свободу у проектуванні будь-яких форм, макетів, анімації, взаємодії, використанні яких завгодно шрифтів. Ці роки можна назвати епохою яскравих анімаційних заставок та інтерактивних ефектів на обкладинках (перших сторінках) сайтів. Для відтворення Flash треба було встановити flash-плагін, бажано останньої версії, а отже такий дизайн був доступним не всім. Крім того, завантаження Flash потребувало значної обчислювальної потужності, а відтак певного часу і терпіння. Після відмови Apple використовувати Flash у своєму першому iPhone (2007) ця технологія поступово занепала.

CSS (1998). Новим способом структурування дизайну став CSS (Cascading Style Sheets). Основна ідея CSS – розділити контент (змістовне наповнення сайту) та його оформлення (візуальну складову). Контент задається засобами HTML, а його візуальне форматування – через CSS. Основною проблемою першої версії CSS було відображення верстки в різних браузерах. З часом браузери стали підтримувати CSS, однак баги зустрічалися часто.

Сітки і фреймворки (2007). Нове випробування веб-дизайну пов’язане з потребою перегляду веб-сайтів через мобільні телефони. Крім різних макетів для девайсів, з’явилася серйозна проблема з контентом: варто підганяти його під маленький екран або зробити можливість перегортати вниз. Куди на такому екрані вставити миготливу рекламу? Швидкість завантаження стала гострим питанням, адже чим повільніше завантажується сторінка, тим швидше інтернет «спалює» гроші за трафік. На початку успішних змін придумали колонки. Наступним кроком стандартизували основні елементи, кнопки, форми, навігацію. Тим самим створили цілу бібліотеку візуальних форм з кодом. Безумовними лідерами стали Bootstrap і Foundation, які продемонстрували, що грань між сайтом і застосунком стирається. Мінусом стала одноманітність дизайну, але дизайнери не могли це змінити, не розбираючись у коді.

Адаптивний веб-дизайн (2010). Ітан Меркот придумав і запропонував

[6] Siegel D. Creating Killer Sites Hayden Books, 1996. 270 p.

[7] A brief history of web design for designers. URL: http://blog.froont.com/brief-history-of-web-design-for-designers (accessed 17.09.2018).

адаптивний веб-дизайн, ідея якого у швидкому відображенні однакового контенту на різних дизайнерських макетах, залежно від розміру екранів. Концептуальний прорив полягає у створенні декількох макетів засобами HTML і CSS з різними способами відображення картинок, швидкістю завантаження сторінки, семантикою, структурою мобільного чи робочого столу тощо. Головний плюс адаптивного дизайну – коректне завантаження сайту на будь-яких пристроях. Мінусом є те, що створення декількох макетів забирає достатню кількість часу на професіоналізму розробника.

Flat-дизайн (2012). Плоский дизайн з простими візуальними ефектами став тенденцією часу. Іконки замінили блискучі кнопки, що дозволило використовувати векторні зображення. Основну увагу віддано саме контенту.

Сьогодення і майбутнє (2018). Сьогодення породжує все нові і нові ідеї. Нові засоби CSS пропонують вищу гнучкість для розміщення модулів. Ще однією цікавою ідеєю є flexbox, які надають змогу проектувати і видозмінювати макети за допомогою однієї властивості замість написання довгого коду. Все більша увага приділяється веб-компонентам – набору пов'язаних елементів, наприклад, галерея, форма реєстрації тощо. Розроблення відбувається швидше, адже такі компоненти стають будівельними блоками, які можна використовувати багаторазово й оновлювати окремо.

6.4. Поширені помилки дизайнерів-початківців

Головна помилка початківців дизайнерів веб-сайтів – це акцентування уваги на малюванні екранів замість того, щоб думати про загальну картину (бізнес-завдання, сценарії взаємодії та очікування користувачів).

Грамотно побудований дизайн-процес позбавляє від типових помилок.

Для запобігання помилок взаємодії веб-дизайн сайту повинен відповідати користувачеві на два основних питання: «Як користуватися тим чи іншим інструментом на сайті для досягнення мети?» і «Чи працює інструмент так, як очікувалося?».[8] І не варто при цьому розраховувати, що користувачі намагатимуться навчитися правильно працювати із сайтом, не отримавши своїх відповідей, адже у них завжди є простий спосіб уникнути будь-яких проблем – піти на інший сайт.

Завданням веб-дизайну є забезпечення зручної подачі інформації користувачеві сайту, задоволення естетичного смаку аудиторії сайту.

Ось декілька основних аспектів веб-дизайну, на які варто звернути увагу в боротьбі з користувацькими помилками.

1) **Складність і незвичність дизайну**. Креативний дизайн високо цінується при створенні сайтів, адже він допомагає створити унікальні рішення, що запам'ятовуються користувачам. Проте коли такий дизайн заважає сайту ефективно виконувати свої завдання, треба його переглянути. Адже не дарма існу-

[8] Веб-дизайн сайту і користувацькі помилки. URL: http://webstudio2u.net/ua/design-web/852-veb-dizain-saita-i-polzovatelskie-oshibki.html (дата звернення 28.06.2018).

ють певні стандарти у веб-дизайні: коли користувачі бачать схожі моделі взаємодії на сайтах, з якими працювали раніше, вони успішно застосовують їх на кожному наступному сайті, не боячись зробити помилку.

2) **Неправильна навігація**. Навігація по сайту має бути інтуїтивно простою, щоб зрозуміти те, як використовувати сайт із найбільшою ефективністю. Наявність зрозумілої для користувача структури дизайну дозволяє легше знаходити потрібну інформацію. Треба створити таку карту сайту, за допомогою якої користувачі зможуть самі вирішувати, яким способом знайти релевантну інформацію. Доречно розміщувати пошуковик по сайту.

3) **Елементи, що вводять в оману**. Наведемо приклади таких елементів. Підкреслений синій шрифт завдяки вже згаданим стандартам веб-дизайну сприймається користувачами як сигнал, що набраний таким шрифтом текст є посиланням. А підсвічений тінню прямокутник, що якби виступає над сторінкою, користувачі «читають» як кнопку і, звичайно ж, намагаються натиснути. Якщо ці очікування не виправдовуються, користувачі вважають, що сталася помилка, і продовжать натискати на «посилання» або «кнопку», навіть не здогадуючись, що це зовсім інші елементи.

4) **Відсутність попереднього перегляду.** Якщо користувач повинен для виконання цільової дії на сайті зробити кілька кроків, йому треба давати можливість переконатися перед прийняттям остаточного рішення, що все зроблено правильно. Для цього здебільшого корисною є опція попереднього перегляду. Якщо взяти приклад з інтернет-магазином, то в цьому разі доречно дати користувачеві можливість побачити всі додані у кошик товари, загальну суму замовлення і персональні дані, щоб він міг перевірити, чи все заповнено правильно.

5) **Замалий шрифт та інші помилки зі шрифтами.** Це доволі поширена помилка при організації інтерфейсу сайту. Раніше 12-й шрифт вважали стандартом, але після появи 24-дюймових екранів, такий розмір літер стало важко читати. Треба розуміти, що для залучення користувачів логічніше використовувати крупні шрифти. Дослідження[9] показали, що є лише 8 секунд на те, щоб зацікавити відвідувача сайту. В середньому люди читають лише 28% тексту на сторінці. Щоб відразу зацікавити користувача, треба:

- створити привабливий і захоплюючий головний заголовок;
- написати цікавий контент, щоб його захотілось прочитати після вступу;
- використовувати крупні шрифти для заголовків. Зараз у тренді[10] спеціальні рукописні шрифти, які створюють унікальний характер і шарм;
- впевнитись, що шрифт основного тексту достатньо крупний і легко читається. Оптимально – 16–18 пікселів. Треба також враховувати особливості вибраного шрифту. Так, 16 пікселів Times New Roman візуально виглядають меншими, аніж 16 Arial.

---

[9] 7 убийственных ошибок дизайна вашего сайта. URL: https://www.trendsmarketing.ru/single-post/8-chastih-oshibok-v-dizayne-saytov (дата обращения 14.09.2018).

[10] 15 Web Design Trends to Watch in 2018. URL: https://blog.hubspot.com/marketing/web-design-trends-2017 (accessed 14.09.2018).

Ще деякі рекомендації щодо шрифтів:
- при доборі шрифтів варто задавати відносні значення, а не фіксовані розміри. Наприклад, 1em замість 14px;
- шрифт веб-сторінки має бути цікавим, але простим для читання;
- на одній сторінці використовувати не більше трьох різних кольорів тексту. При цьому зважати на те, що користувачі надають однакове значення елементам, які візуально виглядають однаково;
- не використовувати тільки прописні (великі) літери, оскільки це знижує швидкість читання й ускладнює сприйняття;
- пам'ятати, що користувачі здебільшого негативно ставляться до тексту, який мерехтить, та до рухливих рядків.

До речі, існують різні плагіни для того, щоб дізнатись інформацію про шрифти, що використовуються на веб-сторінці. Так, плагін WhatFont[11] для Chrome дозволяє перевірити веб-шрифти, просто натискаючи на них.

6) **Низька контрастність тексту**. Текст і фон мають бути достатньо контрастними, найкращий варіант – чорний текст на білому фоні. Варто уникати поєднань кольорів червоний текст на синьому тлі, помаранчевий – на темно-зеленому тощо, оскільки подібні поєднання створюють «ефект тремтіння» і сильно навантажують зір.

Офтальмологи стверджують, що майже половина людей у віці 40 років бачать удвічі гірше, ніж у 20 років.[12] Зважаючи на це, а також, щоб не збільшувати стомлюваність очей читачів даремно, варто користатись правилом: текст має контрастувати – або фон темний і шрифт світлий, або навпаки – фон світлий і шрифт темний.

До речі, застосування контрасту дозволяє привертати увагу. Визнаний експерт у галузі реклами Девід Огілві[4] провів дослідження про сприйняття тексту різного кольору на різному тлі. Дослідження показало, що 70% людей добре сприймають чорний текст на білому фоні, 19% – посередньо і 11% – погано. Про білий текст на чорному фоні лише 12% учасників експерименту дали посередню оцінку, а 88% – погану. 82% респондентів вважають білий текст на фіолетовому фоні поганим рішенням і тільки 2%-м це поєднання сподобалось. Білий текст на синьому тлі є жахливим на думку 96% опитаних. Отже, чорним по білому набагато краще, ніж білим по кольоровому.

7) **Маленький міжрядковий інтервал**. Висота рядка суттєво впливає на візуальне сприйняття і читабельність тексту. Рядки не повинні розташовуватись занадто близько один до одного, інакше вони перетворять текст на суцільне «покривало» літер, яке мало хто захоче читати.

8) **Неправильна довжина рядків**. Довжина рядків є важливою характеристикою читабельності тексту. Задовгі рядки більшість просто не захоче чита-

[11] WhatFont. URL: https://chrome.google.com/webstore/detail/whatfont/jabopobgcpjmedljpbcaablpmlmfcogm (accessed 14.09.2018).

[12] 7 убийственных ошибок дизайна вашего сайта. URL: https://www.trendsmarketing.ru/single-post/8-chastih-oshibok-v-dizayne-saytov (дата обращения 14.09.2018).

ти, а закороткі дратують читача, і він може через це піти зі сторінки. Оскільки розміри екранів відрізняються, треба задавати оптимальну ширину тексту: приблизно від 50 до 75 символів у рядку.

9) **Непоказний заклик до дії**. Ще одна помилка полягає у відсутності колірного акценту в заклику до дії. Розумні інтернет-маркетологи знають, щоб привернути увагу до ключових фраз або заклику до дії, потрібно виділити їх правильним кольором. Наприклад, якщо є кнопки «Отримати зараз» або «Завантажити», доречно їх акцентувати контрастним кольором, щоб зробити їх помітнішими для клієнта. Колір таких об'єктів має бути достатньо яскравим, щоб привернути увагу. Він повинен гармоніювати з іншими кольорами сайту, але відрізнятися від фонового відтінку. Цей колір має використовуватися тільки для закликів до дії.

10) **Довготривале завантаження та відгуки від сервера**. Якщо сайт занадто довго завантажується, більшість відвідувачів вважатимуть за краще покинути його. Якщо ж обслуговувати відвідувачів достатньо швидко, вони будуть повертатися. Якщо сервер часто недоступний, а час відгуку від сервера тривалий, краще знайти інший сервер, який забезпечить швидку роботу сайту.

11) **Відсутність зворотного зв'язку**. Зворотній зв'язок допоможе встановити контакт із користувачем. Можна вказати адресу електронної пошти, телефонний номер для того, щоб зв'язатися.

12) **Нехватка обслуговування**. Як садівник, що вирощує рослину, підживлює її, щоб підтримувати в ній здоров'я на довгі роки, так адміністратор повинен підтримувати свій сайт, наприклад, шляхом додавання сучасних статей, реклами, оновленням дизайну тощо. Треба оновлювати сайт регулярно через певні періоди часу, надавати корисні і свіжі посилання на джерела і ресурси. Нова актуальна інформація дозволить залишити сайт в топі рейтингу, інакше можна втратити аудиторію і репутацію. Саме тому контент має бути свіжим і правдивим, якщо це не так, треба якомога швидше скоригувати його.

13) **Порушення основних принципів структурування сторінок сайту**. Наприклад, рядок пошуку товару має бути вгорі, а не внизу. Логотипи і слогани – у лівому верхньому кутку, а не в іншому місці. Користувачі звикли до такого розташування елементів. Якщо розташувати їх на своєму сайті інакше, це навряд чи сприятиме зростанню кількості клієнтів. Деякі власники сайтів намагаються бути супер креативними і придумують все нові й нові способи відображення меню. Свіжі креативні ідеї – це круто. Але чи сподобається це користувачам? Перш ніж відійти від засад і впровадити якусь нову фішку, варто двічі подумати. Адже основна мета – зробити сайт максимально зручним для користувача!

Втіливши в дизайні сайту саме той сценарій взаємодії, який буде для користувачів логічний і зрозумілий, можна домогтися того, що непорозумінь при роботі із сайтом у користувачів буде менше. А це означає, що сайт зможе працювати ефективніше, даючи вищу конверсію і, як наслідок, – вищий прибуток компанії.

При цьому треба ретельно продумати те, як краще презентувати дані на головній сторінці. Не варто сліпо копіювати і повторювати за конкурентами. Як-

що вказати одну пропозицію на головній сторінці, це зробить сайт (та саму пропозицію) яскравим і помітним. Не варто акцентувати увагу на всьому і відразу. Треба подивитися на сайт очима користувача і зробити його максимально зручним.

6.5. Сучасні тенденції веб-дизайну

У сфері веб-дизайну, як і в моді, усе піддається впливу часу: популярними стають то одні прийоми та методики веб-розробки, то інші, і водночас кожен рік з'являються все нові й нові тренди. Ось і 2018 року теж з'явилися нові тенденції у веб-дизайні, водночас залишилося кілька «довгограючих» трендів.[13]

Отже, якщо казати про торішні тенденції, які залишилися популярними й у 2018 році, то до них можна віднести сторітеллінг, експериментальну навігацію, «ламану» сітку у шаблонах сторінок тощо. Усі ці прийоми у веб-дизайні актуальні і продовжують розвиватися, надаючи можливість розробникам створювати ще більше цікавих і оригінальних рішень.

Якщо ж розглядати нові тренди веб-дизайну 2018 року, то серед них можна виділити еволюцію плоского дизайну, активне використання 3D, VR та AR, а також насичені кольори і градієнти. Крім того, у тренді ще більше персональних фото та ілюстрацій, виразна типографіка[14] і сінемаграфи. Кожну з подібних тенденцій варто розглянути докладніше.

Плоский дизайн у 2018 році став трохи іншим: тепер він «увібрав» у себе тіні і градієнти, властиві Material Design від Google, і став «об'ємнішим». Але водночас плоский дизайн зберіг головну ідею – створення чистих і простих сайтів для комфортного відображення і взаємодії насамперед на мобільних пристроях, частка використання яких неухильно зростає.

Що стосується **використання 3D та VR/AR**, то цей тренд у веб-дизайні не є абсолютною новинкою. Проте він набув чергового поштовху до розвитку завдяки вдосконаленню і значному поширенню відповідних технологій. Використання цього тренду залучає увагу до своїх ресурсів, підвищує рівень юзабіліті і створює приємний інтерфейс.

Насичені кольори й застосування градієнтів – теж не новинка року, проте саме 2018 року ця тенденція «зачепила» експертів і показала значне зростання популярності. У результаті на сайтах, створених або модернізованих цього року, дизайнери намагалися використовувати максимально сміливі колірні рішення, які зможуть запам'ятатися відвідувачам і виділити проекти серед безлічі аналогів.

Сінемаграфи (від. англ. Cinemagraphs) – високоякісні відео або GIF, які працюють на плавному, безперервному циклі. По суті сінемаграфи є статичними зображеннями з частковою анімацією, коли рухається якась частина ілюст-

[13] Тенденції веб-дизайну в 2018 році. URL: http://webstudio2u.net/ua/design-web/962-tendentsii-veb-dizaina-2018.html (дата звернення 28.06.2018).

[14] Типографіка є основою графічного дизайну і полягає в графічному оформленні текстової інформації.

рації або фотографії. І хоча такий тренд, як використання сінемаграфів, знову ж таки, не є винаходом року, зате цей порівняно новий прийом у 2018 році став вельми актуальним і широко застосовується у розробленні дизайну сайтів різної тематики і спрямованості, оскільки за допомогою сінемаграфів можна легко показати користувачам головну ідею сайту або програми, розкрити їхню суть.

Серед інших трендів веб-дизайну 2018 року – акцент на **індивідуальних зображеннях**. Відповідно до цієї тенденції в дизайні сайтів стрімко знижується частка використання стокових фото та ілюстрацій, а замість них дуже активно застосовуються зображення, створені під замовлення спеціально для того чи іншого проекту. Це дозволяє зробити сайт оригінальним, яскравим, виразним.

Цікаво зазначити, що всі згадані вище тренди стосуються переважно візуальної складової веб-дизайну у 2018 році. А втім, не можна забувати, що водночас вони тісно пов'язані з функціональністю сайтів, адже кожен з таких прийомів не просто прикрашає сторінку і демонструє уміння веб-дизайнера, а покликаний створювати позитивний досвід взаємодії із сайтами, допомагати підвищити конверсію, а також посилювати конкурентоспроможність сайтів. І, звісно, для цього дуже важливо користуватися трендами правильно, оцінюючи не лише їхню «красу», а й користь для бізнесу.

6.6. Сучасні стильові рішення у веб-дизайні

З розвитком і повсюдним поширенням веб-технологій змінюється сприйняття користувачем дизайну й оформлення контенту, змінюються дизайнерські погляди, але в цілому популярність сучасного «чистого» веб-дизайну продовжує зростати. У моді лаконічність оформлення і простота візуальних рішень.

Нинішній 2018 рік у веб-дизайні використовує контрасти і поєднує різні стилі та напрямки. Оформляючи веб-сайт великими зображеннями, сучасний дизайнер вільний додати ретро-патерни, кольори і типографіку.[15] Можна використовувати 2D-оточення у тривимірному дизайні або статичні ілюстрації з ненав'язливими ефектами. Візуальні контрасти дозволяють змінювати веб-дизайн. Прагнучи до ясного розмежування, дизайнери тонко поєднують вишукані штрихи з об'ємними формами.

Популярні елементи і стилі сучасного веб-дизайну:

1) індивідуальні ілюстрації займають центральне місце;
2) великі заголовки, які перекривають графічний контент;
3) асиметричні макети і «ламана» розмітка;
4) Duotone та оверлей поверх зображень;
5) сучасне ретро в ілюстраціях і елементах;
6) брутальний веб-дизайн;
7) стильні текстури у фоні;

[15] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).

8) градієнти 2.0;
9) продумані анімації та ефекти UI;
10) анімована SVG-графіка;
11) шари кольору;
12) ізометрія у дизайні;
13) органічні форми;
14) відсутність меж;
15) великі тіні;
16) яскраві фарби;
17) розділений екран;
18) дизайнерські шрифти;
19) комбінація горизонтального і вертикального тексту;
20) вбудоване відео.

Розглянемо докладніше ці елементи і стилі сучасного веб-дизайну з прикладами їхнього застосування у композиції сучасних сайтів.

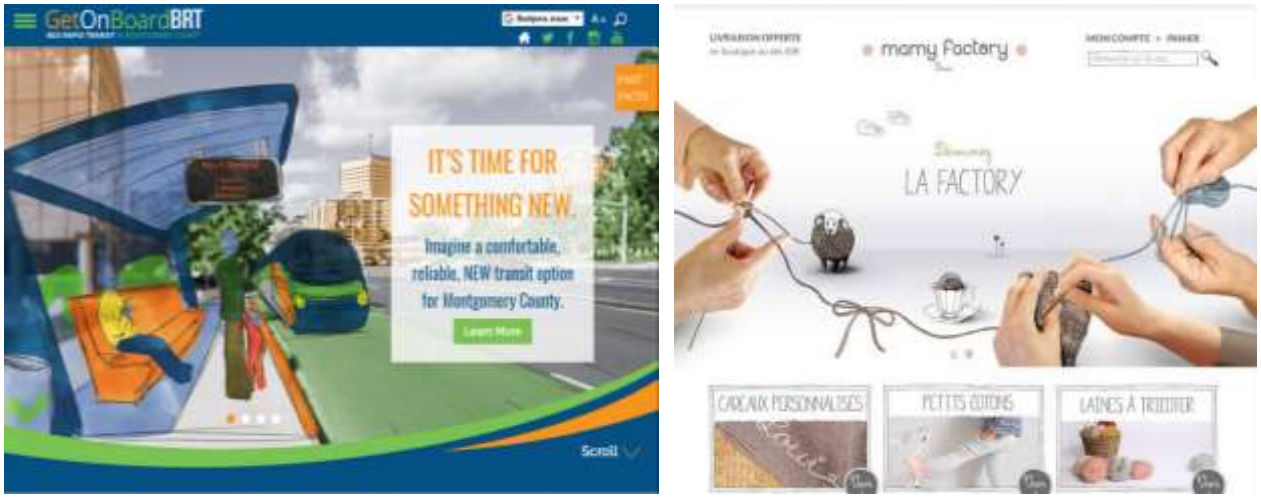
1) **Індивідуальні ілюстрації займають центральне місце**. Ілюстрації виходять на передній план веб-дизайну, допомагаючи показати історії, які прагнуть розповісти компанії. Перевагою цього є здатність залучати різноманітну аудиторію. Новацію вже використовують енергійні підприємства, оскільки це допомагає у створенні і поширенні серйозних якісних брендів.

Унікальна ілюстрація у фоновому режимі здатна створити особливий настрій і навіть вплинути на естетику сайту. Творчо проілюстровані комерційні сайти користуються популярністю, оскільки їхні прагнення до індивідуальності не лишається без уваги. До речі, індивідуально створені зображення ігрового дружнього характеру на сайті додають елемент розваг, що додатково приваблює користувачів ресурсу.

Естетичні сайти з унікальними ілюстраціями[16]:

– https://getonboardbrt.com вдало застосовує акварельні ілюстрації в плоскому дизайні;
– http://www.bostonbrt.org – бостонський сайт інноваційної транспортної системи використовує стильні ілюстрації аквареллю;
– http://www.zingle.me – ілюстрація сайту заохочує до спілкування;
– http://www.mamyfactory.com привертає цільового відвідувача мальованим зображенням процесу в'язки;
– https://lattice.com реалізував заклики до дії позитивними мальованими ілюстраціями;
– http://www.flowmapp.com – ілюстрація сайту-органайзера налаштовує на гармонію і навіює затишок;
– https://www.stride.com – вже сам дизайн сайту залучає до творчості.

[16] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).



2) **Великі заголовки, які перекривають графічний контент**. Не секрет, що багато сайтів виглядають нуднувато і 2017 року з'явилося несподіване рішення – **перекривний контент**. Це різні перетину тексту з зображеннями, плашками, блоками. Застосування великих і жирних шрифтів демонструє впевненість дизайнера, а відвідувачеві дозволяє поглянути на сайт із різних точок зору. На рис. 6.1 показано три приклади дизайнерських рішень використання тексту, який перекриває графічний контент.

Рисунок 6.1 – Сучасний стиль веб-дизайну з ефектом перекриття

Дизайни з нашаруванням і перетином контенту[17] (рис. 6.1):

– https://www.elegantseagulls.com – сайт агентства із суто чорно-білим дизайном і нестандартними підходами;
– https://www.carbonbeauty.com – мінімалістичний дизайн, заголовок зміщується донизу при прокручуванні і проходить водночас і за, і перед блоками;
– https://www.thibaultpailloux.com – розробник урізноманітнив тренд, прикрасивши переходи по своєму сайту різними градієнтами;
– https://www.jansport.com – стильні величезні заголовки з потертостями або з обведенням поверх Hero-зображення;[18]
– https://www.spotify.com – величезні, дещо брутальні літери заголовка розміщуються по центру зображення з перекриттям фігур на зображенні. До речі, тут використовують крапки у заголовках.

3) **Асиметричні макети і «ламана» розмітка (broken grid)**. «Ламане» розміщення елементів – веб-тренд, який стрімко набирає популярність. Для його реалізації не потрібні екстра ресурси розробки.

Нестандартну сітку застосовують, щоб сайт не був схожий на інші зі своєї ніші. Разом з напрямком асиметричного дизайну, стиль нашарування елементів породжує особливу естетику і свої правила. Нетипова, як і будь-яка інша верстка, потребує підтримку візуального балансу. Тут треба планувати перетини так, щоб врівноважити асиметричну структуру оформлення сайту.

Асиметрія в дизайні сайту дозволяє:[19]

– «оживити» цільову сторінку, сконцентрувавши увагу користувачів на найважливіших речах, зробивши візуальний акцент на різних елементах сайту;
– полегшити сприйняття графічної інформації;
– ефективно і вигідно використовувати вільний простір.

---

[17] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).

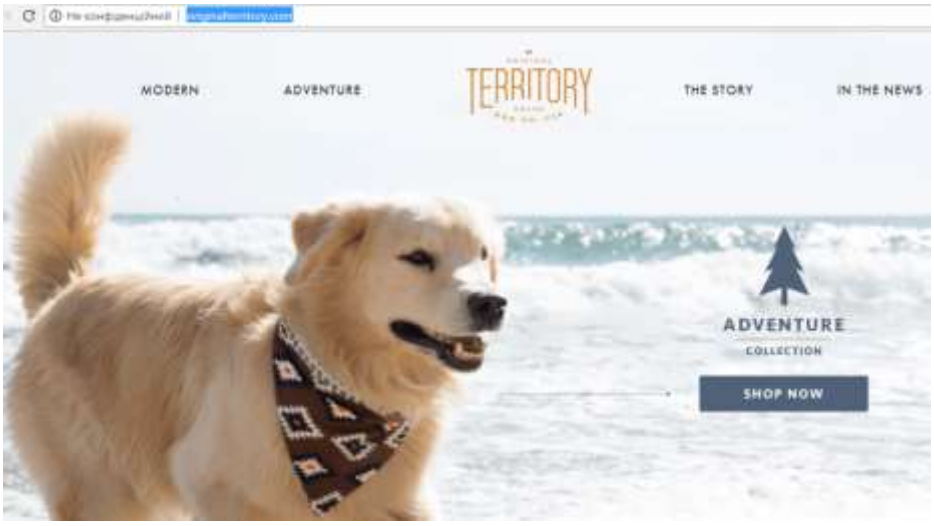
[18] Hero image – це термін, який використовується у веб-дизайні для банерів. Hero image по суті є банером у вигляді великого зображення або фотографії.

[19] Асимметрия в дизайне сайта. URL: http://seo-design.net/design/asymmetry-in-design (дата обращения 28.06.2018).

Відсутність лінії симетрії допомагає природньо відокремити навігаційне меню або сайдбар[20] від решти контенту, залишаючи більше місця для розміщення корисної інформації. Сама обстановка асиметричності добре підходить для емоційного дизайну, використання персонажів, зображень рекламного характеру (брендових товарів), які створюють позитив і підсилюють емоційні мотиви. В цілому асиметричний баланс позитивно впливає на здатність висловлювати за допомогою дизайну різні емоції, створюючи відчуття динамічності.

Дизайни з асиметричною розміткою і нашаруванням елементів:[21]

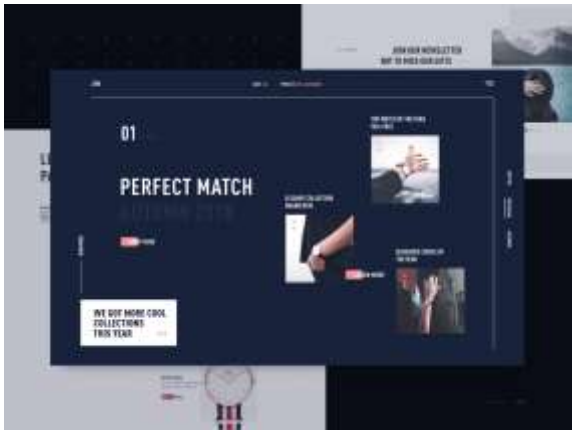
– https://www.welikesmall.com – ультрасучасний дизайн digital-агентства використовує «ламану» сітку з асиметричними перетинами елементів. Круті ефекти і мікро-анімації інтерфейсу;
– https://techstylefashiongroup.com – чорно-біло-синій дизайн з анімацією і паралаксом. Напівпрозорі плашки поверх фото з реалістичними тінями перекривають собою сині круги, наїжджають на гранжеві фони. Множинне нашарування асиметричних flat-елементів створює обсяг, а вертикальний вільний простір додає креативу і стильності;
– http://originalterritory.com – приємний сайт «Територія собак» з асиметричною розміткою, м'яким і компактним сучасним дизайном;



[20] Сайдбар (sidebar) – це бічна колонка сайту або блогу, в якій розміщують додаткову інформацію, наприклад, докладний перелік рубрик на сайті.

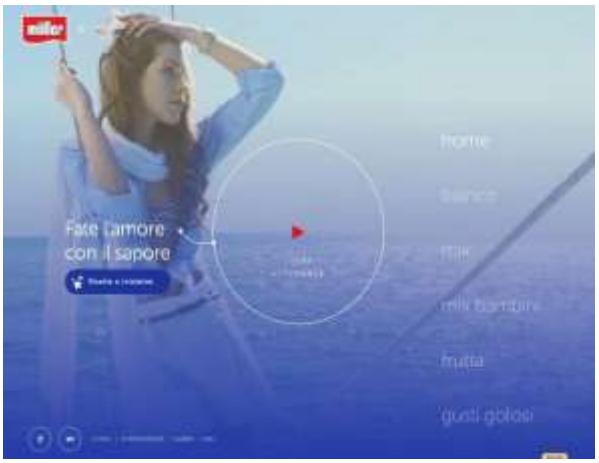
[21] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).

– http://www.imsproductions.com – гарне застосування тренду на сайті американської продакшн-компанії;



– https://www.heeds.eu, http://www.veintidosgrados.com – яскравий і нестандартний дизайн. Двокольорова корекція зображень, градієнти, «ламана» розмітка.

4) **Duotone (дуплекс) та оверлей поверх зображень**. Застосування ефекту подвійної експозиції фотографій з двотоном і накладенням (перекриттям) шару кольору є сучасним естетичним трендом для дизайну сайтів і ресурсів із медіа-контентом.



Приклади вдалого використання цього тренду:[22]

– https://www.muller.it – сайт італійського продуктового бренда з вдалим оформленням двотоновим відеобекграундом з додаванням м'якого градієнта;
– https://cliquestudios.com – ресурс дизайн-агентства, на головній сторінці якого кейси розмежовані колірними фільтрами (overlay);
– https://www.socialplayground.com.au – ресурс соціальних медіа та подій доброчесності використав для оформлення сайту двотонові фото, що надає емоційності та створює відчуття динамічності.

[22] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).

5) **Сучасне ретро в ілюстраціях і елементах.** Стилізація під старовину у дизайні уживається з новими тенденціями і залишається актуальним напрямом. У свідомості споживачів вінтажне оформлення часто асоціюється з високою планкою якості. Для одних ретростиль– синонім розкішного і дорогого. Інші сприймають вінтажний сайт як жвавий і повний енергії. У ретродизайну є чимало поціновувачів, оскільки гарно подана композиція з ретро зачаровує. Ще одна конкурентна перевага: дизайн з елементами минулого викликає ностальгічні нотки. З властивою вінтажному стилю збільшеністю елементів він цілком вписується у мінімалістичну концепцію сучасного веб-дизайну.

Прикладом сучасного сайту з елементами ретро-стилю є http://sugarfirepie.com, де вінтажна ілюстрація на головному екрані створює перше враження про кондитерський бренд, покращує UX-дизайн[23] сайту.

6) **Брутальний веб-дизайн.** Щоб виділитися з океану охайних організованих сайтів, дизайнери використовують грубі і незграбні структури. Бруталізм виник як реакція на зростаючу стандартизацію веб-дизайну, і часто характеризується різкими, асиметричними, неконформістськими візуалами та виразним недоліком ієрархії та порядку. Стиль бруталізму у веб-дизайні доволі різноманітний. По суті це незавершений дизайн і втілення ідеї мінімалістичного контрасту. Багатьох приваблює брутальна естетика: гранж[24], неотесаність, аскетизм. Є думка, що в певних нішах брутальний веб-дизайн підвищує конверсію сайту.

Стиль бруталізму застосовували у веб-дизайнах такі успішні веб-ресурси:

– https://www.bloomberg.com – популярний бренд розмістив агресивні елементи на своєму сайті;
– https://mahzedahrbakery.com виглядає простим мінімалістичним сайтом кондитерської фірми у Нью-Йорку. Судячи з анонсованої продукції, фірма використовує бруталізм і в дизайні своєї смачної продукції;

[23] UX скорочено від User eXperience – дизайн взаємодії. Останнім часом роль дизайну взаємодії (User Experience Design) зросла від простого зосередження уваги на визначенні компонентів для інтерфейсу користувача до багатодисциплінарної гілки дизайну, яка враховує численні технічні аспекти від анімації до програмування при розробленні контекстних інтерфейсів.

[24] Гранж (від англ.grunge) – заперечення загальноприйнятих норм зовнішнього вигляду.

– https://biannual.com – модний бренд одягу використав у стилі сайту певну брутальність шрифтів та гранж у фото;
– https://www.balenciaga.com застосував веб-бруталізм, який критик назвав мінімалізмом на стероїдах;[25]
– http://gift.gucci.com проявляє інтерес з вигадкою до бруталізму у веб-дизайнерському середовищі для свого зоряного бренда.

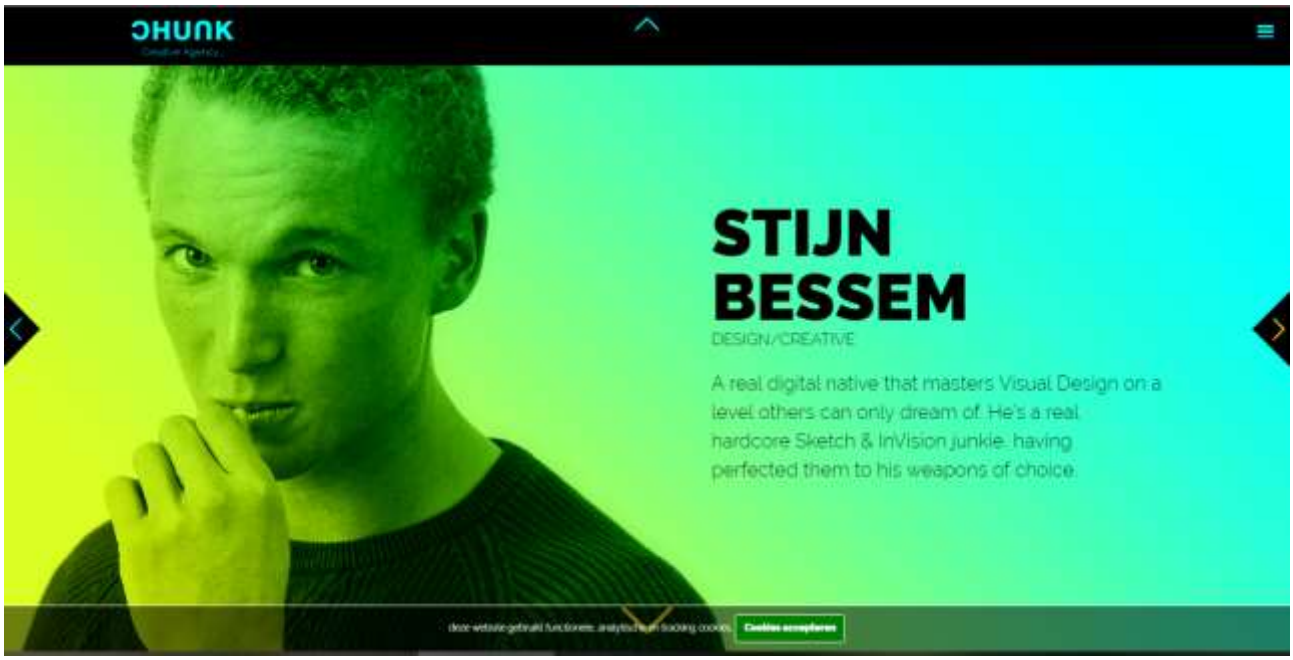
7) **Стильні текстури у фоні**. Подібно до того, як у дизайні інтер'єру шпалери то входять в моду, то виходять з моди, – зараз спостерігається відродження фонових текстур. Знову у тренді плиткові фони і бекграунд з повторюваними елементами (geometric shapes and patterns). Творчо підібрані текстури надають сайту персональності. Також стильний бекграунд може відмінно гармоніювати з нестандартним формами елементів (наприклад, іконками) або користувацькими ілюстраціями.

[25] Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).

Сучасні веб-дизайни творчо застосовують фонові текстури:

– https://simone-simon.fr – стильний французький сайт використовує ефектний дизайн з накладенням шматків текстур, порожнистих фігур, великих літер;
– https://abcdentaltexas.com – незвичайне для сайту стоматології мультяшне оформлення з гарною текстурою фону;
– http://www.djeco.com/ru – картатий фоновий патерн, неначе зі сторінок шкільного зошита, та інші текстури у привабливому мультяшному дизайні сайту компанії;
– http://fontsofchaos.com – геометричні фігури і візерунки (патерни) забезпечують варіації та унікальність дизайну веб-сторінки.

8) **Градієнти 2.0.** 2017 року градієнти повернулись у веб-дизайн з більш чистими насиченими кольорами (без домішку брудних відтінків). Можливо це не надовго, але у 2018-му тренд кріпне і використовується у всьому розмаїтті. Характерні ознаки модного напряму: колорит, жвавість, сайт із градієнтами 2.0 виглядає свіжим і сучасним.



Приклади сайтів з яскравими відтінками і градієнтами у дизайні:

– https://wpengine.com – WordPress-хостинг оформив градієнтами випадне меню і форми підписки;
– https://chunkagency.com – сайт креативного агентства з двоколірним неоновим градієнтом і ефектними переходами;
– https://www.fullstory.com – двоколірні градієнти застосовуються для кнопок та у фонах;
– https://y7k.com – Цюріхське агентство Y7K ілюструє прекрасний приклад того, як двотоновий ефект виглядає свіжим і сучасним на повноекранній, градієнтній домашній сторінці.

9) **Продумані анімації та ефекти UI.** Анімація у веб-дизайні виконала довгий шлях і застосовувалася ще 1999-му для створення агресивних flash-банерів «Клікни мене!»[26] Продумані мікроітерації виглядають стильно і покращують користувацький інтерфейс, цікаві анімаційні сценарії можуть розважати і слугувати навігаційним центром, візуально позначивши інтерактивну взаємодію з користувачем. Без перебільшень – аудиторія прихильників у такого ефектного і корисного тренду широка.

Приклади сайтів із добре продуманою анімацією:

– https://www.impactblacksheepagency.com – креативний сайт Heco Partners, створений для агентства Black Sheep, використовує чудовий ефект побудови навігаційної системи – під час прокручування вниз робиться анімаційний акцент на поточному реченні, що допомагає спрямувати увагу відвідувачів на певний вміст у потрібний час;
– http://insuranceexperiments.org.uk – цікавий промо-сайт британської страхової компанії в стилі Flat[27] з плоским відео і мультяшними анімаціями;
– https://www.aproposducancer.fr – сайт ліонського хоспісу з фантастично ілюстрованим анімованим дизайном, цікавим прелоадером,[28] інтерактивом, скролінгом, UI;
– https://bloom.co – цільова сторінка (Landing Page) ресурсу містить анімацію з високою деталізацією та ефектне навчальне відео, що залучає до співпраці і налаштовує на позитив;
– https://district0x.io – сучасний напівплоский дизайн, анімований по максимуму: креативні хедер / футер, хмари у тлі карток і самі картки;
– http://www.helloheco.com – рухлива хвиля фону створює незабутній ефект;
– http://www.designbetter.co – головна сторінка сайту дизайнерської компанії містить стильний анімований фон;
– http://heystack.is – мультяшна анімація сайту серйозної економічної платформи для оптимізації бізнес-операцій налаштовує довірливу комунікацію з клієнтами.

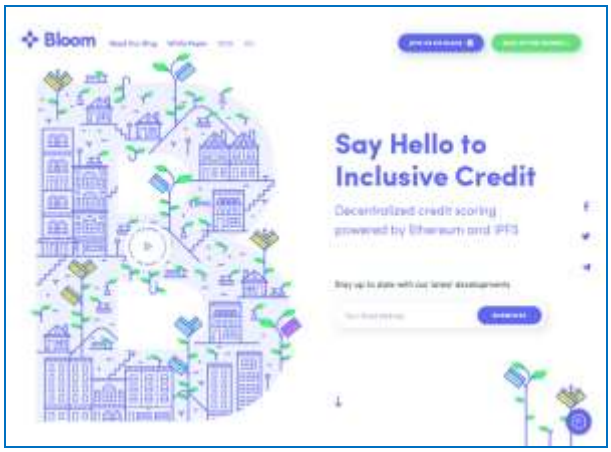
Веб-сайти, які працюють із фоновим відео, здебільшого організовують їх як легкі анімації засобами Javascript. Це дозволяє без зайвого завантаження створювати рухи як природну частину фону.

---

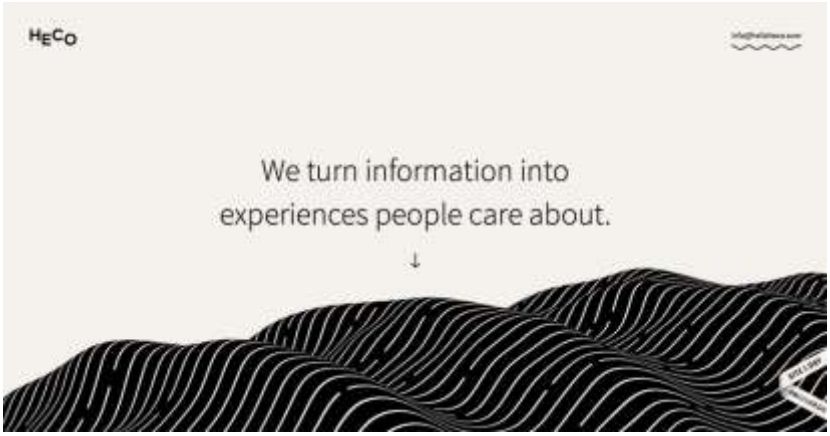
[26] UI дизайн (англ. User Interface Design – інтерфейс користувача) – поняття, яке охоплює певний набір графічно оформлених технічних елементів (кнопки, чекбокси, селектори та інші поля). Його завдання – допомогти користувачеві організувати взаємодію із сайтом. На сьогодні є певні правила UI дизайну:
– організованість елементів інтерфейсу, які повинні бути логічно структуровані і взаємопов'язані;
– угруповання логічно пов'язаних елементів інтерфейсу в групи (меню, форми);
– вирівнювання елементів інтерфейсу;
– єдиний стиль елементів інтерфейсу;
– наявність вільного простору, що дозволяє розмежовувати інформаційні блоки;
Розроблений за цими правилами інтерфейс значно підвищує ефективність ресурсу і надає йому конкурентні переваги.

[27] Плоский дизайн (англ. Flat Design) – дизайн інтерфейсів програм та операційних систем, поданий як протилежність реалізму. За задумом «плоский дизайн» повинен підкреслювати ефект «чарівної простоти» і вишуканості.

[28] Індикатор завантаження сайту.



Фон із частинок – це чудове вирішення проблем, пов'язаних із продуктивністю. Кажуть, що рухливе зображення говорить більше тисячі слів. Так само фони із рухомими частинками відразу привертають увагу користувача, тим самим можуть створити незабутнє враження про бренди лише через кілька секунд.

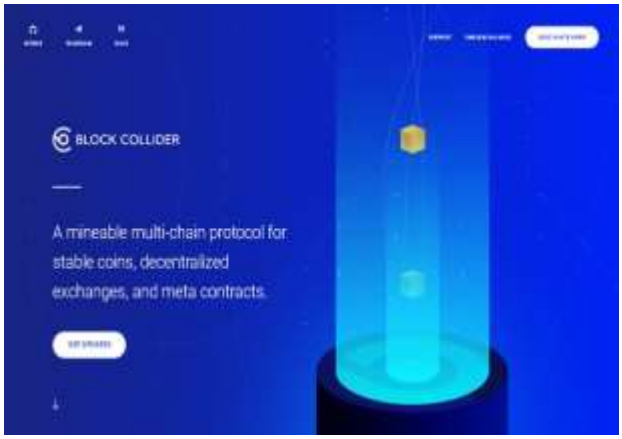


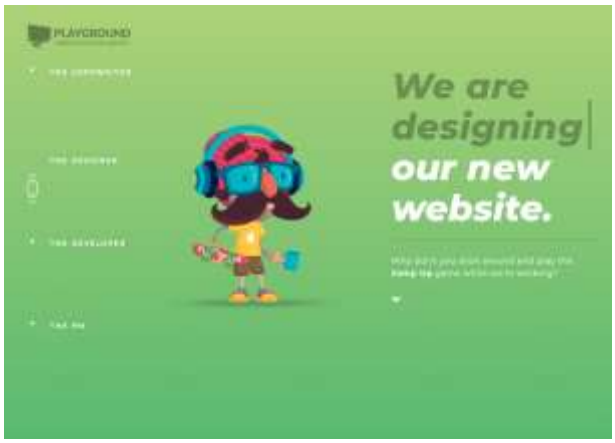
10) **Анімована SVG-графіка.** Важливі переваги масштабованої векторної графіки (SVG[29]) перед растровими форматами: зображення масштабуються без втрат якості, легко редагуються засобами CSS і важать мало. Технології HTML5 дозволяють інтегрувати гнучкі SVG-елементи у веб-сторінку. Із сервісами на кшталт svgator.com можна створювати SVG-анімації без знання кодингу.

Приклади ефектного застосування SVG-графіки:

– http://blockcollider.org – сучасний сайт із SVG-ефектами та анімацією;
– https://www.boite-a-oeufs.com – креативне агентство зробило свій сайт на WordPress, використовуючи SVG з анімацією на GSAP платформі;
– https://www.playground.it – сторінка-«заглушка» ресурсу на час його редизайну. Анімовані SVG-ілюстрації роблять її просто чарівною. А можливість зіграти в інтерактивну гру Jump UP затримує користувача, призвичаює до сайту та заохочує до подальшої співпраці.

[29] SVG (від англ. Scalable Vector Graphics – масштабована векторна графіка) – це формат файлів для двовимірної векторної графіки, як статичної, так і анімованої та інтерактивної.





Існує багато прикладів доволі простого програмного CSS-коду для анімації з використанням SVG.[30]

11) **Шари кольору**. Поєднані накладені шари кольору додають глибини та текстури простому макета сайту.

Стильні приклади застосування яскравих нашарувань у веб-дизайні:

– https://huckberry.com створив ефективну схему сайту з кольорових шарів;
– https://www.bauhaus.de/en поєднав кольори у різних шарах, що надало емоції дизайну веб-сайту;
– http://meiofio.cc – команда Мейо-Фіо із Сан-Паулу завдяки яскравим і структурованим шарам рамок створила на сторінці стильний невимушений настрій;
– https://theoutline.com – нашарування різних яскравих рамок посилює сприйняття та створює глибину і завершеність.

12) **Ізометрія у дизайні.** Вірогідно, повернення ізометричного дизайну стало реакцією на flat-експансію по всьому світу. Цей стиль оформлення достатньо трудомісткий. Для 3D-візуалізації у двовимірному просторі веб-сторінки потріб-

[30] 7 CSS-анимаций с использованием SVG + сайты применяющие эффект. URL: http://seo-design.net/css-markup/css-animations-with-svg-websites-uses-it (дата обращения 11.09.2018).

на робота з графічними елементами. З іншого боку, ізометричні іконки і графіка – це гарний спосіб показати свою індивідуальність.

Приклади сучасних ізометричних дизайнів:

– https://www.pikmykid.com – приємний веб-дизайн у м'яких тонах з плавними ефектами, ізометричними ілюстраціями, фіксованими полігональними фонами і гарними кнопками з градієнтним фоном;
– https://www.brightscout.com – зелений дизайн сайту компанії гарно проілюстрований, використаний напівплоский стиль з тінями, градієнтами та скрол-ефектами;
– https://mibexsoftware.com – сайт софт-компанії із сучасним плоским ізометричним дизайном з фірмовим персонажем.

13) **Органічні форми (Organic Shapes, Oblong Shapes)**. Пройшли дні суворих схем сітки і гострих країв прямокутних екранів. 2018 року в моді вигнуті лінії і м'які натуральні форми.[31]

Фігури з закругленими кутами, зокрема, довгасті форми, надають привабливості та індивідуальності дизайну.[32]

Приклади сайтів із м'якими, органічними формами:

– https://neobi.it – мультяшний фон додає благородства, індивідуальності

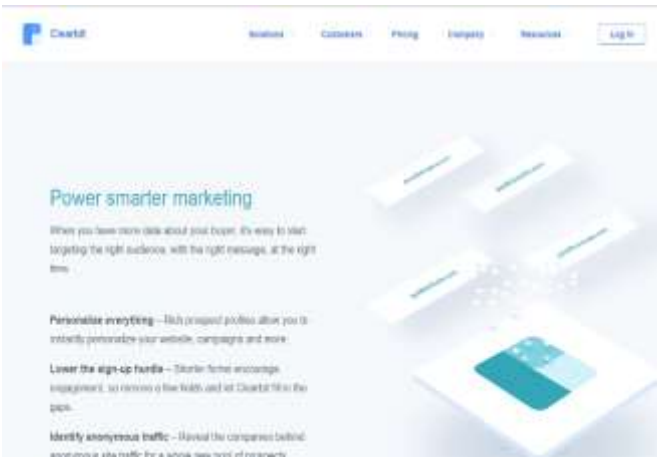
[31] Top 10 Current Web Design Trends for 2018. URL: https://blog.brightscout.com/top-10-current-web-design-trends-2018 (accessed 14.09.2018).
[32] 19 web design trends for 2018. URL: https://webflow.com/blog/19-web-design-trends-for-2018 (accessed 18.09.2018).

та яскравості нескладному дизайну;
– http://www.cdu.org.uy/duy – стильні вигнуті лінії використали як межі анімації.

14) **Відсутність меж**. Безкордонний дизайн пропонує альтернативу традиційному підходу. В минуле відходить розбивка сторінки різноманітними фоновими кольорами або повнорозмірними зображеннями. Замість цього один колір тла простягається від верхньої частини донизу сторінки та створює бездоганний вигляд. Використовуючи цей дизайн, око спрямовується на сторінку без будь-яких перерв. Це чудово працює з адаптивним дизайном, оскільки не потрібно турбуватися про будь-які фонові елементи.[33]

Приклади дизайну без полів для чистого та елегантного відчуття на веб-сторінці:
– https://www.algolia.com – біле полотно веб-сторінки створює простір з вільними комбінаціями розташування текстових і графічних полів;
– http://funemployed.life – демонстрація відсутності меж у дизайні веб-сторінки зумовлена спрямованістю сайту на аудиторію фрілансерів – нового покоління з вільним простором життєдіяльності та заробітку.

15) **Великі тіні**. Хоча тіні були деякий час табу у веб-дизайні, особливо у плоскому дизайні. Нині тіні повертаються на веб-сторінки, їх використовують все більше, створюючи цікаві варіанти дизайну. Завдяки прогресу веб-браузерів, використання тіней може зробити елемент цікавішим, одночасно створюючи акценти. Тіні дозволяють дизайнерам створити глибину та ілюзію світу поза екраном.

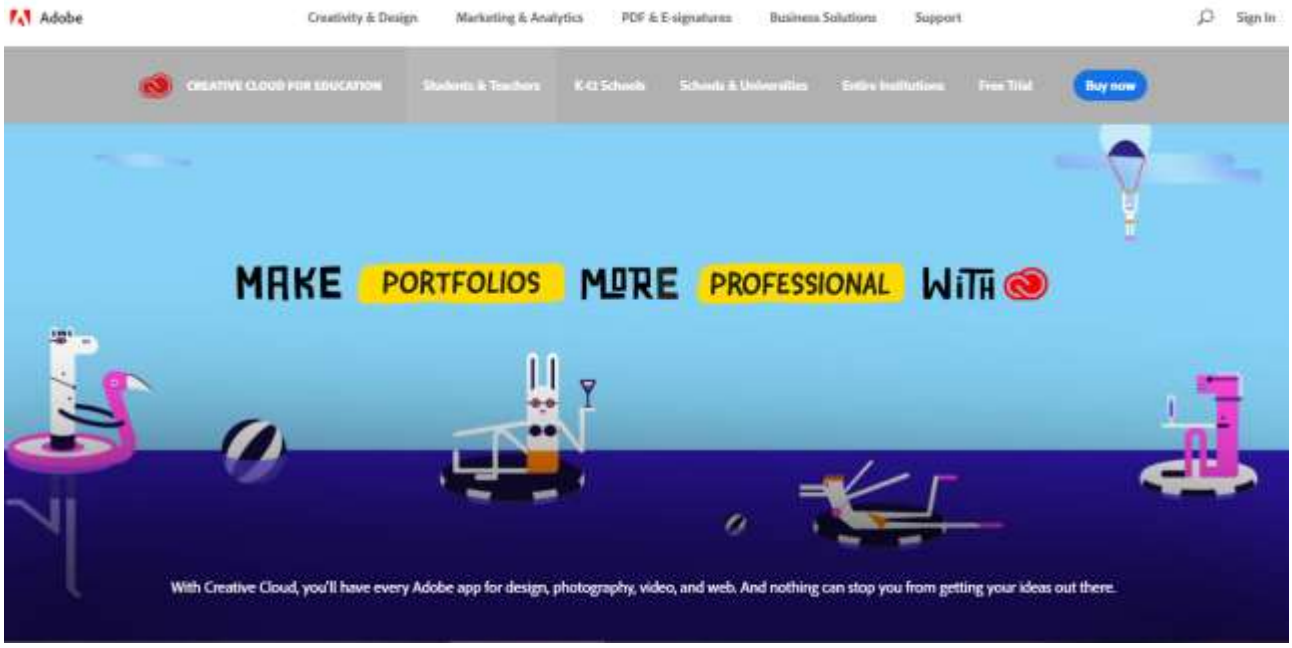


Приклади дизайну з «грою у тіні»:
– https://www.heeds.eu – яскравий і нестандартний дизайн з тіньовою корекцією зображення і градієнтами;
– http://www.algolia.com – гра тіней створює неймовірно універсальний ефект, що збільшує естетику веб-сторінки;
– http://www.clearbit.com – великі тіні створюють тривимірний ефект у доволі простому дизайні;

[33] Top 10 Current Web Design Trends for 2018. URL: https://blog.brightscout.com/top-10-current-web-design-trends-2018 (accessed 14.09.2018).

– http://www.scaleapi.com використовує м’які тіні, які посилюються при наведенні мишею на об’єкт, що дає зрозуміти позначення посилання і допомагає користувачеві (UX), надаючи акцент.

16) **Яскраві фарби**. Колір – це прекрасний інструмент дизайнерів для створення неповторних вражаючих сторінок. Веб-кольори стають більш авантюрними з яскравими та блискучими відтінками. Прогрес у використанні яскравих фарб спричинила поява більш якісних екранів смартфонів із високою роздільною здатністю, які здатні відображати все більше розмаїття кольорів. Ці яскраві фарби можуть бути використані, щоб миттєво привернути увагу та відрізняти веб-сайт від своїх конкурентів.[34] Крім того, вони дають більше свободи при створенні дизайну.



Використання яскравих фарб у дизайні:

– https://www.adobe.com – сайт відомого бренда має безліч яскравих сторінок з цікавими рішенням дизайну;
– https://www.spotify.com – сайт для пошуку і відтворення музикальних треків дивує використанням яскравих кольорів та величезних брутальних літер;
– http://egwineco.com – задля привернення уваги у рекламі лінійки напоїв бренд застосовує яскраві чисті кольори.

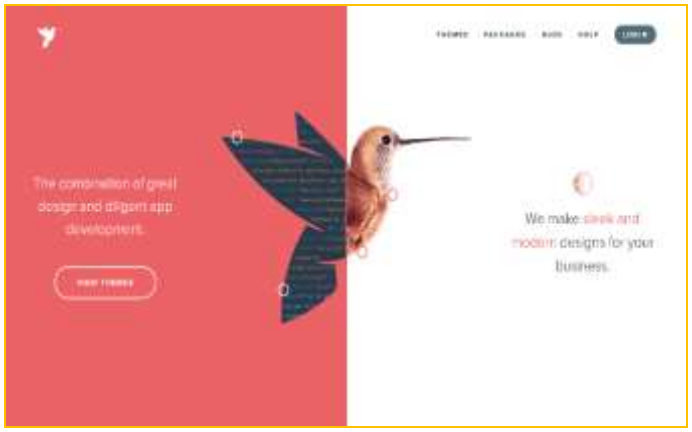
17) **Розділений екран**. Мода на розділення веб-сторінок навпіл з’явився в результаті еволюції Hero-зображень. Вертикальне розділення дозволяє задавати різні стильові елементи. Спліт-екрани надають відмінні можливості для просування контенту.

Сайти з ефектним вертикальним розділенням екрана:

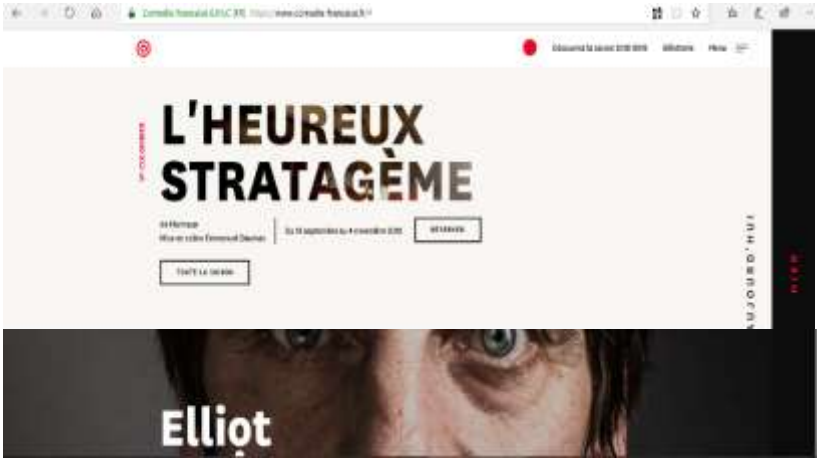
– http://special.bose.eu/fr – логічне розділення контенту на головній сторінці, яскраво подає продукцію Bose і спрощує навігацію по сайту;

[34] 9 cutting-edge web design trends for 2018. URL: https://99designs.com/blog/trends/web-design-trends-2018/#mobile-first (accessed 17.09.2018).

– https://www.enginethemes.com – розділений навпіл елемент фірмового стилю у вигляді стильної пташки в центрі головного спліт-екрана презентує компанію веб-розробників на WordPress;
– https://bigeyeagency.com – нестандартна навігація з розділенням екрана для кейсів, фонове відео з текстурою і фотоефекти у бекграундах (висвітлення, розмиття, боке,[35] подвійна експозиція);
– https://chekhov.withgoogle.com – розділеним контентом подано опитування «Впізнай себе у творах Чехова»;
– https://dropbox.design – стильні яскраві половинки дозволяють структурувати інформацію на сайті.



18) **Дизайнерські шрифти**. Візуальна міць типографіки завжди використовувалася для пробудження емоцій, додавання сайту індивідуальності та передавання важливого посилу. Зараз, коли роздільна здатність екранів стає все більшою, а картинка чіткішою, можна очікувати більшого застосування користувацьких дизайнерських шрифтів. Більшість браузерів вже підтримують CSS стилізовані нестандартні гарнітури. Трендом стають великі літери, шрифтові контрасти sans serif & serif,[36] які поліпшують UX-дизайн, а головне спонукають читати.

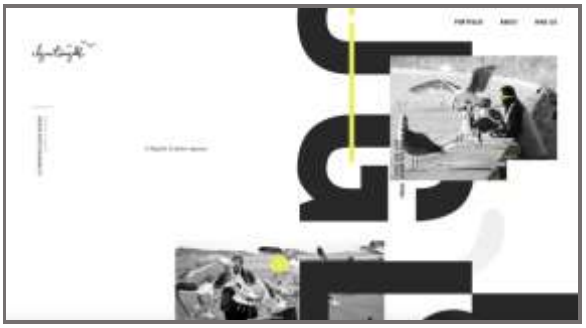


[35] Боке (від яп. ボケ – розмитість, нечіткість) – навмисне розмивання фону зображень, що надає суб'єктивні художні переваги частині зображення для візуального виділення головного об'єкту зйомки.
[36] Рубані шрифти та шрифти із зарубками (від англ. serif – засічка, зарубка та фр. sans – без) – два сімейства шрифтів. Донині за правило було використовувати Sans-Serif для онлайн версії, а Serif для друкованих видань.

Нині дизайнери[37] в повній мірі користуються цим трендом для ефектного оформлення заголовків:

– http://www.nurturedigital.com використав велику незвичайну типографіку з анімацією;
– https://www.ge.com/digitalvolcano поєднав креативні шрифтові гарнітури;
– https://hypedmarketing.co.uk/services/websites – дизайнерські шрифти над рухомим зображенням використані для акценту та естетичного ефекту;
– http://www.danbury-ct.gov – міський сайт використав розмаїття наявних шрифтів, що привертає увагу користувача;
– https://www.comedie-francaise.fr – сайт комедійної театральної студії з ефектом типографіки при наведенні на текст;
– http://www.nurturedigital.com демонструє нео-гротескні стилі шрифтів для створення динамічної взаємодії з користувачем;
– https://www.kikk.be/2017 – сайт фестивалю KIKK[38] задіяв унікальний рукописний шрифт, що надало характер і шарм, створило чіткий вигляд.

19) **Комбінація горизонтального і вертикального тексту**. Змішування горизонтального і вертикального тексту стало освіжаючим підходом, який відрізняється від умов горизонтального вирівнювання. Такі шрифти привертають увагу веб-відвідувачів до вмісту відповідного повідомлення завдяки незвичайному поданню.



Приклади стилістичного підходу до розміщення блоків тексту:

– https://magicpeoplevoodoopeople.com – компанія Magic People Voodoo People використав чергування горизонтальної і вертикальної орієнтації тексту;

[37] 12 huge web design trends for 2018. URL: https://www.creativebloq.com/features/web-design-trends (accessed 01.12.2018).
[38] 15 Web Design Trends to Watch in 2018. URL: https://blog.hubspot.com/marketing/web-design-trends-2017 (accessed 01.12.2018).

– https://takewhatyoucancarry.com – сайт фільму режисера Метта Портерфілда «Take That You Can Carry» порушив горизонтальне вирівнювання тексту за допомогою одного слова;
– https://www.elegantseagulls.com – дизайнери агентства Elegant Seagulls, завдяки експерименту з вертикальним вирівнюванням тексту, створили стійкий ефект прокрутки.

20) **Вбудоване відео** – це тенденція, яка досі переважає в онлайн-світі. Це зумовлено тим, що люди вважають за краще дивитися відео, а швидкі інтернет-з'єднання зняли усілякі обмеження на використання вбудованого фонового відео у веб-дизайні. Через відео бренди можуть ефективно передавати велику кількість інформації в невеликому просторі та за короткий проміжок часу.[39] Крім того, вбудоване у веб-дизайн відео може залучити відвідувачів, допомогти збільшити час перебування на сторінці, а також допомогти у створенні навігації.

Приклади сайтів із вбудованим у веб-дизайн відео:
– https://www.adobe.com – яскраві вбудовані відео демонструють дизайнерські новації фірми;
– http://egwineco.com – фонове відео створює настрій і при цьому не навантажує і не відвертає від мети сайту – реклами напою.

6.7. Zero UI у веб-дизайні

6.7.1. Поняття та складові Zero UI

Рух до мінімалізму у веб-дизайні з'явився доволі давно, проте нині парадигма дизайну змінюється докорінно, оскільки важливою і затребуваною характеристикою сайтів стає не стилістика сайту, а швидкість надання персоналізованої інформації та послуг користувачу.

Якоб Нільсен[40] сказав: «Вивчення поведінки користувачів у веб показує, що вони погано сприймають повільні сайти та сайти зі складним дизайном.

[39] Top 10 Current Web Design Trends for 2018. URL: https://blog.brightscout.com/top-10-current-web-design-trends-2018 (accessed 14.09.2018).

[40] Якоб Нільсен (англ. Jakob Nielsen) – один з провідних консультантів з веб-юзабіліті, доктор наук наукового напрямку людино-комп'ютерної взаємодії Технічного університету Данії (Technical University of Denmark) у Копенгагені, засновник компанії Nielsen Norman Group. Нільсена називають королем юзабіліті, гуру юзабіліті веб-сторінок, новим піонером медіа.

Люди не прагнуть чекати. Також вони не прагнуть вивчати, як користуватися домашньою сторінкою. Не існує таких речей, як навчання веб-сайту або інструкція з веб-сайту. Люди прагнуть схопити функціональність сайту відразу ж після швидкого сканування сторінки, тобто за кілька секунд».[41]

Дослідження Forrester Research[42] показали,[43] скільки втрачають компанії через поганий юзабіліті сайтів: інтернет-магазини втрачають приблизно 50% покупців, які не можуть знайти потрібний товар. До 40% користувачів не вертаються на сайт, з яким мали негативний досвід роботи.

Пристрої, створені людьми для повсякденного користування, пройшли шлях еволюції елементів керування: фізичні кнопки, машинний код, командний рядок, графічний інтерфейс, тачскрін. Усі ці інтерфейси дозволяли людям спілкуватися з пристроями у більш чи менш зручній формі мовою машин. Нині, коли стрімко зростає кількість користувачів інтернет-послуг із мобільних пристроїв та планшетів, для взаємодії з пристроєм потрібно на одного «посередника» менше: атавізмами стають миша і клавіатура. Проте, графічний інтерфейс (GUI) і на екрані робочого столу, і на тачскріні планшета вимагає від користувача для взаємодії задавати ті чи інші зрозумілі для відповідного пристрою команди.

Zero UI[44] – парадигма дизайну, яку запропонував Енді Гудман, дизайнер Fjord. Явище точно описує вислів: «Дизайн Zero UI – це дизайн невидимих інтерфейсів».[45] Більше того, коли йдеться про «дизайн» або «інтерфейс» у контексті Zero UI, мова йде радше не про гарно намальований веб-дизайн, а про зручно спроектований дизайн продукту з винахідницькими компетенціями у широкому сенсі слова.

Zero UI характеризують дві головні складові:

1) **Відсутність звичного графічного інтерфейсу**. Zero UI – найприроднішій спосіб керувати пристроєм. Інакше кажучи, це touchless tech, інтерфейс без видимої частини інтерфейсу. З найвідоміших прикладів таких інтерфейсів: Microsoft Kinect[46] і Apple Siri,[47] керування жестами і голосом відповідно.

---

[41] Що таке юзабіліті сайту і чому це так важливо? URL: http://webstudio2u.net/ua/faq/50-design/93-useability.html (дата звернення 20.09.2018).

[42] Forrester Research – незалежна аналітична компанія, яка займається дослідженнями ринку інформаційних технологій. Також компанія надає своїм клієнтам інформацію про те, який вплив ці технології здійснюють на бізнес та їхніх споживачів. Forrester Research має 8 дослідницьких центрів у США, Канаді та Європі.

[43] Юзабіліті (Usability) – це властивість продукту бути придатним до використання та визначає загальну степінь зручності його використання. Веб-юзабіліті – це підхід, покликаний зробити веб-сайти інтуїтивно простими у використанні для кінцевого користувача, без потреби проведення спеціалізованого навчання.

[44] UI (англ. User Interface) – користувацький інтерфейс.

[45] Попков С. Веб-дизайн в прошлом. Знакомьтесь, Zero UI. URL: http://blog.aic.ru/zero-ui (дата обращения 20.09.2018).

[46] Kinect – безконтактний сенсорний ігровий контролер, розроблений компанією Microsoft. Kinect дозволяє користувачу взаємодіяти з нею без допомоги контактного ігрового контролера через усні команди, пози тіла і показувані об'єкти або малюнки.

[47] Siri (від англ. Speech Interpretation and Recognition Interface) – це розробка Міжнародного центру штучного інтелекту, яка по суті є хмарним персональним помічником у складі iOS, watchOS, macOS і tvOS компанії Apple. Ця програма побудована на діалоговому мовному інтерфейсі, інтелектуальному розпізнаванні та обробленні природної мови. Siri пристосовується до кожного користувача індивідуально, вивчаючи його вподобання протягом тривалого часу.

2) **Попереджувальні технології**. Відсутність візуальної частини інтерфейсу – це лише зовнішня оболонка Zero UI. Внутрішнє наповнення гарантує виконання частини дій замість користувача, бо пристрій вгадує наперед його бажання. Тут звісно йдеться про найпопулярніші в IT останнім часом: Big Data, машинне навчання, штучний інтелект, а також персоналізацію. Наприклад, інтелектуальний термостат Google Nest запам'ятовує розпорядок дня мешканців будинку, визначає, коли вони виходять і приходять, щоб автоматично регулювати подачу тепла. При цьому Nest має і звичайний тач-інтерфейс для ручного налаштування. Подібні пристрої вже не сприймаються як наукова фантастика. Це стало можливим завдяки передумовам як у сфері технологій, так і в галузі самого дизайн-підходу.

6.7.2. Причини виникнення Zero UI

Наведемо короткий список основних технологій і тенденцій у дизайні, які зрештою зумовили неминучу появу Zero UI (у дужках приклади).

Технології:

- голосове керування (Siri, Microsoft Cortana);
- керування жестами (Microsoft Kinect);
- керування мікрожестами (Leap Motion);
- локалізація обличь на фото і відео;
- розпізнавання обличь, відбитків пальців;
- бездротова передача енергії;
- визначення точної геолокації користувача (iBeacon);
- інтернет речей (Google Nest);
- VR і AR, віртуальна і доповнена реальність;
- Big Data, персоналізація, машинне навчання.

Все це свідчить про те, що з'явилася технологічна база для Zero UI, оскільки стало можливим керувати пристроєм без допомоги контролерів-посередників, а також використовувати аналіз даних для попереджувальної дії інтерфейсів.

Тенденції у дизайні:

- мінімалізм в інтерфейсах;
- уніфікація стилів веб-дизайну (material design: виробники ОС диктують стиль);
- скорочення варіативності, зовнішня типізація інтерфейсів;
- популяризація готових типових рішень (конструктори сайтів, бібліотеки іконок, немає потреби витрачати багато ресурсів на графіку);
- мультиплатформовий дизайн (єдиний інтерфейс для різних носіїв: сайт, застосунок, термінал – вимагає розроблення єдиного UI Kit).
- безшовний досвід (користувач може вибрати товар з планшета, продовжити покупку з робочого ПК, а оповіщення про доставку отримувати через push – при цьому ланцюжок не переривається, для користувача це єдиний процес).

Отже, цінність дизайну як графічного мистецтва знизилася за рахунок витіснення його типовими якісними рішеннями. Орієнтація на декілька платформ відразу змусила розробляти більш продумані дизайн-рішення, цінність аналітики зросла. Загальну тенденцію у дизайні на сьогодні можна описати так: «Знецінення графічної частини дизайну разом з одночасним зростанням цінності проектування інтерфейсу. Сильна пре-аналітика, концентрація на зручність користувача і використання технологічних можливостей»[48].

6.7.3. Як Zero UI позначиться на дизайні

Вплив парадигми Zero UI на веб-дизайн – це, в першу чергу, не безконтактне керування, а саме інтелектуальність у дизайні. Сенс Zero UI у веб-дизайні – передбачати, що користувач захоче зробити, і перемістити його відразу в середину користувацького сценарію.

Дизайнерам доведеться розбиратися, зокрема, у біології та психології, щоб створити ці нові пристрої, які вже не будуть покладатися на зображення, як на шлях до взаємодії з програмою. Очевидно, що нова парадигма породить нові різновиди дизайну і розширить перелік назв професій дизайнера. Але на цьому її вплив не закінчується. Можна було б наводити приклади дуже довго і з самих різних галузей: програми-навігатори, онлайн-магазини, сервіси замовлення таксі та доставки тощо.

Веб-сервіси, що відповідають принципам Zero UI, виконують частину призначених для користувача сценаріїв, що робить процес взаємодії з ними швидким, зручним і зрештою істотно підвищує лояльність клієнтів: пересідати на веб-сервіс з «ручним керуванням» після них не хочеться.

Забігати надто далеко в майбутнє – справа невдячна, проте можна констатувати, що дизайн перебуває на дуже інтенсивному етапі становлення зовсім нових підходів і парадигм. Очевидно, Zero UI сьогодні виглядає найбільш логічним рішенням всіх наболілих проблем у проектуванні інтерфейсів та й просто даниною сучасному ритму життя, коли секунди продуктивності змушують відмовитися від користування сервісом. Це зумовлюється розвитком продуктів під впливом штучного інтелекту та «нових реальностей»: доповненої і віртуальної.

Zero UI – тренд, який змінює сам підхід до дизайну. Тепер гіпотези ґрунтуються не на досвіді конкретної людини та її припущеннях, в основі – точні бізнес-показники, аналітика, постійне тестування цих гіпотез. Дизайн стає самонавчальним механізмом, а дизайн-команда, передусім, – аналітичною командою, яка змінює процес дизайну і процеси самого бізнесу[49].

Можливо, Zero UI радикально вирішить проблеми взаємодії користувачів із пристроями, але він очевидно вже пропонує нові рішення і нові підходи, які значним чином сприяють прогресу дизайну.

---

[48] Попков С. Веб-дизайн в прошлом. Знакомьтесь, Zero UI. URL: http://blog.aic.ru/zero-ui (дата обращения 20.09.2018).

[49] Що таке Zero UI, та до чого рухається майбутнє дизайну. URL: https://designtalk.club/shho-take-zero-ui-do-chogo-ruhayetsya-majbutnye-dyzajnu (дата звернення 21.09.2018).

Також можна бути впевненими, що еволюція інтерфейсу на цьому не закінчиться і рано чи пізно вона позбудеться будь-яких посередників і перейде на пряму взаємодію думкою, яка також поставить перед дизайнерами нові завдання та виклики.

6.7.4. Новації штучного інтелекту для дизайнерів

Машинне навчання і штучний інтелект (англ. Artificial intelligence, AI) створюють все нові й нові реальні технології, використовувані у дизайні:[50]

- Google навчив AI розпізнавати якість фото та їхню «естетичну привабливість»;[51]
- сервіс Gfycat, що містить мільйони анімованих картинок, придумав, як завдяки AI покращувати їхню якість, що відкриває нові обрії якості GIF;[52]
- у Photoshop з'явилася потужна новація: виділення об'єкта в один клік;[53]
- штучний інтелект допомагає розпізнавати шрифти.[54]

Уже нині завдяки штучному інтелекту дизайнери можуть відчути певне полегшення у своїй роботі з огляду на нові автоматизовані фічі у застосунках для оброблення зображень. І можна бути впевненим, що таких полегшень буде дедалі більше. Комп'ютер братиме на себе рутинну роботу: виставляти нормальну експозицію для тисяч фотографій, відсіюватиме неякісні фото, виділятиме предмети, накидатиме примітивний інтерфейс (UI) тощо. У недалекому майбутньому всю чорнову роботу візьме на себе машина. Це заощадить багато часу і дозволить дизайнеру зосередитися не на техніці, а на змісті.

Новації у програмному забезпеченні, передусім, покликані полегшити роботу з ним. Це також знизить поріг входження до професії дизайнера. Однак початківцям не варто радіти. Поріг входження може знизитися настільки, що будь-хто зможе зробити якісну ретуш фотографії або накидати цілком пристойний UI застосунку натисканням кількох клавіш.

Вже зараз комп'ютер здатен писати новини і створювати програми, водити авто. Те саме буде і з дизайном. І чим далі, тим складніші завдання зможе виконувати штучний інтелект. Власне кажучи, це призведе до нерадісної картини для початківців: їм буде значно важче, ніж раніше, адже їхню роботу зможе дуже швидко і доволі якісно виконувати AI.

---

[50] Чи може штучний інтелект відібрати роботу у дизайнерів? URL: https://designtalk.club/chy-mozhe-shtuchnyj-intelekt-vidibraty-robotu-u-dyzajneriv/ (дата звернення 12.10.2018).

[51] Google навчив AI розпізнавати якість фото та їхню «естетичну привабливість». URL: https://designtalk.club/google-navchyv-ai-rozpiznavaty-yakist-foto-i-navit-yihnyu-estetychnu-pryvablyvist/ (дата звернення 12.10.2018).

[52] Gfycat відкриває нові обрії якості GIF завдяки AI. URL: https://designtalk.club/gfycat-vidkryvaye-novi-obiriyi-yakosti-gif-zavdyaky-ai/ (дата звернення 12.10.2018).

[53] У Photoshop CC з'явиться вбивча фіча: виділення особи одним кліком. URL: https://designtalk.club/u-photoshop-z-yavytsya-vbyvcha-ficha-vydilennya-osoby-odnym-klikom/ (дата звернення 12.10.2018).

[54] Як розпізнати шрифт: підбірка корисних додатків. URL: https://designtalk.club/yak-rozpiznaty-shryft-pidbirka-korysnyh-dodatkiv/ (дата звернення 12.10.2018).

Також треба усвідомлювати, що автоматизація однозначно породжує тенденцію падіння попиту на низькоякісних фахівців і підвищення попиту на висококласних професіоналів.

6.8. Аналіз веб-дизайну сайтів

Основна задача аналізу веб-дизайну сайту – визначити, як вплине той чи інший дизайн сайту на вирішення завдань його власника. Цільовий відвідувач заходить на сайт не захоплюватись веб-дизайном. Цільовий відвідувач сайту повинен швидко віднайти сайт у пошуковій системі, швидко знайти потрібну інформацію на сайті, можливо зв'язатися з власником сайту.

Чим чіткіше сформульовані вимоги до дизайну сайту, тим ефективніше можна проаналізувати веб-дизайн.

Аналіз веб-дизайну найбільш популярних сайтів показує[55]:

– чим простіший веб-дизайн, тим простіше і зручніше найти відвідувачеві потрібну йому інформацію, і тим вища ефективність сайту, тобто дохід і прибуток від сайту;
– домогтися зовнішньої простоти дизайну сайту непросто. Створити наворочений веб-дизайн набагато легше, ніж зовні простий і непомітний;
– авангардний веб-дизайн у відвідувачів менш популярний, ніж класичний веб-дизайн;
– смак і відчуття кольору у всіх різні. Чим авангардніший дизайн сайту, тим вище ймовірність, що у багатьох відвідувачів сайту він викличе почуття роздратування;
– чим старший середній вік відвідувачів сайту, тим нейтральнішою має бути веб-дизайн.

При здачі сайту веб-дизайнер передає сайт, але не свою голову. Не все, що здається простим і легким, виявляється таким насправді. Роки життя йдуть на те, щоб стати професійним веб-дизайнером. Підтримка сайту без достатнього досвіду і знань, призводить до зниження якості сайту.

Два підходи до аналізу веб-дизайну:

1) подобається чи ні веб-дизайн замовнику, тобто аналіз на рівні суб'єктивних відчуттів: гарний чи негарний веб-дизайн;
2) вирішує чи ні веб-дизайн конкретні завдання власника сайту.

Аналіз веб-дизайну сайту тільки за критерієм «подобається – не подобається», «гарно – негарно» призводить до створення свідомо збиткових сайтів. Аналіз веб-дизайну у сенсі його впливу на вирішення конкретних завдань власника призводить до створення ефективних сайтів, які приносять прибуток.

Веб-дизайн сайту ніколи не може вважатися закінченим. Немає межі досконалості. Ніколи не рано і ніколи не пізно аналізувати веб-дизайн сайту.

---

[55] Анализ веб-дизайна. URL: http://www.antula.ru/web-design_analysis.htm (дата обращения 18.09.2018).

Контрольні запитання

1) Назвати елементи веб-дизайну.
2) Охарактеризувати елемент веб-дизайну «світлотінь».
3) Назвати та охарактеризувати принципи веб-дизайну.
4) У чому полягає принцип контрасту?
5) Що таке принцип балансу у веб-дизайні? Які види балансу бувають?
6) Охарактеризувати принцип зручності сприйняття веб-дизайну.
7) У чому важливість принципу акцентування?
8) Назвати етапи в історії веб-дизайну
9) Охарактеризувати поширену помилку дизайну «неправильна навігація».
10) Назвати сучасні тенденції веб-дизайну.
11) У чому полягає специфіка використання сінемаграфів?
12) Які сучасні стильові рішення у веб-дизайні?
13) Якими є сучасні тенденції у веб-дизайні щодо ілюстрацій?
14) Роз’яснити стиль асиметричних макетів і «ламаної» розмітки.
15) У чому переваги застосування Duotone та оверлей поверх зображень?
16) Що таке брутальний веб-дизайн?
17) Чому градієнти у веб-дизайні знову стали затребуваними?
18) Чим градієнти 2.0 відрізняються від попередніх?
19) Чи є стильним використання анімації у сучасному веб-дизайні? Чому?
20) У чому специфіка анімованої SVG-графіки?
21) Що таке ізометрія у дизайні?
22) У чому переваги використання відсутності меж у сучасному веб-дизайні?
23) Які сучасні тенденції щодо використання кольорів у веб-дизайні?
24) Чим зумовлена сучасна тенденція використання нестандартних дизайнерських шрифтів?
25) Яка основна задача аналізу веб-дизайну сайту?

Розділ 7
Аналіз та оптимізація роботи веб-сайтів

7.1. Аналіз роботи веб-сайтів

7.1.1. Основні складові аналізу роботи сайту

Нині сайти для компаній є одним з найважливіших інструментів бізнесу: з його допомогою можна рекламувати свої товари і послуги та безпосередньо співпрацювати з клієнтами. Проте ефективність сайту як інструмента для бізнесу може знижуватися, якщо сайт працює не так, як було задумано. Саме тому важливим етапом у підтримці веб-сайту компанії є **аналіз роботи сайту**.

Аналіз роботи свого сайту варто проводити на регулярній основі, адже інакше неможливо оцінити: чи реалізує сайт усі 100% свого потенціалу або ж є просто «мертвим вантажем» для компанії. Також без регулярного аналізу неможливо вчасно помітити проблеми або перебої в роботі сайту і внести необхідні правки.

Аналіз роботи сайту має **кілька основних складових**[1], серед яких можна назвати SEO-аналіз сайту, аналіз його відвідуваності, юзабіліті-аналіз, технічний аналіз тощо. Кожен з таких аналізів має важливе значення для загальної оцінки ефективності сайту і визначення подальшого шляху його розвитку.

1) **SEO-аналіз сайту** дозволяє визначити, як сайт «бачать» пошукові системи, виявити сильні і слабкі місця сайту щодо пошукового просування. Наприклад, за допомогою SEO-аналізу можна дізнатися, за якими з тематичних ключових слів сайт перебуває у верхніх рядках пошукової видачі, оцінити, наскільки релевантні ті чи інші сторінки сайту пошуковим запитам, під які вони просуваються, а також з'ясувати, чи добре перелінковані між собою внутрішні сторінки сайту.

2) **Аналіз відвідуваності сайту** дозволяє дізнатися детальну інформацію про те, як і звідки приходять відвідувачі на сайт, скільки часу проводять на його сторінках, з яких сторінок йдуть тощо. Такий аналіз є дуже важливим, адже з його допомогою можна не лише оцінювати поточну відвідуваність сайту, а й прогнозувати майбутню.

Здебільшого для аналізу відвідуваності сайтів користуються спеціальними онлайн-сервісами аналітики, такими як Google Analytics, 24Log[2], HitMeter[3] тощо. Подібні сервіси дають розгорнуту інформацію про кількість відвіданих сторінок, ключові слова, за якими на сайт заходили відвідувачі, шляхи перехо-

---

[1] Найпопулярніший безкоштовний лічильник для сайту. Широкі можливості збору, обробки і подальшого аналізу даних відвідуваності інтернет-ресурсів. Безкоштовний. із роботи сайту. URL: http://webstudio2u.net/ua/design-web/713-analiz-raboty-saita.html (дата звернення 29.09.2018).

[2] 24Log (http://www.24log.ru) – популярніший безкоштовний лічильник для сайту. Надає три варіанти лічильника – з детальною статистикою про відвідувачів, лічильник «скільки людей на сайті» і простий лічильник відвідуваності.

[3] HitMeter (https://hitmeter.ru/) – простий безкоштовний лічильник відвідувачів на сайті. Можна перевірити відвідуваність сайту, показники окремих сторінок, джерела трафіка.

дів по сайту, конверсію, час, проведений відвідувачами на сайті, і про багато-багато іншого. Користуватися ними можна безкоштовно.

3) **Юзабіліті-аналіз сайту** та аналіз дизайну його сторінок – дві тісно взаємопов'язані складові у загальному аналізі роботи сайту. Завданням таких аналізів є виявлення проблем взаємодії відвідувачів із сайтом. Наприклад, за допомогою юзабіліті-аналізу можна визначити, чи добре видно на сторінках сайту важливу інформацію, чи зручна навігація по сайту, чи зрозумілі інструкції або заклики до дії. Аналіз дизайну сайту, у свою чергу, покликаний показати, які саме помилки в дизайні спричинили проблеми з юзабіліті.

4) **Технічний аналіз веб-сайту** потрібен для виявлення відомостей про коректність роботи сайту з технічної точки зору. Так, у рамках цього аналізу може перевірятися коректність роботи різних скриптів на сайті, доступність різних сторінок сайту, швидкість завантаження даних. За потреби подібний аналіз може перевіряти програмний код сайту для виявлення помилок, які можуть заважати нормальній роботі ресурсу.

Оцінюючи основні етапи проведення загального аналізу роботи сайту, можна помітити, що подібний аналіз – справа аж ніяк не проста. При цьому сам по собі аналіз є лише допоміжним засобом у роботі із сайтом, оскільки треба ще й правильно інтерпретувати проаналізовані відомості, щоб скласти якісний та ефективний план подальших дій з розвитку і просування сайту. Без цього проведення аналізу роботи сайту – марна витрата часу.

7.1.2. SEO-аналіз сайту

SEO (Search Engine Optimization – оптимізація для пошукових машин або скорочено пошукова оптимізація сайтів) – це оптимізація внутрішніх і зовнішніх параметрів сайту, які враховуються пошуковими системами для оцінки релевантності, з метою просування сайту в топ і зростання відвідуваності сайту.

SEO полягає в комплексі робіт, покликаних підвищити рейтинг сайту в пошукових системах. Чим вищі позиції в результатах пошуку посідає сайт, тим більше зацікавлених відвідувачів зможе його знайти. Метою SEO-просування є досягнення першої сторінки результатів пошуку за певним запитом. На першій сторінці розміщено лише 10 релевантних сторінок, але мільйони інших намагаються підняти себе сюди, до топ-3 або навіть першого місця (топ-1). Комплекс робіт із розкрутки та просування має багато складових.[4]

Зокрема, це аналіз (моніторинг позиції та сторінки у пошуковиках, SEO-аудит, конверсія та юзабіліті тощо), внутрішня оптимізація (оптимізація HTML-коду, тексту (контенту), заголовків, посилань, службових директив тощо), зовнішнє просування (зовнішні посилання (беклінки), індексування, реєстрації тощо).

Ціна пошукової оптимізації залежить від конкретного завдання сайту і конкуренції у відповідному сегменті, а тому може відрізнятись, через що за висококонкурентними та високочастотними запитами вряд чи вдасться піднятися

[4] SEO просування, розкрутка та оптимізація сайту в Україні. URL: http://seoweb.in.ua/services (дата звернення 29.09.2018).

безкоштовно.[5] Якщо на сайті пропонуються певні послуги або продаються товари (сайт-візитка, корпоративна або комерційна сторінка, інтернет-магазин), то така реклама в інтернеті зможе значно підвищити рівень продажів, оскільки після професійного й ефективного розкручування значно більша кількість потенційних покупців буде переходити на цей ресурс за цільовими запитами.

7.1.3. Аналіз відвідуваності

Щоб дізнатися повну інформацію про відвідувачів сайту, є різні інструменти **веб-аналітики**. Її задачами є повний аналіз та оцінка відвідуваності сайту для подальшого його розкручування, аналіз ефективності рекламної кампанії, оптимізація сторінок сайту для підвищення коефіцієнта конверсії[6] відвідувачів у покупців.

У сфері веб-аналітики застосовуються різні інструменти, серед яких популярністю користуються переважно безкоштовні. Завдяки спеціальним веб-інструментам, наприклад, Google Analytics, можна не лише довідатися кількість хітів (завантажень сторінки) і хостів (тобто відвідувачів з унікальною IP-адресою), а й провести детальний аналіз поведінки відвідувачів на сайті. Крім того, є спеціальні **аналізатори логів** (наприклад, Webalizer, AWStats) – локальні програми, які збирають записи подій (логи) сервера, на якому розміщений сайт.[7]

Якщо потрібна грамотна оптимізація сторінок сайту або ж необхідно провести його **редизайн**, то найбільш корисними для цього будуть такі дані:

- з яких джерел відвідувачі приходять на сайт;
- які ключові слова залучають просто відвідувачів, а які – саме покупців;
- які сторінки сайту є найбільш відвідуваними й чому;
- які сторінки сайту є найменш відвідуваними й чому;
- на яких сторінках відвідувачі покидають сайт, не замовивши послуги або не зробивши покупку.

Так, знаючи, звідки приходять на сайт відвідувачі, можна точніше вибирати методи розкрутки. Якщо головне джерело – **пошукові системи**, то треба більше уваги приділяти пошуковій оптимізації сторінок сайту. Якщо ж, наприклад, більшість відвідувачів приходять із форумів і соціальних мереж, то, навпаки, варто більше займатися розкруткою сайту в цьому напрямі.

А, знаючи, наприклад, IP-адресу відвідувача, можна визначити його географічне місцерозташування. Це надасть власникові сайту можливість визначити, на яку аудиторію йому краще орієнтуватися: вітчизняну або іноземну. Зібра-

---

[5] SEO просування, розкрутка та оптимізація сайту в Україні. URL: http://seoweb.in.ua/services (дата звернення 29.09.2018).

[6] **Конверсія** (англ. CV, convertion rate, CTR або close rate) – це одне з базових понять інтернет-маркетингу. Попростому, конверсія – це співвідношення унікальних відвідувачів будь-якого ресурсу з активними діями, які вони здійснюють там. Під такими діями маються на увазі закінчені угоди (купівля-продаж), реєстрація, зворотній зв'язок (інтернет-листування або дзвінок), перехід по банеру та багато інших дій, вчинення яких принесе замовнику вигоду. Конверсія вимірюється у відсотках за певний період часу: тиждень, місяць, сезон, рік.

[7] Розкрутка сайту. Веб-аналітика. URL: http://webstudio2u.net/ua/web-promotion/245-web-analytics.html (дата звернення 29.09.2018).

ти корисну для веб-аналітики інформацію нескладно й можна впоратися власними силами. А от проаналізувати здобуті результати й розробити необхідний план дій під силу вже досвідченим професіоналам.[8]

7.1.4. Юзабіліті-аналіз

Юзабіліті є показником зручності користування різними сайтами. За допомогою юзабіліті-тестування (юзабіліті-аудиту) можна цей показник оцінити. Юзабіліті-тестування може проводитися як на етапі розроблення сайту, так і протягом усього життєвого циклу сайту.

Юзабіліті-аудит допомагає дізнатися, чи потребує сайт редизайну або перепроектування інтерфейсу користувача. Такий аудит необхідний усім сайтам, але особливо його потребують інтернет-магазини, оскільки цей тип сайтів працює «напряму» з потенційними покупцями. Для інтернет-магазинів юзабіліті-аудит допомагає вирішити такі завдання:

- дізнатися, чому рівень продажів нижчий, ніж очікувалося;
- визначити, на якому етапі взаємодії у відвідувачів найчастіше виникають проблеми;
- виробити рішення для виявлених проблем.

Перш ніж проводити юзабіліті-аудит інтернет-магазину, та й будь-якого іншого типу сайтів теж, варто спочатку якомога конкретніше визначити його цільову аудиторію. Для цього зазвичай складають **портрет типового користувача сайту**[9], який демонструє:

- вік і матеріальне становище;
- інтереси і властиву поведінку;
- професійні навики;
- рівень навиків інтернет-користувача.

Юзабіліті-тестування може бути якісним і порівняльним. При якісному юзабіліті-тестуванні вивчається поведінка і дії користувачів при роботі із сайтом, що тестується, а при порівняльному вивчаються дії користувачів при роботі відразу з декількома сайтами: тим, що тестується, і його конкурентами.

Необхідне **обладнання для юзабіліті-тестування сайту** – це комп'ютер користувача з доступом до інтернету та записувальна (реєструвальна) апаратура для фіксації дій і поведінки респондента під час проведення тестування. В якості такої апаратури зазвичай застосовуються відеокамери або ж диктофони, ще може підійти для цієї мети веб-камера (слід вести запис потоку даних, що передається камерою).

Останнім часом для проведення юзабіліті-тестів стала використовуватися технологія **eye tracking** (айтрекінг), яка дозволяє визначити, як і куди по сторінці сайту рухається погляд користувача. Для застосування подібної технології

[8] Розкрутка сайту. Веб-аналітика. URL: http://webstudio2u.net/ua/web-promotion/245-web-analytics.html (дата звернення 29.09.2018).

[9] Що таке юзабіліті тестування (юзабіліті аудит). URL: http://webstudio2u.net/ua/design-web/655-usabiliti-testirovanie.html (дата звернення 29.09.2018).

потрібен спеціальний прилад eye tracker (айтрекер): за допомогою інфрачервоних променів цей прилад фіксує положення очей при перегляданні, і потім ці дані наносяться на спеціальні карти. Технологія айтрекінга дозволяє отримувати дуже точні дані, проте вона доволі дорога.

Щоб **провести юзабіліті-аудит**, треба відібрати певну кількість респондентів (5–8 буде достатньо), що відповідають портрету типового користувача сайту, підготувати обладнання. Також треба правильно сформулювати завдання для респондентів і скласти сценарій їхньої взаємодії із сайтом, що тестується. Крім того, треба визначити перелік метрик, за якими буде проводитися аналіз зібраних у ході тестування даних.

Під час проведення юзабіліті-тестування важливо дати респондентам зрозуміти, що перевіряється саме сайт, а не вони самі, тому боятися зробити якусь помилку не потрібно. При тестуванні респонденти повинні обов'язково промовляти свої зауваження та емоції при взаємодії із сайтом, адже це теж характеризує рівень комфорту роботи із сайтом: якщо респондент супроводжує свої дії незадоволеними коментарями або ж сердиться, то це може свідчити про те, що користування сайтом є незручним.

Юзабіліті-аудит не закінчується просто фіксацією даних про виявлені проблеми або помилки в роботі сайту. Після збору даних настає черга аналізу – найскладнішого етапу у всьому тестуванні. Фахівці з юзабіліті повинні вивчити всі здобуті дані, систематизувати їх і на їхній основі зробити висновки про те, чи вимагає сайт доопрацювання або перепроектування.

Окремо юзабіліті-тестуванням займаються компанії,[10] які спеціалізуються у цьому напрямі. Також юзабіліті-аудит часто проводиться веб-студіями при розробленні або редизайні сайтів.

7.2. SEO-аудит

SEO-аудит – комплексна перевірка параметрів сайту з метою з'ясування показників до його подальшого просування у пошукових системах. Список параметрів, що перевіряються, кожного разу може бути різним – це залежить і від типу сайту, який треба перевірити, і від його цілей і задач, і від компанії, що надає послуги SEO-аудиту. Переважно SEO-аудит виконується перед початком просування сайту – в цьому разі він потрібен для визначення готовності сайту до виконання активних дій з його розкручування. Також SEO-аудит може виконуватися і після певного періоду просування – для оцінки здобутих результатів.

Грамотно виконаний SEO-аудит дає власнику сайту багато корисних відомостей, допомагаючи йому зрозуміти, де саме на сайті є «слабкі місця», а також дізнатися, які кроки можна почати виконувати з усунення цих «слабких місць». Розглянемо приклад можливого алгоритму проведення SEO-аудиту.

[10] Що таке юзабіліті тестування (юзабіліті аудит). URL: http://webstudio2u.net/ua/design-web/655-usabiliti-testirovanie.html (дата звернення 29.09.2018).

Алгоритм виконання SEO-аудиту:[11]

1) **SEO-аналіз**. На цьому етапі виконання SEO-аудиту проводиться аналіз основних факторів зовнішньої і внутрішньої оптимізації сайту:

– аналіз поточного статусу сайту в пошукових системах за тематичними ключовими словами;
– аналіз кількості та якості зворотних посилань на сайт;
– аналіз кількості та якості проіндексованих пошуковими системами сторінок сайту;
– аналіз показників трастовості сайту (PR[12], тІЦ[13]);
– аналіз унікальності текстів на сторінках сайту;
– перевірка наявності заголовків (h1 – h6) на сторінках сайту;
– аналіз метатеґів (title, keywords, description) кожної сторінки;
– аналіз ключових слів у текстах сторінок сайту;
– аналіз внутрішньої перелінковки сайту;
– аналіз sitemap сайту і файлу robots.txt.

Для виконання усіх перевірок на цьому етапі можуть використовуватися різні ручні або автоматизовані методи. При цьому всі результати заносяться у тій чи іншій формі до спеціального звіту, що дозволить надалі скласти рекомендації щодо виправлення помилок і недоліків, якщо такі виявляться в процесі аналізу.

2) **Аналіз юзабіліті**. Цей етап SEO-аудиту передбачає ретельне вивчення юзабіліті сайту. Фахівці перевіряють, наскільки легко і зручно користуватися сайтом, наскільки зрозумілі і доступні користувачам дії, необхідні для здійснення купівлі або замовлення послуги. На цьому етапі можуть виконуватися такі роботи:[14]

– аналіз різних форм на сайті, зокрема реєстраційні форми замовлення, форми зворотного зв'язку;
– аналіз кнопок купівлі товару / замовлення послуги, а також інших інтерактивних елементів на сторінках;
– аналіз загальної структури головної і другорядних сторінок сайту.

Результатом юзабіліті-аналізу в рамках проведення SEO-аудиту є звіт, в якому вказуються «плюси» і «мінуси» юзабіліті поточної версії сайту.

Перелік рекомендацій SEO-аудиту може бути різним – все залежить від конкретного виконавця. Найчастіше до списку рекомендацій потрапляють пора-

---

[11] Що таке SEO аудит і навіщо він потрібен. URL: http://webstudio2u.net/ua/optimization/689-seo-audit.html (дата звернення 29.09.2018).

[12] **PR** (Page Rank) – алгоритм, який дозволяє визначити авторитетність сайту в пошуковій системі, наприклад, Google. Важливо, що PR розраховується для кожної сторінки сайту. Для простоти можна сказати так: чим більше у сторінки, яка на Вас посилається, Page Rank, тим імовірніше збільшення PR і Вашої сторінки. Значення PR мають діапазон від 0 до 10. Перераховується показник доволі рідко, за деякими даними – раз на 4–6 місяців.

[13] **тІЦ** – тематичний індекс цитування сайту. На відміну від PR, тІЦ розраховує авторитетність всього сайту, а не окремої сторінки. При цьому враховується не лише кількість посилань на сайт, а й, передусім, якість посилань, тобто – наскільки тематика сайту відповідає темі. Ось тому він і названий тематичним індексом цитування. Попростому можна сказати так: чим більше сайтів, близьких за тематикою, посилаються на сайт, тим вище у тІЦ. Варто сказати, що тІЦ використовується лише для ранжирування сайту, але не в пошуковій видачі. Для збільшення тІЦ треба, щоб контент сайту був актуальним й унікальним.

[14] Що таке SEO аудит і навіщо він потрібен. URL: http://webstudio2u.net/ua/optimization/689-seo-audit.html (дата звернення 29.09.2018).

ди щодо заповнення метатегів сторінок сайту відповідно до вимог пошукових систем, поради з оптимізації текстів на сайті, поради з нарощування посилальної маси сайту тощо. На основі цих рекомендацій власник сайту приймає рішення про те, чи варто виконувати роботи з просування сайту.

7.3. Аналіз найпоширеніших SEO-помилок

На практиці нині найбільш поширеними на сайтах є проблеми з метатеґами, розміткою і посиланнями. Як свідчать дослідження[15], часто зустрічаються помилки із заголовками h1–h6, сертифікатами https і помилки, пов'язані з редиректами. Охарактеризуємо їх.

Метатеґи. Як показує статистика, серед помилок з метатеґами найбільш розповсюдженими є занадто довгі title і description та їхнє дублювання. Поширеним є використання неунікального тексту в description. Інколи трапляється, що метатеґи з кодуванням, title та/або description взагалі відсутні на веб-сторінці, оскільки ними знехтували або просто про них забули.

*Мета*тег *title*. Різні пошукові системи по-різному ставляться до ширини title. Рекомендується при оформленні title максимально точно і лаконічно писати текст про те, що можна найти або про що дізнатися, відвідавши відповідну сторінку. При цьому не варто використовувати прикметники на кшталт «кращий», «найліпший» тощо, оскільки здебільшого їх ігнорують роботи. Не треба на головній сторінці сайту в title писати слово «головна». І недоречно копіювати title в інших, позаяк назва сторінки має бути унікальною і змістовною.

*Мета*тег *description*. Нині «вага» description в оптимізації вже не така значима, як раніше. До того ж цей метатеґ все частіше пошуковики почали замінювати на свої, підставні. Проте, хоча атрибут description все слабше враховується роботами в ранжируванні, не варто його ігнорувати, позаяк він добре підвищує конверсію (CTR). Подібно до title, у різних пошукових системах різна оптимальна довжина description. Рекомендовано опис сторінки в description робити змістовним і привабливим. Якщо на сторінці немає багато інформації, не варто створювати великий опис. Проте, якщо сторінка містить багато інформації, доречно гарненько описати сторінку і підказати роботам і людям, про що вони можуть дізнатися на цій сторінці. Доречно використовувати в них іконки і спеціальні символи для залучення уваги. В описі сторінки бажано використовувати основний ключ сторінки в першому реченні. Недоречно наповнювати опис «водою» або лише рекламними пропозиціями, вони погано ранжуються. Не варто копіювати фрагменти тексту в опис, це може зробити за вас і робот. І вже точно не треба копіювати description в інших.

Посилання. Пошукові системи при аналізі профілю сайту ретельно оцінюють як зовнішні, так і внутрішні посилання. Оптимізацію посилальної маси

[15] Самые распространенные SEO-ошибки на сайте: инфографика. URL: https://serpstat.com/ru/blog/samie-rasprostranennie-seo-oshibki-issledovanie-serpstat/ (дата обращения 2.04.2019).

простіше розпочати з внутрішніх факторів. Відповідно до результатів досліджень[16] найчастіше зустрічаються такі види помилок при організації посилань:

1) не використовується атрибут rel = "nofollow" для зовнішніх посилань, який означає, що пошуковий робот не повинен переходити за таким посиланням. Оскільки використання атрибута rel = "nofollow" закриває сторінку від індексації зовнішніх рекламних посилань або посилань на неперевірені сайти, варто не забувати про нього у зовнішніх посиланнях;
2) використовується атрибут rel = "nofollow" для внутрішніх посилань, що нівелює частину можливостей з розподілу посилальної ваги між сторінками сайту;
3) відсутній значок сайту favicon, який відображається поруч з сайтом у результатах пошуку пошукової системи, поряд з адресою сайту в адресному рядку браузера і поруч з назвою сайту у розділі браузера «Вибране» або в «Закладка» замість стандартного значка;
4) велика кількість вихідних посилань є не такою поширеною помилкою, проте вона може призвести до некоректного розподілу посилальної ваги, а також до санкцій від пошукових систем за використання пошукового спаму.

Мікророзмітки[17] дозволяють в повну силу використовувати можливості соціальних мереж у пошуковій оптимізації, зробити якісний наочний попередній перегляд (превью) веб-сторінок у соціальних мережах, підвищити клікабельність посилань на сайт, якими діляться користувачі в своїх профілях. Нині найпоширенішими видами мікророзмітки є Open Graph, Twitter Card, Schema та ін. Пошукові системи звертають увагу на поведінкові фактори, зокрема – увагу до сайту в соціальних мережах. Це робиться в глобальному сенсі для втілення ідеї семантичної павутини, тобто стандартизації всієї інформації в інтернеті, щоб вона стала придатною для автоматичного оброблення.

Протокол Open Graph, створений фахівцями Facebook, зараз підтримує більшість популярних соцмереж і месенджерів: Facebook, Telegram, WhatsApp, Twitter, Viber та Pinterest.

Schema.org – стандарт семантичної розмітки даних у мережі, використовуваний пошуковими системами Google, Bing і Yahoo!.

Twitter Card – окрема розмітка для Twitter.

Помилково забувати зазначати різні види мікророзмітки, позаяк це зможе забезпечити сайту зростання і популярність в інтернеті, залучити нових підписчиків, збільшити кількість лайків і перепостів, підвищити релевантність контенту і підняти інтерес до сайту в соціальних мережах, поліпшити клікабельність та отримати трафік на сайт.

---

[16] Самые распространенные SEO-ошибки на сайте: инфографика. URL: https://serpstat.com/ru/blog/samie-rasprostranennie-seo-oshibki-issledovanie-serpstat/ (дата обращения 2.04.2019).

[17] **Мікророзмітки** (мікроформат) – це особливий спосіб семантичної розмітки інформації на веб-сторінках, який дозволяє пошуковим роботам точно визначати сутність інформації (стаття, контакти, новина, подія, фільм, люди тощо) і структурувати інформацію на сторінках сайту. Нині найпоширенішими видами мікророзмітки є: Open Graph, Schema, Twitter Card, мікроформати (hCard, hRecipe, hReview, hProduct) і семантична розмітка HTML5.

Заголовки. Відповідно до результатів дослідження[18] поширеними є такі помилки з використання заголовків:

28.25% – відсутній заголовок h1;

24.37% – більше одного заголовка h1;

24.37% – порушена черговість заголовків н1–н6;

22.99% – заголовок h1 дублює тег title.

Отже, усі помилки у використанні заголовків допускаються однаково часто, але з невеликим відривом лідирує відсутність заголовка h1. Багато власників сайтів нехтують заголовком h1 через те, що не можуть його лаконічно вписати у дизайн сторінки. Кілька заголовків h1 або відсутність ієрархії найчастіше зустрічаються через те, що простіше виділити заголовок тегом й орієнтуватися на видимий результат, ніж підбирати стилі для здобуття такого самого візуального ефекту. Тобто просто некоректно застосовують доступні інструменти.

Заголовки h1–h6 варто використовувати лише для структурування основного контенту сторінки і не використовувати для виділення окремих блоків, заголовків цих блоків або елементів меню. Можливе використання декількох заголовків h2–h6 на одній сторінці сайту і тільки одного заголовка h1. При цьому заголовок h1 на кожній сторінці має бути унікальний у межах сайту.

Сертифікат https. При використанні https-сертифікатів більшість усіх помилок складають посилання зі сторінок з https на http. Також доволі часто проблемним є наявність незахищених елементів на сторінках з https.

Суть помилки: після переведення сайту з http на https частина адрес на сайті залишилися з http. При цьому, якщо налаштувати 301 редирект з http на https, то фізично для користувача ніяких проблем не буде, він перейде на цільову сторінку. Проте, якщо редирект налаштований неправильно і сайт доступний як за https, так і за http, то це переведете користувача з безпечної зони у небезпечну, він наразиться на небезпеку крадіжки персональних даних. Крім того, зайві редіректи ускладнюють індексацію сайту. Щоби уникнути цієї помилки, варто правильно підготувати сайт до переходу на https: змінити URL-адреси на відносні, перевірити посилання на медіаконтент, просканувати сайт після переведення на новий протокол краулером[19] і виправити помилки.

Мультимедіа. Переважна більшість усіх помилок оптимізації мультимедійних файлів на сторінках сайту складають занадто великі зображення. Великий обсяг зображень (100 Кб і більше) уповільнює їхнє завантаження і завантаження сторінки в цілому. Саме тому варто оптимізувати зображення для сторінок сайту. Інколи на сторінках виявляються посилання на неіснуючі зображення, а отже відвідувачі сайту не зможуть побачити такі зображення, що може негативно вплинути на їхню поведінку і ставлення до сайту.

---

[18] Самые распространенные SEO-ошибки на сайте: инфографика. URL: https://serpstat.com/ru/blog/samie-rasprostranennie-seo-oshibki-issledovanie-serpstat/ (дата обращения 2.04.2019).

[19] Краулер (павук, пошуковий робот, бот) – спеціальна пошукова програма, що є складовою частиною пошукової системи і призначена для перебору веб-сторінок в інтернеті з метою внесення інформації про них у базу даних пошукової системи.

Редіректи[20], по суті, не є помилками як такими, проте усі посилання на сайті повинні перенаправляти користувача безпосередньо до цільової сторінки. Прикладом доцільності використання редиректу є ситуація, коли власник сайту купує більш гарне доменне ім'я, наприклад, для інтернет-магазину. При цьому змінення адреси загрожує втратою постійної клієнтури, а то й повним руйнуванням бізнесу в інтернеті. За допомогою редиректу відбуватиметься автоматичний перехід користувачів на новий сайт навіть у тому разі, якщо вони звертаються за колишньою веб-адресою. Іншим поширеним прикладом, коли потрібен редирект, є так звана склейка доменних імен.

Справа в тому, що пошукові системи ставляться до редиректів з підозрою, позаяк вони перешкоджають якісній індексації сайтів, знижуючи загальну ефективність пошукових машин. Саме тому не всі скрипти редиректів однаково сприймаються пошуковими системами. Особливо небезпечні в цьому сенсі редиректи на рівні клієнта, оскільки робот індексує вміст сторінки, з якої йде редирект, а користувач фактично потрапляє на сторінку, куди спрямовує редирект. Виняток становить так званий «301-й редирект» – редирект на рівні сервера. До такого редиректу пошукові системи дружні, позаяк його корінна відмінність від інших редиректів – одноразове, остаточне і зрозуміле для робота переміщення ресурсу на новий URL.

Рекомендовано уникати: багатокрокові редиректи, посилання на сторінки з перенаправленнями, редиректи на нерелевантні сторінки, редиректи на неіснуючі або непрацюючі сторінки, редиректи на robots.txt.

Атрибути hreflang. Помилковою є відсутність атрибута hreflang на багатомовних сторінках. Цей атрибут рекомендований для пошукових систем і актуальний, в першу чергу, для Google. Найпоширеніша помилка у використанні hreflang – це відсутність посилань з однієї сторінки на іншу. Адже якщо є кілька версій сторінки на різних мовах, то кожна з них має містити коректні посилання на інші версії.

Індексація.[21] Під час індексації пошукові роботи (краулери) сканують і обробляють веб-сторінки, зображення, відео та інші доступні для сканування файли. Саме індексація сайту дозволяє йому відображатися в результатах пошуку. Сторінки, які пройшли сканування й оброблення, зберігаються в базу даних, яка називається «пошуковий індекс». Саме у цій базі даних пошукова система шукає результати на запити користувачів.

Атрибут Canonical. Здебільшого неполадки, які можуть призвести до помилок індексації, складають помилки з атрибутом canonical тегу <link>, Канонічна сторінка (<link rel = "canonical" …>) – це оригінальна сторінка або першоджерело. Пошукові системи негативно ставляться до дублювання контенту, будь це крадіжка контенту з іншого сайту або дублікати сторінок на одному сайті. Організація канонічних сторінок як суттєва частина SEO-оптимізації сайту дозволяє уникати індексування сторінок-дублів.

---

[20] **Редирект** (redirect) – це спрямування користувачів і роботів на сторінку, відмінну від тієї, яку вони запитували.
[21] **Індексація сайту** – це процес збору інформації пошуковою системою про вміст сайту.

Правила, які визначають коректне сприйняття rel = "canonical" пошуковими системами:

– канонічна сторінка повинна існувати і бути доступною для індексування;
– адреса канонічного посилання не має бути зазначеною у домені іншого сайту;
– на одній сторінці не може бути більше однієї канонічної URL сторінки;
– не повинно існувати ланцюжків канонічних посилань.

Канонічне посилання дозволяє вказати, яку саме сторінку з групи схожих або однакових треба індексувати. Корисність даного інструмента складно переоцінити і нерозумно ігнорувати, адже саме до правильного тлумачення сторінок сайту пошуковими системами в значній мірі і зводиться SEO сайту. Тим паче, що канонічні посилання підтримуються практично будь-якою сучасною CMS на кшталт WordPress або Joomla.

Як уникнути помилок з canonical? Рекомендовано перевірити коректність значення rel = "canonical" на сторінках сайту і переконатися, що вказані саме ті сторінки, які варто проіндексувати пошуковою системою.

*Мета*тег *noindex*. Іншою помилкою індексації є сторінки, закриті від індексації noindex.[22] Метатег noindex забороняє індексувати сторінку пошуковій системі. Різниця між метатегом і закритою сторінкою в тому, що на сторінки з метатегом пошукова система заходить, але не індексує, а закриті в robots.txt – не відвідує. Рекомендовано перевірити ті сторінки, які не повинні індексуватися (і, відповідно, приносити трафік), і лише їх закрити від індексації.

Тег iframe. Найчастіше через iframe в HTML-коді підключають різні корисні онлайн-сервіси на кшталт YouTube. З іншого боку iframe є застарілим тегом, а інформація в ньому не враховується пошуковими системами при індексації. Це зумовлено небезпечністю елемента iframe, адже, використання iframe зі стороннього сервісу не виключає можливості дозавантаження в нього чого завгодно: починаючи з куків, завершуючи вірусами тощо.

Отже, задля безпеки свого сайту через iframe варто долучати лише перевірені корисні сервіси. При цьому варто пам'ятати про відсутність індексації даних у блоках iframe. Через це варто тричі подумати про використання таких блоків і уникати їх використання завжди, коли це можливо.

Контент. Найбільш поширена помилка у контенті сторінки – це наявність сторінок Lorem ipsum[23]. Однак також нерідко зустрічається відсутність тексту в body і занадто великий розмір сторінки.

Текст Lorem ipsum. Часто при верстці розробники використовують Lorem Ipsum як текст за умовчанням. Це може призвести до сприйняття сторінки пошуковою системою як нерелевантної, що не відповідає тематиці сайту, або та-

[22] Самые распространенные SEO-ошибки на сайте: инфографика. URL: https://serpstat.com/ru/blog/samie-rasprostranennie-seo-oshibki-issledovanie-serpstat/ (дата обращения 2.04.2019).

[23] Lorem ipsum – класичний варіант умовного беззмістовного тексту, що вставляється в макет сторінки (у сленгу дизайнерів такий текст називається «рибою»). Lorem ipsum – це перекручений уривок з філософського трактату Цицерона «Про межі добра і зла», написаного в 45 році до нашої ери латиною.

кою, що дублює інші сторінки в інтернеті. Виправлення полягає в заміні на текст, відповідний тематиці сайту і сторінки.

Занадто великий розмір сторінки. Розмір сторінки впливає на швидкість її завантаження. Низька швидкість може погано відбитися на поведінкових факторах. У занадто важких сторінок можуть виникнути проблеми зі скануванням та індексацією. Саме тому варто зменшити розмір сторінки до 2 MB.

Відсутній текст в body. Контент на сторінці важливий для пошукових систем, як показник корисності вмісту сторінки для користувачів. Текстовий контент на сайті повинен бути, бо пошукові системи, як і раніше, ранжують сторінки в інтернеті на основі аналізу саме текстової складової. Саме тому варто наповнити body унікальним контентом.

AMP.[24] Для швидкого завантаження сайту на мобільних пристроях використовують елементи AMP. Важливо відстежувати, щоб робота коду JavaScript і таблиць стилів CSS відбувалася згідно з правилами AMP. У разі виявлення пошуковою системою помилок на сторінках AMP ці сторінки будуть видалені з індексу, позаяк сторінки з такими помилками не допускаються до роботи і по них немає показів у пошуку. Після цього трафік сайту стрімко впаде до нуля.

Найчастішими помилками з AMP є:

– *Застарілі елементи.* За потреби слід замінити застарілі елементи AMP на актуальні, адже сторінки, які не відповідають критеріям AMP, можуть бути проіндексовані пошуковою системою як звичайні;
– *Помилки таблиці стилів CSS*. Якщо на сторінці виявлені невідповідності правилам AMP у таблицях стилів CSS, пошукова система не зможе індексувати цю сторінку;
– *Непрацюючі або неправильно використані теги HTML*;
– *Помилка в коді JavaScript*;
– *AMP не використовується*.

Коди звіту сервера. Наявність помилкових кодів відповіді сервера не є проблемою, допоки це не стає масовим явищем на сайті. Якщо всього кілька сторінок містять неправильні коди, у цьому немає нічого страшного, але, як тільки вони починають накопичуватися, треба терміново втручатися й усувати цю технічну проблему, інакше вона призведе до того, що сайт буде гірше індексуватися або втратить авторитет і, відповідно, позиції в пошуку.

Код звіту сервера 4xx (Server status code: 4xx) повідомляє про помилки, що містяться в клієнтському запиті. Можливо, запит містить неправильний синтаксис або не може бути виконаний, або при посиланні на веб-сторінки припустились помилки. Переважно код помилок 4xx (400–499) свідчить про те, що на сайті знайдені посилання на неіснуючі сторінки. Пошукові системи негативно ставляться до таких посилань, і це впливає на швидкість індексації сторінок сайту і на ранжування сайту в цілому. Саме тому рекомендовано прибрати такі посилання з сайту або замінити їх.

[24] AMP (Accelerated Mobile Pages – прискорені мобільні сторінки) – проект Google, запущений 2015 року і розроблений для прискорення швидкості завантаження контенту на мобільних пристроях.

Код відповіді сервера 5xx показує випадки невдалого виконання операцій з вини сервера. Такі помилки можуть спричинити зменшення швидкості індексації сайту або відмову від індексування сторінок пошуковою системою.

Варто намагатися уникати 4xx і 5xx помилок на сайті. До речі, помилки 404 і 500 пошуковик визначить однаково – сторінки не існує. За наявності помилки 503 він зайде на сайт пізніше, щоб побачити зміни. Проте, якщо вона не буде виправлена упродовж 2-х днів, то розцінюватиметься вже як 404.

Швидкість завантаження безпосередньо впливає на поведінкові фактори, і, як наслідок, на ранжування сайту. Тому варто перевірити швидкість завантаження сторінок сайту і за потреби оптимізувати її. Є різні причини, які впливають на швидкість завантаження. Розглянемо найпоширеніші засоби для оптимізації швидкості завантаження: використання кеша браузера, оптимізація зображень, скорочення коду JavaScript та CSS.

Використання кеша браузера. Наявність системи кешування на сайті дозволяє зберегти копії елементів сторінок прямо у веб-браузері того користувача, який їх відвідує, і при кожному наступному зверненні всі ці елементи братимуться прямо з кеша браузера відвідувача, тобто зі спеціальної папки, яка розташовується на диску його комп'ютера. Кешування файлів браузером у подальшому використанні значно збільшує швидкість завантаження сайту. Якщо задати кешування у браузері користувачів, це сприятиме просуванню веб-ресурсу.

Оптимізація зображень. Правильний формат і стискання зображень дозволяє скоротити їхній розмір, і тим самим прискорити завантаження сторінки з ними. Рекомендовано зменшити розміри і повторно зберегти зображення, використовуючи графічний редактор, замість використання атрибутів ширини і висоти в HTML.

Скорочення коду JavaScript. Мінімізувати навантаження на JavaScript можна шляхом видалення зайвих символів, непотрібних коментарів, щоб в результаті отримати однорядковий JS-файл. Проведення обфускації[25] JS-коду зменшує і захищає вихідний код.

Скорочення коду CSS. Якщо на сайті підключено занадто велика кількість зовнішніх файлів, це позначиться на його швидкості завантаження, позаяк відбуватиметься багато http-запитів до сервера. Скорочення коду CSS – одна з основних рекомендацій щодо оптимізації сайту, позаяк невалідний код CSS-файлів може призвести до помилки.

По-перше, для оптимізації завантаження, варто об'єднати файли CSS. Для цього треба вибрати один файл і послідовно скопіювати у нього вміст інших CSS-файлів. Потім у HTML-коді сайту залишити підключення лише файлу, в який були скопійовані всі стилі.

[25] Обфускація – перетворення вихідного тексту або виконуваного коду програми до вигляду, що зберігає її функціональність, але ускладнює аналіз, розуміння алгоритмів роботи, копіювання, повторне використання і модифікацію при декомпіляції. Проводиться Javascript Obfuscator засобами безкоштовного онлайн-сервісу, наприклад, ShrinkSafe.

По-друге, слід оптимізувати вміст CSS-файлу, скориставшись будь-яким сервісом для скорочення розміру файлу каскадних таблиць стилів, наприклад, CY-PR (https://www.cy-pr.com/tools/css/) або CSS Compressor (https://csscompressor.com/). Для цього треба виділити текстовий код CSS, скопіювати його і вставити у спеціальне поле онлайн-сервісу.

Оптимізація швидкості великих сайтів (проектів) буває досить складним завданням. Для прискорення потрібна комплексна робота з тестуванням усіх веб-сторінок і застосуванням різних інструментів.[26] Часом оптимізатори обмежуються перевіркою головної сторінки в Page Speed Insights, проте, звичайно, цього недостатньо.

Знання про проблеми, які найчастіше виникають у власників сайтів, допоможуть вжити відповідних заходів і передбачити їхню появу у себе. З іншого боку, якщо на сайті немає помилок, то це не означає, що вони не з'являться в майбутньому. Щоб їх помітити ще до падіння позицій, варто налаштувати періодичне оновлення аудиту і дізнаватися про помилки задовго до падіння позицій у пошукових системах.

7.4. Оптимізація сайтів: як покращити готовий сайт

Оптимізація сайтів допомагає покращити юзабіліті, ефективність методів розкручування й просування. Оптимізація є важливою частиною життєвого циклу сайтів. Зрештою оптимізація сайтів сприяє підвищенню їхньої прибутковості, а отже, позитивно впливає на фінансове благополуччя компанії. Як можна поліпшити вже готовий сайт?

Якщо оптимізація сайтів не була виконана ще на етапі створення цих сайтів, то як результат створені сайти можуть не задовольняти поставленим задачам і цілям. Однак є кілька простих і доступних методів, які дозволять оптимізувати сайти без їхнього повного або часткового редизайну, модернізації.[27]

Наприклад, можна додати до зображень на сайті атрибути alt та title. Атрибут alt (від англ. alternative) містить текст, який у разі випадкового або спеціального відключення зображення (наприклад, у браузері) надасть необхідні пояснення відвідувачам сайту чи пошуковим роботам. Атрибут title, у свою чергу, відповідає за «підказку», яка з'явиться при наведенні вказівника миші на зображення. Атрибут title подібним чином працює і для гіперпосилань (тег <a>).

До слова, **гіперпосилання можна зробити зручнішими для відвідувачів**, якщо це не було зроблено раніше. Текстовий вміст посилань, так званий анкор, повинен чітко інформувати відвідувачів про сторінки, на які ведуть посилання. Крім того, оптимізація сайтів із використанням ключових слів в анкорах сприяє не лише кращій навігації по сайту, а й пошуковому просуванню сайту.

---

[26] Самые распространенные SEO-ошибки на сайте: инфографика. URL: https://serpstat.com/ru/blog/samie-rasprostranennie-seo-oshibki-issledovanie-serpstat/ (дата обращения 2.04.2019).

[27] Оптимізація сайтів: як покращити готовий сайт? URL: http://webstudio2u.net/ua/optimization/392-site-optimization.html (дата звернення 29.09.2018).

Швидко й без шкоди для основного дизайну сайту, його контенту, можна **видалити фонову музику** або, якщо вона все-таки потрібна, прибрати автоматичне програвання музики при завантаженні сторінок сайту. Як показує практика, відвідувачів дратує музика, програвання якої починається автоматично, або ж музика, яку не можна відключити. А якщо відвідувачів щось дратує, вони покидають сайт.

Оптимізація сайтів у «лайт»-версії для підвищення ефективності пошукового просування виконується просто. Можна **відредагувати заголовки сторінок сайту** (title, відображається у лівій верхній частині вікна браузера) відповідно до головних ключових слів сайту. Якщо помістити у заголовок сторінки ключові слова, це допоможе пошуковому роботу проіндексувати сторінку й підвищить її шанси потрапити у топ пошукової видачі.

Можна **підсилити на сайті ключові слова**, що також не потребує «фундаментальних» змінень у структурі або вмісті сайту. Достатньо виділити ключові слова більшим за розміром шрифтом, курсивним або жирним накресленням. Для пошукових роботів величезне значення мають також заголовки у тексті. Однак не можна використовувати занадто багато заголовків. Загальноприйняте правило: не більше 1–2 заголовків h1, дещо більше – h2 і под. Тільки грамотне використання заголовків допоможе підняти позиції сайту.

Оптимізація сайтів без радикальних змінень структури або дизайну вже готового сайту можлива не завжди. Наприклад, змінення всього лише гарнітури шрифту тексту на сайті може викликати потребу перегляду всієї концепції дизайну сайту. А необхідність додавання / видалення одного розділу призведе до змінення усієї структури сайту, появи «битих» посилань тощо.[28]

Тому оптимізація сайтів має здійснюватись ще на етапах їхнього розроблення і тестування. У цьому разі існує набагато більше можливостей для виправлення виявлених проблем, забезпечення високого рівня юзабіліті сайту та його ефективності.

Прикладами безкоштовних онлайн-засобів перевірки сайтів є Уніфікований валідатор W3C (http://validator.w3.org/unicorn) та Browser Shots (http://browsershots.org).

Показник відмов – це одна з найбільш неоднозначних метрик, коли йдеться про вимірювання результатів інтернет-маркетингу. Google визначає показник відмов як «сеанс з переглядом однієї сторінки на сайті».

Наведемо декілька корисних прийомів для оптимізації сайту та зменшення кількості відмов.[29]

1) **Варто приділити увагу швидкості завантаження сторінки**. За даними Google, якщо сайт завантажується довше трьох секунд, 53% користувачів покинуть його. Вміст сторінки не має жодного значення, якщо користувачі не побачать його відразу. На мобільних пристроях швидкість завантаження має критичне зна-

[28] Оптимізація сайтів: як покращити готовий сайт? URL: http://webstudio2u.net/ua/optimization/392-site-optimization.html (дата звернення 29.09.2018).

[29] Hoben N. 20 Proven Tactics That Will Reduce Your High Website Bounce Rate. URL: https://www.searchenginejournal.com/reduce-bounce-rate/258613/ (accessed 06.11.2018).

чення. Так, із січня 2018 року вона стала одним з факторів ранжирування в Google. Крім того, дослідження[30] свідчать, що у мобільних сторінок, швидкість завантаження яких знизилася з 1 до 10 с, показник відмов зріс на 123%. Отже, не варто забувати про те, що низька швидкість завантаження дратує користувачів.

Перевірити швидкість сайту можна за допомогою інструмента PageSpeed Insights (https://developers.google.com/speed/pagespeed/insights/).

2) **Спростити пошук по сайту** (або додати його, якщо ще не зробили цього). Багато ресурсів відмовляються від функції пошуку по сайту, ускладнюючи тим самим життя своїм відвідувачам. Якщо користувач шукає щось конкретне і не бачить цього відразу ж після заходу на сайт, то надзвичайно важливо, щоб він міг скористатися вікном пошуку. Інакше він покине ресурс.

Дослідження Web Traffic Partners[31] показали, що 90% користувачів не переходять далі першої сторінки сайту при пошуку. Тому не варто ховати відповідний рядок за нагромадженням візуальних елементів. Крім того, користувачі, які активно застосовують пошук по сайту, з більшою ймовірністю проведуть на ньому більше часу. Пошук обов'язково треба включати у сайти з великою кількістю сторінок і товарних позицій. Гості ресурсу не повинні клікати на все підряд у пошуках потрібної інформації, вони цінують свій час і прагнуть комфорту.

3) **Спростити навігацію**. Навігація має бути простою, зрозумілою і не вимагати від користувача жодних зусиль. Згідно з даними досліджень[32], проведених представниками Nielsen, гамбургер-меню і прихована навігація значно погіршують зручність інтерфейсу користувача. Так, на десктопах користувачі віддавали перевагу прихованому меню тільки у 27% випадків. Комбіновані списки і видиме меню воліли використовувати у 50% і 48% випадків відповідно. На мобільних пристроях прихована навігація використовувалася у 57% випадків, а комбіновані списки – у 86%. Крім того, результати дослідження показали, що користувачі здебільшого схильні вибирати навігацію по сайту (якщо вона є), а не пошук.

При оформленні блока навігації варто:

- віддавати перевагу структурі дерева меню;
- кількість рівнів вкладеності організовувати не більше двох-трьох;
- показувати користувачеві, в якому пункті меню він перебуває на цей момент;
- за наявності кількох рівнів вкладеності, використовувати механізм «хлібних крихт»,[33] наприклад: *Головна сторінка → Розділ → Підрозділ*;
- у разі об'ємних каталогів можна задіяти випадні пункти меню.

---

[30] How to make every mobile moment a brand-builder. URL: https://www.thinkwithgoogle.com/consumer-insights/consumer-mobile-brand-content-interaction (accessed 06.11.2018).

[31] On-Site and Off-Site Search Engine Marketing. URL: https://www.webtrafficpartners.com/on-site-and-off-site-search-engine-marketing-optimazation (accessed 06.11.2018).

[32] Hamburger Menus and Hidden Navigation Hurt UX Metrics. URL: https://www.nngroup.com/articles/hamburger-menus (accessed 06.11.2018).

[33] «Хлібні крихти» (від англ. Breadcrumbs або Breadcrumb Trails) – навігаційні стежки (ланцюжки) у механізмі навігації, що використовується в інтерфейсах користувача. Механізм призначений надавати користувачеві можливість відстежувати своє місцеперебування в програмі або серед документів.

4) **Приділити увагу дизайну**. Гарний дизайн інтуїтивно зрозумілий і підвищує довіру до інтернет-ресурсу. Малоймовірно, що користувачі витратять свій час на сайт зі складною навігацією або той, що виглядає підозріло. Результати досліджень[34], проведених представниками компанії Social Triggers, доводять, що користувачам не цікавий контент на сайтах з поганим дизайном. Так, заходити на непривабливий сайт відмовилися 94% респондентів.

Варто зробити дизайн максимально зрозумілим і готовим до взаємодії з користувачем на будь-яких пристроях. Технічні моменти безпосередньо впливають на швидкість завантаження і коректне відображення інформації на сторінках, тому треба адаптувати дизайн сайту під різні пристрої.

5) **Не забувати про мобільних користувачів**. Здебільшого мобільні користувачі менш терпимі, ніж десктопні. Тому для забезпечення гарної взаємодії на мобільних пристроях вкрай важливо, щоб дизайн сайту був адаптивним. Результати експериментів[35] показують, що адаптивність може поліпшити позиції сайту у видачі пошукових систем. Крім того, експерти галузі рекомендують віддавати перевагу саме адаптивному дизайну.

6) **Пам'ятати про читабельність**. Контент на сторінці має бути форматованим і легко читатися. Користувачі не бажають читати погано структуровані статті. Тому варто розбивати великі за обсягом тексти на абзаци, додавати відео, зображення і маркіровані списки.

Ще один важливий аспект – індекс легкості читання тексту. Власникам сайтів варто звернути на нього увагу, оскільки читабельність є одним з факторів ранжирування в Google. Крім того, дані досліджень[36] показують, що тільки 16% відвідувачів сайтів читають тексти від початку і до кінця, в той час як 79% вважають за краще пробігати їхніми очима. Саме тому важливо подбати про те, щоб вони змогли засвоїти якомога більше інформації.

Індекс читабельності обчислюється на базі декількох параметрів: довжина речень і слів, питома кількість найбільш частотних (або рідкісних) слів тощо. Найпопулярніша формула була розроблена спеціально для англійської мови – «Індекс легкості читання Флеша»:[37]

$$FRE = 206{,}835 - 1{,}015 \times \text{Всього слів} / \text{Всього речень} - 84{,}6 \times \text{Всього складів} / \text{Всього слів}.$$

Значення індексу Флеша за шкалою FRE інтерпретується так:

– 100 – легко читається. Середня довжина речень становить 12 або менше слів. Немає слів із більш ніж двох складів;

– 65 – проста мова. Середня довжина речень становить від 15 до 20 слів. У середньому в словах по два склади;

---

[34] The «Content is King» myth debunked. URL: https://socialtriggers.com/content-is-king-myth/ (accessed 06.11.2018).

[35] Адаптивность сайта – увеличение мобильного трафика в разы. URL: https://m.seonews.ru/blogs/mnogoslov/adaptivnost-sayta-uvelichenie-mobilnogo-trafika-v-razy/ (дата обращения 06.11.2018).

[36] How Users Read on the Web. URL: https://www.nngroup.com/articles/how-users-read-on-the-web/ (accessed 06.11.2018).

[37] Индекс удобочитаемости. URL: https://ru.wikipedia.org/wiki/Индекс_удобочитаемости (дата обращения 06.11.2018).

– 30 – трохи важко читати. Речення містять до 25 слів. Багато двоскладових слів;

– 0 – дуже важко читати. В середньому речення має 37 слів. Слово має в середньому понад два склади.

Ця формула була адаптована для кирилиці, один з варіантів адаптації виглядає так:

$$FRE = 206{,}835 - 1{,}3 \times ASL - 60{,}1 \times ASW,$$

де ASL – середня довжина речення в словах (англ. average sentence length);

ASW – середня довжина слова у складах (англ. average number of syllables per word).

Якщо формули здаються важкими, можна вдатися до допомоги відповідних сервісів (наприклад, до http://ru.readability.io/) і з'ясувати, наскільки зрозумілі тексти на Вашому сайті.

7) **Використовувати різні типи контенту**. Тут все очевидно – сучасному користувачеві складно зосереджувати свою увагу на чомусь одному протягом тривалого періоду часу. Вчені Південної Кореї навіть придумали відповідний термін для цього явища – digital dementia (цифрове недоумство). 2007 року вчені стали помічати, що все більше підлітків страждають втратою пам'яті, розладом уваги і низьким рівнем самоконтролю. Дослідження показали, що в мозку таких пацієнтів спостерігаються зміни, схожі з наслідками черепно-мозкової травми. Причиною стало широке поширення смартфонів і зростаючий рівень цифровізації світу в цілому.

Як вже відзначалось у попередньому пункті, тільки 16% користувачів переглядають контент на сайтах повністю. Тому вкрай важливо урізноманітнити його, щоб зацікавити користувачів. Крім того, це може позитивно позначитися на позиціях сайту в пошуку. Так, наявність відео на сайті підвищує шанс попадання ресурсу на першу сторінку видачі в 53 рази.[38]

Проте, при розміщенні відео треба бути обережними, інакше є ризик зробити тільки гірше. Ніякого автозапуску або великого вікна, яке переміщається слідом за відвідувачем. Краще створити спеціальний розділ на сайті з корисним контентом, який затримає користувача на десятки хвилин або навіть години.[39]

8) **Бути обережним зі спливаючими вікнами**. Здебільшого користувачі сприймають їх негативно. Поп-апи відвертають увагу, заважають сприймати контент і взаємодіяти із сайтом. Спливаючі вікна на мобільному пристрої гарантовано викликатимуть напади гніву через те, що займатимуть увесь екран, повністю приховавши контент. Саме тому бажано позбавитися від спливаючих вікон або скоротити до мінімуму їхню кількість. Реклами може бути забагато, особливо якщо дивитися на сайт очима користувача. Через банери показники

[38] Digital Marketing & the Impact of Video. URL: https://martech.zone/digital-marketing-video/ (accessed 06.11.2018).

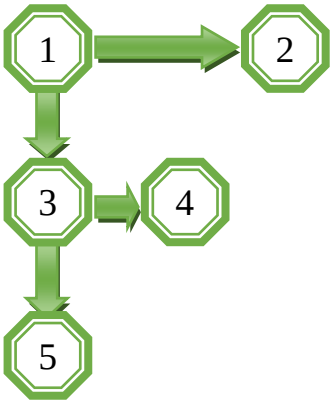
[39] 11 способов снизить показатель отказов на сайте. URL: https://m.seonews.ru/analytics/11-sposobov-snizit-pokazatel-otkazov-na-sayte/?fbclid=IwAR1KXyrEFMLwqoCYk2M0z2ry2y0PiwvpROnT_VDTgE2pTItBkPiDf9vpNUI (дата обращения 06.11.2018).

відмов починають рости. Негативно ставиться до наявності спливаючих вікон і Google. Так, з 2017 року компанія почала знижувати мобільну видачу сторінок, які зловживають міжсторінковою рекламою,[40] різновидом якої є поп-апи.

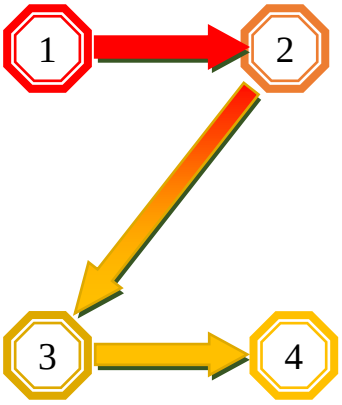
9) **CTA-кнопки**.[41] Якщо на сторінці є заклик до дії, він має бути зрозумілим і розташовуватися на видному місці. Відвідувачі повинні помітити його з перших секунд перебування на сайті. Крім того, CTA-кнопки необхідно привабливо оформити. На думку фахівців, кращими кольорами для таких елементів є зелений і помаранчевий,[42] хоча це залежить від дизайну сайту.

2006 року засновник компанії Nielsen Norman Group Якоб Нільсен дослідив те, як користувачі переглядають сторінки сайтів. З'ясувалося, що найбільш поширений шаблон руху очей користувача – це F-патерн[43]. Згодом це дослідження втратило свою актуальність, бо воно не враховувало поведінку мобільних користувачів, яких тоді просто не було, форми подачі контенту з того часу також суттєво змінилися. F-патерн добре описує поведінку користувачів тільки у випадку з текстами або контентом, розміщеним по монотонній сітці. Для такого контенту доречно розміщувати найважливішу інформацію в перших двох абзацах тексту, а менш важливу інформацію оформляти у вигляді коротких абзаців, підзаголовків, списків уздовж основної осі F. Це дозволить зробити так, щоб навіть при побіжному перегляді користувачі отримають всю потрібну інформацію.

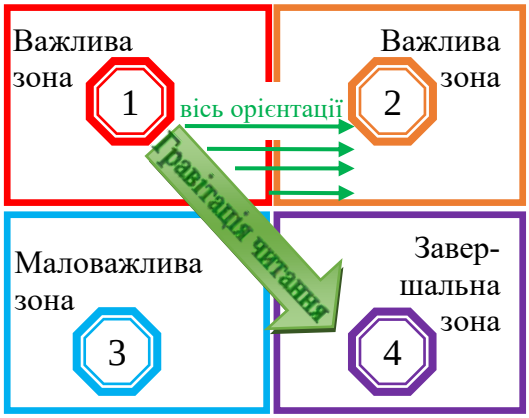
Подальші дослідження[44] експертів 2016 року показали, що поведінка користувачів під час перегляду сторінок сайту описується трьома найбільш типовими моделями: F-патерном, діаграмою Гутенберга і Z-паттернів.



F-патерн



Z-патерн



Діаграма Гутенберга

[40] Google начал понижать в выдаче мобильные страницы с межстраничной рекламой. URL: https://m.seonews.ru/events/google-nachal-ponizhat-v-vydache-mobilnye-stranitsy-s-mezhstranichnoy-reklamoy/ (дата обращения 06.11.2018).

[41] Кнопка «заклику до дії» (CTA, Call To Action) – графічний елемент цільової сторінки з мотивацією відвідувача до певної дії, яку від них очікує розробник сторінки (реєстрації, завантаження, підписки на розсилку тощо).

[42] 11 способов снизить показатель отказов на сайте. URL: https://m.seonews.ru/analytics/11-sposobov-snizit-pokazatel-otkazov-na-sayte/?fbclid=IwAR1KXyrEFMLwqoCYk2M0z2ry2y0PiwvpROnT_VDTgE2pTItBkPiDf9vpNUI (дата обращения 06.11.2018).

[43] F-Shaped Pattern of Reading on the Web: Misunderstood, But Still Relevant (Even on Mobile). URL: https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content (accessed 12.07.2018).

[44] Как пользователи видят сайты: F- и Z- паттерны, диаграмма Гутенберга. URL: https://netology.ru/blog/users-site-patterns (дата обращения 07.11.2018).

Різновид патерну зорової поведінки користувача залежить від розміру екрана, роду даних (текст, графіка тощо), кольорового вирішення дизайну, інформаційного наповнення та інших параметрів. На сьогодні Діаграма Гутенберга – найбільш повна і логічна модель поведінки користувачів при перегляданні сайтів. Саме принципи, які описує діаграма Гутенберга, застосовують і впроваджують великі компанії і відвідувані сайти.

Завдяки цим даним, можна вдало розташувати CTA-кнопки так, щоб користувачі їх не втратили з поля зору.

10) **Подбати про посилання**. Неробочі посилання погіршують взаємодію користувача із сайтом. А дратують вони не менше спливаючих вікон. Перевірити, чи є на сайті биті посилання, можна за допомогою:

– Broken Link Checker;
– Google Search Console;
– Screaming Frog.

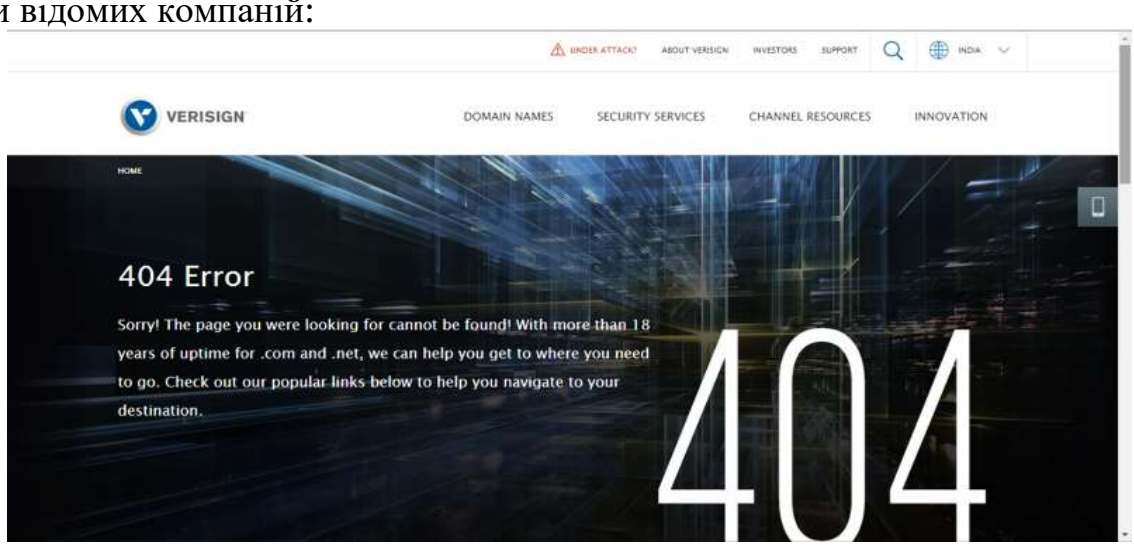
Для збільшення тривалості перебування користувача на сайті варто використовувати внутрішні посилання. Крім того, їхня наявність полегшує навігацію. При цьому варто переконатися, що посилання відкриваються на окремих вкладках. Це дозволить користувачам відкривати відразу кілька сторінок, проводячи на сайті більше часу.

11) **Зробити сторінки з 404 помилкою корисними**. Всупереч поширеній думці компанія Google не знижує показники сайтів за сторінки з 404 помилкою. А це означає, що не треба хапатися за голову, якщо такі раптом виявляться на сайті. Замість цього можна зробити цю сторінку трохи цікавіше, ніж ось це:

404 Not Found

nginx/1.2.9

Наприклад, можна перетворити її на лід-магніт,[45] як це роблять представники відомих компаній:



[45] Лід-магніт (Lead Magnet) – це цінна і безкоштовна пропозиція для потенційного клієнта, яку він може отримати в обмін на вчинення цільової дії. Здебільшого це email або іншого роду контакти.

7.5. Методи SEO-оптимізації сайту

Використовують кілька термінів: пошукова оптимізація, SEO-аудит, SEO-оптимізація[46].

Слід зазначити, що **SEO-оптимізація** (від англ. Search Engine Optimization – пошукова оптимізація) – це комплекс заходів, направлених на підняття позицій сайту в результатах видачі пошукових систем (Google, Yahoo та ін.) за певними запитами (ключовими фразами). Наприклад, у Вас є сайт, на якому Ви даєте корисні поради про пошукову оптимізацію, але за запитом «сео оптимізація» Ваш сайт займає 50-те місце у списку результатів. Відповідно це далеко не перша сторінка зі списком результатів, і дуже малоймовірно, що хтось із потенційних читачів, яким цікаво почитати про SEO (або СЕО), зайде на Ваш сайт із пошукової системи. Заходи, якими можна цю ситуацію виправити (підняти сайт вище у результатах пошуку), називають одним словом SEO. Тобто **основною метою SEO** є збільшення притоку відвідувачів із пошукових систем.

Є різні методи оптимізації сайту і його просування, але виділяють чотири основних: біла, сіра, помаранчева й чорна. Розглянемо їх.

7.5.1. Біла оптимізація сайту

Біла оптимізація сайту – дозволена пошуковими системами. Вона використовує тільки легальні методи просування: змінення щільності ключових фраз у тексті, що не впливає на читабельність, обмін посиланнями з інтернет-ресурсами схожої тематики, коментарі у тематичних блоґах і спеціалізованих форумах. Це природна оптимізація сайту, оскільки спроби вплинути на пошукові алгоритми не здійснюються. Така оптимізація сайту ресурсоємна за тимчасовими і матеріальними витратами, однак має максимально тривалий ефект.

7.5.2. Сіра оптимізація сайту

До сірої оптимізації сайту відносять: використання doorway-сторінок без автоматичної переадресації, з посиланням-запрошенням на цільовий сайт; накручування лічильників відвідувань; створення сателітів (сайтів зі схожим вмістом, що просуваються чорними методами, з переадресацією на цільові сторінки основного ресурсу). Оптимізація сайтів сірими методами може вважатися вищим пілотажем, оскільки попри використання чорної оптимізації, штрафних санкцій щодо цільового сайту пошукові системи не нараховують.

7.5.3. Помаранчева оптимізація сайту

До помаранчевої оптимізації сайту відносять розміщення матеріалів, які не стосуються основної тематики сайту (наприклад, пізнавальних або статистичних фактів). Притягнутий додатковим матеріалом користувач може затриматися на

[46] Оптимізація сайтів. URL: http://webstudio2u.net/ua/optimization/97-seo.html (дата звернення 29.09.2018).

сайті й придивитися уважніше до цільових сторінок, однак така оптимізація сайту достатньо трудомістка й вимагає спеціальних технологій.

7.5.4. Чорна оптимізація сайту

Чорна оптимізація сайту допускає використання методів, заборонених пошуковими системами, працюючи за принципом «на війні всі засоби підходять». Забороненими методами оптимізації сайтів є:[47]

- використання сторінок сформованих лише набором ключових фраз, які не мають сенсу для відвідувачів сайту – пошуковий спам;
- використання сторінок, які містять тільки посилання, linkfarm або посилальний спам;
- використання «невидимого» тексту (текст дрібним шрифтом, що зливається з кольором фону);
- використання клоакінга – підміни контенту сторінки так, що користувач і пошукова система бачать сторінки сайту зовсім по-різному;
- використання doorway-сторінок з автоматичним перенаправленням на цільовий сайт.

Використання цих методів просування швидко виводить сайт у топ, але виявлення пошуковою системою використання заборонених методів просування призводить до усунення сайту з пошукової видачі назавжди.

На сьогодні оптимізація сайту – не розкіш, а необхідність, для тих, хто піклується про просування сайту й розвиток свого бізнесу.

7.6. Інструменти для веб-аналітики

7.6.1. Веб-аналітика засобами Google Analytics

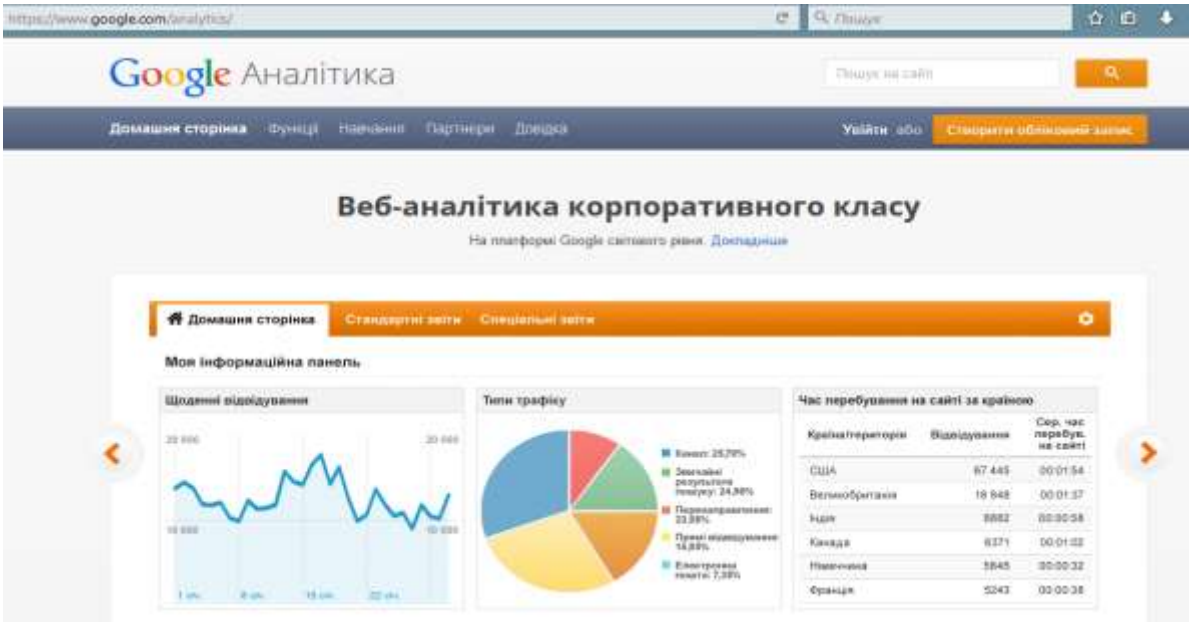
Google Analytics – це онлайн-сервіс від Google для веб-аналітики для розкручування сайту. Якщо потрібно проаналізувати відвідуваність сайту, визначити з яких джерел приходять відвідувачі, розрахувати ефективність ключових слів, то без веб-аналізу не обійтися.

На сьогодні існує безліч інструментів і методів веб-аналітики, проте лідируючі позиції по праву займає безкоштовний сервіс Google Analytics. Фахівці компанії **Google** постаралися втілити в цьому онлайн-сервісі максимум корисних можливостей для веб-аналітики. Google Analytics має інтуїтивно зрозумілий інтерфейс і наочність подання даних (графіки, діаграми, таблиці)[48].

Щоб почати користуватися перевагами Google Analytics, треба зареєструватися в сервісі https://www.google.com/analytics/ або https://analytics.google.com. Для цього увійти до облікового запису на gmail.com (напис-посилання *Увійти* праворуч від помаранчової кнопки *Створити обліковий запис*). У вікні, що з'явиться, натис-

[47] Оптимізація сайтів. URL: http://webstudio2u.net/ua/optimization/97-seo.html (дата звернення 29.09.2018).
[48] Розкрутка сайту. Google Analytics. URL: http://webstudio2u.net/ua/web-promotion/246-google-analytics.html (дата звернення 29.09.2018).

нути кнопку *Реєстрація*. Далі ввести власне ім'я, назву та URL-адресу сайту й інші параметри, після чого натиснути кнопку *Отримати ідентифікатор відстежування*.

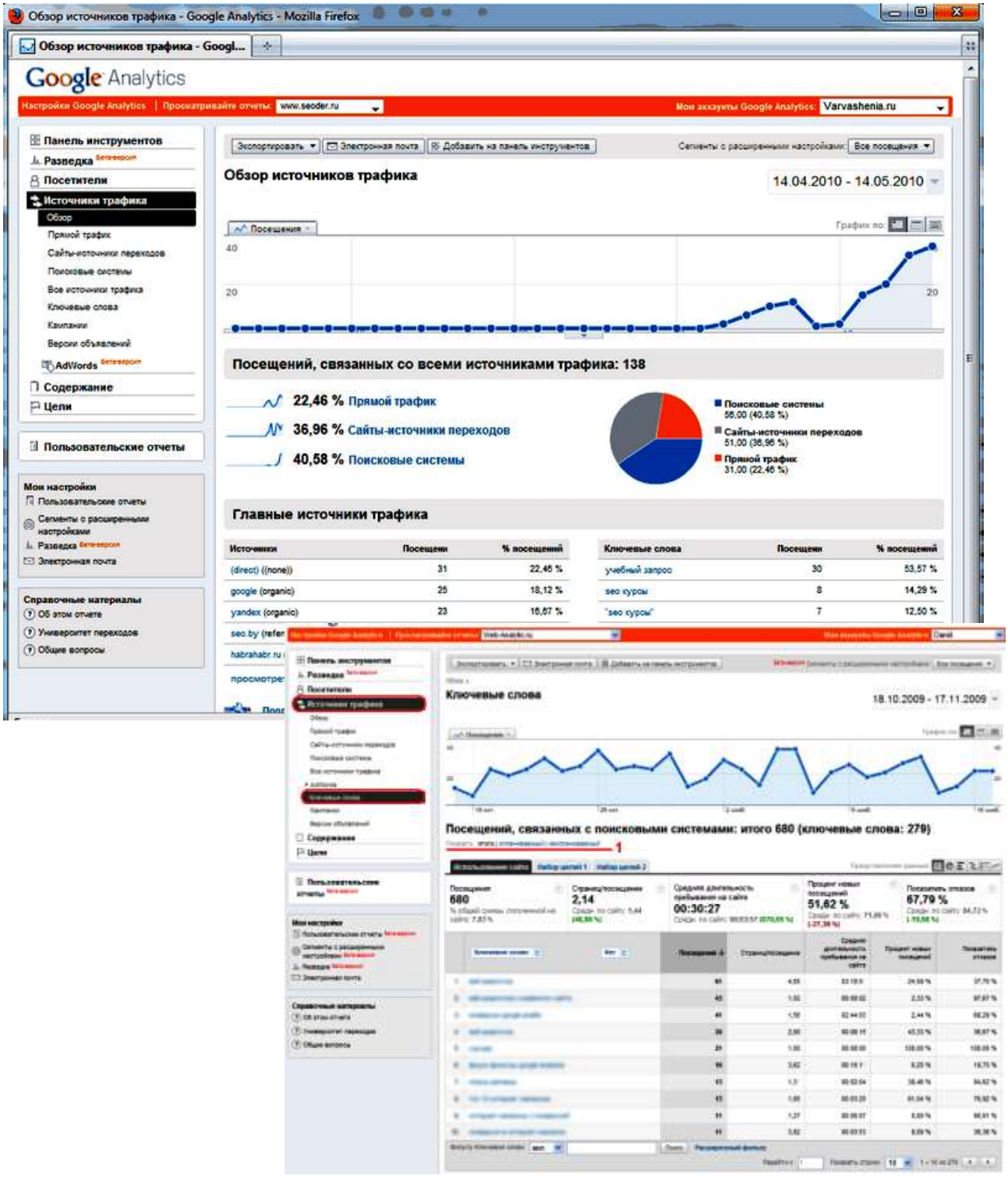


Після успішної реєстрації Google Analytics запропонує згенерований для Вашого сайту спеціальний **скрипт**, що треба вставити в HTML-код кожної сторінки, яку Ви бажаєте відстежувати засобами Google Analytics. Цей скрипт треба помістити безпосередньо перед тілом сторінки, тобто перед теґом <body>. При цьому скрипт ніяк не впливає на завантаження сторінки, не вимагає завантаження додаткових файлів. Текст скрипту може бути таким:

```
<script>
 (function(i,s,o,g,r,a,m)
 { i['GoogleAnalyticsObject']=r;
  i[r]=i[r]||function()
  { (i[r].q=i[r].q||[]).push(arguments)}, i[r].l=1*new Date();
    a=s.createElement(o), m=s.getElementsByTagName(o)[0];
    a.async=1;      a.src=g;      m.parentNode.insertBefore(a,m)
 })
 (window,document,'script','https://www.google-analytics.com/analytics.js','ga');
 ga('create', 'UA-91614276-1', 'auto');
 ga('send', 'pageview');
</script>
```

Після зазначених дій та оновлення вмісту сайту з цим скриптом зайшовши на свою сторінку Google Analytics, можна побачити невелику зведену таблицю огляду цього сайту. Щоб дослідити інформацію про сайт докладніше, треба клацнути посилання *Подивитися звіт* у колонці *Звіти*. Перейшовши за посиланням, можна потрапити на сторінку, на якій доступні такі вкладки:[49]

[49] Розкрутка сайту. Google Analytics. URL: http://webstudio2u.net/ua/web-promotion/246-google-analytics.html (дата звернення 29.09.2018).



– вкладка ***Панель інструментів*** слугує для візуалізації звітів проведеного веб-аналізу. На панелі можна в довільному порядку і кількості розміщувати різні звіти, такі як: огляд відвідувачів, огляд джерел трафіка, огляд браузерів тощо;

– вкладка ***Відвідувачі*** показує повну інформацію про відвідувачів сайту. За допомогою цієї вкладки можна дістати вичерпні відомості про кількість відвідувачів, тривалість їхнього перебування на сайті, про переглянуті ними сторінки, про можливості їхніх браузерів і параметри підключення до інтернету;

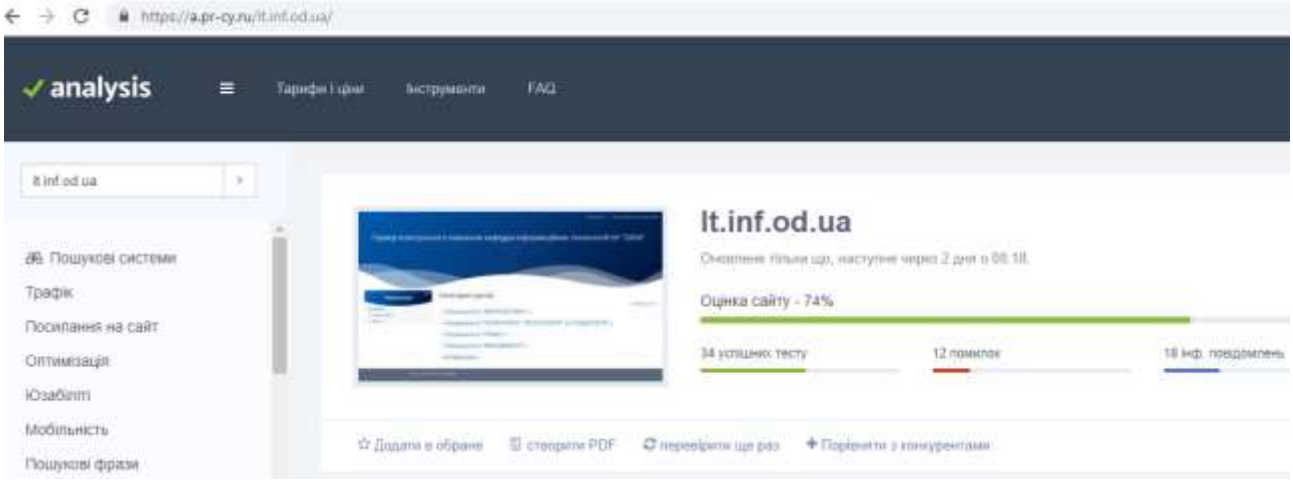
– вкладка ***Джерела трафіка*** дозволяє дізнатися, з яких сайтів приходять відвідувачі до сайту, якими пошуковими системами вони користуються, кількість прямого трафіка;

– графіки і таблиці вкладки ***Вміст*** показують, який вміст на сайті є найбільш популярним, які найпопулярніші сторінки входу і виходу, як відвідувачами здійснюється пошук по сайту. Також на цій вкладці можна відстежувати дії користувачів на сайті;

– ***Цілі*** – це сторінки, на які потрапляє користувач, після здійснення покупки, реєстрації або виконання іншого бажаної дії. Вони дозволяють оцінити, наскільки сайт відповідає цілям компанії. У Google Analytics можна задавати до 20-ти таких цілей.

7.6.2. Інші безкоштовні інструменти для веб-аналітики

1) **PR-CY** (http://pr-cy.ru) пропонує перевірити сайт: його швидкість, виявити помилки, встановити динаміку зростання за ключовими словами. Дозволяє самостійно провести аналіз сайту навіть без реєстрації на сервісі. Для цього потрібно ввести його адресу і через кілька секунд можна побачити повний звіт. Можна дізнатися: чи всі посилання працюють, наскільки доступна мобільна версія, який трафік, чи здійснено перевірку на віруси та багато іншого. PR-CY сформує перелік рекомендацій для оптимізації сайту. Є широкий спектр платних опцій.

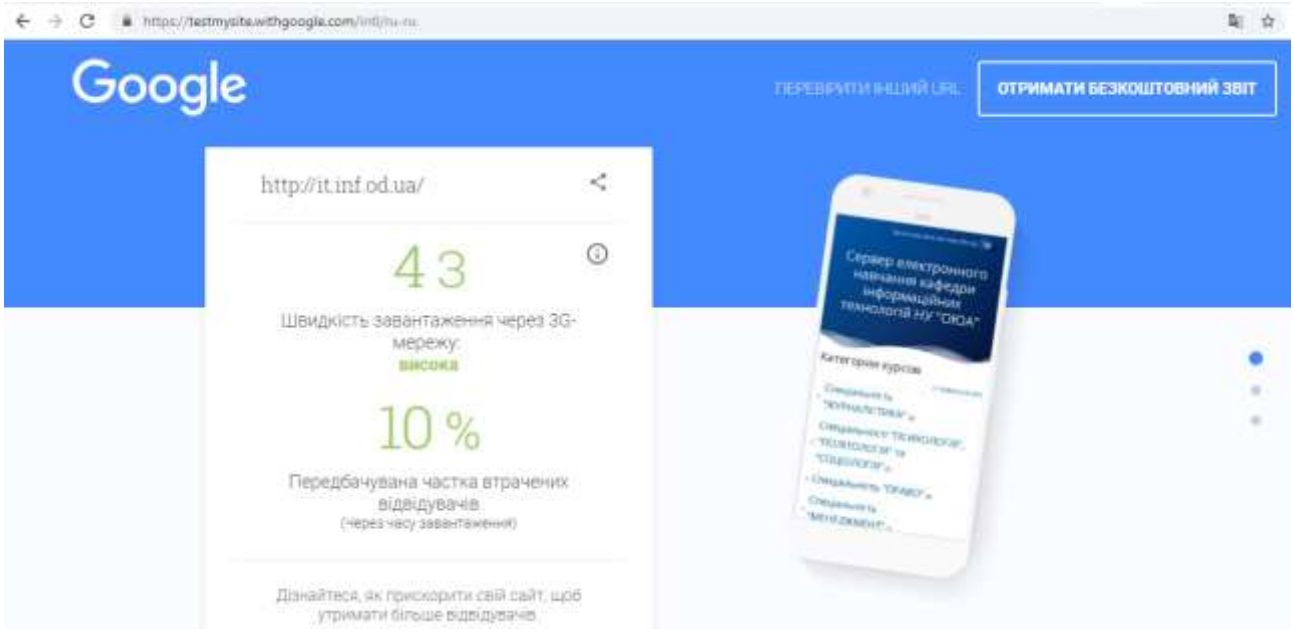


2) **Quill Engage** (https://www.quillengage.com/) – англомовний онлайн-сервіс, який пояснює дані Google Analytics і надає призначені для користувача звіти.

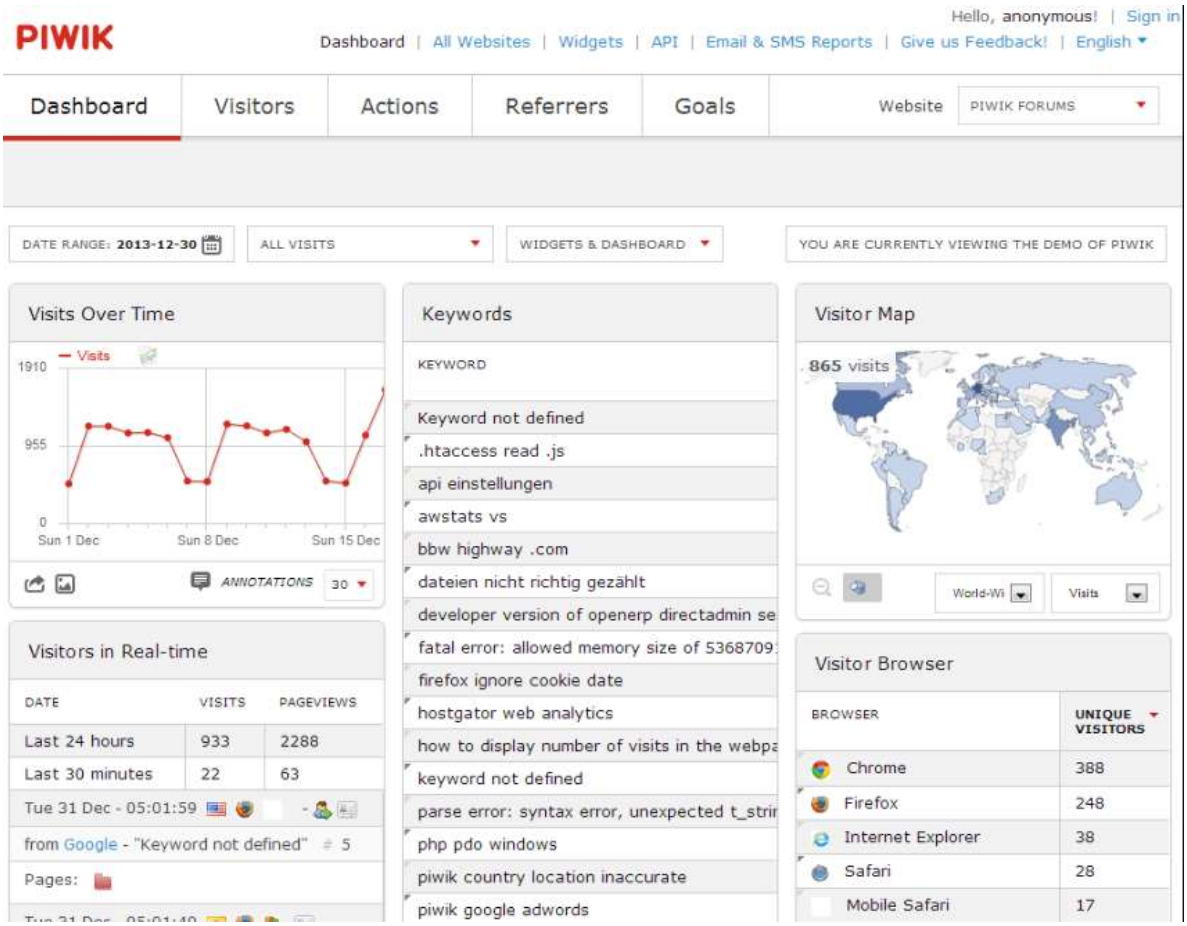
3) **Clicky** (https://clicky.com) виконує аналіз дій відвідувачів, дозволяє подивитися теплову карту (принцип роботи теплової карти в тому, що на сторінку сайту накладається шар, де теплими кольорами показані ті області, на які найчастіше потрапляє курсор користувачів, холодними кольорами – зони сайту, на які не звертають уваги). Можна промоніторити конверсії або нагадування про Ваш бренд. Має англомовний інтерфейс. Інструментом можна користуватися безкоштовно, якщо аналізувати тільки один сайт.

4) **Test My Site** (https://testmysite.thinkwithgoogle.com/intl/en-us) дозволяє перевірити швидкість завантаження того чи іншого сайту на мобільні пристрої, оскільки більшість сайтів через це втрачають половину своїх відвідувачів. Цей

сервіс безкоштовно сформує та надішле на електронну пошту звіт із рекомендаціями щодо покращення ефективності веб-сайту на всіх пристроях.



6) **Piwik** (https://piwik.org) – аналітична платформа сайтів із відкритим вихідним кодом, яку можна завантажити і встановити на свій сервер. Сервіс пропонує стандартний набір даних: запити і пошукові системи, веб-сайти, сторінки, операційні системи, браузери, роздільна здатність екрана, поведінка користувачів.



Сервіс Piwik автоматично формує звіти в PDF або HTML і надсилає їх на електронну пошту.

6) **Open Web Analytics** (http://www.openwebanalytics.com) – англомовний інструмент із відкритим вихідним кодом, який може бути завантажений і встановлений на свій хостинг. Його можна використовувати для декількох ресурсів: обмежень за кількістю даних немає. Доступною є інформація про останні відвідини ресурсу (докладні відомості про останнього відвідувача сайту із зазначенням місця розташування, типу браузера, переглянутих сторінок, тривалості відвідування). Є теплова карта кліків і запис рухів вказівника миші.

Версія з розширеним функціоналом доступна для сайтів на WordPress і MediaWiki. Плюс в тому, що всі дані зберігаються на Вашому власному сервері.

7) **GoingUp** (https://www.goingup.com/) пропонує веб-аналітику і SEO-інструментарій, доступна теплова карта кліків. Разом із цим сервісом бажано використовувати й інші інструменти веб-аналітики, оскільки в GoingUp не такий широкий спектр функцій.

7.6.3. Налаштування сайту на базі Wordpress для роботи з пошуковими машинами

Щоб веб-проект попав у перелік топ-ресурсів, які у числі перших видаються пошуковими системами, потрібно враховувати численні фактори із самого початку розроблення сайту.

Платформа WordPress має базові можливості для пошукової оптимізації. Так, назву кожного публікованого запису WordPress автоматично додає у тег <title> (разом з назвою сайту), а в самому тексті наділяє їй заголовний тег <h2>. При вставленні ілюстрацій можна заповнити поле *Тема зображення* й опціонально – поле *Підпис зображення*. Вміст цих полів запишеться в параметри title та alt тегу <img> і надасть пошуковим системам можливість визначити, що саме зображено на ілюстрації. Також візуальний редактор WordPress може виділяти ключові слова безпосередньо в текстах публікацій.[50]

Проте існують і спеціальні плагіни з потужнішими можливостями для пошукової оптимізації, наприклад, плагін *All in One SEO Pack* для WordPress.

Також рекомендується виконати комплекс кроків для SEO-оптимізації:

– встановити й активувати плагіни *Google XML Sitemaps* (https://wordpress.org/plugins/google-sitemap-generator/), *All in One SEO Pack* (https://wordpress.org/plugins/all-in-one-seo-pack/);
– підготувати яскравий короткий опис сайту;
– створити анонси, які коротко описують опубліковані матеріали;
– створити сторінку *Про нас* (*About*).

Досвід показує, що сайти, зроблені на WordPress CMS, індексуються краще, ніж сайти засновані на інших платформах, за рахунок організації даних та їх відображення. Платформа WordPress продумана максимально лояльно до

[50] Воропай А. Создание Web-сайта на базе WordPress CMS. URL: https://www.ibm.com/developerworks/ru/library/os-wordpress/index.html (дата обращения 29.09.2018).

пошукових систем, тому проекти, реалізовані на ній, буде нескладно просувати у провідних пошукових системах.

7.6.4. Плагіни веб-розробника

У кожного веб-розробника при виконанні тої чи іншої процедури завжди виникає потреба швидкого доступу до різної інформації: HTML-коду уподобаного сайту, кольору якогось блока, інформації про швидкість опрацювання і зчитування інформації з джерела тощо. Всі ці процедури можна виконувати вручну, що займатиме доволі багато часу, і процес може затягуватися на тижні.

На щастя, для всіх цих дій і маневрів існують різні плагіни, які інтегруються в різні браузери: Google Chrome, Mozilla FireFox, Opera тощо. Далі розглянемо популярні зараз плагіни.

1) Плагін **Firebug Lite** дозволяє переглядати і редагувати HTML, Java Script і CSS код будь-якого сайту буквально «на льоту», точно аналізувати використання і продуктивність мережі, не виходячи з браузера. Великим привілеєм є те, що редагувати і переглядати результат можна в самому плагіні, без внесення змін в оригінальні файли.

Скачати плагін для Firefox можна на сайті https://getfirebug.com/firebuglite, а для Google Chrome – на http://getfirebug.com/releases/lite/chrome/.

2) **iMacros for Chrome** – розширення, яке допомагає тестувати веб-сторінки. Замість того, щоб самостійно проробляти одні і ті самі дії на сторінці, розробнику потрібно лише записати необхідну послідовність дій в iMacros і запускати розширення тоді, коли виникає потреба. Розширення здатне працювати із сайтами, реалізованими засобами технологій Java, Flash, Flex, Ajax і Silverlight.

3) **Font Playground** – плагін для тих, хто полюбляє «погратись зі шрифтами». Він дозволяє експериментувати з усім спектром шрифтів із бібліотеки Google Fonts, не вносячи змін в код сторінки. Можна змінювати не лише сам шрифт, а і його розмір, стиль накреслення тощо. За його допомоги зручно перевіряти те, як виглядає шрифт, перед внесенням фактичних змін у код сторінки.

4) **Project Naptha** – розширення для Google Chrome, яке дозволяє виділяти і копіювати текст навіть з картинок. Корисне при роботі з вбудованим текстом.

5) **What Font** – плагін, який дозволяє миттєво визначити, який шрифт використаний на тій чи іншій сторінці, просто навівши вказівник миші на напис.

6) **YSlow** – інструмент, який не лише перевіряє швидкість завантаження тієї чи іншої веб-сторінки, а й підказує розробнику, що її гальмує.[51] Для цього розширення перевіряє сайт на відповідність 23 з 34 правил продуктивності,[52] сформульованих командою компанії Yahoo.

---

[51] 25 полезных Chrome-расширений для веб-разработчиков. URL: https://vc.ru/flood/8572-25-chrome-extensions (дата обращения 29.09.2019).

[52] Best Practices for Speeding Up Your Web Site. URL: https://developer.yahoo.com/performance/rules.html?guccounter=1 (accessed 29.09.2019).

7) **Web Developer** – набір корисних інструментів для керування елементами сайту у вигляді додаткової панелі. Web Developer Chrome Extension є офіційним дублем знаменитого доповнення Firefox. Дозволяє аналізувати веб-ресурс, тестувати код, змінювати параметри і зовнішній вигляд сторінки, контролювати кеш браузера, керувати cookie-файлами, досліджувати і підсвічувати веб-елементи, атрибути заголовків, інформацію про якорі на сторінці, що суттєво економить час.

8) **Web Developer checklist** перевіряє те, чи задовольняє сайт основним принципам SEO, чи достатньо він продуктивний і зручний для користувача. Результати перевірки подаються у вигляді своєрідного чек-списку, де можна побачити детальну інформацію та рекомендації по кожному з невиконаних пунктів, а також моментально виправити помилки.

9) **DevTools Autosave** дозволяє в автоматичному режимі зберігати будь-які змінення, внесені в код сторінки за допомогою інструментів Chrome DevTools, що допомагає розробникам заощадити значну кількість часу.

10) **Instant Wireframe –** розширення, за допомогою якого можна «перетворити» будь-яку сторінку на структурну схему компонування матеріалу – wireframe. Дозволяє розробникам і веб-дизайнерам не виходячи з браузера ознайомитися з компонуванням будь-якої сторінки в мережі.

11) **Ripple Emulator** – емулятор для Google Chrome, який дозволяє тестувати веб-сайти на різних мобільних платформах з різними роздільними здатностями екрана. Може бути використаний у поєднанні з іншими плагінами для тестування і налагодження веб-ресурсів.

12) **Streak** – плагін, який дозволяє перетворити поштову Gmail-скриньку на CRM-систему.[53] Можна відстежувати статус угод і переговорів, які ведуться в електронній пошті з контрагентами, використовувати Streak для оброблення запитів користувачів продукту і відстеження виправлень надісланих помилок тощо.

13) **Search Stackoverflow** – розширення для швидкого пошуку по популярному ресурсу для розробників Stack Overflow.

14) **PHP Ninja Manual** дозволяє отримати миттєвий доступ до документації по PHP 5.5 з браузера.

15) **PerfectPixel** «накладає» на веб-сторінку напівпрозору сітку і звіряє попіксельно по ній задані відстані. Також можна накладати інші зображення, наприклад, початковий макет, щоб переконатися, що така сторінка в точності йому відповідає:

16) **Code Cola** – інструмент для перегляду вихідного коду сторінок і редагування CSS-коду. Можна змінювати тіні, відступи та інше за допомогою перетягування повзунка. Після внесення змін можна скопіювати отриманий код і замінити його в коді сайту.

---

[53] **CRM** (від англ. Customer Relationship Management – управління відносинами з клієнтами) – це особливий підхід до ведення бізнесу, при якому на перше місце діяльності компанії ставиться клієнт. CRM-система – це спеціалізовані програмні засоби, які автоматизують бізнес-процеси, процедури і операції, що реалізують CRM-стратегію компанії.

17) Плагін **Video Slider** для WordPress дозволяє без навичок програмування створити відео-слайдер, що сприятиме залученню більшої кількості людей на веб-сайт. Плагін підтримує відео з Youtube, Vimeo, Vevo та MP4.

18) **Image Slider** для WordPress – це один з найкращих плагінів для створення повнофункціональних слайд-шоу із завантажених зображень. До тогож плагін дозволяє змінювати кольори, шрифти та розміри.

19) **Photo Gallery** – плагін для WordPress з широким спектром інструментів для роботи з галереєю зображень, зокрема ефекти висувних мініатюр, сучасні тенденції для портфоліо, відео та фотогалереї. Джерелами відео можуть використовуватись Youtube або Vimeo медіа ресурси. Плагін має численні макети і гнучкий дизайн інтерфейсу.

20) Функціональність **Autoptimize** дозволяє аналізувати і виправляти проблеми продуктивності сайту, тобто оптимізувати його. Він може агрегувати, мінімізувати і кешировати скрипти та стилі, вставляти CSS у заголовок сторінки за умовчанням тощо. Він також мінімізує сам HTML-код, що робить сторінку дійсно легкою і покращує продуктивність сайту.

21) **Yoast SEO** дозволяє побачити, як виглядатиме публікація чи сторінка у результатах пошуку. Плагін допоможе не лише збільшити рейтинги, а й підвищити показник кількості кліків для звичайних результатів пошуку.

22) **WP Fastest Cache** – російськомовний плагін для WordPress з широким функціоналом для чищення та скорочення коду від зайвого сміття, яке накопичується у блоґу. З його допомогою можна суттєво прискорити роботу блоґу.

23) **All-In-One-SEO-Pack** (AIOSP) є одним з найпотужніших плагінів для пошукової оптимізації. Завдяки цьому плагіну можна максимально оптимізувати свій сайт, блоґ під пошукові машини. З його допомогою топ-видача буде гарантована. AIOSP підтримує Google Analytics, автоматично повідомляє пошуковим системам Google і Bing про зміни на сайті. Плагін автоматично оптимізує заголовки для Google та інших пошукових систем, автоматично генерує метатеґи, дозволяє уникати типового дублювання контенту характерного для WordPress-блоґів. Все це працює автоматично після встановлення і не потребує додаткових налаштувань, що дуже зручно для початківців.

24) Плагін **Akismet** для WordPress-блоґів допоможе захистити блоґ від спаму.

25) **Logaster Logo Generator** допоможе створити логотип для будь-якої теми WordPress.

26) **Contact Form** допоможе налаштувати форму зворотного зв'язку у блоґу для відвідувачів.

27) **Image Watermark** дозволяє нанести на зображення водяні знаки, тим самим убезпечити блоґ від злодійства.

28) **TinyMCE Advanced** – зручний плагін для редагування постів.

29) **WP Security** – плагін для захисту блоґу на WordPress. Дозволяє захистити блоґ від хакерських атак і закриває «дірки» у блоґу.

30) **Chrome Sniffer** – розширення для браузера, яке визначає, які JavaScript-бібліотеки, фреймворк або CMS використовуються на ресурсі.

31) **User-Agent Switcher** дозволяє «маскувати» браузер Google Chrome під Internet Explorer, Opera або будь-який інший браузер.

32) **IE Tab** – вбудований емулятор Internet Explorer для Chrome, який дозволяє тестувати сайт у різних версіях IE, не покидаючи Chrome.

33) **PicMonkey** – простий і безкоштовний онлайн-редактор зображень. Дозволяє «захоплювати» зображення або робити скріншоти браузера і відразу ж редагувати їх за допомогою розширення для Chrome.

34) **Chrome Daltonize** допомагає адаптувати веб-сервіси для тих користувачів, які страждають на дальтонізм, демонструючи розробнику, як сайт бачать ті, хто страждає на це захворювання. Дозволяє веб-дизайнерам і розробникам створювати більш доступні сервіси.

35) **Page Ruler** – простий інструмент, який допомагає визначити висоту, ширину і положення будь-якого елемента на сторінці.

36) **Check My Links** перевіряє веб-сторінку на наявність «битих» або неправильних посилань.

37) **Flickr Tab** допомагає не стільки в розробці, скільки у пошуку натхнення і хороших фотографій. Показує на кожній новій вкладці в Google Chrome одне зображення із сервісу Flickr. При натисканні на нього користувач переходить на сторінку автора, де може ознайомитися з іншими його роботами.

38) **Google Art Project –** розширення для браузера, схоже на попередній плагін у цьому списку, тільки замість фотографій з Flickr у новій вкладці користувач бачитиме визнані твори мистецтва, наприклад, полотна пензля Ван Гога або Мане.

39) **Data Saver** – офіційне розширення від Google для стискання трафіка, яке економить трафік у браузері Google Chrome.

40) **ColorZilla** дозволяє визначити прямо у браузері, який конкретно колір використовується на сторінці. ColorZilla дуже корисний, коли потрібно звірити реально використовувані кольори зі специфікацією.

41) **Spell Checker Spell Checker –** розширення для перевірки орфографії. Перевіряє, чи правильно написані слова на сторінці, і пропонує внести правки для слів із помилками. Це простеньке розширення підтримує 12 мов, і дозволяє додавати свої слова у його словник.

42) **Grammarly** допомагає з вичитуванням тексту, коли текст пишеться онлайн – неважливо, пошта це, коментар або пост у блоґу. У процесі написання Grammarly перевіряє орфографію і граматику і підсвічує помилки прямо у браузері.

43) **XSS Rays** широко використовується тестувальниками безпеки. По суті це чистий Javascript XSS сканер, який допомагає виявити уразливості XSS.[54]

[54] **XSS** (від англ. Cross-Site Scripting – «міжсайтовий скриптинг») – тип атаки на веб-системи у вигляді вбудовування у веб-сторінку шкідливого коду (який буде виконаний на комп'ютері користувача при відкриванні ним цієї сторінки) і взаємодії цього коду з веб-сервером зловмисника.

Плагін аналізує всі посилання і форми завантаженої сторінки і перевіряє XSS за параметрами GET і POST.[55]

44) **Request Maker** – основне розширення Chrome для тестування на проникнення. Дозволяє створювати нові запити, відловлювати запити, зроблені сторінками, змінювати заголовки і дані POST.[56]

45) **d3coder** – ще один плагін для тестування на проникнення. Дозволяє кодувати і декодувати вибраний текст через контекстне меню прямо з Chrome. Достатньо скопіювати текст і вибрати в меню *Перетворити*. Після перетворення d3coder копіює отриманий текст у буфер обміну і можна його вставити. Це розширення дозволяє перетворювати текст у різні формати, наприклад: base64, rot13, CRC32, UNIX-час.

46) **Site Spiderr** повідомляє про биті посилання на сторінці. Суттєво економить час, оскільки не доводиться вручну перевіряти, чи всі посилання працюють. Є можливість додавати регулярні вирази.

47) **Screencastify** – ще одне рекомендоване для встановлення розширення для тестувальників. Воно дозволяє робити запис екрана під час тестування. Результатом є відеозапис поведінки користувача, який можна передати розробникам як підтвердження бага. Треба просто натиснути «записати», і розширення почне записувати те, що відбувається у Вашій вкладці.

48) **Resolution Test** допоможе при тестуванні веб-проектів на різних роздільних здатностях і розмірах екрана. Вибрати поширену роздільну здатність можна зі списку або ж ввести потрібні розміри вручну. Цей плагін змінює розміри браузера й емулює веб-сторінку у потрібній роздільній здатності екрана.

49) **Wave evaluation tool** – зручний інструмент оцінки доступності, який тестує сайт на відповідність принципам WCAG[57], вбудовує додатковий функціонал прямо у браузер і робить видимим зворотний зв'язок по доступності сторінки (вставляє іконки та індикатори прямо на сайт).

50) **Accessibility Developer Tools** додає «Оцінку доступності» і бічну панель доступності до інструментів розробника Chrome. При запуску оцінки доступності розширення формує список правил WCAG 2.0, порушених на сторінці.

Ці та інші плагіни дуже корисні, а іноді просто незамінні для веб-розробника.

---

[55] Для організації взаємодії користувача з мережею Інтернет, розробник сайту повинен передбачити передавання запитів від користувача сайту на сервер, де розташований цей сайт. Запити бувають двох видів: GET- і POST-запити.

[56] 40+ супер-полезных расширений Chrome для тестировщиков. URL: http://software-testing.ru/library/testing/general-testing/2242-useful-google-chrome-extensions-testing-software (дата обращения 29.09.2018).

[57] WCAG2.0 (Web Content Accessibility Guidelines – Керівництво щодо забезпечення доступності веб-контенту) надає рекомендації, як зробити веб-контент більш доступним для широкого загалу користувачів з обмеженими можливостями, зокрема осіб з інвалідністю по зору, слуху, опорно-рухової системи, а також осіб із порушеннями функцій мови, ментальної сфери та з неврологічними порушеннями. WCAG 2.0 є рекомендацією W3C. WCAG 2.0 містить 4 основні принципи: сприйняття, керованість, зрозумілість і надійність.

Контрольні запитання

1) Які складові має аналіз роботи сайту?
2) Яке призначення SEO-аналізу сайту?
3) Яка мета SEO-просування?
4) Які можливості надає аналіз відвідуваності сайту?
5) Яке завдання веб-аналітики?
6) Чому повний аналіз та оцінка відвідуваності сайту може сприяти його розкрутці?
7) Що таке юзабіліті сайту?
8) Як провести юзабіліті аудит?
9) Чому потрібно проводити оптимізацію сайтів?
10) Назвати методи оптимізації сайту та його просування.
11) Охарактеризувати білу оптимізацію сайту.
12) Які методи сірої оптимізації сайту?
13) Що розуміють під помаранчевою оптимізацією сайтів?
14) Назвати методи чорної оптимізації сайтів.
15) Що таке SEO-аудит?
16) Коли і навіщо проводять SEO-аудит?
17) Який алгоритм виконання SEO-аудиту?
18) Назвати та охарактеризувати безкоштовні інструменти для веб-аналітики.
19) Яке призначення оптимізації сайтів?
20) У чому полягає розкручування і просування сайтів?
21) Що таке SEO-оптимізація?
22) Яка основна мета SEO?
23) Яке призначення плагінів у WordPress?
24) Що таке плагін для WordPress?
25) Як додати новий плагін на WordPress?

Підсумковий тест

1. Що означає HTML?
   1) Мова розмітки гіпертексту.
   2) Гіперпосилання та мова розмітки тексту.
   3) Мова розмітки для домашнього інструмента.

2. Хто розробляє веб-стандарти?
   1) Mozilla.
   2) Google.
   3) Консорціум World Wide Web.
   4) Microsoft.

3. Виберіть правильний HTML-тег для найбільшого заголовка.
   1) <h1>
   2) <head>
   3) <h6>
   4) <header>

4. Який HTML-тег вставить розрив рядка?
   1) <lb>
   2)
   3) <break>
   4) <endl>

5. Який HTML-тег правильно задасть фоновий колір веб-сторінки?
   1) <background> yellow </ background>
   2) <body bg = "yellow">
   3) <body style = "background-color: yellow;">
   4) <body bgcolor= "yellow">

6. Виберіть правильний елемент HTML для визначення важливого тексту.
   1) <b>
   2) <important>
   3) <i>
   4)

7. Виберіть правильний HTML-тег для визначення підкресленого тексту.
   1) <i>
   2) <italic>
   3)
   4) <u>

8. Виберіть правильний елемент HTML для створення гіперпосилання?
   1) <a> http://www.w3schools.com </ a>
   2) W3Schools.com
   3) W3schools

9. Який символ використовується у позначенні кінцевого тегу?
   1) *
   2) <
   3) ^
   4) /

10. Як відкрити посилання у новій вкладці браузера?

1) <a href="url" new>
2) <a href="url" target="new">
3) <a href="url" target="_blank">

11. Які з цих тегів є всі елементами <table>?

1) <table> <tr> <td>
2) <table> <tr> <tt>
3) <table> <head> <tfoot>
4) <thead> <body> <tr>

12. Вбудовані (inline) елементи зазвичай відображаються не з нового рядка (лінії).

1) Так.
2) Ні.

13. Яким тегом можна зробити нумерований список?

1)
2)
3) <list>
4) <dl>

14. Яким тегом можна зробити маркірований список?

1)
2)
3) <dl>
4) <list>

15. Який правильний HTML-тег для створення прапорця на формі?

1) <checkbox>
2) <check>
3) <input type = "checkbox">
4) <input type = "check">

16. Як правильно сформувати поле для введення тексту на формі?

1) <textinput type = "text">
2) <textfield>
3) <input type = "textfield">
4) <input type = "text">

17. Який елемент HTML створить випадний список?

1) <list>
3) <select>
3) <input type = "list">
4) <input type = "dropdown">

18. Який елемент HTML створить текстову область у декілька рядків?

1) <input type = "textbox">
2) <input type = "textarea">
3) <textarea>
4) <text>

19. Який тег дозволить правильно вставити зображення на веб-сторінку?

1) <img href = "image.gif" alt = "MyImage">
2) <img alt = "MyImage"> image.gif </ img>
3) <image src = "image.gif" alt = "MyImage">
4) <img src = "image.gif" alt = "MyImage">

20. Який теґ задасть фонове зображення веб-сторінки?

1) <background img = "background.gif">
2) <body background = "background.gif">
3) <body bg = "background.gif">

21. <iframe> використовується для вбудовування вмісту фрейму у веб-сторінку.

1) Не існує теґу <iframe>.
2) Ні.
3) Так.

22. Коментар HTML починається з <!- і закінчується ->

1) Так.
2) Ні.

23. Елементи блока зазвичай відображаються без запуску нової лінії.

1) Ні.
2) Так.

24. Який елемент HTML визначає назву документа на вкладці?

1) <meta>
2) <title>
3) <head>
4) <header>

25. Який атрибут HTML вказує альтернативний текст для зображення, якщо зображення не може відображатися?

1) longdesc
2) src
3) alt
4) title

26. Який тип документа правильний для HTML5?

1) <! DOCTYPE HTML5>
2) <! DOCTYPE html>
3) <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 5.0 // EN"
"http://www.w3.org/TR/html5/strict.dtd">

27. Який HTML-елемент використовується для вказування нижнього колонтитула документа чи розділу?

1) <section>
2) <footer>
3) <bottom>
4) <foot>

28. У HTML можна вставляти зображення SVG безпосередньо на сторінку.

1) Так
2) Ні

29. Який теґ HTML призначений для відтворення відеофайлів?

1) <video>
2) <movie>
3) <media>
4) <img>

30. Який тег HTML призначений для відтворення звукових файлів?

1) <sound>
2) <mp3>
3) <audio>
4) <img>

31. Для чого використовується загальний HTML-атрибут "contenteditable"?

1) Вказує, чи можна вміст елемента редагувати, чи ні.
2) Формує контекстне меню для елемента. Меню з'являється, коли користувач клацає правою кнопкою миші на елементі.
3) Повертає позицію шуканого вмісту всередині рядка.
4) Оновлює вміст із сервера.

32. Що означає CSS?

1) Барвисті таблички стилів.
2) Каскадні таблиці стилів.
3) Таблиці комп'ютерних стилів.
4) Креативні стилі.

33. Який HTML-тег створить посилання на зовнішню таблицю стилів?

1) <stylesheet> mystyle.css </ stylesheet>
2) <style src = "mystyle.css">
3) <link rel = "stylesheet" type = "text / css" href = "mystyle.css">

34. Де в HTML-документі є правильне місце для посилання на зовнішню таблицю стилів?

1) У розділі <body>.
2) У розділі <head>.
3) Наприкінці документа.

35. Який тег HTML використовується для визначення внутрішньої таблиці стилів?

1) <script>
2) <style>
3) <css>
4) <table>

36. Який атрибут HTML використовують для визначення вбудованих стилів?

1) style
2) styles
3) font
4) class

37. Який правильний синтаксис CSS?

1) body {color: black;}
2) { body; color: black;}
3) { body: color=black; }
4) body: color=black;

38. Як правильно вставити коментар у файл CSS?

1) // це коментар
2) / * це коментар * /
3) "це коментар
4) // це коментар //

39. Яка властивість CSS використовується для змінення фонового кольору?

1) color
2) background-color
3) bgcolor
4) bg

40. Як задати фоновий колір для всіх елементів <h1>?

1) all.h1 {background-color:#FFFFFF;}
2) h1.all {background-color:#FFFFFF;}
3) h1 {background-color:#FFFFFF;}

41. Яка властивість CSS дозволяє змінити колір тексту елемента?

1) color
2) text-color
3) fgcolor
4) text

42. Який атрибут CSS задає розмір тексту?

1) text-style
2) text-size
3) font-size
4) font-style

43. Який правильний синтаксис CSS-правила, яке зробить усі елементи <p> напівжирними?

1) p {font-weight: bold;}
2) p {text-size: bold;}
3) <p style = "font-size: bold;">
4) <p style = "text-size: bold;">

44. Яка властивість CSS використовується для змінення шрифту елемента?

1) font-weight
2) font-family
3) font-style
4) font

45. Яке CSS-правило зробить текст напівжирним?

1) font-weight:bold;
2) style:bold;
3) font:bold;
4) font:strong;

46. Яка властивість CSS використовується для змінення лівого зовнішнього поля (відступу) елемента?

1) margin-left
2) indent
3) padding-left
4) left

47. Яка властивість CSS використовується для змінення лівого внутрішнього поля (відступу) елемента?

1) margin-left
2) indent
3) padding-left
4) left

48. Чи дозволяється використовувати від'ємні значення при використанні властивості padding?

1) Ні.
2) Так.

49. Як створити маркірований список з квадратними маркерами?

1) list-type: square;
2) list: square;
3) list-style-type: square;
4) style-type: square;

50. Вибрати елемент з ідентифікатором "demo"?

1) *demo
2) #demo
3) demo
4) .demo

51. Вибрати елемент із класом "test"?

1) #test
2) *test
3) .test
4) test

52. Що таке IP-адреса?

1) Програма, яка здійснює перетворення доменного імені на цифрову IP-адресу та навпаки.
2) Власний унікальний номер (група з чотирьох чисел від 0 до 255), який надається кожному комп'ютеру при підключенні до інтернету.
3) Доменне ім'я сайту.

53. Що називають доменним ім'ям?

1) Програма, яка здійснює перетворення доменного імені на цифрову IP-адресу та навпаки.
2) Власний унікальний номер (група з чотирьох чисел від 0 до 255), який надається кожному комп'ютеру при підключенні до інтернету.
3) Унікальний ідентифікатор у вигляді поєднання символів, який надається певній IP-адресі для ідентифікації сайту серед безлічі інших.

53. Що таке WYSIWYG-редактор?

1) Візуальний редактор коду, за допомогою якого можна створювати веб-сторінки без знання HTML.
2) Текстовий редактор, призначений для роботи з текстом.
3) Редактор CSS.

54. Що таке хостинг?

1) Сукупність веб-сторінок, доступних в інтернет-мережі.
2) Послуга надання ресурсів для розміщення інформації на інтернет-сервер.
3) Перетворення доменного імені на цифрову IP-адресу та навпаки.

55. Яке призначення WordPress?

1) Конструктор сайтів, засобами якого можна створювати сайти «з нуля» і керувати їхнім вмістом без технічних навиків і знань.

2) Система керування вмістом сайту з відкритим вихідним кодом.
3) «Движок» сайту.

56. Що таке блог?
1) Тип веб-ресурсу, який характеризується як публічний онлайн-журнал.
2) Конструктор сайтів.
3) Персональний сайт, основним вмістом якого є власні дописи, зображення, відео, особисті фотографії.

57. Для чого в WordPress потрібна база даних MySQL?
1) Для завантаження сайту.
2) Не потрібна взагалі.
3) Для зберігання даних.

58. Які розрізняють структури сайтів?
1) Лінійна.
2) Ієрархічна.
3) Замкнена.
4) Довільна.

59. Коли доречно застосовувати лінійну структуру сайту?
1) У разі послідовного подання інформації, наприклад, про товари та послуги або матеріали навчального посібника.
2) Для сайтів, що містять різну за тематикою інформацію: каталогів, зібрань статей з різних тем або добірок.

60. Коли доречно застосовувати ієрархічну структури сайтів?
1) У разі послідовного подання інформації, наприклад, про товари та послуги або матеріали навчального посібника.
2) Для сайтів, що містять різну за тематикою інформацію: каталогів, зібрань статей з різних тем або добірок.

61. Яка структура сайтів є найпоширенішою?
1) Лінійна.
2) Ієрархічна.
3) Замкнена.
4) Довільна.

62. Що перевіряють на етапі тестування сайту?
1) Правильність роботи усіх гіперпосилань.
2) Зручність навігації сайтом.
3) Чи відкриваються при відкритті сторінок графічні зображення.
4) Зручність розташування матеріалів на сторінках.

63. Навіщо виконувати адміністрування ресурсу?
1) Для підняття рейтингу й залучення відвідувачів до створеного сайту.
2) Для оновлення вмісту.

3) Для розкрутки й просування сайту.

64. Яка мета розкрутки сайту?

1) Перетворити інтернет-ресурс зі звичайного атрибута компанії на ефективний маркетинговий інструмент.
2) Підвищення позиції сайту у видачі пошукових систем.
3) Ріст відвідуваності сайту за запитами.

65. Які завдання адміністратора сайту?

1) Інформаційна підтримка: наповнення сайту новим контентом, редагування наявних матеріалів.
2) Роботи з модернізації, які не потребують кардинальних змін дизайну або функціональності сайту.
3) Технічна підтримка: своєчасне оновлення компонентів і модулів, захист від мережевих атак тощо.

66. Яка мета оптимізації сайту?

1) Досягти високої релевантності сторінок сайту за важливими для ключовими запитами у пошукових системах.
2) Досягти високої відвідуваності сайту.
3) Підвищення позиції сайту у видачі пошукових систем.

67. Навіщо навколо тексту залишати порожній простір, якщо це так?

1) Наявність «повітря» навколо тексту знижує напругу, завдяки чому стає набагато легше сконцентруватися на суті тексту.
2) Для економії трафіка.
3) Не треба залишати порожній простір.

69. Які критерії якості бізнес-сайтів?

1) Середня кількість відвідувачів сайту на день.
2) Середня кількість сторінок сайту, які переглянули відвідувачі.
3) Час роботи сайту у штатному режимі без потреби його редизайну.
4) Середній час, який відвідувач провів на сайті.

70. За якими критеріями перевіряють код сторінки?

1) Наявність помилок у коді HTML.
2) Працездатність посилань.
3) Відповідність HTML-коду сучасним стандартам мови.
4) Перевантаженість коду.
5) Відповідність таблиць стилів CSS сучасним стандартам.

71. Що перевіряють при оцінюванні дизайну сайту?

1) Якість навігації на сайті.

2) Зайві елементи у дизайні сторінок сайту.
3) Коректність відображення сторінки у різних браузерах.
4) Перевантаженість коду.

72. Вибрати основні елементи веб-дизайну.

1) Простір.
2) Колір.
3) Світлотінь.
4) Лінія.
5) Шрифт.
6) Розмір.

73. Вибрати принципи веб-дизайну, які визначають правила взаємодії всіх елементів.

1) Акцент.
2) Баланс.
3) Стиль.
4) Напрямок уваги.
5) Масштаб.
6) Контраст.
7) Точність.
8) Зручність сприйняття.
9) Пропорції.
10) Ритм.

74. Що таке принцип балансу у веб-дизайні?

1) Візуальне навантаження, яке створює групування елементів у дизайні, повинне бути рівномірно розподілене в рамках веб-сторінки.
2) Веб-сайт не має виглядати порожнім, однак, перевантаження графікою ще більше погіршить ситуацію.
3) Текст має бути збалансований зображенням.

75. Які види балансу бувають у веб-дизайні?

1) Симетричний.
2) Лінійний.
3) Асиметричний.
4) Круговий.

76. Чим градієнти 2.0 відрізняються від попередніх?

1) Вони з чистими і насиченими кольорами (без домішку брудного відтінка).
2) Не відрізняються.

77. У чому специфіка SVG-графіки?

1) SVG-зображення масштабуються без втрат якості.
2) SVG-зображення легко редагуються засобами CSS.
3) Файли SVG-зображень невеликі за розміром.
4) SVG – це векторний формат, який будується з набору об’єктів, на зразок ліній, кривих, прямокутників, кіл тощо.
5) Галуззю застосування SVG-графіки є масштабовані зображення, логотипи, ілюстрації, графіки і діаграми.
6) 3D-візуалізація у двовимірному просторі веб-сторінки.

78. Що таке ізометрія у дизайні?

1) 3D-візуалізація у двовимірному просторі веб-сторінки.
2) Це векторний формат, який будується з набору об’єктів, на зразок ліній, кривих, прямокутників, кіл тощо.
3) Безкордонний дизайн.

79. Які сучасні тенденції щодо використання кольорів у веб-дизайні?

1) Веб-кольори стають більш авантюрними з яскравими та блискучими відтінками.
2) Прогрес у використанні яскравих технологій спричинила поява більш якісних екранів смартфонів із високою роздільною здатністю, які здатні відображати все більш яскраві кольори.
3) Яскраві кольори можуть бути використані, щоб миттєво привернути увагу та відрізняти веб-сайт від конкурентів.

80. Які переваги використання відсутності меж у сучасному веб-дизайні?

1) Безкордонний дизайн пропонує альтернативу традиційному підходу.
2) Фігури з закругленими кутами, зокрема, довгасті форми, надають привабливості та індивідуальності дизайну.
3) В минуле відходить розбивка сторінки різноманітними фоновими кольорами або повнорозмірними зображеннями. Замість цього один колір тла простягається від верхньої частини донизу сторінки та створює бездоганний вигляд.
4) При безкордонному дизайні око спрямовується на сторінку без будь-яких перерв, що чудово працює з чуйним дизайном, оскільки не потрібно турбуватися про будь-які фонові елементи.

81. Чим зумовлена сучасна тенденція використання нестандартних дизайнерських шрифтів?

1) Додавання сайту індивідуальності та передачі важливого посилу.
2) Більшість браузерів вже підтримують CSS стилізовані нестандартні гарнітури шрифтів.
3) Великі літери, шрифтові контрасти sans serif & serif поліпшують UX-дизайн, а головне спонукають читати.

82. Які складові має аналіз роботи сайту?

1) SEO-аналіз сайту.
2) Юзабіліті-аналіз.
3) Виявлення помилок.
4) Аналіз відвідуваності.
5) Технічний аналіз.
6) Визначення рейтингу.

83. Яке призначення SEO-аналізу сайту?

1) Пошукова оптимізація сайтів.
2) Оптимізація внутрішніх і зовнішніх параметрів сайту, які врахову-

ються пошуковими системами для оцінки релевантності, з метою просування сайту в топ і зростання відвідуваності сайту.
3) Конверсія сайту.

84. Яка мета SEO-просування?

1) Безкордонний дизайн.
2) Досягнення першої сторінки результатів пошуку за певним запитом.
3) Додавання сайту індивідуальності та передачі важливого посилу.

85. Які можливості надає аналіз відвідуваності сайту?

1) Додавання сайту індивідуальності та передачі важливого посилу.
2) Дозволяє дізнатися про те, звідки приходять відвідувачі на сайт, скільки часу проводять на його сторінках, з яких сторінок йдуть тощо.
3) З його допомогою можна не лише оцінювати поточну відвідуваність сайту, а й прогнозувати майбутню.

86. Яке завдання веб-аналітики?

1) Додавання сайту індивідуальності та передачі важливого посилу.
2) Повний аналіз та оцінка відвідуваності сайту для подальшого його розкручування.
3) Аналіз ефективності рекламної кампанії.
4) Оптимізація сторінок сайту для підвищення коефіцієнта конверсії відвідувачів у покупців.

87. Що таке юзабіліті сайту?

1) Показник зручності користування сайтом.
2) Додавання сайту індивідуальності та передачі важливого посилу.
3) Аналіз ефективності рекламної кампанії.

88. Назвати методи оптимізації сайту та його просування.

1) Біла.
2) Блакитна.
3) Помаранчева.
4) Сіра.
5) Червона.
6) Чорна.

89. Яка оптимізація дозволена пошуковими системами?

1) Біла.
2) Блакитна.
3) Помаранчева.
4) Сіра.
5) Червона.
6) Чорна.

90. Яка оптимізація сайту допускає використання методів, заборонених пошуковими системами, працюючи за принципом «на війні всі засоби підходять»?

1) Біла.
2) Помаранчева.
3) Сіра.
4) Чорна.

Список використаних і рекомендованих джерел

1) Трофименко О.Г., Козін О.Б. Веб-дизайн та HTML-програмування: навч.-метод. посібник. Одеса: Фенікс, 2017. 194 с.
2) 10 бесплатных инструментов для веб-аналитики. URL: https://te-st.ru/2017/01/18/10-free-web-analytics-tool (дата обращения 14.10.2018).
3) 10 бесплатных шаблонов на HTML5 и CSS3. URL: http://postovoy.net/10-besplatnyh-shablonov-na-html5-i-css3.html (дата обращения 14.10.2018).
4) 10 лучших бесплатных HTML-редакторов. URL: https://techrocks.ru/2017/09/28/10-best-free-html-editors/ (дата обращения 12.07.2018).
5) 12 huge web design trends for 2018. URL: https://www.creativebloq.com/features/web-design-trends (accessed 01.12.2018).
6) 15 Web Design Trends to Watch in 2018. URL: https://blog.hubspot.com/marketing/web-design-trends-2017 (accessed 14.09.2018).
7) 19 web design trends for 2018. URL: https://webflow.com/blog/19-web-design-trends-for-2018 (accessed 18.09.2018).
8) 23 бесплатных адаптивных HTML шаблона. URL: http://postovoy.net/18.html (дата обращения 14.10.2018).
9) 25 полезных Chrome-расширений для веб-разработчиков. URL: https://vc.ru/flood/8572-25-chrome-extensions (дата обращения 29.09.2019).
10) 37 адаптивных HTML5 CSS3 шаблона. URL: http://postovoy.net/25.html (дата обращения 14.10.2018).
11) 40+ супер-полезных расширений Chrome для тестировщиков. URL: http://software-testing.ru/library/testing/general-testing/2242-useful-google-chrome-extensions-testing-software (дата обращения 29.09.2018).
12) 7 CSS-анимаций с использованием SVG + сайты применяющие эффект. URL: http://seo-design.net/css-markup/css-animations-with-svg-websites-uses-it (дата обращения 11.09.2018).
13) 7 убийственных ошибок дизайна вашего сайта. URL: https://www.trendsmarketing.ru/single-post/8-chastih-oshibok-v-dizayne-saytov (дата обращения 14.09.2018).
14) 9 cutting-edge web design trends for 2018. URL: https://99designs.com/blog/trends/web-design-trends-2018/#mobile-first (accessed 17.09.2018).
15) 9 основних принципів чуйного веб-дизайну. URL: http://it-ua.info/news/2014/11/14/9-osnovnih-principv-chuynogo-veb-dizaynu.html (дата звернення 27.07.2018).
16) A brief history of web design for designers. URL: http://blog.froont.com/brief-history-of-web-design-for-designers (accessed 17.09.2018).

17) A Complete Guide to Flexbox. URL: https://css-tricks.com/snippets/css/a-guide-to-flexbox (accessed 26.07.2018).
18) A vocabulary and associated APIs for HTML and XHTML. W3C. URL: https://www.w3.org/TR/html5 (accessed 14.10.2018).
19) Adaptive vs. Responsive Design. URL: https://www.interaction-design.org/literature/article/adaptive-vs-responsive-design (accessed 27.07.2018).
20) Adaptive web design. URL: https://www.jdvorak.eu/adaptive-web-design/ (accessed 27.07.2018).
21) Best Practices for Speeding Up Your Web Site. URL: https://developer.yahoo.com/performance/rules.html?guccounter=1 (accessed 29.09.2019).
22) Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper. January 30, 2015. URL: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html (accessed 26.07.2018).
23) CodePen is a playground for the fron end web. URL: http://codepen.io/ (accessed 14.10.2018).
24) CSS Cascading and Inheritance Level 4. URL: https://drafts.csswg.org/css-cascade (accessed 18.07.2018).
25) CSS Flexible Box Layout Module. URL: https://caniuse.com/#feat=flexbox (accessed 26.07.2018).
26) CSS Snapshot 2017. URL: https://www.w3.org/TR/CSS/ (accessed 18.07.2018).
27) CSS Tutorial. URL: www.w3schools.com/css (accessed 14.10.2018).
28) CSS справочник. URL: http://css.manual.ru (дата обращения 14.10.2018).
29) CSS: Вікіпідручник. URL: http://uk.wikipedia.org/wiki/CSS (дата звернення 14.10.2018).
30) CSS-LIVE: жизнь во фронтенде. URL: http://css-live.ru (дата обращения 14.10.2018).
31) Cybernetic analytic system. URL: http://cys.ru (accessed 28.06.2018).
32) Designing for 10000 Screens: 4 Layout Tips for Responsive Web Design. URL: http://blog.venturepact.com/designing-for-10000-screens-4-layout-tips-for-responsive-web-design (accessed 27.07.2018).
33) Flexbugs. URL: https://github.com/philipwalton/flexbugs#flexbug-9 (accessed 26.07.2018).
34) Gfycat відкриває нові обрії якості GIF завдяки AI. URL: https://designtalk.club/gfycat-vidkryvaye-novi-obiriyi-yakosti-gif-zavdyaky-ai/ (дата звернення 12.10.2018).
35) Google навчив AI розпізнавати якість фото та їхню «естетичну привабливість». URL: https://designtalk.club/google-navchyv-ai-rozpiznavaty-yakist-foto-i-navit-yihnyu-estetychnu-pryvablyvist/ (дата звернення 12.10.2018).

36) HTML Colors. URL: http://www.w3schools.com/ html/html_colors.asp (accessed 14.10.2018).
37) HTML Event Attributes. URL: http://www.w3schools.com/tags/ ref_eventattributes.asp (accessed 12.07.2018).
38) HTML Tutorial. URL: http://www.w3schools.com/html/default.asp (accessed 14.10.2018).
39) HTML справочник. URL: http://html.manual.ru (дата обращения 14.10.2018).
40) HTML: Вікіпідручник. URL: http://uk.wikipedia.org/wiki/HTML (дата звернення 14.10.2018).
41) HTML: Семантическая верстка. URL: https://puzzleweb.ru/ html/16_vidi_verstki3.php (дата звернення 26.07.2018).
42) Современный учебник CSS. URL: https://idg.net.ua/blog/uchebnik-css/razmetka-css/shirina-web-stranitsy (дата обращения 27.07.2018).
43) Marcotte E. Responsive Web design. URL: http://www.alistapart.com/ articles/responsive-web-design (accessed 26.07.2018).
44) Official Google Webmaster Central Blog: Rolling out the mobile-friendly update. URL: https://webmasters.googleblog.com/2015/04/rolling-out-mobile-friendly-update.html (accessed 26.07.2018).
45) Responsive или Adaptive Web Design. URL: https://raskrutka.com.ua/blog/ responsive-ili-adaptive-web-design/ (accessed 27.07.2018).
46) SEO просування, розкрутка та оптимізація сайту в Україні. URL: http://seoweb.in.ua/services (дата звернення 29.09.2018).
47) Siegel D. Creating Killer Sites Hayden Books, 1996. 270 p.
48) The table element. *HTML5: Edition for Web Authors*. URL: https://www.w3.org/TR/2011/WD-html5-author-20110809/the-table-element.html (accessed 25.09.2018).
49) Top 10 Current Web Design Trends for 2018. URL: https://blog.brightscout.com/ top-10-current-web-design-trends-2018 (accessed 14.09.2018).
50) Using Flexbox: Mixing Old and New for the Best Browser Support. URL: https://css-tricks.com/using-flexbox (accessed 26.07.2018).
51) WhatFont. URL: https://chrome.google.com/webstore/detail/whatfont/ jabopobgcpjmedljpbcaablpmlmfcogm (accessed 14.09.2018).
52) Why Flexboxes Aren't Good for Page Layout. URL: https://www.xanthir.com/blog/b4580 (accessed 26.07.2018).
53) Адаптивный и отзывчивый веб-дизайн. URL: https://itkeys.org/responsive-and-adaptive-design/ (дата обращения 27.07.2018).
54) Адміністрування сайту. URL: http://webstudio2u.net/ua/studio-web/673-administrirovanie-saita.html (дата звернення 28.08.2018).
55) Анализ веб-дизайна. URL: http://www.antula.ru/web-design_analysis.htm (дата обращения 18.09.2018).

56) Аналіз роботи сайту. URL: http://webstudio2u.net/ua/design-web/713-analiz-raboty-saita.html (дата звернення 29.09.2018).
57) Асимметрия в дизайне сайта. URL: http://seo-design.net/design/asymmetry-in-design (дата обращения 28.06.2018).
58) Атрибут behavior. URL: http://htmlbook.ru/html/marquee/behavior (дата звернення 28.09.2018).
59) Блоґ. URL: https://uk.wikipedia.org/wiki/Блоґ (дата звернення 28.06.2018).
60) Блочная вёрстка. URL: http://htmlbook.ru/samlayout/blochnaya-verstka (дата обращения 26.07.2018).
61) Веб-дизайн сайту і користувацькі помилки. URL: http://webstudio2u.net/ua/design-web/852-veb-dizain-saita-i-polzovatelskie-oshibki.html (дата звернення 28.06.2018).
62) Веб-дизайн. URL: http://іт.словник.укр/index.php/Веб-дизайн# (дата звернення 17.09.2018).
63) Вёрстка веб-страниц. URL: http://templates.hol.es/ (дата обращения 26.07.2018).
64) Вёрстка веб-страниц. URL: https://seo-wikipedia.org/verstka-web-stranic/ (дата обращения 30.07.2018).
65) Вёрстка с помощью таблиц. URL: http://htmlbook.ru/samlayout/verstka-s-pomoshchyu-tablits (дата обращения 26.07.2018).
66) Використання CSS flexible-боксів. URL: https://developer.mozilla.org/uk/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes (дата звернення 26.07.2018).
67) Використання Flexbox в CSS3 для адаптивного дизайну. URL: https://sebweo.com/vikoristannya-flexbox-v-css3-dlya-adaptivnogo-dizajnu (дата звернення 26.07.2018).
68) Воропай А. Создание Web-сайта на базе WordPress CMS. URL: https://www.ibm.com/developerworks/ru/library/os-wordpress/index.html (дата обращения 29.09.2018).
69) Вотролл Э., Сьярто Дж. Изучаем веб-дизайн: учебн. пособие; [пер. с англ.]. М.: Эксмо 2010. 496 с.
70) Вуль В. А. Электронные издания: учебник. URL: http://www.hi-edu.ru/e-books/xbook119/01/title.htm (дата обращения 12.10.2018).
71) Выбираем HTML редактор – путеводитель для новичков. URL: http://www.internet-technologies.ru/articles/vybiraem-html-redaktor-putevoditel-dlya-novichkov.html (дата обращения 12.07.2018).
72) Електронні видання: довідник. Уклад. Т. Ю. Киричок. К.: НТУУ «КПІ», 2011. 400 с.
73) Етапи розробки сайту. URL: https://websait.uz.ua/website-development/etapy-rozrobky-sajtu/ (дата звернення 28.08.2018).
74) Етапи створення веб-сайтів. URL: http://alextexnok.blogspot.com/p/blog-page_85.html (дата звернення 28.08.2018).

75) Етапи створення веб-сайтів. URL: http://edufuture.biz/index.php?title=Етапи_створення_веб-сайтів (дата звернення 28.08.2018).
76) Етапи створення веб-сторінок. URL: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2013/127/127.html (дата звернення 25.08.2018).
77) Інформаційне наповнення сайту. URL: https://uk.wikipedia.org/wiki/Інформаційне_наповнення_сайту (дата звернення 27.08.2018).
78) Квинт И. Создаем сайты с помощью HTML, XHTML и CSS на 100%; [3-е изд.] СПб.: Питер, 2014. 448 с.
79) Коди спеціальних символів для використання в HTML. URL: http://vvz.nw.ru/Lessons/SymbolCodes/ symbolcodes.htm?n=1 (дата звернення 12.10.2018).
80) Макнейл П. Веб-дизайн. Идеи. Секреты. Советы. Питер, 2000. 272 с.
81) Маркотт И. Отзывчивый веб-дизайн [пер. с англ. П. Миронова]. М.: Манн, Иванов и Фербер, 2012. 163 с.
82) Маслов И. Как создать блог на WordPress? Пошаговая инструкция от вебмастера! URL: http://ivan-maslov.ru/kak-sdelat-sajt/kak-sozdat-blog-na-wordpress.html (дата обращения 12.07.2018).
83) Мержевич В. Основы верстки. URL: http://htmlbook.ru/content/osnovy-verstki (дата звернення 26.07.2018).
84) Мержевич В. Самоучитель по HTML. URL: http://htmlbook.ru/samhtml (дата обращения 26.07.2018).
85) Огірко І., Паславська І., Пілат О. Інформаційна система оцінювання якості електронних видань. Вісник Львівського університету. Серія економічна. 2013. Вип. 49. С. 391–398.
86) Огурцов В.В., Гриньов Д.В., Щербаков О.В. Основи веб та веб-дизайн, програмування на боці клієнта: лабораторний практикум з навчальної дисципліни "Веб-технології та веб-дизайн" для студентів напряму підготовки 6.050101 "Комп'ютерні науки". Харків: ХНЕУ ім. С. Кузнеця, 2015. 208 с.
87) Оптимізація сайтів. URL: http://webstudio2u.net/ua/optimization/97-seo.html (дата звернення 29.09.2018).
88) Оптимізація сайтів: як покращити готовий сайт? URL: http://webstudio2u.net/ua/optimization/392-site-optimization.html (дата звернення 29.09.2018).
89) Основные принципы веб-дизайна и их характеристики. URL: http://www.designonstop.com/webdesign/article/ osnovnye-principy-veb-dizajna-i-ix-xarakteristiki.htm (дата обращения 18.09.2018).
90) Основные принципы веб-дизайна. URL: https://webformyself.com/osnovnye-principy-veb-dizajna (дата обращения 18.09.2018).
91) Основные элементы веб-дизайна. Как благодаря им создать эффективный сайт. URL: https://webformyself.com/osnovnye-elementy-veb-dizajna-kak-blagodarya-im-sozdat-effektivnyj-sajt (дата обращения 18.09.2018).

92) Пасічник О.Г., Пасічник О.В., Стеценко І.В. Основи веб-дизайну: навч. посібник. К.: Вид. група BHV, 2009. 336 с.
93) Перекладаємо слово респонсивний. URL: https://slovotvir.org.ua/words/responsyvnyi (дата звернення 26.07.2018).
94) Плавная трансформация | CSS свойство transition. URL: http://shpargalkablog.ru/2011/07/transformaciya-css.html (дата обращения 20.09.2018).
95) Попков С. Веб-дизайн в прошлом. Знакомьтесь, Zero UI. URL: http://blog.aic.ru/zero-ui (дата обращения 20.09.2018).
96) Практические задания по теме «Web-дизайн и программирование. URL: http://www.modern-computer.ru/practice/web-design/prcatic-web-design-main.html (дата обращения 12.10.2018).
97) Пушкар О.І., Климнюк В.Є., Браткевич В.В. Мультимедійні видання: навч. посібник. Харків: Вид. ХНЕУ, 2012. 144 с.
98) Разница между адаптивным (Adaptive) и отзывчивым (Responsive) дизайном. URL: https://www.templatemonster.com/help/ru/difference-between-adaptive-design-and-responsive-design.html (дата обращения 27.07.2018).
99) Респонсивный или адаптивный: какой дизайн выбрать? URL: https://umbrellait.com/ru/responsive-vs-adaptive-web-design (дата обращения 27.07.2018).
100) Розкрутка сайтів та їх просування. URL: http://webstudio2u.net/ua/promotion.html (дата звернення 28.08.2018).
101) Розкрутка сайту. Google Analytics. URL: http://webstudio2u.net/ua/web-promotion/246-google-analytics.html (дата звернення 29.09.2018).
102) Розкрутка сайту. Веб-аналітика. URL: http://webstudio2u.net/ua/web-promotion/245-web-analytics.html (дата звернення 29.09.2018).
103) Сабін-Вільсон Л. WordPress. Web design. John Wiley & Sons Canada, 2011. 384 с.
104) Сидерхолм Д. CSS3 для веб-дизайнеров [пер. с англ. Е. Кудашева]. М.: Манн, Иванов и Фербер, 2013. 144 с.
105) Сидерхолм Д. Пуленепробиваемый веб-дизайн. Библиотека специалиста; 3-е изд. СПб.: Питер, 2012. 304 с.
106) Создание Web-сайта на базе WordPress CMS. URL: http://www.ibm.com/developerworks/ru/library/os-wordpress/index.html (дата обращения 12.10.2018).
107) Создание разметки: основные правила. URL: https://idg.net.ua/blog/uchebnik-css/razmetka-css/osnovnye-pravila (дата обращения 30.07.2018).
108) Справочник по HTML. URL: http://htmlbook.ru/html (дата обращения 12.07.2018).
109) Справочники по HTML и CSS. URL: https://webref.ru/ref (дата обращения 12.10.2018).

110) Стили и элементы современного веб-дизайна. URL: http://seo-design.net/trendy-web/modern-web-design-trendy-styles-elements (дата обращения 28.06.2018).
111) Таблицы в HTML. URL: https://webref.ru/course/html-content/tables (дата обращения 25.09.2018).
112) Тенденції веб-дизайну в 2018 році. URL: http://webstudio2u.net/ua/design-web/962-tendentsii-veb-dizaina-2018.html (дата звернення 28.06.2018).
113) Типовые макеты. URL: http://htmlbook.ru/samlayout/tipovye-makety (дата обращения 27.07.2018).
114) Типографіка є основою графічного дизайну і полягає в графічному оформленні текстової інформації.
115) У Photoshop CC з'явиться вбивча фіча: виділення особи одним кліком. URL: https://designtalk.club/u-photoshop-z-yavytsya-vbyvcha-ficha-vydilennya-osoby-odnym-klikom/ (дата звернення 12.10.2018).
116) Уолтер А. Эмоциональный веб-дизайн. [пер. с англ. П. Миронова]. М.: Манн, Иванов и Фербер, 2012. 144 с.
117) Уроки по HTML и CSS. URL: https://webref.ru/layout/learn-html-css (дата обращения 12.10.2018).
118) Учебник CSS. URL: http://ru.html.net/tutorials/css (дата обращения 12.10.2018).
119) Учебник по HTML. URL: http://ru.html.net/tutorials/html (дата обращения 12.10.2018).
120) Хоган Б., Уоррен К., Уэбер М., Джонсон К., Годин А. Книга веб-программиста: секреты профессиональной разработки веб-сайтов. СПб.: Питер, 2013. 288 с.
121) Чи може штучний інтелект відібрати роботу у дизайнерів? URL: https://designtalk.club/chy-mozhe-shtuchnyj-intelekt-vidibraty-robotu-u-dyzajneriv/ (дата звернення 12.10.2018).
122) Що таке Flexbox? Опис всіх css властивостей, основні принципи, переваги та недоліки. URL: http://html5.by/blog/flexbox/ (дата звернення 26.07.2018).
123) Що таке SEO аудит і навіщо він потрібен. URL: http://webstudio2u.net/ua/optimization/689-seo-audit.html (дата звернення 29.09.2018).
124) Що таке Zero UI, та до чого рухається майбутнє дизайну. URL: https://designtalk.club/shho-take-zero-ui-do-chogo-ruhayetsya-majbutnye-dyzajnu (дата звернення 21.09.2018).
125) Що таке юзабіліті сайту і чому це так важливо? URL: http://webstudio2u.net/ua/faq/50-design/93-useability.html (дата звернення 20.09.2018).
126) Що таке юзабіліті тестування (юзабіліті аудит). URL: http://webstudio2u.net/ua/design-web/655-usabiliti-testirovanie.html (дата звернення 29.09.2018).
127) Як правильно підібрати домен. URL: https://freehost.com.ua/ukr/domain/domainsel (дата звернення 27.09.2018).
128) Як розпізнати шрифт: підбірка корисних додатків. URL: https://designtalk.club/yak-rozpiznaty-shryft-pidbirka-korysnyh-dodatkiv/ (дата звернення 12.10.2018).

Навчальне видання

ТРОФИМЕНКО Олена Григорівна
КОЗІН Олександр Борисович
ЗАДЕРЕЙКО Олександр Владиславович
ПЛАЧІНДА Ольга Євгеніївна

Веб-технології та веб-дизайн

Навчальний посібник

Підписано до друку 03.05.2019.
Формат 60х84/16. Ум.-друк. арк. 16,3.
Зам. № 1905-05. Наклад 100 прим.

Видавець ПП «Фенікс»
(Свідоцтво суб'єкта видавничої справи ДК № 1044 від 17.09.2002)
65059, м. Одеса, а/я 424, вул. Зоопаркова, 25
тел. +38 048 7959160, +38 050 7775901
e-mail: fenix-izd@ukr.net
www.feniksbooks.com