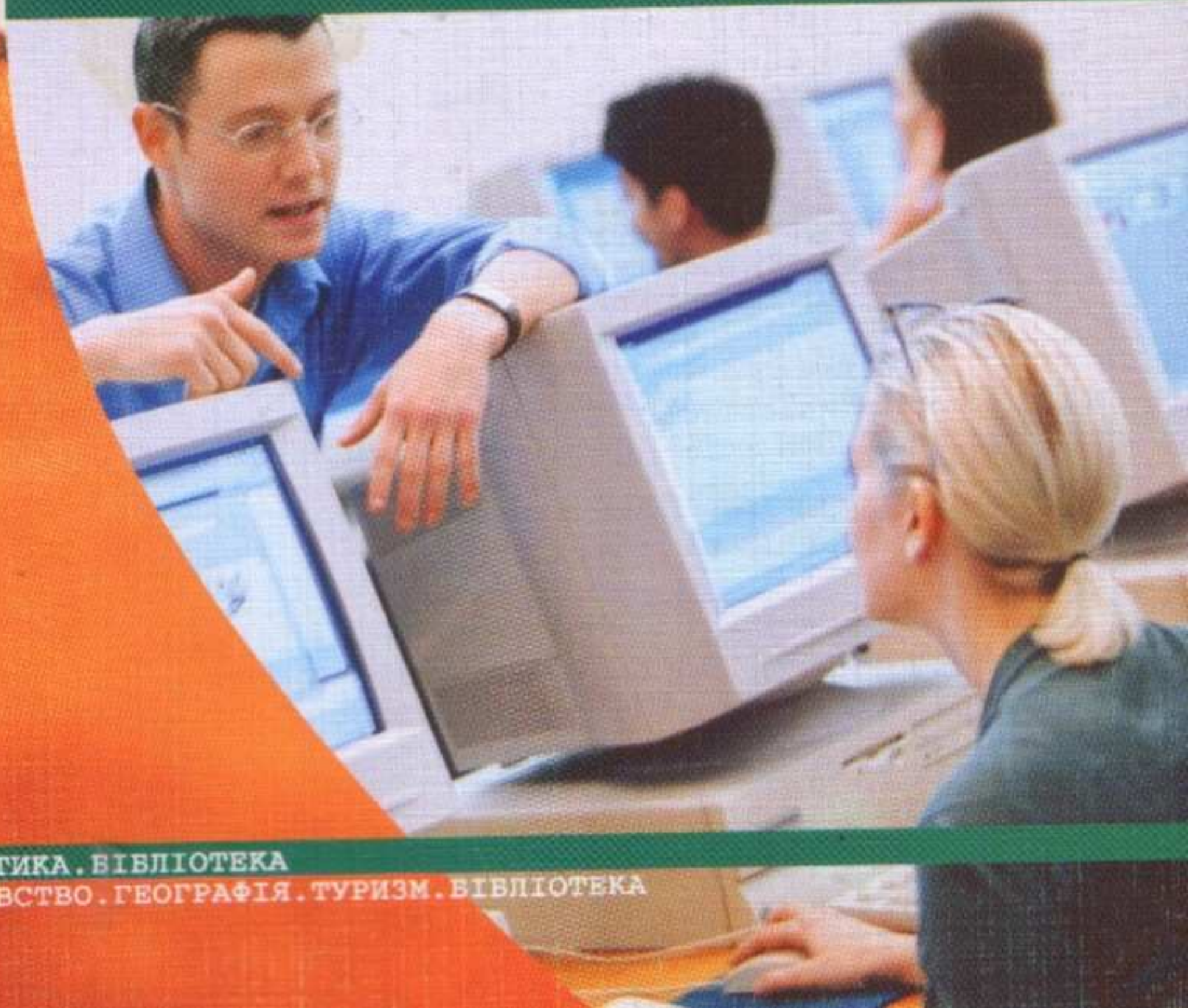
ШКІЛЬНИЙ СВІТ

НАМ 10 РОКІВ

Шкільна бібліотека Бібліотека інформатики

БІБЛІОТЕКА «ШКІЛЬНОГО СВІТУ»

ОСНОВИ БУДУВАННЯ САЙТІВ



ІНФОРМАТИКА. БІБЛІОТЕКА
КРАЄЗНАВСТВО. ГЕОГРАФІЯ. ТУРИЗМ. БІБЛІОТЕКА

Бібліотека «Шкільного світу»
Заснована у 2003 р.

ШКІЛЬНИЙ СВІТ

Володимир Манако
Дмитро Манако
Ольга Данилова
Олексій Войченко

ОСНОВИ БУДУВАННЯ САЙТІВ

Інформатика. Бібліотека
Краєзнавство. Географія.
Туризм. Бібліотека

Київ
Видавничий дім «Шкільний світ»
Видавець Л.Галіцина
2006

ББК 32.973.202
О-75

Основи будування сайтів / В.Манако, Д.Манако, О.Данилова,
О-75 О.Войченко. — К.: Вид. дім «Шкіл. світ»: Вид. Л.Галіцина, 2006. —
120 с., [4] арк. — (Б-ка «Шкіл. світу»).
ISBN 966-8651-59-6.
ISBN 966-420-021-2.
ISBN 966-356-059-2.

Сучасна мережа Інтернет містить понад 8 мільярдів сторінок з будь-якої галузі людської діяльності. Багато сторінок знаходяться у локальних мережах. Отже, розповісти світові про себе, свою діяльність та захоплення можна за допомогою веб-сторінок у Мережі. У цій книжці йдеться про те, як самостійно створити веб-сторінку. Матеріал подано у вигляді практичних уроків.

Рекомендовано широкому загалу читачів з елементарними навичками користування комп'ютером.

ББК 32.973.202

ISBN 966-8651-59-6 (б-ка «Шкіл. світу»)
ISBN 966-420-021-2 (Вид. дім «Шкіл. світ»)
ISBN 966-356-059-2 (вид. Л.Галіцина)

ЗМІСТ

Вступ 4
Розділ 1. Основи проектування веб-сайту
Мета створення сайту 5
Організація головної сторінки 12
Створення власного стилю сайту 16
Створення веб-сторінок для моніторів з різною роздільною здатністю 25
Розділ 2. Практичний web-дизайн
Редактори для верстки веб-сторінок 27
Верстка сайту 29
Оптимізація зображень у форматі GIF і JPEG 31
Розділ 3. Створюємо сайт
Гіпертекст 35
World Wide Web 36
HyperText Markup Language (HTML) 40
Програми перегляду інформації в Інтернеті (броузери) 41
Огляд сучасних HTML-редакторів 43
Створення HTML-сторінки 48
Створення найпростішого HTML-документа 49
Модифікація HOME-сторінки 52
Робота з текстом 53
Створення фреймів 59
Розділ 4. Графіка і звук у документах HTML
Вставка графічних зображень. 69
Вставка звуку в документ HTML. 73
Розділ 5. Використання JavaScript на практиці
JavaScript: необхідно знати 75
Вміщення JavaScript в HTML-сторінки 76
Основні поняття JavaScript 78
Огляд вбудованих об'єктів JavaScript 86
DHTML і об'єктна модель Internet Explorer 90
DHTML і JavaScript (7 практичних прикладів) 91
Корисні посилання 94
Розділ 6. Каскадні Стилі CSS
Стилі: інструмент для сучасних веб-дизайнерів 97
Імпортування стилів 100
Розділ 7. Робота із сайтом
Як організувати пошук на сайті 106
Як організувати «розкрутку» сайту 109
Оптимізація сайту 109
Оптимізація контенту сайту 111
Оптимізація для пошукової системи Yandex 111
Оптимізація для пошукової системи Rambler 111
Технологія пошуку Google 112
Зміст сайту 112
Корисні посилання для викладачів шкіл 113

ВСТУП

Сайт** — це місце в Інтернеті, що визначається своєю адресою (URL), має свого власника й складається з веб-сторінок, які сприймаються як єдине ціле.* Чіткого визначення поняття сайту не існує — наприклад, деякі розділи великих сайтів цілком можуть сприйматися й навіть визначатися їхніми власниками як окремі сайти. Стартову сторінку, що з'являється при звертанні до доменного імені сайту, часто називають ***головною (або індексною) сторінкою сайту.

***Веб-сторінка** — це логічна одиниця Інтернету, однозначно обумовлена адресою.* Можна сказати, що Інтернет складається із сайтів, а сайти, у свою чергу, — зі сторінок. URL — це адреса сторінки в Інтернеті. URL складається з доменного імені, шляхів до сторінки на сайті та назви файла сторінки. Наприклад: *www.dlab.kiev.ua/index.html/*. Тут *www.dlab.kiev.ua* — доменне ім'я сайту, *index.html* — ім'я файла. Як правило, файли, що містять веб-сторінки, мають розширення *.htm* або *.html*. Коли говорять «адреса сайту», мають на увазі його доменне ім'я, при звертанні до якого завантажується стартова сторінка сайту.

За міжнародною згодою кожній країні в Інтернеті зарезервоване деяке кодове позначення довжиною у 2—3 літери, що називається ***доменом першого рівня*** або доменом цієї країни. Так, наприклад, якщо адреса сайту закінчується на *.ua* — сайт знаходиться в домені України, *.fr* — Франції, *.de* — Німеччини. До того ж, існують деяки домени першого рівня, пов'язані не з географією, а зі спрямованістю сайту — наприклад, *.com* для комерційних, *.org* для некомерційних, *.edu* для освітніх організацій.

розділ **1**

ОСНОВИ ПРОЕКТУВАННЯ ВЕБ-САЙТУ

МЕТА СТВОРЕННЯ САЙТУ

Перш ніж створювати сайт слід з'ясувати такі питання: цільова аудиторія, концепція проекту, тип і топологія сайту та скласти план інформаційної й функціональної структури.

Проектування сайту зазвичай передбачає такі кроки:

Крок 1. Визначити мету створення сайту, його цільову аудиторію й тип. Для сайтів у сфері освіти цільова аудиторія, як правило, складається зі школярів, студентів, учителів, викладачів, батьків, дирекції.

Відповідно, освітні сайти можна класифікувати в такий спосіб:

- домашня сторінка;
- сайт школи;
- освітній портал.

Як правило, якісний шкільний сайт:

- містить довідкову інформацію, що цікавить батьків, дитина яких вступає до школи (зокрема, про вчителів, навчальні програми, традиції школи тощо);
- відображає події, що відбуваються в школі (свята, конференції, конкурси олімпіади);
- відображає постійно діючі напрями в роботі школи (шкільний музей, участь у проектах);
- є місцем, де учні можуть подати свої творчі роботи (зокрема, гумористичні);

- надає можливість учителям розмістити свої матеріали (аж до окремого розділу з предмета);
- містить елементи дистанційної підтримки навчання (наприклад, віртуальний консультаційний пункт);
- підтримує особисті сторінки учнів, учителів, цілих класів;
- містить спеціальний розділ для випускників;
- представляє освітню установу міжнародному загалу.

Крок 2. Визначаємо інформаційну структуру сайту. ***Інформаційна структура*** — це розподіл інформації, яку планується розмістити на веб-сайті, на розділи, підрозділи й сторінки, а також розміщення інформації на сторінці, зокрема з визначенням форматів (рис. 1.1).

Структура шкільного сайту може бути такою:

- **наші учні** — шкільний колектив (кількість учнів, класів); успішність, захоплення, список медалістів; поточні оцінки учнів (семестрові, підсумкові);
- **наші вчителі** — відомості про вчительський колектив (звання, категорії, стаж); фотографії, публікації матеріалів учителів;
- **адміністрація** — П.І.Б. директора школи, заступників, обов'язки кожного;
- **новини школи** — репортажі зі шкільних свят; цікаві шкільні події й заходи; прес-конференції (з колишніми учнями, керівництвом, гостями); спортивні рекорди школярів; найкращі творчі роботи школярів; досягнення учнів (олімпіади, конкурси, змагання);
- **інформація про школу** — адреса школи, факс, адреса електронної пошти, наявні телефони; устав, гімн, герб, емблема, девіз, прапор школи; права й обов'язки учнів; історія і традиції школи; освітня політика; наявність комп'ютерних класів; технічна оснащеність школи (лабораторії, майстерні, кабінети); фотографії (школи, кабінетів, залів, майстерень); умови прийому до школи, межі району, що обслуговується; фрагмент карти міста або схема розташування школи, прив'язки;
- **гуртки й захоплення** — можливості одержання додаткових освітніх послуг (через гуртки, клуби, факультативи, спортивні секції); постійно діючі напрямки в роботі школи (шкільний музей, участь у проектах, спортивні команди, студії);
- **бібліотека** — список книжок у шкільній бібліотеці (база даних, пошук книжок);
- **фотоальбом** — електронний альбом з фотографіями учнів, учителів, сцен зі шкільного життя.

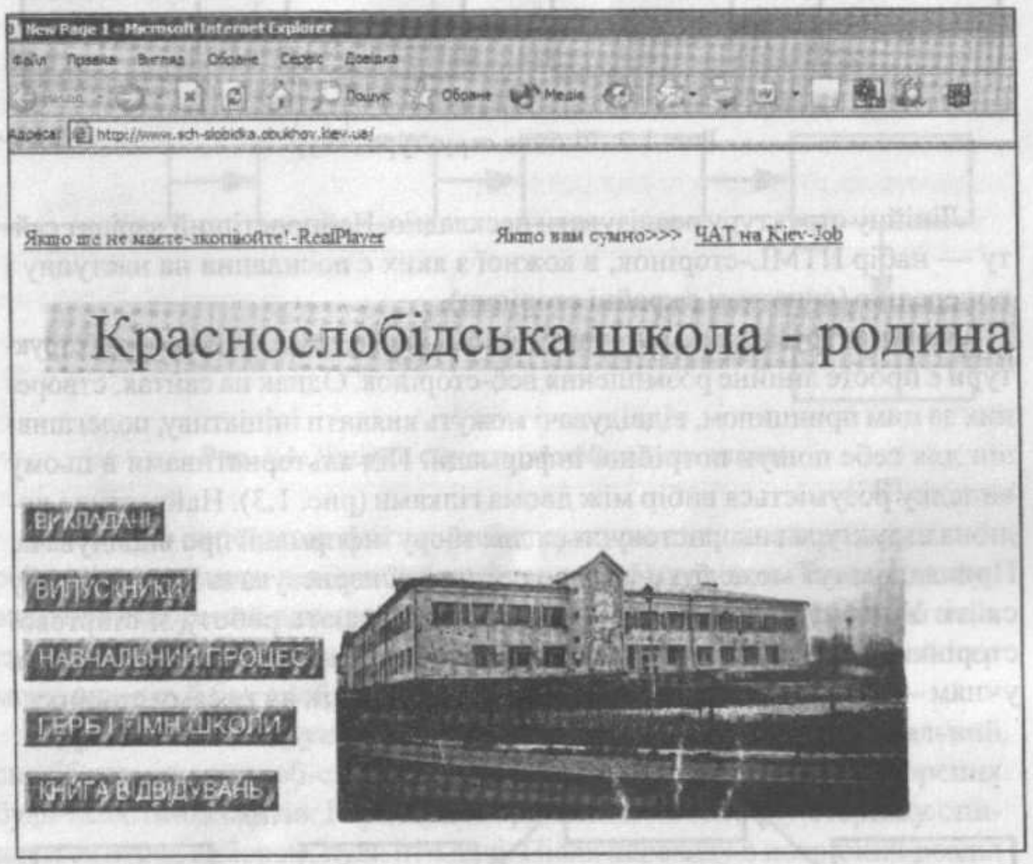


Рис. 1.1. Приклад веб-сторінки шкільного сайту

Крок 3. Визначаємо топологію сайту. Виділяють п'ять основних структур сайту — лінійна, лінійна з альтернативними варіантами, лінійна структура з відгалуженнями, деревоподібна структура, ґратчаста структура.

Лінійна структура — *це найпростіша структура сайту*. Веб-сторінки йдуть одна за одною, і користувач має переглядати їх як слайд-шоу (рис. 1.2 на с. 8). У лінійній структурі не існує поділу контенту на рівні. Всі сторінки на таких сайтах рівноправні, і їх має побачити кожний відвідувач. Незважаючи на простоту реалізації лінійної структури, недоліків у неї набагато більше, ніж переваг. А тому сфера її застосування чітко обмежена: вона може використовуватися на сайтах-презентаціях і в он-лайнових навчальних посібниках.

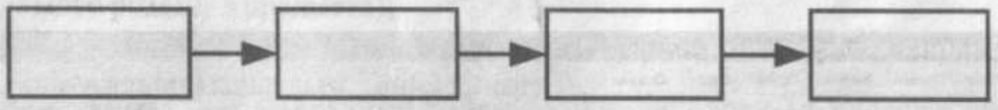


Рис. 1.2. Лінійна структура сайту

Лінійну структуру реалізувати нескладно. Найпростіший варіант сайту — набір HTML-сторінок, в кожної з яких є посилання на наступну і попередню (винятком є крайні сторінки).

Лінійна структура з альтернативами й варіантами. Основою цієї структури є просте лінійне розміщення веб-сторінок. Однак на сайтах, створених за цим принципом, відвідувачі можуть виявити ініціативу, полегшивши для себе пошук потрібної інформації. Під альтернативами в цьому випадку розуміється вибір між двома гілками (рис. 1.3). Найчастіше подібна структура використовується для збору інформації про відвідувача. Прикладом тут може бути процес реєстрації користувача на шкільному сайті. У цьому випадку всі користувачі починають роботу зі стартової сторінки. Однак потім учителям пропонується ввести одну інформацію, а учням — іншу. Після цього й ті, і інші потрапляють на ту саму сторінку.

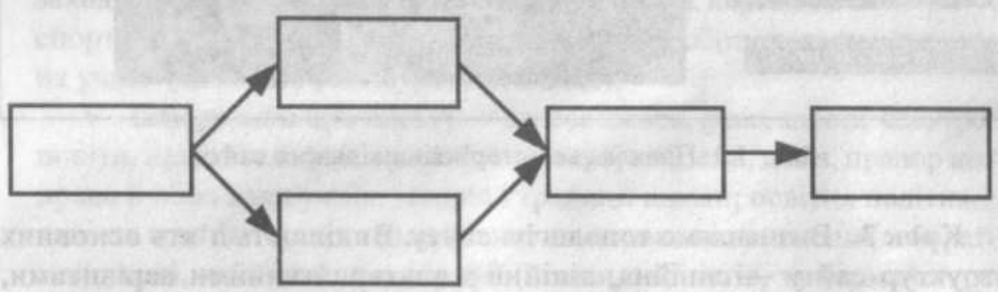


Рис. 1.3. Лінійна структура з альтернативними варіантами

Лінійна структура з відгалуженнями. Це теж контрольована структура, що нагадує дорогу з тупиковими стежками, що відгалужуються від неї час від часу (рис. 1.4). У такий спосіб відвідувач послідовно переходить з однієї сторінки на іншу. Якщо інформація, розміщена на деякій зі сторінок, його зацікавила і він хоче довідатися подробиці, то може перейти на відгалуження, а потім повернутися назад на основну «доріжку».

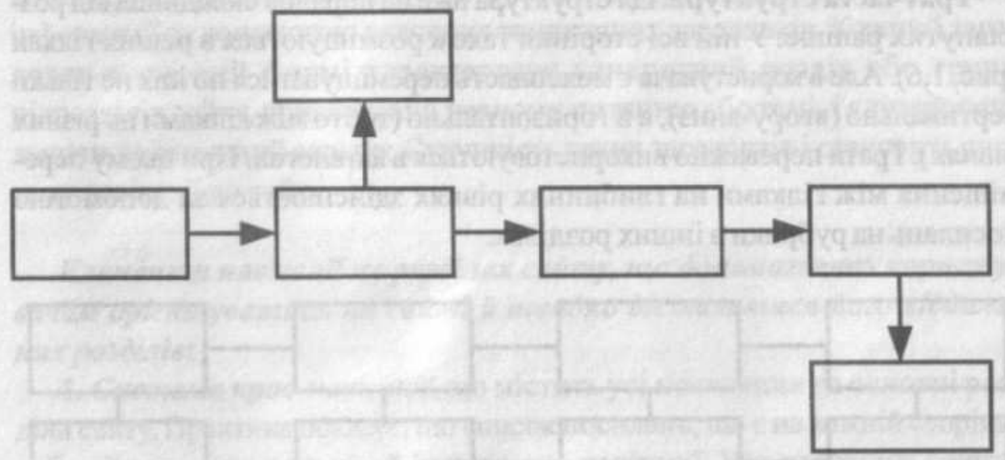


Рис. 1.4. Лінійна структура з відгалуженнями

Головною перевагою розглянутої структури є те, що до неї легко перейти зі звичайного лінійного розміщення веб-сторінок. Таке часто буває, коли створений один раз веб-сайт перестає задовольняти вимогам, а глобальна переробка сайту неможлива. У цьому випадку веб-майстер може швидко й без проблем розширити проект.

Деревоподібна структура. Деревоподібна структура — універсальний спосіб розміщення веб-сторінок (рис 1.5). Вона підходить для створення будь-яких типів сайтів. Користувач при заході на головну сторінку опиняється перед вибором, куди йти далі. Після переходу в потрібний розділ він добирає необхідний підрозділ.

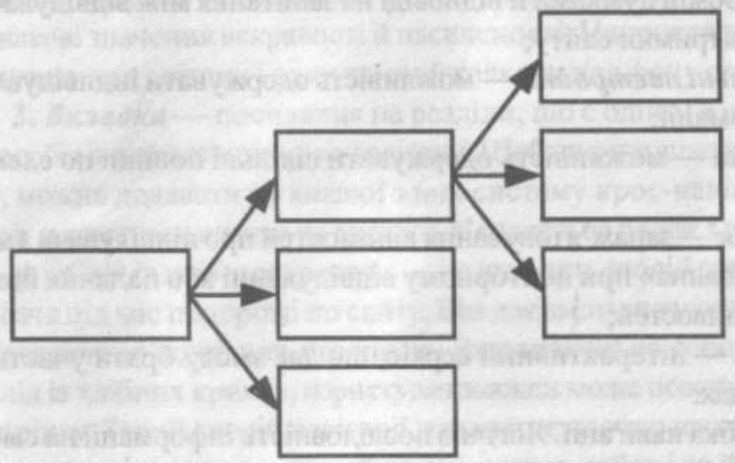


Рис. 1.5. Деревоподібна структура

Ґратчаста структура. Ця структура вже на порядок складніша від розглянутих раніше. У ній всі сторінки також розміщуються в різних гілках (рис. 1.6). Але в користувача є можливість переміщуватися по них не тільки вертикально (вгору-вниз), а й горизонтально (тобто між гілками на різних рівнях). Ґрати переважно використовуються в каталогах. При цьому переміщення між гілками на глибинних рівнях здійснюється за допомогою посилань на рубрики в інших розділах.

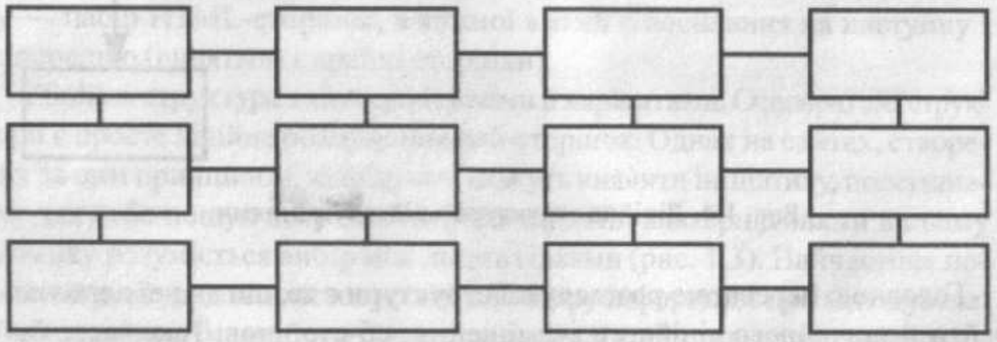


Рис. 1.6. Ґратчаста структура

Крок 4. Визначаємо функціональну структуру. *Функціональна структура* демонструє, які можливості обробки інформації й інтерактивної взаємодії ви надаєте відвідувачеві.

Приклад функціональної структури типового шкільного сайту:

- ***віртуальний хол*** для спілкування вчителів, учнів, батьків і вчителів;
- ***форум*** — обмін думками й відповіді на запитання між відвідувачами й службою підтримки сайту;
- ***індивідуальні настройки*** — можливість одержувати індивідуальне подання інформації;
- ***розсилання*** — можливість одержувати шкільні новини по електронній пошті;
- ***реєстрація*** — запам'ятовування відомостей про відвідувача з метою його «впізнавання» при повторному відвідуванні або надання йому спеціальних можливостей;
- ***олімпіади*** — інтерактивний сервіс, що дає змогу брати участь у шкільних олімпіадах.

Крок 5. Розробка навігації. Логічна послідовність інформації на сайті досягається через влаштування системи навігації по сайту. Наявність на-

вігаційної системи дає змогу відвідувачеві візуально визначати цінність інформації за допомогою вивчення тематичних заголовків. Кожний заголовок у стислій формі характеризує конкретний розділ або групу підрозділів сайту, присвячених певному питанню або темі, і є гіперпосиланням на їх повний варіант. Сукупність таких заголовків і становить систему навігації по сайту.

Елементи навігації по розділах сайту, що допомагають користувачам орієнтуватися на сайті й швидко діставатися його віддалених розділів:

1. ***Система крос-навігації***, що містить усі посилання на основні розділи сайту. Практика показує, що список посилань, що є на кожній сторінці веб-сайту, — дуже зручний інструмент навігації. Він дає змогу бачити схематичну карту сайту на кожній сторінці. За допомогою системи крос-навігації можна не тільки легко пересуватися по розділах, а й одержати інформацію про розмір сайту, а також визначити місцезнаходження користувача в цей момент. І якщо виділяти назву розділу, в якому знаходиться користувач, він ніколи не заблукає.

2. ***Колірне позначення розділів***, коли для кожного розділу застосовується свій колір фону й кнопок, що нагадуватиме користувачеві про те, де він зараз знаходиться. Якщо в дизайні вирішено використовувати колірні позначення розділів, необхідно відбити це й у системі крос-навігації. Колір кожного посилання має відповідати кольору розділу. Рекомендується використовувати колірні позначення розділів, якщо на сайті їх не більше п'яти-семи. Але якщо розділів більше, сайт виглядатиме строкато. До того ж, бажано вибрати кольори, які добре сполучаються між собою й мають однакові значення яскравості й насиченості. Наприклад, не слід використовувати три світлих і один темний кольори для фону сторінок.

3. ***Вкладки*** — посилання на розділи, що є одним з найпопулярніших способів використання крос-навігації. Щоб ширше використовувати вкладки, можна додавати до кожної з них систему крос-навігації. Наприклад, коли користувач клацає на вкладці, з'являється рядок з посиланнями.

4. ***«Слід із хлібних крихт»*** — це ще один спосіб зорієнтувати користувача під час подорожі по сайту. Він дає змогу позначати маршрут його пересування по різних сторінках. Залишаючи за собою в такий спосіб «слід із хлібних крихт», користувач завжди може повернутися на головну сторінку. Такий спосіб навігації допомагає наочно продемонструвати користувачеві, як далеко він заблукав у нетрях сайту і як йому звідти вийти.

У розробці сайту можна використовувати такі варіанти навігації:

1. Текстові посилання — найпростіший у плані реалізації варіант інформування користувача про те, що його очікує всередині сайту.

2. Графічний варіант запису системи навігації є, мабуть, найпоширенішим в Інтернеті. Застосовуються фотографічні зображення, малюнки, анімація.

3. HTML-форми. Специфікація мови гіпертекстової розмітки HTML дає змогу розміщувати на веб-сторінках спадаючі й вибіркові меню, які, як правило, дають змогу заощадити місце на сторінці і являють собою інтуїтивно зрозумілі користувачеві елементи робочого середовища Windows.

4. Java-апплети являють собою невеликі програми, які вбудовуються в HTML-код веб-сторінок і можуть містити як текст, так і графіку. Відображаються за наявності в броузері користувача опції підтримки Java.

5. Flash. Системи навігації, розроблені на основі технології Macromedia Flash, можуть сполучати в собі растрову й векторну графіку, анімацію, аудіо й відео, а також реагувати на різні користувацькі маніпуляції: натискання «гарячих» клавіш, переміщення курсору та ін. Для їхнього відображення на веб-сторінці на комп'ютері відвідувача сайту має бути встановлений спеціальний *plug-in — Macromedia Flash Player*.

Кожна сторінка сайту повинна мати зручну й інтуїтивно зрозумілу систему навігації. Важливо, щоб відвідувач відразу зміг зрозуміти, у якому підрозділі якого розділу й на якій сторінці сайту він знаходиться. Існують вертикальний (ліворуч і праворуч) і горизонтальний (зверху та знизу) способи організації навігації (рис. 1.7).

ОРГАНІЗАЦІЯ ГОЛОВНОЇ СТОРІНКИ

Головна сторінка має забезпечувати пряму реалізацію головної мети сайту — бути інформаційно-рекламним і організуючим ресурсом організації. Звідси — вимога постановки короткої стратегічної мети одразу, а також застосування ідеографічних символів і зорових образів, призначених до засвоєння відвідувачем сайту одразу, у готовому вигляді й через найбільш інформативні канали сприйняття — зорові образи, оминаючи критичне осмислення інформації.

Головна сторінка веб-сайту — це:

- ***Обличчя закладу у віртуальному світі.*** Відомо, що перше враження можна скласти лише один раз. Головна сторінка веб-сайту має бути гарною. Але гарною настільки, щоб не закривати своєю красою зміст. Тобто оформлення головної сторінки має бути стильним і непомітним, без колірної й анімаційної вакханалії.

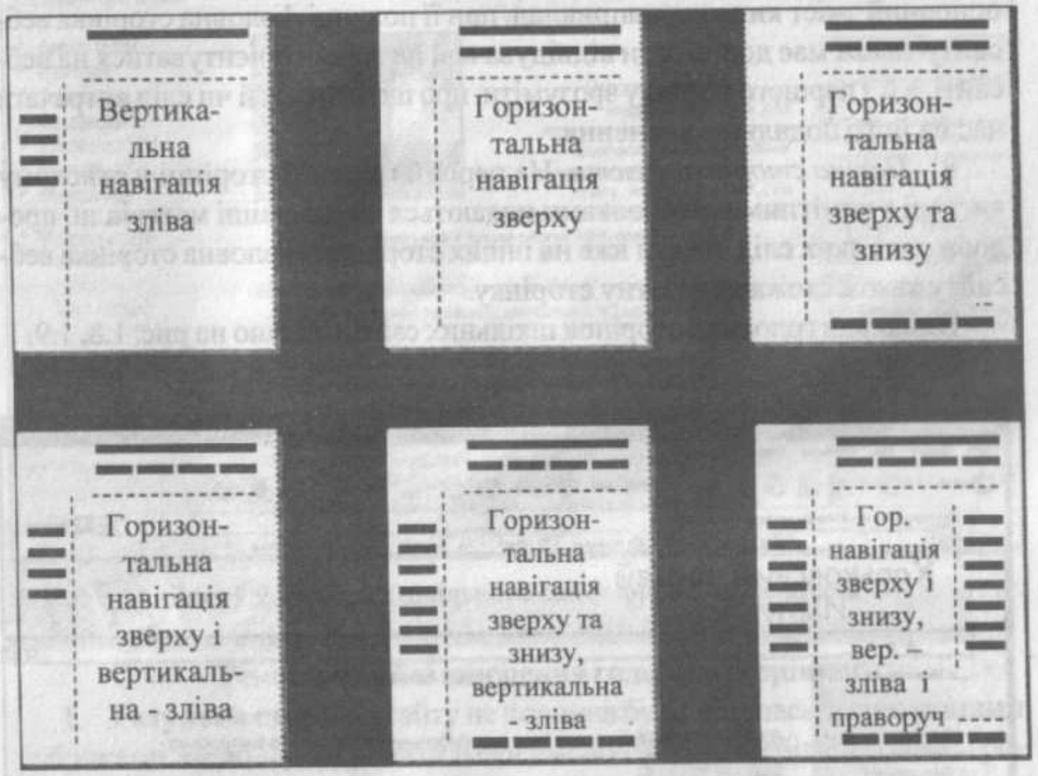


Рис. 1.7. Приклади організації навігації

- ***Фойє будинку.*** Фойє призначене для входу в будинок. Головна сторінка — для входу на веб-сайт. І, як у фойє, на головній сторінці веб-сайту має бути система покажчиків, причому не всіх підряд, а найбільш важливих. Їхнє призначення — зорієнтувати користувача й проінформувати його, що він зайшов саме туди, куди потрібно. Якщо ж таких покажчиків буде занадто багато, користувач заплутається й розгубиться. Тому питанню, проте які саме покажчики-посилання мають бути на головній сторінці веб-сайту, а які можна винести на другорядні сторінки, слід приділити багато уваги. Причому в просторі екрана посилання слід розта-

шувати певним чином, так, щоб у найбільш привабливій його частині були найважливіші й популярні посилання, а інші можна розмістити в інших частинах головної сторінки.

- ***Зміст книги.*** Створення ієрархічної структури змісту — дуже важливе завдання оформлення книги. Правильно складений зміст допомагає читачеві не тільки легко орієнтуватися в книжці, а й швидко «схопити» основний зміст книжки, наприклад, при її покупці. Головна сторінка вебсайту також має допомогти відвідувачеві не тільки орієнтуватися на вебсайті, а й з першого погляду зрозуміти, про що йдеться й чи слід витрачати час на його подальше вивчення.
- *Перша сторінка газети.* На першій газетній сторінці у стислому вигляді з помітними заголовками подаються найцікавіші матеріали, продовження яких слід читати вже на інших сторінках. Головна сторінка вебсайту також схожа на газетну сторінку.

Приклади головних сторінок шкільних сайтів подано на рис. 1.8, 1.9.

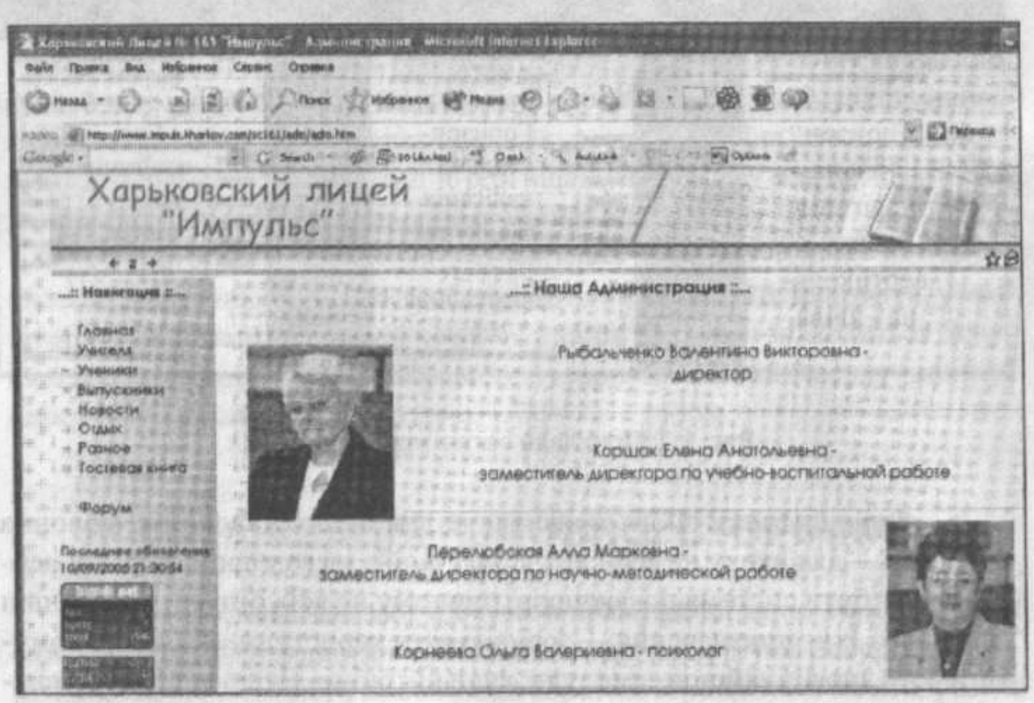


Рис. 1.8. Приклад шкільного сайту

Рис. 1.9. Приклад шкільного сайту фінансового ліцею

Рекомендації зі створення головної сторінки:

1. **Титульна сторінка** сайту не повинна бути перенасиченою різними рубриками. Недоцільно робити її фреймовою й робити більше однієї градації рубрик.
2. **Кількість рубрик** на титульній і кожній з навігаційних сторінок сайту бажано обмежити п'ятьма-шістьма. Принцип рубрикації має бути єдиним і логічним, назви простими й однозначними. Якщо не вдається підібрати простої назви, можна дати як назву коротку анотацію змісту рубрики.
3. **Інформаційна структура** сайту має бути деревом, тобто перехід з одного рівня на інший має бути вертикальним.
4. **Навігаційний апарат** має наочно підкріплювати рівневу структуру на кожній сторінці сайту, щоб легко можна було рухатися вгору-вниз за ієрархічною структурою.
5. **Горизонтальні зв'язки** бажано робити тільки всередині сайту. Якщо є необхідність дати посилання на матеріал, що входить до іншого пакета

веб-сторінок, то рекомендується попередити користувача, що, виходячи за посиланням, він залишає сайт. Це можна зробити, наприклад, записами-нагадуваннями. Також не слід робити посилання на інші сайти прямо в тексті. Краще для цього використовувати посилання внизу сторінки (у крайньому разі дужки) і ввести посилання з попереднім записом *«Більш докладно з матеріалами можна ознайомитися...»*.

6. **Посилання всередині матеріалу** доцільно розміщувати тільки на сторінки наступного вертикального рівня або по горизонталі одного з цією сторінкою рівня.

7. **При виставлянні** великого багатосторінкового матеріалу, наприклад, методичного посібника, на кожній сторінці на видному місці бажано вміщувати основні розділи посібника, причому розділ, з якого подається сторінка, слід виділяти таким чином, щоб він впадав у око.

8. **У дизайні титульної сторінки** й сторінок зі змістом слід вибирати яскраві насичені кольори, колірні гами слід робити різко контрастними, малюнки і банери також допустимі, однак на колірному фоні не повинні губитися позначення рубрик.

СТВОРЕННЯ ВЛАСНОГО СТИЛЮ САЙТУ

Стильний веб-сайт — це коли кожна сторінка сайту має яскраво виражену приналежність до всього веб-сайту. Коли легко орієнтуватися й пошук інформації не пов'язаний з небезпекою заблукати й згаяти час, коли сторінки завантажуються менш ніж 1 хв.

Стиль веб-сайту залежить від:

- ***шрифту*** — у межах публікації він має мати однакові характеристики — гарнітура (накреслення), кегль (висота), колір;
- ***абзаців*** — бажано, щоб переважав якийсь один з видів вирівнювання на сторінці, наприклад, публікація зроблена з відступом від лівого краю й вирівнюванням ліворуч;
- ***колірної схеми веб-сайту*** — вона починається з вибору трьох кольорів сторінки, які використовуються для подання звичайно тексту, посилань і відвіданих посилань. Колірна схема має повторюватися на всіх сторінках публікації, це створить у відвідувача відчуття зв'язності сайту. Колір посилань намагайтеся вибирати таким чином, щоб, з одного боку, читач бачив, що це посилання, а з іншого боку, воно не заважало б йому читати основний текст. Для того, щоб кольори посилань і відвіданих посилань відрізнялися, зробіть колір відвіданих посилань трохи темнішим. Підкреслений текст у Web символізує посилання, тому не використовуйте підкреслений текст у публікації, скористайтеся іншим способом виділення.

Графічне оформлення сайту має відповідати загальній колірній схемі; по-друге, необхідно продумати загальну концепцію графічного оформлення. Всі графічні елементи можна поділити на два класи: мальовані й фотореалістичні. У випадку, якщо ви використовуєте на сайті фотографії як ілюстрації, перед використанням обробіть їх — зробіть, якщо буде потреба, тонову й колірну корекцію, кадрування, виберіть орієнтований розмір фотографій у публікації, знайдіть спосіб обробити краї фотографії. А потім використовуйте оформлення по всій публікації. І завжди пишіть пояснення до фотографій у параметрі ALT-тега IMG — це буде сприйматися як підпис до фотографії (про це йдеться далі).

Всі головні елементи сторінки веб-сайту мають знаходитися ***вище від «лінії згину»*** сторінки, тобто на першому екрані. Якщо ж цього зробити не вдається, слід використовувати інші дизайнерські прийоми, які підказують користувачеві, що нижче на екрані ці елементи є. Не рекомендується залишати занадто багато вільного місця по вертикалі між елементами сторінки веб-сайту, щоб користувач не подумав, що нижче нічого немає.

Навігація по сайту — саме вона не дає відвідувачеві заплутатися в нетрях вашого сайту. Завжди залишайте для відвідувача можливість переходу на головну сторінку публікації.

Головне навігаційне меню слід розташувати в помітному місці сторінки, бажано поруч з її основною частиною. Не рекомендується горизонтальну навігаційну панель сайту розміщувати вище від графічних елементів на зразок банерів. Справа в тому, що користувачі вже не сприймають банери — діє так званий ефект «банерної сліпоти». Тому все, що перебуває вище від елемента банера, може залишитися поза полем зору відвідувача веб-сайту.

1. Як правило, навігаційні елементи мають бути згруповані за якимись ознаками. Тобто неправильно вміщувати в один стовпчик, наприклад, перелік категорій каталогу й посилання на сторінку «Про компанію», «Контакти», «Новини». Але цей поділ має бути логічно обгрунтованим. Інакше замість допомоги відвідувачеві веб-сайту буде зворотний ефект — він заплутається.

2. Не рекомендується на одній сторінці створювати різні частини навігації, які ведуть на ті самі сторінки. Особливо якщо оформлення цих частин відрізняється.

Стилі оформлення типів сторінок. Відповідно до певних типів сторінок сайту потрібно розробити для кожного з них свій стиль оформлення, інакше ці типи просто не відрізнятимуться. Проте в них має бути щось спільне.

Існують такі стилі оформлення сторінок:

- ***текстовий дизайн*** — визначається змістом і концепцією автора. Текстовий дизайн означає майже повну відсутність зображень як таких, що значно прискорює завантаження сторінки, а навігаційні й декоративні елементи виконуються символьними прийомами.Слід добре знати класичні верстальні прийоми й особливості шрифтів;
- ***поліграфічний дизайн*** — веб-сторінка імітує друковане видання (особливо буклетні типи). Такий дизайн розповсюджений на корпоративних сайтах, сайтах з рекламним ухилом і там, де особливо необхідне образно-емоційне наповнення основного змісту. Як правило, основне враження користувачі одержують за рахунок піксельної графіки;
- ***інтерфейсний дизайн*** — зараз його ще називають *usability*. Стиль, покликаний максимально полегшити життя користувачеві у всіх його виявах: від завантаження сторінки (мінімізування коду й граничної оптимізації зображень) до ретельного виконання кожного елемента;
- ***динамічний дизайн*** — у примітивному варіанті рухливі зображення переважно флеш, але сюди ж належать і DHTML, аплети на Java і навіть анімаційний GIF). У добре продуманому сценарії це може бути цілий витвір мистецтва, що послідовно розгортає перед глядачем думку автора або художній образ;
- ***змішані типи*** — комбінація всіх цих типів.

Пропонуємо стилі для сайту:

Назва	Опис сторінки
жовтий + блакитний	Колір тексту: темно-сірий Фон: білий Колонтитули: жовтий фон Таблиці: блакитний фон заголовків Колір рамки: чорний
жовтий + персиковий	Колір тексту: темно-сірий Фон: білий Колонтитули: жовтий фон Таблиці: персиковий фон заголовків Колір рамки: чорний

Продовження

Назва	Опис сторінки
жовтий + аквамарин	Колір тексту: темно-сірий Фон: білий Колонтитули: жовтий фон Таблиці: фон заголовків — аквамарин Колір рамки: чорний
жовтий + зелений	Колір тексту: темно-сірий Фон: білий Колонтитули: жовтий фон Таблиці: зелений фон заголовків Колір рамки: чорний
блакитний + жовтий	Колір тексту: темно-сірий Фон: білий Колонтитули: блакитний фон Таблиці: жовтий фон заголовків Колір рамки: чорний
блакитний + персиковий	Колір тексту: темно-сірий Фон: білий Колонтитули: блакитний фон Таблиці: персиковий фон заголовків Колір рамки: чорний
блакитний + аквамарин	Колір тексту: темно-сірий Фон: білий Колонтитули: блакитний фон Таблиці: фон заголовків — аквамарин Колір рамки: чорний
блакитний + зелений	Колір тексту: темно-сірий Фон: білий Колонтитули: блакитний фон Таблиці: зелений фон заголовків Колір рамки: чорний
персиковий + жовтий	Колір тексту: темно-сірий Фон: білий Колонтитули: персиковий фон Таблиці: жовтий фон заголовків Колір рамки: чорний
персиковий + блакитний	Колір тексту: темно-сірий Фон: білий Колонтитули: персиковий фон Таблиці: блакитний фон заголовків Колір рамки: чорний

Продовження

Назва	Опис сторінки
персиковий + аквамарин	Колір тексту: темно-сірий Фон: білий Колонтитули: персиковий фон Таблиці: фон заголовків — аквамарин Колір рамки: чорний
персиковий + зелений	Колір тексту: темно-сірий Фон: білий Колонтитули: персиковий фон Таблиці: зелений фон заголовків Колір рамки: чорний
аквамарин + жовтий	Колір тексту: темно-сірий Фон: білий Колонтитули: фон — аквамарин Таблиці: жовтий фон заголовків Колір рамки: чорний
аквамарин + блакитний	Колір тексту: темно-сірий Фон: білий Колонтитули: фон — аквамарин Таблиці: блакитний фон заголовків Колір рамки: чорний
аквамарин + персиковий	Колір тексту: темно-сірий Фон: білий Колонтитули: фон — аквамарин Таблиці: персиковий фон заголовків Колір рамки: чорний
аквамарин + зелений	Колір тексту: темно-сірий Фон: білий Колонтитули: фон — аквамарин Таблиці: зелений фон заголовків Колір рамки: чорний
зелений + жовтий	Колір тексту: темно-сірий Фон: білий Колонтитули: зелений фон Таблиці: жовтий фон заголовків Колір рамки: чорний
зелений + блакитний	Колір тексту: темно-сірий Фон: білий Колонтитули: зелений фон Таблиці: блакитний фон заголовків Колір рамки: чорний

Закінчення

Назва	Опис сторінки
зелений + персиковий	Колір тексту: темно-сірий Фон: білий Колонтитули: зелений фон Таблиці: персиковий фон заголовків Колір рамки: чорний
зелений + аквамарин	Колір тексту: темно-сірий Фон: білий Колонтитули: зелений фон Таблиці: фон заголовків — аквамарин Колір рамки: чорний

Компоненти оформлення. Ідею з оформлення слід реалізувати покроково:

- ***Композиція.*** Іншими словами, компонування — загальний вигляд сторінки: що, де і як. Це співвідношення між різними об'єктами, їхніх позицій. Потрібно врахувати, що на сторінці будуть деякі службові компоненти: елементи навігації/субнавігації, рекламні банери, вибір кодування, форми для взаємодії з користувачем, написи (останнє відновлення, copyright…) — все це якимось чином має вкластися в загальну композицію сторінки.
- ***Колірна гама.*** Потрібно визначитися з кольорами. Вибір кольорів залежить від особистих переваг, концепції сайту.
- ***Графіка.*** Використання графіки має бути виправданим.
- ***Шрифти.*** У межах сайту або типу сторінок повинні бути однакові стилі оформлення тексту (заголовки різних рівнів, цитати, посилання).

Під накресленням шрифту розуміється комплект знаків певного малюнка. Незважаючи на величезну кількість шрифтів, створених для комп'ютерних видавничих систем, шрифти можна поділити на групи:

- ***шрифти із зарубками*** *(антиква — serif)*;
- ***шрифти без зарубок*** *(гротески — sans serif)*;
- ***декоративні*** *(decorative)*;
- ***рукописні*** *(script)*.

Різні дослідження показали, що шрифти із зарубками читаються легше, тому що зарубки допомагають погляду пересуватися від букви до букви, і букви при цьому не зливаються. З іншого боку, букви без зарубок легше читати в шрифтах дуже великого або дуже малого розміру. Але встановити однакові правила дуже складно (а якщо точніше — майже

неможливо), оскільки крім накреслення, величезне значення має кегль шрифту, довжина рядків, інтерлін'яж, вільний простір і навіть папір.

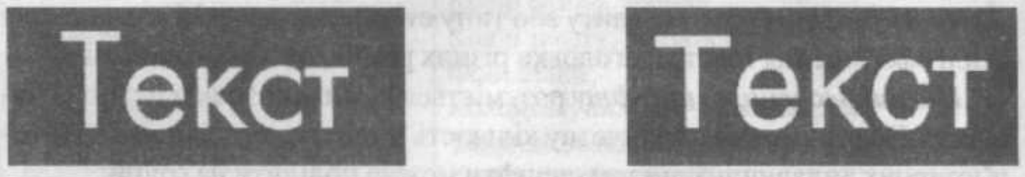
Шрифти із зарубками (serif). *Зарубки (serif)* — це поперечні елементи на кінцях штрихів букви. Шрифти із зарубками також називають *антиквенними*, тобто античними, древніми. Справа в тому, що вперше такі елементи в буквах застосували ще римляни (рис 1.10)

Шрифт **AG-Benguiat** Шрифт **AG-CenturyOldStyle**

Шрифт **Baltica**

Рис. 1.10. Шрифти із зарубками

Шрифти без зарубок (sans-serif). У шрифтах без зарубок відсутні завершальні елементи на кінцях штрихів. Назва sans-serif походить від французького *sans* — без (рис. 1.11).



Шрифт **AG-Futura** Шрифт **AG-Helvetica**

Шрифт **AG-OptimaCyr**

Рис 1.11. Шрифти без зарубок

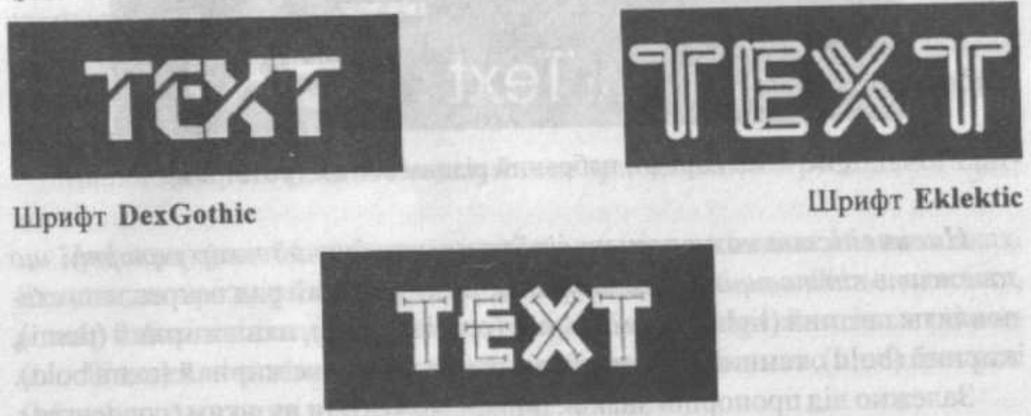
Рукописні шрифти. Рукописні шрифти (script) нагадують ручне письмо. Традиційно до них належать каліграфічні шрифти (рис. 1.12).

Шрифт **BetinaScript** Шрифт **BrushScript**

Шрифт **Decor**

Рис. 1.12. Каліграфічні шрифти

Декоративні шрифти. Цю категорію становлять численні шрифти, які не вкладаються у звичайні групи. Найчастіше їх використовують, щоб підкреслити новизну, яскравість, індивідуальність. Але ніколи не слід використовувати їх як основний текст. Вони незручні для читання, до того ж зникає ефект. Ці шрифти використовують у заголовках і для виділень (рис. 1.13).

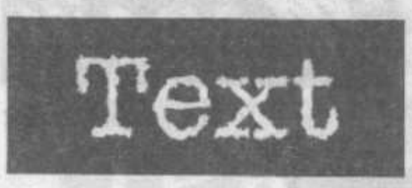


Шрифт **DexGothic** Шрифт **Eklektic**

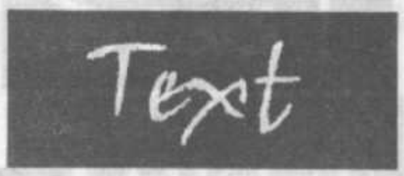
Шрифт **Follies LET**

Рис. 1.13. Декоративні шрифти

Альтернативні шрифти. До цієї групи входять шрифти, створені у власному ні на що не схожому стилі. Це нові види шрифтових форм, створені виробниками зовсім недавно, їх основу не становлять інші шрифти. Ці шрифти застосовують як заголовки, так і декоративні шрифти (рис. 1.14).



Шрифт **IceAge**



Шрифт **Jonh Handy LET**

Шрифт **Jokerman**

Рис. 1.14. Альтернативні шрифти

Кегль (size), або розмір, шрифту визначається його висотою, що вимірюється в типографських пунктах (*point* або *pt*): 12 пунктів = 1 пік, 6 пік = 1 дюйм (рис. 1.15).

Рис. 1.15. Шрифт, набраний різним кеглем (розміром)

Насиченістю називається візуально сприйняний колір шрифту, що залежить від товщини його штрихів. Безперервний ряд накреслень становлять: світлий (light), нормальний (regular, book), напівжирний (demi), жирний (bold), темний (heavy), чорний (black) і дуже жирний (extra bold).

Залежно від пропорцій знаків, шрифт може бути вузьким (condensed), нормальним (normal) і широким (extended). Звичайно у комплекті йдуть шрифти з нормальним накресленням, але більшість програм дають змогу міняти співвідношення висота-ширина шрифту й одержувати потрібний ефект.

***Виключка** — це параметр, що показує розміщення тексту в параграфі.* Виключка буває: за лівим краєм, за правим краєм, за центром, за форматом і повна виключка. Залежно від обраного типу виключки текст розміщується по-різному.

Кожний шрифт має індивідуальний вигляд. Упевнений, елегантний, богемний, оригінальний, романтичний, дружній — шрифт може мати дуже різний вигляд. Можливості практично безмежні, вам тільки потрібно визначити яке враження ви хочете зробити й вибрати придатний для цього шрифт.

СТВОРЕННЯ ВЕБ-СТОРІНОК ДЛЯ МОНІТОРІВ З РІЗНОЮ РОЗДІЛЬНОЮ ЗДАТНІСТЮ

При створенні дизайну веб-сторінки важливим моментом, на який слід звернути увагу, є *розмір монітора* й *роздільна здатність екрана* відвідувачів сайту.

Веб-сторінки повинні бути досить гнучкими для того, щоб коректно відображатися в різних броузерах і конфігураціях екрана.

***Роздільна здатність** — це кількість пікселів, що може відобразити монітор.* Більшість моніторів можуть працювати з роздільною здатністю 640 × 480, 800 × 600 і 1024 × 768 пікселів. За більшої роздільної здатності можна відобразити більше інформації, ніж за меншої. Наприклад, за роздільної здатності 800 × 600 відображається 480.000 пікселів, а за роздільної здатності 1024 × 768 — 786.432 піксела.

Отже, розмір кожного піксела зменшується при збільшенні роздільної здатності. Це відбувається тому, що на однаковій площі екрана відображається більше пікселів.

Малюнки й текст також зменшуються при збільшенні роздільної здатності.

При створенні таблиць ви задаєте їхні розміри в пікселах або відсотках. Якщо задати розміри в пікселах, буде *фіксована таблиця*. Якщо, навпаки, розмір таблиці визначений у відсотках, таблиця називатиметься *відносною*. Різниця між цими двома типами таблиць є очевидною при перегляді з різною роздільною здатністю екрана.

Щоб це проілюструвати, розглянемо такий приклад. Якщо ви створюєте таблицю із 4 комірок, кожна шириною в 100 пікселів, ця таблиця буде фіксованою, тому що задані точні розміри. Ширина таблиці завжди дорівнюватиме 400 пікселів (4 × 100). Навпаки, якщо створити таблицю

з чотирьох комірок, ширина яких дорівнюватиме 25 % від загальної ширини екрана, ширина таблиці в пікселах залежатиме від поточної роздільної здатності екрана.

Наприклад, якщо встановлено роздільну здатність 800 × 600, кожна комірка таблиці матиме ширину 200 пікселів. Ширина всієї таблиці дорівнюватиме 800 пікселам. Якщо ту саму таблицю переглянути на екрані роздільною здатністю 1024 × 768, то вона буде ширшою. Кожна комірка матиме ширину 256 пікселів, а ширина таблиці дорівнюватиме 1024 піксела.

Кожний спосіб має свої переваги й недоліки. З фіксованими таблицями простіше працювати, тому що не доведеться турбуватися про зміну розмірів таблиці в різних відвідувачів веб-сторінки.

Всі елементи таблиці повинні бути повністю завантажені перед тим, як броузер почне їх відображати. При використанні фіксованих таблиць дизайнер може змусити броузер почати відображати дані в таблиці, не чекаючи їхнього повного завантаження. Для цього застосовується стиль CSS: `<Table Style="table-layout: fixed">`. Головним недоліком використання фіксованих таблиць є невикористання площі екрана за високої роздільної здатності екрана.

Повернемося до відносних таблиць, вони потребують від дизайнера більшої роботи. Веб-сторінка однаково добре відображається за різної роздільної здатності. Якщо все зроблено правильно, сторінка добре масштабується й використовує всю площу екрана.

Піксел є основною одиницею вимірювання роздільної здатності екрана. Тому все, що стосується розміру екрана монітора, обчислюється в пікселах. Стандартною одиницею вимірювання розміру шрифту є «пункт». Коли ви визначаєте розмір тексту в пунктах, комп'ютер має перетворити його на піксели перед початком відображення на екрані. На жаль, перетворення між пікселами й пунктами не є універсальним. Це стає очевидним при створенні сторінок, призначених як для користувачів PC, так і для користувачів «Макінтош», які використовують менший коефіцієнт перетворення пунктів у піксели. Тому текст, розмір якого заданий у пунктах, виглядатиме меншим при перегляді на «Макінтош».

Важко розробити сайт із правильним балансом форми й функціональності, який би однаково добре відображався на різних платформах. Це є одним з основних завдань дизайнера. Ви маєте бути впевнені, що зміна роздільної здатності не вплине на враження відвідувача про ваш сайт.

розділ 2

ПРАКТИЧНИЙ WEB-ДИЗАЙН

РЕДАКТОРИ ДЛЯ ВЕРСТКИ ВЕБ-СТОРІНОК

Редактори для верстки веб-сторінок бувають двох типів:

- візуальні;
- текстові.

Для роботи у візуальних редакторах не потрібно спеціальних знань HTML, CSS і інших технологій для розмітки сторінок. У візуальному редакторі ви розташовуєте різні елементи вашого сайту, наче на папері, а редактор самостійно пише за вас код. Саме тому візуальні редактори ще називають WYSIWYG-редакторами. Абревіатура WYSIWYG розшифровується як *What You See Is What You Get* — що бачиш, те й одержуєш.

У текстових редакторах, як правило, бувають різні функції, що полегшують написання коду: підсвічування коду (так легше бачити, де в коді вставлено стилі або скрипти, а де просто текст), різні гарячі кнопки й клавіші, які вставляють уже готові конструкції (фрагменти коду, спецсимволи) у код тощо.

Якого ж типу слід використовувати редактор? Якщо ви вивчаєте HTML, CSS або інші технології для розмітки сторінок, якщо ви хочете навчитися створювати якісні сторінки, то, безумовно, вам потрібен текстовий редактор. Якщо ж ви не маєте часу на вивчення HTML, CSS та інших технологій,

якщо перед вами не стоять дуже складні завдання на виконання сторінки, то ставте собі візуальний редактор і користуйтеся ним, він заощаджує час і сили. А найкраще мати в себе на комп'ютері і візуальний, і текстовий редактори для різних потреб.

Візуальні редактори

Macromedia Dreamweaver MX. Професійний інструмент для створення web-сайтів і прикладних програм. Мабуть, це найкраща програма з візуальних редакторів, принаймні її полюбляють користувачі.

Розробники стверджують, що Macromedia Dreamweaver MX призначено для проектування, розробки й адміністрування професійних web-сайтів і прикладних програм. До того ж, Dreamweaver легко інтегрується з іншими програмами від Macromedia, наприклад у програму Macromedia Flash.

Adobe GoLive і LiveMotion. Можливо, Adobe GoLive сподобається тим, хто любить програми від Adobe і багато з ними працював: знайоме середовище, досить легко з'ясувати, що до чого. Крім того, ще один плюс для прихильників Adobe — всі програми від Adobe чудово взаємодіють одна з одною і доповнюють одна одну, і GoLive не виняток. Однак GoLive не більш ніж візуальний редактор для верстки веб-сторінок, підтримки таких технологій, як HTML, DHTML, CSS, XML і кількох готових Java-скриптів не слід очікувати від цієї програми. Однак слід зазначити, що вбудований редактор коду (текстовий) у цій програмі дуже якісний.

Microsoft FrontPage. Якщо вірити розробникам, то програма FrontPage дає змогу створювати веб-вузли з широкими можливостями, а також надає засоби керування ними. FrontPage «дружить» із HTML, CSS, DHTML, Javascript, дає досить широкі можливості з керування зображеннями й flash-роликами. До того ж, FrontPage «дружить» із такими технологіями, як ASP, XML, VBScript, XSL.

Hotdog. Програма має простий і зрозумілий користувачеві інтерфейс. До того ж Hotdog «дружить» із користувачем, програма також підтримує технології з Flash, SQL, PHP, ASP, має працювати з GIF-зображеннями (оптимізація, анімація), містить Html-компресор, може створювати файли довідки.

Текстові редактори

Homesite. Цей редактор, мабуть, найпопулярніший і досить потужний серед текстових. До того ж, у ньому досить легко працювати не тільки з HTML-кодом (є все — від списку атрибутів і всіх тегів аж до перевірки коду (правильність перевіряється через W3C.org)), у ньому також є підтримка технології XHTML, CSS-редактор.

HTML Pad. Ця програма теж користується великою популярністю серед користувачів. HTMLPad підтримує технології JavaScript, VBScript, SSI, ASP і Perl, уміє створювати макроси, містить багато довідкових матеріалів з CSS і HTML.

Notepad. Блокнот. У цій програмі немає додаткових функцій: ані підсвічування коду, ані вставки готових конструкцій коду — нічого, але ця програма є в наборі стандартних програм на комп'ютері в кожного користувача. За допомогою неї ви можете робити свої перші кроки в написанні коду.

Якщо у вас не надто швидкісний Інтернет, щоб отримати редактори, то ви можете купити їх у магазині, де продаються програми для комп'ютера.

ВЕРСТКА САЙТУ

Найпопулярнішим типом верстки HTML-сторінок є ***таблична верстка***. Таблиці, які спочатку проектувалися як засіб подання двовимірних масивів даних, зараз використовуються переважно як інструмент для точного позиціонування контента на веб-сторінці.

Стандарт HTML не дає змоги розташувати картинки й текст відносно один одного на чіткій відстані, а таблична верстка, в якій різні частини контента вміщуються в різні комірки таблиць, дає змогу вирішити цю проблему. До того ж, таблична верстка дуже зручна у проектуванні сайтів з великою кількістю графічних елементів: у великих малюнках майже завжди можна виділити частини, зафарбовані однаковим кольором, які можна викинути, замінивши на порожні комірки з відповідним фоном і заощадивши в такий спосіб на загальному обсязі графіки й часі завантаження сторінок.

На найпопулярніших сайтах новин ми бачимо, що текст розбито на стовпчики. Причому один стовпчик виділяється для основного тексту,

а інший для так званих «врізів». ***Врізом*** називається коротка анотація статті.

Друге застосування стовпчика зводитися до встановлення навігації по сайту. Там можуть бути посилання, новини, банери.

Існують правила верстки тексту. Отже, якщо ви відводите малий стовпчик під текст, його розмір вираховують у такий спосіб. Вибираєте гарнітуру, кегль, накреслення й пишете:

abcdefghijklmnopqrstuvwxyzabcdefghijklm

Саме така ширина є оптимальною для цього шрифту. Однак за розміру стандартної сторінки в 620 пікселів це надто багато. Якщо вас це не влаштовує, візьміть найдовше слово вашого тексту й надрукуйте його півтора рази (як у випадку з алфавітом). При використанні стовпчика для розміщення посилань тримайте його ширину від 130 до 150 пікселів. Детально про табличну верстку дивіться розділ про HTML про те, як створювати таблиці.

У плануванні, наповненні й компонуванні сайту слід орієнтуватися на 2 правила:

1. Оптимальним для веб-сторінки є обсяг тексту для однієї HTML-сторінки — в межах 4—5 «екранів» при роздільній здатності монітора 600 × 800 пікселів.

2. Користувач сприймає сайт як збірник потрібної інформації й віддає перевагу предметним ресурсам. Отже, основна інформаційна частина кожної окремої HTML-сторінки повинна перебувати в межах першого екрана.

Для коротких матеріалів досить розгорнутого підзаголовка, а для великих можлива коротка анотація. Одним з варіантів може стати розміщення матеріалів у вигляді коротких анотацій з потенційною можливістю прочитати повний документ.

До того ж, краще уникати розміщення малюнків і фотографій, що займають всю ширину першої сторінки екрана. Тоді відвідувач одразу зможе прочитати статтю, не чекаючи завантаження картинки й не «прокручуючи» екран.

Тексти повинні містити максимум корисної інформації й мінімум рекламної. Спочатку в стислому вигляді викладається інформація загального характеру, а для ознайомлення з деталями користувач переходить по по-

силанню до нового файла. При цьому дається коротка попередня анотація про те, яка інформація подана за цим посиланням.

Обсяг тексту — кількість рядків, знаків тощо — залежить від конкретного матеріалу й системи посилань. Загальна рекомендація: основний текст має міститися на одній сторінці, якщо можливо, без прокручування. На практиці це вдається не завжди. Тут може допомогти система посилань: докладніша інформація (наприклад, потрібна не всім) міститься в окремому файлі.

Експерти зазначають, що на кожній сторінці має бути не більш ніж 20-30 кБ HTML-тексту.

ОПТИМІЗАЦІЯ ЗОБРАЖЕНЬ У ФОРМАТІ GIF І JPEG

Перша версія графічного формату GIF (*Graphics Interchange Format*) була розроблена в 1987 році фахівцями комп'ютерної мережі CompuServe як простий растровий формат для обміну малюнками в Мережі. Через якийсь час у цього формату виявили ряд недоліків, що гальмують його ефективне використання. У 1989 році було розроблено нову версію формату GIF (GIF89a).

Формат GIF

Формат GIF застосовує ті самі алгоритми стискання, що й звичайні програми-архіватори, тому при записі й зчитуванні GIF-зображення ніяких втрат інформації не відбувається. Однак, на відміну від них, GIF-файли архівуються й розархівовуються автоматично.

У форматі GIF слід зберігати:

- кнопки й інші елементи навігації;
- тексти, оформлені у вигляді зображення;
- скриншоти (зверніть особливу увагу: їх у жодному разі не можна зберігати у форматі JPEG);
- будь-які зображення з малою кількістю кольорів і плавних колірних переходів.

Корисна властивість формату GIF — підтримка черезрядкового розгортання (Interlaced).

При завантаженні черезрядкового GIF-зображення броузер спочатку показує кожний восьмий рядок, потім кожний четвертий і т.д. У такому

випадку відвідувач сайту зможе зрозуміти, що зображено на малюнку, не чекаючи його повного завантаження. У деяких ситуаціях це буває зручно, отже при збереженні GIF-файлів слід завжди включати підтримку черезрядкового розгортання.

При створенні зображення, що надалі буде переведене в GIF-формат, слід враховувати особливість алгоритму LZW-стискання. Ступінь стискання графічної інформації в GIF залежить не тільки від рівня її повторюваності й передбачуваності (однотонне зображення має менший розмір, ніж безладно «зашумлене»), а й від напрямку, тому малюнок сканується порядково. Це добре видно на прикладах GIF-файлів з різним напрямом рядків. Файл із горизонтальними рядками має розмір 369 байт, а з вертикальними 883 байт (в 2,4 рази більше!).

В GIF можна призначити один або більше кольорів прозорими, вони стануть невидимими в броузерах і деяких інших програмах. Прозорість забезпечується за рахунок додаткового Alpha-каналу, що зберігається разом з файлом. До того ж, файл GIF може містити не одну, а кілька растрових картинок, які броузери можуть довантажувати одну за одною із зазначеною у файлі частотою. Це називається GIF-анімацією.

Основне обмеження формату GIF полягає в тому, що кольорове зображення може бути записане тільки в режимі 256 кольорів.

Як показує досвід, для нефотографічних зображень у більшості випадків не потрібно 256 кольорів. Видаляючи інформацію про зайві кольори із зображення, ми тим самим зменшуємо його розмір. Теоретично ми можемо використовувати будь-яку кількість кольорів у діапазоні від 1 до 256, але на практиці краще вибирати це значення з такого ряду: 2, 4, 8, 16, 32, 64, 128, 256. Це пов'язане з тим, що для зберігання колірної інформації завжди використовується ціле число біт (від одного до восьми на кожний). Приклад: якщо ви використовуєте в малюнку тільки 50 кольорів, для збереження одного пікселя однаково буде використано 6 біт. Розмір кінцевого файла буде таким, як і у випадку використання палітри з 64 кольорів.

Орієнтація на наведений вище ряд дозволить вам створити якісні компактні зображення.

При збереженні картинки у форматі GIF будь-який сучасний графічний редактор обов'язково запитає вас, яку палітру необхідно використовувати: фіксовану або оптимізовану. Фіксована палітра являє собою певний набір кольорів. У цьому випадку графічний редактор пе-

реглядає кожний піксель зображення й добирає найближчий до нього колір.

З погляду якості цей спосіб дає найгірший результат: справа в тому, що фіксовані палітри створюються без урахування особливостей конкретного зображення.

При використанні оптимізованої палітри графічний редактор спочатку аналізує зображення й створює список усіх його кольорів. Далі на підставі частоти появи відтінків формується палітра з необхідної кількості тонів. Після цього малюнок проглядається описаним вище способом, колір кожного пікселя при цьому замінюється на найближчий з палітри.

Фактично оптимізація кожного конкретного зображення полягає в доборі оптимальної кількості кольорів. Тут слід згадати про так званий *дизеринг (dithering) кольорів*. Уявімо, що у вас є палітра з синіми й жовтими кольорами, а треба одержати зелений. У цьому випадку ми можемо скласти його із синіх і жовтих точок, чергуючи їх на малюнку. Якщо поглянути здалеку, буде здаватися, що ми бачимо зелений колір. Використання дизерингу при збереженні зображень у форматі GIF може дати результат, але, на жаль, дизеринг украй негативно позначається на розмірі кінцевого файла. Саме тому доцільність застосування дизеринга слід визначати в кожному конкретному випадку.

JPEG

JPEG (Joint Photographic Experts Group) — це формат для стискання графічних файлів. JPEG — це не формат, а алгоритм стискання, заснований не на пошуку однакових елементів, як у RLE і LZW, а на різниці між пікселями. JPEG шукає плавні колірні переходи у квадратах 8 × 8 пікселів. Замість дійсних значень JPEG зберігає швидкість зміни від піксела до піксела. Зайву, з його погляду, колірну інформацію він відкидає, усереднюючи деякі значення. Чим вищий рівень компресії, тим більше даних відкидається й тим нижчою є якість. Використовуючи JPEG, можна одержати файл у 10—500 разів менший, ніж BMP!

Отже, форматом JPEG краще стискаються растрові картинки фотографічної якості, ніж логотипи або схеми, — у них більше напівтонових переходів, серед однотонних заливань з'являються небажані перешкоди. Краще і з меншими втратами стискаються зображення з високою роздільною здатністю (200-300 і більше dpi), ніж з низькою (72-150 dpi), оскільки в кожному квадраті 8 × 8 пікселів переходи виходять м'якіші за

рахунок того, що їх (квадратів) у файлах з високою роздільною здатністю більше. В JPEG слід зберігати тільки кінцевий варіант роботи, тому що кожне перезбереження призводить до нових втрат (відкидання). Як це не парадоксально, можливості алгоритму стисканя JPEG реалізовані у форматі JPEG не повністю.

Дуже корисним може бути використання прогресивного розгортання. Якщо при записі картинки у звичайному JPEG виведення на екран здійснюється рядками, то прогресивне JPEG-зображення з'являється на екрані одразу цілком, але в грубому вигляді.

У міру завантаження файла дефекти поступово згладжуються. Це дає відвідувачам можливість одразу ж оцінити фотографію й вирішити, чи слід чекати її завантаження.

розділ 3

СТВОРЮЄМО САЙТ

ГІПЕРТЕКСТ

Термін *«гіпертекстове посилання»* з'явився задовго до того, як British Telecom розробила власну версію гіперпосилань у 70-х роках. Вважається, що першим слово «гіпертекст» вжив англійський учений Тед Нельсон у 1963 році, а в 1965 році термін увійшов до його книги «Literary Machine» («Вчена машина»). До того ж, про гіперпосилання було відомо комп'ютерним фахівцям зі Стенфордського університету в 1968 році. На сайті цього університету можна подивитися чорно-білий кліп, що демонструє, мабуть, перший приклад використання гіперпосилань. У цьому кліпі «батько» комп'ютерної миші Дуглас Енгелбарт показує, як, клацаючи мишею по певних словах у комп'ютерній програмі, можна виводити на екран нові текстові сторінки.

Гіпертекст — це засіб презентації знань у багатьох вимірах, наприклад, у параграфах документа. Зараз, наприкінці XX століття, будь-який текст продовжує жити в гіперпросторі (hyperspace), де технологія електронного запису, так званий гіпертекст, робить можливим усебічний та різно-спрямований зв'язок між текстуальними сегментами, утворюючи єдину специфічну мережу з файлів локального або віддаленого комп'ютера, які тематично прив'язані до головного документа або до документа, який зберігається на комп'ютері віддаленого сайта.

Візуально гіпертекст *— це текст, у якому виділені (кольором) слова, які в будь-який час можна розкрити, тобто підвести до них курсор і натиснути ліву кнопку миші (або* ***Enter****).* Таким чином можна отрима-

ти додаткову інформацію про ці слова, тобто ці слова є посиланням на інші документи, які можуть бути текстом, малюнками, файлами тощо. Для швидкого маніпулювання з гіпертекстовими документами було створено протокол **HTTP (HyperText Transfer Protocol)**.

WORLD WIDE WEB

***WWW** — це розподілена мережева гіпертекстова інформаційна система*. Абревіатура WWW розшифровується як World Wide Web, що дослівно перекладається як «Всесвітнє павутиння». Альтернативною назвою WWW є Web. WWW побудована на основі моделі «клієнт—сервер». На так званих Web-серверах у вигляді гіпертекстових документів зберігається інформація. Її запитують, одержують і відображають Web-клієнти. Клієнтом Web виступає програма, названа Web-броузером.

Найпоширеніші програми: Internet Explorer фірми Microsoft і Netscape Communicator фірми Netscape. Проте зараз у світі спостерігається «бум» серед розробників броузерів, професіонали і аматори створюють власні програми перегляду інформації в Інтернеті та надають їх на спеціальних серверах безкоштовного або умовно безкоштовного програмного забезпечення. Залежно від контексту під Web-сервером будемо розуміти як програму, встановлену на комп'ютері, так і сам комп'ютер. Як Web-сервер часто використовують спеціально виділений комп'ютер — WWW-сервер.

Розробка системи WWW переважно проводилася в Європейській лабораторії фізики елементарних частин (CERN) поблизу Женеви.

World Wide Web (WWW) — *гіпертекстова система пошуку ресурсів Інтернету та доступу до них*. Доступ до ресурсів Інтернету (зокрема засобів гіпермедіа) здійснюється за допомогою спеціального програмного забезпечення, що називається WWW (*або Web*) browser — ***броузер***.

У вітчизняній літературі WWW-броузер перекладають як «програма перегляду системи WWW», що зазвичай відповідає функціональній спрямованості цього програмного забезпечення, але у викладанні матеріалу краще використовувати транслітерацію англійського терміну WWW броузер. Термін ***WWW броузер*** часто помилково використовують як синонім World Wide Web. Документи для системи WWW готуються спеціальним чином з використання формату HTML.

***HyperText Markup Language (HTML)** — це система для розмітки різних частин Web-документів*, що вказує *Web-броузеру*, як відображати текст, зв'язки (посилання), графіку та різноманітні медіа.

*Системою адресації, що використовується в WWW і запропонована як стандарт адресації для всієї мережі Інтернет, є **URL** (Uniform Resourse Locator — уніфікований локатор ресурсів*).

URL вміщує інформацію про метод доступу, сервер доступу, номер порту (*інформації про порт може й не бути*) та шлях до файлів.

Тобто для кращого розуміння того, що являє собою WWW і як готуються документи до подання на WWW-сервер, вам необхідно засвоїти чотири основних поняття: **WWW броузер, URL, HTTP, HTML**.

Uniform Resource Locators (URLs)

***Uniform Resource Locators (URLs)** — це схема (шаблон), з використанням якої адресуються інтернет-ресурси в WWW*. URL — це стандарт для визначення місцезнаходження об'єкта мережі Інтернет. URL діє як адреси не лише для даних, а й для ресурсів Інтернету. Дуже важливо знати, як правильно писати цю адресу.

Анатомія URL

Протокол://адреса сервера(host domain):[порт]/шлях /ім'я_даних

- *Протокол (або метод доступу) — перша частина адреси, яка відділяється від решти адреси двокрапкою та двома слешками* (**://**). Ця частина адреси визначає метод доступу (http, file, ftp, telnet, тощо).
- ***Адреса сервера** — це адреса інтернет-сервера, на якому розміщені дані або аплікація.*
- ***Порт** — додавання номера порту до **URL**, потрібне лише тоді, коли сервер даних не розміщений на стандартному порті* (наприклад, передбачається, що FTP-сервер знаходиться на порті 21, telnet — 23). Якщо сервер розміщений не на стандартному порті, то експліцитне задання номера порту розташування сервера просто необхідне;шлях/ім'я_даних — може бути від директорії файлів до повного шляху знаходження файла. Часто трапляються URL, які складаються лише з 2 частин (переважно з протоколу та адреси сервера).

Приклади URL

а) HyperText Transfer Protocol (http) означає, що дані знаходяться на WWW-сервері. Приклади:

http://www.dlab.kiev.ua/;

http://www.education.gov.ua/;

б) **file** означає, що дані розміщені на вашій локальній системі або на анонімному FTP-сервері. Приклади:

file://ftp.unesco.org/pub/doc/technoques.pdf;

в) **ftp** (File Transfer Protocol), означає, що дані розташовані на ftp-сервері. Приклад: ftp://ftp.mmsc.org/;

г) **telnet** — означає зв'язок по telnet. Приклад: telnet://idaho.com/.

HyperText Transfer Protocol (HTTP)

***HTTP** — це інтернет-протокол, що створено спеціально для швидкого маніпулювання з гіпертекстовими документами.* Подібно до інших Інтернет-інструментарів, як FTP, HTTP є протоколом «клієнт—сервер». У моделі «клієнт—сервер» програма-клієнт, що виконується на комп'ютері користувача, надсилає повідомлення на запит до програми сервера, що виконується на іншому комп'ютері в мережі Інтернет. Відповідь на запит сервер знову надсилає клієнту. У процесі обміну цими повідомленнями клієнт і сервер використовують протокол (сукупність правил, згідно з якими комп'ютери взаємодіють між собою). FTP — інші приклади протоколів «клієнт-сервер» мережі Інтернет, кожен з яких також є доступним через WWW-броузер. Проте HTTP було сконструйовано спеціально для роботи з гіпертекстовими документами.

На найпростішому рівні HTTP-сервери діють подібно до анонімних FTP-серверів, надаючи файли за запитами клієнтів.

HTTP-сервери підтримують деякі додаткові функції:

- здатність надсилати клієнту не лише файли, а й додаткову інформацію, що генерується програмами, які виконуються на сервері;
- здатність брати дані, надіслані клієнтом, і передавати цю інформацію іншим програмам, що знаходяться на сервері, для подальшої обробки.

Спеціальні програми, що знаходяться на сервері й виконують ці функції, називаються програмами **gateway** (шлюз), оскільки діють як шлюз між HTTP-сервером та іншими локальними ресурсами, такими як, наприклад, бази даних. Як FTP-сервер надає можливість доступу до величезної кількості файлів, так і HTTP-сервер надає можливість доступу до великої кількості програм: у цих двох випадках ви визначаєте (шляхом завдання URL), які (файлові або програмні) ресурси ви б хотіли отримати.

Взаємодія між сервером і gateway-програмами регулюється специфікаціями **Common Gateway Interface (CGI)**. Використовуючи CGI специфікації, програміст може легко писати прості програми або скрипти обробки запитів користувача, взаємодії з базами даних.

***Common Gateway Interface (CGI)** — це стандарт розширення функціональності WWW, що дає змогу WWW-серверам виконувати про-*

грами, аргументами роботи яких може керувати користувач. WWW-сервери дають змогу запитувати статичні HTML Web-сторінки і переглядати їх за допомогою Web browser. CGI розширює можливості користувача і дає змогу виконувати програми, асоційовані з цією Web-сторінкою, та допомагає отримати динамічну інформацію з WWW-сервера. Наприклад, користувач такого WWW-сервера може одержати останню інформацію про погоду, виконавши програму, що запитує прогноз погоди на даний момент із бази даних.

CGI-інтерфейс переважно є шлюзом між WWW-сервером і зовнішніми користувачами. Він одержує запит від користувача, передає його на виконання серверу й потім повертає результати користувачу через динамічно створену Web-сторінку. При цьому отримані Web-сторінки можуть докорінно відрізнятися одна від одної, якщо створені залежно від переданих користувачем параметрів.

Частіше за все CGI використовується для обирання інформації з баз даних. Користувач уводить запит у Web-сторінку, сервер його отримує, запускає відповідний процес для його опрацювання, одержує результат запиту й надсилає його користувачу.

Механізм CGI платформо-незалежний і може передавати дані між будь-якими WWW-серверами, що підтримують механізм CGI. Оскільки CGI заснований на файлах виконання, немає обмежень на тип програми, яка буде реалізовуватися в CGI. Програма може бути написана будь-якою з мов програмування, що дає змогу створювати модулі на виконання: c/c++, Pascal, Visual Basic або PowerBuilder. CGI-програма також може бути написана з використанням командних мов операційних систем (Shell тощо).

Принципи створення HTTP-з'єднання

Протокол HTTP створений за моделлю «запит/відповідь». Іншими словами, клієнт встановлює з'єднання з сервером і надсилає запит. У ньому зазначено: тип запиту, URL (URI, URN), версія протоколу HTTP (оскільки формат запиту може щоразу змінюватися) і місце запиту: інформація клієнта (параметри) і, можливо, супровідна інформація або тіло повідомлення. Сервер HTTP відповідає рядком статусу опрацювання запиту, що містить: версію підтримуваного протоколу, код опрацювання запиту або код помилки та інформацію, яка повертається на запит. HTTP-з'єднання ініціюється користувачем і складається із запиту до ресурсу певного сервера. У найпростішому випадку з'єднання являє собою потік даних між клієнтом-ініціатором з'єднання і сервером.

У складній ситуації в процесі передавання даних беруть участь кілька проміжних об'єктів. Вони можуть бути трьох видів: **proxy** (проміжний засіб), **gateway** (шлюз) і **tunnel** (тунель).

Proxy являє собою проміжний засіб, що приймає запит клієнта і, залежно від своєї конфігурації, опрацьовує його, а також передає оброблений запит далі за ланцюжком, наприклад, іншим проміжним серверам або запитуваному серверу. У момент прийняття запитів Proxy може працювати як сервер, а при передаванні запитів — як клієнт.

Gateway (шлюз) — це проміжний сервер. На відміну від proxy, шлюз приймає запити клієнта, ніби він і є запитуваний сервер, і передає їх далі. Робота шлюзу цілком прозора для клієнта. Шлюз використовується як ретранслятор запитів зовнішньої мережі у внутрішню, до ресурсів сервера та навпаки.

HTTP — це протокол прикладного рівня, що, як правило, працює поверх транспортного протоколу TCP/IP, хоча, як будь-який протокол прикладного рівня, може працювати на будь-якому іншому транспорті, що забезпечує надійне з'єднання. Під час роботи з TCP сервер HTTP використовує, як правило, порт — 80, хоча можливе використання й інших портів.

HTTP-з'єднання має відкриватися клієнтом перед кожним запитом і закриватися сервером після надсилання відповіді. Як клієнт, так і сервер повинні мати на увазі, що з'єднання може бути передчасно закрите або користувачем, або після закінчення часу з'єднання, або через збій системи.

HYPERTEXT MARKUP LANGUAGE (HTML)

HyperText Markup Language (HTML) — це мова з розмітками (маркерами), якою пишуться гіпертекстові документи для WWW і яка дає змогу створювати гіпертекстові зв'язки (посилання), заповнювати форми, додавати малюнки, які можна вибирати мишкою. Html-сторінка (веб-сторінка) — це гіпертекстовий документ, написаний мовою HTML (HyperText Markup Language — мова розмітки гіпертексту), що містить звичайний текст і елементи керування (теги): засоби форматування тексту й посилання на інші об'єкти (графіку, аудіо- та відеозапис, а також на інші web-сторінки).

Типи Web-документів

HTML-документи бувають двох типів:

- статичні — це Web-сторінки, що існують у форматі HTML-файлів на момент їхнього запиту.

- динамічні Web-сторінки формуються спеціальними програмами під час надходження запиту. При створенні динамічних сторінок використовуються дані, які задаються самим користувачем. Ці дані передаються Web-броузером серверу як параметри. Web-сервер одержує запит на створення динамічної сторінки й запускає програму, що називається «скрипт», або «сценарій», передаючи їй отримані від користувача дані. Цей механізм передавання параметрів від користувача сценарію називається Common Gateway Interface (CGI), а самі програми іменуються CGI-сценаріями або CGI-скриптами. Назва «скрипт» походить від невеликих командних файлів (scripts) операційної системи Unix, у яких порядково записуються директиви командного інтерпретатора shell. Скрипти служать для виконання певного набору дій, що часто повторюються. Як скрипти використовуються програми, написані мовами C, Perl, BASIC та ін. CGI-скрипти створюються для забезпечення доступу засобами WWW до різних баз даних і пошукових систем, надаючи користувачу Інтернет як засоби віддаленого доступу до різних служб, так і єдиний механізм самого доступу. Це робить Web-броузер універсальним інструментом для роботи в Інтернет.

ПРОГРАМИ ПЕРЕГЛЯДУ ІНФОРМАЦІЇ В ІНТЕРНЕТ (БРОУЗЕРИ)

WWW-броузер — це програма, яка використовується для перегляду матеріалів (документів), підготовлених для WWW.

Прикладами WWW-броузерів можуть бути такі програми: **Netscape Navigator**, **Microsoft Internet Explorer, Opera** та ін.

Броузери можуть інтерпретувати Інтернет-адреси (URL), розмітку мови HTML, а також розуміти кілька Інтернет-протоколів: HTTP та FTP.

Броузери: коротка історія

Унаслідок використання гіпертекстових технологій у 1991 році з'явилися броузери — програми перегляду інформації.

У 1993 р. Марко Андріссен, студент університету штату Іллінойс, запропонував на суд громадськості версію броузера **Mosaic**, яка мала графічний інтерфейс і працювала під ОС Unix. Елементи інтерфейсу виявились настільки зручними та вдалими, що й досі броузери концептуально повторюють їх. На початку 1994 р. Джим Кларк (керівник фірми Silicon Graphics) та Марк Андерсен (співробітник NCSA Mosaic) розпочали нову

справу, сфокусувавши свої зусилля на розробці нового Web-інтерфейсу. У результаті з'явився найпопулярніший WWW-броузер — Netscape Navigator, що зробив фірму Netscape Communications всесвітньо відомою. Програма швидко завоювала серця користувачів, і популярність її стрімко росла, поки не втрутилась всюдисуща Microsoft.

Якщо перший броузер Internet Explorer 2.0 ще значно поступався Netscape Navigator 2.0, то вже версії 3.х були приблизно рівні за можливостями. Наступні версії програм — 4.х — були вже не просто броузери, а інтегровані пакети, і здатні забезпечити потреби компанії або індивідуального користувача в галузі електронних комунікацій.

Майже кожен користувач Інтернету значну частину часу роботи в мережі проводить за переглядом інформації в WWW (World Wide Web), що дослівно перекладається як «всесвітня павутина». І від того, наскільки ефективно він це робитиме, залежить тривалість сеансів зв'язку в on-line, а отже і сума, яку доведеться платити провайдеру за користування телефоном.

Якими навичками треба володіти, щоб мінімізувати ці показники? Зазвичай, треба вміти швидко знаходити в мережі необхідну інформацію. Навчитися цьому можна, попрацювавши в Інтернеті тривалий час. Складно вигадати універсальну допомогу для всіх. Але багато хто забуває ще про одну сторону проблеми ефективного Web-серфінгу: використання всіх можливостей програми, в якій переглядається інформація WWW-броузера Web.

Використовують два броузери — Netscape Navigator (NN) та Microsoft Internet Explorer (IE).

Який броузер вибрати?

Взагалі, це не має великого значення. Одні функції краще реалізовані в Internet Explorer, інші — в Netscape Navigator. Кожен з пакетів може успішно розв'язувати покладені на нього завдання. Існує русифікована версія Internet Explorer. Інтерфейс броузера дає змогу одночасно працювати з великою кількістю вікон, розташуванням яких дуже легко керувати. Зауваження: як і у всіх інших, у нього є недоліки.

Основи роботи в броузері

Робота з Netscape Navigator

http://www.ziet.zhitomir.ua/ua/net/met/g4.html
http://www.slovnyk.org/doc/guide/met/g4.html

Робота з Internet Explorer

http://netblaze.narod.ru/FILES/doki.html
http://netblaze.narod.ru/FILES/doki.html

Для завантаження деяких сторінок з мультимедійними елементами можуть знадобитися додаткове програмне забезпечення, наприклад, плеєри, що пітримують програми перегляду мультимедійної інформації.

Їх можна знайти за адресою:
http://www.macromedia.com/

off-line броузер

Головною особливістю офф-лайнових броузерів є те, що вони призначені для ефективного зберігання Web-сторінок на локальному диску з метою їх подальшого перегляду.

Сторінки можуть зберігатися цілком, з усіма графічними елементами. Цим користувач значно заощаджує свій час. Всі офф-лайнові броузери мають поліпшений алгоритм завантаження та зберігання сторінок, завдяки якому всі процеси відбуваються значно швидше.

Такі броузери можуть зберігати навіть цілі сайти, що дуже зручно для створення «дзеркальних» сайтів. Більшість офф-лайнових броузерів має набір додаткових можливостей, з яких дуже корисними є такі:

- виконання задач за розкладом;
- здатність до оновлення вже скопійованих сторінок;
- пошук сторінок за заданими критеріями.

Завдяки тому, що копіювання може здійснюватись на будь-який носій, можна переглядати інформацію навіть за відсутності прямого доступу до Інтернету.

ОГЛЯД СУЧАСНИХ HTML-РЕДАКТОРІВ

Загальна інформація

Сучасні тенденції у світі нових інформаційних технологій можна розглядати в двох ракурсах: з точки зору розробників та користувачів.

У першому випадку для якісної роботи потрібні високоінтелектуальні рішення, широкий науковий світогляд тощо.

В іншому випадку потрібно пам'ятати, що величезна армія розробників працює над тим, щоб зробити використання нових інформаційних технологій простою і доступною справою навіть для тих, хто погано уявляє собі, як користуватися мишкою. Якість розробки залежить від того,

наскільки зручно зроблене програмне забезпечення, наскільки передбачено всі дії та бажання користувача-початківця, наскільки добрі зроблена система допомоги в кожній системі, редакторі.

Нині сервіс WWW дуже популярний в Інтернеті. Всі документи для нього пишуться на HTML (HyperText Markup Language) — мові розмітки гіпертексту. Останній від зазвичайго тексту відрізняється тим, що в ньому існують посилання — активні області, клікаючи на які мишкою, ми змушуємо броузер відображати вміст нових файлів або фрагментів.

Користувач може користуватися редактором, який допомагає «генерувати» теги. Сторінки будуть сумісні з усіма типами броузерів.

Основні принципи роботи в будь-якому HTML-редакторі

За допомогою спеціальних міток (їх система і називається розміткою) розробник повідомляє броузеру про те, що хоче закінчити абзац, змінити колір, вставити зображення. Ці мітки називаються тегами і беруться в кутові дужки < >. Сучасний HTML досить складний. Але початківцям не обов'язково знати всі тонкощі та нюанси. Досить засвоїти основи роботи в редакторі, а з досвідом роботи прийде й розуміння нових можливостей редакторів.

Усі популярні нині текстові процесори — Microsoft Word 2000, Corel WordPerfect 2000 тощо — також оснащені безліччю різноманітних засобів для створення та редагування HTML-документів, причому їх арсенал постійно удосконалюється. Аналітики прогнозують, що в майбутньому спеціальні HTML-редактори можуть зникнути як клас. Дійсно, користувачам Microsoft Word, наприклад, не треба вивчати коди HTML, Java-скрипти тощо.

Існує думка, що для створення сайту необхідно одночасно працювати з кількома редакторами. Це майже завжди правда. Однак новачків це не стосується.

Функції сучасних HTML-редакторів дуже різноманітні: деякі з них, наприклад, забезпечують формування тільки окремих сторінок, інші, навпаки, застосовуються для проектування цілих веб-вузлів з подальшим завантаженням їх на віддалені сервери в Інтернеті. Сьогодні важко провести межу між професійними HTML-редакторами і тими, що призначені для аматорської веб-творчості.

Спочатку поговоримо про найпопулярніші та порівняно недорогі продукти, а потім розглянемо властивості тих, про які не згадати просто неможливо.

AOLpress 2.0 (http://aol.com)

За допомогою AOLpress можна навчитися:

- проектувати не тільки окремі сторінки, а й увесь сайт: (*File/ New New/ MiniWeb);*
- створювати активні зони графічних елементів оформлення: *(Element/ Image Map);*
- автоматично перевіряти англійську орфографію: *(Tools/ Spell/ Check);*
- створювати посилання: Tools/ *Check Links);*
- формувати карту Веб-сервера: (Tools/ *Webize Directory).*

Вікно редагування AOLpress одночасно виконує й функції броузера, AOLpress має непогані інструменти для роботи з таблицями (меню Table), фреймами *(Format/ Frames)*, формами *(Format/ Form).*

FTP-засоби для управління Веб-вузлом дають змогу використовувати аплети Java (Element/ Java Apple*t)* та виправляти помилки в імпортованих файлах (правда, без повідомлень). У AOLpress 2.0 повністю реалізована лише підтримка специфікацій HTML 3.2. Інтерфейс програми важко назвати інтуїтивно зрозумілим, і за замовчуванням пакет не підтримує кирилицю.

HomeSite 4.5.1 http://www.allaire.com/products/homesite/index.cfm

HomeSite має низку властивостей, які традиційно подобаються професіоналам:

- має простий та зручний інтерфейс, не займає багато місця на диску;
- настроювати в ньому можна майже все — від панелей інструментів і клавіатурних комбінацій (Options Customize) до кольорів, якими будуть відображатися теги на екрані, та алгоритмів валідаторів (Options -> Settings);
- редактор створений не з метою позбавити вас необхідності вивчати HTML-код, а для того, щоб прискорити та спростити роботу з його використання.

Принцип, якого намагаються дотримуватися автори програми, називається «Pure HTML» (чистий HTML). Редактор підтримує всі популярні нині стандарти й технології Інтернету: DHTML, SMIL, CSS, JavaScript, ASP.JSP, Perl, CFML, VBScript, XML і XSL. Але не треба лякатися скорочень! Є в HomeSite і майстри, і візуальні засоби розробки, щоправда, їх присутність тут — швидше данина моді, ніж необхідність та сама проблема зайвого коду.

Більшість «помічників» розташовані у вікні New Document (File New). Після їх використання можна вдатись до послуг інспектора коду (*Tools CodeSweeper*), і хоч у його «розумових здібностях» ніхто не сумнівається, але все-таки інтелекту йому бракує, він може іноді щось і зіпсувати (від цього застерігають навіть автори програми).

HomeSite містить потужний модуль для редагування CSS-таблиць (кнопка *Style Editor* на панелі інструментів); володіє прекрасним механізмом пошуку (меню Search); його вбудований FTP-клієнт за своїми можливостями нічим не поступається спеціалізованим програмам (рекомендації щодо його настроювання ви знайдете в довідковій системі). Цей редактор дає змогу проектувати не тільки окремі сторінки, а й цілі Веб-вузли (*Tools Project*), а також автоматично перевіряти в них правильність посилань (*Tools Verify Links)*, щоправда, ці дії все-таки краще робити вручну.

CoffeeCup HTML Editor 8.5 http://www.coffeecup.com/editor/

- Назвою своєї фірми автори, мабуть, хотіли наголосити, що працювати з їх продуктами так само приємно, як вести бесіду в колі друзів за кавою. Справді, CoffeeCup HTML Editor 8.5 — прекрасний редактор з барвистим інтерфейсом і великими можливостями. Він сподобається як новачкам, так і професіоналам.

До складу CoffeeCup HTML Editor 8.5 входять:

- досить простий графічний редактор (Tools *Image Companion);*
- засоби для нарізки зображень (Tools *Launch Image Slicer);*
- вбудований FTP-клієнт (Tools FTP Upload and Download, Tools Launch *Direct FTP)*;
- потужні інструменти для роботи з таблицями (*Insert Table, Insert Quick Table)* і кадрами (Tools *Frame Designer).*

З його допомогою ви зможете редагувати *ASP-, PHP-, XML-, XSL-файли*, чистити *HTML-, XML-, XHTML-код*, проводити глобальний пошук із заміною по всіх пов'язаних сторінках, додавати ваші особисті теги. Тут є непогана довідкова система, величезний набір різних майстрів, готових форм, таблиць. До HTML Editor 8.5 можна підключити й інші окремі програми, розроблені CoffeeCup. Ліцензійна версія редактора передбачає понад 3000 графічних елементів для оформлення Веб-сторінок (піктограм, фотографій, малюнків) та близько 100 сценаріїв JavaScripts, 175 анімованих GIF-файлів. Безкоштовний варіант ПЗ має меншу кількість допоміжних елементів у бібліотеках. Недоліком можна вважати створення редактором зайвого коду.

Home Page 3.0 (http://macromedia.com)

Цей редактор повністю виправдовує свою назву і призначений для створення простих домашніх сторінок. В останній версії пакета з'явилися майстри, які полегшують проектування невеликих сайтів і сторінок з фреймами (File/ New), значно поліпшилась довідкова система. До того ж, є безліч аплетів, Java-скриптів, шаблонів Веб-документів, бібліотека графічних елементів (меню Insert).

Можливості Home Page цілком задовольняють сучасні вимоги до такого класу продуктів:

- у ньому є засоби для перенесення файлів по протоколу FTP і редагування Веб-документів на віддаленому сервері (File Remote);
- непогано виконаний пошуковий механізм (Edit Find/Change), перевірка посилань (Tools Verify Links and References), є засоби англійської орфографії (Tools Spelling);
- відносно проста процедура побудови таблиць (Insert Table) і форм (Insert Form);
- реалізоване розфарбування коду. Є і досить корисна функція Tools Document Statistics, за допомогою якої легко оцінити час завантаження на лініях з різною пропускною здатністю.

У Home Page можна використовувати будь-які типи растрових файлів — програма автоматично перетворює їх на формат GIF. І все-таки, незважаючи на «домашню» назву, пакет розрахований швидше на досвідчених користувачів.

Microsoft Frontpage 98/2000 (http://microsoft.com)

Про таке «явище», як Microsoft Frontpage, веб-майстри відгукуються досить суперечливо: одним він подобається, іншим — ні. Frontpage виконаний у стилі Microsoft: знайомі з Office кнопки, меню, панелі інструментів, «досвідчені» майстри готові послідовно, крок за кроком, провести користувача за руку (точніше, «за мишку») по всіх етапах створення веб-сторінки або цілого сайту. Тому якщо ви — досвідчений користувач Microsoft Office, освоїти цей HTML-редактор вдасться без особливих зусиль.

До складу Frontpage входять три основних продукти: Frontpage Explorer (управління елементами Веб-вузла), Frontpage Editor (безпосередньо сам HTML-редактор) і Frontpage Web (засіб, за допомогою якого можна перевіряти правильність використання HTTP-зв'язків).

Можливості Frontpage щороку зростають: якщо версія, випущена в 1998 р., підкуповувала користувачів, наприклад, наявністю компонента

WebBot, що дає змогу навіть новачкам, які не мають найменшого уявлення про програмування, створювати сценарії обробки форм або забезпечувати функціонування форумів, то Frontpage 2000 характеризується потужними засобами для колективної роботи, що дає можливість погоджувати дії багатьох професіоналів, залучених до проекту.

Макромедіа Dreamweaver (http://*www.macromedia.com/*)

Dreamweaver *— єдиний редактор, що заслуговує на увагу, оскільки має чудові механізми очищення коду*. Сама Macromedia рекламує його так: «редактор для тих, хто думає на HTML». У ньому є все плюс тісна інтеграція з Macromedia Flash — популярного нині додатком для створення Веб-сторінок, заснованих на векторній графіці. 30 днів можна працювати безкоштовно.

Інструментарію для створення власних сайтів та сторінок — безліч. Проте професіоналізм виконання не залежить від середовища, у якому проводитися розробка.

СТВОРЕННЯ HTML-СТОРІНКИ

HyperText Markup Language (HTML) — це мова з розмітками (маркерами), якою пишуться гіпертекстові документи для WWW і яка дає змогу створювати гіпертекстові зв'язки (посилання), заповнювати форми, включати малюнки, які можна вибирати мишкою.

Написання вишуканого HTML-документа передбачає такі два аспекти: технічний *(правильне конструювання HTML-документа)* та дизайнерський (*цікаве подання документа*).

Перший аспект пов'язаний зі знанням основ мови HTML та правил конструювання HTML-документа. Другий — зі смаком розробника HTML-документа, а також з досвідом роботи з іншими зразками HTML-документа. Ми зупинимось спочатку на технічному аспекті. Ті з користувачів, що вже мають деякий досвід роботи в Інтернеті, можуть зауважити, що знання мови HTML не є таким необхідним, оскільки існує велика кількість HTML-редакторів, що дають змогу автоматизувати процес перетворення текстових файлів у HTML-файли. У дечому вони праві, але в Україні найчастіше використовуються некомерційні HTML-редактори і почасти в процесі перетворення допускаються синтаксичні помилки, які без знання основ мови HTML виправити просто неможливо.

СТВОРЕННЯ НАЙПРОСТІШОГО HTML-ДОКУМЕНТА

Почнемо з прикладу найпростішого HTML-документа:

```
<HTML>
<HEAD>
<TITLE> Це назва документа </TITLE>
</HEAD>
<BODY>
<H1> Це заголовок - заповнювати обов'язково</H1>
<P>ТЕКСТ - абзац </P>
<B> Сьогодні ми будемо вивчати, як створювати
списки, що не містять переліку (1, 2 тощо): </B>
<UL>
<LI> перший елемент списку
<LI> другий елемент списку
<LI> третій елемент списку
<LI> останній елемент списку
</UL>
<HR>
Це була горизонтальна лінія.
</BODY>
</HTML>
```

Як бачимо, HTML-документ виглядає як звичайний текстовий документ, у якому надруковано спеціально маркери. У мові HTML ці маркери називаються **теги (tags)**. Тобто можна готувати такі документи в простих текстових редакторах: NotePad тощо.

Перший крок до створення Веб-сторінки — підготовка документів у HyperText Markup Language (HTML). Ця мова складається з кількох тегів, які читаються броузерами. Теги визначають стилі тексту, розташування сторінок, їх зв'язки з іншими документами та файлами.

Для підготовки HTML-документа можна скористатись як текстовим редактором, так і спеціальним HTML-редактором, щоб прискорити процес створення Веб-сторінки.

Перед початком роботи дамо декілька порад:

1. Ви, можливо, знайдете таку сторінку, яка вам дуже сподобається. Можна її скопіювати до свого комп'ютера. Для цього використайте команду **Save as**, що знаходиться в **File** меню вашого броузера.

2. Щоб скопіювати малюнок, який вам сподобався, підведіть мишку до цього малюнка і натисніть праву кнопку. Потім виберіть команду **Save this Image as..**, далі виберіть директорію, куди ви бажаєте зберегти малюнок, і натисніть **OK**.

Що потрібно для створення Веб-сторінки?

- ідентифікувати призначення HTML-тегів;
- відкрити робочий простір для створення нової веб-сторінки;
- створити загальну структуру Веб-сторінки в HTML-форматі за допомогою текстового редактора;
- відкрити документ у межах Веб-броузера.

HTML-документ складається з двох частин: заголовка **<HEAD>** та тіла (**<BODY>**). Заголовок містить інформацію про документ, але на екрані не відображається. Тіло містить те, що відображається на екрані.

Загальна структура будь-якої HTML-сторінки

```
<HTML>
<HEAD>
** Інформація, яка вноситися до заголовока і
використовується Веб-сервером**
</HEAD>
<BODY>
** Тіло вміщує інформацію, яка відображається на
екрані **
</BODY>
</HTML>
```

Заголовок і тіло заходяться всередині тега **<HTML> </HTML>**.

Зауважте, що Веб-сторінка може відображуватись вашим броузером і без наведених вище тегів. Проте за наявності наведених тегів сторінка відповідатиме міжнародним стандартам і гарантуватиме сумісність з усіма Веб-броузерами.

**
** — кінець рядка;

<P> — параграф (тобто наступний текст або малюнок буде розміщений через рядок);

<LI> — тег, який помічає елемент списку;

<HR> — горизонтальна лінія;

теги-контейнери, що вказують на якусь дію і складаються з початкового та заключного тегів. Синтаксис початкового тега такий, як і в простого

тега (тобто назва тега подається між кутовими дужками), а перед назвою тега ставитися слеш.

У наведених прикладах, окрім тегів **<HTML>**, **<HEAD>** та **<BODY>**, це теги:

<TITLE>..</TITLE> — назва документа, що друкується окремо від текста документа (у титул-барі броузера);

<H1>..</H1> — вказує, що розміщений між тегами текст є заголовком. У мові HTML є 6 рівнів заголовків (від 1 до 6). H1 — це найвищий рівень;

<B>..</B> — вказує, що розміщений усередині текст буде відображатись жирним шрифтом;

<UL>..</UL> — вказує, що розміщений усередині текст є **ненумерованим списком;**

<OL>..</OL> — вказує, що розміщений усередині текст є нумерованим списком.

Послідовність створення першого HTML-документа:

Крок 1. Відкрийте вікно редактора.

Якщо ви збираєтесь створювати свою власну Веб-сторінку українською чи російською мовою, вам треба вибрати шрифт, який задовольняє кодування – Win1251, щоб потім не виникло проблем з переглядом сторінки броузером

Крок 2. Введіть такий текст:

```
<HTML>
<HEAD>
<TITLE> Home page of (надрукуйте ваше прізвище та
ім'я латинськими літерами)<TITLE>
<meta http-equiv=«Content-Type» content=«text/
html; charset=windows-1251»>
</HEAD>
<BODY>
** У цьому місці напишіть інформацію про себе в
довільній формі:
чим ви займаєтесь, чим захоплюєтесь (українсь-
кою, російською або
англійською мовами) **
</BODY>
</HTML>
```

Зверніть увагу, що **<TITLE>.....</TITLE>** розташовано всередині тега **<HEAD>....</HEAD>**. Тобто інформацію, що знаходиться в титулі, не видно на екрані. Цей тег унікальним чином ідентифікує кожний документ і відображається зверху в титул-барі вікна броузера.

Збережіть набраний вами документ під назвою «first.htm» у робочому каталозі.

Крок 3. Відображення документа за допомогою веб-броузера.

Увага! Якщо при відображенні в броузері ви бачите замість текстових невідомі символи, перенастройте відображення символів (View Encoding).

У меню **File** виберіть **Open File...** . За допомогою вікна діалогу знайдіть і відкрийте ваш файл «first.htm». Ви побачите, як виглядає Веб-сторінка. У титул-барі броузера ви маєте побачити «**Home page of...**» (ваше прізвище та ім'я), а в робочому просторі броузера — інформацію про вас.

Ваша перша Веб-сторінка зроблена! Можливо, вона виглядає не так, як ви собі уявляли, можливо вам треба виправити деякі помилки, або ж додати деяку інформацію. Тобто у вас виникає необхідність модифікувати вашу Веб-сторінку.

МОДИФІКАЦІЯ HOME-СТОРІНКИ

Крок 4. Модифікація веб-сторінки.

Для модифікації вашої веб-сторінки вам треба:

а) активізувати або ж повторно завантажити текстовий редактор, у якому ви створювали свою веб-сторінку, і відкрити її (тобто завантажити файл first.htm),

б) внести зміни до тексту веб-сторінки і за допомогою Save збережіть її.

Крок 5. перегляд модифікованої веб-сторінки.

Поверніться до броузера (активізуйте його або повторно завантажте, якщо раніше з нього вийшли). У тому випадку, коли ви просто активізували броузер, знову поверніться до веб-сторінки, але змін ви не побачите. Побачити їх можна, лише натиснувши кнопку **Reload** (тобто повторно завантажити документ), або ж вибрати пункт меню **Reload** у веб-броузері (у броузері Netscape Navigator **Reload** знаходиться в меню **View**). Ця команда дає змогу броузеру читати той самий HTML-файл і показувати всі зміни. Користувач має побачити новий текст, який ви модернізували.

Якщо треба повторно завантажити веб-броузер, повторіть усі дії, наведені в кроці 3.

РОБОТА З ТЕКСТОМ

Теги керування стилем тексту

Жирний шрифт *(bold):*

Для керування густиною забарвлення шрифту застосовуються теги:

```
<b>текст</b>
<strong>текст</strong>
```

Відмінність тега **<b>** від тега **** у тому, що тег **<b>** вказує броузерові виводити текст жирним шрифтом, а тег **** указує, що текст треба виділити. Як броузер буде виділяти текст (жирним шрифтом, курсивом або жирним курсивом), залежить від конкретного броузера.

Приклад:

Дія тега

Дія тега <b>

Курсив (italic):

Курсивний шрифт виводиться за допомогою тегів:

```
<i>текст</i>
<em>текст</em>
```

Приклад:

Курсив

Примітка: тег **** використовується для виділення слова з тексту, і в різних броузерах може працювати по-різному.

Підкреслений шрифт (underline)

Виводиться за допомогою тега:

```
<u>текст</u>
```

Приклад:

Підкреслений шрифт

Перекреслений шрифт (strike):

Виводиться за допомогою тега:

```
<strike>текст</strike>
```

Приклад:

~~Перекреслений шрифт~~

Надрядковий індекс (Superscript):

Виводиться за допомогою тега:

```
<sup>текст</sup>
```

Приклад:

основний текст надрядковий індекс

$2^3=8$

Підрядковий індекс (Subscript):

Виводиться за допомогою тега:

_{текст}

Приклад:

основний текст підрядковий індекс

C_2H_5OH.

Імітація стилю друкарської машинки (Teletype)

<tt>текст</tt>

Приклад:

Teletype.

Шрифт для виведення цитат (citation)

<cite>текст</cite>

Приклад:

основний текст *цитата.*

Шрифт для виведення вихідного тексту програм (code)

Виводиться за допомогою тега:

<code>текст</code> або **<samp>текст</samp>**

Відображається моноширинним шрифтом Courier.

Приклад:

Звичайний шрифт

`Шрифт для текстів програм.`

Шрифт для виділення змінної в програмі

Виводиться за допомогою тега:

<var>текст</var>

Приклад:

текст програми *змінна.*

Шрифт для імітації введення з клавіатури

Виводиться за допомогою тега:

<kbd>текст</kbd>

Відображається шрифтом Courier.

Приклад:

Звичайний шрифт.

Шрифт для імітації введення з клавіатури.

Заголовки

Виводяться за допомогою тегів:

```
<h1>Найбільший заголовок</h1>
...
<h5>Дуже маленький заголовок</h5>
```


Блок з відступом

Виводиться за допомогою тега:

<BLOCKQUOTE>текст</BLOCKQUOTE>

Приклад:

основний текст

блок тексту з відступом

Теги керування кольором і розміром шрифту

Задання базового шрифту

Розмір, колір і стиль шрифту за замовчуванням задається за допомогою тега **<BASEFONT>** і не застосовується до заголовків.

Якщо базовий шрифт не заданий, за замовчуванням використовується шрифт із розміром 3

атрибути елемента **<BASEFONT>**

color = (колір).Колір шрифту.

size = (ціле число від 1 до 7). Розмір шрифту

face = (список розділених комами назв шрифтів).

Приклад:

<BASEFONT SIZE=«2»>

Встановлюємо розмір базового шрифту рівним двом.

Збільшення розміру шрифту:

Виконується за допомогою тега:

<big>текст</big>

Приклад:

основний текст;

Збільшений текст.

Зменшення розміру шрифту

Виконується за допомогою тега:

<small>текст</small>

Приклад:

основний текст;

Зменшений текст.

Керування розміром шрифту за допомогою тега <FONT>

Розмір шрифту змінюється за допомогою атрибута SIZE тега **<FONT>**

Приклад:

Текст програми:

```
<font size=«1»>size=1</font>
<font size=«2»>size=2</font>
```

```
<font size=«3»>size=3</font>
<font size=«4»>size=4</font>
<font size=«5»>size=5</font>
<font size=«6»>size=6</font>
<font size=«7»>size=7</font>
```

В атрибуті SIZE можна вказувати розмір шрифту щодо поточного розміру SIZE

Текст програми:

```
<font size=« + 1»>size  + 1</font>
<font size=« + 2»>size  + 2</font>
<font size=«-3»>size -3</font>
```

Керування кольором шрифту за допомогою тега <FONT>

Колір шрифту змінюється за допомогою атрибута COLOR тега **<FONT>**

Приклад:

Текст програми:

```
<FONT COLOR=«#FF0000»>FONT COLOR=«#FF0000»</FONT>
<FONT COLOR=«red»>FONT COLOR=«red»</FONT>
```

Результат виконання:

FONT COLOR=«#FF0000»
FONT COLOR=«red»

Задання шрифтів за допомогою тега <FONT>

Ім'я шрифту задається за допомогою атрибута FACE тега **<FONT>**

```
<font face="ім'я шрифту"></font>
```

Задає ім'я шрифту. Можна задавати кілька шрифтів через кому, у цьому випадку використовується перший-ліпший шрифт.

Приклад:

Текст програми:

```
<font face=«Impact», «Arial Cyr», «Arial»,
«MS Sans Serif»>
текст буде виведений шрифтом "Impact" за наявності його на вашому комп'ютері.
</font>
```

Теги для форматування тексту:

Виведення на екран відформатованого тексту:

Виконується за допомогою тега **<pre>**текст**</pre>**

Текст, що знаходиться між цими тегами, буде виведений у тому ж вигляді, у якому ви його надрукували в документі, тобто з усіма пробілами й переносами.

Приклад:

Текст програми:

```
<PRE>
Відформотований
                текст
</PRE>
```

Результат виконання:

Відформатований
текст

Коментарі в програмі:

Коментарі вставляються в програму за допомогою тегів:

```
<!—коментарі—>
```

Для MSIE ще можна застосовувати теги:

```
<COMMENT>коментарі</COMMENT>
```

Ці теги призначені для вставки в документ HTML-рядків коментаря, що не відображаються броузером.

Приклад:

```
<!-
і коментар
на декілька рядків
—>
```

Перехід на наступний рядок:

Примусове перенесення рядка виконується за допомогою тега **
**

Приклад:

Текст програми:

```
Виконуємо <br> перехід на <BR> наступний рядок
```

Результат виконання:

Виконуємо
перехід на
наступний рядок

Заборона переносу:

Теги **<nobr>**текст**</nobr>** вказує броузерові, що виведення тексту між ними має виконуватися одним рядком.

Якщо рядок не вміщується у вікно броузера, з'являється горизонтальна лінійка прокрутки.

Керування вирівнюванням тексту:

Вирівнювання блоку тексту здійснюється за допомогою атрибута ALIGN тега **<DIV>**текст**</DIV>**

Атрибут ALIGN може набувати значення:

ALIGN=«LEFT» — вирівнювання за лівим краєм;

ALIGN=«RIGHT» — вирівнювання за правим краєм;

ALIGN=«CENTER» — вирівнювання за центром;

ALIGN=«JUSTIFY» — вирівнювання за обома краями.

Приклад:

Текст програми:

```
<DIV ALIGN="CENTER">Текст, вирівняний за центром</DIV>
```

Результат виконання:

Текст, вирівняний за центром

Взагалі, атрибут ALIGN можна застосовувати в багатьох тегах, наприклад:

<P ALIGN="JUSTIFY">текст</P> — вирівнювання абзацу;

<TD ALIGN="CENTER">текст</TD> — вирівнювання тексту в комірці таблиці;

<H1 ALIGN="CENTER">текст</H1> — вирівнювання заголовка і т.д.

Відцентрувати блок тексту можна також і за допомогою тега **<CENTER>текст</CENTER>**

Горизонтальна роздільна лінія:

Виведення горизонтальної лінії здійснюється за допомогою тега **<hr>**

У цьому тегу можна застосовувати атрибути:

ALIGN=LEFT, CENTER, RIGHT — вирівнювання лінії;

NOSHADE — лінія без тіні;

SIZE — товщина лінії в пікселях;

WIDTH — ширина лінії.

Приклад:

Текст програми:

```
<HR>
<HR ALIGN=«CENTER» SIZE=«10» WIDTH=«50%» NOSHADE>
```

Результат виконання:

звичайна лінія:

лінія шириною 10 пікселів, вирівняна за центром, шириною 50 відсотків від ширини сторінки, без тіні:

СТВОРЕННЯ ФРЕЙМІВ

Використовуючи фрейми, що дають змогу розбивати Веб-сторінки на підвікна, можна значно поліпшити зовнішній вигляд і функціональність Веб-програм. Кожне підвікно, або фрейм, має свою URL-адресу, що дає змогу завантажувати його незалежно від інших фреймів.

Кожен фрейм має власне ім'я (параметр NAME), що дає змогу переходити до нього з іншого фрейму.

Фрейми використовуються для:

- розміщення інформації, яку необхідно показувати постійно (логотип фірми, копірайт, набір кнопок управління);

- приміщення змісту всіх або частини Веб-документів, що знаходяться на Веб-сервері. Це дає змогу користувачеві швидко знаходити інформацію;

- створення вікон результатів запитів, коли в одному фреймі знаходиться власне запит, а в іншому — результат запиту;

- створення форми типу «майстер-деталь» для Веб-програм, що обслуговують бази даних.

***Фрейм-документ** — це специфічний вид HTML-документа.* Його структура не містить тега **<BODY>** і, відповідно, не несе інформаційного навантаження. Він описує тільки підвікна (фрейми), що міститимуть інформацію.

Структура HTML-документа, що описує фрейми:

```
<HTML>
<HEAD>…</HEAD>
<FRAMESET>
…
</FRAMESET>
</HTML>
```

Тег **<FRAMESET>** описує усі фрейми, на які поділяється екран. Можна розділити екран на кілька вертикальних або горизонтальних фреймів.

Параметр тега **<FRAMESET>** ROWS визначає горизонтальні підвікна, а COLS — вертикальні підвікна.

Описи підвікон розділяються комами.

Для опису підвікон використовується:

- просте числове значення, що визначає фіксовану висоту (ширину) підвікна в пікселях;

8*

- значення величини підвікна у відсотках від 1 до100;
- символ «*» вказує, що все решта місця належатиме цьому фрейму, цифра перед зірочкою вказує пропорцію для фрейму.

Приклади:

<FRAMESET COLS="50, *,50" > — описує 3 фрейми, 2 по 50 пікселів праворуч і ліворуч, і 1 усередині цих смуг.

<FRAMESET ROWS ="20%,3*,*" > — описує 3 фрейми, перший з яких займає 20 % площі вікна броузера зверху, другий — 3/4 місця, що залишилося від першого фрейму, тобто 60 % усієї площі вікна.

У тегу ***<FRAME>*** *задаються параметри для кожного фрейму окремо:*

параметр SRC задає ім'я файла, що завантажується у цьому фреймі;

параметр NAME задає ім'я фрейму. Ім'я фрейму може бути використане для визначення дії з фреймом з іншого HTML-документа або фрейму (як правило, із сусіднього фрейму того ж документа). Ім'я обов'язкове має починатися із символу.

Приклад:

```
<FRAME SRC ="homepage.htm" NAME="frame1">
```

Ім'я фрейму необхідно задавати для того, щоб згодом указати, до якого фрейму використовувати гіперпосилання. У документі HTML у цьому випадку, у тегу **<A HREF>** (опис гіперпосилання) має бути параметр TARGET. Цей параметр визначає фрейм, у якому вказується вміст вебсторінки.

Приклад:

```
<A HREF=vfile.htm" TARGET="frame1"> file </A>
```

У цьому випадку, після вибору гіперпосилання file, у фреймі з ім'ям frame1 буде показано вміст документа file.htm.

Приклад:

Створення двох фреймів, один із яких займає 20 % від площі вікна зверху, на другий приділяється решта частини.

```
<HTML>
<HEAD>
<TITLE> Опис фреймів </TITLE>
</HEAD>
<FRAMESET ROWS="20%,*">
<FRAME SRC="file.htm" NAME="FRAME1">
<FRAME SRC="home.htm" NAME="FRAME2">
</FRAMESET>
</HTML>
```


Результат такої розбивки на фрейми наведений на рисунку 3.1.

Верхній фрейм

Нижній фрейм

Рис. 3.1. Розбивка на фрейми

Таблиці

Таблиця *— один з найпоширеніших елементів HTML.* Таблиці застосовуються для створення структури сторінки, виведення в зручному вигляді текстової та іншої інформації.

Таблиця створюється за допомогою тегів **<TABLE>таблиця</TABLE>.** Атрибутами цього тега задається загальний вигляд таблиці:

Атрибути тега ***<TABLE>****:*

- ***align*=«left», «right», «center»** — вирівнювання таблиці по горизонталі. За замовчуванням таблиця вирівнюється по лівому краю сторінки;
- ***width*** — задання ширини таблиці. Якщо ширина не задана, броузер сам добирає оптимальну ширину таблиці;
- ***border*** — ширина границі таблиці в пікселях. За замовчуванням border = 0, тобто границі таблиці не відображаються;
- ***cellspacing*** — ширина вільного простору між комірками таблиці (за замовчуванням 2 піх);
- ***cellpadding*** — ширина вільного простору між вмістом комірка і її меж;

- ***bgcolor*** — колір тла таблиці;
- ***bordercolor*** — колір рамки таблиці (тільки для IE);
- ***bordercolordark*** — для тривимірних таблиць колір нижньої і правої частини рамки таблиці (тільки для IE);
- ***bordercolorlight*** — для тривимірних таблиць колір верхньої та лівої ділянки рамки таблиці (тільки для IE);
- ***background*** — зображення, виведене як тло таблиці;
- ***summary*** — коротка інформація про таблицю;
- ***clear*** — форматування тексту, розташованого в документі HTML після таблиці.

✓ «no» — текст розміщується відразу після таблиці;

✓ «left» — текст розміщується на першому рядку після таблиці. Цей рядок вирівнюється за лівим краєм;

✓ «right» — текст розміщується на першому рядку після таблиці. Цей рядок вирівнюється за правим краєм.

- ***frame*** — зовнішній вигляд рамки, розташованої навколо таблиці.

✓ «border» — рамка з усіх боків таблиці (за замовчуванням);

✓ «void» — рамка не відображається;

✓ «above» — відображається верхня рамка;

✓ «below» — відображається нижня рамка;

✓ «hsides» — відображається верхня і нижня рамка;

✓ «lhs» — відображається ліва рамка;

✓ «rhs» — відображається права рамка;

✓ «vsides» — відображається ліва і права рамка;

✓ «box» — рамка відображається тільки з зовнішньої сторони таблиці.

- ***nowrap*** — не виконується перенесення рядків, якщо вони не містяться по горизонталі у вікні броузера.
- ***rules*** — зовнішній вигляд ліній, що розділяють комірки таблиці.

✓ «none» — розділові лінії між комірками таблиці не відображаються;

✓ «groups» — відображаються горизонтальні розділові лінії між усіма групами таблиць, визначеними тегами <thread>, <tbody>, <tfoot> і <colgroup>;

✓ «rows» — відображаються горизонтальні розділові лінії між усіма рядками таблиці;

✓ «cols» — відображаються вертикальні розділові лінії між усіма стовпчиками таблиці;

✓ «all» — відображаються всі лінії між усіма стовпчиками і рядками таблиці.

Заголовок таблиці

Заголовок створюється за допомогою тегів **<CAPTION>заголовок</CAPTION>**

Атрибути заголовка: ***align*=«left», «right», «center», «bottom».** Вирівнювання заголовка за замовчуванням, заголовок вирівнюється по центру. Якщо вказати align=«bottom», заголовок буде виведено внизу таблиці.

Приклад:

Текст програми:

```
<TABLE border="1" width="100%" bgcolor="teal">
<CAPTION align="left"> заголовок таблиці </CAPTION>
<TR>
<TD>комірка1,1</TD>
<TD>комірка1,2</TD>
</TR>
</TABLE>
```

Результат виконання...

заголовок таблиці	
комірка1,1	Комірка 1,2

Рядок таблиці

Рядок створюється за допомогою тегів **<TR>**рядок**</TR>**

Атрибути рядка:

***align* = «left», «center», «right», «justify», «char»** — горизонтальне вирівнювання даних у комірці:

- «left» — вирівнювання за лівим краєм;
- «center» — вирівнювання за центром;
- «right» — вирівнювання за правим краєм;
- «justify» — вирівнювання за шириною комірки;
- «char» — вирівнювання за зазначеним символом (за замовчуванням символ десяткової точки для поточної мови;

***valign* = «top», «middle», «bottom», «baseline»** Вертикальне вирівнювання даних в комірках рядка, де:

- «top» — вирівнювання за верхнім краєм;
- «middle» — вирівнювання за центром;
- «bottom» — вирівнювання за нижнім краєм;

• «baseline» — вирівнювання за базовою лінією, спільною для всіх комірок;

✓ ***bgcolor*** — колір тла рядка.

✓ ***bordercolor*** — колір рамки рядка (тільки для IE).

✓ ***background*** — зображення, виведене як тіло рядка.

Комірка таблиці

Комірка створюється за допомогою тегів **<TD></TD>** або тегів **<TH></TH>**.

Текст знаходиться між тегами **<TH></TH>**, відображається жирним шрифтом, тобто **<TH>текст</TH>** рівносильний **<TD>текст</TD>**

Атрибути комірки: містять у собі атрибути для рядка й додаткові:

• ***rowspan*** = **«число»** — об'єднання зазначеної кількості рядків.

• ***colspan*** = **«число»** — об'єднання зазначеної кількості стовпчиків.

• ***width*** — ширина комірки, у пікселях.

• ***height*** — висота комірки, у пікселях.

• ***nowrap*** — відключення автоматичної розбивки тексту для цієї комірки.

Щоб вивести порожню комірку, необхідно ввести в неї закодований пробіл

Приклад:

Текст програми:

```
<TABLE border="10" width="100%"
bordercolordark="red" bordercolorlight="brown">
<CAPTION>приклад різнобарвної таблиці</CAPTION>
<TR bgcolor="white">
<TD>комірка1,1</TD>
<TD>комірка1,2</TD>
</TR>
<TR>
<TD bgcolor="gray">комірка2,1</TD>
<TD>комірка2,2</TD>
</TR>
</TABLE>
```


Результат виконання:

приклад різнобарвної таблиці

комірка1,1	комірка1,2
комірка2,1	комірка2,2

Текст програми:

```
<TABLE width="100%" bgcolor="teal" border="1">
<CAPTION>приклад об'єднання комірок</CAPTION>
<TR bgcolor="white">
<TD colspan="2"> </TD>
<TD> </TD>
<TD rowspan="3"> </TD>
</TR>

<TR bgcolor="brown">
<TD rowspan="2"> </TD>
<TD> </TD>
</TR>

<TR>
<TD> </TD>
<TD> </TD>
</TR>

</TABLE>
```

Результат виконання:

приклад об'єднання комірок

colspan=2 rowspan=1		colspan=1 rowspan=1	
colspan=1 rowspan=2	colspan=1 rowspan=1	colspan=1 rowspan=1	colspan=1 rowspan=3
	colspan=1 rowspan=1	colspan=1 rowspan=1	

Посилання в документах HTML

Посилання створюється тегом **<A>**посилання**</A>**

Атрибути тега <A>:	
href=«URL»	URL-адреса об'єкта посилання
name=«ім'я посилання»	Ім'я посилання в документі. Використовується для організації посилань усередині того самого документа HTML
target	Ім'я вікна, у яке має бути завантажено документ. Може набувати значення: • _blank — документ буде завантажений у нове вікно броузера; • _parent — документ буде завантажений у вікно, що є первинним стосовно поточного; • _self — документ буде завантажений у те саме вікно, де розташоване посилання; • _top — при використанні фреймів документ займе усе вікно броузера
title	Назва, що з'явиться внизу броузера при наведенні миші на посилання

Внутрішні посилання

Припустимо, ваш сайт має таку структуру каталогів:

folder1
|
file.htm
folder2
|
file.htm
... index.htm

• Щоб задати посилання на документ file.htm, що знаходиться в папці folder1, з документа index.htm, слід вказати:

```
<a href="folder1/file.htm">посилання на folder1/
file.htm</a>
```

• Щоб задати посилання на документ file.htm, що знаходиться в папці folder2, з документа index.htm і зробити так, щоб він відкрився в новому вікні броузера, слід вказати:


```
<a href="folder2/file.htm" target="_blank·
>посилання на folder2/file.htm</a>
```

• Щоб задати посилання з документа file.htm, що знаходиться в папці folder1, на документ index.htm, слід вказати:

```
<a href="../index.htm">посилання на index.htm</a>
```

У цьому випадку команда ../ указує серверові перейти в батьківський щодо поточного каталог. Відповідно команда ../../ укаже серверові зробити два переходи вниз.

• Щоб задати посилання на документ file.htm, що знаходиться в папці folder2, з документа file.htm, що знаходиться в папці folder1, слід вказати:

```
<a href="../folder2/file.htm">посилання на folder2/
file.htm з folder1/file.htm</a>
```

Посилання в межах одного документа

У випадку, якщо у вас є великий за обсягом документ, для навігації по ньому буває зручно вказати посилання всередині цього документа.

Наприклад, ваш документ містить три розділи зі змістом.

```
<!– зміст –>
<a href="#chapter1">Перший розділ </a>
<a href="#chapter1">Другий розділ </a>
<a href="#chapter1">Третій розділ </a>

<!– текст –>
<p><a name="chapter1">Перший розділ </a>
. . .
Зміст першого розділу
. . .
<p><a name="chapter2">Другий розділ </a>
. . .
Зміст другого розділу
. . .
<p><a name="chapter3">Третій розділ </a>
. . .
Зміст третього розділу
. . .ї
```

У цьому випадку тегом **name** ми задаємо ім'я для кожного розділу (chapter1, chapter2, chapter3) і посилаємося на ці імена на початку документа. Коли відвідувач клацає мишею на посилання, броузер автоматич-

но прокручує сторінку на те місце, де розташоване відповідне ім'я.

Зовнішні посилання

Для задання зовнішнього посилання атрибуту **href** задається протокол, шлях і, якщо необхідно, порт необхідного ресурсу у такому вигляді:

```
<a href="протокол://шлях/:порт>
```

Значення параметра атрибута href	
href=«http://...»	Посилання на об'єкт, що буде передаватися з використанням протоколу HTTP. Може бути будь-яким довільним об'єктом
href=«ftp://...»	Посилання на FTP-сервер
href=«mailto:...»	Посилання на адресу електронної пошти. При виборі цього посилання завантажується поштова програма

Приклади посилань:

```
<a href=«http://www.dlab.kiev.ua/soft/
test.zip">завантажити test.zip</a>
<a href=«mailto:vova@dlab.kiev.ua">написати мені
лист</a>
<a href=«http://www.dlab.kiev.ua/»
target="_blank">відкрити головну сторінку сайту в
новому вікні</a>,
```

де останній символ «/» в адресі www.dlab.kiev.ua/ вказує серверу, що треба зайти в кореневий каталог сервера www.dlab.kiev.ua і завантажити початкову сторінку.

розділ **4**

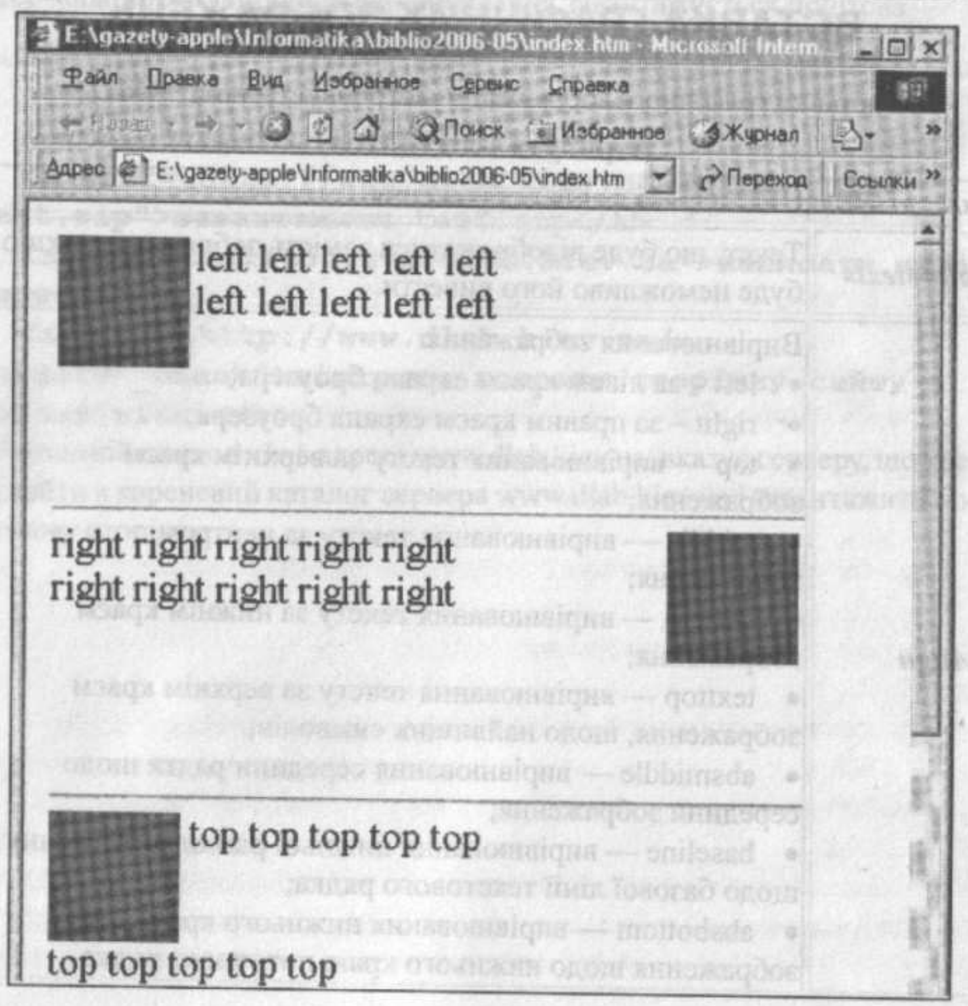
ГРАФІКА І ЗВУК У ДОКУМЕНТАХ HTML

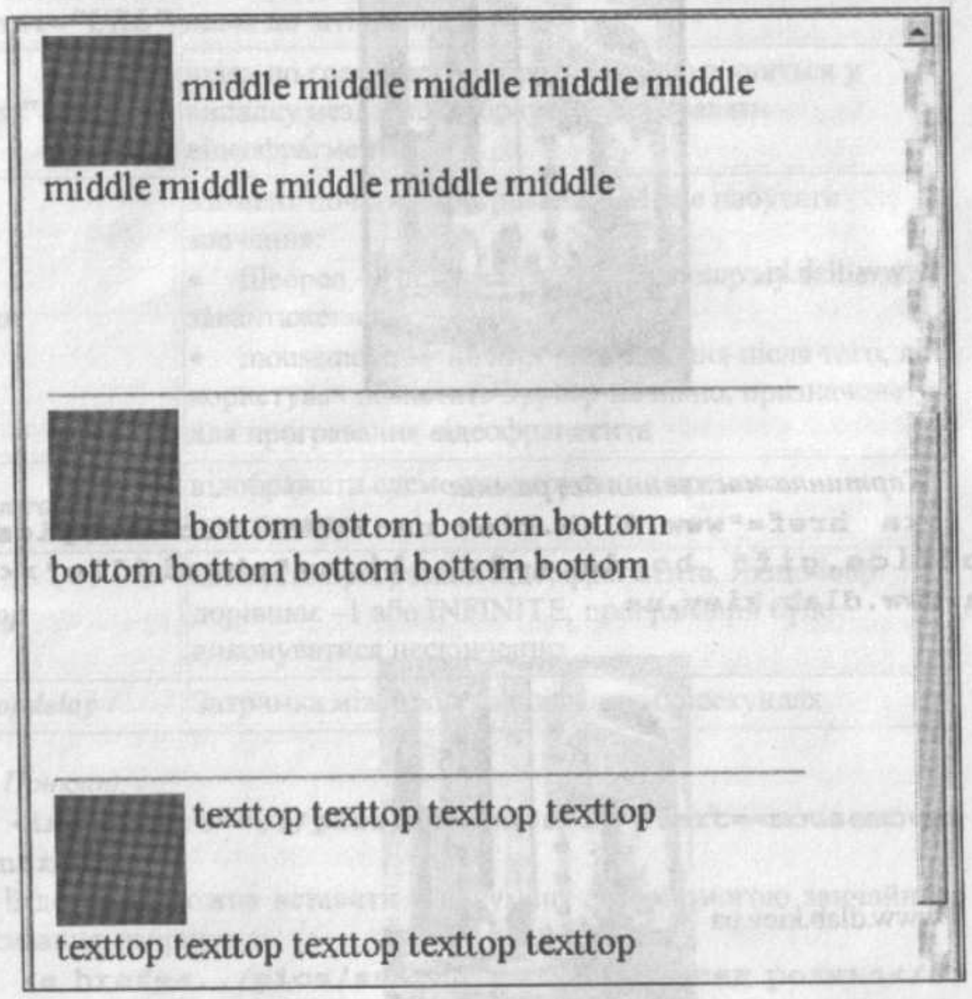
ВСТАВКА ГРАФІЧНИХ ЗОБРАЖЕНЬ

Для вставки графічного зображення застосовується тег <IMG>

Атрибути тега <IMG>	
src=«URL»	URL-адреса файла зображення
alt=«text»	Текст, що буде відображатися замість зображення, якщо буде неможливо його вивести.
Align	Вирівнювання зображення: • left – за лівим краєм екрана броузера; • right – за правим краєм екрана броузера; • top — вирівнювання тексту за верхнім краєм зображення; • middle — вирівнювання тексту за центром зображення; • bottom — вирівнювання тексту за нижнім краєм зображення; • texttop — вирівнювання тексту за верхнім краєм зображення, щодо найвищих символів; • absmiddle — вирівнювання середини рядка щодо середини зображення; • baseline — вирівнювання нижньої рамки зображення щодо базової лінії текстового рядка; • absbottom — вирівнювання нижнього краю зображення щодо нижнього краю поточного рядка

Атрибути тега <IMG>	
Height	Висота картинки в пікселях
Width	Ширина картинки в пікселях
border	Ширина рамки навколо картинки в пікселях (тільки NN)
hspace	Ширина вільного простору в пікселях, що має відокремлювати зображення від тексту по горизонталі
vspace	Ширина вільного простору в пікселях, що має відокремлювати зображення від тексту по вертикалі
usemap	URL-адреса файла карти зображення
ismap	Указує, що зображення є картою





Графічне зображення-посилання

Якщо вставити картинку між тегами **<A></A>**, то картинка стане посиланням. Такий вид посилання дуже часто використовують у створенні мапи сайту, організації меню або організації посилання на будь-яку інформацію через піктограму. У цьому випадку броузер обведе картинку рамкою, видалити яку можна, вказавши значення атрибута border=«0» тега **<IMG>**.

Приклад:

Картинка-посилання з рамкою:

```
<a href="www.dlab.kiev.ua"><IMG src=v../pics/
office.gif" align="absmiddle"></a> www.dlab.kiev.ua
```

www.dlab.kiev.ua 

Картинка-посилання без рамки:

```
<a href="www.dlab.kiev.ua"><IMG src="../pics/
office.gif" border=«0» align="absmiddle"></
a>www.dlab.kiev.ua
```

www.dlab.kiev.ua 

Малюнок як задній фон (шпалери, background)

Зробити задній фон сторінки можна за допомогою атрибута background тега **<body>**, наприклад:

```
<body background=«bgr.gif»>
```

Вставка відео в документ HTML

За допомогою тега **<IMG>** можна вставити в документ відеофрагмент у форматі *.avi*. Як правило, у таких випадках малюнок має зображувати рупор або щось подібне, наприклад: 

Атрибути тега <IMG> для вставки *.avi*-файла	
dynsrc="URL"	шлях до .avi-файла
src="URL"	шлях до графічного зображення, що з'явиться у випадку нездатності браузера програвати відеофрагменти
start	Момент початку програвання. Може набувати значення: • fileopen — почати програвання відразу ж після завантаження; • mousemove — почати програвання після того, як користувач помістить курсор на вікно, призначене для програвання відеофрагмента
controls	відображати елементи керування процесом програвання
loop	кількість програвань відеофрагмента. Якщо loop дорівнює –1 або INFINITE, програвання буде виконуватися нескінченно
loopdelay	Затримка між програваннями в мілісекундах

Приклад:

```
<img dynsrc=«../pics/search.avi» start=«mousemove»
controls>
```

Відеофайл можна вставити в документ за допомогою звичайного посилання, наприклад:

```
<a href=«../pics/search.avi»>перегляд ролика</a>
```

У цьому випадку при натисканні на посилання броузер завантажить плагін для відтворення відео і запуску ролика.

ВСТАВКА ЗВУКУ В ДОКУМЕНТ HTML

Щоб вставити звуковий файл у документ, застосовують теги:

- для MSIE — **<bgsound>** з атрибутами:
  - balance — керування стереобалансом. Припустимі значення від –10 000 до 10 000;
  - volume — керування гучністю звучання. Припустимі значення від –10 000 до 0 (максимальна гучністю);

– loop — скільки разів програвати файл. Якщо loop-»infinite» або «–1», програвання буде виконуватися нескінченно.

src — шлях до файла.

- для NN — **<embed>** з атрибутами:

– loop — повторювати спочатку (true — так, false — ні);

– play_loop=«число» -якщо повторювати спочатку, то скільки разів;

– src — шлях до *.mid*, *.wav* або *.avi* файла;

– autostart — програвати одразу після завантаження (true — так, false — ні);

– hidden — сховати пульт керування (true — так, false — ні);

– width — ширина пульта керування;

– height -висота пульта керування.

Взагалі, **<embed>** призначений для вставки об'єктів з використанням технології OLE, що застосовуються в Windows-системах. В інших ОС цей тег не працюватиме.

У користувачів виникають запитання: який тег використовувати? Користуйтесь одразу обома.

```
<embed src="bgmusic.mid» autostart="true" hidden="true" loop="0">
  <bgsound src="bgmusic.mid" loop="infinite">
```

розділ 5

ВИКОРИСТАННЯ JAVASCRIPT НА ПРАКТИЦІ

***JavaScript** — мова для складання скриптів, розроблена компанією Netscape*. Компанія Netscape поширила в 1995 році JavaScript як механізм керування сторінками на стороні клієнта.

Нині JavaScript — найпопулярніша скриптова мова в Інтернеті, що підтримується всіма основними броузерами: Internet Explorer, Mozilla, Firefox, Netscape, Opera.

JavaScript застосовується в мільйонах Web-сторінок для поліпшення дизайну, перевірки форм, розпізнання типів та версій броузерів тощо.

JavaScript нескладно вивчити!

JAVASCRIPT: НЕОБХІДНО ЗНАТИ

- JavaScript було розроблено, щоб додати інтерактивності HTML-сторінкам;
- JavaScript — це полегшений варіант мови програмування;
- JavaScript являє собою рядки виконуваного коду;
- JavaScript можна вставляти в HTML-сторінки;
- JavaScript — відкрита для використання скриптова мова, нею може користуватися кожен без ліцензії.

Використання JavaScript

JavaScript може вміщувати в HTML-сторінку текст, що динамічно змінюється.

Директива мовою JavaScript:

document.write(?<h1>? + name + ?</h1>?)

- Може виводити при показі HTML-сторінки змінний текст так, як це статично робиться в HTML: **<h1>Опанас Приходько</h1>**.
- JavaScript може реагувати на події. Можна зробити так, що JavaScript буде виконуватися, наприклад, після того, як сторінка закінчує завантажуватися, або після того, як користувач клікнув якийсь HTML-елемент.
- JavaScript може зчитувати й вписувати HTML-елементи
- JavaScript може зчитувати HTML-елементи й змінювати вміст HTML-елементів.

JavaScript можна застосовувати для оцінки даних.

JavaScript можна застосовувати для оцінки введених у форму даних перш ніж вони відправляться на сервер. Ця функція особливо добре пристосована для того, щоб заощаджувати обчислювальні ресурси сервера.

ВМІЩЕННЯ JAVASCRIPT В HTML-СТОРІНКИ

JavaScript уміщується в HTML-сторінку. Код JavaScript уміщується безпосередньо в тексті самої HTML-сторінки.

Для вставки скритпа в HTML-документ застосовується тег

```
<script>
```

А закінчується скрипт так:

```
</script>
```

Розглянемо наступний простий приклад:

```
<html>
<body>
<br>
Звичайний текст. <br>
<script language="JavaScript">
document.write("Цей текст створений за допомогою
JavaScript!")
</script>
<br>
Знову звичайний текст.
</body>
</html>
```

Щоб бачити, як цей скрипт працює, запишіть приклад як звичайний файл HTML і завантажте його в броузер, що має підтримку мови JavaScript.

Ця процедура дасть результат:

```
Звичайний текст.
Цей текст створений за допомогою JavaScript!
Знову звичайний текст.
```

А як виглядатиме наша сторінка, якщо броузер не розуміє JavaScript?

Броузери, що не мають підтримки JavaScript, «не знають» і тега **<script>**. Вони ігнорують його й друкують всі наступні коди як звичайний текст. Іншими словами, читач побачить, як код JavaScript, наведений у програмі, виявиться вписаний відкритим текстом прямо всередині HTML-документа.

Щоб цього не відбулося, використайте HTML-тег коментаря:

```
<script language=«JavaScript»>
<!-
   рядки JavaScript
//->
</script>
```

Іноді потрібно застосовувати той самий скрипт на кількох сторінках, для цього доводиться писати скрипт на кожній сторінці.

JavaScript розміщується в зовнішньому файлі

Можна розмістити скрипт у зовнішньому файлі з розширенням .js. Наприклад, ми маємо скрипт: document.write («Використаємо зовнішній файл»).

Збережемо цей скрипт як файл my.js. При цьому в тексті файла не повинно бути тегу **<script>.**

Тепер ви можете викликати цей скрипт із будь-якої частини сторінки, використовуючи атрибут **src**:

```
<html>
<body>
<script src=«my.js»>
</script>
</body>
</html>
```

ОСНОВНІ ПОНЯТТЯ JAVASCRIPT

Змінні

Змінні використовуються для зберігання даних.

Змінні — це так звані контейнери, у яких можна зберігати інформацію. Значення змінної може змінюватися.

Оголошення змінної

Змінна створюється за допомогою ключового слова var:

var my_name = «user»

Приклад:

```
<html>
<body>
<script language="JavaScript">
var
my_name = "user"
my_age=16
document.write(name)
document.write("<h1>" +  my_name  + "</h1>");
document.write("<h1>" +  my_age  + "</h1>")
</script>
</body>
</html>
```

Арифметичні оператори

Оператор	Опис	Приклад	Результат
+	Додавання	2+2	4
–	Віднімання	5–2	3
*	Множення	4*5	20
/	Ділення	5/2	2.5
%	Повертає залишок від ділення	5%2	1
++	Інкремент	x=5; x++	6
– –	Декремент	x=5; x– –	4

Логічні оператори

Оператор	Опис	Приклад	Результат
&&	and	x=6; y=3; (x < 10 && y > 1)	true
||	or	x=6; y=3; (x==5 || y==5)	false
!	not	X=6; y=3; x != y	true

Оператори порівняння

Оператор	Опис	Приклад	Результат
==	Дорівнює	5==8	false
!=	Не дорівнює	5!=8	true
>	Більше, ніж	5>8	false
<	Менше, ніж	5<8	true
>=	Більше або дорівнює	5>=8	false
<=	Менше або дорівнює	5<=8	true

Скорочений запис операторів

Оператор	Приклад	Те саме, що й..
+=	x+=y	x=x+y
–=	X–=y	x=x–y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Рядкові оператори

Рядок найчастіше є текстом, наприклад, «Який сьогодні гарний день!». Щоб склеїти дві або більше рядкових змінних, використайте оператор + :

```
txt1=«Який сьогодні»
txt2=«гарний день!»
txt3=txt1 + txt2
```

Тепер змінна txt3 має значення «Який сьогодні гарний день!»

Функції

Функція містить деякий код, що виконується з появою якоїсь події або при виклику цієї функції. Функція складається з набору виразів. Ви можете багаторазово застосовувати функції в тому самому скрипті або в інших документах. Ви визначаєте функції на початку файла (у розділі HEAD) і далі викликаєте їх з наступних частин коду. Тепер час ознайомитися з тим, як викликати повідомлення, що вискакує (alert-box).

Цей метод JavaScript застосовується для видачі повідомлень користувачеві.

```
Alert («тут є текст повідомлення»).
```

Як визначити функцію

Для того, щоб створити функцію, ви визначаєте її ім'я, деякі значення («аргументи») і набір виразів:

```
function myfunction (argument1,argument2,etc)
{
набір виразів
}
```

Навіть якщо у функції немає аргументів, вона повинна бути написана з парою круглих дужок:

```
function myfunction()
{
набір виразів
}
```

Аргументи — це змінні, які будуть використовуватися у функції. Значення цих змінних при виклику функції передаються їй.

Розміщуючи функції в розділі HEAD документа, ви даєте змогу коду функцій завантажитися раніше, ніж вони будуть викликатися.

Деякі функції повертають певні структури:

```
function result(a,b)
{
c=a + b
return c
}
```

Як викликати функцію

Функція не виконується доти, доки не буде викликана.

Ви можете викликати функцію, що містить аргументи:

```
myfunction(argument1,argument2,etc)
```

або без аргументів:

```
myfunction()
```

Вираз return

У функції, що повертає деяке значення, має застосовуватися вираз «return». Цей вираз визначає значення, що буде передано туди, звідки була викликана функція. Наприклад, якщо ваша функція повертає суму двох чисел:

```
function total(a,b)
{
result=a + b
return result}
```


коли ви викликаєте цю функцію, ви повинні передати їй два аргументи:

```
sum=total(2,3)
```

Значення, що повертає функцією, (5) буде збережено в змінній sum.

Умовні вирази

Умовні вирази в JavaScript застосовуються для організації виконання кількох різних дій залежно від виконання різних умов.

Вираз if (якщо)

Застосовуйте цей вираз, коли ви хочете, щоб певний фрагмент коду виконувався тільки за виконання якоїсь умови:

```
<html>
<body>
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time < 10)
{
document.write("<b> Добрий ранок </b>")
}
</script>

<p>
Цей приклад демонструє вираз if.
</p>
<p>
Якщо часу у броузері менше, ніж 10, то ви отримає-
те привітання "Добрий ранок ".
</p>
</body>
</html>
```

Результат:

"Добрий ранок"
Цей приклад демонструє вираз if.
Якщо часу у броузері менше, ніж 10, то ви отримаєте привітання "Добрий ранок".

Вираз if...else (якщо...інакше)

Застосовуйте цей вираз, коли хочете, щоб один фрагмен коду виконувався за виконання якоїсь умови, а інший — за його не виконання:

```
<html>
<body>
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time < 10)
{
document.write("<b> Добрий ранок </b>")
}
else
{
document.write("<b> Добрий день </b>")
}
</script>
<p>
Цей приклад демонструє вираз If...Else.
</p>
<p>
Якщо часу у вашому броузері менше, ніж 10, ви отримаєте привітання "Добрий ранок". У протилежному випадку ви отримаєте привітання "Добрий день".
</p>
</body>
</html>
```

Результат:

Добрий ранок

Цей приклад демонструє вираз If...Else.

Якщо час у вашому броузері менше, ніж 10, то ви отримаєте вітання "Добрий ранок" У протилежному випадку ви отримаєте привітання "Добрий день".

Вираз for (організації циклу)

Вираз організації циклу використовується для виконання певного фрагмента коду задану кількість разів.

Організація циклу на основі виразу **for**. Використайте цей вираз для виконання заданого фрагмента коду певну кількість разів.

```
<html>
<body>
<script type=«text/javascript»>
for (i = 0; i <= 5; i +  + )
{
document.write("Число дорівнює "  +  i)
document.write(«<br>«)
}
</script>

<p> Пояснення:</p>
<p> Цикл починається при i=0.</p>
<p> Поки <b>i</b> — менше або дорівнює 5, цикл продовжить виконуватися</p>
<p>
<b>i</b> збільшується на 1 за кожне виконання циклу
</p>
</body>
</html>
```

Результат:

Число дорівнює 0
Число дорівнює 1
Число дорівнює 2
Число дорівнює 3
Число дорівнює 4
Число дорівнює 5

Пояснення:

Цикл починається при i=0.

Поки **i** — менше або дорівнює 5, цикл продовжить виконуватися.

i збільшується на 1 за кожне виконання циклу.

Вираз while

Організація циклу на основі виразу **while**. Застосовуйте цей вираз для виконання елементу коду доти, доки не буде виконуватися деяка умова.

```
<html>
<body>
<script type="text/javascript">
i = 0
while (i <= 5)
{
document.write("Число дорівнює " + i)
document.write("<br>")
i + +
}
</script>
<p> Пояснення:</p>
<p><b>i</b> дорівнює 0.</p>
<p> Поки <b>i</b> – менше або дорівнює 5,
цикл продовжить виконуватися.</p>
<p><b>i</b> збільшується на 1 за кожне
виконання циклу.</p>
</body>
</html>
```

Результат:

Число дорівнює 0
Число дорівнює 1
Число дорівнює 2
Число дорівнює 3
Число дорівнює 4
Число дорівнює 5

Пояснення:
i дорівнює 0
Поки **i** менше або дорівнює 5, цикл продовжить виконуватися.
i програм збільшується на 1 за кожне виконання циклу.

Вираз do while

Організація циклу на основі виразу **do while**. Застосовуйте цей вираз для організації циклічного виконання фрагмента коду доти, доки дотримується певна умова. Заданий у виразі фрагмент коду буде виконуватися хоча б один раз, навіть якщо умова не дотримується, оскільки він виконується до того, як перевіряється умова.

```
<html>
<body>
<script type="text/javascript">
i = 0
do
{
document.write("Число дорівнює " + i)
document.write("<br>")
i + +
}
while (i <= 5)
</script>

<p> Пояснення:</p>
<p><b>i</b> і дорівнює 0.</p>
<p> Цикл виконуватиметься </p>
<p><b>i</b> і збільшується на 1 за кожне виконання
циклу.</p>
<p> Поки <b>i</b> – менше або дорівнює 5, цикл
продовжить виконуватися.</p>
</body>
</html>
```

Результат:

Число дорівнює 0
Число дорівнює 1
Число дорівнює 2
Число дорівнює 3
Число дорівнює 4
Число дорівнює 5

Пояснення:
i дорівнює 0.
Цикл виконуватиметься
i збільшується на 1 за кожне виконання циклу.
Поки **i** менше або дорівнює 5, цикл продовжить виконуватися.

ОГЛЯД ВБУДОВАНИХ ОБ'ЄКТІВ JAVASCRIPT

Об'єкт Math

Метод	Повертає як результат
abs (x)	абсолютне значення *x*
acos (x)	аркосинус *x*
asin (x)	арксинус *x*
atan (x)	арктангенс *x*
atan2 (x, y)	кут від осі *x* до точки *a*
ceil (x)	найближче ціле число, більше, ніж *x*
cos (x)	косинус *x*
exp (x)	експоненту *x*
floor (x)	найближче ціле число, менше, ніж *x*
log (x)	натуральний логарифм *x*
max (x, y)	число, більше з пари *x* та *y*
min (x, y)	число, менше з пари *x* та *y*
pow (x, y)	*x* у степені *y*
random	випадкове число з проміжку від 0 до 1
round (x)	округлює *x* до найближчого цілого числа
sin (x)	синус *x*
sqrt (x)	квадратний корінь *x*
tan (x)	тангенс *x*

Об'єкт Boolean

Метод	Повертає як результат
ToString	булевське значення у вигляді рядка (true або false)
ValueOf	значення заданого об'єкта

Об'єкт String

Метод	Повертає як результат
Length	довжину рядка
anchor	рядок усередині тега <a>
bold	рядок
charAt	символ, що має заданий індексний номер
charCodeAt	Unicode символу із заданим індексним номером
concat	об'єднання двох рядків
fontColor	рядок
fontSize	рядок
fromCharCode	рядкове значення заданого символа Unicode
indexOf	індексний номер появи в рядку заданого символа або число -1, якщо символа в рядку немає
italics	рядок
lastIndexOf	як indexOf, тільки метод діє від кінця рядка
link	рядок
replace	Замінює задані символи іншими символами
search	ціле числове значення, якщо рядок містить задані символи, або -1, якщо не містить
slice	рядок, що містить заданий набір символів
small	рядок
split	замінює задані символи коми
strike	<strike>рядок</strike>
sub	_{рядок}
substr	задані символи
sup	^{рядок}
toLowerCase	рядок у нижньому регістрі
toUpperCase	рядок у верхньому регістрі

Об'єкт Array

Метод	Повертає як результат
length	кількість елементів у масиві
concat	масив — об'єднання двох інших масивів
join	рядок, що складається з усіх об'єднаних елементів масиву
reverse	масив, у якому порядок проходження елементів перевернутий
slice	задану частину масиву
sort	відсортований масив

Об'єкт Date

Метод	Повертає як результат
Date	об'єкт Date
getDate	день місяця об'єкта Date (від 1 до 31)
getDay	день тижня об'єкта Date (від 0 до 6.0=неділя, 1=понеділок, і т.д.)
getMonth	місяць об'єкта Date (від 0 до 11.0=січень, 1=лютий, і т.д.)
getFullYear	рік об'єкта Date (чотири цифри)
getYear	рік об'єкта Date (від 0 до 99). Використайте краще метод getFullYear!
getHours	годину об'єкта Date (від 0 до 23)
getMinutes	хвилину об'єкта Date (від 0 до 59)
getSeconds	секунду об'єкта Date (від 0 до 59)
getMilliseconds	мілісекунди об'єкта Date (від 0 до 999)
getTime	число мілісекунд, що минули з напівночі 1 січня 1970 року
getTimezoneOffset	різницю місцевого часу й Грінвіча
getUTCDate	день місяця об'єкта Date в універсальному часі (UTC)
getUTCDay	день тижня об'єкта Date в універсальному часі (UTC)
getUTCMonth	місяць об'єкта Date в універсальному часі (UTC)
getUTCFullYear	рік об'єкта Date в універсальному часі (UTC) (чотири цифри)
getUTCHourc	годину об'єкта Date в універсальному часі (UTC)

Метод	Повертає як результат
getUTCMinutes	хвилину об'єкта Date в універсальному часі (UTC)
getUTCSeconds	секунду об'єкта Date в універсальному часі (UTC)
getUTCMilliseconds	мілісекунди об'єкта Date в універсальному часі (UTC)
parse	рядок, що містить кількість мілісекунд минулих починаючи з 1 січня 1970 року (північ)
setDate	встановлює день місяця об'єкта Date (від 1 до 31)
setFullYear	встановлює рік об'єкта Date (чотири цифри)
setHours	встановлює годину об'єкта Date (від 0 до 23)
setMilliseconds	встановлює мілісекунди об'єкта Date (від 0 до 999)
setMinutes	встановлює хвилину об'єкта Date (від 0 до 59)
setMonth	встановлює місяць об'єкта Date (від 0 до 11.0=січень, 1=лютий, і т.д.)
setSeconds	встановлює секунду об'єкта Date (від 0 до 59)
setTime	встановлює кількість мілісекунд, що минули з 1 січня 1970 року
setYear	встановлює рік об'єкта Date (00-99)
setUTCDate	встановлює день місяця об'єкта Date в універсальному часі (від 1 до 31)
setUTCDay	встановлює день тижня об'єкта Date в універсальному часі (від 0 до 6.0=неділя, 1=понеділок, і т.д.)
setUTCMonth	встановлює місяць об'єкта Date в універсальному часі (від 0 до 11.0=січень, 1=лютий, і т.д.)
setUTCFullYear	встановлює рік об'єкта Date в універсальному часі (чотири цифри)
setUTCHour	встановлює годину об'єкта Date в універсальному часі (від 0 до 23)
setUTCMinutes	встановлює хвилину об'єкта Date в універсальному часі (від 0 до 59)
setUTCSeconds	встановлює секунду об'єкта Date в універсальному часі (від 0 до 59)

Метод	Повертає як результат
setUTCMilliseconds	встановлює мілісекунди об'єкта Date в універсальному часі (від 0 до 999)
toGMTString	перетворює об'єкт Date на рядок, встановлює його на час за Грінвічем
toLocaleString	перетворює об'єкт Date на рядок, установлює його на місцевий час
toString	перетворює об'єкт Date на рядок

DHTML ТА ОБ'ЄКТНА МОДЕЛЬ INTERNET EXPLORER

За JavaScript, усі елементи на web-сторінці являють собою ієрархічну структуру. Кожен елемент з'являється у вигляді об'єкта. І кожен такий об'єкт може мати певні властивості й методи. У свою чергу, мова JavaScript дає змогу легко керувати об'єктами web-сторінки. Для правильної роботи дуже важливо розуміти ієрархію об'єктів, на які спирається розмітка HTML.

Як приклад розглянемо загальну структуру об'єктної моделі Internet Explorer версії 4 і вище.

window — об'єкт, що становить вікно броузера:

- frames — фрейми у вікні;
  - window;
  - ...;
  - window ;
- document — HTML-документ;
  - all — повна колекція елементів документа;
  - forms — колекція форм;
  - anchors — колекція якорів;
  - embeds — колекція впроваджених об'єктів;
  - filters — колекція фільтрів;
  - images — колекція зображень;
  - links — колекція посилань;
  - scripts — колекція скриптових блоків;
  - selection — колекція виділених елементів;
  - stylesheets — колекція об'єктів з індивідуальними стилями;
- event — об'єкт, що дає доступ до подій;
- history — об'єкт, що дає доступ до історії відвіданих посилань;

- navigator — об'єкт, що дає доступ до характеристик броузера;
- location — об'єкт, що містить поточний URL;
- screen — об'єкт, що дає доступ до характеристик екрана.

Структура об'єктної моделі досить складна, але чітко визначена. Існує однозначний спосіб доступу до будь-якої властивості або методу. Синтаксис повністю відповідає тому, що використовується в об'єктних мовах. Тобто для доступу до конкретної властивості потрібно створити рядок доступу. Для цього й використовується скрипт, що дає змогу працювати з об'єктами.

DHTML І JAVASCRIPT (7 ПРАКТИЧНИХ ПРИКЛАДІВ)

Що таке динамічний HTML (DHTML)?

***DHTML** — технологія для роботи з об'єктною моделлю броузера шляхом використання деякої скриптової мови* (JavaScript або VBScript). Ми розглядаємо JavaScript, тому що він більше поширений і функціональний (підтримує визначення користувачем типів), підтримується всіма основними броузерами і нагадує C ++ , у такий спосіб його легко вивчити.

Виникає запитання: чому недостатньо тегів HTML?

Відповідь проста: теги — простіший, але менш ефективний спосіб для того, щоб працювати з DOM (об'єктною моделлю документа). JavaScript набагато гнучкіший.

Відмінності нескладно помітити на перших двох прикладах.

Приклад 1: використання тегів для задання title

```
<HTML>
<HEAD>
<TITLE>приклад</TITLE>
</HEAD>
</HTML>
```

Загальноприйнятний спосіб – це використання **<TITLE>**. Тепер використаємо JavaScript для одержання того самого результату.

Приклад 2: title на основі JavaScript

```
<SCRIPT language=JavaScript>
document.title=' приклад '
</SCRIPT>
```

Обидва способи здійснюють доступ до властивості document.title, але JavaScript — ефективніший. Наприклад, він може встановити title відповідно до роздільної здатності екрана.

Приклад 3: динамічний вибір title

```
<SCRIPT language=JavaScript>
if (window.screen.width==640)
document.title="640x480 версія!"
if (window.screen.width==800)
document.title="800x600 версія!"
if (window.screen.width==1024)
document.title="1024x768 версія!"
else document.title="Інша роздільна здатність!"
</SCRIPT>
```

Window.screen.width повертає значення, яке дорівнює ширині екрана клієнта.

А тепер зробимо годинник у рядку стану броузеру:

Приклад 4: рядок стану

```
<SCRIPT language=JavaScript>
function time()
{
var d = new Date()
var result = "Місцевий час: "
result + = d.getHours() + " : "
result + = d.getMinutes() + " : "
result + = d.getSeconds()
window.status = result
setTimeout(«time()», 1000))
}
</SCRIPT>
<HTML>
<HEAD>
<TITLE>Годинник у рядку стану</TITLE>
</HEAD>
<BODY onload = time()>
</BODY>
</HTML>
</SCRIPT>
```

Функція time() викликається у момент завантаження сторінки, тому що це задано відповідною директивою <BODY onload = time()>. Використаємо onmouseover-подію для того, щоб усе стало більш зрозумілим.

Приклад 5: обробка події Onmouseover

```
<SCRIPT language=JavaScript>
function warning()
{
alert("OnMouseOver! Warning() було викликано!")
}
</SCRIPT>
<HTML>
<BODY>
<a href="" onmouseover=warning()>Наведіть вказівник мишки на посилання!</a>
</BODY>
</HTML>
```

Об'єкт Window має дуже зручний метод Navigate(url), який перенаправляє броузер до URL.

Приклад 6: Window.navigate

```
<SCRIPT language=JavaScript>
function go()
{
if(window.screen.width<=800)
window.navigate("low_res.htm")
else window.navigate("hi_res.htm")
}
</SCRIPT>
<HTML>
<BODY onload=go()>
</BODY>
</HTML>
```

Розглянемо скрипт для відображення налаштувань дисплея:

Приклад 7: настройки дисплея

```
<SCRIPT language=JavaScript>
function getdesktop()
{
var x = window.screen.width
var y = window.screen.height
var c = window.screen.colorDepth
x.toString()
y.toString()
```

```
  c.toString()
  window.alert ("Ваші настройки дисплею: " + x +
"x" + y + "x" + c + " bpp!")
  }
  </SCRIPT>
  <HTML>
  <HEAD>
  <TITLE> настройки дисплея </TITLE>
  </HEAD>
  <BODY>
  <FORM>
  <INPUT type = button value = "Подивитись на-
стройки дисплея " onclick = getdesktop()></INPUT>
  </FORM>
  </BODY>
  </HTML>
```

КОРИСНІ ПОСИЛАННЯ

Ви можете знайти дискусії, присвячені JavaScript, онлайнові навчальні програми, посилання, приклади коду й сотні корисних програм на сайтах.

Англійською:

- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dninvbs/html/javascript.asp — Microsoft's JScript, DHTML и т.д.
- http://developer.mozilla.org/en/docs/JavaScript
- http://www.w3schools.com/js/
- http://www.javascripter.net/faq/index.htm
- http://javascript.internet.com — JavaScript Source: Free JavaScripts, Tutorials, Example Code, Reference, Resources, And Help
- http://www.WebReference.com/JavaScript/ — tutorials and in-depth discussions
- http://www.JavaScripts.com — lots of scripts and code examples
- http://www.irt.org — a FAQ collection

Російською:

- http://javascript.fud.ru/docum_javascript/docum/ Документація та підручники по JavaScript
- http://gnezdo.webscript.ru/links/e_javascript/ JavaScript та DHTML скрипти, документація, посилання
- http://www.compdoc.ru/internet/java/ — Документація по JavaScript

розділ 5 КАСКАДНІ СТИЛІ CSS

Насамперед каскадні стилі забезпечують якісне форматування веб-сторінки. До того ж, вони допомагають централізовано контролювати зовнішній вигляд усіх сторінок вашого сайту.

Отже, стилі допомагають нам:

- розділити форму сторінки та її структуру;
- простіше керувати форматуванням;
- швидко формувати нові сторінки;
- змінювати водночас кілька сторінок.

Розглянемо переваги каскадних стилів.

Поділ на форму та структуру

Мова HTML не створювалася як засіб керування формою веб-сторінки. Це мова, яка лише визначає структуру сторінки: вона вказує броузеру, як повинні відображатися елементи.

Але це не влаштовувало розробників. Фірма Netscape винайшла нові теги HTML, що дають змогу краще контролювати веб-сторінки. Для форматування стали застосовуватися теги `<p>`, `<font>`, `<i>`. Потім, коли вимоги до оформлення сторінок зросли, треба було використати громіздкі таблиці для правильного розміщення елементів на сторінці, що у свою чергу істотно сповільнювало її завантаження.

Стилі впоралися з цією проблемою: з їхньою допомогою стало можливим відокремити частину, що визначає структуру частини, що визначає форму. HTML залишається чистим, а код CSS завантажується з окремого файла.

Основи керування форматуванням

Безумовно, тег **<Font Size>** дає змогу нам міняти розмір тексту, а таблиці допомагають створювати поля. Але можливості HTML обмежені. Стилі запобігають нагромадженню тегів сторінки, що ускладнює HTML-код. Зменшення обсягу коду приводить до зменшення кінцевого файла.

Ви можете просто створювати сторінки

***Стилі** — це простий текст, заснований на HTML.* За допомогою стилів ви можете робити речі, для яких раніше використовували графіку.

Підтримувати й змінювати кілька сторінок одночасно

Раніше, якщо було потрібно змінити або скоригувати шрифт на веб-сторінці, потрібно було вручну редагувати кожен документ. Навіть якщо сторінка обслуговується через базу даних, усе ще необхідно було коригувати всі документи, і в кожному них змінювати відповідні рядки з тегом <Font Face>.

За допомогою стилів стало можливо об'єднати керування всіма документами в одному загальному файлі CSS. Якщо потрібно внести зміни у веб-сторінки, це робиться зміною кількох рядків у CSS-файлі.

Ви вже переконалися, що стилі — це чудово? Тоді спробуємо створити ваш перший стиль.

Завантажте ваш улюблений HTML-редактор і створіть у ньому веб-сторінку:

```
<HTML>
<HEAD>
<TITLE>Мій Перший Стиль </TITLE>
</HEAD>
<BODY>
<H1>Стилі: інструмент для сучасних веб-дизайнерів</H1>
<P>А Вашим конкурентам і не снилося!</P>
</BODY>
</HTML>
```

Тепер додамо стилі — просто впишіть цей код між тегами **<HTML>** й **<BODY>**:


```
<STYLE TYPE=«text/css»>
<!-
H2 { color: green;
  font-size: 37px;
  font-family: impact }
P { text-indent: 1cm;
 background: yellow;
 font-family: courier
 }
->
</STYLE>
```

СТИЛІ: ІНСТРУМЕНТ ДЛЯ СУЧАСНИХ ВЕБ-ДИЗАЙНЕРІВ

Отже, ви створили першу веб-сторінку зі стилями. *(Якщо рядок про конкурентів не має жовтих кольорів, вам доведеться оновити версію броузера. Рекомендуються Internet Explorer не нижче 5-ї версії.)*

Термінологія стилів

Розглянемо, з чого складається цей код.

Стилі задаються командами. Прості команди виглядають так:

```
H2 { color: green }
```

Ця команда повідомляє броузеру, що текст між **<H2>** й **</H2>** має бути зеленим.

У прикладі H2 — селектор, до якого застосовуються стилі. Опис визначає, який стиль використовується, й теж складається із двох частин. У нашому прикладі — властивість (кольори) і значення (зелений).

Все це може бути застосоване до будь-якого тегу: **<P>**, **<CODE>** або **<TABLE>**. Стилі також можна застосовувати з графікою, вміщуючи їх з тегом **<IMG>**.

Як видно з першого прикладу, ви можете групувати команди стилів. Вище ми встановили для тега **<P>** одразу три команди. Аналогічно ви можете групувати теги:

```
H2, P, BLOCKQUOTE { Font-family: arial }
```

Це значить, що весь текст у межах тегів **<H2>**, **<P>**, **<BLOCKQUOTE>** відобразиться шрифтом Arial.

Успадкування

Властивості стилів передаються у спадщину від «батька» (основного тега) до «нащадка» (тега, що знаходиться всередині основного).

Приклад:

```
B { color: blue }
```

Ця команда повідомляє броузеру, що весь текст у межах тегу <B> буде відображатися синім. Що ми бачимо?

```
<B>Даєш підтримку стилів на <I>всіх сторінках</I>
до 2006 року!</B>
```

Ця команда не встановлювалося для тегу <I>. Але тут він знаходиться в межах тегу <B> й успадковує поточну команду. Отже, «нащадок» відображається синім, як його «батько»:

Даєш підтримку стилів на *всіх сторінках* до 2006 року!

Тепер ви знаєте основні правила роботи стилів. Ви також вивчили один із способів додавання стилів до веб-сторінки. Але є й інші способи. Давайте їх розглянемо.

Додавання стилів до веб-сторінки

Ми розглянули один спосіб додавання стилів до веб-сторінки. Усього існує чотири способи, кожний має свої переваги:

- вкладання стилів в HTML-документ;
- посилання з HTML-документа на зовнішній документ стилів;
- імпортування документа стилів у HTML-документ;
- додавання стилів безпосередньо в рядки HTML-документа.

Вкладання стилів

Вся інформація, пов'язана зі стилями, знаходиться в шапці HTML-документа й не контактує з умістом тегу **<BODY>**.

Приклад:

```
<HTML>
<STYLE TYPE=«text/css»>
<!-
H1 { color: green; font-family: impact }
P { background: yellow; font-family: courier }
->
</STYLE>
```



```
<HEAD>
<TITLE>Мій Перший Стиль </TITLE>
</HEAD>
<BODY>
<H1>Стилі: інструмент для сучасних веб-дизай-
нерів</H1>
<P>А Вашим конкурентам і не снилося!</P>
</BODY>
</HTML>
```

Вкладені стилі виконуються тільки для поточного HTML-документа. Якщо ви хочете додати стилі тільки на єдину сторінку, цей шлях є найкращим. Ви, імовірно, звернули увагу на два цікавих моменти в цьому коді:

- атрибут і коментар тегу TYPE=«text/css». TYPE=«text/css» визначає тип MIME, щоб броузери, які не підтримуюють CSS, могли проігнорувати стилі;
- ще важливіші теги коментаря **<!- й ->**. Деякі старі броузери не визнають стилі, незважаючи на атрибут TYPE=«text/css», і відображають код текстом! З тегом коментаря цього не трапиться.

Посилання на документ стилів

Замість розміщення CSS-коду в кожній конкретній сторінці ви можете прив'язати безліч HTML-документів до одного документа стилів. Цей зовнішній CSS-файл установить правила для всіх ваших веб-сторінок. Приміром, якщо ви змінюєте розмір шрифту в документі стилів, шрифт негайно зміниться на всіх ваших сторінках. Це дуже зручно за підтримки великого сервера з великою кількістю файлів.

Ось як це виглядатиме: створіть звичайну веб-сторінку, але замість тегу <STYLE> використайте <LINK> у межах тегу <HEAD>:

```
<HTML>
<HEAD>
<TITLE> Мій Перший Стиль </TITLE>
<LINK REL=stylesheet HREF="mystyles.css"
TYPE="text/css">
</HEAD>
<BODY>
<H1>Стилі: інструмент для сучасних веб-дизайнерів</H1>
```

```
<P>А Вашим конкурентам і не снилося!</P>
</BODY>
</HTML>
```

Тепер створіть окремий текстовий файл, назвіть його mystyles.css (якщо хочете, назвіть його будь-яким іншим ім'ям). Він містить:

```
H1 { color: green;
     font-family: impact }
P { background: yellow;
    font-family: courier }
```

Ваш HTML-документ вказує серверу шлях до CSS-файла. Коли ви розглядаєте сторінку в броузері, ви побачите, як броузер відстежить тег <LINK> і завантажить настроювання сторінки з CSS-файла. Цей CSS-файл може застосовуватися в необмеженій кількості HTML-документів, що посилаються на нього. Ви можете використати відносні або абсолютні URL з атрибутом HREF.

ІМПОРТУВАННЯ СТИЛІВ

Аналогічно виглядає імпортування стилів. Відмінність у тому, що ви можете об'єднати імпортовані стилі зі своїми, що є всередині документа.

Приклад:

```
<HTML>
<STYLE TYPE=«text/css»>
<!—
@import url(company.css);
H1 { color: orange; font-family: impact }
—>
</STYLE>
<HEAD>
<TITLE> Мій Перший Стиль </TITLE>
</HEAD>
<BODY>
<H1>Стилі: інструмент для сучасних веб-дизайнерів</H1>
<P>А Вашим конкурентам і не снилося!</P>
</BODY>
</HTML>
```


У цьому випадку файл company.css виглядає так:

```
H1 { color: green; font-family: times }
P { background: yellow; font-family: courier }
```

У цьому прикладі броузер спочатку імпортує стилі з файла company.css (рядок @import завжди має бути першим), а потім додає до нього вкладені команди стилів. На сторінці застосовуються і ті, і інші стилі.

Проте уточнимо, що команда для H1 є і в зовнішніх, і у вкладених стилях. Що броузер робить у випадку такого конфлікту? Вкладені правила головують, і текст відображається жовтогарячим стилі мають велику гнучкість. Ви можете імпортувати безліч CSS-файлів і за потреби змінювати їхні властивості за допомогою вкладених стилів на ваш розсуд.

Додавання стилів у рядки веб-сторінки

Ви можете додавати стилі безпосередньо в рядки будь-якого HTML-документа.

Приклад:

```
<HTML>
<HEAD>
<TITLE> Мій Перший Стиль </TITLE>
</HEAD>
<BODY>
<H1 STYLE="color: orange; font-family: impact">
Стилі: інструмент для просунутих веб-дизайнерів
</P></H1>
<P STYLE="background:yellow;font-family:courier">
А Вашим конкурентам і не снилося!</P>
</BODY>
</HTML>
```

У цьому прикладі не потрібно додавати стилі в заголовок HTML-документа. Вбудовані атрибути стилів нададуть броузеру всю необхідну інформацію.Недолік цього методу в тому, що ви повинні додавати вбудований код стилів щоразу, коли ви захочете їх використати. Кожен наступний тег <H1> відображається за замовчуванням, якщо ви не привласните йому команду стилів.

Вбудовані стилі менш зручні, ніж імпортовані або вставлені за допомогою посилання, але ви можете знайти застосування і для них.

Запам'ятайте: ви можете використати один і більше методів у кожному конкретному випадку. Фактично сила стилів — в об'єднанні всіх цих способів.

У CSS є ще кілька хитрощів. Щоб дізнатися їх, ми поговоримо про синтаксис стилів.

Класи й інші хитрощі

Ви вже ознайомилися з основами синтаксису стилів. Тепер іще кілька хитрощів.

Класи

Вище було сказано, що команди стилів можна використати з будь-яким конкретним тегом. Але що робити, якщо потрібно щось більше? Наприклад, одному параграфу вказати зелені кольори, другому — пурпурний і сірий — третьому?

У цьому допоможуть класи. Ви створите три різних класи для параграфів з різними командами стилів у кожному. Стилі (або вкладені, або ті, що знаходяться у зовнішньому CSS-файлі) виглядатимуть приблизно так:

```
P.first { color: green }
P.second { color: purple }
P.third { color: gray }
```

Ваш код HTML має виглядати так:

```
<P CLASS=first>Це перший параграф, ім'я класу
"first."</P>
<P CLASS=second>Це другий параграф, ім'я класу
"second."</P>
<P CLASS=third>Це третій параграф, ім'я класу
"third."</P>
```

Ви також можете створити класи, які не будуть додаватися до конкретного HTML-тегу:

```
.first { color: green }
```

Цей метод — більш гнучкий, оскільки тепер ми можемо використати **CLASS=first** з будь-яким HTML-тегом у межах тега <BODY>, і текст буде відображатися зеленим.

Контекстні селектори

Наприклад, ви хочете, щоб «жирний» текст відображався червоним, але тільки в тому випадку, якщо цей текст знаходиться в певному місці документа.

Контекстні селектори виконуються тільки в конкретній ситуації, що вказана в їхніх властивостях.

```
P B { color: red }
<H1><B>Олексій Александров</B>, програміст</H1>
<P>Знає HTML, Java, Javascript й CSS.
Чи є <B>щось,</B>що йому не під силу?</P>
```

Команда стилів повідомляє броузеру, щоб «жирний» текст відображався червоним, тільки якщо він знаходиться в межах тегу **<P>**. Таким чином, коли наведений вище HTML-код відображається броузером, колір «жирного» тексту в першому рядку — той, що встановлений за замовчуванням, а в другому рядку — червоний.

Коментарі

Навіть незважаючи на те, що структура стилів проста й зрозуміла, коментарі вам не зашкодять. Але коментарі можуть бути використані в будь-якому рядку коду CSS:

```
/* зелений для параграфа зі стилем first*/
P.first { color: green }
H1 { text-indent: 10px; font-family: verdana }
/* зрушувати всі картинки на 100px зверху */
IMG { margin-top: 100px }
```

Тепер виникає запитання: чому ця технологія називається «каскадні стилі»?

Каскадування: конфлікт стилів

Причина конфлікту: в одному HTML-документі у роботі є три різних CSS, і всі вони використовують тег <P> як селектор. Імпортований CSS вказує броузеру: відобразити текст у межах тегу <P> червоним. Вкладений CSS вказує броузеру — використати синій. А вбудований CSS вказує використати жовтий. Що ж у такому випадку робити броузеру?

Броузери, що підтримують стилі, мають вбудований порядок каскадування команд, якими й керуються в таких ситуаціях. Зрештою, деякі команди стилів для нас важливіші, ніж інші.

Порядок «важливості», відповідно до офіційної специфікації каскадування стилів:

- вбудовані стилі;
- вкладені стилі;
- посилання на зовнішній документ стилів;
- імпортовані стилі;
- стилі броузера за замовчуванням.

Так, вбудовані стилі скасовують ефекти вкладених стилів, які, у свою чергу, скасовують ефект стилів, доданих до документа за допомогою посилання.

Дуже зручно, чи не так? Але не все так просто, як хотілося б. У броузерах від Netscape й Microsoft є відмінності у виконанні цього порядку. Наприклад, обидва броузери приділяють більше уваги стилям, на які проставлені посилання, ніж вкладеним стилям. Тому поки що краще застосовувати на сторінці тільки один метод додавання стилів, особливо коли ви знаєте особливості основних броузерів.

Можуть виникнути й інші проблеми. Що трапляється, коли численні правила стилів одного типу суперечать одні одним? Що трапляється, якщо один вкладений стиль повідомляє текст у межах тегу **<P>** зеленим, а інший вкладений стиль повідомляє його червоним?

Завдяки специфікації стилів існує порядок вирішення цих конфліктів. Дотримуйтеся стилю.

Приклад:

```
BODY {color:green} P {color:red}
```

Текст у тегу **<P>** оголошений червоним, але він також успадковує вказівки з тегу **<BODY>**, що текст має бути зеленим. (*Якщо ви вказуєте значення стилів у тегу* **<BODY>**, *то воно діє на всій сторінці.*) У цьому випадку конкретний покажчик тегу **<P>** переважає і текст показується червоним.

Команди стилів застосовуються відповідно до порядку їхнього знаходження в HTML-документі. Броузер «слухає» завжди останню команду стилів.

Приклад:

```
P {color:green} P {color:red}
```

Якщо є конфлікт, броузер застосовує команди в порядку появи. У наведеному прикладі текст у межах тегу <P> має дотримуватися останньої команди й відображатися червоним.

Останнє запитання: що трапляється, коли стилі вступають у протиріччя з HTML-тегами?

Приклад:

```
I { font-family: impact }
<P>I think <I><FONT FACE="Times New Roman">HTML
i CSS</FONT></I> — брати навік!</P>
```

Команди стилів вказують броузеру, що потрібно використати гарнітуру Impact, але HTML-тег **<Font Face>** потребує, щоб зображувався Times new roman. Існує конфлікт. Відповідно до офіційної специфікації стилів, стилі переважають і вказується той шрифт, що зазначений у них. Якщо ж вкладених CSS-команд немає, броузер використає HTML-теги за замовчуванням.

Основні броузери, на жаль, не настроєні на це. Вони звертаються до HTML-тегів як до важливіших.

розділ 6

РОБОТА ІЗ САЙТОМ

ЯК ОРГАНІЗУВАТИ ПОШУК НА САЙТІ

Якщо ви не знаєте навіть основ HTML – зверніться до професіоналів. Проте, якщо ви уявляєте собі хоча б основні правила створення html-сторінки, можете впоратися самі.

Деякі пошукові системи надають сервіси з організації пошуку на вашому сайті.

Приклад: найкраща українська пошукова система «META». (http://www.meta.ua)

На першій сторінці ви знайдете розділ «*Інформація*». У рубриці «*Власникам сайту*» є поради й корисні сервіси для тих, хто хоче вдосконалити свій сайт (рис. 6.1).

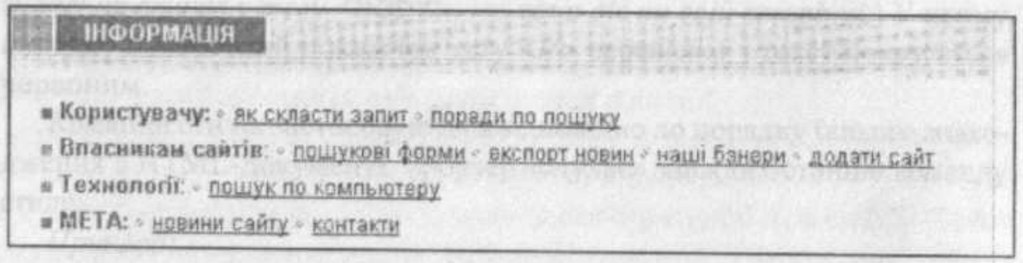


Рис. 6.1

Натисніть посилання «*пошукові форми*». З'явиться сторінка з докладним описом кількох варіантів створення пошуку на сайті за допомогою пошукової системи «META» (рис. 6.2).

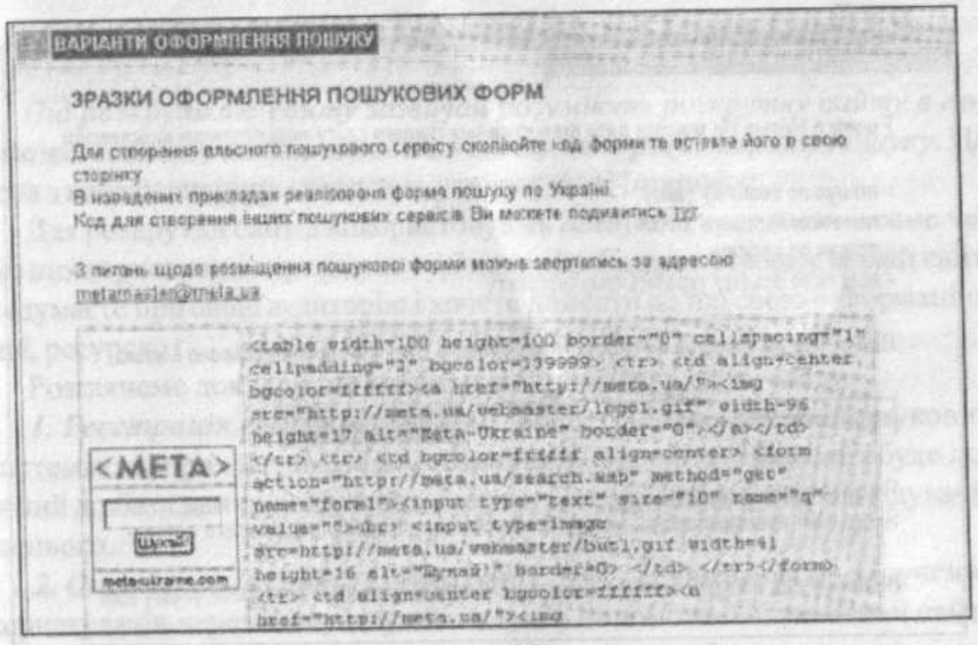


Рис. 6.2

Звертаємо вашу увагу, що на цій сторінці наведено також кілька зразків оформлення пошуку на сайті. Далі потрібно:

- уважно прочитати інструкцію;
- вибрати із запропонованих зразок пошукової форми, який вам найбільше сподобався.
- відкрити, наприклад, NotePad або інший засіб для створення .html-сторінок (наприклад, будь-який редактор) і додати коди, які надані на сторінці у зручне для вас місце сторінки.

Якщо у вас виникли проблеми, надішліть листа веб-майстру на адресу, вказану на цій сторінці пошукової системи «META».

«META» також надає додаткові можливості користувачам – пошукові форми, які можна знайти на відповідній сторінці (рис. 6.3 на с. 108).

Знайдіть на цій сторінці розділ, присвячений вибору пошуку:

- тематичний пошук;
- регіональний пошук;
- пошук по вашому сайту;
- пошук по всьому українському Інтернету.
- Натиснувши кнопку «Отримати код форми», ви побачите нове вікно, в якому буде розміщена додаткова інформація, і вид форми для пошуку (рис. 6.4 на с. 108).

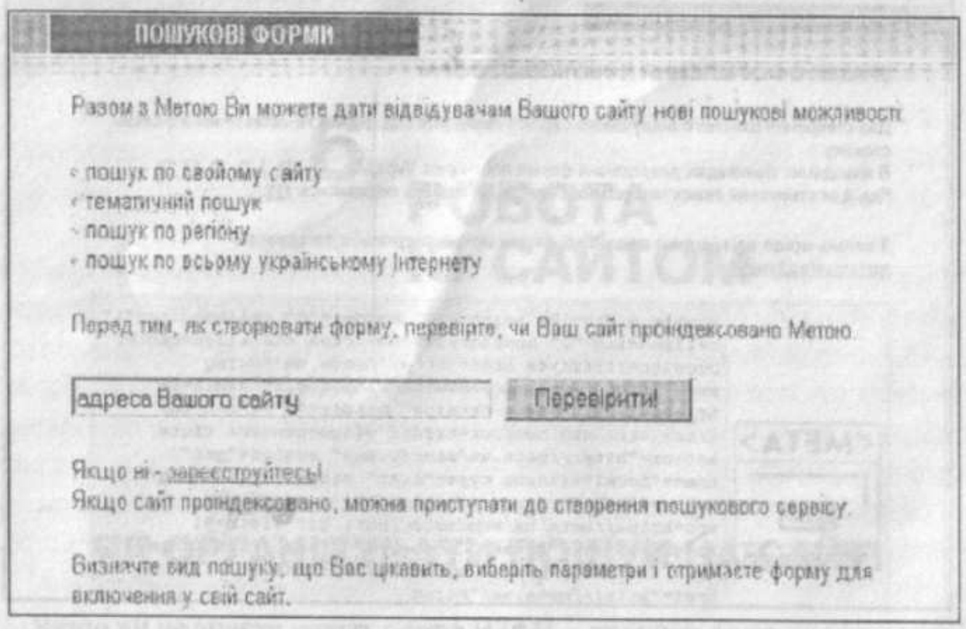


Рис. 6.3

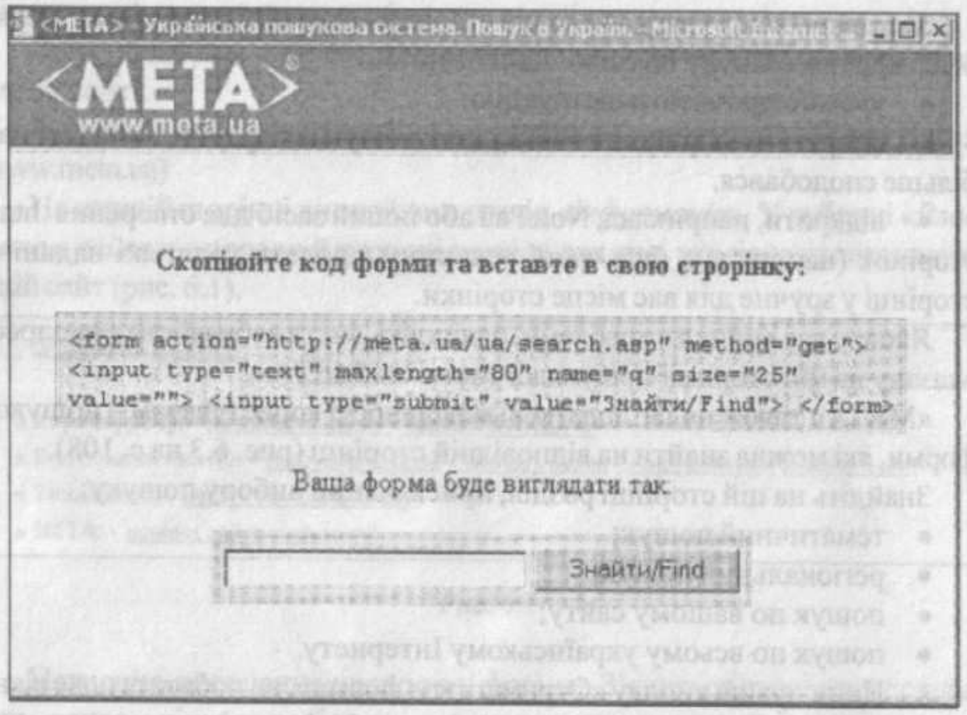


Рис. 6.4

ЯК ОРГАНІЗУВАТИ «РОЗКРУТКУ» САЙТУ

*Під **розкруткою сайту** зазвичай розуміють розкрутку сайту в пошуковій системі, тобто виведення сайту на першу сторінку пошуку.* Це одна з найважливіших складових успіху у світі Інтернету.

Для розкрутки сайтів використовують ***пошукові системи*** — саме через них користувачі виражають свої потреби. Коли ви створюєте свій сайт, то думаєте про свою аудиторію і хочете донести до неї свою інформацію, ідеї, ресурси.

Розглянемо докладніше окремі питання «розкрутки» сайту.

1. Реєстрація сайту. Необхідно зареєструвати сайт на пошукових системах: наприклад «META», «Rambler», «Google». Ваш сайт буде доданий до бази даних пошукових систем, що спростить вихід відвідувачів на нього.

2. Оптимізація сайту. Оптимізація сайту спрямована на залучення користувачів через пошукову систему. За допомогою оптимізації сайту пошукова машина бачитиме ваш сайт і ставитиме його на перші місця. Статистика свідчить, що понад 80 % відвідувачів сайтів приходять з пошукових систем (Rambler, Yandex, Aport тощо). До інших 20 % належать каталоги, тематичні сайти, рейтинги, банерна реклама.

Мета оптимізації сайту — покращення рейтингу вашого сайту, бажання бачити його на першій сторінці пошуку, хоча активні користувачі переглядають понад 10 сторінок результатів пошуку. Оптимізація сайту — це робота з кодом і текстом web-сторінок для використання сукупності внутрішніх факторів, що впливають на результат видачі за пошуковим запитом.

ОПТИМІЗАЦІЯ САЙТУ

Оптимізація сайту починається із з'ясування вашої потенційної аудиторії і термінології. Наприклад, якщо ваш сайт присвячений діяльності школи і ви пишаєтеся своїми методичними розробками, спробуйте щонайкраще їх описати не тільки в текстах, а й у заголовках документів через перелік ключових слів у метатегах заголовка HTML-документа. Вибір правильних ключових слів і висловів — один з найважливіших кроків.

Відповідно до власного пошукового алгоритму, пошукові машини формують семантичну мапу ключових слів із запитів, що буде викори-

стано в тексті сторінок сайта. Іншими словами, потрібно уявити собі, що і якими словами будуть користувачі шукатимуть, і відобразити це у тексті.

ОПТИМІЗАЦІЯ КОНТЕНТУ САЙТУ

***Контент сайту** — це його зміст: графіка і тексти.* Для того, щоб пошукові системи правильно сприймали наповнення сайту, необхідно його «підігнати» під них.

Види оптимізації сайту:

***«Біла оптимізація»** — комплексна розкрутка сайту зі складанням семантичного ядра, необхідного для успішного пошуку*, оптимізація окремих сторінок сайту під ці запити шляхом виправлення контенту (змісту), розкрутка сайту реєстрацією в численних каталогах і пошукових системах, а також обмін посиланнями з сайтами схожої тематики з використанням ключових словосполучень, на яке усе більше орієнтуються пошукові системи при розробці алгоритмів підрахунку релевантності сайтів.

***«Сіра оптимізація»** — процес хаотичного обміну посиланнями, швидший, але малоефективний.* До сірих методів належить активна діяльність із обміну посиланнями, розміщення платних посилань, невиправдане використання тегів заголовків, використання на сторінках видимого, але майже непомітного тексту тощо. До «сірих» методів просування сайту належить створення спеціальних вхідних сторінок (дорвеїв) без редиректу і нетематична робота з посиланнями. Відмінність дорвею без редиректу (переадресації посилання) від «чорних» дорвеїв у тому, що вони містять хоч і оптимізований, але цілком придатний для читання текст, а замість редиректу — явне посилання, що запрошує відвідувача перейти вже на основну сторінку.

Намагайтеся реєструвати свій сайт у каталогах. Нетематична робота з посиланнями також важлива.

***«Чорна» оптимізація** — так званий пошуковий спам.* До «чорних» методів належать: невидимий текст (або поданий дуже дрібним шрифтом, що майже не читається, або захований у невидимий шар), дорвеї з редиректом (це сторінка на іншому домені, вкрай насичена ключовими словами, найчастіше зовсім без зв'язку) і клоакінг (методи, що дають змогу видавати різним користувачам різні сторінки за однаковою адресою).

ОПТИМІЗАЦІЯ ДЛЯ ПОШУКОВОЇ СИСТЕМИ YANDEX

При написанні сторінок намагайтеся вкладати ключові слова в теги, використовувати ключові слова в тегах заголовків. META-теги не грають особливої ролі при визначенні позиції сайту, але опис сторінки в результатах пошуку може містити слова з Description або Comment.

ОПТИМІЗАЦІЯ ДЛЯ ПОШУКОВОЇ СИСТЕМИ RAMBLER

Особливості:

- для **Rambler** важлива участь у його рейтингах.
- Rambler зовсім не розглядає META-теги.
- важливі теги заголовків тощо.

Про особливості пошуку на цьому пошуковому сервері можна дізнатися, звернувшись до спеціальних рубрик — «Пошук» на Rambler (рис. 6.5).

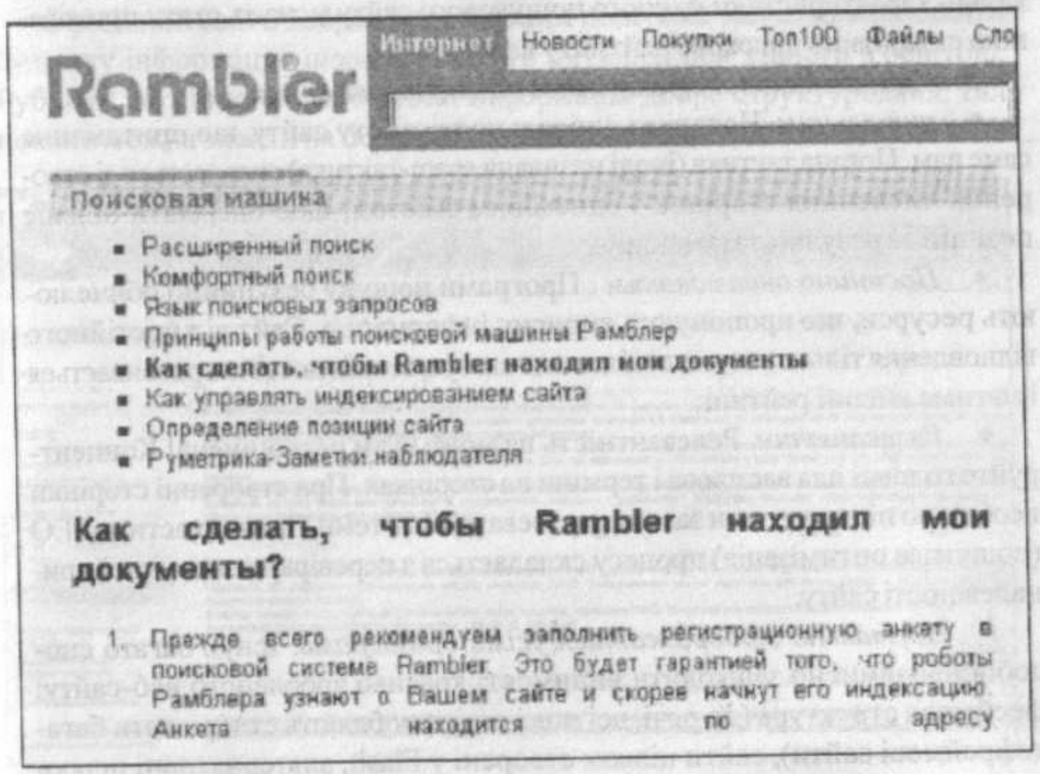


Рис. 6.5

ТЕХНОЛОГІЯ ПОШУКУ GOOGLE

Розробники пошукової системи Google стверджують, що для ранжування сторінок слід враховувати:

- індекс цитування (Link popularity);
- щільність і частоту ключових слів (Keyword proximity and density);
- ключові слова в посиланнях (Keywords in the link text);
- виділений текст (Emphasized text).

Google видає такі результати: посилання на сторінку, опис, складений зі слів, що оточують пошуковий запит. Оскільки опис сторінки залежить від конкретного пошукового запиту, потрібно орієнтуватися на потреби тих, хто користуватиметься вашим сайтом.

ЗМІСТ САЙТУ

Продуманий і грамотний зміст — головний фактор успіху вашого сайта, але що означає добрий зміст із погляду пошукових алгоритмів? Унікальні характеристики кожного пошукового сайту можуть стати прихованою складовою високого рейтингу сайту.

Для максимальної ефективності зміст має бути:

- *унікальним*. Наявність унікальності змісту сайту, що притаманна саме вам. Погана тактика (іноді називана spam-тактика) складається зі створення численних сторінок з однаковим змістом, щоб одержати більше позицій за результататам пошуку.
- *Постійно оновлюваним* . Програми пошуку безупинно обчислюють ресурси, що пропонують корисну інформацію. Сайт від постійного відновлення тільки виграє, оскільки це вказує, що він постійно розвивається і матиме вищий рейтинг.
- *Релевантним*. Релевантність не може бути переоцінена! Концентруйте головні для вас слова і терміни на сторінках. При створенні сторінки необхідно підтримувати загальну релевантність теми. Значна частина SEO (пошукова оптимізація) процесу складається з перевірки тематичної приналежності сайту.
- *Нормально відображатися усіма броузерами.* Існує багато способів ненавмисно зашкодити видимості красиво зробленого веб-сайту: фреймова структура (до речі, всі новачки полюбляють створювати багатофреймові сайти), сайти цілком створені у Flash, альтернативні шляхи навігації (прописуючи атрибут ALT тега Image кожної кнопки меню тема-

тичним описом сторінки призначення, ви робите меню видимим для пошукових програм. Є, щоправда, кілька альтернативних навігаційних методів, а саме: текстове меню внизу сторінки і текстові посилання на карту сайту, розташовані десь посередині кожної сторінки).

Карта сайту корисна через добре відому особливість пошукових програм; вони часто витрачають багато часу, щоб проіндексувати сторінки, розташовані глибоко всередині сайта. Так, створюючи карту сайту (переважно з текстовими посиланнями), доступну з будь-якої сторінки, ви забезпечуєте пошуковий механізм прямим маршрутом на кожну сторінку в межах вашого сайту. Це не тільки полегшує огляд вашого сайту, а й надає текстові посилання , що можуть поліпшити позицію вашого ресурсу.

КОРИСНІ ПОСИЛАННЯ ДЛЯ ВИКЛАДАЧІВ ШКІЛ

1. Міністерство освіти і науки України
http://www.mon.gov.ua

Офіційний сайт з добре організованим пошуком. Легко можна знайти будь-яку інформацію щодо будь-якого питання, пов'язаного з освітою. Рубрики постійно оновлюються. Інформація добре структурована. Всі новини можна знайти на першій сторінці (рис. 6.6).

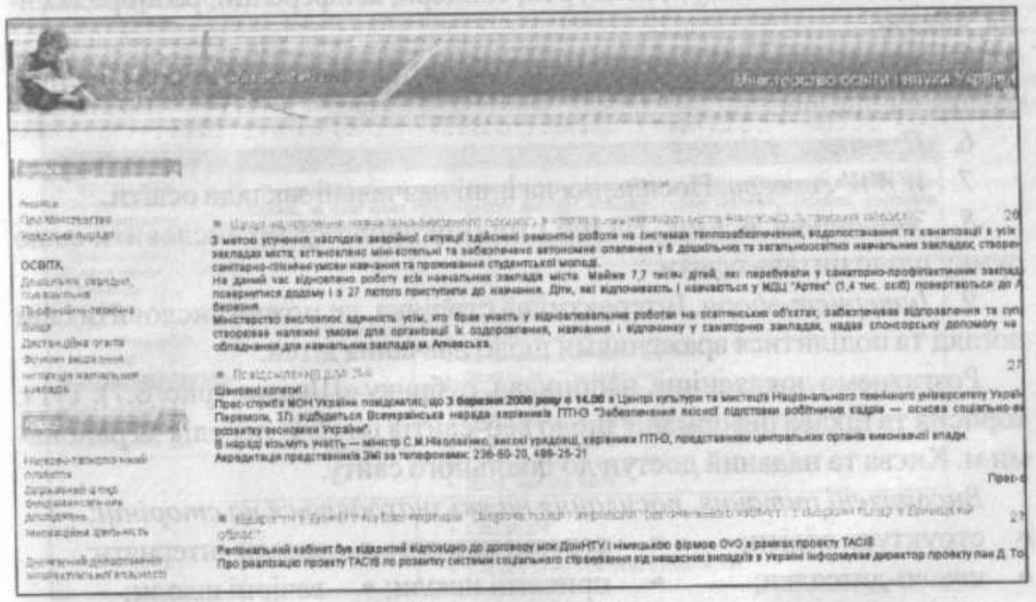


Рис. 6.6

2. Сайт міністерства освіти і науки України
http://education.gov.ua/pls/edu/educ.home.ukr

Системі освіти в Україні наданий загальний опис, зокрема, дошкільній, загальній середній, позашкільній професійно-технічній, вищій післядипломній педагогічній освіті в Україні.

Наданий доступ до бази даних законодавства з питань освіти. Працює пошук: за словами в назві та тексті документа, автором, датами, видами та статусами документів. Можна переглядати документи за рубриками та оглянути повний перелік документів освітнього законодавства в базі даних. Реалізований сервіс побудови діаграм за типами документів, рубриками та кількістю документів за напрямами.

3. Головне управління освіти і науки
http://www.edu.kiev.ua
Основні рубрики:

1. *Про сервер.* Цей сервер створено як Web-портал Головного управління освіти і науки...
2. *Концепції.* Надана концепція інформатизації шкіл м. Києва; інформація про проект «Internet — кожній київській школі» та закон про інформатизацію...
3. *Служби* ГУОН.
4. *Новини.* Семінари, конкурси, тендери, конференції, розпорядження, накази ...
5. *Школи.* Шкільні веб-сторінки; школи, підключені до мережі Internet; заявка на Internet.
6. *Преса.*
7. *WWW-сервери.* Посилання на інші навчальні заклади освіти.
8. *Зворотний зв'язок.* На цій сторінці пропонуємо висловити свою думку щодо питань освіти
9. *Інтернет-збори.* Інтерактивна сторінка, де можна висловити свій погляд та поділитися враженнями щодо навчання дітей.

Розглянемо докладніше, наприклад, рубрику «Школи» (рис. 6.7). Тут є корисна та цікава інформація щодо веб-сайтів шкіл. Є перелік за районами м. Києва та наданий доступ до шкільного сайту.

Висвітлені питання, посилання на які знаходяться на сторінці:

- структура освіти;
- середні школи;
- школи-інтернати;
- школи-дитсадки;
- приватні школи;
- вечірні школи;
- спецшколи;
- гімназії, ліцеї;
- школа Монтессорі.

Інформація надана українською та англійською мовами.

Рис. 6.7

Кожна школа може створити та підтримувати свої веб-сторінки, на яких може бути наведено дані про методики, окремі навчальні предмети, результати творчості викладачів та учнів тощо. Зміст сайту визначає сама школа (рис. 6.8).

Рис. 6.8

На цій сторінці користувачі можуть знайти допомогу, а саме:

- заявка на Internet (http://www.edu.kiev.ua/schools/zayava/zayava.htm);
- розміщення сайту;
- веб-сторінки шкіл.

Що пропонує інформаційний центр «Електронні вісті» в разі підключення закладу освіти до мережі Internet — дивись проект «Internet — кожній київській школі» (рис. 6.9).

ПРОЕКТ "INTERNET - КОЖНІЙ КИЇВСЬКІЙ ШКОЛІ"

ElVisti

ІНФОРМАЦІЙНИЙ ЦЕНТР "ЕЛЕКТРОННІ ВІСТІ" http://www.visti.net

Глобальна мережа Internet з кожним днем відіграє все більшу роль у житті суспільства. Безумовно, це стосується і такої сфери людської діяльності як освіта.

Визнаючи це, Головне управління освіти Київської міської держадміністрації разом з Інформаційним центром "Електронні вісті" і компанією "Інформаційні комп'ютерні системи" (ICS) виступили ініціаторами проекту надання послуг мережі Internet школам міста Києва.

Мета проекту - забезпечення доступу для викладачів та школярів до сучасних інформаційних технологій і інформаційних ресурсів мережі Internet, формування передумов для розвитку самосвідомості і самостійного навчання.

Проект було започатковано у 1996 році. Тоді ж за сприянням Головного управління освіти Київської міської держадміністрації Інформаційний центр "Електронні вісті" зареєстрував адресну зону edu.kiev.ua і надав доступ до Internet Головному управлінню освіти і районним відділам освіти м. Києва.

Подальшим розвитком проекту явилися: безкоштовна реєстрація київських шкіл в адресній зоні edu.kiev.ua, підключення їх до вузла ElVisti, надання можливості роботи в Мережі школам м. Києва. На сьогодні загальне число шкіл, що працюють у рамках проекту - 79. Проводиться підготовча роботи з підключення до мережі Internet ще 155 шкіл. У цілому зареєстровані в проекті школи відпрацювали в мережі Internet у 1999 році 3445 часів, а в 2000 році - 8073 часів. Крім цього, Інформаційним центром "Електронні вісті" було виконане підключення по виділених каналах зв'язку Головного управління освіти Київської міської держадміністрації і школи № 79 з поглибленим вивченням інформаційних технологій. Цього року до мережі Internet був залучений Київський Міжрегіональний інститут удосконалення вчителів ім. Бориса Грінченка. На ресурсах Інформаційного центру "Електронні вісті" побудований Web-сервер Головного управління освіти Київської міської держадміністрації та шкіл м. Києва (http://www.edu.kiev.ua), на якому подана інформація про стан і перспективи розвитку освіти. Одночасно даний сервер використовується для розміщення власних Web-сторінок шкіл міста Києва.

Керуючись Указом Президента України N 928/2000 від 31 липня 2000 року "Про заходи щодо розвитку національної складової глобальної інформаційної мережі Internet і забезпеченню широкого доступу до цієї

Рис. 6.9

У рубриці «Новини», яка постійно оновлюється, можна знайти інформацію щодо нарад, засідань, всеукраїнських конкурсів тощо (рис. 6.10).

ГОЛОВНЕ УПРАВЛІННЯ ОСВІТИ І НАУКИ КИЇВСЬКОЇ МІСЬКОЇ ДЕРЖАДМІНІСТРАЦІЇ

MAIN EDUCATION AND SCIENCE BOARD OF KYIV CITY STATE ADMINISTRATION

Дні відкритих дверей

Шановні випускники!

Бажаючих оволодіти магією сучасних інформаційних, комп'ютерних, математичних та програмних технологій, системною аналітикою складних систем (банківських, економічних, технічних і т.п.) запрошуємо до НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ» на дні відкритих дверей факультету прикладної математики, що відбудуться ***28 березня і 11 квітня 2006 р. о 15. 00.***

Порядок денний включає в себе зустріч з деканом та завідувачами кафедр, зі студентами факультету, а також ознайомлення з навчально-лабораторною базою факультету.

Адреса: *м. Київ, вул. Політехнічна, навчальний корпус № 19, аудиторія 345.*

Прес-служба ГУОН

Рис. 6.10

4. Освітянська Мережа України

http://www.ednu.kiev.ua/index_u.htm (рис. 6.11)

Educational Network Ukraine

EdNU

All about education in Ukraine - schools, colleges, universities, educational organizations, study abroad programs, foreign language study, exhibitions, educational press and much more

English Ukrainian

Рис. 6.11

Інформація надана двома мовами: українською та англійською.

Україномовна інформація доступна при виборі мови на першій сторінці (рис. 6.12).

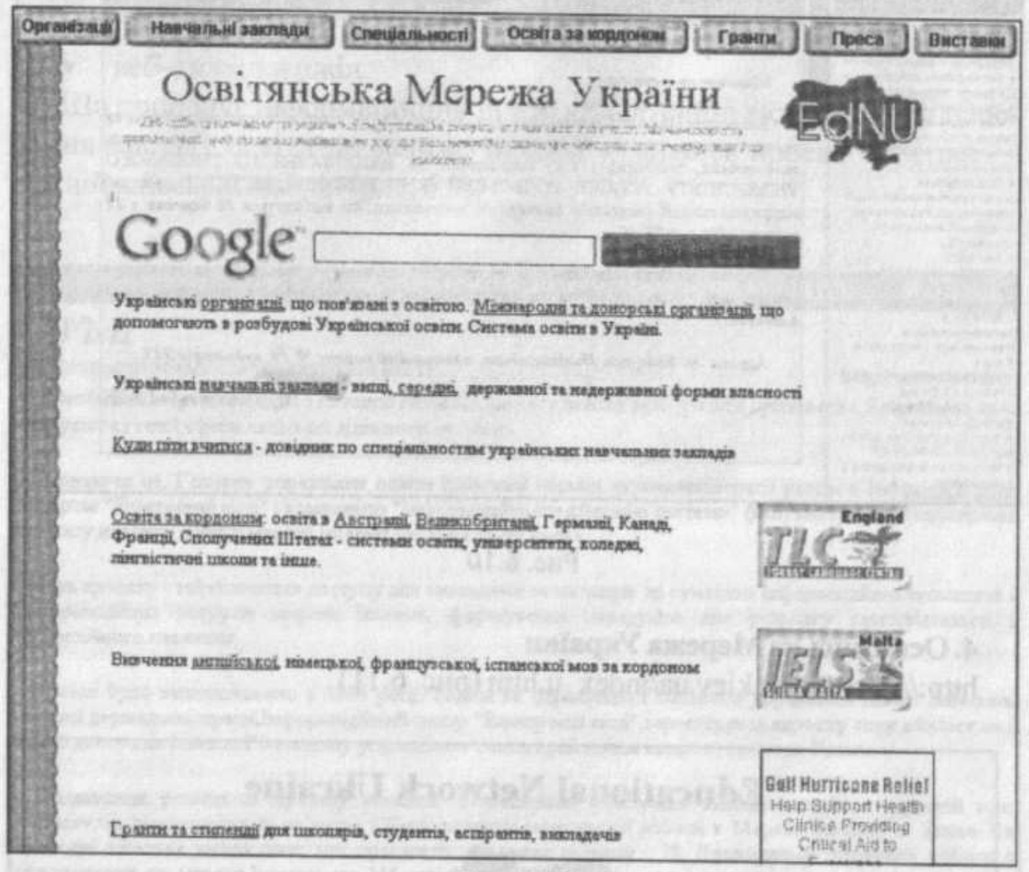


Рис. 6.12

Кожний навчальний заклад може знайти себе у каталогах, якщо його сторінка там зареєстрована. В іншому випадку можна зареєструватися, і посилання на ваш навчальний заклад буде доступне не лише українській спільноті, а й зарубіжним колегам (рис. 6.13).

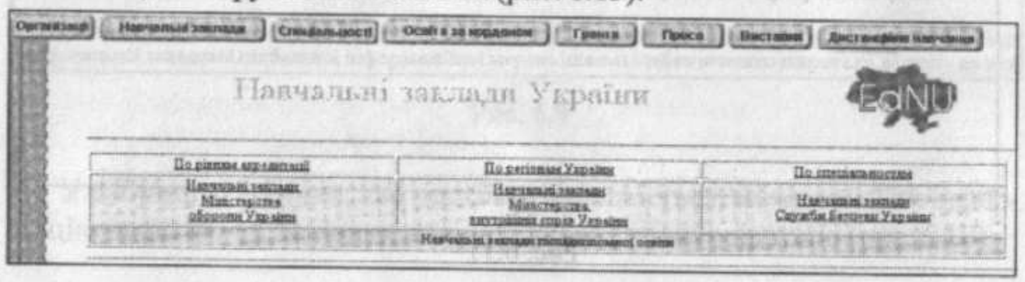


Рис. 6.13

5. Освітній портал

http://www.osvita.org.ua/

Багато цікавих рубрик та посилань. Рекомендуємо обов'язково відвідати рубрику «Освітні ресурси»).

6. Он-лайнові перекладачі

http://www.translate.ru/text.asp#tr_form.

Перекладає з російської на англійську, німецьку, французьку, італійську, іспанську мови. Якість перекладу невисока. Однак, якщо потрібно написати листа особі, яка розмовляє, наприклад французькою мовою, а ви нею зовсім не володієте, то, якщо свою думку викладати простими реченнями — переклад буде таким, що адресат вас зрозуміє.

7. Он-лайновий перекладач для української та російської мови

http://www.uaportal.com

Перекладає в он-лайні з російської на англійську, німецьку, французьку, італійську, іспанську мови.

8. Словники України

http://www.ulif.org.ua

9. Он-лайновий перекладач для української та російської мови

http://www.uaportal.com

Перекладач можна знайти у рубриці «Переклад».

Науково-виробниче видання

Бібліотека «Шкільного світу»

Манако Володимир, **Манако** Дмитро,
Данилова Ольга, **Войченко** Олексій

Науковий консультант — **Манако** Алла Федорівна

Основи будування сайтів

Редактор *Н.Вовковінська*
Художній редактор *Т.Мосієнко*
Літературний редактор *Ю.Желєзна*
Коректор *О.Ліннік*
Комп'ютерна верстка *К.Яскевич*

Підписано до друку 19.04.06. Формат 60x84/16.
Папір офсетний № 1. Гарнітура Таймс. Друк офсетний.
Умовн. друк. арк. 7,44. Обл.-вид. арк. 7,2. Тираж 3000 пр.
Зам. № 68.

Видавничий дім «Шкільний світ»
01014, м.Київ, вул. Тимірязєвська, 2
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції
серія ДК № 623 від 04.10.2001 р.

ФО-П Галіцина Л.В.
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції
серія ДК № 1181 від 27.12.2002 р.

Видрукувано з готових діапозитивів в ОП «Житомирська облдрукарня»
10014, Житомир, вул. Мала Бердичівська, 17
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції
серія ЖТ № 1 від 06.04.2001 р.